COMPUTER NETWORKS LABORATORY

Sub Code : 18CSL57 | IA Marks : 20

Hrs / Week : 01I + 02P | Exam Hours : 03

Total Hrs : 40 | Exam Marks : 80

PART A - Simulation Exercises

For the experiments below modify the topology and parameters set for the experiment and take multiple rounds of reading and analyze the results available in log files. Plot necessary graphs and conclude. Use NS2/NS3.

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:

1 Write a program for error detecting code using CRC-CCITT (16- bits).
2 Write a program to find the shortest path between vertices using bellman-ford algorithm.
3 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.
4 Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.
5 Write a program for simple RSA algorithm to encrypt and decrypt the data.
6 Write a program for congestion control using leaky bucket algorithm.

Conduction of Practical Examination:

1. All laboratory experiments are to be included for practical examination.
2. Students are allowed to pick one experiment from part A and part B with lot.
3. Strictly follow the instructions as printed on the cover page of answer script
4. Marks distribution: Procedure + Conduction + Viva: 80 Part A: 10+25+5 =40 Part B: 10+25+5 =40
5. Change of experiment is allowed only once and marks allotted to the procedure part to be made zero.

Course Outcome

At the end of the course students will be able to
CO1- Analyze and Compare various networking protocols.

CO2- Demonstrate the working of different concepts of networking.

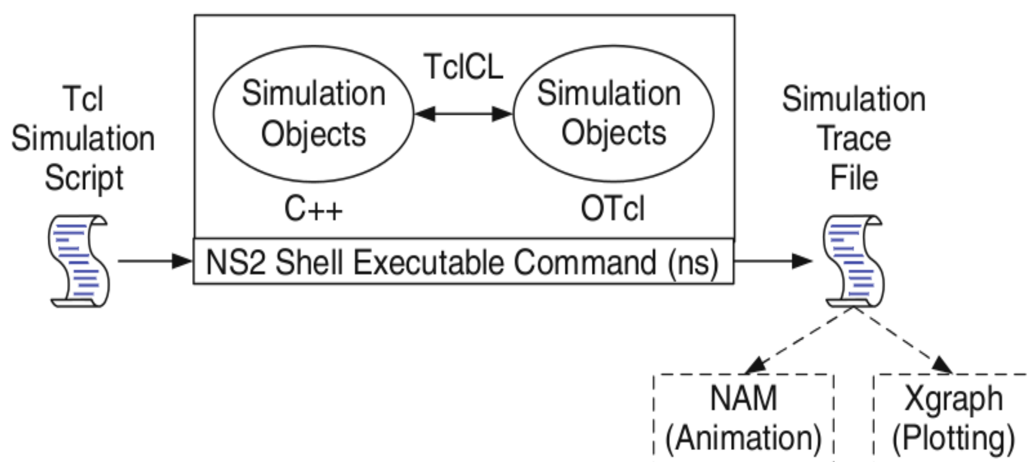
CO3 - Implement, analyze and evaluate networking protocols in NS2 / NS3

PART A Simulation

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 becauseit 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. Itworks 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

PART - A

Experiment 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.

Step1: Open text editor, type the below program and save with extention .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
```

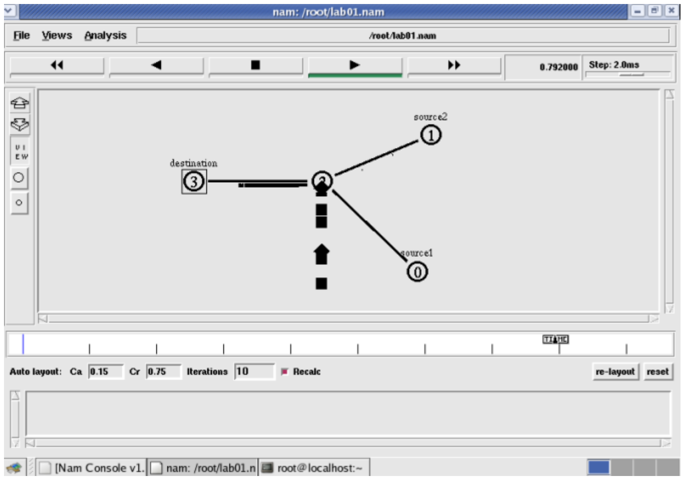
Step2: Open text editor, type the below program and save with extention .awk (**prog1.awk**)

```
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

[root@localhost~]# ns prog1.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.

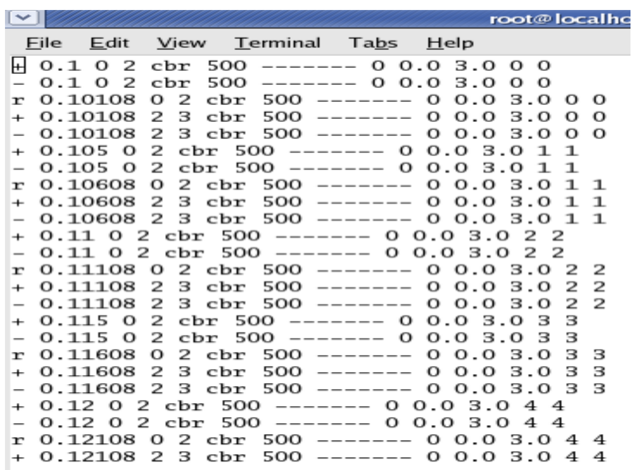
Step 5: After simulation is completed run **awk file** to see the output ,

[root@localhost~]# awk –f 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



```
File Edit View Terminal Tabs Help
+ 0.1 0 2 cbr 500 ------- 0 0.0 3.0 0 0
- 0.1 0 2 cbr 500 ------- 0 0.0 3.0 0 0
r 0.10108 0 2 cbr 500 ------- 0 0.0 3.0 0 0
+ 0.10108 2 3 cbr 500 ------- 0 0.0 3.0 0 0
- 0.10108 2 3 cbr 500 ------- 0 0.0 3.0 0 0
+ 0.105 0 2 cbr 500 ------- 0 0.0 3.0 1 1
- 0.105 0 2 cbr 500 ------- 0 0.0 3.0 1 1
r 0.10608 0 2 cbr 500 ------- 0 0.0 3.0 1 1
+ 0.10608 2 3 cbr 500 ------- 0 0.0 3.0 1 1
- 0.10608 2 3 cbr 500 ------- 0 0.0 3.0 1 1
+ 0.11 0 2 cbr 500 ------- 0 0.0 3.0 2 2
- 0.11 0 2 cbr 500 ------- 0 0.0 3.0 2 2
r 0.11108 0 2 cbr 500 ------- 0 0.0 3.0 2 2
+ 0.11108 2 3 cbr 500 ------- 0 0.0 3.0 2 2
- 0.11108 2 3 cbr 500 ------- 0 0.0 3.0 2 2
+ 0.115 0 2 cbr 500 ------- 0 0.0 3.0 3 3
- 0.115 0 2 cbr 500 ------- 0 0.0 3.0 3 3
r 0.11608 0 2 cbr 500 ------- 0 0.0 3.0 3 3
+ 0.11608 2 3 cbr 500 ------- 0 0.0 3.0 3 3
- 0.11608 2 3 cbr 500 ------- 0 0.0 3.0 3 3
+ 0.12 0 2 cbr 500 ------- 0 0.0 3.0 4 4
- 0.12 0 2 cbr 500 ------- 0 0.0 3.0 4 4
r 0.12108 0 2 cbr 500 ------- 0 0.0 3.0 4 4
+ 0.12108 2 3 cbr 500 ------- 0 0.0 3.0 4 4
```

Experiment 2:

Implement transmission of ping messages/trace route over a network topologyconsisting of 6 nodes and find the number of packets dropped due to congestion.

Step1: Open text editor, type the below program and save with extention .tcl (**prog3.tcl**)

```
set ns [new Simulator]
set nf [open prog3.nam w]
$ns namtrace-all $nf
set nd [open prog3.tr w]
$ns trace-all $nd

proc finish {} {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam prog4.nam &
exit 0
}
set n0 [$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 extention .awk (**prog3.awk**)

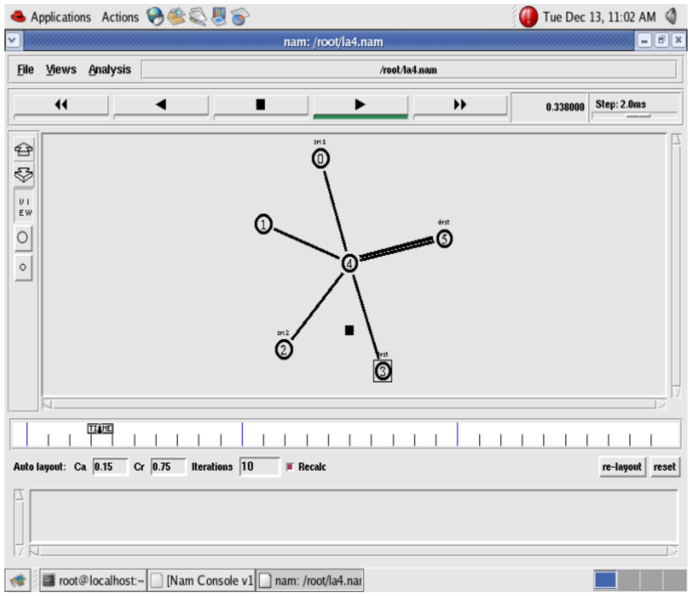
```
BEGIN {
count=0;
}
{
event=$1;
if(event=="d")
{
count++;
}
```

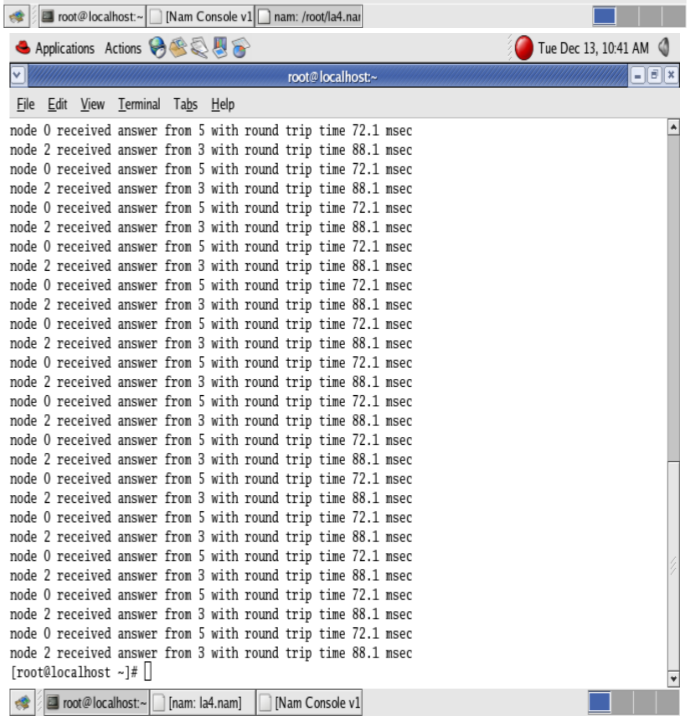
}
END {
printf("No of packets dropped : %d\n",count);
}

Step3: Run the simulation program

[root@localhost~]# ns prog3.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 prog3.awk prog3.tr

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

[root@localhost~]# vi prog3.tr

Experiment 3:

Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plotcongestion window for different source / destination.

Step1: Open text editor, type the below program and save with extention .tcl (**prog5.tcl**)

```
set ns [new Simulator]
set nf [open prog5.nam w]
$ns namtrace-all $nf
set nd [open prog5.tr w]
$ns trace-all $nd

$ns color 1 Blue
$ns color 2 Red

proc finish { } {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam prog5.nam &
exit 0
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]

$n7 shape box
$n7 color Blue
$n8 shape hexagon
$n8 color Red

$ns duplex-link $n1 $n0 2Mb 10ms DropTail
$ns duplex-link $n2 $n0 2Mb 10ms DropTail
$ns duplex-link $n0 $n3 1Mb 20ms DropTail
```

```
$ns make-lan "$n3 $n4 $n5 $n6 $n7 $n8" 512Kb 40ms LL Queue/DropTail Mac/802_3

$ns duplex-link-op $n1 $n0 orient right-down
$ns duplex-link-op $n2 $n0 orient right-up
$ns duplex-link-op $n0 $n3 orient right

$ns queue-limit $n0 $n3 20

set tcp1 [new Agent/TCP/Vegas]
$ns attach-agent $n1 $tcp1
set sink1 [new Agent/TCPSink]
$ns attach-agent $n7 $sink1
$ns connect $tcp1 $sink1
$tcp1 set class_ 1
$tcp1 set packetsize_ 55

set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1

set tfile [open cwnd.tr w]
$tcp1 attach $tfile
$tcp1 trace cwnd_

set tcp2 [new Agent/TCP/Reno]
$ns attach-agent $n2 $tcp2
set sink2 [new Agent/TCPSink]
$ns attach-agent $n8 $sink2
$ns connect $tcp2 $sink2
$tcp2 set class_ 2
$tcp2 set packetSize_ 55

set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2

set tfile2 [open cwnd2.tr w]
$tcp2 attach $tfile2
$tcp2 trace cwnd_
$ns at 0.5 "$ftp1 start"
$ns at 1.0 "$ftp2 start"
```

```
$ns at 5.0 "$ftp2 stop"
$ns at 5.0 "$ftp1 stop"

$ns at 5.5 "finish"
$ns run
```

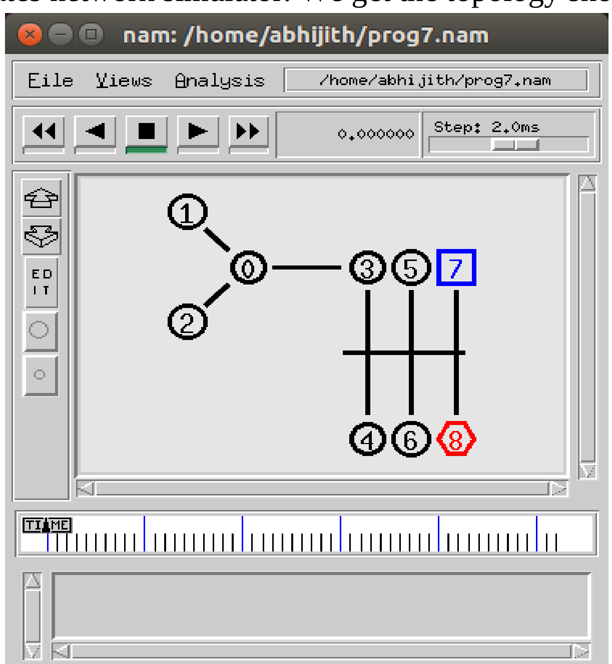
Step2: Open text editor, type the below program and save with extention .awk (**prog5.awk**)

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

Step3: Run the simulation program

[root@localhost~]# ns prog5.tcl

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



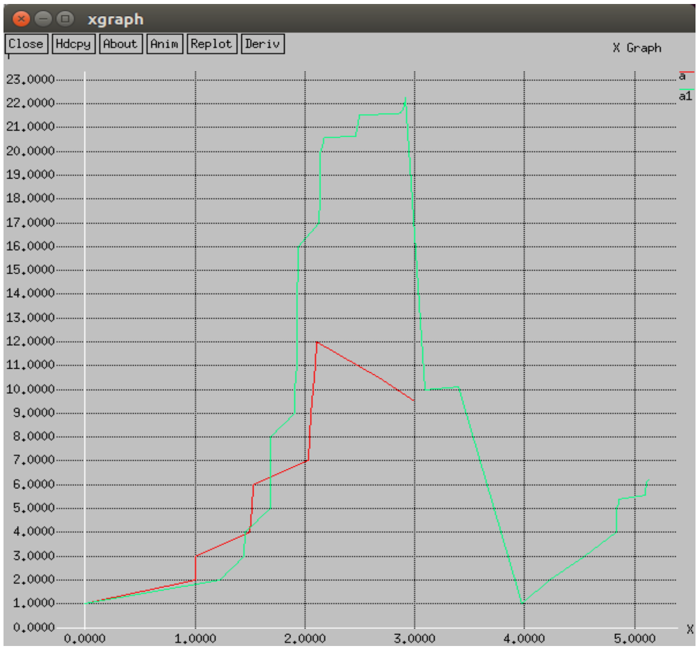
Step 4: Now press the play button in the simulation window and the simulation will begins.

Step 5: After simulation is completed run **awk file** and generate the graph ,

[root@localhost~]# awk –f prog5.awk cwnd.tr> a1

[root@localhost~]# awk –f prog5.awk cwnd2.tr> a2

[root@localhost~]#xgraph a1 a2



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

[root@localhost~]# vi prog5.tr

Experiment 4:

Implement simple ESS and with transmitting nodes in wire-less LAN by simulationand determine the performance with respect to transmission of packets.

Step1: Open text editor, type the below program and save with extention .tcl (**prog6.tcl**)

```
set ns [new Simulator]
set tf [open prog6.tr w]
$ns trace-all $tf
set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open prog6.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
```

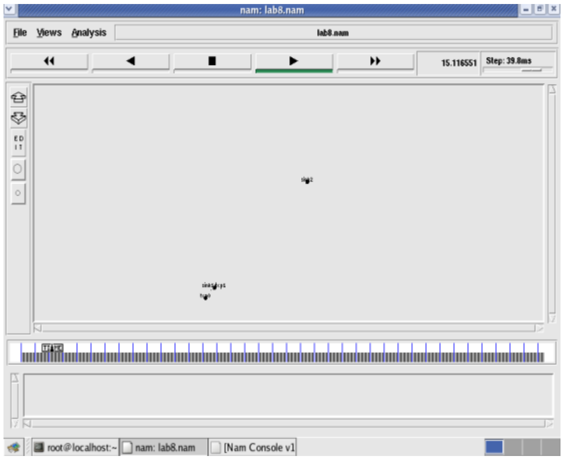
Step2: Open text editor, type the below program and save with extention .awk (**prog6.awk**)

```
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)));
}
```

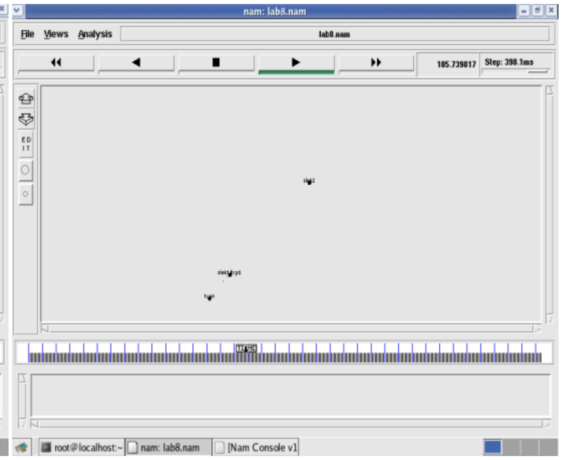
Step3: Run the simulation program

[root@localhost~]# ns prog3.tcl

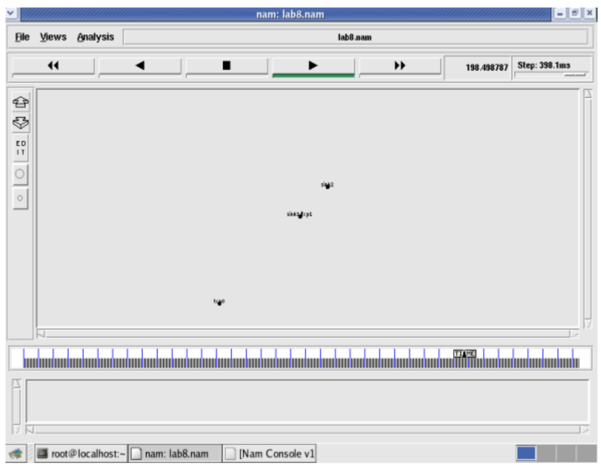
(Here **“ns”** indicates network simulator. We get the topology shown in the snapshot.)



Node 1 and 2 are communicating



Node 2 is moving towards node 3

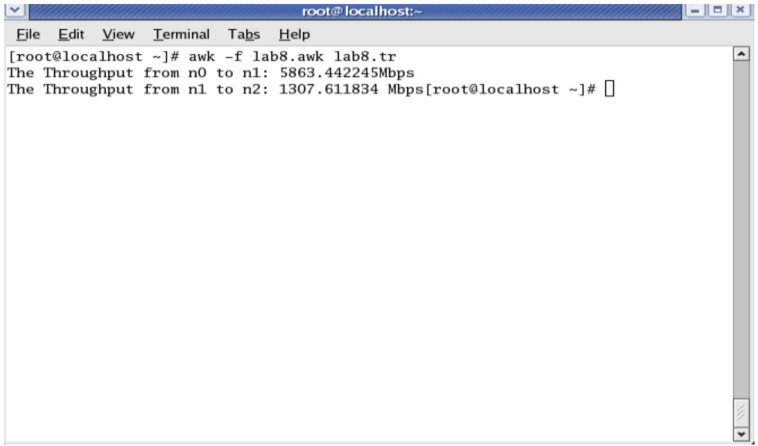


Node 2 is coming back from node 3 towards node1

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 ,

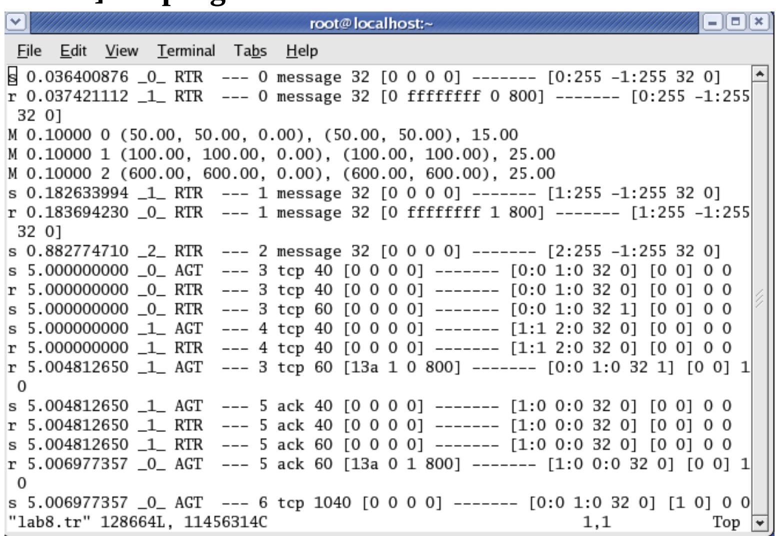
[root@localhost~]# awk –f prog6.awk prog6.tr



```
root@localhost:~
File Edit View Terminal Tabs Help
[root@localhost ~]# awk -f lab8.awk lab8.tr
The Throughput from n0 to n1: 5863.442245Mbps
The Throughput from n1 to n2: 1307.611834 Mbps[root@localhost ~]#
```

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

[root@localhost~]# vi prog2.tr



```
root@localhost:~
File Edit View Terminal Tabs Help
s 0.036400876 _0_ RTR  --- 0 message 32 [0 0 0 0] ------- [0:255 -1:255 32 0]
r 0.037421112 _1_ RTR  --- 0 message 32 [0 ffffffff 0 800] ------- [0:255 -1:255
 32 0]
M 0.10000 0 (50.00, 50.00, 0.00), (50.00, 50.00), 15.00
M 0.10000 1 (100.00, 100.00, 0.00), (100.00, 100.00), 25.00
M 0.10000 2 (600.00, 600.00, 0.00), (600.00, 600.00), 25.00
s 0.182633994 _1_ RTR  --- 1 message 32 [0 0 0 0] ------- [1:255 -1:255 32 0]
r 0.183694230 _0_ RTR  --- 1 message 32 [0 ffffffff 1 800] ------- [1:255 -1:255
 32 0]
s 0.882774710 _2_ RTR  --- 2 message 32 [0 0 0 0] ------- [2:255 -1:255 32 0]
s 5.000000000 _0_ AGT  --- 3 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
r 5.000000000 _0_ RTR  --- 3 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
s 5.000000000 _0_ RTR  --- 3 tcp 60 [0 0 0 0] ------- [0:0 1:0 32 1] [0 0] 0 0
s 5.000000000 _1_ AGT  --- 4 tcp 40 [0 0 0 0] ------- [1:1 2:0 32 0] [0 0] 0 0
r 5.000000000 _1_ RTR  --- 4 tcp 40 [0 0 0 0] ------- [1:1 2:0 32 0] [0 0] 0 0
r 5.004812650 _1_ AGT  --- 3 tcp 60 [13a 1 0 800] ------- [0:0 1:0 32 1] [0 0] 1
 0
s 5.004812650 _1_ AGT  --- 5 ack 40 [0 0 0 0] ------- [1:0 0:0 32 0] [0 0] 0 0
r 5.004812650 _1_ RTR  --- 5 ack 40 [0 0 0 0] ------- [1:0 0:0 32 0] [0 0] 0 0
s 5.004812650 _1_ RTR  --- 5 ack 60 [0 0 0 0] ------- [1:0 0:0 32 0] [0 0] 0 0
r 5.006977357 _0_ AGT  --- 5 ack 60 [13a 0 1 800] ------- [1:0 0:0 32 0] [0 0] 1
 0
s 5.006977357 _0_ AGT  --- 6 tcp 1040 [0 0 0 0] ------- [0:0 1:0 32 0] [1 0] 0 0
"lab8.tr" 128664L, 11456314C                                   1,1          Top
```

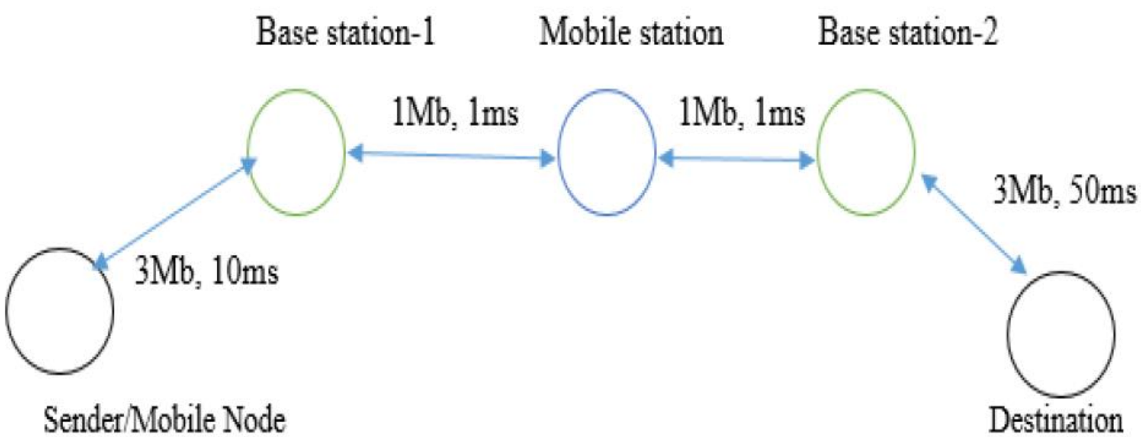
Here **"M"** indicates mobile nodes, **"AGT"** indicates Agent Trace, **"RTR"** indicates Router

Experiment 5:

Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

Second Generation (2G) technology is based on the technology known as global system for mobile communication (GSM). This technology enabled various networks to provide services like text messages, picture messages and MMS. The technologies used in 2G are either TDMA (Time Division Multiple Access) which divides signal into different time slots or CDMA (Code Division Multiple Access) which allocates a special code to each user so as to communicate over a multiplex physical channel.
GSM uses a variation of time division multiple access (TDMA). 2G networks developed as a replacement for first generation (1G) analog cellular networks, and the GSM standard originally described as a digital, circuit-switched network optimized for full duplex voice telephony. This expanded over time to include data communications, first by circuit-switched transport, then by packet data transport via GPRS (General Packet Radio Services).
GSM can be implemented on all the versions of NS2 (Since year 2004: ns-2.27, and later versions of NS2)

Design:



Program:
Step1: Change the directory using
cd ~/ns-allinone-2.35/ns-2.35/tcl/ex/wireless-scripts

Step 2: Type the following program using gedit and save as gsm.tcl

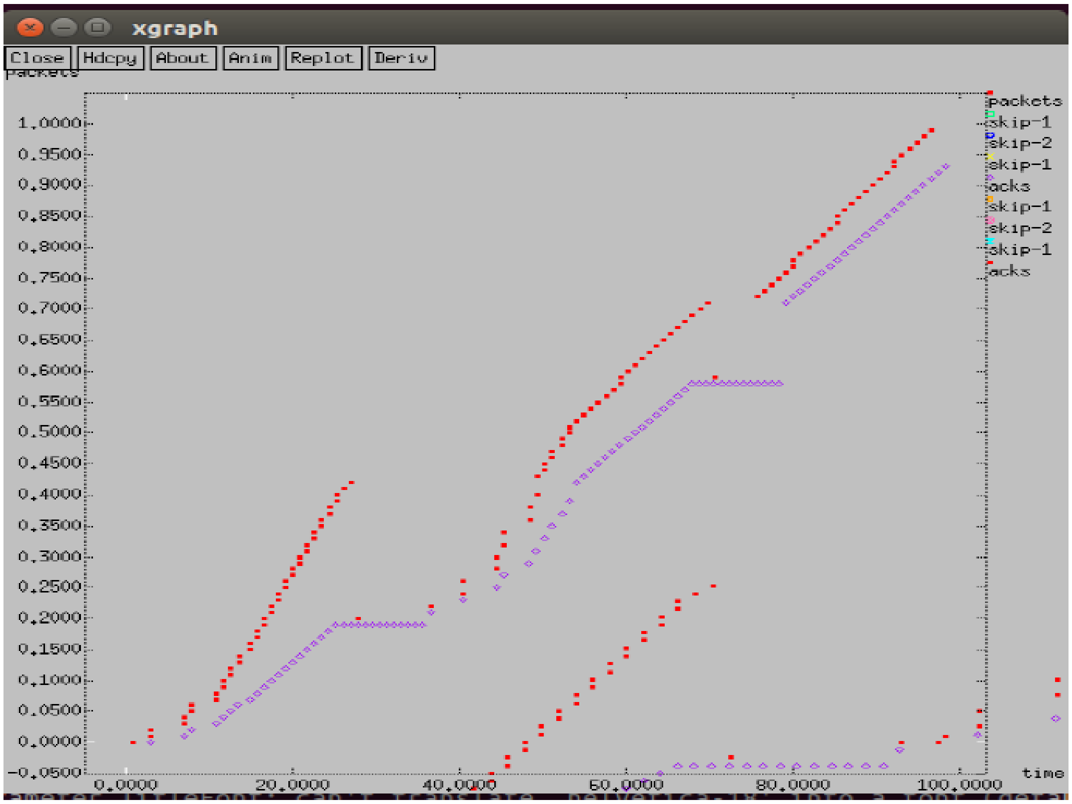
```
set ns [new Simulator]

set tf [open out.tr w]
$ns trace-all $tf
set td [open out.nam w]
$ns namtrace-all $td

# General Parameters
set stop 100       ;# Stop time.
# Topology
```

```
set type gsm      ;#type of link:
# AQM parameters
set minth 30
set maxth 0
set adaptive 1    ;# 1 for Adaptive RED, 0 for plain RED
# Traffic generation.
set flows 0       ;# number of long-lived TCP flows
set window 30   ;# window for long-lived traffic
# Plotting statistics.
set opt(wrap) 100 ;# wrap plots?
set opt(srcTrace) is ;# where to plot traffic
set opt(dstTrace) bs2 ;# where to plot traffic
#default downlink bandwidth in bps
set bwDL(gsm)  9600
#default downlink propagation delay in seconds
set propDL(gsm)  .500

set nodes(is) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(lp) [$ns node]

proc cell_topo {} {
  global ns nodes
  $ns duplex-link $nodes(lp) $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(is) 3Mbps 50ms DropTail
  puts "GSM Cell Topology"
}

proc set_link_params {t} {
  global ns nodes bwDL propDL
  $ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) duplex
  $ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) duplex

  $ns delay $nodes(bs1) $nodes(ms) $propDL($t) duplex
  $ns delay $nodes(bs2) $nodes(ms) $propDL($t) duplex

  $ns queue-limit $nodes(bs1) $nodes(ms) 10
  $ns queue-limit $nodes(bs2) $nodes(ms) 10
}

# RED and TCP parameters
Queue/RED set adaptive_ $adaptive
Queue/RED set thresh_ $minth
Queue/RED set maxthresh_ $maxth
Agent/TCP set window_ $window

source web.tcl
```

```
#Create topology
switch $type {
 gsm -
 cdma {cell_topo}
 }
  set_link_params $type
  $ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
  $ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]

# Set up forward TCP connection
if {$flows == 0} {
            set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
            set ftp1 [[set tcp1] attach-app FTP]
            $ns at 0.8 "[set ftp1] start"
}

proc stop {} {
            global nodes opt tf td
            set wrap $opt(wrap)
            set sid [$nodes($opt(srcTrace)) id]
            set did [$nodes($opt(dstTrace)) id]

            set a "out.tr"

            set GETRC "../../../bin/getrc"
        set RAW2XG "../../../bin/raw2xg"

        exec $GETRC -s $sid -d $did -f 0 out.tr | \
          $RAW2XG -s 0.01  -m $wrap -r > plot.xgr
        exec $GETRC -s $did -d $sid -f 0 out.tr | \
          $RAW2XG -a -s 0.01 -m $wrap  >> plot.xgr

            exec nam out.nam &
            exec xgraph -x time -y packets plot.xgr &

            exit 0
}
$ns at $stop "stop"
$ns run
```

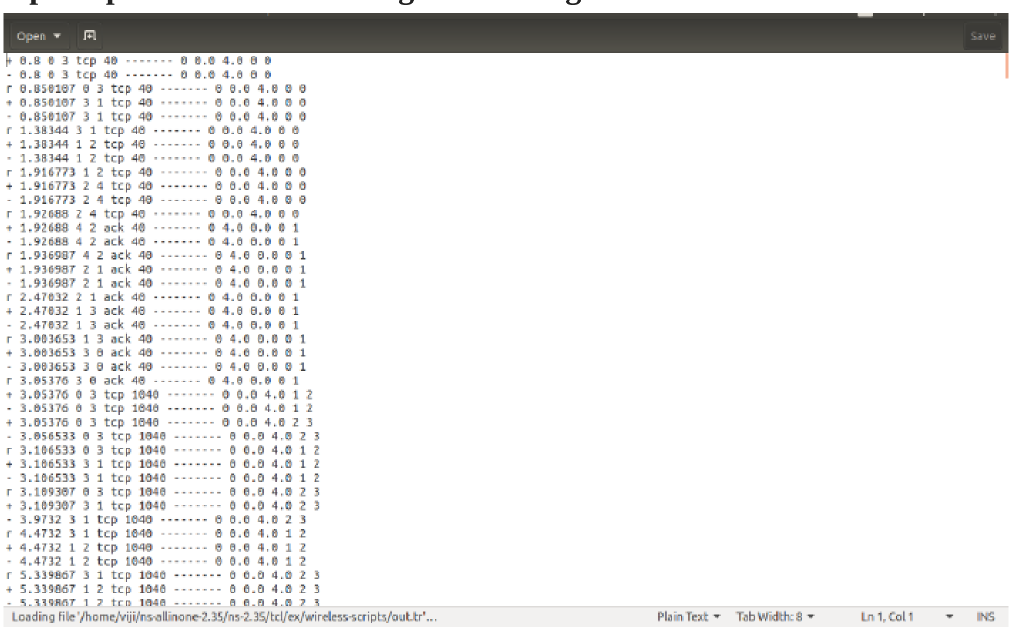
Output:

Step 3: Run the program using command ns gsm.tcl



GSM Trace File:

Step 4: open the trace file using command gedit out.tr



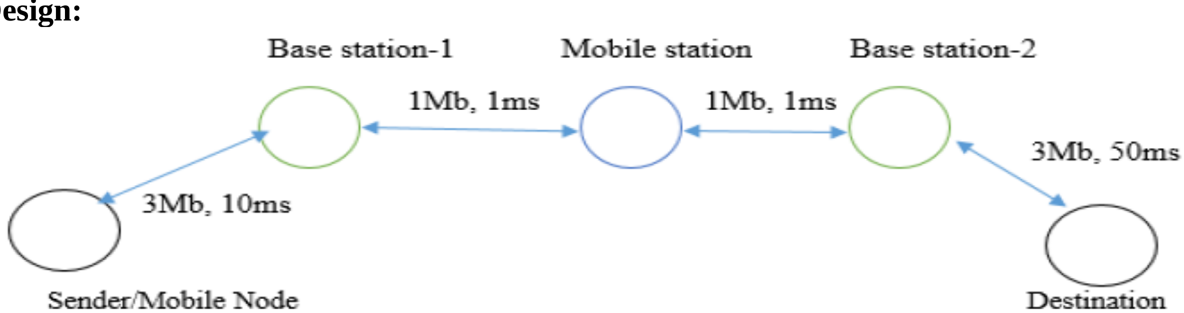
```
+ 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
- 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
r 0.850107 0 3 tcp 40 ------- 0 0.0 4.0 0 0
+ 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
- 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
r 1.38344 3 1 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.38344 1 2 tcp 40 ------- 0 0.0 4.0 0 0
- 1.38344 1 2 tcp 40 ------- 0 0.0 4.0 0 0
r 1.916773 1 2 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.916773 2 4 tcp 40 ------- 0 0.0 4.0 0 0
- 1.916773 2 4 tcp 40 ------- 0 0.0 4.0 0 0
r 1.92688 2 4 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.92688 4 2 ack 40 ------- 0 4.0 0.0 0 1
- 1.92688 4 2 ack 40 ------- 0 4.0 0.0 0 1
r 1.936987 4 2 ack 40 ------- 0 4.0 0.0 0 1
+ 1.936987 2 1 ack 40 ------- 0 4.0 0.0 0 1
- 1.936987 2 1 ack 40 ------- 0 4.0 0.0 0 1
r 2.47032 2 1 ack 40 ------- 0 4.0 0.0 0 1
+ 2.47032 1 3 ack 40 ------- 0 4.0 0.0 0 1
- 2.47032 1 3 ack 40 ------- 0 4.0 0.0 0 1
r 3.003653 1 3 ack 40 ------- 0 4.0 0.0 0 1
+ 3.003653 3 0 ack 40 ------- 0 4.0 0.0 0 1
- 3.003653 3 0 ack 40 ------- 0 4.0 0.0 0 1
r 3.05376 3 0 ack 40 ------- 0 4.0 0.0 0 1
+ 3.05376 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
- 3.05376 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
+ 3.05376 0 3 tcp 1040 ------- 0 0.0 4.0 2 3
- 3.056533 0 3 tcp 1040 ------- 0 0.0 4.0 2 3
r 3.106533 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
+ 3.106533 3 1 tcp 1040 ------- 0 0.0 4.0 1 2
- 3.106533 3 1 tcp 1040 ------- 0 0.0 4.0 1 2
r 3.109307 0 3 tcp 1040 ------- 0 0.0 4.0 2 3
+ 3.109307 3 1 tcp 1040 ------- 0 0.0 4.0 2 3
- 3.9732 3 1 tcp 1040 ------- 0 0.0 4.0 2 3
r 4.4732 3 1 tcp 1040 ------- 0 0.0 4.0 1 2
+ 4.4732 1 2 tcp 1040 ------- 0 0.0 4.0 1 2
- 4.4732 1 2 tcp 1040 ------- 0 0.0 4.0 1 2
r 5.339867 3 1 tcp 1040 ------- 0 0.0 4.0 2 3
+ 5.339867 1 2 tcp 1040 ------- 0 0.0 4.0 2 3
- 5.339867 1 2 tcp 1040 ------- 0 0.0 4.0 2 3
```

Loading file '/home/viji/ns-allinone-2.35/ns-2.35/tcl/ex/wireless-scripts/out.tr'... Plain Text Tab Width: 8 Ln 1, Col 1 INS

Experiment 6:
Implement and study the performance of CDMA on NS2/NS3 (Using stack calledCall net) or equivalent environment.

3G networks developed as a replacement for second generation (2G) GSM standard network with full duplex voice telephony. CDMA is used as the access method in many mobile phone standards. IS-95, also called cdmaOne, and its 3G evolution CDMA2000, are often simply referred to as CDMA, but UMTS(The Universal Mobile Telecommunications System is a third generation mobile cellular system for networks based on the GSM standard.), the 3G standard used by GSM carriers, also uses wideband CDMA. Long-Term Evolution (LTE) is a standard for high-speed wireless communication which uses CDMA network technology.
3G technology generally refers to the standard of accessibility and speed of mobile devices. The standards of the technology were set by the International Telecommunication Union (ITU). This technology enables use of various services like GPS (Global Positioning System), mobile television and video conferencing. It not only enables them to be used worldwide, but also provides with better bandwidth and increased speed. The main aim of this technology is to allow much better coverage and growth with minimum investment.
CDMA can be implemented on all the versions of NS2 (Since year 2004: ns-2.27, and later versions of NS2)

Design:



Program:
Step1: Change the directory using
cd ~/ns-allinone-2.35/ns-2.35/tcl/ex/wireless-scripts

Step 2: Type the following program using gedit and save as cdma.tcl

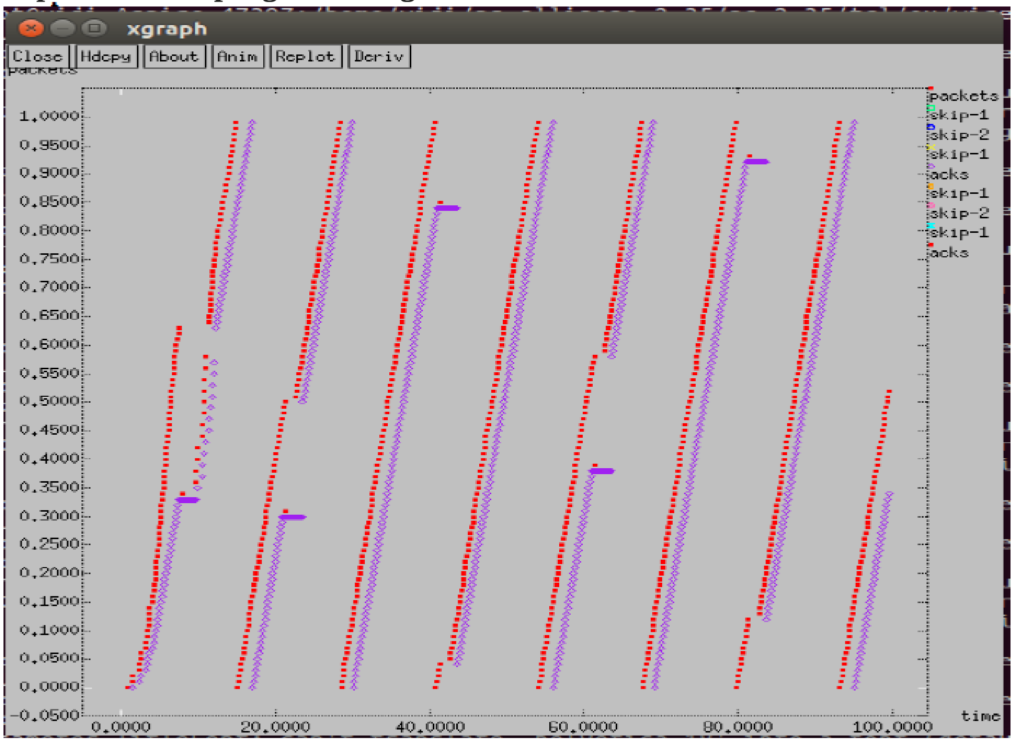
```
set ns [new Simulator]
set tf [open out.tr w]
$ns trace-all $tf
set td [open out.nam w]
$ns namtrace-all $td

# General Parameters
set stop 100       ;# Stop time.
# Topology
```

```
set type cdma ;#type of link
# AQM parameters
set minth 30
set maxth 0
set adaptive 1 ;# 1 for Adaptive RED, 0 for plain RED
# Traffic generation.
set flows 0 ;# number of long-lived TCP flows
set window 30 ;# window for long-lived traffic
# Plotting statics.
set opt(wrap) 100 ;# wrap plots?
set opt(srcTrace) is ;# where to plot traffic
set opt(dstTrace) bs2 ;# where to plot traffic
#default downlink bandwidth in bps
set bwDL(cdma) 384000
#default downlink propagation delay in seconds
set propDL(cdma) .150

set nodes(is) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(lp) [$ns node]

proc cell_topo {} {
global ns nodes
$ns duplex-link $nodes(lp) $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(is) 3Mbps 50ms DropTail
puts " cdma Cell Topology"
}

proc set_link_para {t} {
global ns nodes bwDL propDL
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) duplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) duplex

$ns delay $nodes(bs1) $nodes(ms) $propDL($t) duplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) duplex

$ns queue-limit $nodes(bs1) $nodes(ms) 20
$ns queue-limit $nodes(bs2) $nodes(ms) 20
}
# RED and TCP parameters
```

```
Queue/RED set adaptive_ $adaptive
Queue/RED set thresh_ $minth
Queue/RED set maxthresh_ $maxth
Agent/TCP set window_ $window

source web.tcl
#Create topology
switch $type {
cdma {cell_topo}
}

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

# Set up forward TCP connection
if {$flows == 0} {
set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp1 [[set tcp1] attach-app FTP]
$ns at 0.8 "[set ftp1] start"
}

proc stop {} {
global nodes opt tf td
set wrap $opt(wrap)
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]

set a "out.tr"

set GETRC "../../../bin/getrc"
set RAW2XG "../../../bin/raw2xg"

exec $GETRC -s $sid -d $did -f 0 out.tr | \
$RAW2XG -s 0.01 -m $wrap -r > plot.xgr
exec $GETRC -s $did -d $sid -f 0 out.tr | \
$RAW2XG -a -s 0.01 -m $wrap >> plot.xgr

exec nam out.nam &
exec xgraph -x time -y packets plot.xgr &
exit 0
}
$ns at $stop "stop"
$ns run
```

Output:

Step 3: Run the program using command ns cdma.tcl



CDMA Trace File:

Step 4: open the trace file using command gedit out.tr

```
+ 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
- 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
r 0.850107 0 3 tcp 40 ------- 0 0.0 4.0 0 0
+ 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
- 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
r 1.00094 3 1 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.00094 1 2 tcp 40 ------- 0 0.0 4.0 0 0
- 1.00094 1 2 tcp 40 ------- 0 0.0 4.0 0 0
r 1.15594 1 2 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.15594 2 4 tcp 40 ------- 0 0.0 4.0 0 0
- 1.15594 2 4 tcp 40 ------- 0 0.0 4.0 0 0
r 1.166047 2 4 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.166047 4 2 ack 40 ------- 0 4.0 0.0 0 1
- 1.166047 4 2 ack 40 ------- 0 4.0 0.0 0 1
r 1.176153 4 2 ack 40 ------- 0 4.0 0.0 0 1
+ 1.176153 2 1 ack 40 ------- 0 4.0 0.0 0 1
- 1.176153 2 1 ack 40 ------- 0 4.0 0.0 0 1
r 1.326987 2 1 ack 40 ------- 0 4.0 0.0 0 1
+ 1.326987 1 3 ack 40 ------- 0 4.0 0.0 0 1
- 1.326987 1 3 ack 40 ------- 0 4.0 0.0 0 1
r 1.481987 1 3 ack 40 ------- 0 4.0 0.0 0 1
+ 1.481987 3 0 ack 40 ------- 0 4.0 0.0 0 1
- 1.481987 3 0 ack 40 ------- 0 4.0 0.0 0 1
r 1.532093 3 0 ack 40 ------- 0 4.0 0.0 0 1
+ 1.532093 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
- 1.532093 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
+ 1.532093 0 3 tcp 1040 ------- 0 0.0 4.0 2 3
- 1.534867 0 3 tcp 1040 ------- 0 0.0 4.0 2 3
r 1.584867 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
+ 1.584867 3 1 tcp 1040 ------- 0 0.0 4.0 1 2
- 1.584867 3 1 tcp 1040 ------- 0 0.0 4.0 1 2
r 1.58764 0 3 tcp 1040 ------- 0 0.0 4.0 2 3
+ 1.58764 3 1 tcp 1040 ------- 0 0.0 4.0 2 3
- 1.606533 3 1 tcp 1040 ------- 0 0.0 4.0 2 3
r 1.756533 3 1 tcp 1040 ------- 0 0.0 4.0 1 2
+ 1.756533 1 2 tcp 1040 ------- 0 0.0 4.0 1 2
- 1.756533 1 2 tcp 1040 ------- 0 0.0 4.0 1 2
r 1.7782 3 1 tcp 1040 ------- 0 0.0 4.0 2 3
+ 1.7782 1 2 tcp 1040 ------- 0 0.0 4.0 2 3
- 1.806533 1 2 tcp 1040 ------- 0 0.0 4.0 2 3
```

Part B Programs

Experiment No 1
CRC

Problem Statement

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.io.*;
class crc_gen
{
   public static void main(String args[])
   {
      BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
      int[] data;
      int[] div;
      int[] divisor;
      int[] rem;
      int[] crc;
      int data_bits, divisor_bits, tot_length;

      System.out.println("Enter number of data bits : ");
      data_bits=Integer.parseInt(br.readLine());
      data=new int[data_bits];

      System.out.println("Enter data bits : ");
      for(int i=0; i<data_bits; i++)
         data[i]=Integer.parseInt(br.readLine());

divisor_bits = 17;
divisor = new int[]{1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1};

      tot_length=data_bits+divisor_bits-1;

      div=new int[tot_length];
      rem=new int[tot_length];
      crc=new int[tot_length];
   /*------------------- CRC GENERATION------------------------*/
      for(int i=0;i<data.length;i++)
         div[i]=data[i];

      System.out.print("Dividend (after appending 0's) are : ");
      for(int i=0; i< div.length; i++)
         System.out.print(div[i]);
      System.out.println();

      for(int j=0; j<div.length; j++){
           rem[j] = div[j];
```

```
    }

    rem=divide(divisor, rem);

    for(int i=0;i<div.length;i++)          //append dividend and ramainder
    {
       crc[i]=(div[i]^rem[i]);
    }

    System.out.println();
    System.out.println("CRC code : ");
    for(int i=0;i<crc.length;i++)
       System.out.print(crc[i]);

  /*-------------------ERROR DETECTION---------------------*/
    System.out.println();
    System.out.println("Enter CRC code of "+tot_length+" bits : ");
    for(int i=0; i<crc.length; i++)
       crc[i]=Integer.parseInt(br.readLine());

    for(int j=0; j<crc.length; j++){
         rem[j] = crc[j];
    }

    rem=divide(divisor, rem);

    for(int i=0; i< rem.length; i++)
    {
       if(rem[i]!=0)
       {
          System.out.println("Error");
          break;
       }
       if(i==rem.length-1)
          System.out.println("No Error");
    }

    System.out.println("THANK YOU.... :)");
  }
   static int[] divide(int divisor[], int rem[])
```

```
    {
        int cur=0;
        while(true)
        {
            for(int i=0;i<divisor.length;i++)
                rem[cur+i]=(rem[cur+i]^divisor[i]);

            while(rem[cur]==0 && cur!=rem.length-1)
                cur++;

            if((rem.length-cur)<divisor.length)
                break;
        }
        return rem;
    }
}
```

OUTPUT:

<u>RUN1:</u>

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

C:\Users\abhijith\Desktop>java crc_gen
Enter number of data bits :
3
Enter data bits :
1 1 0
Dividend (after appending 0's) are : 1100000000000000000

CRC code :
1100110000011000110
Enter CRC code of 19 bits :
1 1 0 0 1 1 0 0 0 0 0 1 1 0 0 0 1 1 0
No Error
THANK YOU.... :)
```

<u>RUN 2:</u>

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

C:\Users\abhijith\Desktop>java crc_gen
Enter number of data bits :
3
Enter data bits :
1 1 0
Dividend (after appending 0's) are : 1100000000000000000

CRC code :
1100110000011000110
Enter CRC code of 19 bits :
1 1 0 1 1 1 0 1 1 0 1 1 0 0 1 0 1 0 1
Error
THANK YOU.... :)
```

Experiment No 2

Bellman-Ford algorithm

Problem Statement

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.

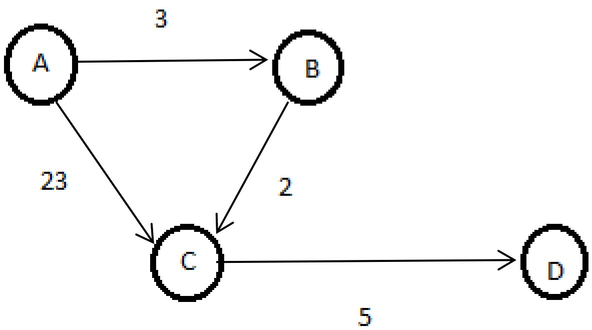
The Routing tables are shared among the neighbors, and the tables at the router are updated, such that the router will know the shortest path to the destination.

Program

```
import java.io.*;
import java.util.Scanner;
class dist_vec
{
public static void main(String args[])
{
        int dmat[][];
        int dist[][];
        int via[][];
```

```
        int n=0,i=0,j=0,k=0,count=0;
        Scanner in = new Scanner(System.in);

        System.out.println("enter the number of nodes\n");
        n = in.nextInt();

        dmat = new int[n][n];
        dist = new int[n][n];
        via = new int[n][n];
        System.out.println("enter the cost matrix\n");
        for(i=0;i<n;i++)
        for(j=0;j<n;j++)
        {
                dmat[i][j] = in.nextInt();
                dmat[i][i]=0;
                dist[i][j]=dmat[i][j];
                via[i][j]=j;
        }
        do
        {
                count=0;
                for(i=0;i<n;i++)
                for(j=0;j<n;j++)
                for(k=0;k<n;k++)
                if(dist[i][j]>dmat[i][k]+dist[k][j])
                {
                        dist[i][j]=dist[i][k]+dist[k][j];
                        via[i][j]=k;
                        count++;
                }
        }while(count!=0);
        for(i=0;i<n;i++)
        {
                System.out.println("state value for router"+i+" is");
                for(j=0;j<n;j++)
                {
                        System.out.println("To "+j+" -Via "+via[i][j]+" distance is "+dist[i][j]);
                }
        }
}
}
```

Output 1




```
enter the number of nodes

4
enter the cost matrix

0 3 23 999
999 0 2 999
999 999 999 5
999 999 999 999
state value for router0 is
To 0 -Via 0 distance is 0
To 1 -Via 1 distance is 3
To 2 -Via 1 distance is 5
To 3 -Via 2 distance is 10
state value for router1 is
To 0 -Via 0 distance is 999
To 1 -Via 1 distance is 0
To 2 -Via 2 distance is 2
To 3 -Via 2 distance is 7
state value for router2 is
To 0 -Via 0 distance is 999
To 1 -Via 1 distance is 999
To 2 -Via 2 distance is 0
To 3 -Via 3 distance is 5
state value for router3 is
To 0 -Via 0 distance is 999
To 1 -Via 1 distance is 999
To 2 -Via 2 distance is 999
To 3 -Via 3 distance is 0
```

Experiment 3

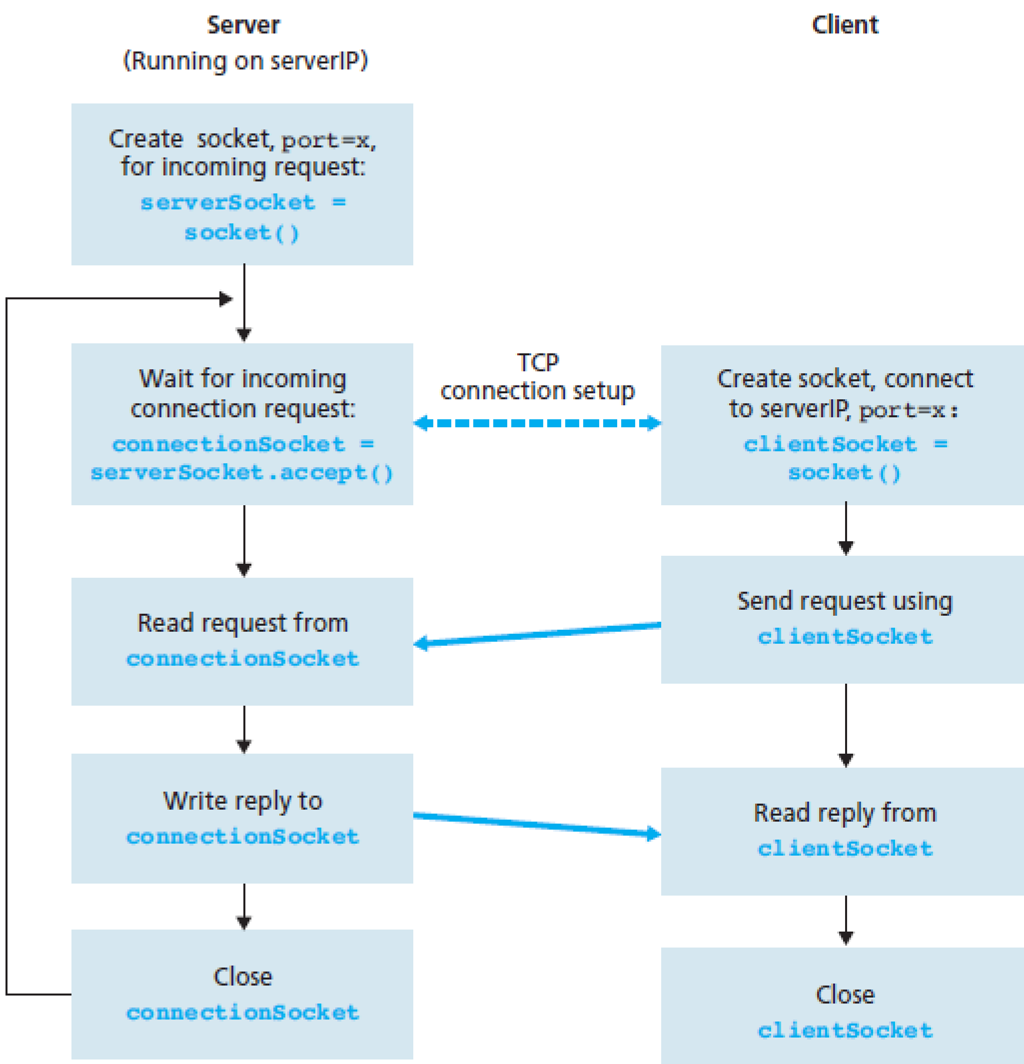
Client-Server Program using TCP/IP sockets

Problem statement:

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.

Theory:

Procedure:



Server Program

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

public class TCPServer
{
```

```
  public static void main(String args[]) throws Exception
{
// establishing the connection with the server
     ServerSocket sersock = new ServerSocket(4000);
     System.out.println("Server ready for connection");
     Socket sock = sersock.accept();            // binding with port: 4000
     System.out.println("Connection is successful and wating for chatting");
     // reading the file name from client
     InputStream istream = sock.getInputStream( );
     BufferedReader fileRead =new BufferedReader(new InputStreamReader(istream));
     String fname = fileRead.readLine( );
     // reading file contents
     BufferedReader contentRead = new BufferedReader(new FileReader(fname) );

     // keeping output stream ready to send the contents
     OutputStream ostream = sock.getOutputStream( );
     PrintWriter pwrite = new PrintWriter(ostream, true);

     String str;
     while((str = contentRead.readLine()) !=  null) // reading line-by-line from file
     {
          pwrite.println(str);         // sending each line to client
     }

     sock.close();  sersock.close();       // closing network sockets
     pwrite.close();  fileRead.close(); contentRead.close();
  }
}
```

Client Program:

```
import java.net.*;
import java.io.*;
public class TCPClient
{
  public static void main( String args[ ] ) throws Exception
  {
     Socket sock = new Socket( "127.0.0.1", 4000);

     // reading the file name from keyboard. Uses input stream
     System.out.print("Enter the file name");
```

```
    BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
    String fname = keyRead.readLine();

// sending the file name to server. Uses PrintWriter
    OutputStream  ostream = sock.getOutputStream( );
    PrintWriter pwrite = new PrintWriter(ostream, true);
    pwrite.println(fname);

// receiving the contents from server.  Uses input stream
    InputStream istream = sock.getInputStream();
    BufferedReader socketRead = new BufferedReader(new InputStreamReader(istream));

    String str;
    while((str = socketRead.readLine())  !=  null) // reading line-by-line
    {
       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

```
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC: ~
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ javac TCPServer.java
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ java TCPServer
Server ready for connection
Connection is successful and wating for chatting
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$
```

Open new terminal, compile and execute client side program

```
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC: ~
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ javac TCPClient.java
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ java TCPClient
Enter the file nameabc.txt
HI
HELLO

abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$
```

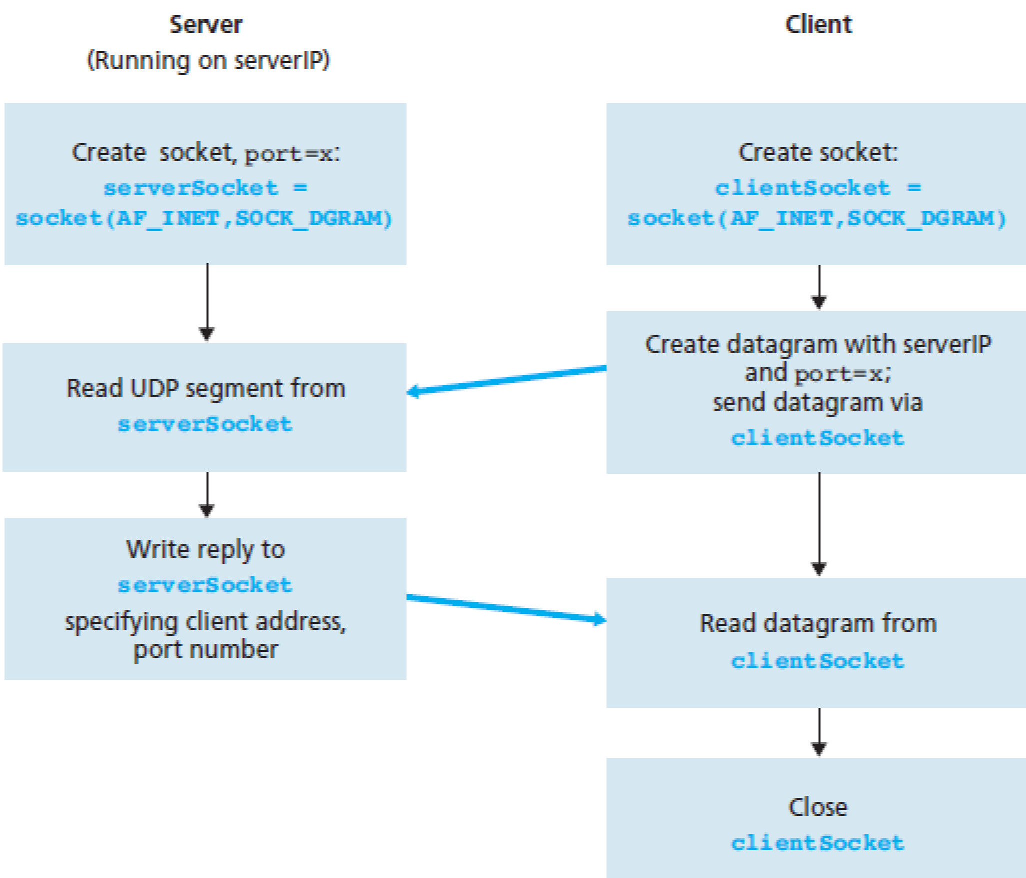
Experiment No 4
Client-Server Program using UDP Socket

Problem Statement

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

Theory

Procedure:



Server Program

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

class UDPServer
{
  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();
            DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
IPAddress, port);
            serverSocket.send(sendPacket);
            serverSocket.close();
      }
}
```

Client Program

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

class UDPClient
{
   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

```
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC: ~
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ javac UDPServer.java
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ java UDPServer
RECEIVED: Hello Server
Enter the Message
Hello Client
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$
```

Open new terminal, compile and execute client side program

```
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC: ~
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ javac UDPClient.java
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$ java UDPClient
FROM SERVER:Hello Client
abhijith@abhijith-HP-Pavilion-g6-Notebook-PC:~$
```

Experiment No 5
RSA Algorithm

Problem Statement

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* x *Q* = 11 x 3 = 33
   Z = (*P*-1) x (*Q*-1) = 10 x 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* x *D* ≡ 1 (mod *Z*)
   compute $D = E^{\wedge-1} \bmod Z = 3^{\wedge-1} \bmod 20$
   find a value for *D* such that *Z* divides ((*E* x *D*)-1)
   find *D* such that 20 divides 3D-1.
   Simple testing (D = 1, 2, ...) gives D = 7
   Check: (*E* x *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 mod 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.BigInteger;
import java.util.Random;

public class RSA
{
   private BigInteger p;
   private BigInteger q;
   private BigInteger N;
   private BigInteger phi;
```

```
private BigInteger e;
private BigInteger 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 / 2, 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;
}

@SuppressWarnings("deprecation")
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()));
   // encrypt
   byte[] encrypted = rsa.encrypt(teststring.getBytes());
```

```
        // decrypt
        byte[] decrypted = rsa.decrypt(encrypted);
        System.out.println("Decrypting Bytes: " + bytesToString(decrypted));
        System.out.println("Decrypted String: " + new String(decrypted));
    }


private static String bytesToString(byte[] encrypted)
{
        String test = "";
        for (byte b : encrypted)
        {
            test += Byte.toString(b);
        }
        return test;
}

    // Encrypt message
    public byte[] encrypt(byte[] message)
    {
        return (new BigInteger(message)).modPow(e, N).toByteArray();
    }

    // Decrypt message
    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: 1041053298971101039710811114101
Decrypting Bytes: 1041053298971101039710811114101
Decrypted String: hi bangalore
```

Experiment No 6
Leaky Bucket

Problem Statement

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.

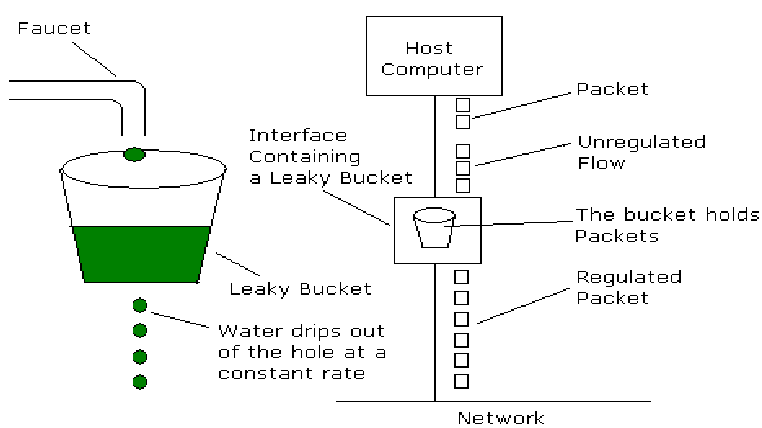
In contrast, closed loop solutions are based on the concept of a feedback loop if there is any congestion. Closed loop algorithms are also divided into two sub categories: explicit feedback and implicit feedback. In explicit feedback algorithms, packets are sent back from the point of congestion to warn 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 widelyused 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.lang.*;
import java.util.Random;
import java.io.*;
import java.util.Scanner;
class leaky_bucket
{
public static void main(String args[])
{
	int drop=0,mini,nsec,p_remain=0;
	int o_rate,b_size,i,packet[];

	packet = new int[100];

	Scanner in = new Scanner(System.in);

	System.out.println("Enter bucket size:");
	b_size = in.nextInt();

	System.out.println("Enter the output rate:");
	o_rate = in.nextInt();
	System.out.println("Enter the number of seconds you want to simulate:");
	nsec = in.nextInt();
	Random rand = new Random();
	for(i=0;i<nsec;i++)
		packet[i]=((rand.nextInt(9)+1)*10);
	System.out.println("Seconds|packets received|packets 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.print(drop+"                    ");
                    System.out.println();
                    drop=0;
            }
        }
        while(p_remain!=0)
        {
            if(p_remain>b_size)
            {
                    drop=p_remain-b_size;
                    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

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-relatedproblems 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.