SHRI MADHWA VADIRAJA INSTITUTE OF TECHNOLOGY & MANAGEMENT

Vishwothama Nagar, Bantakal – 574 115,Udupi Dist.

(A unit of Shri Sode Vadiraja Matt Education Trust)

LABORATORY MANUAL

for

Computer Network Laboratory

[As per Choice Based Credit System (CBCS) scheme]
(Effective from the academic year 2018 -2019)

Semester**:** Vth SEMESTER B.E.

Subjectcode:18CSL57

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

COMPUTER NETWORK LABORATORY **(Effective from the academic year 2018 -2019)** **SEMESTER – V**			
Subject Code	**:18CSL57**	**IA Marks**	**:40**
Number of Contact Hours/Week	**:02IT+ 02P**	**Exam Marks**	**:60**
Total Number of Lab Contact Hours	**:36**	**Exam Hours**	**:03**
Credits – 2			
Course Learning Objectives: This course (18CSL57) will enable students to:			
• Demonstrate operation of network and its management commands • Simulate and demonstrate the performance of GSM and CDMA • Implement data link layer and transport layer protocols.			
Description (If any):			
• 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. • Installation procedure of the required software must be demonstrated, carried out in groups and documented in the journal.			
Programs List:			
PART A			
1. Implement three nodes point–to–point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.			
2. Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.			
3. Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.			
4. Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.			
5. Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.			
6. Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment			
PART B (Implement the following in Java)			
7. Write a program for error detecting code using CRC-CCITT (16- bits).			
8. Write a program to find the shortest path between vertices using bellman-ford algorithm.			
9. Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.			
10. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.			
11. Write a program for simple RSA algorithm to encrypt and decrypt the data.			
12. Write a program for congestion control using leaky bucket algorithm.			

Trace Format Example

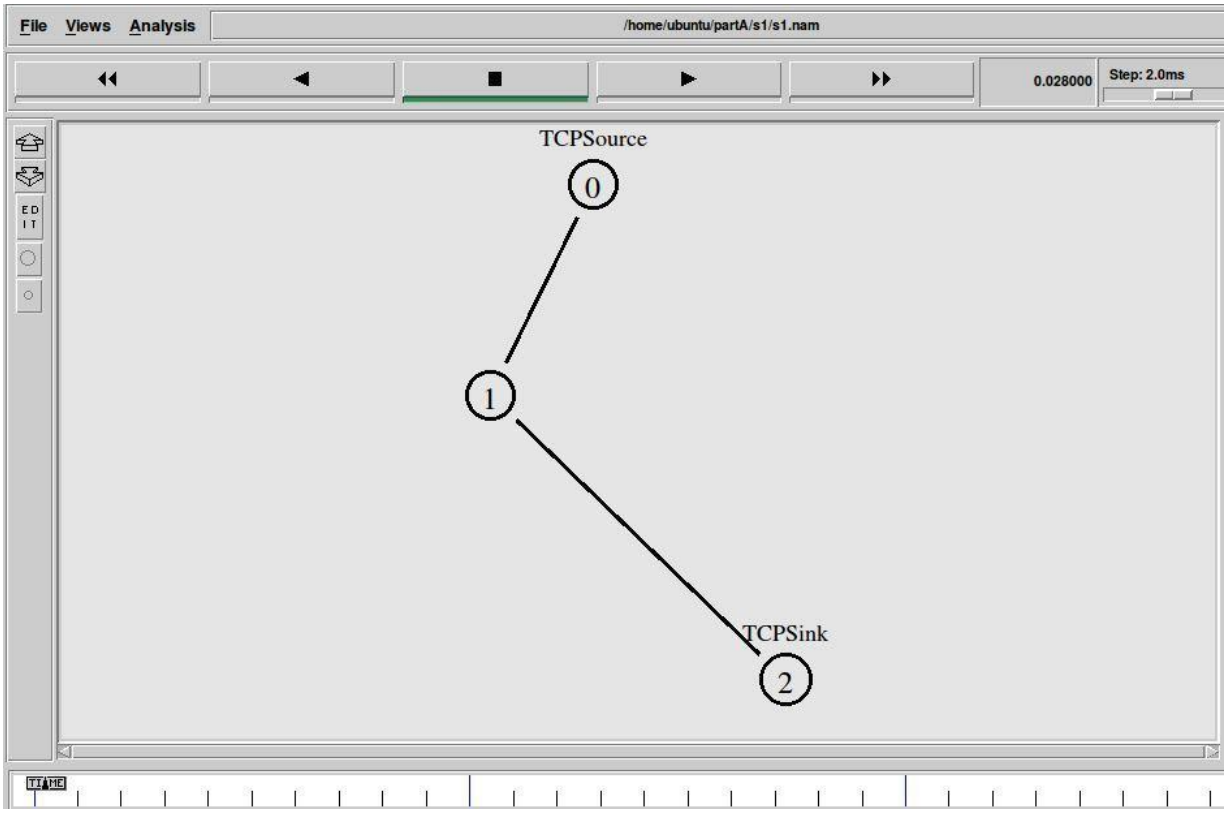
event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

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

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

1. An event (+, -, d, r) descriptor
2. Simulation time (in seconds) of that event
3. From node
4. To node (3,4 identifies the link on which the event occurred)
5. Packet type (in Bytes)
6. Packet size (in Bytes)
7. Flags (appeared as "----- " since no flag is set)
8. Flow id (fid) of IPv6 that a user can set for each flow at the input OTcl script. Even though fid field may not be used in a simulation, users can use this field for analysis purposes. The fid field is also used when specifying stream color for the NAM display.
9. Source address in forms of "**node.port**".
10. Destination address in forms of "**node.port**".
11. Network layer protocol's packet sequence number. Note that even though UDP implementations do not use sequence number, NS keeps track of UDP packet sequence number for analysis purposes.
12. Unique id of the packet

1.Simulate a three nodes point – to – point network with duplex links between them.Set the queue size and vary the bandwidth and find the number of packets dropped.



s1.awk

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

END{
        printf("\nNumber of packets dropped is: %d\n", count);
}
```

s1.tcl

```
#create new simulation instance
set ns [new Simulator]

#open trace file
settracefile [open s1.tr w]
$ns trace-all $tracefile

#open nam:animation file

setnamfile [open s1.nam w]
$ns namtrace-all $namfile

#define finish procedure to perform at the end of simulation
proc finish {} {
        global ns tracefilenamfile
        $ns flush-trace

        #dump all traces and close files
        close $tracefile
        close $namfile

        #execute nam animation file
        execnam s1.nam &

        #execute awk file in background
        execawk -f s1.awk s1.tr &
        exit 0
}

#create 3 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

#create labels
$n0 label "TCPSource"
$n2 label "TCPSink"

#set color
$ns color 1 red

#create link between nodes /create topology
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail

#set queue size of N packets between n1 and n2
$ns queue-limit $n1 $n2 5
```

#create TCP agent and attach to node 0

settcp [new Agent/TCP]
$ns attach-agent $n0 $tcp

#create TCP sink agent and attach to node 2
settcpsink [new Agent/TCPSink]
$ns attach-agent $n2 $tcpsink

#create traffic: FTP: create FTP source agent on top of TCP and attach to TCP agent
set ftp [new Application/FTP]
$ftp attach-agent $tcp

#connect TCP agent with TCP sink agent
$ns connect $tcp $tcpsink

#set the color
$tcp set class_ 1

#schedule events
$ns at 0.2 "$ftp start"
$ns at 2.5 "$ftp stop"
$ns at 3.0 "finish"
$ns run

Output

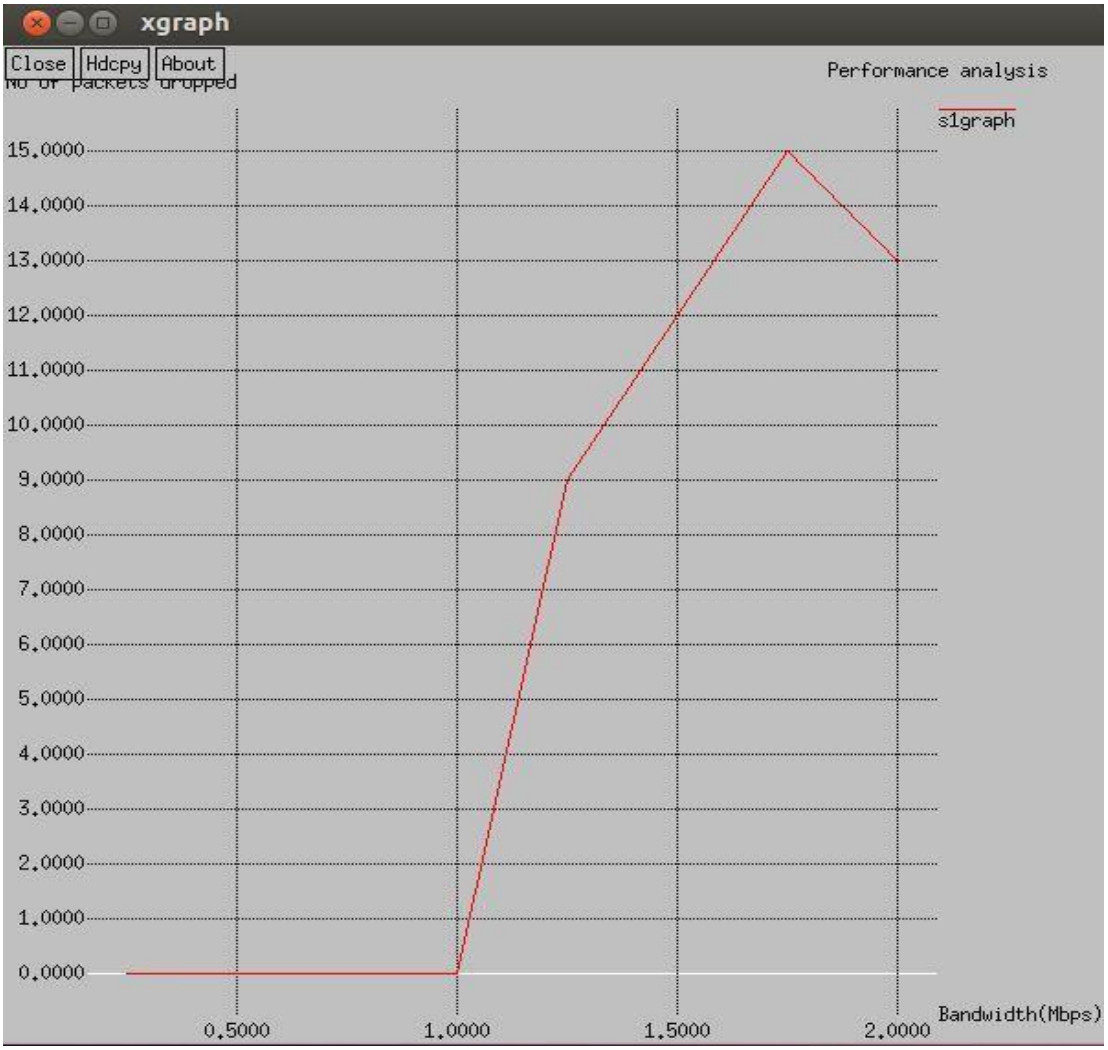
#gedit s1.tcl
#sudo ns s1.tcl
Number of packets dropped is: 0

#gedit s1graph

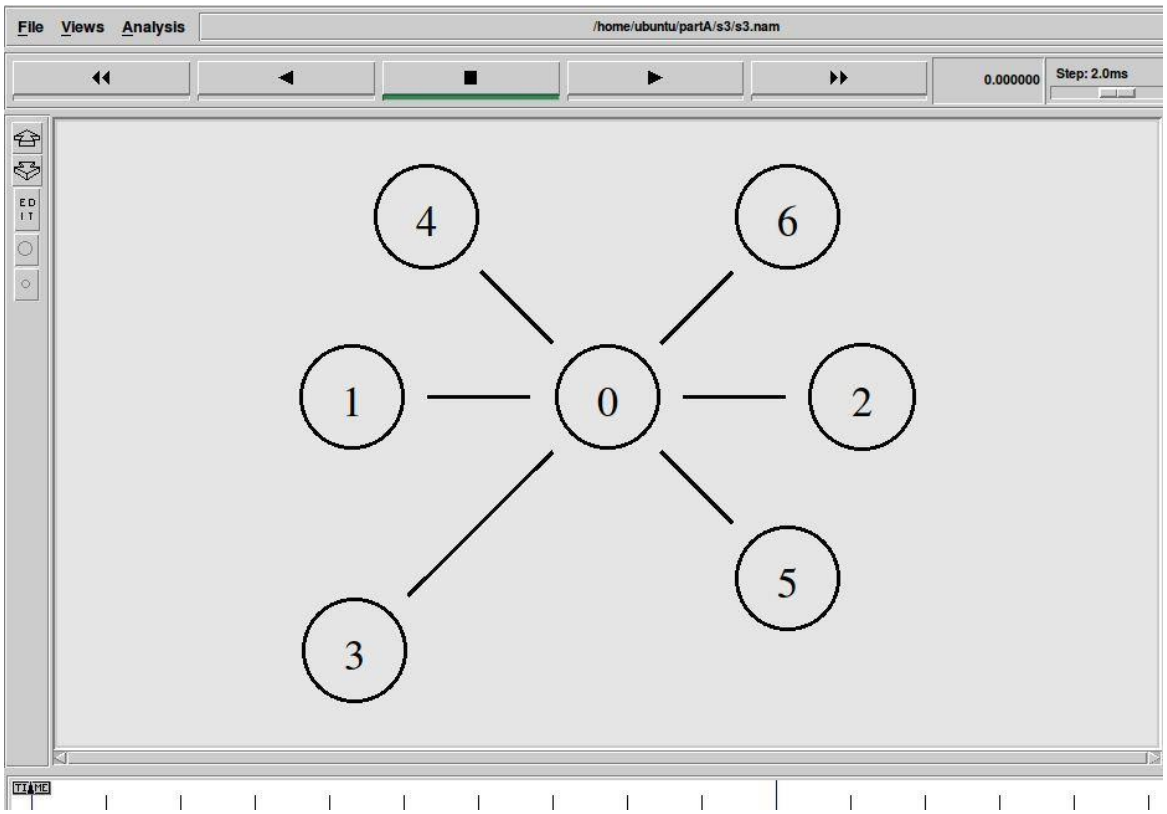
s1graph

0.25	0
0.50	0
0.75	0
1.00	0
1.25	9
1.50	12
1.75	15
2.00	13

#xgraph -x "Bandwidth(Mbps)" -y "No of packets dropped" -t "Performance analysis" **s1graph**


xgraph
Close
Hdcpy
About
No of packets dropped
Performance analysis
s1graph
15.0000
14.0000
13.0000
12.0000
11.0000
10.0000
9.0000
8.0000
7.0000
6.0000
5.0000
4.0000
3.0000
2.0000
1.0000
0.0000
0.5000
1.0000
1.5000
2.0000
Bandwidth(Mbps)

2. Simulate the transmission of ping messages over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.



s2.awk

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

s2.tcl

```
set ns [new Simulator]

setnamfile [open s2.nam w]
$ns namtrace-all $namfile
settracefile[open s2.tr w]
$ns trace-all $tracefile

proc finish {} {
        global ns namfiletracefile
        $ns flush-trace
        close $namfile
        close $tracefile
        execnam s2.nam &
        execawk -f s2.awk s2.tr &
        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 1.75Mb 20ms 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


$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n0 $n2 orient left
$ns duplex-link-op $n0 $n3 orient right-up
$ns duplex-link-op $n0 $n4 orient right-down
$ns duplex-link-op $n0 $n5 orient left-up
$ns duplex-link-op $n0 $n6 orient left-down

Agent/Ping instprocrecv {from rtt} {
        $self instvar node_
        puts "node [$node_ id] received ping answer from $from with round-trip-time $rttms"
}
```

```
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 1
$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
```

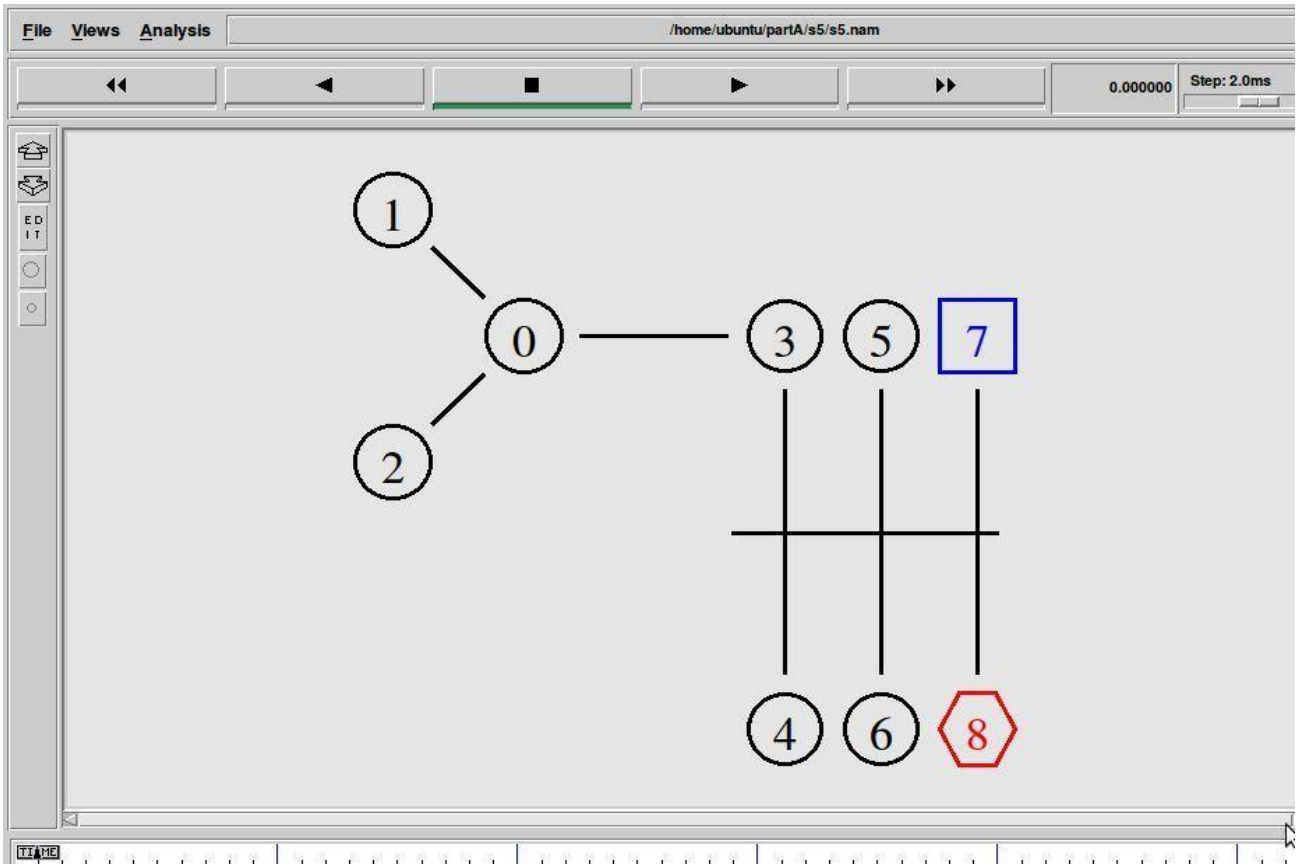
Output

```
#gedit s2.tcl
#sudo ns s2.tcl
```

node 2 received ping answer from 5 with round-trip-time 42.0 ms.
node 3 received ping answer from 6 with round-trip-time 61.6 ms.
node 5 received ping answer from 2 with round-trip-time 42.0 ms.
node 6 received ping answer from 3 with round-trip-time 61.6 ms.

No of packets dropped : 2

3. Simulate an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.



s3.awk

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

s3.tcl

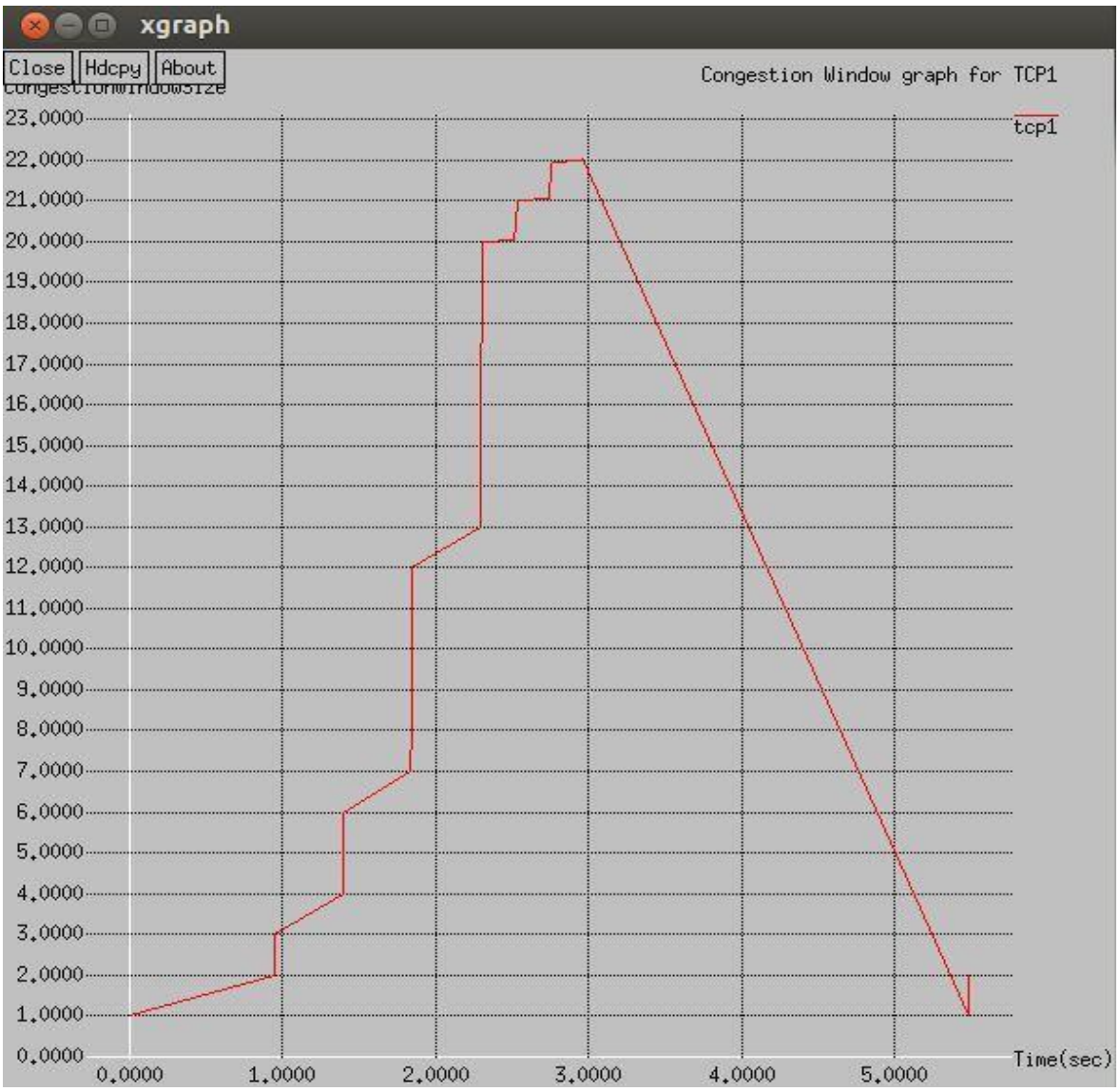
```
set ns [new Simulator]

setnamfile [open s3.nam w]
$ns namtrace-all $namfile

settracefile [open s3.tr w]
$ns trace-all $tracefile

proc finish {} {
        global ns namfiletracefile
        $ns  flush-trace
        close  $namfile
        close $tracefile

        execnam s3.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]

$ns color 1 Blue
$ns color 2 Red

$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

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

$ns connect $tcp1 $sink1

$tcp1 set class_ 1
$tcp1 set packetSize_ 55

set tfile1 [open cwnd1.tr w]
$tcp1 attach $tfile1
$tcp1 trace cwnd_

set tcp2 [new Agent/TCP/Reno]
$ns attach-agent $n2 $tcp2

set sink2 [new Agent/TCPSink]
$ns attach-agent $n8 $sink2

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

$ns connect $tcp2 $sink2

$tcp2 set class_ 2
$tcp2 set packetSize_ 55

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
```

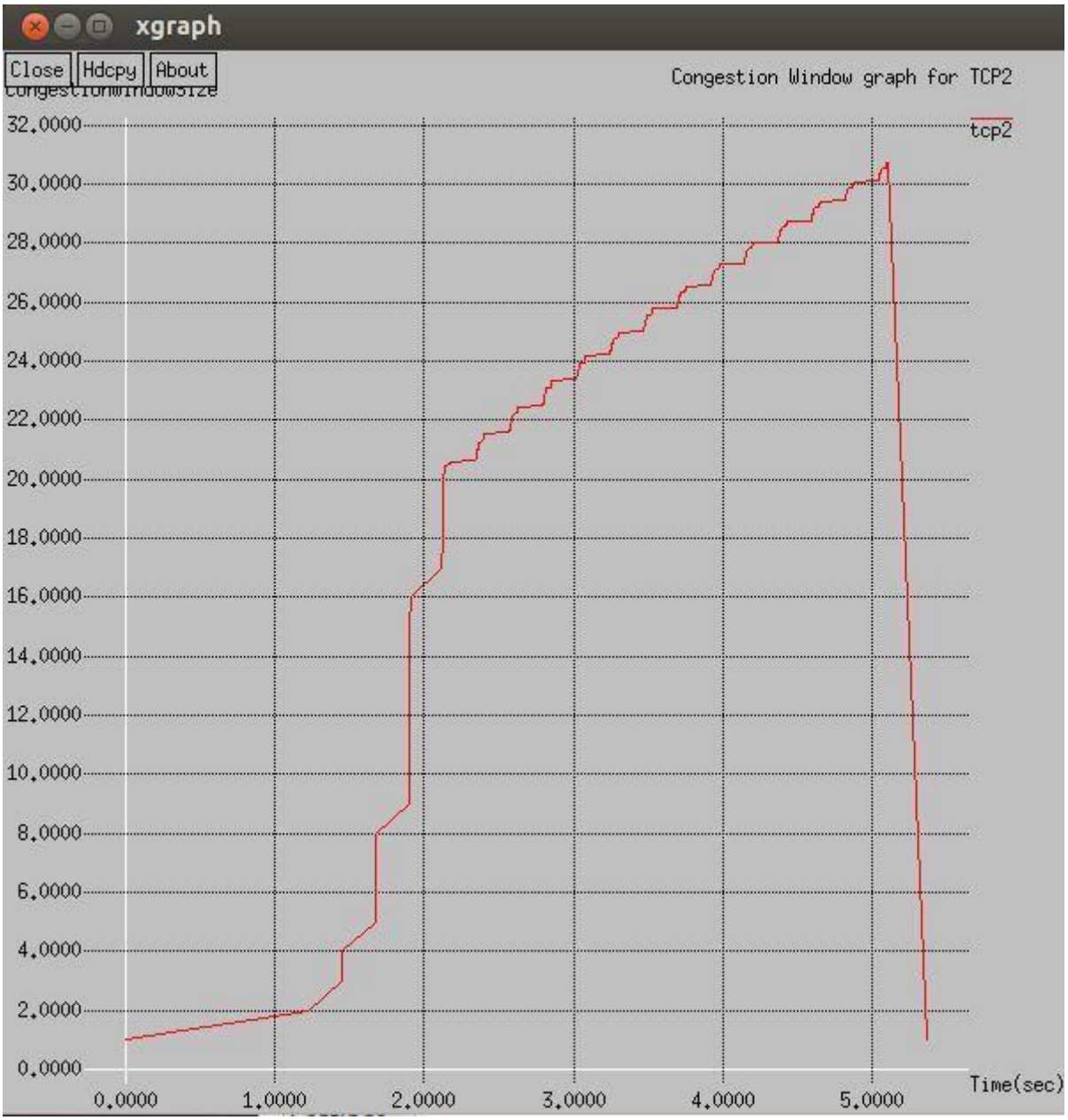
Output

#gedit s3.tcl
#sudo ns s3.tcl

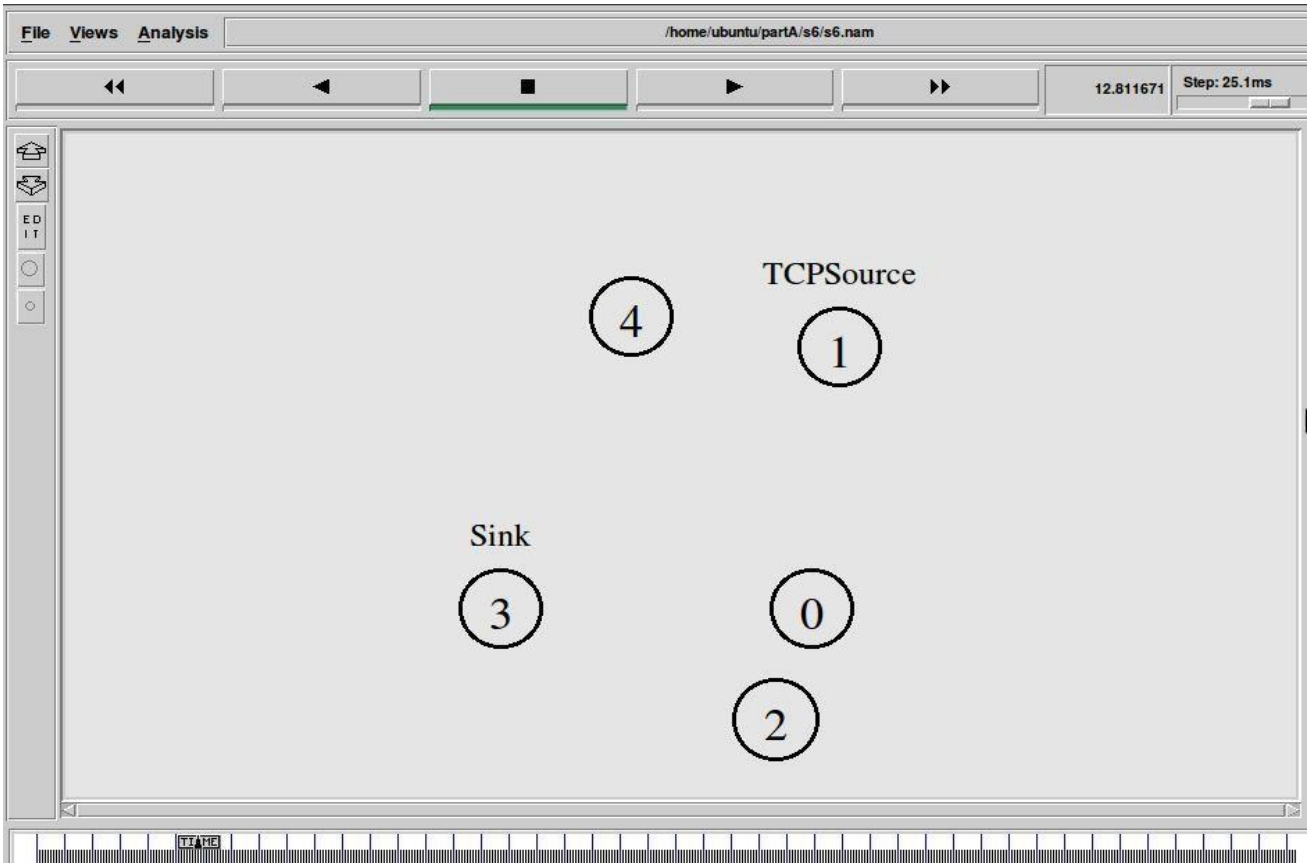
#awk -f s3.awk **cwnd1.tr** >**TCPVegas**
#xgraph -x "Time(sec)" -y "CongestionWindowSize" -t "Congestion Window graph for TCP1" **TCPVegas**



#awk-f s3.awk **cwnd2.tr**>**TCPReno**
#xgraph -x "Time(sec)" -y "CongestionWindowSize" -t "Congestion Window graph for TCP2" **TCPReno**



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



s4.awk

```
BEGIN{
        PacketRcvd = 0;
        Throughput = 0.0;
}
{
        if(($1=="r")&&($3=="_3_")&&($4=="AGT")&&($7=="tcp")&&($8>1000))
        {
                PacketRcvd++;
        }
}
END {
        Throughput=((PacketRcvd*1000*8)/(95.0*1000000));
        printf("\nThe throughput is:%f\n",Throughput);
}
```

s4.tcl

```
if {$argc != 1} {
        error "Command: ns <ScriptName.tcl><Number_of_Nodes>"
        exit 0
}

set ns [new Simulator]


settracefile [open s4.tr w]
$ns trace-all $tracefile

setnamfile [open s4.nam w]
$ns namtrace-all-wireless $namfile 750 750

proc finish {} {
        global ns tracefilenamfile
        $ns flush-trace
        close $tracefile
        close $namfile
        execnam s4.nam &
        execawk -f s4.awk s4.tr &
        exit 0
}

#get the number of nodes value from the user
set val(nn) [lindex $argv 0]

#create new topography object
set topo [new Topography]
$topo load_flatgrid 750 750

#Configure the nodes
$ns     node-config-adhocRoutingAODV \
-llTypeLL \
-macType        Mac/802_11 \
-ifqType        Queue/DropTail \
-channelType Channel/WirelessChannel \
-propType       Propagation/TwoRayGround \
-antType        Antenna/OmniAntenna \
-ifqLen50 \
-phyType        Phy/WirelessPhy \
-topoInstance $topo \
-agentTrace     ON \
-routerTrace    ON \
-macTrace       OFF \
-movementTrace ON

#general operational descriptor storing the hop details in the network
set god_ [create-god $val(nn)]
```

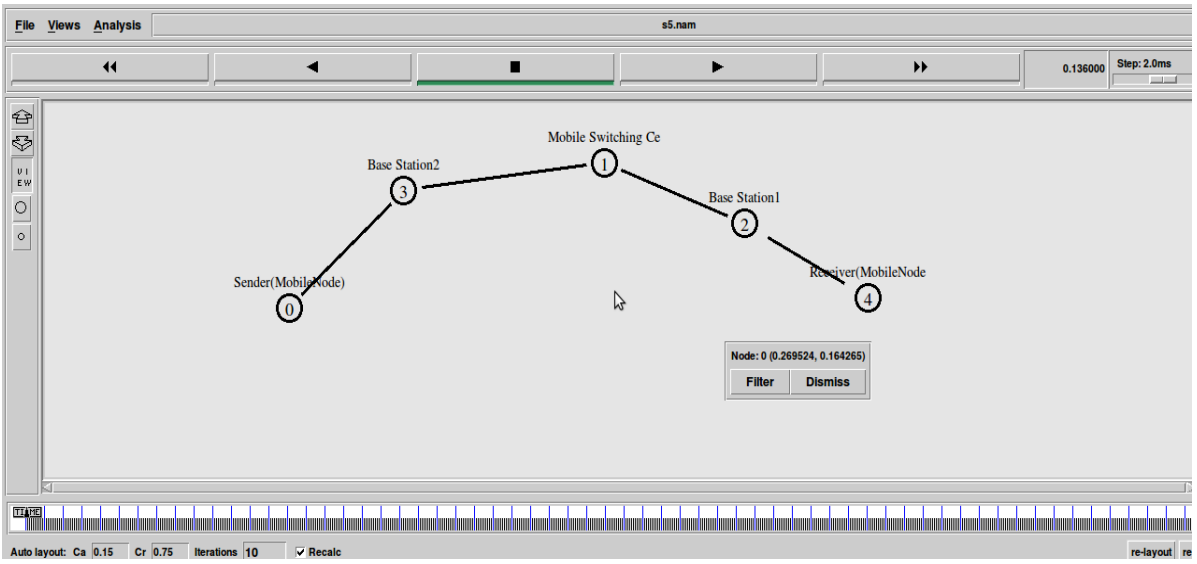
```
#create mobile nodes
for {set i 0} {$i< $val(nn)} {incri} {
        set n($i) [$ns node]
}

#label node
$n(1) label "TCPSource"
$n(3) label "Sink"

#Randomly placing the nodes
for {set i 0} {$i< $val(nn)} {incri} {
        set XX [expr rand()*750]
        set YY [expr rand()*750]
        $n($i) set X_ $XX
        $n($i) set Y_ $YY
}

#define the initial position for the nodes
for {set i 0} {$i< $val(nn)} {incri} {
        $ns initial_node_pos $n($i) 100
}

#define the destination procedure to set the destination to each node
proc destination {} {
        global ns val n
        set now [$ns now]
        set time 5.0
        for {set i 0} {$i< $val(nn)} {incri} {
                set XX [expr rand()*750]
                set YY [expr rand()*750]
                $ns at [expr $now + $time] "$n($i) setdest $XX $YY 20.0"
        }
        $ns at [expr $now + $time] "destination"
}

settcp [new Agent/TCP]
$ns attach-agent $n(1) $tcp

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

set sink [new Agent/TCPSink]
$ns attach-agent $n(3) $sink

$ns connect $tcp $sink

$ns at 0.0 "destination"
$ns at 1.0 "$ftp start"
$ns at 100 "finish"
$ns run
```

Output

#gedit s4.tcl
#sudo ns s4.tcl **5**

The throughput is:0.579368

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



 # General Parameters
set opt(title) zero
set opt(stop) 100 ***;# Stop time.***
set opt(ecn) 0 ***;# Explicit Congestion Notification***
 # Topology
set opt(type) gsm***;#type of link:***
set opt(secondDelay) 55 ***;# average delay of access links in ms***
 # AQM(Active Queue Management) parameters
set opt(minth) 30
set opt(maxth) 0
set opt(adaptive) 1***;# 1 for Adaptive RED, 0 for plain RED***
 # Traffic generation.
set opt(flows) 0 ***;# number of long-lived TCP flows***
set opt(window) 30 ***;# window for long-lived traffic***
set opt(web) 2***;# number of web sessions***
 # Plotting statistics.
set opt(quiet) 0 ***;# popup anything***
set opt(wrap) 100 ***;# wrap plots***
set opt(srcTrace) is ***;# where to plot traffic***
set opt(dstTrace) bs2 ***;# where to plot traffic***
set opt(gsmbuf) 10 ***; # buffer size for gsm***
#default downlink bandwidth in bps
setbwDL(gsm) 9600
#default uplink bandwidth in bps
setbwUL(gsm) 9600
#default downlink propagation delay in seconds
setpropDL(gsm) .500
#default uplink propagation delay in seconds

```
setpropUL(gsm) .500
#default buffer size in packets

setbuf(gsm) 10

set ns [new Simulator]
settracefile [open s5.tr w]
$ns trace-all $tracefile

setnamfile [open s5.nam w]
$ns namtrace-all $namfile

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]

$nodes(lp) label "Receiver(MobileNode)"
$nodes(is) label "Sender(MobileNode)"
$nodes(ms) label "Mobile Switching Center"
$nodes(bs1) label "Base Station1"
$nodes(bs2) label "Base Station2"


proccell_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 "Cell Topology"
}

procset_link_params {t} {
global ns nodes bwULbwDLpropULpropDLbuf
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) simplex
$ns bandwidth $nodes(ms) $nodes(bs1) $bwUL($t) simplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) simplex
$ns bandwidth $nodes(ms) $nodes(bs2) $bwUL($t) simplex
$ns delay $nodes(bs1) $nodes(ms) $propDL($t) simplex
$ns delay $nodes(ms) $nodes(bs1) $propDL($t) simplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) simplex
$ns delay $nodes(ms) $nodes(bs2) $propDL($t) simplex
$ns queue-limit $nodes(bs1) $nodes(ms) $buf($t)
$ns queue-limit $nodes(ms) $nodes(bs1) $buf($t)
$ns queue-limit $nodes(bs2) $nodes(ms) $buf($t)
$ns queue-limit $nodes(ms) $nodes(bs2) $buf($t)
}

# RED and TCP parameters
```

```
Queue/RED set summarystats_ true
Queue/DropTail set summarystats_ true
Queue/RED set adaptive_ $opt(adaptive)

Queue/RED set q_weight_ 0.0
Queue/RED set thresh_ $opt(minth)
Queue/RED set maxthresh_ $opt(maxth)
Queue/DropTail set shrink_drops_ true
Agent/TCP set ecn_ $opt(ecn)
Agent/TCP set window_ $opt(window)
DelayLink set avoidReordering_ true
sourceweb.tcl
```

#Create topology

```
switch $opt(type) {
gsm -
gprs -
umts {cell_topo}
}

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

 # Set up forward TCP connection

```
if {$opt(flows) == 0} {
set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp1 [[set tcp1] attach-app FTP]
$ns at 0.8 "[set ftp1] start"
}
if {$opt(flows) > 0} {
set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp1 [[set tcp1] attach-app FTP]
$tcp1 set window_ 100
$ns at 0.0 "[set ftp1] start"
$ns at 3.5 "[set ftp1] stop"
set tcp2 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp2 [[set tcp2] attach-app FTP]
$tcp2 set window_ 3
$ns at 1.0 "[set ftp2] start"
$ns at 8.0 "[set ftp2] stop"
}

proc stop {} {
global nodes opt nf
set wrap $opt(wrap)
setsid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
if {$opt(srcTrace) == "is"} {
```

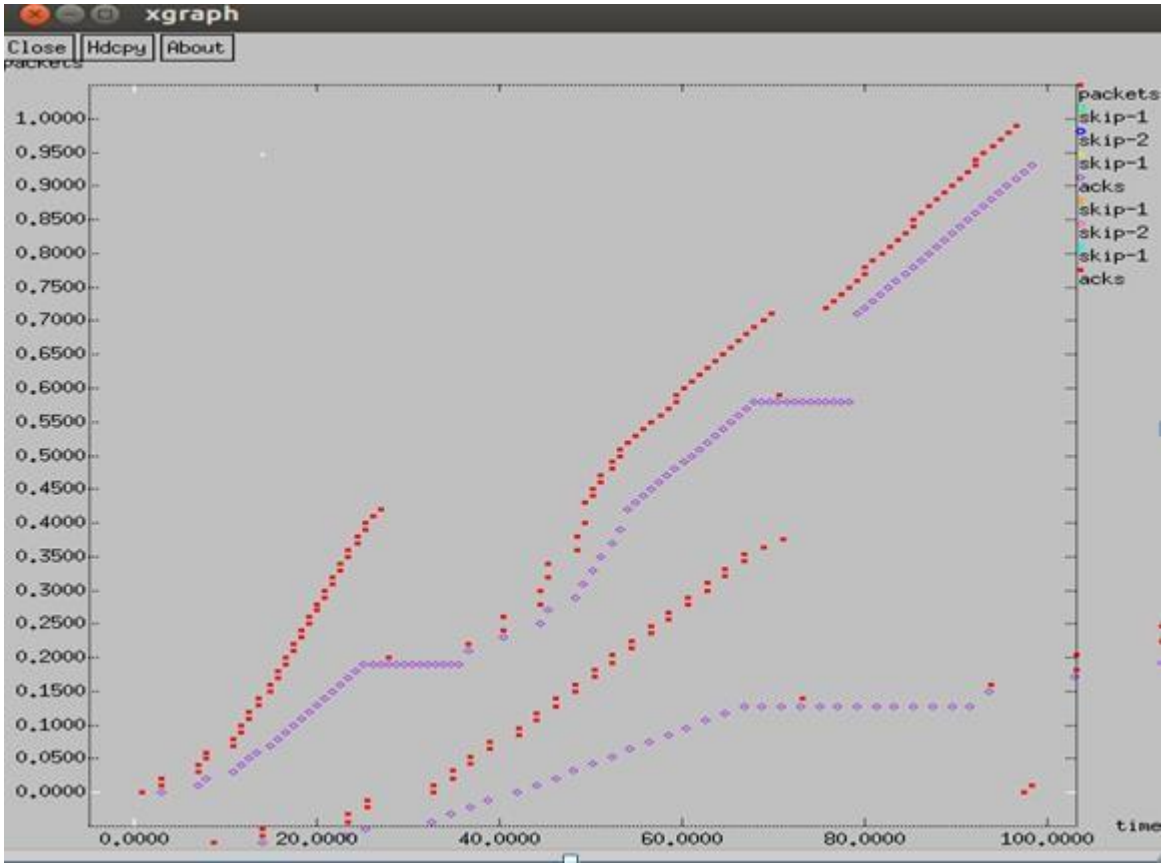
```
set a "-a s5.tr"
} else {
set a "s5.tr"
}

set GETRC "/opt/ns-allinone-2.34/ns-2.34/bin/getrc"
set RAW2XG "/opt/ns-allinone-2.34/ns-2.34/bin/raw2xg"

exec $GETRC -s $sid -d $did -f 0 s5.tr | \
$RAW2XG -s 0.01 -m $wrap -r >plot.xgr
exec $GETRC -s $did -d $sid -f 0 s5.tr | \
$RAW2XG -a -s 0.01 -m $wrap >>plot.xgr
exec $GETRC -s $sid -d $did -f 1 s5.tr | \
$RAW2XG -s 0.01 -m $wrap -r >>plot.xgr
exec $GETRC -s $did -d $sid -f 1 s5.tr | \
$RAW2XG -s 0.01 -m $wrap -a >>plot.xgr

execnam s5.nam &

exec ./xg2gp.awk plot.xgr
if {!$opt(quiet)} {
execxgraph -bb -tk -nl -m -x time -y packets plot.xgr&
}
exit 0
}

$ns at $opt(stop) "stop"
$ns run
```

Execution Procedure:

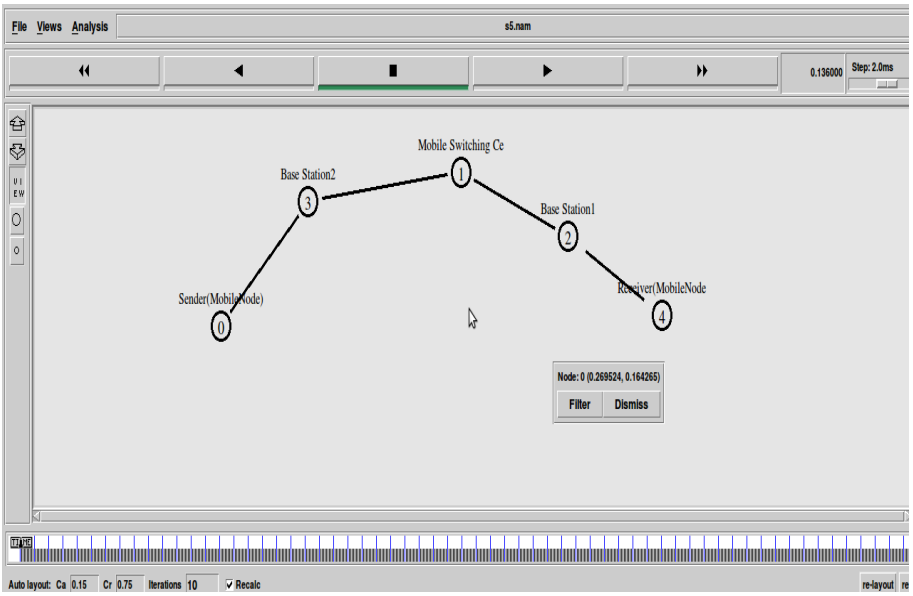
Copy the files web.tcl and xg2gp.awk from the path /opt/ns-allinone-2.34/ns-2.34/tcl/ex/wireless-scripts to the working directory(i.e where source code of the program is present).

Output:

#gedit s5.tcl
#sudo ns s5.tcl



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



 # General Parameters
set opt(title) zero;
set opt(stop) 100***;# Stop time.***
set opt(ecn) 0;
 # Topology
set opt(type) umts***;#type of link:***
set opt(secondDelay) 55***;# average delay of access links in ms***
 # AQM parameters
set opt(minth) 30;
set opt(maxth) 0;
set opt(adaptive) 1***;# 1 for Adaptive RED, 0 for plain RED***
 # Traffic generation.
set opt(flows) 0***;# number of long-lived TCP flows***
set opt(window) 30 ***;# window for long-lived traffic***
set opt(web) 2***;# number of web sessions***
 # Plotting statistics.
set opt(quiet) 0***;# popup anything***
set opt(wrap) 100 ***;# wrap plots***
set opt(srcTrace) is ***;# where to plot traffic***
set opt(dstTrace) bs2 ***;# where to plot traffic***
set opt(umtsbuf) 10***; # buffer size for umts***
#default downlink bandwidth in bps
setbwDL(umts) 384000
#default uplink bandwidth in bps
setbwUL(umts) 64000
#default downlink propagation delay in seconds
setpropDL(umts) .150

#default uplink propagation delay in seconds
setpropUL(umts) .150

#default buffer size in packets
setbuf(umts) 20

```
set ns [new Simulator]
settracefile [open s6.tr w]
$ns trace-all $tracefile

setnamfile [open s6.nam w]
$ns namtrace-all $namfile

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]

proccell_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 "Cell Topology"
}

procset_link_params {t} {
global ns nodes bwULbwDLpropULpropDLbuf
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) simplex
$ns bandwidth $nodes(ms) $nodes(bs1) $bwUL($t) simplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) simplex
$ns bandwidth $nodes(ms) $nodes(bs2) $bwUL($t) simplex
$ns delay $nodes(bs1) $nodes(ms) $propDL($t) simplex
$ns delay $nodes(ms) $nodes(bs1) $propDL($t) simplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) simplex
$ns delay $nodes(ms) $nodes(bs2) $propDL($t) simplex
$ns queue-limit $nodes(bs1) $nodes(ms) $buf($t)
$ns queue-limit $nodes(ms) $nodes(bs1) $buf($t)
$ns queue-limit $nodes(bs2) $nodes(ms) $buf($t)
$ns queue-limit $nodes(ms) $nodes(bs2) $buf($t)
}
```

 # RED and TCP parameters

```
Queue/RED set summarystats_ true
Queue/DropTail set summarystats_ true
Queue/RED set adaptive_ $opt(adaptive)
Queue/RED set q_weight_ 0.0
Queue/RED set thresh_ $opt(minth)
Queue/RED set maxthresh_ $opt(maxth)
```

```
Queue/DropTail set shrink_drops_ true
Agent/TCP set ecn_ $opt(ecn)
Agent/TCP set window_ $opt(window)

DelayLink set avoidReordering_ true
```

sourceweb.tcl

```
#Create topology
switch $opt(type) {
umts {cell_topo}
}

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


# Set up forward TCP connection
if {$opt(flows) == 0} {
set tcp1 [$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"
}

if {$opt(flows) > 0} {
set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp1 [[set tcp1] attach-app FTP]
$tcp1 set window_ 100
$ns at 0.0 "[set ftp1] start"
$ns at 3.5 "[set ftp1] stop"
set tcp2 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp2 [[set tcp2] attach-app FTP]
$tcp2 set window_ 3
$ns at 1.0 "[set ftp2] start"
$ns at 8.0 "[set ftp2] stop"
}

proc stop {} {
global nodes opt nf
set wrap $opt(wrap)
setsid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
if {$opt(srcTrace) == "is"} {
set a "-a s6.tr"
} else {
set a "s6.tr"
}
```

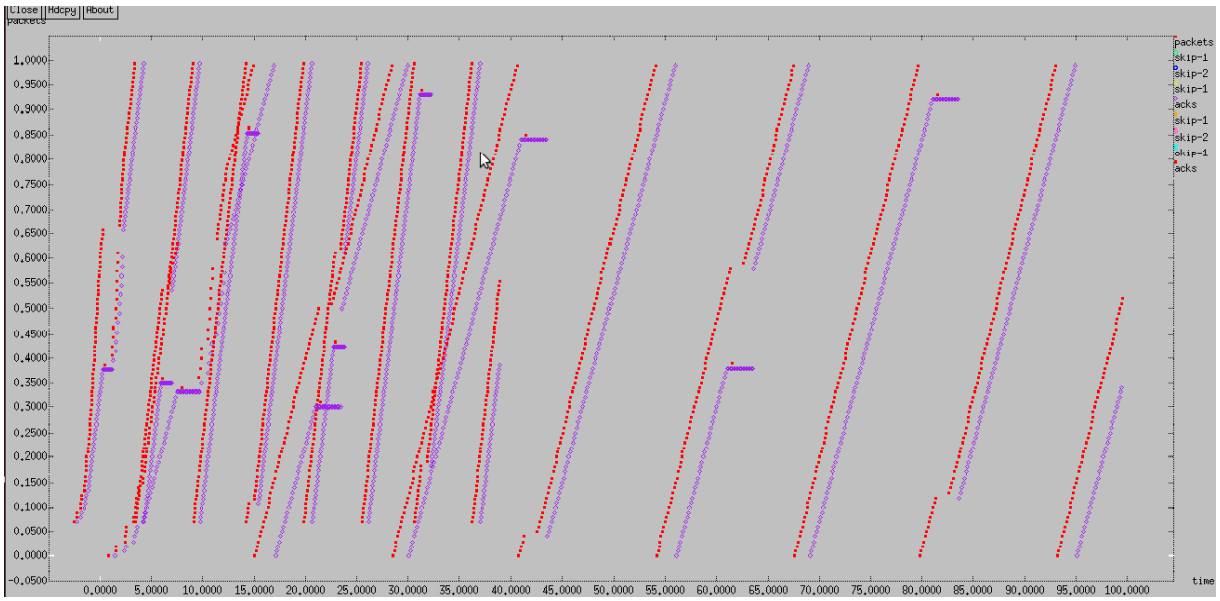
```
set GETRC "/opt/ns-allinone-2.34/ns-2.34/bin/getrc"
set RAW2XG "/opt/ns-allinone-2.34/ns-2.34/bin/raw2xg"

exec $GETRC -s $sid -d $did -f 0 s6.tr | \
$RAW2XG -s 0.01 -m $wrap -r >plot.xgr
exec $GETRC -s $did -d $sid -f 0 s6.tr | \
$RAW2XG -a -s 0.01 -m $wrap >>plot.xgr
exec $GETRC -s $sid -d $did -f 1 s6.tr | \
$RAW2XG -s 0.01 -m $wrap -r >>plot.xgr
exec $GETRC -s $did -d $sid -f 1 s6.tr | \
$RAW2XG -s 0.01 -m $wrap -a >>plot.xgr

execnam s6.nam &

exec ./xg2gp.awk plot.xgr
if {!$opt(quiet)} {
execxgraph -bb -tk -nl -m -x time -y packets plot.xgr&
}
exit 0
}
$ns at $opt(stop) "stop"
$ns run
```

Execution Procedure:

Copy the files web.tcl and xg2gp.awk from the path /opt/ns-allinone-2.34/ns-2.34/tcl/ex/wireless-scripts to the working directory(i.e where source code of the program is present).

Output:

#gedit s6.tcl
#sudo ns s6.tcl



7. Write a program for error detecting code using CRC-CCITT (16- bits).

```
importjava.util.Scanner;
public class CRCDemo
{
static String msg;
static String genPoly = "10001000000100001";
//static String genPoly = "1101";
static char t[] = new char[128]; /* Array for message appended with checksum */
static char cs[] = new char[128]; /* Temporary array for checksum calculation */
staticchar g[] = new char[128]; /* Array to store generator polynomial */
staticintmlen,glen,x,c,flag=0,test;

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

  System.out.println("Enter the message to be transferred");
  msg = in.nextLine();
  mlen = msg.length(); /* Length of the original message */

  for(inti=0;i<mlen;i++)
  t[i] = msg.charAt(i);


  System.out.println("Predefined Generator Polynomial is: " + genPoly);
  g = genPoly.toCharArray();

  glen = genPoly.length(); /* Length of Generator Polynomial */

  for(x=mlen;x<(mlen+glen-1);x++)
  t[x] = '0';

  System.out.println("Zero extended message is: "+ new String(t));
  crc(); /* Checksum computation */

  System.out.println("CheckSum is:" + new String(cs));

  /* Subtract the checksum from zero extended message which means
  we do simple "XORing" */

  for(x=mlen;x<mlen+glen-1;x++)
  t[x] = cs[x-mlen];

  System.out.println("Final codeword generated is:"+ new String(t));
  System.out.println("\n\nTest Error detection 1(yes) 0(no) ? : ");
  test = in.nextInt();
```

```
if(test==1)
{
System.out.println("Enter position where error is to inserted : ");
  x = in.nextInt();
t[x] = (t[x]=='0')?'1':'0';
System.out.println("Errorneous data :"+ new String(t));
}

crc(); // computation at receiver side

for(x=0;x<(glen-1); x++)
{
if(cs[x] =='1')
        {
flag =1;
break;
        }
}
if(flag==1)
System.out.println("Error was detected during transfer");
else
System.out.println("No Error Detected during transfer");

 }
public static void crc()
  {
   for(x=0; x<glen; x++)
   cs[x] = t[x];
   do
      {
   if(cs[0] == '1')
   xor();

   for(c=0; c<glen-1; c++)
          {
   cs[c] = cs[c+1];
          }
   cs[c] = t[x++];
   }while(x<=mlen+glen-1);
   }

   public static void xor()
    {
   for(c=1; c<glen; c++)
   cs[c] = ((cs[c]==g[c])? '0' : '1');
    }
}
```

Output:

Run 1:

Enter the message to be transferred
10011
Predefined Generator Polynomial is: 10001000000100001
Zero extended message is: 10011000000000000000
CheckSum is:0010001001010010
Final codeword generated is:10011000100010010010010

Test Error detection 1(yes) 0(no) ? :
1
Enter position where error is to inserted :
4
Erroneousdata :10010001000100101001 0
Error was detected during transfer

Run 2:

Enter the message to be transferred
10011
Predefined Generator Polynomial is: 10001000000100001
Zero extended message is: 10011000000000000000
CheckSum is:0010001001010010
Final codeword generated is:10011000100010010010010

Test Error detection 1(yes) 0(no) ? :
0
No Error Detected during transfer

8.Write a program to find the shortest path between vertices using bellman-ford algorithm.

```
importjava.util.*;

public class BellmanDemoFinal
{
  static Scanner in = new Scanner(System.in);
  public static void main(String [] args)
    {
  int V,E=1,chckNegative=0;
  int w[][] = new int [20][20];
  int edge[][] = new int [50][2];

  /* Read the no of vertices in the graph */
  System.out.println("Enter the no of vertices ");
      V = in.nextInt();

  System.out.println("Enter the weight matrix");
  for(inti=1;i<=V;i++)
  for(int j=1;j<=V;j++)
        {
  w[i][j] = in.nextInt();
  if(w[i][j]!=0)
            {  edge[E][0]=i;
  edge[E++][1]=j;
            }
        }

  chckNegative = bellmanFord(w,V,E,edge);

  if(chckNegative == 1)
  System.out.println("\nNo negative weight cycle\n");
  else
  System.out.println("\nNegative weight cycle exists\n");
  }

public static intbellmanFord(int w[][],intV,intE,int edge[][])
  {
  intu,v,S,flag=1;
  int distance [] = new int[20];
  int parent [] = new int [20];
  /* Assign the distance of all the vertices to 999 */
  for(inti=1;i<=V;i++)
     {
```

```
 distance[i] = 999;
 parent[i]=-1;
    }

 System.out.println("Enter the source vertex");
    S = in.nextInt();

 /* Assign the distance of source vertex to 999 */
 distance[S]=0;
 /* Relax each edge for V-1 times */
 for(inti=1;i<=V-1;i++)
    {
 for(int k=1;k<=E;k++)
       {
          u = edge[k][0];
          v = edge[k][1] ;
 /* Relaxing each edge */
 if(distance[u]+w[u][v] < distance[v])
          {
 distance[v] = distance[u] + w[u][v] ;
 parent[v]=u ;
          }
      }
   }

/* Relax all the edges one more time to check for negative weight cycle */
for(int k=1;k<=E;k++)
{
          u = edge[k][0] ;
          v = edge[k][1] ;
if(distance[u]+w[u][v] < distance[v])
flag = 0 ;
    }
if(flag==1)
for(inti=1;i<=V;i++)
System.out.println("Vertex " + i + " -> cost = " + distance[i] + " parent = "+ (parent[i]));

return flag;

  }
}
```

Output:

```
Enter the no of vertices
6
Enter the weight matrix
0 3 2 5 999 999
3 0 999 1 4 999
2 999 0 2 999 1
5 1 2 0 3 999
999 4 999 3 0 2
999 999 1 999 2 999
Enter the source vertex
1
Vertex 1 -> cost = 0 parent = -1
Vertex 2 -> cost = 3 parent = 1
Vertex 3 -> cost = 2 parent = 1
Vertex 4 -> cost = 4 parent = 2
Vertex 5 -> cost = 5 parent = 6
Vertex 6 -> cost = 3 parent = 3
No negative weight cycle

Enter the no of vertices
5
Enter the weight matrix
0 -1 4 999 999
999 0 3 2 2
999 999 0 999 999
999 1 5 0 999
999 999 999 -3 0
Enter the source vertex
1
Vertex 1 -> cost = 0 parent = -1
Vertex 2 -> cost = -1 parent = 1
Vertex 3 -> cost = 2 parent = 2
Vertex 4 -> cost = -2 parent = 5
Vertex 5 -> cost = 1 parent = 2
No negative weight cycle

Enter the no of vertices
5
Enter the weight matrix
0 -1 4 999 999
999 0 3 2 2
999 999 0 999 999
999 1 5 0 999
999 999 999 -5 0
Enter the source vertex
1
Negative weight cycle exists
```

9. Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.

Client Side Program

```
import java.net.*;
import java.io.*;
importjava.util.*;
public class SocketClient
{
public static void main( String args[ ] ) throws Exception
  {
    Scanner in = new Scanner(System.in);
/* Create socket at client side */
    Socket clientSocket = new Socket( "127.0.0.1",4000);
System.out.println("****Client side****");
/* Reading the filename from keyboard. */
System.out.println("Enter the file name to transfer");
    String fname = in.nextLine();
/* Sending the filename to server. Uses PrintWriter to write filename to outputstream  */
OutputStreamostream = clientSocket.getOutputStream( );
PrintWriterpwrite = new PrintWriter(ostream, true);
pwrite.println(fname);

/* Receiving the contents from server.Uses input stream */
InputStreamistream = clientSocket.getInputStream();
BufferedReadersocketRead = new BufferedReader(new InputStreamReader(istream));
System.out.println("Contents of the file " + fname + " are");
    String str;
while((str = socketRead.readLine()) != null)
     /* Reading line-by-line */
    {
System.out.println(str);
    }
pwrite.close();
socketRead.close();
  }
}
```

Server Side Program

```
import java.net.*;
import java.io.*;
public class SocketServer
{
public static void main(String args[]) throws Exception
  {
/* Create a socket at server side */
```

```
ServerSocketservSocket = new ServerSocket(4000);
System.out.println("****Server Side****");

System.out.println("Server ready for connection");
/* Accept the connection with port: 4000 */
    Socket connSock = servSocket.accept();

System.out.println("Connection is successful and ready for file transfer");
/* Reading the filename from client using connection socket */
InputStreamistream = connSock.getInputStream( );
BufferedReaderfileRead =new BufferedReader(new InputStreamReader(istream));
    String fname = fileRead.readLine();
    File fileName = new File(fname);

/* Keeping output stream ready to send the contents */
OutputStreamostream = connSock.getOutputStream( );
PrintWriterpwrite = new PrintWriter(ostream, true);
/* If file exists read the contents of file and send to the client */
if(fileName.exists())
    {
BufferedReadercontentRead = new BufferedReader(new FileReader(fname));
System.out.println("Writing file Contents to the socket");
    String str;
while((str = contentRead.readLine()) != null)
    /* Reading line-by-line from file */
    {
/* Sending each line to the client */
pwrite.println(str);
    }
contentRead.close();
    }
else
    {
System.out.println("Requested file does not exist");
        String msg = "Requested file does not exist at server side";
pwrite.println(msg);
    }

connSock.close();
servSocket.close();       /* Close network sockets */
fileRead.close();
pwrite.close();
  }
}
```

Output:

Client Side	Server Side
****Client side**** Enter the file name to transfer name.txt Contents of the file name.txt are SMVITM NITK IIT Madras Stanford University	****Server Side**** Server ready for connection Connection is successful and ready for file transfer Writing file Contents to the socket

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

Client Side Program

```
import java.net.*;
public class DatagramSocketClient
{
public static void main(String [] args) throws Exception
   {
     String line = "Connected with Client";
     /* Create new datagram socket at client side */
DatagramSocketclientSocket = new DatagramSocket();
     /* Get IPAddress using the InetAddress class */
InetAddressIPAddress = InetAddress.getByName("localhost");

byte[] sendData = new byte[1024];
byte[] receiveData = new byte[1024];

sendData = line.getBytes();

/* Create the send datagram packet */
DatagramPacketsendPacket = new DatagramPacket(sendData, sendData.length, IPAddress,
9000);
/* Send the packet using client socket */
clientSocket.send(sendPacket);

System.out.println("*****Client Display Terminal****");
while(true)
     {
/* Create the receive datagram packet */
DatagramPacketreceivePacket = new DatagramPacket(receiveData, receiveData.length);
/*Receive the packet using client socket */
clientSocket.receive(receivePacket);
/* Convert the message received into the string */
String messageReceived = new
String(receivePacket.getData(),receivePacket.getOffset(),receivePacket.getLength());
System.out.println("Message typed at server side is : " + messageReceived);
     }
   }
}
```

Server Side Program

```
import java.net.*;
importjava.util.*;

public class DatagramSocketServer
{
public static void main(String [] args) throws Exception
  {
     Scanner in = new Scanner(System.in);
    DatagramSocketserverSocket = new DatagramSocket(9000
byte[] receiveData = new byte[1024];
byte[] sendData = new byte[1024];
System.out.println("***Server Side***");
/* Create the receive datagram packet */
DatagramPacketreceivePacket = new DatagramPacket(receiveData, receiveData.length);
serverSocket.receive(receivePacket); /* Receive the packet using server socket */
System.out.println(new String(receivePacket.getData()));
InetAddressIPAddress = receivePacket.getAddress();
int port = receivePacket.getPort();
while(true)
    {
System.out.println("Type some message to display at client end");
      String message = in.nextLine();
sendData = message.getBytes();
System.out.println("Message sent from the server:" + new String(sendData));
DatagramPacketsendPacket = new DatagramPacket(sendData,sendData.length, IPAddress,
port);
/* Send the packet using server socket */
serverSocket.send(sendPacket);
    }
  }
}
```

Output:

Client Side	Server Side
*******Client Display Terminal******	*****Server Side*****
Message typed at server side is :	**Connected with Client**
hello how are you	
Message typed at server side is :	**Type some message to display at client end**
I am studying at SMVITM Bantakal	**hello how are you**
	Message sent from the server:
	hello how are you
	Type some message to display at client end
	I am studying at SMVITM Bantakal
	Message sent from the server:
	I am studying at SMVITM Bantakal

11. Write a program for simple RSA algorithm to encrypt and decrypt the data.

```
import java.util.*;
public class RSADemo
{
  public static void main(String [] args)
  {
     String msg;

     int pt [] = new int[100];
     int ct [] = new int[100];
     int z, n, d, e, p, q,mlen;

     Scanner in = new Scanner(System.in);
     do
     {
       System.out.println("Enter the two large prime numbers for p and q");
       p = in.nextInt();
       q = in.nextInt();
     }while( prime(p)==0 || prime(q)==0);

     n = p*q; // Calculate the n value
     z=(p-1)*(q-1); // Calculate the z value

     System.out.println("Value of n " + n + "\n Value of z is :"+ z);

  // Key generation (Encryption key)
     for(e=2;e<z;e++)
     {
        if(gcd(e,z)==1)
          break;
     }
    System.out.println("Encryption key e is " + e);
    System.out.println("Public key is (e, n) : " + e +","+ n);

  // Key generation(Decryption key)
    for(d=2;d<z;d++)
    {
                if((e*d)%z==1)
                        break;
    }
    System.out.println("Decryption key d is : " + d);
    System.out.println("Private key is (d, n) => "+ d +"," +n);

    in.nextLine(); // To avoid the new line character in input buffer

    System.out.println("Enter the message for encryption ");
    msg = in.nextLine();
```

```
    mlen = msg.length();
    for(int i=0;i<mlen;i++)
       pt[i] = msg.charAt(i); // Extract the individual characters from string
    System.out.println("ASCII Values of PT array is");
    for(int i=0;i<mlen;i++)
       System.out.println(pt[i]);

      System.out.println(" Encryption: Cipher Text Obtained : ");
        for(int i=0; i<mlen; i++)
        ct[i] = mult(pt[i], e, n);
        for(int i=0; i<mlen; i++)
        System.out.print( ct[i] + "\t");

      System.out.println("\nDecryption: Plain Text Obtained: ");
        for(int i=0; i<mlen; i++)
                pt[i] = mult(ct[i], d, n) ;
        for(int i=0; i<mlen; i++)
        {
        System.out.println(pt[i] + ":" + (char)pt[i]);
        }
  }
  // Method to calculate the GCD of two numbers
   public static int gcd(int x, int y)
   {
     if(y == 0)
        return x;
     else
        return gcd(y, x%y);
   }
  // Method to check a number prime or not
   public static int prime(int num)
  {
        int i;
        for(i=2; i<=num/2; i++)
        {
                if(num%i == 0)
                        return 0;
        }
        return 1;
}
// Method for encryption and decryption of specific values
  public static int mult(int base, int exp, int n)
  {
         int res=1, j;
        for (j=1; j<=exp; j++)
        res = ((res * base) % n);
        return res;
  }
} // end of class
```

Output

Run1:
Enter the two large prime numbers for p and q
19
23
Value of n 437
Value of z is :396
Encryption key e is 5
Public key is (e, n) : 5,437
Decryption key d is : 317
Private key is (d, n) => 317,437
Enter the message for encryption
smvitm bantakal

ASCII Values of PT array is
115
109
118
105
116
109
32
98
97
110
116
97
107
97
108

Encryption: Cipher Text Obtained :
115 67 36 326 70 67 261 186 89 325 70 89 122
89 52

Decryption: Plain Text Obtained:
115:s
109:m
118:v
105:i
116:t
109:m
32:
98:b
97:a
110:n
116:t
97:a
107:k
97:a

108:l

Run2:

Enter the two large prime numbers for p and q
23
29
Value of n 667
Value of z is :616
Encryption key e is 3
Public key is (e, n) : 3,667
Decryption key d is : 411
Private key is (d, n) => 411,667
Enter the message for encryption
!@#$%
ASCII Values of PT array is
33
64
35
36
37
Encryption: Cipher Text Obtained :
586 13 187 633 628
Decryption: Plain Text Obtained:
33:!
64:@
35:#
36:$
37:%

12. Write a program for congestion control using leaky bucket algorithm.

```
import java.util.Random;
import java.util.Scanner;

public class LeakyDemo
{
public static void main(String [] args)
{
  int bcktsize,iter, oprate, flow, total=0, rem_bytes;
  int pkt[] = new int[25];

  Scanner in = new Scanner(System.in);

     System.out.println("Enter the no of input flows");
     flow = in.nextInt();

     System.out.println("Enter the no of iterations");
     iter = in.nextInt();

     System.out.println("Enter the bucket size and output rate :");
     bcktsize = in.nextInt();
     oprate = in.nextInt();

   Random rand = new Random();
  for(int i=0;i<iter;i++)
    {
     System.out.println("Iteration: " + (i+1));
     for(int j=0;j<flow;j++)
     {
       rand.setSeed(System.nanoTime()); // To generate packets of unique size
       pkt[j] = rand.nextInt(500); // Unique random number is assigned as packet size
       total += pkt[j];

       if(total<=bcktsize) // Check whether packetsize is less than bucket Size
       {
         System.out.println(" Input from flow " + (j+1) + " with packet size " + pkt[j] +" bytes are
accepted for the bucket Number of bytes in bucket is : "+ total);
       }

       else
       {
             total-=pkt[j];
             System.out.println(" Input from flow " + (j+1) + " with packet size " + pkt[j]+" bytes
are discarded from the bucket Number of bytes in bucket is: " + total);
       }
     }
       if(oprate> total) // check whether the output rate exceeds the bucket content
```

```
{
          rem_bytes = total;
          total = 0;
          System.out.println("..........");
          System.out.println("Output rate of the bucket is: "+ oprate + " \n Bytes sent out of the
bucket is " + rem_bytes);
          System.out.println("..........");
       }
       else
       {
          total -= oprate;
          System.out.println("..........");
          System.out.println(" Output rate of the bucket is " + oprate +" \n Number of bytes
           remaining in bucket is" +total);
          System.out.println("---------------------------------");
       }
    }

}
}
```

Output :

Run1:

Enter the no of input flows
3
Enter the no of iterations
2
Enter the bucket size and output rate :
500
100
Iteration: 1
Input from flow 1 with packet size 77 bytes are accepted for the bucket Number of bytes in bucket is : 77
Input from flow 2 with packet size 207 bytes are accepted for the bucket Number of bytes in bucket is : 284
Input from flow 3 with packet size 208 bytes are accepted for the bucket Number of bytes in bucket is : 492

Output rate of the bucket is 100
Number of bytes remaining in bucket is 392

Iteration: 2
Input from flow 1 with packet size 257 bytes are discarded from the bucket Number of bytes in bucket is: 392
Input from flow 2 with packet size 118 bytes are discarded from the bucket Number of bytes in bucket is: 392

Input from flow 3 with packet size 307 bytes are discarded from the bucket Number of bytes in bucket is: 392

Output rate of the bucket is 100
Number of bytes remaining in bucket is 292

Run2:

Enter the no of input flows
3
Enter the no of iterations
3
Enter the bucket size and output rate :
500
350
Iteration: 1
Input from flow 1 with packet size 65 bytes are accepted for the bucket Number of bytes in bucket is : 65
Input from flow 2 with packet size 230 bytes are accepted for the bucket Number of bytes in bucket is : 295
Input from flow 3 with packet size 452 bytes are discarded from the bucket Number of bytes in bucket is: 295

Output rate of the bucket is: 350
Bytes sent out of the bucket is 295

Iteration: 2
Input from flow 1 with packet size 336 bytes are accepted for the bucket Number of bytes in bucket is : 336
Input from flow 2 with packet size 193 bytes are discarded from the bucket Number of bytes in bucket is: 336
Input from flow 3 with packet size 357 bytes are discarded from the bucket Number of bytes in bucket is: 336

Output rate of the bucket is: 350
Bytes sent out of the bucket is 336

Iteration: 3
Input from flow 1 with packet size 454 bytes are accepted for the bucket Number of bytes in bucket is : 454
Input from flow 2 with packet size 260 bytes are discarded from the bucket Number of bytes in bucket is: 454
Input from flow 3 with packet size 369 bytes are discarded from the bucket Number of bytes in bucket is: 454

Output rate of the bucket is 350
Number of bytes remaining in bucket is 104