

JYOTHY INSTITUTE OF TECHNOLOGY

Tataguni, Off Kanakapura Road, Bengaluru – 560 082
APPROVED BY AICTE & AFFILIATED TO VTU, BELAGAVI



Department of
Computer Science & Engineering

Accredited by NBA, New Delhi

Lab Manual

COMPUTER NETWORK LABORATORY

18CSL57

Prepared By:

Dr. Swathi K
Associate Professor
Dept. of CSE, JIT,
Bengaluru

Institution Vision & Mission

Vision:

“To be an Institution of Excellence in Engineering education, Innovation and Research and work towards evolving great leaders for the country's future and meeting global needs.”

Mission:

The Institution aims at providing a vibrant, intellectually and emotionally rich teaching learning environment with the State of the Art Infrastructure and recognizing and nurturing the potential of each individual to evolve into ones own self and contribute to the welfare of all.

Department Vision & Mission

Vision:

“To be a center of excellence in Computer Science and Engineering education, focus on research, innovation and entrepreneurial skill development with professional competency.”

Mission:

M1: To provide state of the art ICT infrastructure and innovative, research-oriented teaching-learning environment and motivation for self-learning & problem-solving abilities by recruiting committed faculty.

M2: To encourage Industry Institute Interaction & multi-disciplinary approach for problem-solving and adapt to ever-changing global IT trends.

M3: To imbibe awareness of societal responsibility and leadership qualities with professional competency and ethics.

PART –A PROGRAMS

1. Implement three node point-to-point networks with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.

TCL:

```
#creates a new simulator
set ns [new Simulator]
set nf [open prog1.nam w]
$ns namtrace-all $nf

set nd [open prog1.tr w]
$ns trace-all $nd

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

#creating nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

#linking nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 512Kb 10ms DropTail
```

```
#setting queue size of the link
$ns queue-limit $n1 $n2 5

#creating a udp connection in network simulator
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

#set up CBR over udp
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0

set sink [new Agent/Null]
$ns attach-agent $n2 $sink
$ns connect $udp0 $sink

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

AWK Script:

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

```

}

if(event == "r")
{
rcount++;
}
}

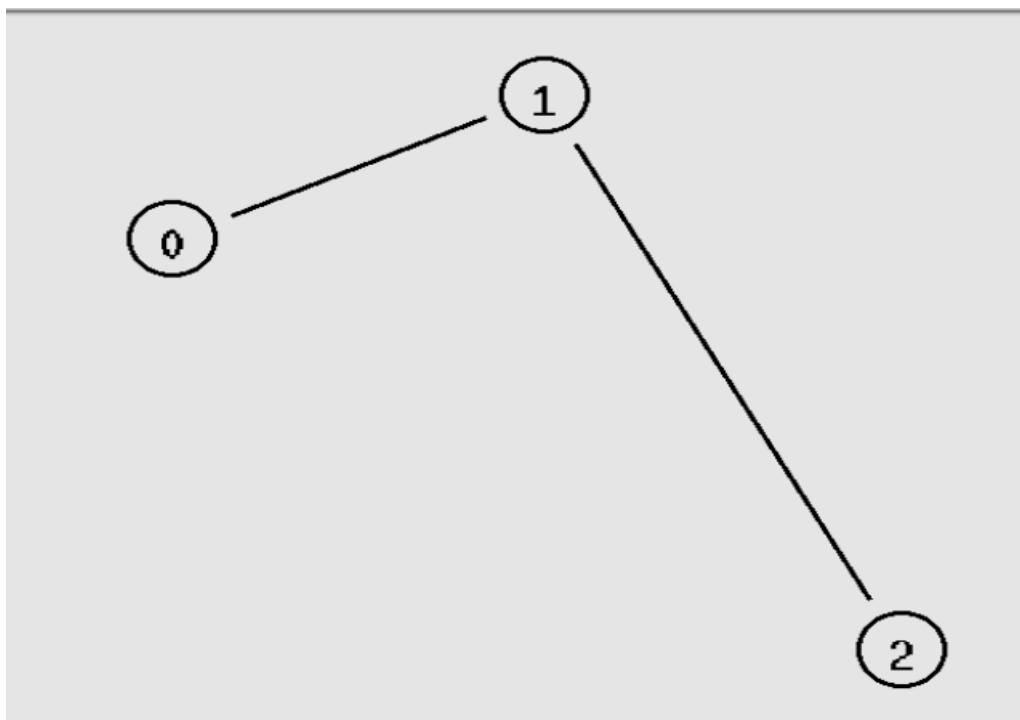
END {

printf("The no.of packets dropped : %d\n ",dcount);
printf("The no.of packets recieved : %d\n ",rcount);}

```

Steps for execution

- 1) Open vi editor and type program. Program name should have the extension “.tcl”
`[root@localhost ~]# vi lab1.tcl`
- 2) Save the program by pressing “ESC key” first, followed by “Shift and :” keys simultaneously and type “wq” and press **Enter key**.
- 3) Open vi editor and type **awk** program. Program name should have the extension “.awk”
`[root@localhost ~]# vi lab1.awk`
- 4) Save the program by pressing “ESC key” first, followed by “Shift and :” keys simultaneously and type “wq” and press **Enter key**.
- 5) Run the simulation program
`[root@localhost~]# ns lab1.tcl`
 - i) Here “ns” indicates network simulator. We get the topology shown in the snapshot.
 - ii) Now press the play button in the simulation window and the simulation will begins.
- 6) After simulation is completed run **awk file** to see the output ,
`[root@localhost~]# awk -f lab1.awk lab1.tr`
- 7) To see the trace file contents open the file as ,
`[root@localhost~]# vi lab1.tr`



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 lab2.tr w]
```

```
$ns trace-all $tf
```

```
set nf [open lab2.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 $n1 $n2 1Mb 300ms DropTail
```

```
$ns duplex-link $n3 $n2 1Mb 300ms DropTail
```

```
$ns duplex-link $n5 $n2 100Mb 300ms DropTail
```

```
$ns duplex-link $n2 $n4 1Mb 300ms DropTail
```

```
$ns duplex-link $n2 $n6 1Mb 300ms DropTail
```

```
$ns queue-limit $n0 $n2 5
```

```
$ns queue-limit $n2 $n4 3
```

```
$ns queue-limit $n2 $n6 2
```

```
$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
```

```
#Define a 'recv' function for the class 'Agent/Ping'
```

```
#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 lab2.nam &
```

```
$ns flush-trace
```

```
close $tf
```

```
close $nf
```

```
exit 0
```

```
}
```

```
#Schedule events
```

```
$ns at 0.1 "$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"
```

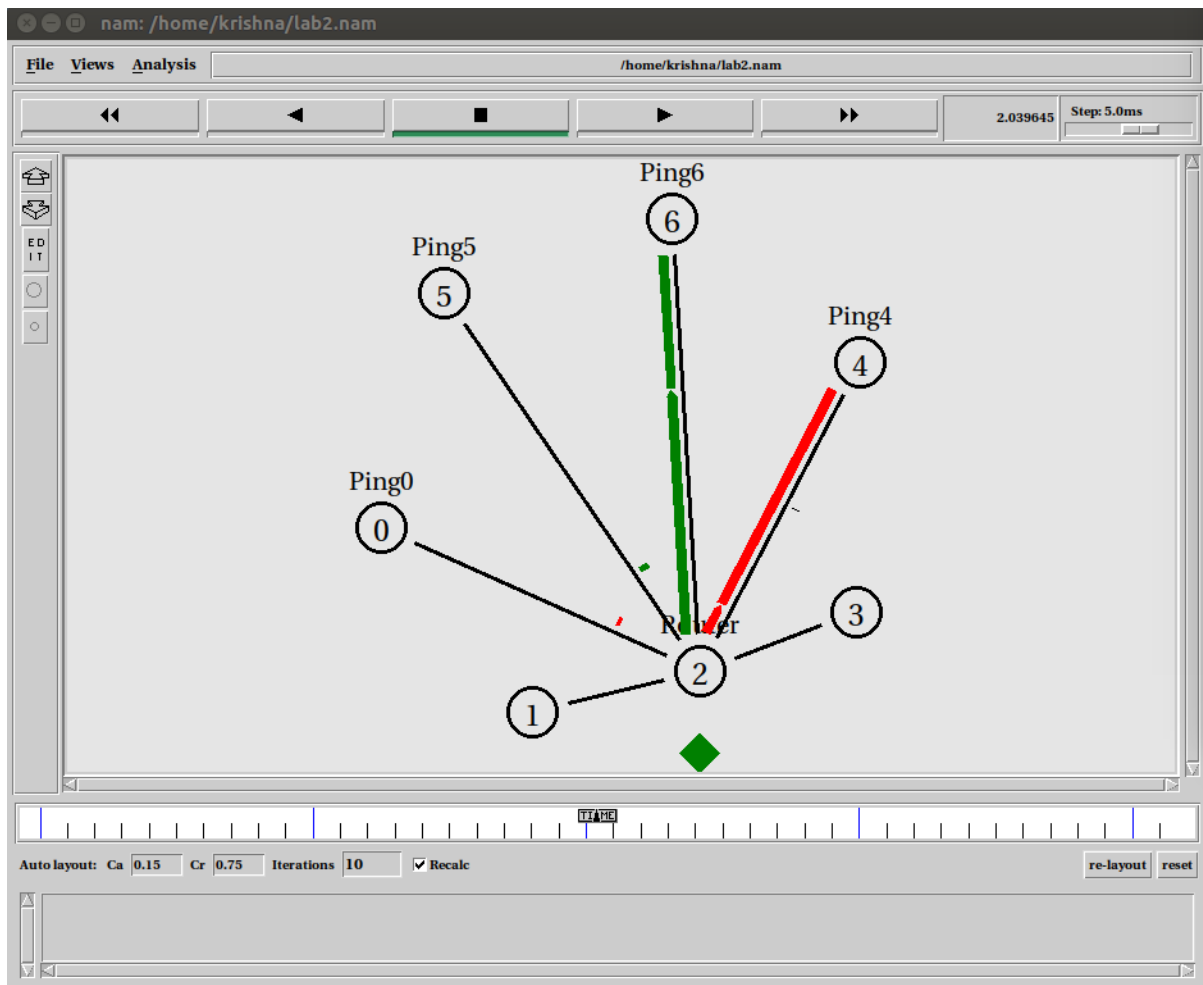
```
$ns at 5.0 "finish"
```

```
$ns run
```

AWK:

```
BEGIN{  
count=0;  
}  
{  
if($1=="d")  
count++;  
}  
END{  
printf("The Total no of Packets Drop is :%d\n\n", count);  
}
```

Topology:



Output:

```
krishna@ubuntu: ~
krishna@ubuntu:~$ vi lab2.tcl
krishna@ubuntu:~$ vi lab2.awk
krishna@ubuntu:~$ awk -f lab2.awk lab2.tr
The Total no of Packets Drop is :24
krishna@ubuntu:~$
```

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

```
#Make a NS simulator
set ns [new Simulator]
set tf [open lab3.tr w]
$ns trace-all $tf

set nf [open lab3.nam w]
$ns namtrace-all $nf

# Create the nodes,color and label
set n0 [$ns node]
$n0 color "magenta"
$n0 label "src1"
set n1 [$ns node]
$n1 color "red"
set n2 [$ns node]
$n2 color "magenta"
$n2 label "src2"
set n3 [$ns node]
$n3 color "blue"
$n3 label "dest2"
set n4 [$ns node]
$n4 shape square
set n5 [$ns node]
$n5 color "blue"
$n5 label "dest1"
```

#Creates a lan from a set of nodes given by <nodelist>. Bandwidth, delay
#characteristics along with the link-layer, Interface queue, Mac layer and
#channel type for the lan also needs to be defined.

```
$ns make-lan "$n0 $n1 $n2 $n3 $n4" 50Mb 100ms LL Queue/DropTail  
Mac/802_3
```

Create the link

```
$ns duplex-link $n4 $n5 1Mb 1ms DropTail
```

Create the node position

```
$ns duplex-link-op $n4 $n5 orient right
```

Add a TCP sending module to node n0

```
set tcp0 [new Agent/TCP]
```

```
$ns attach-agent $n0 $tcp0
```

Setup a FTP traffic generator on "tcp0"

```
set ftp0 [new Application/FTP]
```

```
$ftp0 attach-agent $tcp0
```

```
$ftp0 set packetSize_ 500
```

```
$ftp0 set interval_ 0.0001
```

Add a TCP receiving module to node n5

```
set sink0 [new Agent/TCPSink]
```

```
$ns attach-agent $n5 $sink0
```

Direct traffic from "tcp0" to "sink1"

```
$ns connect $tcp0 $sink0
```

Add a TCP sending module to node n2

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

```
$ns attach-agent $n2 $tcp1
```

```
# Setup a FTP traffic generator on "tcp1"
```

```
set ftp1 [new Application/FTP]
```

```
$ftp1 attach-agent $tcp1
```

```
$ftp1 set packetSize_ 600
```

```
$ftp1 set interval_ 0.001
```

```
# Add a TCP receiving module to node n3
```

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

```
$ns attach-agent $n3 $sink1
```

```
# Direct traffic from "tcp1" to "sink1"
```

```
$ns connect $tcp1 $sink1
```

```
set file1 [open file1.tr w]
```

```
$tcp0 attach $file1
```

```
set file2 [open file2.tr w]
```

```
$tcp1 attach $file2
```

```
$tcp0 trace cwnd_
```

```
$tcp1 trace cwnd_
```

```
# Define a 'finish' procedure
```

```
proc finish { } {
```

```
global ns nf tf
```

```
$ns flush-trace
```

```
close $tf
```

```
close $nf
```

```
exec nam lab3.nam &
```

```
exit 0
```

```
}
```

Schedule start/stop times

\$ns at 0.1 "\$ftp0 start"

\$ns at 5 "\$ftp0 stop"

\$ns at 7 "\$ftp0 start"

\$ns at 0.2 "\$ftp1 start"

\$ns at 8 "\$ftp1 stop"

\$ns at 14 "\$ftp0 stop"

\$ns at 10 "\$ftp1 start"

\$ns at 15 "\$ftp1 stop"

Set simulation end time

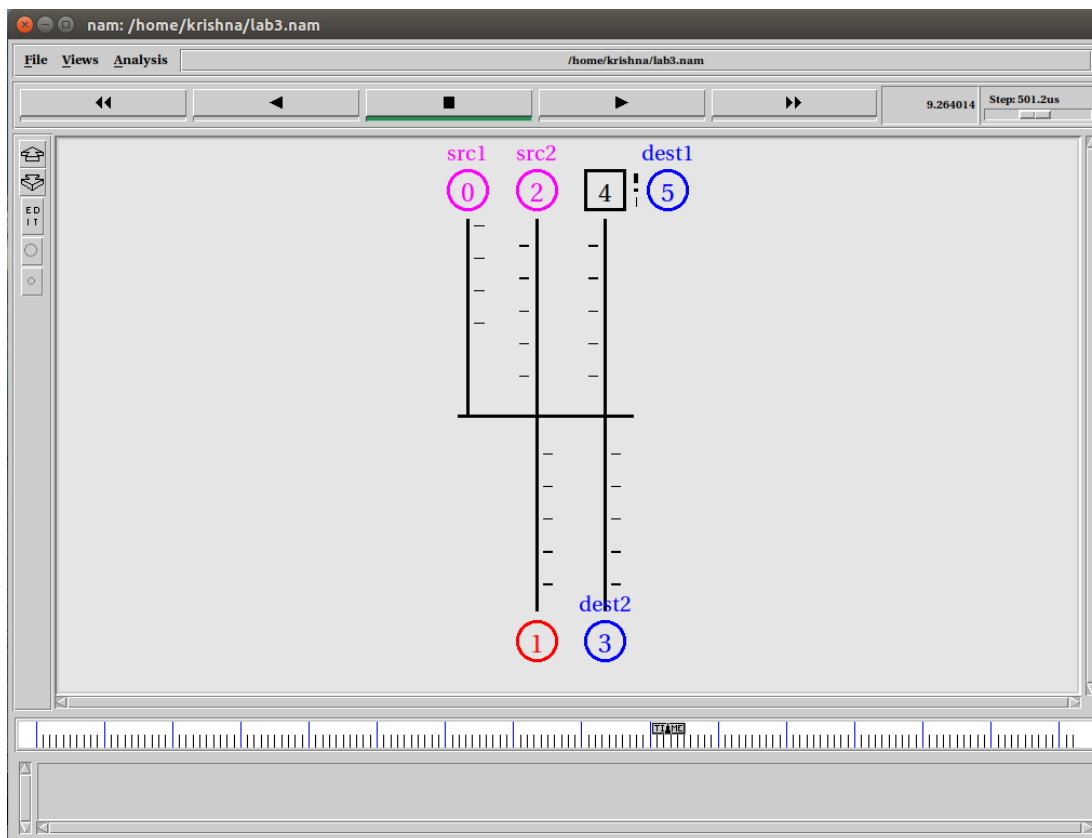
\$ns at 16 "finish"

\$ns run

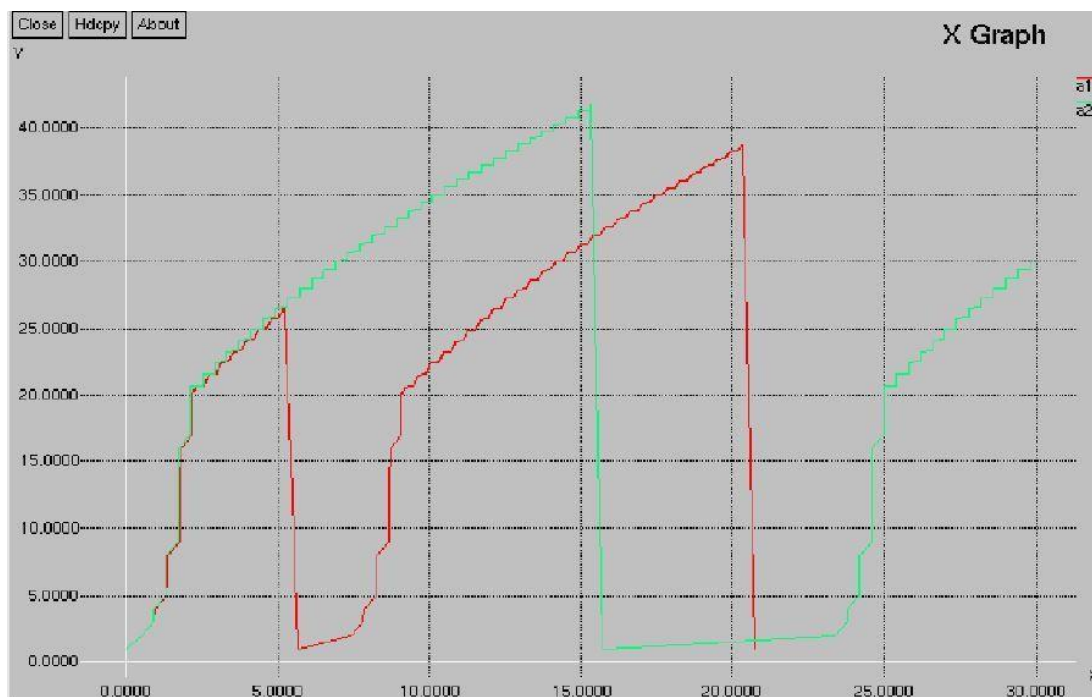
AWK:

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

Topology:



Output: xgraph



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

```
# Create a NS simulator object
```

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

```
#setup trace support by opening file lab4.tr and call the procedure trace-all
```

```
set tf [open lab4.tr w]
```

```
$ns trace-all $tf
```

```
#create a topology object that keeps track of movements of mobile nodes
```

```
#within the topological boundary.
```

```
set topo [new Topography]
```

```
$topo load_flatgrid 1000 1000
```

```
set nf [open lab4.nam w]
```

```
$ns namtrace-all-wireless $nf 1000 1000
```

```
# creating a wireless node you MUST first select (configure) the node
```

```
#configuration parameters to "become" a wireless node.
```

```
$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 object  
create-god 3
```

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

```
$n0 label "tcp0"  
$n1 label "sink1/tcp1"  
$n2 label "sink2"
```

```
$n0 set X_ 50  
$n0 set Y_ 50  
$n0 set Z_ 0
```

```
$n1 set X_ 100  
$n1 set Y_ 100  
$n1 set Z_ 0
```

```
$n2 set X_ 600  
$n2 set Y_ 600  
$n2 set Z_ 0
```

```
$ns at 0.1 "$n0 setdest 50 50 15"  
$ns at 0.1 "$n1 setdest 100 100 25"
```

```
$ns at 0.1 "$n2 setdest 600 600 25"
```

```
set tcp0 [new Agent/TCP]
```

```
$ns attach-agent $n0 $tcp0
```

```
set ftp0 [new Application/FTP]
```

```
$ftp0 attach-agent $tcp0
```

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

```
$ns attach-agent $n1 $sink1
```

```
$ns connect $tcp0 $sink1
```

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

```
$ns attach-agent $n1 $tcp1
```

```
set ftp1 [new Application/FTP]
```

```
$ftp1 attach-agent $tcp1
```

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

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

```
$ns connect $tcp1 $sink2
```

```
$ns at 5 "$ftp0 start"
```

```
$ns at 5 "$ftp1 start"
```

```
$ns at 100 "$n1 setdest 550 550 15"
```

```
$ns at 190 "$n1 setdest 70 70 15"
```

```
proc finish { } {  
    global ns nf tf  
    $ns flush-trace  
    exec nam lab4.nam &  
    close $tf  
    exit 0  
}  
$ns at 250 "finish"  
$ns run
```

AWK

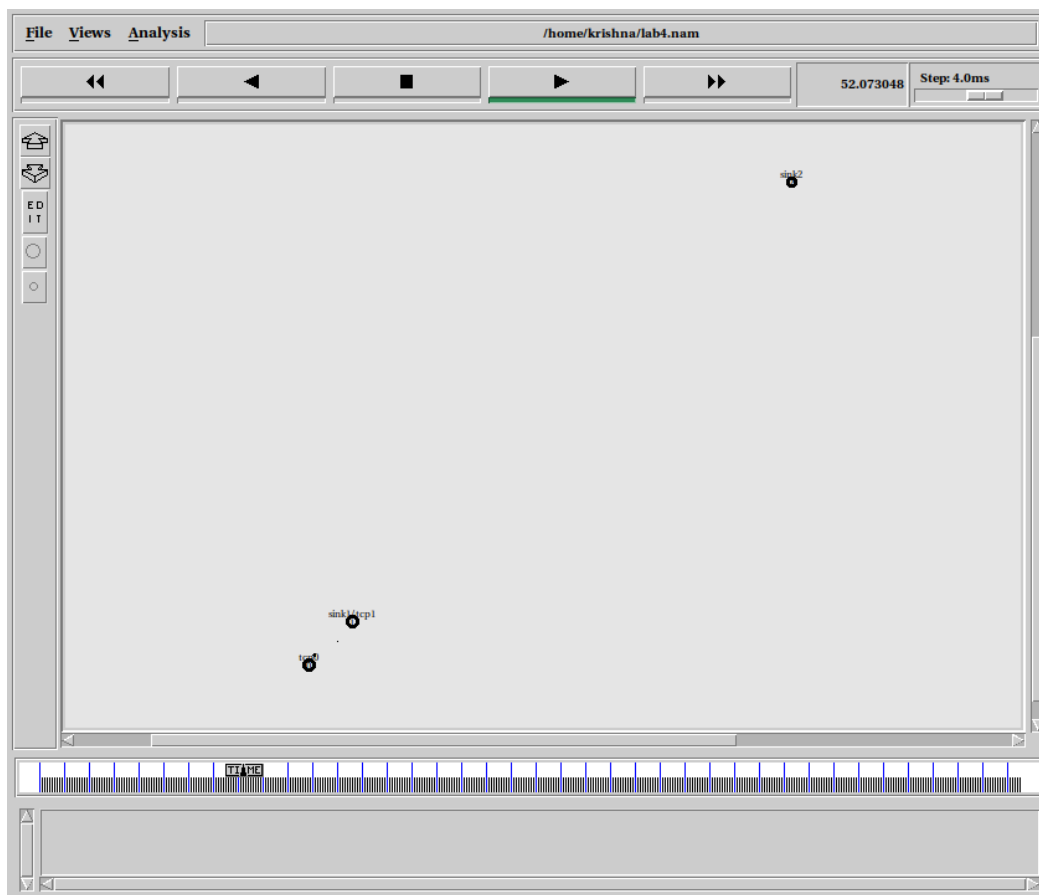
```
BEGIN{  
    count1=0  
    count2=0  
    pack1=0  
    pack2=0  
    time1=0  
    time2=0  
}  
{  
    if($1 == "r" && $3 == "_1_" && $4 == "AGT")  
    {  
        count1++  
        pack1=pack1+$8  
        time1=$2  
    }  
    if($1 == "r" && $3 == "_2_" && $4 == "AGT")  
    {  
        count2++
```

```

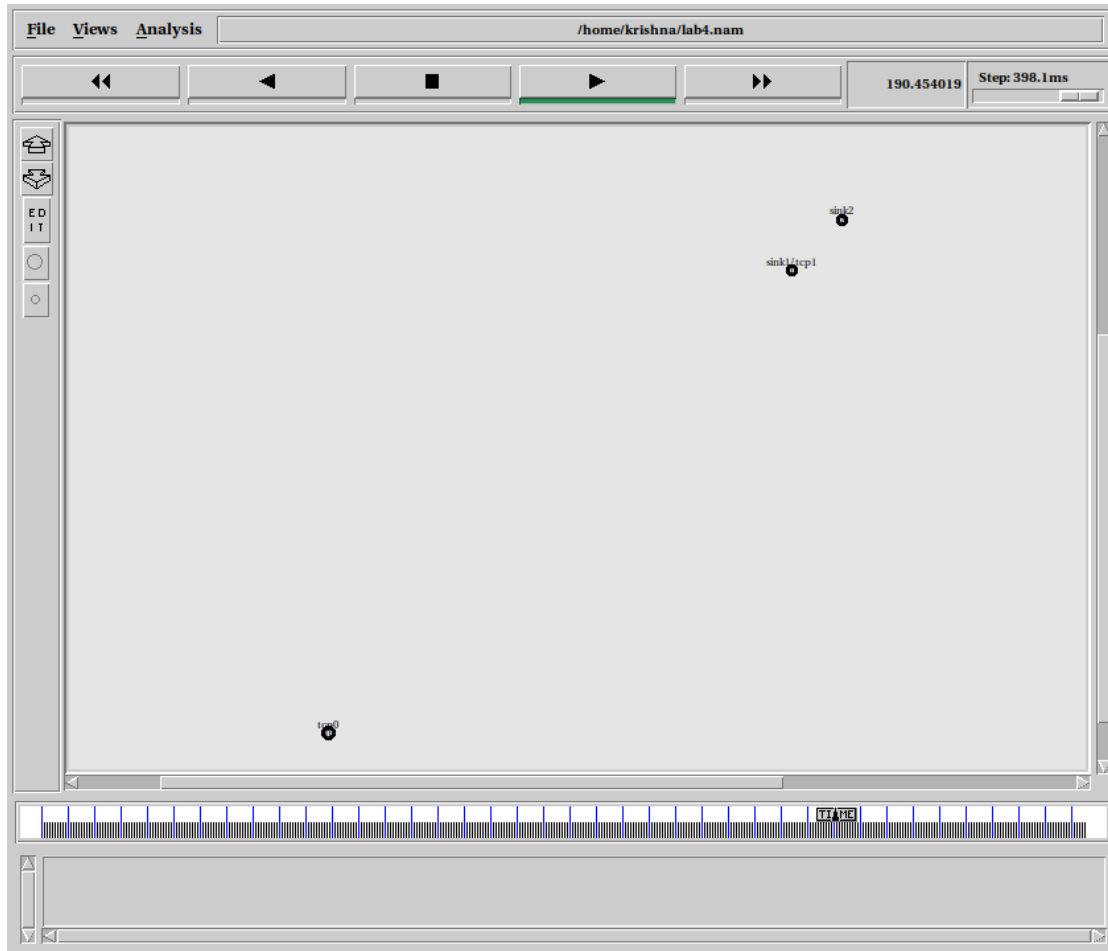
pack2=pack2+$8
time2=$2
}
}
END{
printf("The Throughput from n0 to n1: %f Mbps \n",
((count1*pack1*8)/(time1*1000000)));
printf("The Throughput from n1 to n2: %f Mbps \n",
((count2*pack2*8)/(time2*1000000)));
}

```

Topology:



Topology:



Output:

```
krishna@ubuntu:~$ vi lab4.awk
krishna@ubuntu:~$ awk -f lab4.awk lab4.tr
The Throughput from n0 to n1: 5863.442245 Mbps
The Throughput from n1 to n2: 1307.611834 Mbps
krishna@ubuntu:~$
```

The screenshot shows a terminal window with the following output:

```
krishna@ubuntu:~$ vi lab4.awk
krishna@ubuntu:~$ awk -f lab4.awk lab4.tr
The Throughput from n0 to n1: 5863.442245 Mbps
The Throughput from n1 to n2: 1307.611834 Mbps
krishna@ubuntu:~$
```

The terminal window also shows a file explorer view with the following files and folders:

- lab3.tcl
- lab3.tr
- lab4.awk
- lab4.nam
- lab4.tcl
- lab4.tr
- ns-allmone-2.35
- src-483.tcl
- qt

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

source web.tcl

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

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

proc stop {} {
    global nodes opt tf
    set wrap $opt(wrap)

```

```
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
set a "out.tr"

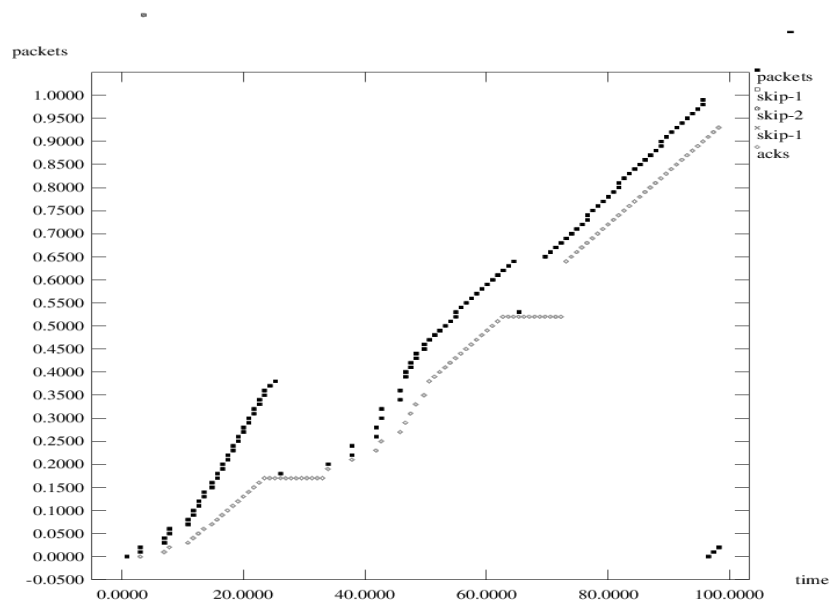
set GETRC "../..../bin/getrc"
set RAW2XG "../..../bin/raw2xg"

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

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

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

Output:



GSM Trace File:

```

Open ▾  Save
+ 0.8 0 3 tcp 40 ----- 0 0.0 4.0 0 0
- 0.8 0 3 tcp 40 ----- 0 0.0 4.0 0 0
r 0.850107 0 3 tcp 40 ----- 0 0.0 4.0 0 0
+ 0.850107 3 1 tcp 40 ----- 0 0.0 4.0 0 0
- 0.850107 3 1 tcp 40 ----- 0 0.0 4.0 0 0
r 1.38344 3 1 tcp 40 ----- 0 0.0 4.0 0 0
+ 1.38344 1 2 tcp 40 ----- 0 0.0 4.0 0 0
- 1.38344 1 2 tcp 40 ----- 0 0.0 4.0 0 0
r 1.916773 1 2 tcp 40 ----- 0 0.0 4.0 0 0
+ 1.916773 2 4 tcp 40 ----- 0 0.0 4.0 0 0
- 1.916773 2 4 tcp 40 ----- 0 0.0 4.0 0 0
r 1.92688 2 4 tcp 40 ----- 0 0.0 4.0 0 0
+ 1.92688 4 2 ack 40 ----- 0 4.0 0.0 0 1
- 1.92688 4 2 ack 40 ----- 0 4.0 0.0 0 1
r 1.936987 4 2 ack 40 ----- 0 4.0 0.0 0 1
+ 1.936987 2 1 ack 40 ----- 0 4.0 0.0 0 1
- 1.936987 2 1 ack 40 ----- 0 4.0 0.0 0 1
r 2.47032 2 1 ack 40 ----- 0 4.0 0.0 0 1
+ 2.47032 1 3 ack 40 ----- 0 4.0 0.0 0 1
- 2.47032 1 3 ack 40 ----- 0 4.0 0.0 0 1
r 3.003653 1 3 ack 40 ----- 0 4.0 0.0 0 1
+ 3.003653 3 0 ack 40 ----- 0 4.0 0.0 0 1
- 3.003653 3 0 ack 40 ----- 0 4.0 0.0 0 1
r 3.05376 3 0 ack 40 ----- 0 4.0 0.0 0 1
+ 3.05376 0 3 tcp 1040 ----- 0 0.0 4.0 1 2
- 3.05376 0 3 tcp 1040 ----- 0 0.0 4.0 1 2
+ 3.05376 0 3 tcp 1040 ----- 0 0.0 4.0 2 3
- 3.056533 0 3 tcp 1040 ----- 0 0.0 4.0 2 3
r 3.106533 0 3 tcp 1040 ----- 0 0.0 4.0 1 2
+ 3.106533 3 1 tcp 1040 ----- 0 0.0 4.0 1 2
- 3.106533 3 1 tcp 1040 ----- 0 0.0 4.0 1 2
r 3.109307 0 3 tcp 1040 ----- 0 0.0 4.0 2 3
+ 3.109307 3 1 tcp 1040 ----- 0 0.0 4.0 2 3
- 3.9732 3 1 tcp 1040 ----- 0 0.0 4.0 2 3
r 4.4732 3 1 tcp 1040 ----- 0 0.0 4.0 1 2
+ 4.4732 1 2 tcp 1040 ----- 0 0.0 4.0 1 2
- 4.4732 1 2 tcp 1040 ----- 0 0.0 4.0 1 2
Plain Text ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

```

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

set stop 100 ;

```
set type cdma ;
```

```
set minth 30
```

```
set maxth 0
```

```
set adaptive 1 ;
```

```
set flows 0 ;
```

```
set window 30 ;
```

```
set opt(wrap) 100 ;
```

```
set opt(srcTrace) is ;
```

```
set opt(dstTrace) bs2 ;
```

```
set bwDL(cdma) 384000
```

```
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
}

```

```

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

```

```

source web.tcl

```

```

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]

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 ".././../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 xgraph -x time -y packets plot.xgr &

```

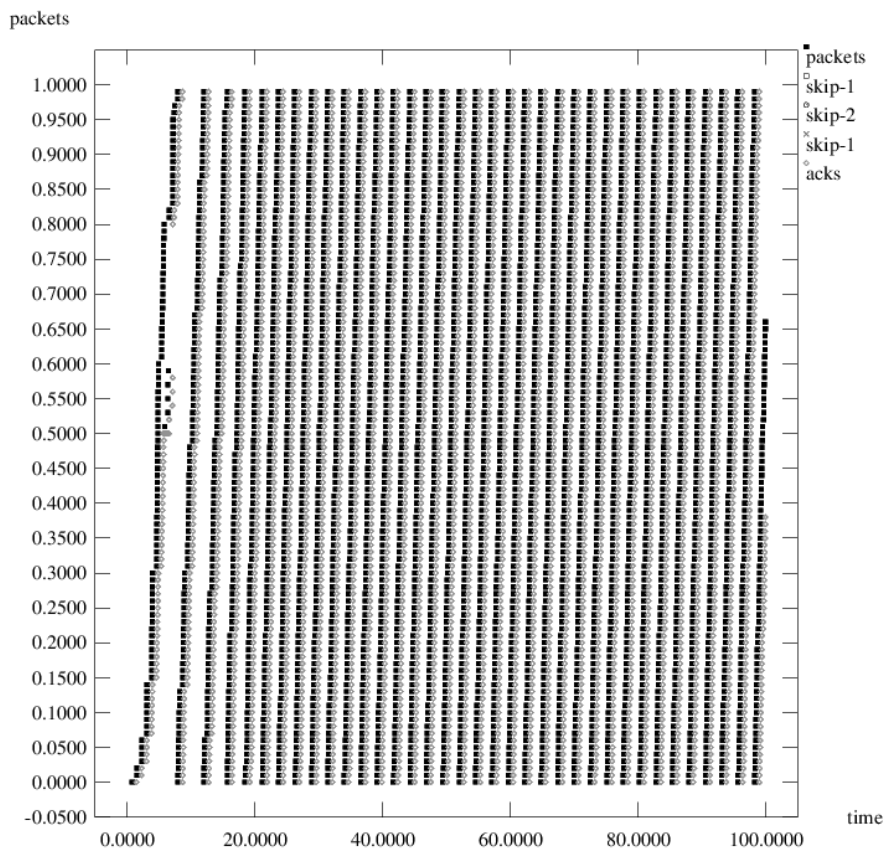
```
exit 0
```

```
}
```

```
$ns at $stop "stop"
```

```
$ns run
```

Output:



CDMA Trace File:

Open
+
Save

```

+ 0.8 0 3 tcp 40 ----- 0 0.0 4.0 0 0
- 0.8 0 3 tcp 40 ----- 0 0.0 4.0 0 0
r 0.850107 0 3 tcp 40 ----- 0 0.0 4.0 0 0
+ 0.850107 3 1 tcp 40 ----- 0 0.0 4.0 0 0
- 0.850107 3 1 tcp 40 ----- 0 0.0 4.0 0 0
r 1.00094 3 1 tcp 40 ----- 0 0.0 4.0 0 0
+ 1.00094 1 2 tcp 40 ----- 0 0.0 4.0 0 0
- 1.00094 1 2 tcp 40 ----- 0 0.0 4.0 0 0
r 1.151773 1 2 tcp 40 ----- 0 0.0 4.0 0 0
+ 1.151773 2 4 tcp 40 ----- 0 0.0 4.0 0 0
- 1.151773 2 4 tcp 40 ----- 0 0.0 4.0 0 0
r 1.16188 2 4 tcp 40 ----- 0 0.0 4.0 0 0
+ 1.16188 4 2 ack 40 ----- 0 4.0 0.0 0 1
- 1.16188 4 2 ack 40 ----- 0 4.0 0.0 0 1
r 1.171987 4 2 ack 40 ----- 0 4.0 0.0 0 1
+ 1.171987 2 1 ack 40 ----- 0 4.0 0.0 0 1
- 1.171987 2 1 ack 40 ----- 0 4.0 0.0 0 1
r 1.32282 2 1 ack 40 ----- 0 4.0 0.0 0 1
+ 1.32282 1 3 ack 40 ----- 0 4.0 0.0 0 1
- 1.32282 1 3 ack 40 ----- 0 4.0 0.0 0 1
r 1.473653 1 3 ack 40 ----- 0 4.0 0.0 0 1
+ 1.473653 3 0 ack 40 ----- 0 4.0 0.0 0 1
- 1.473653 3 0 ack 40 ----- 0 4.0 0.0 0 1
r 1.52376 3 0 ack 40 ----- 0 4.0 0.0 0 1
+ 1.52376 0 3 tcp 1040 ----- 0 0.0 4.0 1 2
- 1.52376 0 3 tcp 1040 ----- 0 0.0 4.0 1 2
+ 1.52376 0 3 tcp 1040 ----- 0 0.0 4.0 2 3
- 1.526533 0 3 tcp 1040 ----- 0 0.0 4.0 2 3
r 1.576533 0 3 tcp 1040 ----- 0 0.0 4.0 1 2
+ 1.576533 3 1 tcp 1040 ----- 0 0.0 4.0 1 2
- 1.576533 3 1 tcp 1040 ----- 0 0.0 4.0 1 2
r 1.579307 0 3 tcp 1040 ----- 0 0.0 4.0 2 3
+ 1.579307 3 1 tcp 1040 ----- 0 0.0 4.0 2 3
- 1.5982 3 1 tcp 1040 ----- 0 0.0 4.0 2 3
r 1.7482 3 1 tcp 1040 ----- 0 0.0 4.0 1 2
+ 1.7482 1 2 tcp 1040 ----- 0 0.0 4.0 1 2
- 1.7482 1 2 tcp 1040 ----- 0 0.0 4.0 1 2

```

Plain Text
Tab Width: 8
Ln 29, Col 46
INS

PART-B

Program 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-1; div=new
int[tot_length];

    rem=new int[tot_length];
    crc=new int[tot_length];

    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++)
    {
        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]);
    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");
    }
}

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;
} }

```

Output:



```

krishna@ubuntu:~$ javac Crc.java
krishna@ubuntu:~$ java Crc
Enter number of data bits :
7
Enter data bits :
1
0
1
0
0
1
0
1
Enter number of bits in divisor :
3
Enter Divisor bits :
1
0
1
Dividend (after appending 0's) are : 101100100
CRC code :
101100111
Enter CRC code of 9 bits :
1
0
1
1
0
0
1
1
1
No Error

```

Explanation

Error Detection

Data transmission can contain errors

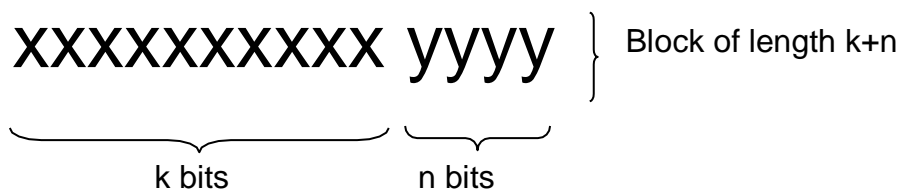
- Single-bit
- Burst errors of length n
(n : distance between the first and last errors in data block)

How to detect errors

- If only data is transmitted, errors cannot be detected
 - ✓ Send more information with data that satisfies a special relationship
 - ✓ Add redundancy

What is Cyclic Redundancy Check?

- Powerful error detection scheme
- Rather than addition, binary division is used
- Can be easily implemented with small amount of hardware
 - ✓ Shift registers
 - ✓ XOR (for addition and subtraction)
- Let us assume k message bits and n bits of redundancy



- Associate bits with coefficients of a polynomial

$$\begin{aligned}
 &1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1 \\
 &1x^6 + 0x^5 + 1x^4 + 1x^3 + 0x^2 + 1x + 1 \\
 &= x^6 + x^4 + x^3 + x + 1
 \end{aligned}$$
- Let $M(x)$ be the **message polynomial**
- Let $P(x)$ be the **generator polynomial**
- $P(x)$ is fixed for a given CRC scheme
- $P(x)$ is known both by sender and receiver
- Create a block polynomial $F(x)$ based on $M(x)$ and $P(x)$ such that $F(x)$ is divisible by $P(x)$

Sending

1. Multiply $M(x)$ by x^n
2. Multiply $P(x)$ by x^n
3. Divide $x^n M(x)$ by $P(x)$
4. Ignore the quotient and keep the remainder $C(x)$
5. Form and send $F(x) = x^n M(x) + C(x)$

Receiving

1. Receive $F'(x)$
2. Divide $F'(x)$ by $P(x)$
3. Accept if remainder is 0, reject otherwise

Example

• Send

- $M(x) = 110011 \rightarrow x^5 + x^4 + x + 1$ (6 bits)
- $P(x) = 11001 \rightarrow x^4 + x^3 + 1$ (5 bits, $n = 4$)
→ 4 bits of redundancy
- Form $x^n M(x) \rightarrow 110011 \underline{0000}$
→ $x^9 + x^8 + x^5 + x^4$
- Divide $x^n M(x)$ by $P(x)$ to find $C(x)$

$$\begin{array}{r}
 100001 \\
 11001 \overline{) 1100110000} \\
 \underline{11001} \\
 10000 \\
 \underline{11001} \\
 1001 = C(x)
 \end{array}$$

Send the block 110011 1001

• Receive

$$\begin{array}{r}
 11001 \overline{) 1100111001} \\
 \underline{11001} \\
 11001 \\
 \underline{11001} \\
 00000 \\
 \downarrow \\
 \text{No remainder} \\
 \rightarrow \text{Accept}
 \end{array}$$

1 0 1 0 1 1 0 0

Most
significant bit:
First bit

Least
significant bit:
Last bit

Least
significant bit:
Last bit

Most
significant bit:
First bit

Create/ Compile/ Run

- `gedit filename.java` → Create
- `javac filename.java` → Compile
- `java filename` → Run

Program 2:**Shortest path between vertices using bellman-ford algorithm**

```
import java.util.Scanner;

public class BellmanFord
{
    private int distances[ ];
    private int numberofvertices;
    public static final int MAX_VALUE = 999;

    public BellmanFord(int numberofvertices)
    {
        this.numberofvertices = numberofvertices;
        distances = new int[numberofvertices + 1];
    }

    public void BellmanFordEvaluation(int source, int adjacencymatrix[ ][ ])
    {
        for (int node = 1; node <= numberofvertices; node++)
        {
            distances[node] = MAX_VALUE;
        }

        distances[source] = 0;

        for (int node = 1; node <= numberofvertices - 1; node++)
```

```

{
    for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
    {
        for (int destinationnode = 1; destinationnode <= numberofvertices;
destinationnode++)
        {
            if (adjacencymatrix[sourcenode][destinationnode] != MAX_VALUE)
            {
                if (distances[destinationnode] > distances[sourcenode] +
adjacencymatrix[sourcenode][destinationnode])
            {
                distances[destinationnode] = distances[sourcenode] +
adjacencymatrix[sourcenode][destinationnode];
            }
        }
    }
}

for (int vertex = 1; vertex <= numberofvertices; vertex++)
{
    System.out.println("distance of source " + source + " to " + vertex + " is " +
distances[vertex]);
}
}

public static void main(String[ ] args)
{
    int numberofvertices = 0;
    int source, destination;
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter the number of vertices");
    numberofvertices = scanner.nextInt();
    int adjacencymatrix[ ][ ] = new int[numberofvertices + 1][numberofvertices + 1];
    System.out.println("Enter the adjacency matrix");

```

```

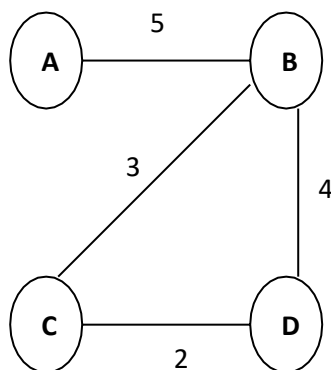
for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
{
    for (int destinationnode = 1; destinationnode <= numberofvertices;
destinationnode++)
    {
        adjacencymatrix[sourcenode][destinationnode] = scanner.nextInt();
        if (sourcenode == destinationnode)
        {
            adjacencymatrix[sourcenode][destinationnode] = 0;
            continue;
        }

        if (adjacencymatrix[sourcenode][destinationnode] == 0)
        {
            adjacencymatrix[sourcenode][destinationnode] = MAX_VALUE;
        }
    }
}

System.out.println("Enter the source vertex");
source = scanner.nextInt();
BellmanFord bellmanford = new BellmanFord(numberofvertices);
bellmanford.BellmanFordEvaluation(source,adjacencymatrix);
scanner.close();
}
}

```

Input Graph:

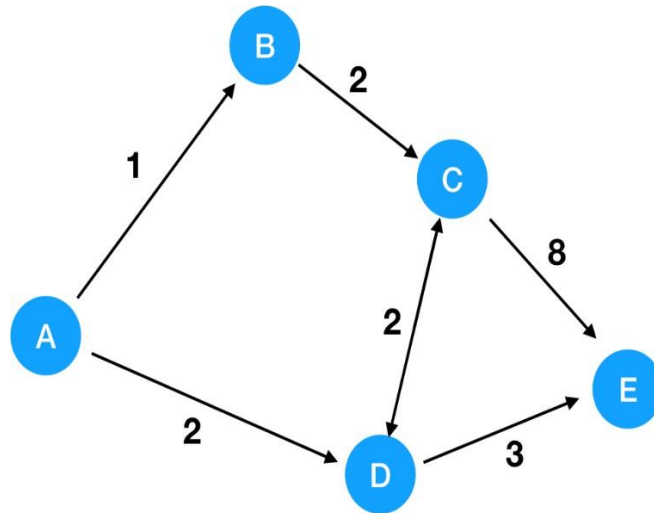


Output:

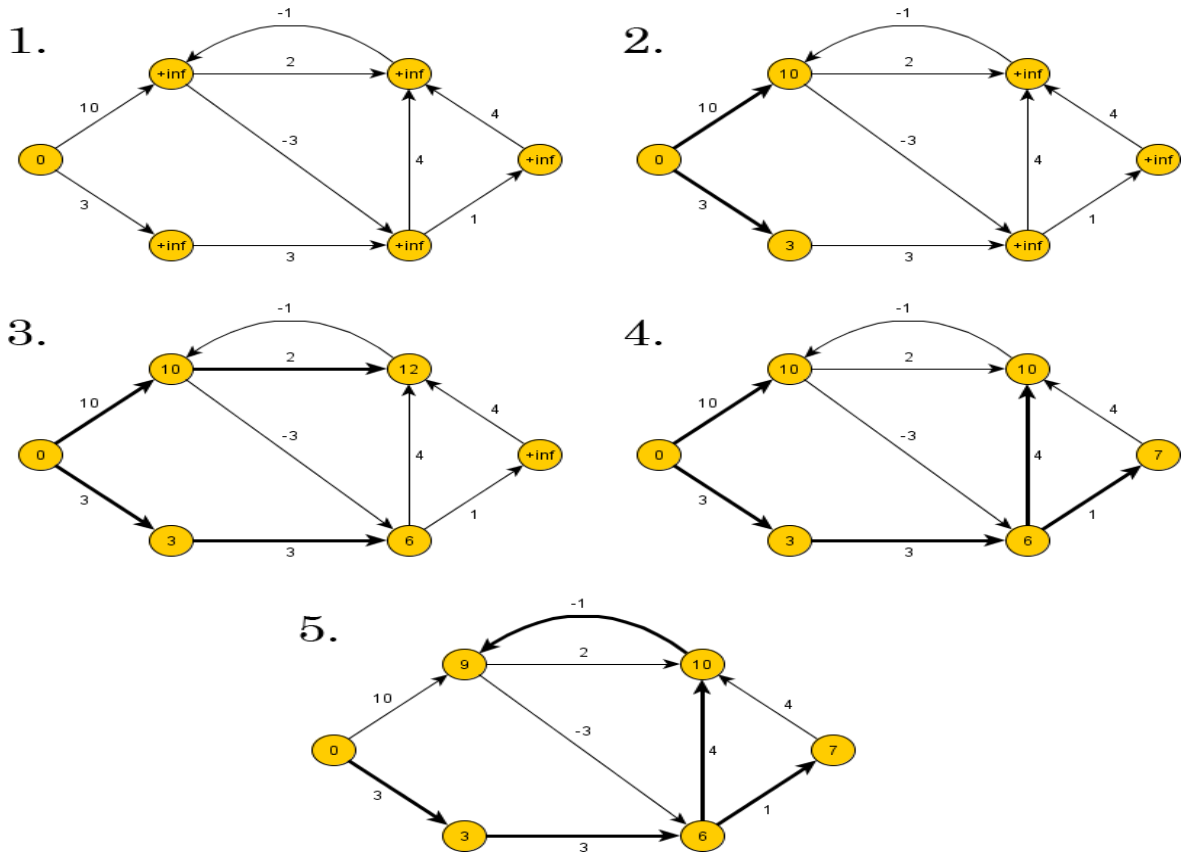
```
krishna@ubuntu:~$ javac BellmanFord.java
krishna@ubuntu:~$ java BellmanFord
Enter the number of vertices
4
Enter the adjacency matrix
0 5 0 0
5 0 3 4
0 3 0 2
0 4 2 0
Enter the source vertex
2
distance of source 2 to 1 is 5
distance of source 2 to 2 is 0
distance of source 2 to 3 is 3
distance of source 2 to 4 is 4
krishna@ubuntu:~$
```

Explanation**Bellman-Ford**

- Shortest path between vertices using bellman-ford algorithm.
- It computes shortest paths from a single source vertex to all of the other vertices in a weighted digraph.
- It is slower than Dijkstra's algorithm for the same problem, but more versatile, as it is capable of handling graphs in which some of the edge weights are negative numbers.
- The algorithm was first proposed by Alfonso Shimbel (1955), but is instead named after Richard Bellman and Lester Ford, Jr., who published it in 1958 and 1956, respectively.
- Edward F. Moore also published the same algorithm in 1957, and for this reason it is also sometimes called the Bellman–Ford–Moore algorithm.
- Bellman–Ford proceeds by relaxation, in which approximations to the correct distance are replaced by better ones until they eventually reach the solution.



node	shortest cost from A
A	0
B	1
C	3
D	2
E	5



Program 3:

TCP/IP sockets, 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 Program

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

public class ContentsClient
{
    public static void main( String args[ ] ) throws Exception {

        Socket sock = new Socket("127.0.0.1", 4000);

        // reading the file name from keyboard. Uses input
        stream    System.out.print("Enter the file name");
        BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
        String fname = keyRead.readLine();

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

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

        BufferedReader socketRead = new BufferedReader(new InputStreamReader(istream));

        String str;
        while((str = socketRead.readLine()) != null) // reading line-by-line
        {
            System.out.println(str);
        }
        pwrite.close();
        socketRead.close();
    }
}
```

```

        keyRead.close();
    }
}

```

Server Program

```

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

public class ContentsServer
{
    public static void main(String args[]) throws Exception {

        // establishing the connection with the server
        ServerSocket sersock = new ServerSocket(4000);
        System.out.println("Server ready for
        connection");

        Socket sock = sersock.accept(); // binding with port: 4000
        System.out.println("Connection is successful and wating for client request");

        // reading the file name from client
        InputStream istream = sock.getInputStream(
        );
        BufferedReader fileRead =new BufferedReader(new
        InputStreamReader(istream)); String fname = fileRead.readLine( );

        // reading file contents
        BufferedReader contentRead = new BufferedReader(new FileReader(fname) );

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

        String str;

        while((str = contentRead.readLine()) != null) // reading line-by-line from file

```

```

{

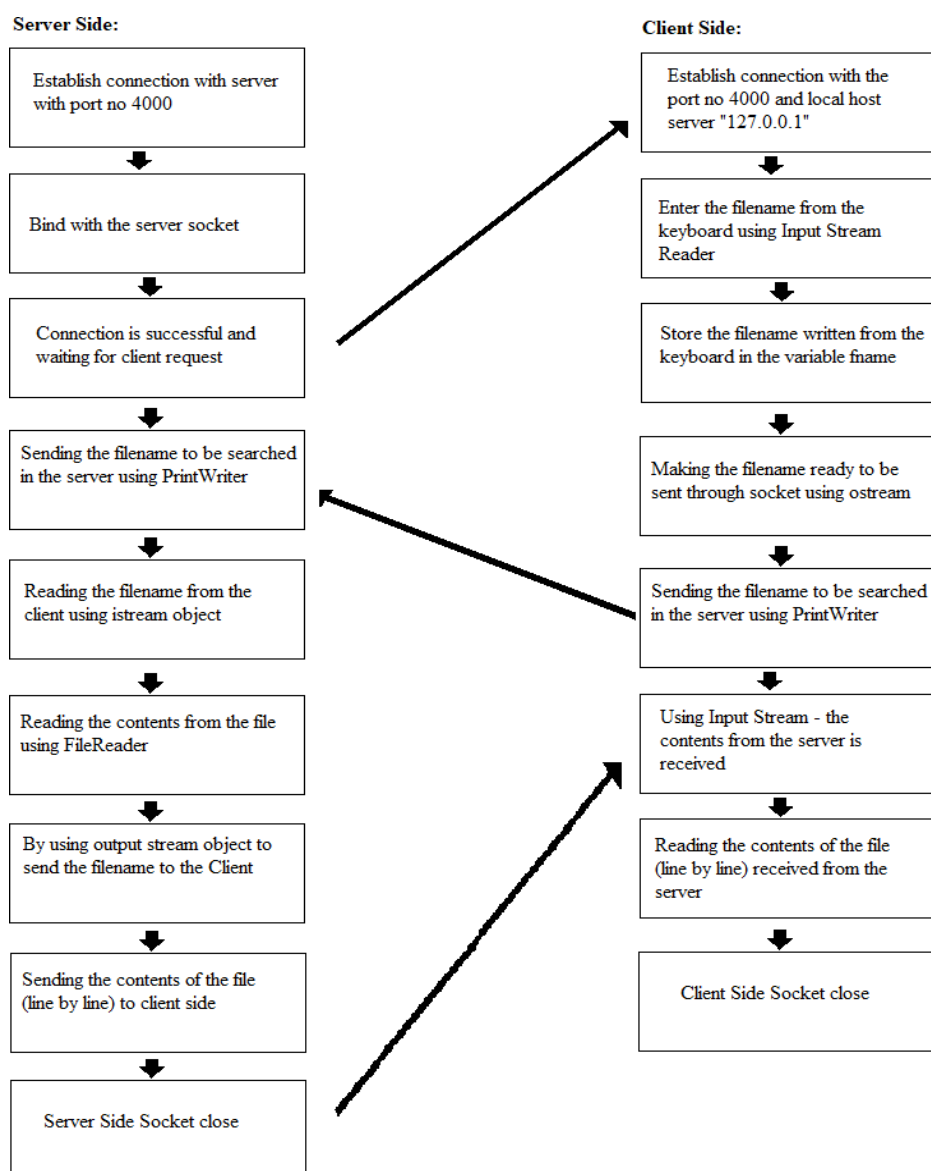
    pwrite.println(str); // sending each line to client
}

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

}
}

```

Explanation



Output:**In the server side terminal**

```
javac server.java
```

```
java server
```

```
>Server ready for connection
```

```
>Connection is successful and waiting for client request
```

In the client side terminal

```
javac client.java
```

```
java client
```

```
Enter the file name: swathi.txt
```

```
>swathi k, Dept. of CSE, JIT
```

Program 4:

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

Sender.java

```
import java.io.*;
import java.net.*;
public class Sender
{
    public static void main(String[] args) throws IOException
    {
        InetAddress addr = InetAddress.getByName(args[0]);
        byte[] buf = args[1].getBytes();
        DatagramPacket packet = new DatagramPacket(buf, buf.length, addr, 4444);
        DatagramSocket socket = new DatagramSocket();
        socket.send(packet);
    }
}
```

Receiver.java

```
import java.io.*;
import java.net.*;
public class Receiver
{
    public static void main(String[] args) throws IOException
    {
        DatagramSocket socket = new DatagramSocket(4444);
        byte[] buf = new byte[256];
        DatagramPacket packet = new DatagramPacket(buf, buf.length);
        System.out.println("Waiting ...");
        socket.receive(packet);
    }
}
```

```
String s = new String(packet.getData(), 0, packet.getLength());  
System.out.println(packet.getAddress().getHostName() + ": " + s);  
}  
}
```

- Compile the program.
- Start the receiver by running "java Receiver".
- Assuming that the receiver is running on a host with IP address 127.0.0.1
Start the sender by running:

```
java Sender 127.0.0.1 "My String"
```

- The receiver program should now display the string "My String".
- Repeat this exercise, with the difference, that you run the sender and receiver on two different hosts.

Output:

```
krishna@ubuntu:~$ javac Sender.java  
krishna@ubuntu:~$ java Sender 127.0.0.1 "Hello Ubuntu"  
krishna@ubuntu:~$
```

```
krishna@ubuntu:~$ javac Receiver.java  
krishna@ubuntu:~$ java Receiver  
Waiting ...  
localhost: Hello Ubuntu  
krishna@ubuntu:~$
```

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

```

import java.math.BigInteger;
import java.util.*;
class rsa
{
    public static void main(String args[])
    {
        Scanner ip=new Scanner(System.in);
        int p,q,n,e=1,j;
        int d=1,i1;
        int t1,t2;
        int pt[]= new int[10];
        int ct[]= new int[10];
        int rt[]= new int[10];
        int temp[]= new int[10];
        String i=new String();

        System.out.println("Enter the two prime numbers:");
        p=ip.nextInt();
        q=ip.nextInt();

        System.out.println("Enter the message to be sent");
        i=ip.next();
        i1=i.length();
        n=p*q;
        t1=p-1;
        t2=q-1;

        System.out.println("\n -----");
    }
}

```

```

System.out.println("Sender Side:");
while((t1*t2)%e==0)
{
    e++;
}
System.out.println("Public Key(e)= "+e);
System.out.println("-----");
for(j=0;j<i1;j++)
{
    pt[j]=(i.charAt(j))-96;
//    System.out.println("Plain Text= "+pt[j]);
    ct[j]=((int)Math.pow(pt[j],e))%n;
    System.out.println("Cipher Text= "+ct[j]);
}

System.out.println("\nTransmitted Message:");
for(j=0;j<i1;j++)
{
    temp[j]=ct[j]+96;
    System.out.print((char)temp[j]);
}

System.out.println("\n\n -----");
System.out.println("Receiver Side:");
while((d*e)%(t1*t2)!=1)
{
    d++;
}

System.out.println("Private Key(d)= "+d);
System.out.println("-----");

for(j=0;j<i1;j++)
{

```

```
//System.out.println("cipher Text= "+ct[j]);
    BigInteger very_big_no = BigInteger.valueOf(ct[j]);
    very_big_no = very_big_no.pow(d);
    very_big_no = very_big_no.mod(BigInteger.valueOf(n));
    rt[j] = very_big_no.intValue();
    System.out.println("Plain Text= "+rt[j]);
}

System.out.println("\n-----");
System.out.println("Decrypted Message:");
for(j=0;j<i1;j++)
{
    rt[j]=rt[j]+96;
    System.out.print((char)rt[j]);
}

System.out.println("\n -----");
ip.close();
}
}
```

Output:

```
java rsa
Enter the two prime numbers:
5
11
Enter the message to be sent
global
```

```
-----
Sender Side:
Public Key(e)= 3
-----
Cipher Text= 13
Cipher Text= 23
Cipher Text= 20
Cipher Text= 8
Cipher Text= 1
Cipher Text= 23
```

Transmitted Message:
mwthaw

Receiver Side:
Private Key(d)= 27

Plain Text= 7
Plain Text= 12
Plain Text= 15
Plain Text= 2
Plain Text= 1
Plain Text= 12

Decrypted Message:
global

Explanation

- RSA (Rivest–Shamir–Adleman) is an algorithm used by modern computers to encrypt and decrypt messages.
- It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys.
- This is also called public key cryptography, because one of the keys can be given to anyone.
- The other key must be kept private.

What is Assymmetric Key Encryption?

- Asymmetric Encryption is a form of Encryption where keys come in pairs. What one key encrypts, only the other can decrypt.
- Frequently (but not necessarily), the keys are interchangeable, in the sense that if key A encrypts a message, then B can decrypt it, and if key B encrypts a message, then key A can decrypt it.
- While common, this property is not essential to asymmetric encryption.
- Asymmetric Encryption is also known as Public Key Cryptography, since users typically create a matching key pair, and make one public while keeping the other secret.

- Users can "sign" messages by encrypting them with their private keys.
- This is effective since any message recipient can verify that the user's public key can decrypt the message, and thus prove that the user's secret key was used to encrypt it.
- If the user's secret key is, in fact, secret, then it follows that the user, and not some impostor, really sent the message.
- Users can send secret messages by encrypting a message with the recipient's public key.
- In this case, only the intended recipient can decrypt the message, since only that user should have access to the required secret key.
- The key to successful use of Asymmetric Encryption is a Key Management system, which implements a Public Key Infrastructure.
- Without this, it is difficult to establish the reliability of public keys, or even to conveniently find suitable ones.

RSA Algorithm

1. Consider two large prime numbers p, q
2. Calculate $n=p*q$
3. Calculate euler's totient function $\phi(n) = (p-1)*(q-1)$
4. Assume e such that $\gcd(e, \phi(n))=1$ // e should be co-prime to $\phi(n)$
5. Assume d such that $d \cong e^{-1} \text{ mod } \phi(n)$ // Remove Congruence next slide
// After removing congruence $d*e \text{ mod } \phi(n) = 1$
6. Public Key = $\{e, n\}$ and Private Key = $\{d, n\}$

*****Removing Congruence*****

$$d*e \cong 1 \text{ mod } \phi(n)$$

Apply mod to both the sides

$$d*e \text{ mod } \phi(n) = 1 \text{ mod } \phi(n) \text{ //where } 1 \text{ mod } \phi(n) = 1$$

$$d*e \text{ mod } \phi(n) = 1$$

RSA Algorithm Explanation

1. Encryption Side

- $M < n$ (plain text must be smaller than n)
- Cipher text conversion $C = M^e \bmod n$
- Cipher text = Encryption of plain text mod n
- Public Key is used for Encryption

2. Decryption Side

- $M = C^d \bmod n$
- Plain text = Decryption of cipher text mod n
- Private Key is used for Decryption

Euler's totient function

- **Consider Euler's totient function for example, $\phi(10)$**
- Write numbers within the range 1,2,3,4,5,6,7,8,9,10
- Factors of 10 = 2,5
- Cancel those variables which are multiples of 2 and 5 \rightarrow 2,4,5,6,8,10
- Remaining are 1,3,7,9
- Hence $\phi(10) = 4$
-
- **Another example $\phi(5)$ where the Factors of 5 = 5**
- Hence values are 1,2,3,4
- Therefore $\phi(5)=4$

RSA Algorithm Example

- For Simple calculation consider 2 smaller prime numbers
- Where $p=3$, $q=5$
- $n = p * q = 3 * 5 = 15$
- $\phi(n) = (p-1) * (q-1) = (3-1) * (5-1) = 2 * 4 = 8$

- In Encryption Side, $\gcd(e, \phi(n))=1$
- Let's assume e as 3
- Why 3?
- Here e and $\phi(n)$ are co-prime hence $\phi(n)=8$ where factors not divisible 3,5 or 7 are taken
- Hence \gcd
- $\gcd(e, \phi(n)) = \gcd(3,8)$
- Which is equal to 1
- In Decryption Side, $d \cdot e \bmod \phi(n) = 1$
- $d \cdot 3 \bmod 8 = 1$
- Assume $d = 3$
- $d \cdot e \bmod \phi(n) = 3 \cdot 3 \bmod 8$
- $9 \bmod 8 = 1$

Public Key

- $\{e,n\} = \{3,15\}$

Private Key

- $\{d,n\} = \{3,15\}$

At the Encryption Side,

- Consider plain text message as $M=4$, which is less than n
- Cipher text $C=M^e \bmod n$
- $C = 4^3 \bmod 15$
- $C = 64 \bmod 15 = 4$
- Therefore $C=4$

At the Decryption Side,

- $M = C^d \bmod n$

- $M = 4^3 \bmod 15$
- $M = 64 \bmod 15 = 4$
- Therefore $M=4$

Program 6:**Write a program for Congestion control using leaky bucket algorithm.**

```

Import java.util.Scanner;

```

```

public class LeakyBucket {

    Public static void main(String[] args) throws InterruptedException {

        Int n, incoming, outgoing, store=0, bucketsize;
        Scanner scan = new Scanner(system.in);
        System.out.println("Enter bucket size, outgoing rate, number of inputs and
incoming size);
        bucketsize = scan.nextInt();
        outgoing = scan.nextInt();
        n = scan.nextInt();
        Incoming = scan.nextInt();
        while(n!=0)
        {
            System.out.println("Incoming size is " + incoming);
            if(incoming <= (bucketsize-store))
            {
                store+= incoming;
                System.out.println("Bucket buffer size is " + store + " out of " +
bucketsize);
            }
            else
            {
                System.out.println("Packet loss : " + (incoming-(bucketsize-size)));
                store=bucketsize;
                System.out.println("Bucket buffer size is " + store + " out of " +
bucketsize);
            }
            store-=outgoing;
            System.out.println("After outgoing: " + store + " packets left out of "
+ bucketsize + " in buffer);
            n--;
            Thread.sleep(3000);
        }
        scan.close();
    }
}

```

Output:

```
krishna@ubuntu:~$ javac Leaky.java
krishna@ubuntu:~$ java Leaky

Enter the packets to be sent:
12

Enter 0 element: 2
Enter 1 element: 3
Enter 2 element: 5
Enter 3 element: 6
Enter 4 element: 8
Enter 5 element: 9
Enter 6 element: 4
Enter 7 element: 5
Enter 8 element: 6
Enter 9 element: 2
Enter 10 element: 7
Enter 11 element: 3

Queue is full
Lost Packet: 3

Leaked Packet: 2
Leaked Packet: 3
Leaked Packet: 5
Leaked Packet: 6
Leaked Packet: 8
Leaked Packet: 9
Leaked Packet: 4
Leaked Packet: 5
Leaked Packet: 6
Leaked Packet: 2
Leaked Packet: 7
```

Explanation

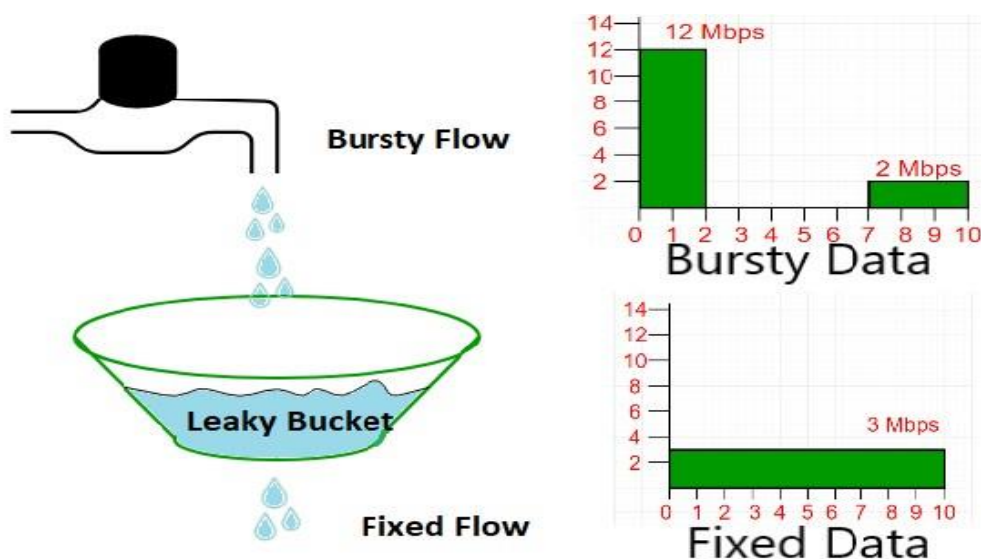
- In the network layer, before the network can make Quality of service guarantees, it must know what traffic is being guaranteed.
- One of the main causes of congestion is that traffic is often bursty.
- To understand this concept first we have to know little about traffic shaping.
- Traffic Shaping is a mechanism to control the amount and the rate of the traffic sent to the network.
- Approach of congestion management is called Traffic shaping.
- Traffic shaping helps to regulate rate of data transmission and reduces congestion.

There are 2 types of traffic shaping algorithms:

1. Leaky Bucket
2. Token Bucket

Leaky Bucket Concept

- Suppose we have a bucket in which we are pouring water in a random order but we have to get water in a fixed rate, for this we will make a hole at the bottom of the bucket.
- It will ensure that water coming out is in a some fixed rate, and also if bucket will full we will stop pouring in it.
- The input rate can vary, but the output rate remains constant.
- Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic.
- Bursty chunks are stored in the bucket and sent out at an average rate.



- In the figure, we assume that the network has committed a bandwidth of 3 Mbps for a host.

- The use of the leaky bucket shapes the input traffic to make it conform to this commitment.
- In Figure the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data.
- The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data.
- In all, the host has sent 30 Mbits of data in 10 s.
- The leaky bucket smooths the traffic by sending out data at a rate of 3 Mbps during the same 10 s.
- Without the leaky bucket, the beginning burst may have hurt the network by consuming more bandwidth than is set aside for this host.
- We can also see that the leaky bucket may prevent congestion.
- A simple leaky bucket algorithm can be implemented using FIFO queue.
- A FIFO queue holds the packets. If the traffic consists of fixed-size packets (e.g., cells in ATM networks), the process removes a fixed number of packets from the queue at each tick of the clock.
- If the traffic consists of variable-length packets, the fixed output rate must be based on the number of bytes or bits.

Leaky Bucket Algorithm

- The following is an algorithm for variable-length packets:
 1. Initialize a counter to n at the tick of the clock.
 2. If n is greater than the size of the packet, send the packet and decrement the counter by the packet size. Repeat this step until n is smaller than the packet size.
 3. Reset the counter and go to step 1.

Let's choose an example

Example – Let $n=1000$

200	700	500	450	400	200
-----	-----	-----	-----	-----	-----

Packet=

Since $n >$ front of Queue i.e. $n > 200$

Therefore, $n = 1000 - 200 = 800$

Packet size of 200 is sent to the network.

200	700	500	450	400
-----	-----	-----	-----	-----

Now Again $n >$ front of the queue i.e. $n > 400$

Therefore, $n = 800 - 400 = 400$

Packet size of 400 is sent to the network.

200	700	500	450
-----	-----	-----	-----

Since $n <$ front of queue

Therefore, the procedure is stop.

- Initialize $n=1000$ on another tick of clock.
- This procedure is repeated until all the packets are sent to the network.

VIVA QUESTIONS

1. Explain What is Network?

A network is a set of devices connected by physical media links. A network is recursively is a connection of two or more nodes by a physical link or two or more networks connected by one or more nodes.

2. What is a Link?

At the lowest level, a network can consist of two or more computers directly connected by some physical medium such as coaxial cable or optical fiber. Such a physical medium is called as Link.

3. What is a node?

A network can consist of two or more computers directly connected by some physical medium such as coaxial cable or optical fiber. Such a physical medium is called as Links and the computer it connects is called as Nodes.

4. What is a gateway or Router?

A node that is connected to two or more networks is commonly called as router or Gateway. It generally forwards message from one network to another.

5. What is point-point link?

If the physical links are limited to a pair of nodes it is said to be point-point link.

6. What is Multiple Access?

If the physical links are shared by more than two nodes, it is said to be Multiple Access.

7. What are the advantages of Distributed Processing?

- a. Security/Encapsulation
- b. Distributed database
- c. Faster Problem solving
- d. Security through redundancy
- e. Collaborative Processing

8. What are the criteria necessary for an effective and efficient network?

- a. Performance

It can be measured in many ways, including transmit time and response time.

b. Reliability
It is measured by frequency of failure, the time it takes a link to recover from a failure, and the networks robustness.

- c. Security

Security issues includes protecting data from unauthorized access and viruses.

9. Name the factors that affect the performance of the network?

- a. Number of Users
- b. Type of transmission medium
- c. Hardware
- d. Software

10. Name the factors that affect the reliability of the network?

- a. Frequency of failure
- b. Recovery time of a network after a failure

11. Name the factors that affect the security of the network?

- a. Unauthorized Access
- b. Viruses

12. What is Protocol?

A protocol is a set of rules that govern all aspects of information communication.

13. What are the key elements of protocols?

The key elements of protocols are

- a. Syntax

It refers to the structure or format of the data that is the order in which they are presented.

- b. Semantics

It refers to the meaning of each section of bits.

- c. Timing

Timing refers to two characteristics: When data should be sent and how fast they can be sent.

14. What are the key design issues of a computer Network?

- a. Connectivity
- b. Cost-effective Resource Sharing
- c. Support for common Services
- d. Performance

15. Define Bandwidth and Latency?

Network performance is measured in Bandwidth (throughput) and Latency (Delay). Bandwidth of a network is given by the number of bits that can be transmitted over the network in a certain period of time. Latency corresponds to how long it takes a message to travel from one end of a network to the other. It is strictly measured in terms of time.

16. Define Routing?

The process of determining systematically how to forward messages toward the destination nodes based on its address is called routing.

17. What is a peer-peer process?

The processes on each machine that communicate at a given layer are called peer-peer process.

18. When a switch is said to be congested?

It is possible that a switch receives packets faster than the shared link can accommodate and stores in its memory, for an extended period of time, then the switch will eventually run out of buffer space, and some packets will have to be dropped and in this state is said to be congested state.

19. What is semantic gap?

Defining a useful channel involves both understanding the applications requirements and recognizing the limitations of the underlying technology. The gap between what applications expects and what the underlying technology can provide is called semantic gap.

20. What is Round Trip Time?

The duration of time it takes to send a message from one end of a network to the other and back, is called RTT.

21. Define the terms Unicast, Multicast and Broadcast?

If the message is sent from a source to a single destination node, it is called Unicast.

If the message is sent to some subset of other nodes, it is called Multicast.

If the message is sent to all the nodes in the network it is called Broadcast.

22. What is Multiplexing?

Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.

23. Name the categories of Multiplexing?

- a. Frequency Division Multiplexing (FDM)
- b. Time Division Multiplexing (TDM)
 - i. Synchronous TDM
 - ii. Asynchronous TDM Or Statistical TDM.
- c. Wave Division Multiplexing (WDM)

24. What is FDM?

FDM is an analog technique that can be applied when the bandwidth of a link is greater than the combined bandwidths of the signals to be transmitted.

25. What is WDM?

WDM is conceptually the same as FDM, except that the multiplexing and demultiplexing involve light signals transmitted through fiber optics channel.

26. What is TDM?

TDM is a digital process that can be applied when the data rate capacity of the transmission medium is greater than the data rate required by the sending and receiving devices.

27. What is Synchronous TDM?

In STDM, the multiplexer allocates exactly the same time slot to each device at all times, whether or not a device has anything to transmit.

28. List the layers of OSI

- a. Physical Layer
- b. Data Link Layer
- c. Network Layer
- d. Transport Layer
- e. Session Layer
- f. Presentation Layer
- g. Application Layer

29. Which layers are network support layers?

- a. Physical Layer
- b. Data link Layer and
- c. Network Layers

30. Which layers are user support layers?

- a. Session Layer
- b. Presentation Layer and
- c. Application Layer

31. Which layer links the network support layers and user support layers?

The Transport layer links the network support layers and user support layers.

32. What are the concerns of the Physical Layer?

Physical layer coordinates the functions required to transmit a bit stream over a physical medium.

- a. Physical characteristics of interfaces and media
- b. Representation of bits
- c. Data rate
- d. Synchronization of bits
- e. Line configuration
- f. Physical topology
- g. Transmission mode

33. What are the responsibilities of Data Link Layer?

The Data Link Layer transforms the physical layer, a raw transmission facility, to a reliable link and is responsible for node-node delivery.

- a. Framing
- b. Physical Addressing
- c. Flow Control
- d. Error Control
- e. Access Control

34. What are the responsibilities of Network Layer?

The Network Layer is responsible for the source-to-destination delivery of packet possibly across multiple networks (links).

- a. Logical Addressing
- b. Routing

35. What are the responsibilities of Transport Layer?

The Transport Layer is responsible for source-to-destination delivery of the entire message.

- a. Service-point Addressing
- b. Segmentation and reassembly
- c. Connection Control
- d. Flow Control
- e. Error Control

36. What are the responsibilities of Session Layer?

The Session layer is the network dialog Controller. It establishes, maintains and synchronizes the interaction between the communicating systems.

- a. Dialog control
- b. Synchronization

37. What are the responsibilities of Presentation Layer?

The Presentation layer is concerned with the syntax and semantics of the information exchanged between two systems.

- a. Translation
- b. Encryption
- c. Compression

38. What are the responsibilities of Application Layer?

The Application Layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as e-mail, shared database management and other types of distributed information services.

- a. Network virtual Terminal
- b. File transfer, access and Management (FTAM)
- c. Mail services
- d. Directory Services

39. What are the two classes of hardware building blocks?

Nodes and Links.

40. What are the different link types used to build a computer network?

- a. Cables
- b. Leased Lines
- c. Last-Mile Links
- d. Wireless Links

41. What are the categories of Transmission media?**a. Guided Media****i. Twisted Pair cable****1. Shielded TP****2. Unshielded TP****ii. Coaxial Cable****iii. Fiber-optic cable****b. Unguided Media****i. Terrestrial microwave****ii. Satellite Communication****42. What are the types of errors?****a. Single-Bit error**

In a single-bit error, only one bit in the data unit has changed

b. Burst Error

A Burst error means that two or more bits in the data have changed.

43. What is Error Detection? What are its methods?

Data can be corrupted during transmission. For reliable communication errors must be deducted and Corrected. Error Detection uses the concept of redundancy, which means adding extra bits for detecting errors at the destination. The common Error Detection methods are

a. Vertical Redundancy Check (VRC)**b. Longitudinal Redundancy Check (VRC)****c. Cyclic Redundancy Check (VRC)****d. Checksum**

44. What is Redundancy?

The concept of including extra information in the transmission solely for the purpose of comparison. This technique is called redundancy.

45. What is VRC?

It is the most common and least expensive mechanism for Error Detection. In VRC, a parity bit is added to every data unit so that the total number of 1s becomes even for even parity. It can detect all single-bit errors. It can detect burst errors only if the total number of errors in each data unit is odd.

46. What is LRC?

In LRC, a block of bits is divided into rows and a redundant row of bits is added to the whole block. It can detect burst errors. If two bits in one data unit are damaged and bits in exactly the same positions in another data unit are also damaged, the LRC checker will not detect an error. In LRC a redundant data unit follows n data units.

47. What is CRC?

CRC, is the most powerful of the redundancy checking techniques, is based on binary division.

48. What is Checksum?

Checksum is used by the higher layer protocols (TCP/IP) for error detection

49. List the steps involved in creating the checksum.

- a. Divide the data into sections
- b. Add the sections together using 1s complement arithmetic
- c. Take the complement of the final sum, this is the checksum.

50. What are the Data link protocols?

Data link protocols are sets of specifications used to implement the data link layer. The categories of Data Link protocols are

Asynchronous Protocols

Synchronous Protocols

a. Character Oriented Protocols

b. Bit Oriented protocols

51. Compare Error Detection and Error Correction:

The correction of errors is more difficult than the detection. In error detection, checks only any error has occurred. In error correction, the exact number of bits that are corrupted and location in the message are known. The number of the errors and the size of the message are important factors.

52. What is Forward Error Correction?

Forward error correction is the process in which the receiver tries to guess the message by using redundant bits.

53. Define Retransmission?

Re transmission is a technique in which the receiver detects the occurrence of an error and asks the sender to resend the message. Re sending is repeated until a message arrives that the receiver believes is error-free.

54. What are Data Words?

In block coding, we divide our message into blocks, each of k bits, called datawords. The block coding process is one-to-one. The same dataword is always encoded as the same codeword.

55. What are Code Words?

r redundant bits are added to each block to make the length $n = k + r$. The resulting n -bit blocks are called codewords. $2^n - 2^k$ codewords that are not used. These codewords are invalid or illegal.

56. What is a Linear Block Code?

A linear block code is a code in which the exclusive OR (addition modulo-2) of two valid codewords creates another valid codeword.

57. What are Cyclic Codes?

Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword.

58. Define Encoder?

A device or program that uses predefined algorithms to encode, or compress audio or video data for storage or transmission use. A circuit that is used to convert between digital video and analog video.

59. Define Decoder?

A device or program that translates encoded data into its original format (e.g. it decodes the data). The term is often used in reference to MPEG-2 video and sound data, which must be decoded before it is output.

60. What is Framing?

Framing in the data link layer separates a message from one source to a destination, or from other messages to other destinations, by adding a sender address and a destination address. The destination

address defines where the packet has to go and the sender address helps the recipient acknowledge the receipt.

61. What is Fixed Size Framing?

In fixed-size framing, there is no need for defining the boundaries of the frames. The size itself can be used as a delimiter.

62. Define Character Stuffing?

In byte stuffing (or character stuffing), a special byte is added to the data section of the frame when there is a character with the same pattern as the flag. The data section is stuffed with an extra byte. This byte is usually called the escape character (ESC), which has a predefined bit pattern. Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not a delimiting flag.

63. What is Bit Stuffing?

Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.

64. What is Flow Control?

Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.

65. What is Error Control?

Error control is both error detection and error correction. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender. In the data link layer, the term error control refers primarily to methods of error detection and retransmission.

66. What Automatic Repeat Request (ARQ)?

Error control is both error detection and error correction. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender. In the data link layer, the term error control refers primarily to methods of error detection and retransmission. Error control in the data link layer is often implemented simply: Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request (ARQ).

67. What is Stop-and-Wait Protocol?

In Stop and wait protocol, sender sends one frame, waits until it receives confirmation from the receiver (okay to go ahead), and then sends the next frame.

68. What is Stop-and-Wait Automatic Repeat Request?

Error correction in Stop-and-Wait ARQ is done by keeping a copy of the sent frame and retransmitting of the frame when the timer expires.

69. What is usage of Sequence Number in Reliable Transmission?

The protocol specifies that frames need to be numbered. This is done by using sequence numbers. A field is added to the data frame to hold the sequence number of that frame. Since we want to minimize the frame size, the smallest range that provides unambiguous communication. The sequence numbers can wrap around.

70. What is Pipelining ?

In networking and in other areas, a task is often begun before the previous task has ended. This is known as pipelining.

71. What is Sliding Window?

The sliding window is an abstract concept that defines the range of sequence numbers that is the concern of the sender and receiver. In other words, the sender and receiver need to deal with only part of the possible sequence numbers.

72. What is Piggy Backing?

A technique called piggybacking is used to improve the efficiency of the bidirectional protocols. When a frame is carrying data from A to B, it can also carry control information about arrived (or lost) frames from B; when a frame is carrying data from B to A, it can also carry control information about the arrived (or lost) frames from A.

73. What are the two types of transmission technology available?

(i) Broadcast and (ii) point-to-point

74. What is subnet?

A generic term for section of a large networks usually separated by a bridge or router.

75. Difference between the communication and transmission.

Transmission is a physical movement of information and concern issues like bit polarity, synchronisation, clock etc.

Communication means the meaning full exchange of information between two communication media.

76. What are the possible ways of data exchange?

(i) Simplex (ii) Half-duplex (iii) Full-duplex.

77. What is SAP?

Series of interface points that allow other computers to communicate with the other layers of network protocol stack.

78. What is meant by Router ?

A router is an electronic device that interconnects two or more computer networks, and selectively interchanges packets of data between them. A router is a networking device whose software and hardware are customized to the tasks of routing and forwarding information.

79. What is meant by Hubs ?

A common connection point for devices in a network. Hubs are commonly used to connect segments of a LAN. A hub contains multiple ports. When a packet arrives at one port, it is copied to the other ports so that all segments of the LAN can see all packets.

80. What is meant by Bridges ?

A network bridge connects multiple network segments at the data link layer (layer 2) of the OSI model. Bridges come in three basic types:

Local bridges: Directly connect local area networks (LANs)

Remote bridges: Can be used to create a wide area network (WAN) link between LANs. Remote bridges, where the connecting link is slower than the end networks, largely have been replaced with routers.

Wireless bridges: Can be used to join LANs or connect remote stations to LANs.

81. What do u mean by NIC (Network Interface Card) ?

A network card, network adapter, or NIC (network interface card) is a piece of computer hardware designed to allow computers to communicate over a computer network. It provides physical access to a networking medium and often provides a low-level addressing system through the use of MAC addresses.

82. What do u mean by Repeater ?

A repeater is an electronic device that receives a signal, cleans it of unnecessary noise, regenerates it, and retransmits it at a higher power level, or to the other side of an obstruction, so that the signal can cover longer distances without degradation.

83. Definitions of Firewall ?

Firewalls are the most important aspect of the network and its security in today's era. Due to maximization of attacks on the networks from various groups stealing data, denying services etc the firewall is playing a vital role in computer networks.

84. What is the Difference between HUB and SWITCH ?

HUB-it is a network device that provides a central connection for cables from work station, server etc. The hub takes incoming single one port and provides every port so that is the main reason for collision. **Switch**-it also provides central connection to the work station. It provides the unique cast. It is better than a hub. **HUB** (1) works on physical layer in OSI model (2) Hub is half duplex (3) collision detection is on in hub (4) Hub broadcasts transmitted message. **Switch** (1) switch works in data link layer in OSI model (2) switch works half and full duplex (3) collision detection is off (4) switch transmits message unicast and sometimes broadcast.

85. Difference between Physical Address and Logical Address ?

A Physical address is a 48-bit flat address burned into the ROM of the NIC card which is a Layer 1 device of the OSI model. This is divided into 24-bit vendor code and 24-bit serial address. This is unique for each system and cannot be changed. A Logical address is a 32-bit address assigned to each system in a network. This works in Layer-3 of OSI Model. This would be generally the IP address.

86. What is MAC Address ?

A unique 48-bit address assigned to each network card. It is also called as physical address.

87. What is PING Utility ?

PING stands Packet Internet Gopher. This is a utility for ensuring connectivity between computers. ICMP protocol works behind this utility. Under it, sending node sends packets to destination node and reply is received if there is proper communication between two. Ping command is used to check the Destination host or router connectivity to use the icmp and echo packet.

88. What do u mean by Gateway ?

They relay packets among networks that have different protocols (e.g. between a LAN and a WAN). They accept a packet formatted for one protocol and convert it to a packet formatted for another protocol before forwarding it. They operate in all seven layers of the OSI model.