VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnanasangama", Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V.Puram, Bangaluru-560 004

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Laboratory Report

15CSL57

"COMPUTER NETWORK LABORATORY"

Submitted By

KARTIK G
1BI15CS068

for the academic year 2017-18

Department of Computer Science & Engineering
Bangalore Institute of Technology
K.R. Road, V.V.Puram, Bangaluru-560 004

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnanasangama", Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V.Puram, Bangaluru-560 004

Department of Computer Science & Engineering

Certificate

This is to certify that the implementation of **Laboratory Report** entitled "Computer Network Laboratory" has been successfully completed by

USN: 1BI15CS068 NAME : KARTIK G

of 5TH semester B.E. for the partial fulfillment of the requirements for the Bachelor's degree in Computer Science & Engineering of the Visvesvaraya Technological University during the academic year 2017-2018.

Lab Incharges :

Prof. Girija J
Associate Professor
Dept. of CS&E
Bangalore Institute of Technology
Bangalore

Prof. Suma L
Assistant Professor
Dept. of CS&E
Bangalore Institute of Technology
Bangalore

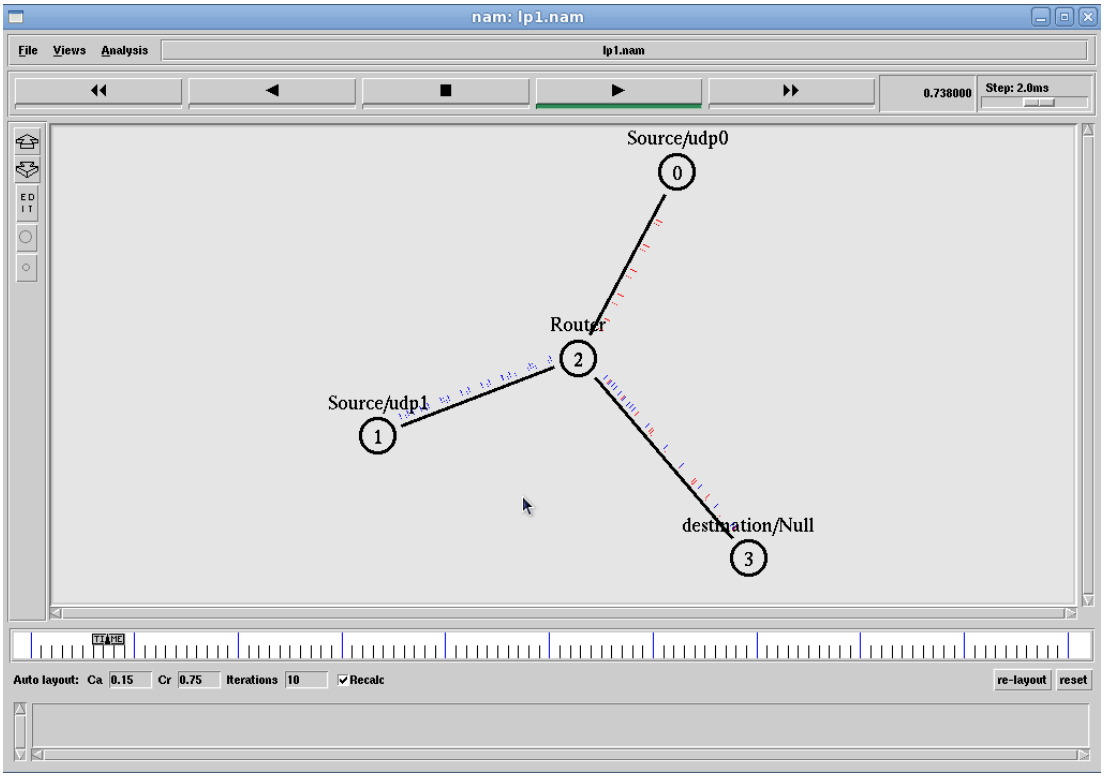
Contents

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.

```
# TCL script to implement three-nodes point-point network with duplex links between
#them.set the queue size, vary the bandwidth and find the number of packets dropped
set ns [ new Simulator]
set nt [ open lab1.tr w]
$ns trace-all $nt
set nf [ open lab1.nam w]
$ns namtrace-all $nf
# create the nodes.
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
#Assign color to the packets.
$ns color 1 Red
$ns color 2 Blue
#Label the nodes
$n0 label "Source/udp0"
$n1 label "Source/udp1"
$n2 label "Router"
$n3 label "Destination/Null"
# Create links, vary bandwidth to check the number of packets dropped.
$ns duplex-link $n0 $n2 10Mb 300ms DropTail
$ns duplex-link $n1 $n2 10Mb 300ms DropTail
$ns duplex-link $n2 $n3 1Mb 300ms DropTail
#The below code is used to set the queue size b/w the nodes
$ns set queue-limit $n0 $n2 10
$ns set queue-limit $n1 $n2 10
$ns set queue-limit $n2 $n3 5
#Create and attach UDP agent to n0,n1 and null agent to n3.
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
set null3 [new Agent/Null]
$ns attach-agent $n3 $null3
#Set udp0 packets to red and udp1 packets to blue color
```

```
$udp0 set class_ 1
$udp1 set class_ 2
#Connect the agents.
$ns connect $udp0 $null3
$ns connect $udp1 $null3
#Set the packet size to 500
$cbr1 set packetSize_ 500Mb
#set the Data rate of the packets. if the data rate is high #then packets drops
are high.
$cbr1 set interval_ 0.005
#Finish Procedure
proc finish { } {
global ns nf nt
$ns flush-trace
exec nam lab1.nam &
close $nt
close $nf
exit 0
}
$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr1 start"
$ns at 10.0 "finish"
$ns run
```

TOPOLOGY:



OUTPUT:

```
The number of packets dropped = 7002
The number of packets received = 29924
The throughput is: 1.23393
```

AWK Script:

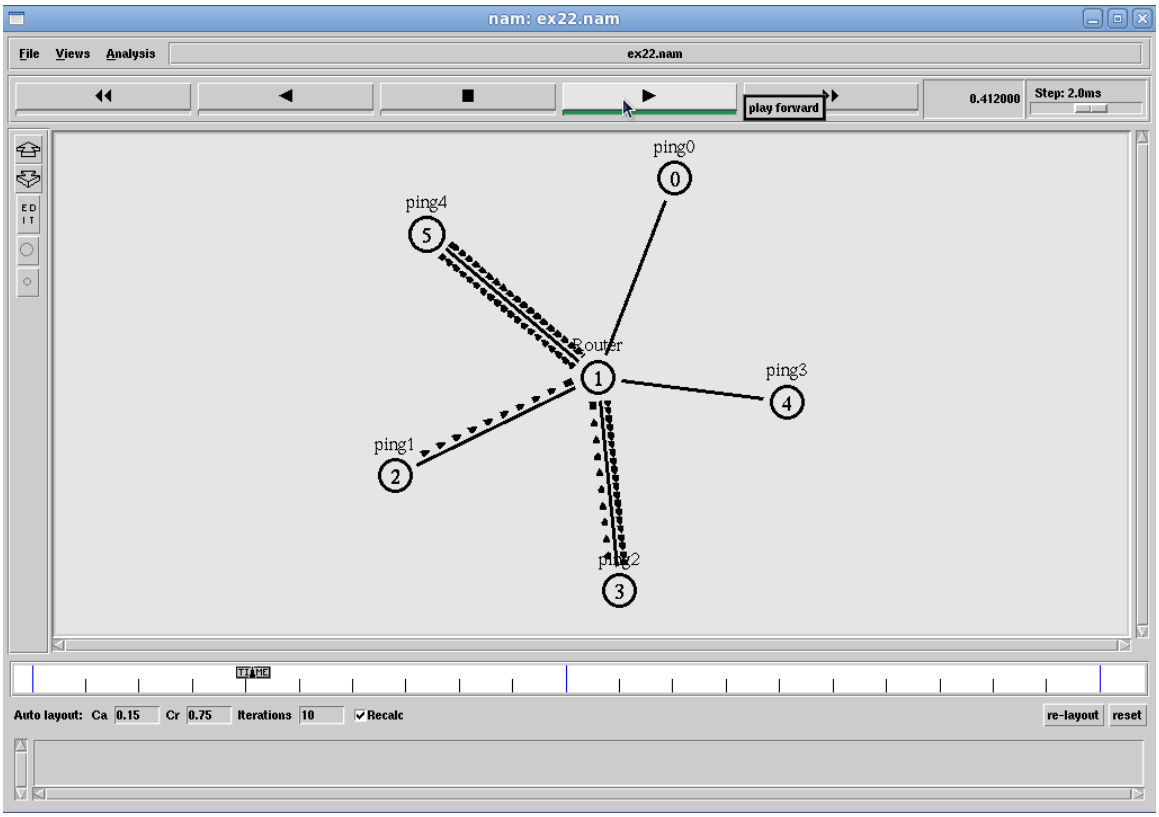
```
BEGIN{c=0;x=0;}
{
        if($1=="r")
        {c++;
        printf("%s\t %s\n",$5,$11);
        }
        if($1=="d")
        {x++;
        printf("%s\t %s\n",$5,$11);
        }

}
END{
printf("the number of packets dropped = %d\n",x);
printf("the number of packets received = %d\n",c);
printf("the thorughput is = %f\n",(c/(c+x))*100);
}
```

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.

```
#create a simulator object
set ns [new Simulator]
#Open a nam trace file
set nf [open PING.nam w]
$ns namtrace-all $nf
#Open a trace file
set nt [open PING.tr w]
$ns trace-all $nt
#Define a 'finish' procedure
proc finish {} {
global ns nf nt
$ns flush-trace
close $nf
close $nt
exec nam PING.nam &
exit 0
}
#Create six nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
#Connect the nodes with two links
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n2 $n1 1Mb 10ms DropTail
$ns duplex-link $n3 $n1 1Mb 10ms DropTail
$ns duplex-link $n4 $n1 1Mb 10ms DropTail
$ns duplex-link $n5 $n1 1Mb 10ms DropTail
#set queue length
$ns queue-limit $n0 $n1 5
$ns queue-limit $n2 $n1 2
$ns queue-limit $n3 $n1 5
$ns queue-limit $n4 $n1 2
$ns queue-limit $n5 $n1 2
#Label the nodes
$n0 label "ping0"
$n1 label "Router"
$n2 label "ping2"
```

```
$n3 label "ping3"
$n4 label "ping4"
$n5 label "ping5"
#color the flow
$ns color 2 Blue
$ns color 3 Red
$ns color 4 Yellow
$ns color 5 Green
#Define a 'recv' function for the class 'Agent/Ping'
Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id] received ping answer from \
$from with round-trip-time $rtt ms."
}
#Create ping agents and attach them to the nodes
set p0 [new Agent/Ping]
$ns attach-agent $n0 $p0
$p0 set class_ 1
set p2 [new Agent/Ping]
$ns attach-agent $n2 $p2
$p2 set class_ 2
set p3 [new Agent/Ping]
$ns attach-agent $n3 $p3
$p3 set class_ 3
set p4 [new Agent/Ping]
$ns attach-agent $n4 $p4
$p4 set class_ 4
set p5 [new Agent/Ping]
$ns attach-agent $n5 $p5
$p4 set class_ 5
#Connect the two agents
$ns connect $p2 $p5
$ns connect $p3 $p5
proc SendPingPacket {} {
global ns p2 p3
set intervalTime 0.001
set now [$ns now]
$ns at [expr $now+$intervalTime] "$p2 send"
$ns at [expr $now+$intervalTime] "$p3 send"
$ns at [expr $now+$intervalTime] "SendPingPacket"
}
$ns at 0.1 "SendPingPacket"
$ns at 2.0 "finish"
$ns run
```

TOPOLOGY:



OUTPUT:

The number of packets dropped = 41
The number of packets received = 14721
The throughput is: 1.002785

AWK Script:

```
BEGIN{c=0;x=0;}
{
        if($1=="r")
        {c++;
        printf("%s\t %s\n",$5,$11);
        }
        if($1=="d")
        {x++;
        printf("%s\t %s\n",$5,$11);
        }

}
END{
printf("the number of packets dropped = %d\n",x);
```

```
printf("the number of packets received = %d\n",c);
printf("the thorughput is = %f\n",(c/(c+x))*100);
}
```

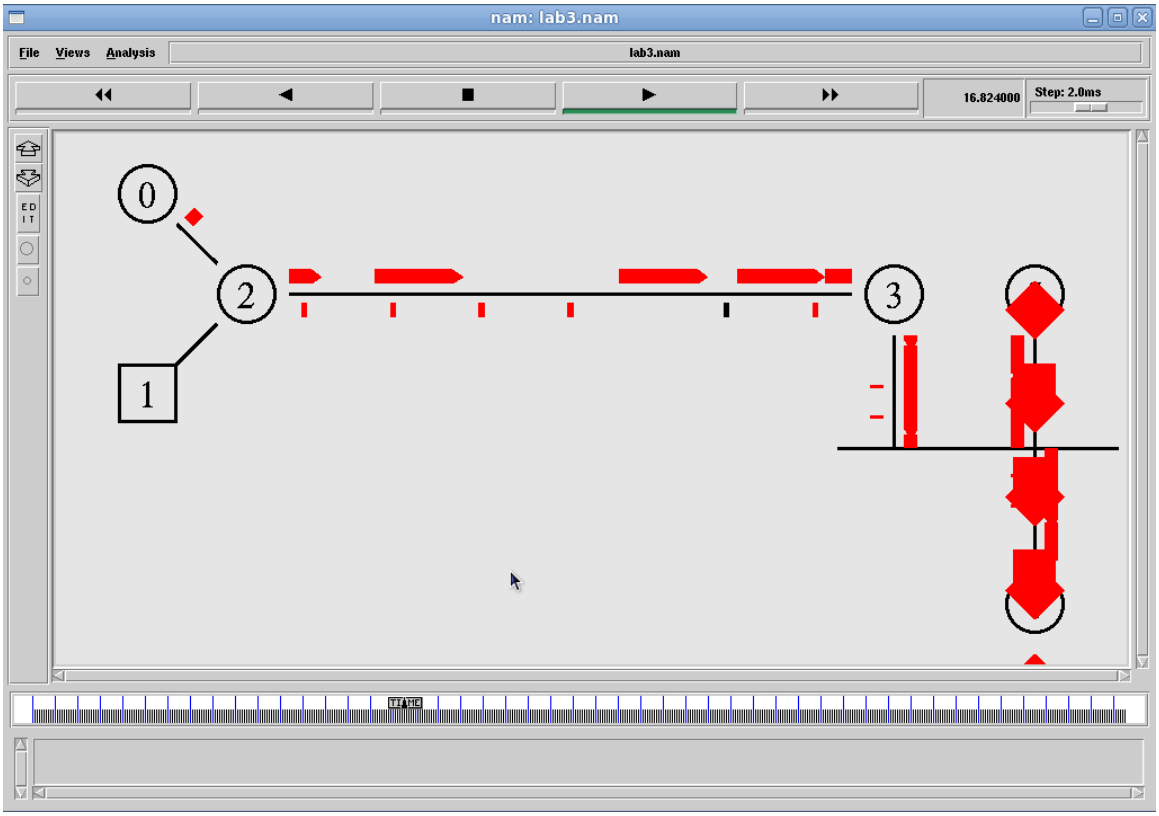
3. Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

```
#PART A:Lab3: Implement an Ethernet LAN using n nodes and set multple traffic
nodes and plot the congestion window for different source/destionation.
# set ns simulation
set ns [new Simulator]
#define color for data flows
$ns color 1 Blue
$ns color 2 Red
#open tracefiles
set tracefile1 [open lab3.tr w]
set winfile [open WinFile w]
$ns trace-all $tracefile1
#open nam file
set namfile [open lab3.nam w]
$ns namtrace-all $namfile
#define the finish procedure
proc finish {} {
global ns tracefile1 namfile
$ns flush-trace
close $tracefile1
close $namfile
exec nam lab3.nam &
exit 0
}
#create six nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
$n1 shape box
#create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns simplex-link $n2 $n3 0.3Mb 100ms DropTail
$ns simplex-link $n3 $n2 0.3Mb 100ms DropTail
set lan [$ns newLan "$n3 $n4 $n5" 0.5Mb 40ms LL Queue/DropTail MAC/802_3]
#Give node position
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
```

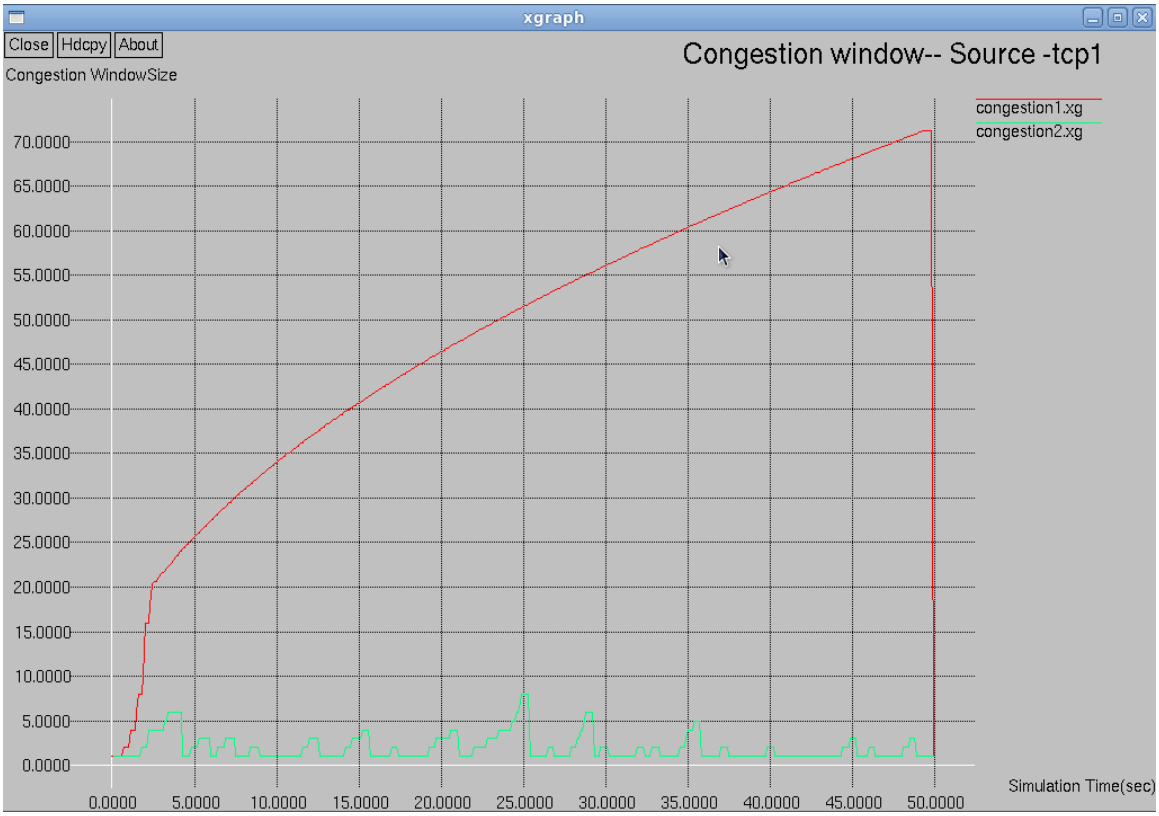
```
$ns simplex-link-op $n2 $n3 orient right
$ns simplex-link-op $n3 $n2 orient left
#set queue size of link(n2-n3) to 20
$ns queue-limit $n2 $n3 20
#setup TCP connection
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set packet_size_ 552
#set ftp over tcp connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
#setup a TCP1 connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1
set sink1 [new Agent/TCPSink]
$ns attach-agent $n5 $sink1
$ns connect $tcp1 $sink1
$tcp1 set packetSize_ 552
$tcp set fid_ 2
#setup a telnet over TCP1 connection
set telnet0 [new Application/Telnet]
$telnet0 attach-agent $tcp1
# Title of Congestion Window1
set outfile1 [open congestion1.xg w]
puts $outfile1 "TitleText: Congestion Window-- Source _tcp"
puts $outfile1 "xUnitText: Simulation Time(Secs)"
puts $outfile1 "yUnitText: Congestion windowSize"
# Title of Congestion Window2
set outfile2 [open congestion2.xg w]
puts $outfile2 "TitleText: Congestion Window-- Source _tcp1"
puts $outfile2 "xUnitText: Simulation Time(Secs)"
puts $outfile2 "yUnitText: Congestion WindowSize"
proc plotWindow {tcpSource outfile} {
global ns
set time 0.1
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $outfile "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $outfile"
}
$ns at 0.1 "plotWindow $tcp $winfile"
```

```
#scheduling the events
$ns at 0.0 "plotWindow $tcp $outfile1"
$ns at 0.1 "plotWindow $tcp1 $outfile2"
$ns at 0.3 "$ftp start"
$ns at 0.5 "$telnet0 start"
$ns at 49.0 "$ftp stop"
$ns at 49.1 "$telnet0 stop"
$ns at 50.0 "finish"
$ns run
```

TOPOLOGY:



OUTPUT:

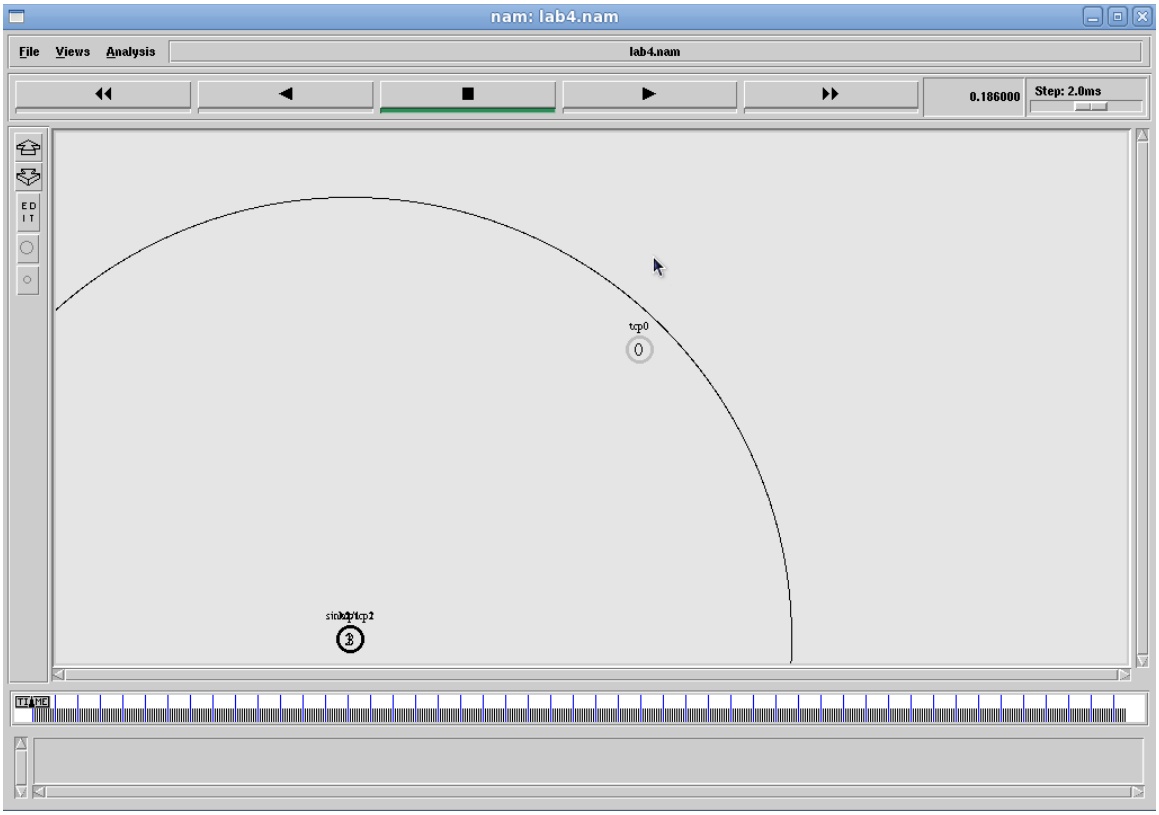

xgraph
Close
Hdcpy
About
Congestion WindowSize
Congestion window-- Source -tcp1
congestion1.xg
congestion2.xg
70.0000
65.0000
60.0000
55.0000
50.0000
45.0000
40.0000
35.0000
30.0000
25.0000
20.0000
15.0000
10.0000
5.0000
0.0000
0.0000
5.0000
10.0000
15.0000
20.0000
25.0000
30.0000
35.0000
40.0000
45.0000
50.0000
Simulation Time(sec)

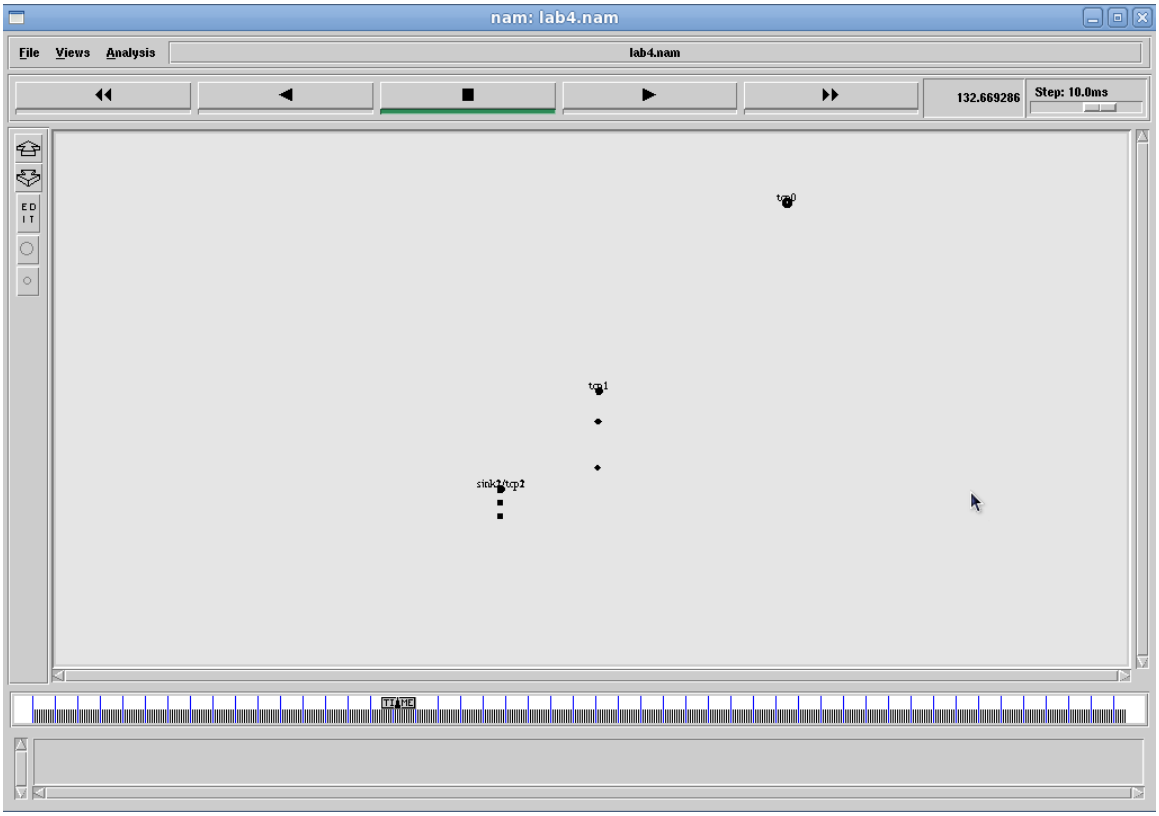
4. Implement simple ESS and with transmitting nodes in wireless LAN by simulation and determine the performance with respect to transmission of packets.

```
#create new simulator object
 set ns [new Simulator]
#open the trace file
 set nt [open lab42.tr w]
$ns trace-all $nt
#create new topology grid
set topo [new Topography]
$topo load_flatgrid 1000 1000
# open namfile
set nf [open lab42.nam w]
$ns namtrace-all-wireless $nf 1000 1000
#define wireless node config
$ns node-config -adhocRouting DSDV \
-llType LL \
-macType Mac/802_11 \
-ifqType Queue/DropTail \
-ifqLen 20 \
-phyType Phy/WirelessPhy \
-channelType Channel/WirelessChannel \
-propType Propagation/TwoRayGround \
-antType Antenna/OmniAntenna \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON
#create General operations Director GOD object
create-god 4
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
#label the nodes
$n0 label "tcp0"
$n1 label "sink0"
$n2 label "bs1"
$n3 label "bs2"
#set the destination
$n0 set X_ 110
$n0 set Y_ 500
$n0 set Z_ 0
$n1 set X_ 600
```

```
$n1 set Y_ 500
$n1 set Z_ 0
$n2 set X_ 300
$n2 set Y_ 500
$n2 set Z_ 0
$n3 set X_ 450
$n3 set Y_ 500
$n3 set Z_ 0
#create and attach agents
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
#define mobility
$ns at 0.5 "$ftp0 start"
$ns at 0.3 "$n0 setdest 110 500 10"
$ns at 0.3 "$n1 setdest 600 500 20"
$ns at 0.3 "$n2 setdest 300 500 30"
$ns at 0.3 "$n3 setdest 450 500 30"
$ns at 10.0 "$n0 setdest 100 550 5"
$ns at 10.0 "$n1 setdest 630 450 5"
$ns at 70.0 "$n0 setdest 170 680 5"
$ns at 70.0 "$n1 setdest 580 380 5"
$ns at 120.0 "$n0 setdest 140 720 5"
$ns at 135.0 "$n0 setdest 110 600 5"
$ns at 140.0 "$n1 setdest 600 550 5"
$ns at 155.0 "$n0 setdest 89 500 5"
$ns at 190.0 "$n0 setdest 100 440 5"
$ns at 210.0 "$n1 setdest 700 600 5"
$ns at 240.0 "$n1 setdest 650 500 5"
#define finish procedure
proc finish { } {
global ns nt nf
$ns flush-trace
exec nam lab42.nam &
close $nt
close $nf
exit 0
}
$ns at 400 "finish"
$ns run
```

TOPOLOGY:


nam: lab4.nam
File Views Analysis
lab4.nam
0.186000
Step: 2.0ms
tcp0
0
sink2/tcp2
2


nam: lab4.nam
File Views Analysis
lab4.nam
132.669286
Step: 10.0ms
tcp1
sink2/tcp2

OUTPUT:

```
Number of packets sent = 9933
Number of packets received = 9896
Packet Delivery Ratio = 99.6275
Routing Load = 1.00283
```

AWK Script:

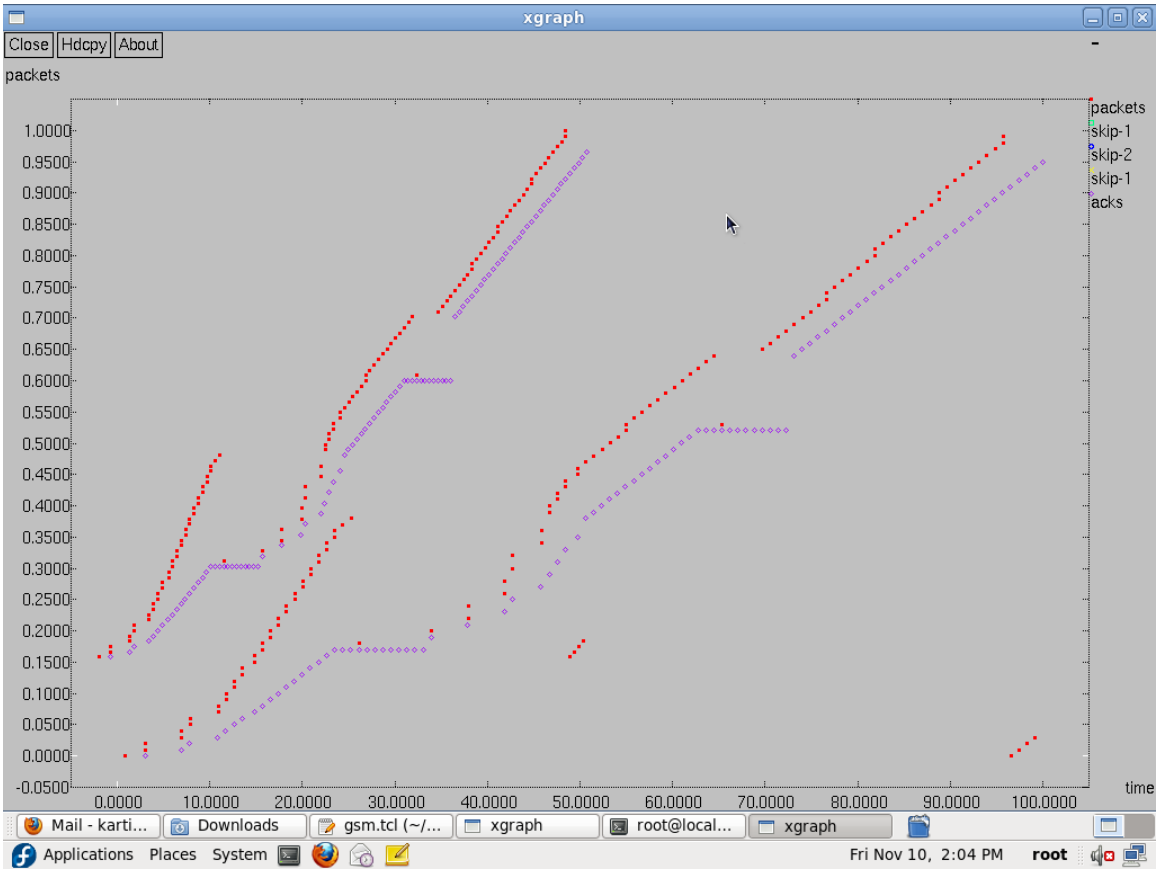
```
BEGIN{
pktsent=0;
pktrcvd=0;
pktrtr=0;
}
{
        if($1=="s" && $4 == "RTR" && $7 == "tcp")
        {
   pktrtr++;
        }
        if($1=="s" && $4 == "AGT" && $7 == "tcp")
        {
   pktsent++;
        }
   if($1=="r" && $4 == "AGT" && $7 == "tcp")
        {
   pktrcvd++;
        }

}
END{
printf("the number of packets sent = %d\n",pktsent);
printf("the number of packets received = %d\n",pktrcvd);
printf("\nPacket Delivery Ratio = %f",pktrcvd/pktsent * 100);
printf("\nRouting Load = %f",pktrtr/pktrcvd);
}
```

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

```
# 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 ns [new Simulator]
set tf [open out.tr w]
$ns trace-all $tf
set nodes(is) [$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
#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
set wrap $opt(wrap)
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
set a "out.tr"
set GETRC "/root/ns-allinone-2.35/ns-2.35/bin/getrc"
set RAW2XG "/root/ns-allinone-2.35/ns-2.35/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 xgraph -x time -y packets plot.xgr &
exit 0
}
$ns at $stop "stop"
$ns run
```

OUTPUT:

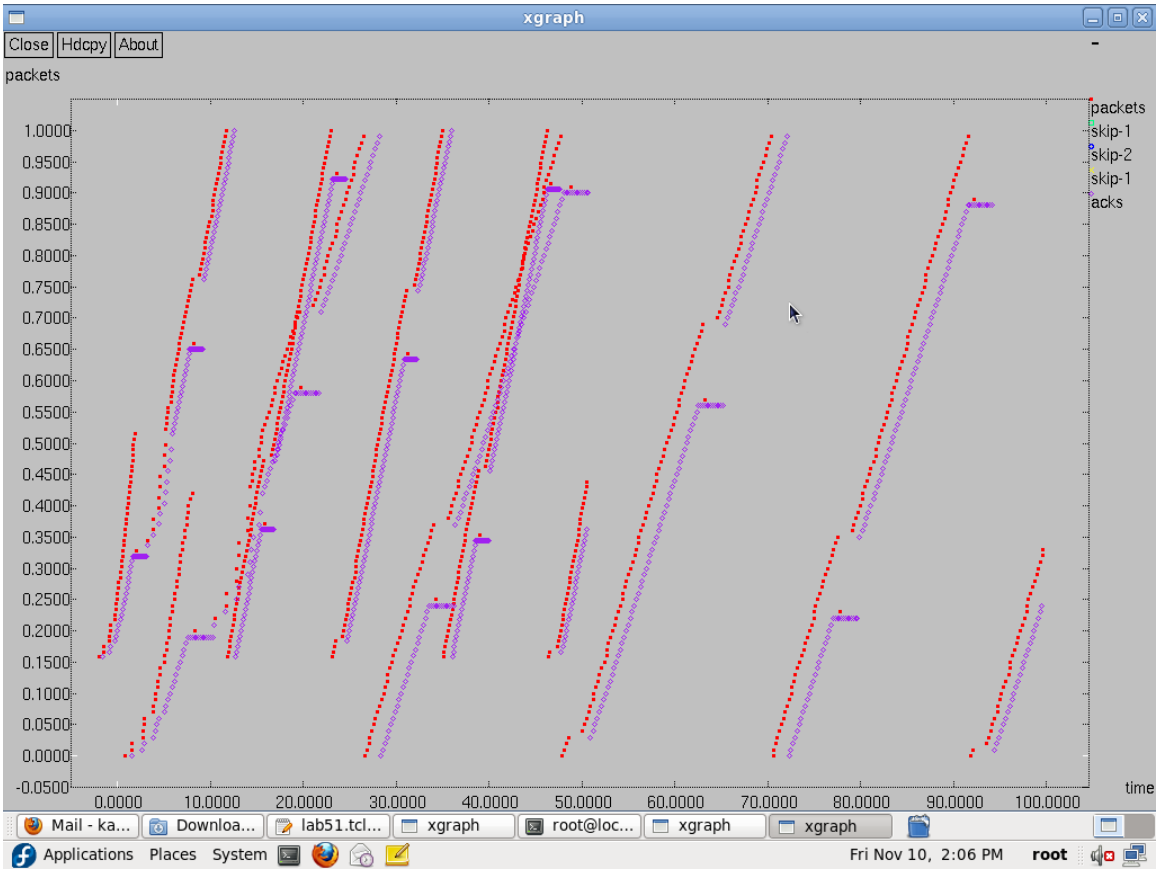

xgraph
Close
Hdcpy
About
packets
1.0000
0.9500
0.9000
0.8500
0.8000
0.7500
0.7000
0.6500
0.6000
0.5500
0.5000
0.4500
0.4000
0.3500
0.3000
0.2500
0.2000
0.1500
0.1000
0.0500
0.0000
-0.0500
0.0000
10.0000
20.0000
30.0000
40.0000
50.0000
60.0000
70.0000
80.0000
90.0000
100.0000
time
packets
skip-1
skip-2
skip-1
acks
Mail - karti...
Downloads
gsm.tcl (~/...
xgraph
root@local...
xgraph
Applications
Places
System
Fri Nov 10, 2:04 PM
root

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

```
# 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 ns [new Simulator]
set tf [open out.tr w]
$ns trace-all $tf
set nodes(is) [$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
#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
set wrap $opt(wrap)
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
set a "out.tr"
set GETRC "/root/ns-allinone-2.35/ns-2.35/bin/getrc"
set RAW2XG "/root/ns-allinone-2.35/ns-2.35/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 xgraph -x time -y packets plot.xgr &
exit 0
}
$ns at $stop "stop"
$ns run
```

OUTPUT:


xgraph
Close
Hdcpy
About
packets
1.0000
0.9500
0.9000
0.8500
0.8000
0.7500
0.7000
0.6500
0.6000
0.5500
0.5000
0.4500
0.4000
0.3500
0.3000
0.2500
0.2000
0.1500
0.1000
0.0500
0.0000
-0.0500
0.0000
10.0000
20.0000
30.0000
40.0000
50.0000
60.0000
70.0000
80.0000
90.0000
100.0000
time
packets
skip-1
skip-2
skip-1
acks
Mail - ka...
Downloa...
lab51.tcl...
xgraph
root@loc...
xgraph
xgraph
Applications
Places
System
Fri Nov 10, 2:06 PM
root

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

```
import java.util.Scanner;

public class CRC {
	public static int n;
	static String divide(String s){
		String div = "10001000000100001";
		char x;
		int i,j;
		for (i=0;i<n;i++)
		{
			x = s.charAt(i);
			for (j=0;j<17;j++)
			{
				if(x=='1')
					if (s.charAt(i+j) != div.charAt(j))
						s =
s.substring(0,i+j)+"1"+s.substring(i+j+1);
				else
						s =
s.substring(0,i+j)+"0"+s.substring(i+j+1);
			}
		}
		return s;
	}
	/**
	 * @param args
	 */
	public static void main(String[] args) {
		Scanner s = new Scanner(System.in);
		String code,copy,rec,zero="0000000000000000";
		System.out.println("Enter the message to be transmitted");
		code = s.nextLine();
		n = code.length();
		copy = code;
		code += "0000000000000000";
		code = divide(code);
		System.out.println("Message= "+copy);
		copy = copy.substring(0,n)+code.substring(n);
		System.out.println("CRC="+code.substring(n));
		System.out.println("Sender transmitted from is="+copy);
		System.out.println("Enter recieved data:");
		rec = s.nextLine();
		if (zero.equals(divide(rec).substring(n)))
			System.out.println("Frame recieved without errors");
```

```
                else
                        System.out.println("Recieved frame contains one or more
errrors");
                s.close();
                }
        }
```

OUTPUT:

1)
Enter the message to be transmitted
10110011000111
Message= 10110011000111
CRC=1110101000100000
Sender transmitted from is=10110011000111111010101000100000
Enter recieved data:
10110011000111111010010001000000
Recieved frame contains one or more errrors

2) Enter the message to be transmitted
1111000011110000
Message= 1111000011110000
CRC=1111110011011110
Sender transmitted from is=11110000111100001111110011011110
Enter recieved data:
11110000111100001111110011011110
Frame recieved without errors

3) Enter the message to be transmitted
10110011000111
Message= 10110011000111
CRC=1110101000100000
Sender transmitted from is=10110011000111111010101000100000
Enter recieved data:
10110011000111111010101000100000
Frame recieved without errors

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

```
import java.util.Scanner;

public class BellmanFord {

	private int distance[];
	private int numb_vert;
	public static final int MAX = 999;
	public BellmanFord(int numb_vert)
	{
		this.numb_vert = numb_vert;
		distance = new int[numb_vert + 1];
	}
	public void BellmanFordEvaluation(int source,int adj_matrix[][])
	{
		for(int node = 1;node <= numb_vert; node++)
		{
			distance[node] = MAX;
		}
		distance[source] = 0;
		for (int node = 1; node <= numb_vert - 1; node++) {
			for (int src_node = 1; src_node <= numb_vert; src_node++) {
				for (int dest_node = 1; dest_node <= numb_vert;
dest_node++) {
					if (adj_matrix[src_node][dest_node] != MAX)
					{
						if(distance[dest_node] > distance[src_node]
+ adj_matrix[src_node][dest_node])
							distance[dest_node] =
distance[src_node] + adj_matrix[src_node][dest_node];

					}
				}

			}
		}
		for (int src_node = 1; src_node < numb_vert; src_node++) {
			for (int dest_node = 1; dest_node <= numb_vert; dest_node++) {
				if (adj_matrix[src_node][dest_node] != MAX)
				{
					if(distance[dest_node] > distance[src_node] +
adj_matrix[src_node][dest_node])
						System.out.println("Negative edges
detected");
				}
```

```
                        }
                }
                System.out.println("Routing table for router = "+source+"i");
                System.out.println("Destination device/cost \t");
                for (int vertex = 1; vertex < numb_vert; vertex++) {
                        System.out.println(""+vertex+"\t\t\t"+distance[vertex]);
                }
        }
        public static void main(String[] args) {
                int numb_vert = 0;
                Scanner s = new Scanner(System.in);
                System.out.println("Enter the number of vertices");
                numb_vert = s.nextInt();
                int[][] adj_matrix = new int[numb_vert+1][numb_vert+1];
                System.out.println("Enter the adjacency matrix");
                for (int src_node = 1; src_node <= numb_vert; src_node++) {
                        for (int dest_node = 1; dest_node <= numb_vert; dest_node++) {
                                adj_matrix[src_node][dest_node] = s.nextInt();
                                if(src_node == dest_node)
                                        adj_matrix[src_node][dest_node] = 0;
                                if (adj_matrix[src_node][dest_node] == 0)
                                        adj_matrix[src_node][dest_node] = MAX;

                        }
                }
                for (int i = 1; i < numb_vert; i++) {
                        BellmanFord bf = new BellmanFord(numb_vert);
                        bf.BellmanFordEvaluation(i, adj_matrix);
                }
        }
}
```

OUTPUT:

```
1) Enter the number of vertices
5
Enter the adjacency matrix
0 1 999 999 999
999 0 4 999 3
999 999 0 2 999
999 999 999 0 999
999 999 5 999 0
Routing table for router = 1i
Destination device/cost
1                       0
2                       1
3                       5
```

```
4                       7
Routing table for router = 2i
Destination device/cost
1                       999
2                       0
3                       4
4                       6
Routing table for router = 3i
Destination device/cost
1                       999
2                       999
3                       0
4                       2
Routing table for router = 4i
Destination device/cost
1                       999
2                       999
3                       999
4                       0
Routing table for router = 5i
Destination device/cost
1                       999
2                       999
3                       5
4                       7
```

2)

```
Enter the number of vertices
5
Enter the adjacency matrix
0 1 999 999 999
999 0 3 9 999
999 999 0 999 4
2 999 999 0 999
999 7 999 5 0
Routing table for router = 1i
Destination device/cost
1                       0
2                       1
3                       4
4                       10
5                       8
Routing table for router = 2i
Destination device/cost
1                       11
2                       0
3                       3
4                       9
```

```
5                       7
Routing table for router = 3i
Destination device/cost
1                       11
2                       11
3                       0
4                       9
5                       4
Routing table for router = 4i
Destination device/cost
1                       2
2                       3
3                       6
4                       0
5                       10
Routing table for router = 5i
Destination device/cost
1                       7
2                       7
3                       10
4                       5
5                       0
```

3)

```
Enter the number of vertices
6
Enter the adjacency matrix
0 999 4 3 999 999
5 0 10 999 999 999
999 999 0 6 8 1
999 999 999 0 999 9
999 999 999 7 0 999
999 11 999 999 2 0
Routing table for router = 1i
Destination device/cost
1                       0
2                       999
3                       4
4                       3
5                       12
6                       5
Routing table for router = 2i
Destination device/cost
1                       5
2                       0
3                       9
4                       8
5                       17
```

6 10
Routing table for router = 3i
Destination device/cost
1 17
2 999
3 0
4 6
5 8
6 1
Routing table for router = 4i
Destination device/cost
1 36
2 999
3 19
4 0
5 27
6 8
Routing table for router = 5i
Destination device/cost
1 9
2 999
3 13
4 12
5 0
6 14
Routing table for router = 6i
Destination device/cost
1 28
2 999
3 11
4 17
5 19
6 0

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

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.util.Scanner;


public class TCPClient {

	public static void main(String[] args) {
		try {
			Scanner sc = new Scanner(System.in);
			Socket s = new Socket("localhost",998);
			DataInputStream di = new DataInputStream(s.getInputStream());
			DataOutputStream dos = new
DataOutputStream(s.getOutputStream());
			dos.writeUTF("connected to 127.0.0.1 \n");
			System.out.println("\nConnection established to server");
			System.out.println("\nEnter the full path of the file to be
displayed");
			String path = sc.nextLine();
			dos.writeUTF(path);
			System.out.println(new String(di.readUTF()));
			di.close();
			dos.close();
			s.close();
			sc.close();
		}
		catch(IOException e){
			System.out.println("\n error:"+e.getMessage());
		}
	}
}

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.net.ServerSocket;
```

```
import java.net.Socket;
import java.util.Scanner;


public class TCPServer {

	public static void main(String[] args) {
		try {
			ServerSocket s = new ServerSocket(998);
			System.out.println("\n server ready\n waiting.....");
			Socket s1 = s.accept();
			DataOutputStream dop = new
DataOutputStream(s1.getOutputStream());
			DataInputStream di = new DataInputStream(s1.getInputStream());
			System.out.println(di.readUTF());
			String path = di.readUTF();
			System.out.println("Request received \n Processing....");
			try{
				File myFile = new File(path);
				Scanner sc = new Scanner(myFile);
				String st = sc.nextLine();
				st = "\n The contents of file are \n"+st;
				dop.writeUTF(st);
				dop.close();
				di.close();
				s.close();
				s1.close();
			} catch (FileNotFoundException e) {
				System.out.println("Error file not found");
				dop.writeUTF("File not found");
			}
		} catch (IOException e) {
			System.out.println("error"+e.getMessage());
		}
		finally {
			System.out.println("\n Connection terminated \n");
		}

	}

}
```

OUTPUT:

Server:
server ready
 waiting.....

connected to 127.0.0.1

Request received
Processing....

Connection terminated

Client:
Connection established to server

Enter the full path of the file to be displayed
F:\CNLAB\test.txt

The contents of file are
HELLO WORLD

2)
Server:
server ready
waiting.....
connected to 127.0.0.1

Request received
Processing....
Error file not found

Connection terminated
Client:
Connection established to server

Enter the full path of the file to be displayed
F:\CNLAB\est1.txt
File not found

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

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;

public class UDPClient {
	public static void main(String[] args){
		DatagramSocket aSocket = null;
		int clientPort = 999;
		try {
			aSocket = new DatagramSocket(clientPort);
			byte[] buf = new byte[1000];
			byte[] buf1 = new byte[1000];
			DatagramPacket data = new DatagramPacket(buf, buf.length);
			String conf = "Connected to client";
			buf1 = conf.getBytes();
			DatagramPacket data1 = new DatagramPacket(buf1,
buf1.length,InetAddress.getLocalHost(),998);
			aSocket.send(data);
			System.out.println("Connected to server");
			aSocket.receive(data);
			byte[] msg = new byte[1000];
			msg = data.getData();
			System.out.println("Message:"+new
String(msg,0,data.getLength()));
		} catch (SocketException e) {
			System.out.println("Socket:"+e.getMessage());
		}
		catch (IOException e) {
		System.out.println("IO:"+e.getMessage());
	}
		finally {
			if (aSocket != null)
				aSocket.close();
		}
	}
}

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.util.Scanner;
```

```
public class UDPServer {


        public static void main(String[] args) {
                DatagramSocket aSocket = null;
                Scanner s = new Scanner(System.in);
                int serverPort = 998;
                System.out.println("Server Ready \n waiting");
                try {
                        aSocket = new DatagramSocket(serverPort);
                        byte[] buffer = new byte[1000];
                        byte[] buf = new byte[1000];
                        DatagramPacket data1 = new DatagramPacket(buf,buf.length);
                        aSocket.receive(data1);
                        byte[] msg = new byte[1000];
                        msg = data1.getData();
                        System.out.println(new String(msg,0,data1.getLength()));
                        System.out.println("\n Enter message to be sent");
                        String str = s.nextLine();
                        buffer = str.getBytes();
                        DatagramPacket data = new DatagramPacket(buffer,
buffer.length,InetAddress.getLocalHost(),998);
                        aSocket.send(data);
                } catch(SocketException e){
                        System.out.println(e.getMessage());
                } catch (Exception e) {
                        System.out.println(e.getMessage());
                }
                finally {
                        System.out.println("Message sent \n Connection closed.");
                        if(aSocket != null)
                                aSocket.close();
                        s.close();
                }
        }

}
```

OUTPUT:

1)
Server:
Server Ready
 waiting
Connected to client

Enter message to be sent
Hello client
Message sent
Connection closed.
Client:
Connected to server
Message:Hello client

2)
Server:
Server Ready
waiting
Connected to client

Enter message to be sent
Hello world
Message sent
Connection closed.
Client:
Connected to server
Message:Hello world

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

```
public class BTS {
	public String bytesToString(byte[] encrypted){
		String test = " ";
		for (byte b:encrypted)
			test += Byte.toString(b);
		return test;
	}
}

import java.io.DataInputStream;
import java.io.IOException;

public class RSA {

	@SuppressWarnings("deprecation")
	public static void main(String[] args) throws Exception {
		RSALab rsa = new RSALab();
		DataInputStream in = new DataInputStream(System.in);
		String testString;
		System.out.println("Enter plain text");
		testString = in.readLine();
		BTS s1 = new BTS();
		System.out.println("Encrypted String:"+testString);
		System.out.println("String in
bytes:"+s1.bytesToString(testString.getBytes()));
		BTS s2 = new BTS();
		byte[] encrypted = rsa.encrypt(testString.getBytes());
		System.out.println("Encrypted String:"+s2.bytesToString(encrypted));
		BTS s3 = new BTS();
		byte[] decrypted = rsa.decrypt(encrypted);
		System.out.println("Encrypted String:"+s2.bytesToString(decrypted));
		System.out.println("Decrypted String:"+new String(decrypted));
	}
}

import java.math.BigInteger;
import java.util.Random;


public class RSALab {
	private BigInteger p,q,N,phi,e,d;
	@SuppressWarnings("unused")
	private int bitlength = 1024,blocksize = 256;
```

```
        private Random r;

        public RSALab(){
                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 RSALab(BigInteger e,BigInteger d,BigInteger N){
                this.e = e;
                this.d = d;
                this.N = N;
        }
        public byte[] encrypt(byte[] message){
                return (new BigInteger(message).modPow(e, N).toByteArray());
        }
        public byte[] decrypt(byte[] message){
                return (new BigInteger(message).modPow(d, N).toByteArray());
        }
}
```

OUTPUT:

1)

```
Enter plain text
Hello
Encrypted String:Hello
String in bytes: 72101108108111
Encrypted String: 11142-47125-6988-29473363118-111-127-57112-63-51-102376554-
11235107-21125-45-9590-936384-31-98-78126-101-11-1064710976-71-1018677-
71931-111-2410-1161056-3-99-48-7-529-272110-6274-7214-8991116-90390-
38894361105-30-7846-58-1098135-77-70-53-59125-38-128-12389104-78119369497-
444249-1243-12845-891919-40-52125-89516-94105-761-548412-33-4761243-74-71-64-
120115-74-10031-1492-64-21-114-5166-261412611370-14-78106-3108-94-24-485-
1221275121126-4712-66124-49-968-100-1626-89114-82171236436108569515110-
1168357-92-125400-67-10379-577-93623636-5-151121-14-32-82124-832-492678-69-
106-76-2365110-2312790-3733-56148476-11311392332839102-77-48126-9290-107-
9472-639111-110-70-127-9173-673-110-104
Encrypted String: 72101108108111
Decrypted String:Hello
```

2)
Enter plain text
world
Encrypted String:world
String in bytes: 119111114108100
Encrypted String: 206851-93102-106-564962-113968252-67102-85-1193037318412-1131279219-5-604961-10941574-86107-44114111-123126-1121-53-13-8117-280-108-41674101-35-10-3753-94-4-24-1297175584-253910-56-61105-15-76812049-64122-97-76-601073-113-42-184162-88-98-109536577-21112-90641078575-93-60851980-65-4012147118-72-100-261208066-10486-27-126-29109-5-18-827-3347-101-437914712-43-34-729880-101-89-67-122696-9710767123-44-55106-115-63122692-433108-115-1-18-73-29-27-110116-113-41-115-25-44-76-66120-101-1210-128-11010093-79-96-120-539111986-406910346-98-890085-117-12-1231041987104-336-59-11619-9071-28102-73115-9532-11673-74-24-116-84394611647-672411-1083813212736-78-105-44118-7011094-62103-8489-4579-47-116-47
Encrypted String: 119111114108100
Decrypted String:world

3)
Enter plain text
bye
Encrypted String:bye
String in bytes: 98121101
Encrypted String: 91-12182-1208411144100-3026-70104-4212267-4458-271953-7885-7692-30-87-74-266536109118-1181863-19-63109-71108-124-43-60-729162-117-1021278298-96-9740-74-84111-70-39495316-83-12887-956-44-43-126-93-5699-49104816030-107-61883-4155-74-38-23101791320376181584l-115-26-100-25-4853-36-121-989-20642899-70121-11879-103-56-102-8990722268114-9886973449-31113296-9312785-8872-28-1007106-5279-17-6923-68-769-78-8239108399499-71-109593315-3303892-59-112-269735-10244111265834-4412-65-58106-128-86-86-5358126-1-81-32-651053-656034-41111-86111-1031-962-3768-2834-4334-77-38-68-69-21-101-122-123-31-96644092-10636797-88-120108-754359112123-116-33-385222-1387-25-38534-27115-3-4111-6974-8-115
Encrypted String: 98121101
Decrypted String:bye

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

```
import java.util.Scanner;

public class LeakyBucket {
	public static void main(String args[])
	{
		Scanner sc= new Scanner(System.in);
		int bucket =0,op_rate,i,n,bsize;
		System.out.println("Enter the number of packet");
		n=sc.nextInt();
		int pkt[]=new int[n];
		System.out.println("Enter the output rate of bucket");
		op_rate=sc.nextInt();
		System.out.println("Enter the bucket size");
		bsize=sc.nextInt();
		System.out.println("Enter the arriving packet size");
		for(i=0;i<n;i++)
			pkt[i]=sc.nextInt();
		System.out.println("\nSec\tpsize\tBucket\tAccept/Reject\tpkt_send");
		System.out.println("-------------------------------------------------");
		for(i=0;i<n;i++)
		{
			System.out.print(i+1+"\t"+pkt[i]+"\t");
			if(bucket+pkt[i]<=bsize)
			{
				bucket += pkt[i];

	System.out.println(bucket+"\tAccept\t\t"+min(bucket,op_rate)+"\n");
				bucket=sub(bucket,op_rate);
			}
			else
			{
				System.out.println(bucket+"\tReject"+(bucket+pkt[i]-
bsize)+"\t"+min(bsize,op_rate)+"\n");
				bucket=bsize;
				bucket=sub(bucket,op_rate);
			}
		}
		while(bucket!=0)
		{
			System.out.println(i+++"\t 0
\t"+bucket+"\tAccept/t/t"+min(bucket,op_rate)+"\n");
			bucket=sub(bucket,op_rate);
		}
		sc.close();
```

```
        }
        static int min(int a, int b)
        {
                return (a<b)?a:b;
        }
        static int sub(int a,int b)
        {
                return (a-b)>0?(a-b):0;
        }

}
```

OUTPUT:

1)
Enter the number of packet
5
Enter the output rate of bucket
10
Enter the bucket size
5
Enter the arriving packet size
1
2
3
4
5

Sec	psize	Bucket	Accept/Reject	pkt_send
1	1	1	Accept	1
2	2	2	Accept	2
3	3	3	Accept	3
4	4	4	Accept	4
5	5	5	Accept	5

2)
Enter the number of packet
6
Enter the output rate of bucket
5
Enter the bucket size

```
5
Enter the arriving packet size
2
4
6
8
10
12

Sec     psize   Bucket Accept/Reject pkt_send
-------------------------------------------------
1       2       2       Accept          2

2       4       4       Accept          4

3       6       0       Reject1         5

4       8       0       Reject3         5

5       10      0       Reject5         5

6       12      0       Reject7         5

3)
Enter the number of packet
5
Enter the output rate of bucket
2
Enter the bucket size
10
Enter the arriving packet size
1
3
5
7
9

Sec     psize   Bucket Accept/Reject pkt_send
-------------------------------------------------
1       1       1       Accept          1

2       3       3       Accept          2

3       5       6       Accept          2

4       7       4       Reject1         2

5       9       8       Reject7         2
```

5	0	8	Accept/t/t2
6	0	6	Accept/t/t2
7	0	4	Accept/t/t2
8	0	2	Accept/t/t2