Computer Network Lab Manual [17CSL57]

Prepared By,
Mr. Ranjan G
Assistant Professor
Department of CSE
MSEC

Department of Computer Science and Engineering

M.S Engineering College

(NAAC Accredited and an ISO 9001:2015 Certified Institution)
(Affiliated to Visvesvaraya Technological University Belgaum and approved by AICTE, New Delhi)
NAVARATHNA AGRAHARA, SADAHALLI POST, BANGALORE- 562 110, Tel: 080-3252 9939, 3252 957

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.

```
set ns [ new Simulator ]

set tf [ open abc.tr w ]
$ns trace-all $tf

set nf [ open exp1.nam w ]
$ns namtrace-all $nf

# The below code is used to create the nodes.
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#This is used to give color to the packets.
$ns color 1 "red"
$ns color 2 "blue"
$n0 label "Source/udp0"
$n1 label "Source/udp1"
$n2 label "Router"
$n3 label "Destination/Null"

#Vary the below Bandwidth and see the number of packetsdropped.
$ns duplex-link $n0 $n2 100Mb 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 1
$ns set queue-limit $n1 $n2 1
$ns set queue-limit $n2 $n3 1

#The below code is used to attach an UDP agent to n0, UDP
#agent to n1 and null agent to n3.
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0

set null3 [new Agent/Null]
$ns attach-agent $n3 $null3
set udp1 [new Agent/UDP]
```

```
$ns attach-agent $n1 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1

#The below code sets the udp0 packets to red and udp1
#packets to blue color
$udp0 set class_ 1
$udp1 set class_ 2

#The below code is used to connect the agents.
$ns connect $udp0 $null3
$ns connect $udp1 $null3

#The below code is used to set the packet size to 500
$cbr1 set packetSize_ 500Mb

#The below code is used to set the interval of the packets,
#i.e., Data rate of the packets. if the data rate is high
#then packets drops are high.

$cbr1 set interval_ 0.005
proc finish {} {
        global ns nf tf
        $ns flush-trace
        exec nam exp1.nam &
        close $tf
        close $nf
        exit 0
}
$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr1 start"
$ns at 10.0 "finish"
$ns run
```

awk script

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

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.

```
set ns [new Simulator]

set tf [open exp2.tr w]
$ns trace-all $tf

set nf [open exp2.nam w]
$ns namtrace-all $nf

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]

$n0 label "Ping0"
$n4 label "Ping4"
$n5 label "Ping5"
$n6 label "Ping6"
$n2 label "Router"
$ns color 1 "red"
$ns color 2 "green"

$ns duplex-link $n0 $n2 100Mb 300ms DropTail
$ns duplex-link $n2 $n6 1Mb 300ms DropTail
$ns duplex-link $n5 $n2 100Mb 300ms DropTail
$ns duplex-link $n2 $n4 1Mb 300ms DropTail
$ns duplex-link $n3 $n2 1Mb 300ms DropTail
$ns duplex-link $n1 $n2 1Mb 300ms DropTail

$ns queue-limit $n0 $n2 5
$ns queue-limit $n2 $n6 2
$ns queue-limit $n2 $n4 3
$ns queue-limit $n5 $n2 5

#The below code is used to connect between the ping agents
#to the node n0, n4 , n5 and n6.
set ping0 [new Agent/Ping]
$ns attach-agent $n0 $ping0

set ping4 [new Agent/Ping]
$ns attach-agent $n4 $ping4
```

```
set ping5 [new Agent/Ping]
$ns attach-agent $n5 $ping5

set ping6 [new Agent/Ping]
$ns attach-agent $n6 $ping6

$ping0 set packetSize_ 50000
$ping0 set interval_ 0.0001
$ping5 set packetSize_ 60000
$ping5 set interval_ 0.00001
$ping0 set class_ 1
$ping5 set class_ 2

$ns connect $ping0 $ping4
$ns connect $ping5 $ping6

#The below function is executed when the ping agent receives
#a reply from the destination
Agent/Ping instproc recv {from rtt} {
        $self instvar node_
        puts " The node [$node_ id] received an reply from $from with round trip
time of $rtt"
}
proc finish {} {
        global ns nf tf
        exec nam exp2.nam &
        $ns flush-trace
        close $tf
        close $nf
        exit 0
}
#The below code makes the link down(failure) at 0.9 from n2
#to n6 and when the time becomes 1.5 the link between n2 to
#n6 is enabled.
$ns rtmodel-at 0.9 down $n2 $n6
$ns rtmodel-at 1.5 up $n2 $n6
$ns at 0.1 "$ping0 send"
$ns at 0.2 "$ping0 send"
$ns at 0.3 "$ping0 send"
$ns at 0.4 "$ping0 send"
$ns at 0.5 "$ping0 send"
$ns at 0.6 "$ping0 send"
$ns at 0.7 "$ping0 send"
$ns at 0.8 "$ping0 send"
$ns at 0.9 "$ping0 send"
$ns at 1.0 "$ping0 send"
$ns at 1.1 "$ping0 send"
$ns at 1.2 "$ping0 send"
```

```
$ns at 1.3 "$ping0 send"
$ns at 1.4 "$ping0 send"
$ns at 1.5 "$ping0 send"
$ns at 1.6 "$ping0 send"
$ns at 1.7 "$ping0 send"
$ns at 1.8 "$ping0 send"

$ns at 0.1 "$ping5 send"
$ns at 0.2 "$ping5 send"
$ns at 0.3 "$ping5 send"
$ns at 0.4 "$ping5 send"
$ns at 0.5 "$ping5 send"
$ns at 0.6 "$ping5 send"
$ns at 0.7 "$ping5 send"
$ns at 0.8 "$ping5 send"
$ns at 0.9 "$ping5 send"
$ns at 1.0 "$ping5 send"
$ns at 1.1 "$ping5 send"
$ns at 1.2 "$ping5 send"
$ns at 1.3 "$ping5 send"
$ns at 1.4 "$ping5 send"
$ns at 1.5 "$ping5 send"
$ns at 1.6 "$ping5 send"
$ns at 1.7 "$ping5 send"
$ns at 1.8 "$ping5 send"
$ns at 5.0 "finish"
$ns run
```

awk script

```
BEGIN{
#include<stdio.h>
count=0;
}
{
if($1=="d")
count++
}
END{
printf("The Total no of Packets Dropped due toCongestion:%d \n", count)
}
```

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

```
set ns [new Simulator]

set tf [open lab7.tr w]
$ns trace-all $tf

set nf [open lab7.nam w]
$ns namtrace-all $nf

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

$ns make-lan "$n0 $n1 $n2 $n3" 10mb 10ms LL Queue/DropTail Mac/802_3

set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0

set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0

set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3
$ns connect $tcp0 $sink3

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

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

set sink1 [new Agent/TCPSink]
$ns attach-agent $n1 $sink1

$ns connect $tcp2 $sink1

######To trace the congestion window###########
set file1 [open file1.tr w]
$tcp0 attach $file1
$tcp0 trace cwnd_
$tcp0 set maxcwnd_ 10

set file2 [open file2.tr w]
$tcp2 attach $file2
$tcp2 trace cwnd_
```

```
proc finish {} {
        global nf tf ns
        $ns flush-trace
        exec nam lab7.nam &
        close $nf
        close $tf
        exit 0
}

$ns at 0.1 "$ftp0 start"
$ns at 1.5 "$ftp0 stop"

$ns at 2 "$ftp0 start"
$ns at 3 "$ftp0 stop"

$ns at 0.2 "$ftp2 start"
$ns at 2 "$ftp2 stop"

$ns at 2.5 "$ftp2 start"
$ns at 4 "$ftp2 stop"

$ns at 5.0 "finish"
$ns run
```

awk script

```
BEGIN {
}
{
if($6= ="cwnd_") /* don't leave space after writing cwnd_ */
printf("%f\t%f\t\n",$1,$7); /* you must put \n in printf */
}
END {
}
```

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

```
set ns [new Simulator]

set tf [open exp4.tr w]
$ns trace-all $tf

set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open exp4.nam w]

$ns namtrace-all-wireless $nf 1000 1000
$ns node-config -adhocRouting DSDV \
        -llType LL \
        -macType Mac/802_11 \
        -ifqType Queue/DropTail \
        -ifqLen 50 \
        -phyType Phy/WirelessPhy \
        -channelType Channel/WirelessChannel \
        -propType Propagation/TwoRayGround \
        -antType Antenna/OmniAntenna \
        -topoInstance $topo \
        -agentTrace ON \
        -routerTrace 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"

#The below code is used to give the initial node positions.
$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"
#The below code is used to provide the node movements.
$ns at 100 "$n1 setdest 550 550 15"
$ns at 190 "$n1 setdest 70 70 15"

proc finish {} {
        global ns nf tf
        $ns flush-trace
        exec nam exp4.nam &
        close $tf
        exit 0
}
$ns at 250 "finish"
$ns run
```

awk script

```
BEGIN{
        #include<stdio.h>
        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: %fMbps\n",
        ((count1*pack1*8)/(time1*1000000)))
        printf("The Throughput from n1 to n2: %f Mbps\n",
        ((count2* pack2 * 8) /(time2*1000000)))
}
```

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;
# Topology
set opt(type) gsm;#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(gsmbuf) 10; # buffer size for gsm
```

```
#default downlink bandwidth in bps
set bwDL(gsm) 9600
#default uplink bandwidth in bps
set bwUL(gsm) 9600
#default downlink propagation delay in seconds
set propDL(gsm) .500
#default uplink propagation delay in seconds
set propUL(gsm) .500
#default buffer size in packets
set buf(gsm) 10
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 "Cell Topology"
}

proc set_link_params {t} {
global ns nodes bwUL bwDL propUL propDL buf
$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
#source web.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 $opt(wrap)
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
if {$opt(srcTrace) == "is"} {
set a "-a out.tr"
} else {
set a "out.tr"
}
```

```
set GETRC "/home/ranjan/ns2/ns-allinone-2.35/ns-2.35/bin/getrc"
set RAW2XG "/home/ranjan/ns2/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 $GETRC -s $sid -d $did -f 1 out.tr | \
$RAW2XG -s 0.01 -m $wrap -r >> plot.xgr
exec $GETRC -s $did -d $sid -f 1 out.tr | \
$RAW2XG -s 0.01 -m $wrap -a >> plot.xgr
exec ./xg2gp.awk plot.xgr
if {!$opt(quiet)} {
exec xgraph -bb -tk -nl -m -x time -y packets plot.xgr &
}
exit 0
}
$ns at $opt(stop) "stop"
$ns run
```

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 anythng
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
set bwDL(umts) 384000
#default uplink bandwidth in bps
set bwUL(umts) 64000
#default downlink propagation delay in seconds
set propDL(umts) .150
#default uplink propagation delay in seconds
set propUL(umts) .150
#default buffer size in packets
set buf(umts) 20
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 "Cell Topology"
}

proc set_link_params {t} {
global ns nodes bwUL bwDL propUL propDL buf
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) simplex
$ns bandwidth $nodes(ms) $nodes(bs1) $bwUL($t) simplex
$ns delay $nodes(bs1) $nodes(ms) $propDL($t) simplex
$ns delay $nodes(ms) $nodes(bs1) $propDL($t) simplex
$ns queue-limit $nodes(bs1) $nodes(ms) $buf($t)
$ns queue-limit $nodes(ms) $nodes(bs1) $buf($t)
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) simplex
$ns bandwidth $nodes(ms) $nodes(bs2) $bwUL($t) simplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) simplex
$ns delay $nodes(ms) $nodes(bs2) $propDL($t) simplex
$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
source web.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 $opt(wrap)
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
if {$opt(srcTrace) == "is"} {
set a "-a out.tr"
} else {
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 $GETRC -s $sid -d $did -f 1 out.tr | \
$RAW2XG -s 0.01 -m $wrap -r >> plot.xgr
exec $GETRC -s $did -d $sid -f 1 out.tr | \
$RAW2XG -s 0.01 -m $wrap -a >> plot.xgr
exec ./xg2gp.awk plot.xgr
if {!$opt(quiet)} {
exec xgraph -bb -tk -nl -m -x time -y packets plot.xgr &
}
exit 0
}
$ns at $opt(stop) "stop"
$ns run
```

Part B

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

```
import java.io.*;
class Crc
{
	public static void main(String args[]) throws IOException
	{
		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());

		System.out.println("Enter number of bits in divisor : ");
		divisor_bits=Integer.parseInt(br.readLine());

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

		tot_length=data_bits+divisor_bits;
		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(div, 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(crc, 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 div[],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;
```

```
        }
}
```

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

```
import java.util.Scanner;
public class BellmanFord
{
	private int D[];
	private int num_ver;
	public static final int MAX_VALUE = 999;
	public BellmanFord(int num_ver)
	{
		this.num_ver = num_ver;
		D = new int[num_ver + 1];
	}
	public void BellmanFordEvaluation(int source, int A[][])
	{
		for (int node = 1; node <= num_ver; node++)
		{
			D[node] = MAX_VALUE;
		}
		D[source] = 0;
		for (int node = 1; node <= num_ver - 1; node++)
		{
			for (int sn = 1; sn <= num_ver; sn++)
			{
				for (int dn = 1; dn <= num_ver; dn++)
				{
					if (A[sn][dn] != MAX_VALUE)
					{
						if (D[dn] > D[sn]+ A[sn][dn])
							D[dn] = D[sn] + A[sn][dn];
					}
				}
			}
		}
		for (int sn = 1; sn <= num_ver; sn++)
		{
			for (int dn = 1; dn <= num_ver; dn++)
			{
				if (A[sn][dn] != MAX_VALUE)
				{
					if (D[dn] > D[sn]+ A[sn][dn])
						System.out.println("The Graph contains
negative egde cycle");
				}
			}
		}
		for (int vertex = 1; vertex <= num_ver; vertex++)
```

```
                {
                        System.out.println("distance of source " + source + " to "+
vertex + "is " + D[vertex]);
                }
        }
        public static void main(String[ ] args)
        {
                int num_ver = 0;
                int source;
                Scanner scanner = new Scanner(System.in);
                System.out.println("Enter the number of vertices");
                num_ver = scanner.nextInt();
                int A[][] = new int[num_ver + 1][num_ver + 1];
                System.out.println("Enter the adjacency matrix");
                for (int sn = 1; sn <= num_ver; sn++)
                {
                        for (int dn = 1; dn <= num_ver; dn++)
                        {
                                A[sn][dn] = scanner.nextInt();
                                if (sn == dn)
                                {
                                        A[sn][dn] = 0;
                                        continue;
                                }
                                if (A[sn][dn] == 0)
                                {
                                        A[sn][dn] = MAX_VALUE;
                                }
                        }
                }
                System.out.println("Enter the source vertex");
                source = scanner.nextInt();
                BellmanFord b = new BellmanFord (num_ver);
                b.BellmanFordEvaluation(source, A);
                scanner.close();
        }
}
```

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

Server Program

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Scanner;
class Server
{
	public static void main(String args[])throws Exception
	{
		String filename;
		System.out.println("Enter File Name: ");
		Scanner sc=new Scanner(System.in);
		filename=sc.nextLine();
		sc.close();
		while(true)
		{
			//create server socket on port 5000
			ServerSocket ss=new ServerSocket(5000);
			System.out.println ("Waiting for request");
			Socket s=ss.accept();
			System.out.println ("Connected With "+s.getInetAddress().toString());
			DataInputStream din=new DataInputStream(s.getInputStream());
			DataOutputStream dout=new DataOutputStream(s.getOutputStream());
			try
			{
				String str="";
				str=din.readUTF();
				System.out.println("SendGet....Ok");
				if(!str.equals("stop"))
				{
					System.out.println("Sending File: "+filename);
					dout.writeUTF(filename);
					dout.flush();
					File f=new File(filename);
					FileInputStream fin=new FileInputStream(f);
					long sz=(int) f.length();
					byte b[]=new byte [1024];
```

```
                                int read;
                                dout.writeUTF(Long.toString(sz));
                                dout.flush();
                                System.out.println ("Size: "+sz);
                                System.out.println ("Buf size:
"+ss.getReceiveBufferSize());
                                while((read = fin.read(b)) != -1)
                                {
                                        dout.write(b, 0, read);
                                        dout.flush();
                                }
                                fin.close();
                                System.out.println("..ok");
                                dout.flush();
                        }
                        dout.writeUTF("stop");
                        System.out.println("Send Complete");
                        dout.flush();
                }
                catch(Exception e)
                {
                        e.printStackTrace();
                        System.out.println("An error occured");
                }
                din.close();
                s.close();
                ss.close();
        }
    }
}
```

Client Program

```
import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.EOFException;
import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.net.Socket;
import java.util.Scanner;

class Client
{
        public static void main(String args[])throws Exception
        {
                String address = "";
```

```
                Scanner sc=new Scanner(System.in);
                System.out.println("Enter Server Address: ");
                address=sc.nextLine();
                //create the socket on port 5000
                Socket s=new Socket(address,5000);
                DataInputStream din=new DataInputStream(s.getInputStream());
                DataOutputStream                                        dout=new
DataOutputStream(s.getOutputStream());
                BufferedReader          br=new          BufferedReader(new
InputStreamReader(System.in));
                System.out.println("Send Get to start...");
                String str="",filename="";
                try
                {
                        while(!str.equals("start"))
                                str=br.readLine();

                        dout.writeUTF(str);
                        dout.flush();
                        filename=din.readUTF();
                        System.out.println("Receving file: "+filename);
                        filename="client"+filename;
                        System.out.println("Saving as file: "+filename);
                        long sz=Long.parseLong(din.readUTF());
                        System.out.println ("File Size: "+(sz/(1024*1024))+" MB");
                        byte b[]=new byte [1024];
                        System.out.println("Receving file..");
                        FileOutputStream        fos=new        FileOutputStream(new
File(filename),true);
                        long bytesRead;

                        do
                        {
                                bytesRead = din.read(b, 0, b.length);
                                fos.write(b,0,b.length);
                        }while(!(bytesRead<1024));

                        System.out.println("Comleted");
                        fos.close();
                        dout.close();
                        s.close();
                }
                catch(EOFException e)
                {
                //do nothing
                }
        }
}
```

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

```
import java.io.*;
import java.net.*;
public class UDPS
{
	public static void main(String[] args)
	{
		DatagramSocket skt=null;
		try
		{
			skt=new DatagramSocket(6788);
			byte[] buffer = new byte[1000];
			while(true)
			{
				DatagramPacket request = new DatagramPacket(buffer,buffer.length);
				skt.receive(request);
				String[] message = (new String(request.getData())).split(" ");
				byte[] sendMsg= (message[1]+ " server processed").getBytes();
				DatagramPacket reply = new DatagramPacket(sendMsg,sendMsg.length,request.getAddress(),request.getPort());
				skt.send(reply);
			}
		}
		catch(Exception ex)
		{
		}
	}
}
```

client program

```
import java.io.*;
import java.net.*;
public class UDPC
{
	public static void main(String[] args)
	{
		DatagramSocket skt;
		try
		{
			skt=new DatagramSocket();
```

```
                String msg= "text message ";
                byte[] b = msg.getBytes();
                InetAddress host=InetAddress.getByName("127.0.0.1");
                int serverSocket=6788;

                        DatagramPacket request =new DatagramPacket
(b,b.length,host,serverSocket);
                        skt.send(request);
                        byte[] buffer =new byte[1000];
                        DatagramPacket reply= new
DatagramPacket(buffer,buffer.length);
                        skt.receive(reply);
                        System.out.println("client received:" +new
String(reply.getData()));
                        skt.close();

            }
            catch(Exception ex)
            {
            }
        }
}
```

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

```
import java.util.*;
import java.util.Scanner;
class RSA{
   public static void main(String arg[]){
      Scanner in=new Scanner(System.in);
      long p,q,d,z,e,n,c;
      int choice;
      System.out.println("Enter two distinct prime numbers");
      p=in.nextLong();
      q=in.nextLong();
      n=p*q;
      z=(p-1)*(q-1);
      System.out.println("Enter a value for d which is less than and relatively prime to "+z);
      d=in.nextLong();

      for(e=1;e<z;++e)
      {
         if(((e*d)%z)==1)
            break;
      }

System.out.println("p="+p+"\nq="+q+"\nn="+n+"\nz="+z+"\nd="+d+"\ne="+e);
      do{
        System.out.println("1.Encription \n2.Decription\n3.Exit");
            System.out.println("choose an option");
            choice = in.nextInt();
            switch(choice){
               case 1:System.out.println("Enter a plain text");
                     String s = in.next();

                     System.out.println("Plain Text" +"\t" + "Cipher Text");
                     for(int i = 0; i < s.length(); i++){
                        long pl = (int) s.charAt(i);
                        c = modexp(pl,e,n);
                        System.out.println(s.charAt(i)+"\t\t" + c);
                     }
                     break;
               case 2: System.out.println("Enter a cipher text (0 to stop input)");
                      long [] ci = new long[50];
                      int j =0;
                      do{
                         ci[j] = in.nextLong();
                      }while(ci[j++] != 0);
```

```
                    System.out.println("Cipher Text" +"\t" + "Plain Text");
                    for(int i = 0; i < j-1; i++){
                        long pl = modexp(ci[i],d,n);
                        System.out.println(ci[i]+"\t\t" + (char) pl);
                    }
                    break;
                case 3: System.out.println("Program Terminated");
                    System.exit(0);

            }
        }while(choice != 3);
    }
    static long modexp(long a,long x,long n){
        long r=1;
        while(x>0){
            if(x%2==1){
                r=(r*a)%n;
            }
        a=(a*a)%n;
        x/=2;
        }
        return(r);
    }
}
```

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

```
import java.io.*;
import java.util.*;
class Queue
{
	int q[],f=0,r=0,size;
	void insert(int n)
	{
		Scanner in = new Scanner(System.in);
		q=new int[10];
		for(int i=0;i<n;i++)
		{
			System.out.print("\nEnter " + i + " element: ");
			int ele=in.nextInt();
			if(r+1>10)
			{
				System.out.println("\nQueue is full \nLost Packet: "+ele);
				break;
			}
			else
			{
				r++;
				q[i]=ele;
			}

	}
	void delete()
	{
		Scanner in = new Scanner(System.in);
		Thread t=new Thread();
		if(r==0)
			System.out.print("\nQueue empty ");
		else
		{
			for(int i=f;i<r;i++)
			{
				try
				{
					t.sleep(1000);
				}
				catch(Exception e){}
				System.out.print("\nLeaked Packet: "+q[i]);
				f++;
			}
		}
		System.out.println();
```

```
        }
}
class Leaky extends Thread
{
        public static void main(String ar[]) throws Exception
        {
                Queue q=new Queue();
                Scanner src=new Scanner(System.in);
                System.out.println("\nEnter the packets to be sent:");
                int size=src.nextInt();
                q.insert(size);
                q.delete();
        }
}
```