# Nitte Meenakshi Institute of Technology

# Bangalore

# Computer Communication Networks Laboratory Manual

**Course: B.E(ECE)**

**Sub Code: ECL77**

## Course Outcomes

1. Student will be able to apply knowledge of communication standards and interfaces to model and analyze the flow of data on a communication network
2. Student will be able to apply the knowledge of protocols to design and implement a local area network of computers
3. Student will learn to use tools and software implementation for simulation and study of computer networks

**Course Outcomes mapping to Program Outcomes**

| PO | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **CO1** | S |  |  |  | M |  |  |  |  |  |  |  |
| **CO2** |  | S |  | M |  |  |  |  |  |  |  |  |
| **CO3** |  |  |  | M | S |  |  |  |  |  |  |  |

# CONTENTS

# Introduction to NS2

**What is Simulation?**

"The process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behavior of the system and/or evaluating various strategies for the operation of the system."

Network Simulator (NS2) is an open-source event-driven simulator designed specifically for research in computer communication networks. A computer network is usually defined as a collection of computers interconnected for gathering, processing, and distributing information. The Internet is a good example of computer networks. In fact, it is a network of networks, within which, tens of thousands of networks interconnect millions of computers worldwide.

**Advantages of NS2**

- Cost effective
- Flexible
- Easier to analyze
- Provide substantial support to simulate bunch of protocols like TCP, UDP, FTP, and HTTP

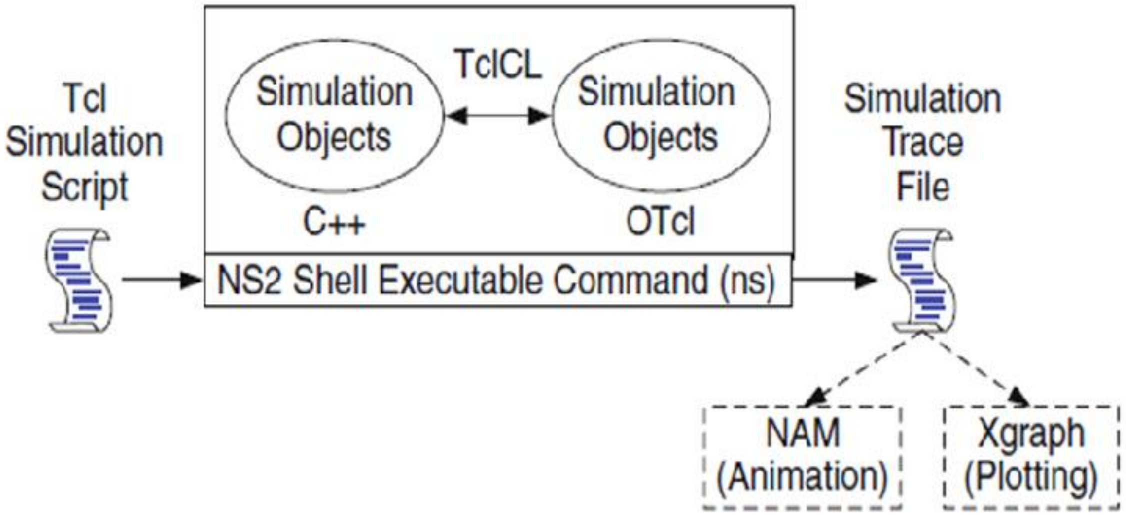
**Programming Language:** C++

**Scripting Language:** Tool Kit Command Language (TCL)

### Architecture of NS-2

The network simulator NS is a discrete event network simulator developed at UC Berkeley that focuses on the simulation of IP networks on the packet level

The NS project (the project that drives the development of NS) is now part of the Virtual InterNetwork Testbed (VINT) project, that develops tools for network simulation research . Researchers have used NS to develop and investigate protocols such as TCP and UDP, router queuing policies (RED, ECN, CBQ), Multicast transport, Multimedia and more .



### SIMPLIFIED USER VIEW OF NS

NS is basically an Object-oriented Tcl (Otcl) script interpreter with network simulation object libraries. NS has a simulation event scheduler, network component object libraries and network setup (plumbing) modul libraries (see figure 7).

To use NS for setting up and running a network simulation, a user writes a simulation program in Otcl script language.

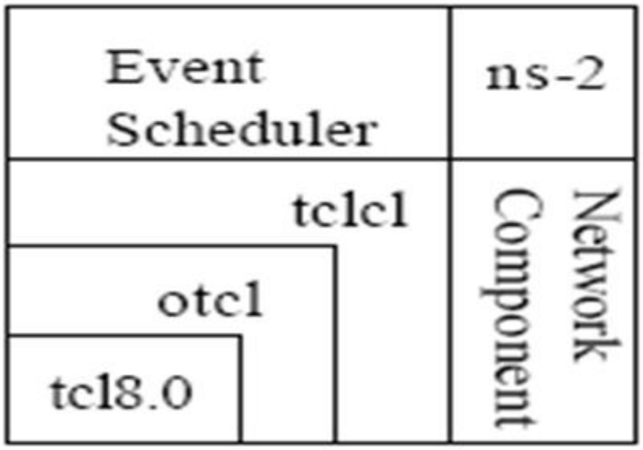
Such an OTcl script initiates an event scheduler, sets up the network topology and tells traffic sources when to start and stop transmitting packets through the event scheduler.

### Architectural Overview

The NS-2 architecture is composed of five parts:

- Event scheduler
- Network components
- Tclcl

- ● OTcl library
- ● Tcl 8.0 scipt language



The above figure shows a graphical overview of the NS-2 architecture. A user can be thought of standing at the left bottom corner, designing and running simulations in Tcl using the simulator objects in the OTcl library.” The event schedulers and most of the network components are implemented in C++ because of efficiency reasons. These are available to OTcl through an OTcl linkage that is implemented using tclcl. These five components together make up NS, which is an object-oriented extended Tcl interpreter with network simulator libraries.

NS models all network elements through a class hierarchy. For example Agent is a class TCP and UDP under it. To drive the execution of the simulation, to process and schedule simulation events, NS makes use of the concept of discrete event schedulers . In NS, network components that simulate packet-handling delay or that need timers use event schedulers.

## Experiment No 1

**Problem statement**

**Simulate a 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**

**TOPOLOGY:-**

**Program**

```
#Create Simulator
set ns [new Simulator]

#Open Trace file and NAM file
set ntrace [open prog1.tr w]
$ns trace-all $ntrace
set namfile [open prog1.nam w]
$ns namtrace-all $namfile

#Finish Procedure
```

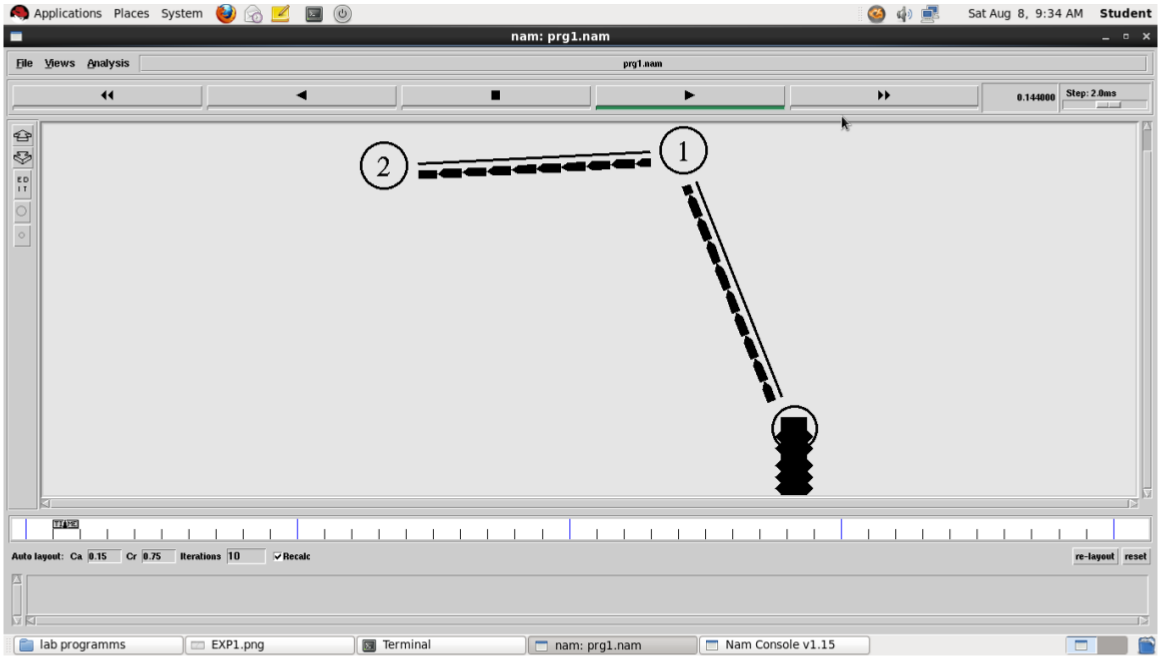
```
proc Finish {} {
      global ns ntrace namfile

      #Dump all the trace data and close the files
      $ns flush-trace
      close $ntrace
      close $namfile

      #/Execute the nam animation file
exec nam prog1.nam
exec echo “The number of packets drop is”&
exec grep –c “^d” prog1.tr &
exit 0
}
#Create 3 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
```

```
#Create Links between nodes
#You need to modify the bandwidth to observe
the variation in packet drop
$ns duplex-link $n0 $n1 0.75Mb 10ms DropTail
$ns duplex-link $n1 $n2 0.75Mb 10ms DropTail

#Set Queue Size
#You can modify the queue length as well to
observe the variation in packet drop
$ns queue-limit $n0 $n1 10
$ns queue-limit $n1 $n2 10

#Set up a Transport layer connection.
set udp [new Agent/UDP]
$ns attach-agent $n0 $udp
set null [new Agent/Null]
$ns attach-agent $n2 $null
$ns connect $udp $null
#Set up an Application layer Traffic
```

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set type_ CBR
$cbr0 set packetSize_ 100
$cbr0 set rate_ 1Mb
$cbr0 set random_ false
$cbr0 attach-agent $udp

#Schedule Events
$ns at 0.0 "$cbr0 start"
$ns at 5.0 "Finish"

#Run the Simulation
$ns run
```

**Output**

```
no. of packets drop is 1554
```



0.144000 Step: 2.0ms

Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑ Recalc

## Experiment No 2

**Problem statement**

**Simulate a four node point-to-point network, and connect the links as follows: n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3 and UDP n1-n3. Apply relevant applications over TCP and UDP agents changing the parameter and determine the number of packets sent by TCP/UDP**

**Topology:-**

**Program**

```
set  ns [ new Simulator]
set ntrace [open prog2.tr w]
$ns trace-all $ntrace
set namfile [open prog2.nam w]
$ns namtrace-all $namfile
```

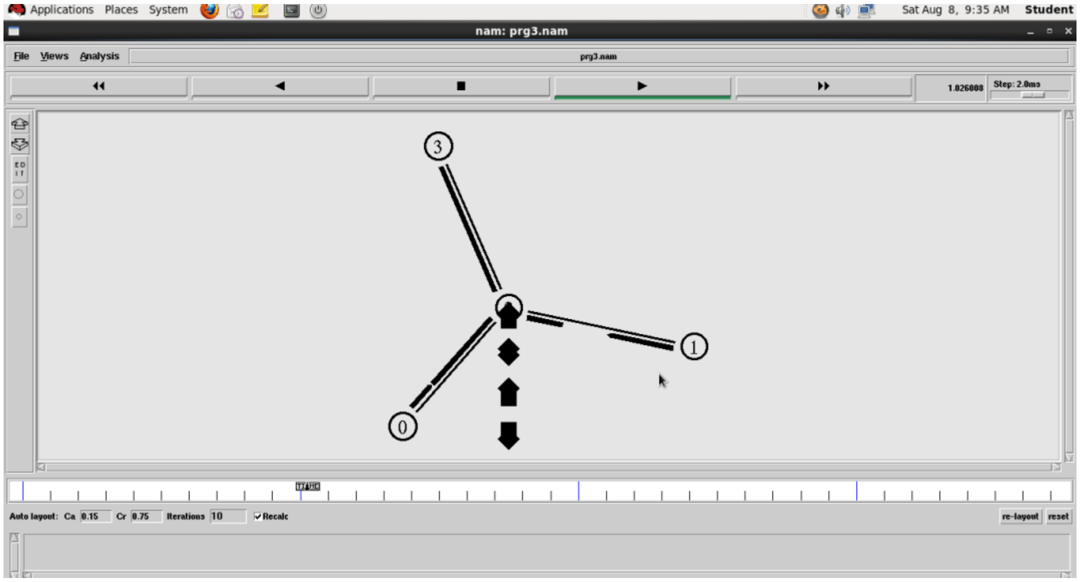
```
proc Finish {} {
        global ns ntrace namfile

        $ns flush-trace
        close $ntrace
        close $namfile

        exec nam prog2.nam &

        exec echo "The number of TCP packets sent are" &
        exec grep "^+" prog2.tr   | cut -d " " -f 5 | grep -c
"tcp" &
        exec echo "The number of UDP packets sent are" &
        exec grep "^+" prog2.tr | cut -d " " -f 5 | grep -c "cbr"
&
        exit 0
}

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

```
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 0.07Mb 20ms DropTail

set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0
$ns connect $tcp0 $sink0

set udp0 [new Agent/UDP]
$ns attach-agent $n1 $udp0
set null0 [new Agent/Null]
$ns attach-agent $n3 $null0
$ns connect $udp0 $null0

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


set cbr0 [new Application/Traffic/CBR]
$cbr0 set type_    CBR
```

```
$cbr0 set packetSize_  1000
$cbr0 set rate_ 0.01Mb
$cbr0 set random_ false
$cbr0 attach-agent $udp0

$ns at 0.1 "$cbr0 start"
$ns at 1.5 "$ftp0 start"
$ns at 1.0 "$cbr0 stop"
$ns at 2.5 "$ftp0 stop"
$ns at 5.0 "Finish"

$ns run
```

**output**

```
no. of tcp packets sent are 384
no. of udp packets sent are 60
```



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑ Recalc

**Experiment No 3**

**Problem statement**

# Simulate an Ethernet LAN using N nodes . Determine collision across different nodes

**TOPOLOGY:-**

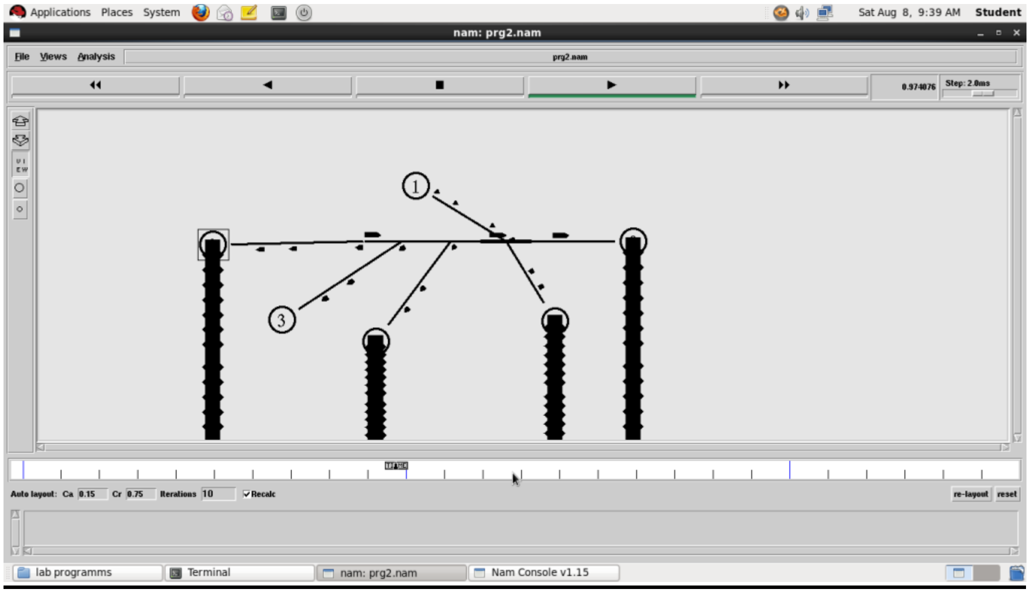
**Program**

```
set ns [new Simulator]

set trf [open 6.tr w]
$ns trace-all $trf
set naf [open 6.nam w]
$ns namtrace-all $naf

#to create nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

set lan [$ns newLan "$n0 $n1 $n2 $n3
$n4 $n5" 5Mb 10ms LL Queue/DropTail
Channel]

set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink

set udp [new Agent/UDP]
```

```
$ns attach-agent $n1 $udp
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null

proc finish {} {
global ns naf trf
$ns flush-trace
exec nam 6.nam &
close $trf
close $naf
exec echo "The number of packet drops
due to collision is" &
exec grep -c "^d" 6.tr  &
exit 0
}
$ns at 0.1 "$cbr start"
$ns at 2.0 "$ftp start"
$ns at 1.9 "$cbr stop"
$ns at 4.3 "$ftp stop"
$ns at 6.0 "finish"
$ns run
```

## OUTPUT

**no. of packets drop 6012**



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑Recalc

**Experiment No 4**

**Problem Statement**

# Simulate an Ethernet LAN using N-nodes(1-7), and determine the throughput

**Topology:-**

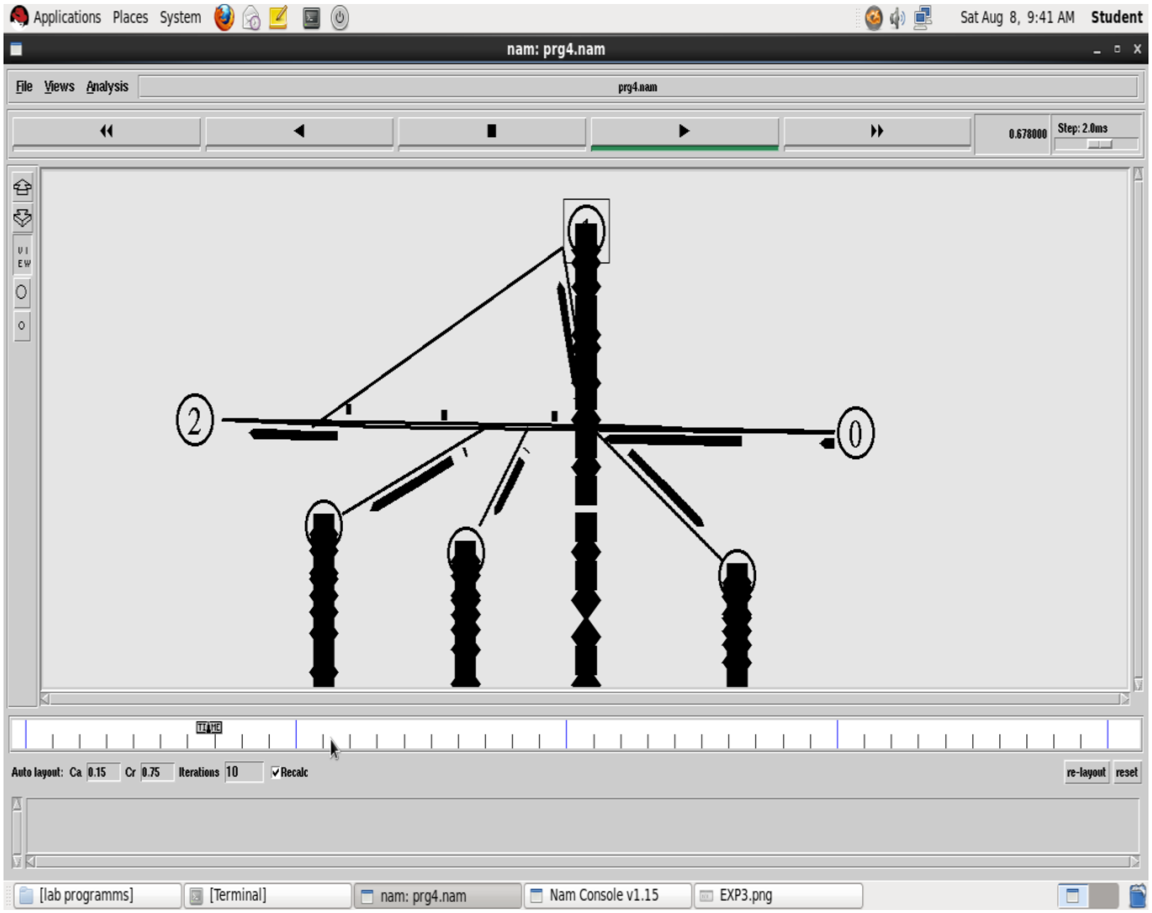
**Program**

```
set ns [new Simulator]
set trf [open prog5.tr w]
$ns trace-all $trf
Set naf [open prog5.nam w]
$ns namtrace-all $naf
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set lan [$ns newLan "$n0 $n1 $n2 $n3 $n4 $n5 $n6 $n7" 5Mb 10ms LL Queue/DropTail Channel]

set tcp [new Agent/TCP]

$ns attach-agent $n0 $tcp

set ftp [new Application/FTP]

$ftp attach-agent $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n7 $sink

$ns connect $tcp $sink

Set udp [new Agent/UDP]

$ns attach-agent $n1 $udp

Set cbr [new Application/Traffic/CBR]

$cbr attach-agent $udp

set null [new Agent/Null]

$ns attach-agent $n5 $null

$ns connect $udp $null

proc finish {} {

global ns naf trf

$ns flush-trace

Exec nam prog5.nam &

close $trf
close $naf
set tcpsize [ exec grep "^r" prog5.tr | grep "tcp" | tail -n 1 | cut -d " " -f 6]
set numtcp [ exec grep "^r" prog5.tr | grep -c "tcp"]
set tcptime  2.3
set udpsize [ exec grep "^r" prog5.tr | grep "cbr" | tail -n1 | cut -d " " -f6]
set numudp [ exec grep "^r" prog5.tr | grep -c "cbr"]
set udptime 4.0
puts "The throughput of FTP is"
puts "[ expr ($numtcp*$tcpsize)/$tcptime] bytes per second"
puts "The throughput of CBR is"
puts "[ expr ($numudp*$udpsize)/$udptime] bytes per second"
exit 0
 }
$ns at 0.1 "$cbr start"
$ns at 2.0 "$ftp start"
$ns at 1.9 "$cbr stop"
```

```
$ns at 4.3 "$ftp stop"

$ns at 6.0 "finish"

$ns run
```

### Output

```
throughput of ftp is 228106.666bytes/sec
throughput of cbr is b 54600.00 bytes/sec
```



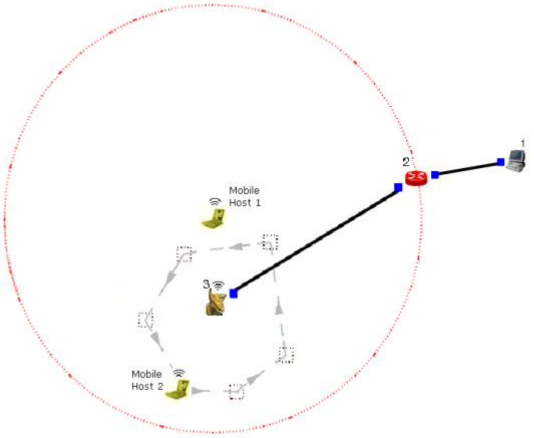
0.678000 Step: 2.0ms

Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑ Recalc

## Experiment No 5

### Problem Statement

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



### Program

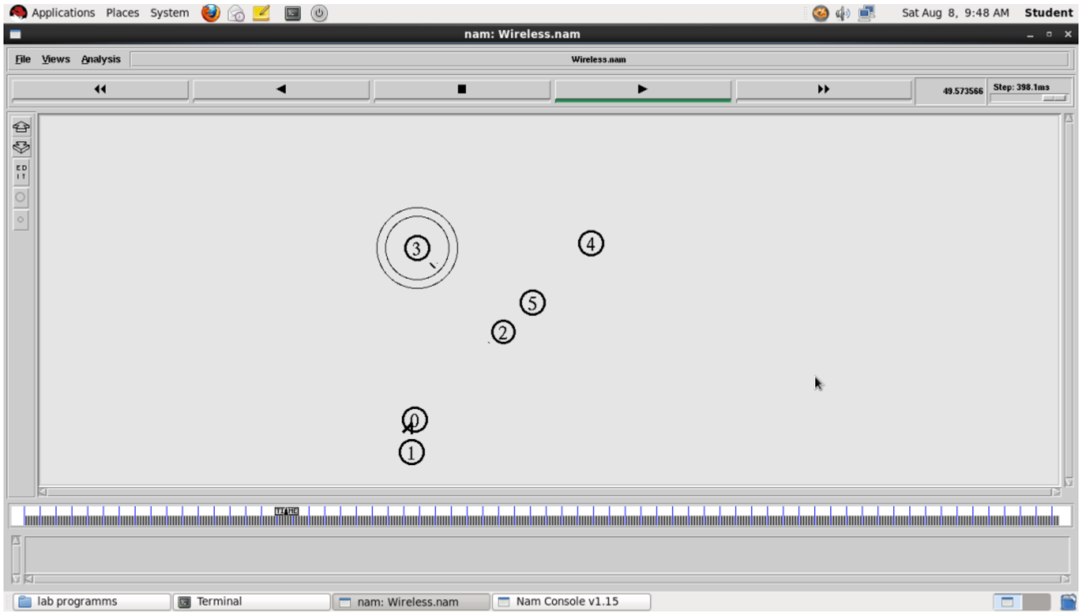
```
set val(chan) Channel/WirelessChannel

set val(prop) Propagation/TwoRayGround

set val(netif) Phy/WirelessPhy

set val(mac) Mac/802_11

set val(ifq) Queue/DropTail/PriQueue

set val(ll) LL

set val(ant) Antenna/OmniAntenna
set val(X) 500

set val(Y) 500

set val(ifqlen) 50

set val(nn) 6

set val(stop) 200.0

set val(rp) AODV

set ns [new Simulator]

set nf [open Wireless.tr w]

$ns trace-all $nf

set namtrace [open Wireless.nam w]

$ns namtrace-all-wireless $namtrace $val(X) $val(Y)

set topo [new Topography]

$topo load_flatgrid $val(X) $val(Y)

create-god $val(nn)

$ns node-config -adhocRouting $val(rp) \

 -llType $val(ll) \

 -macType $val(mac) \

 -ifqType $val(ifq) \

 -ifqLen $val(ifqlen) \

 -antType $val(ant) \

 -propType $val(prop) \

 -phyType $val(netif) \

 -channelType $val(chan) \

 -topoInstance $topo \

 -agentTrace ON \

 -routerTrace ON \
 -macTrace ON

for {set i 0} {$i<$val(nn)} {incr i} {

set node_($i) [$ns node]

$node_($i) random-motion 0

}


for {set i 0} {$i<$val(nn)} {incr i} {

$ns initial_node_pos $node_($i) 40

}

set lan [$ns newLan "$node_(0) $node_(1)" 0.5Mb 40ms LL DropTail Channel]

$ns at 10.0 "$node_(0) setdest 10.0 10.0 20.0"

$ns at 10.0 "$node_(2) setdest 160.0 160.0 20.0"

$ns at 10.0 "$node_(3) setdest 10.0 310.0 20.0"

$ns at 10.0 "$node_(4) setdest 310.0 310.0 20.0"

$ns at 10.0 "$node_(5) setdest 210.0 210.0 20.0"


set tcp [new Agent/TCP]

set tcpsink [new Agent/TCPSink]

$ns attach-agent $node_(0) $tcp

$ns attach-agent $node_(3) $tcpsink

$ns connect $tcp $tcpsink


set ftp [new Application/FTP]

$ftp set packetSize_ 500

$ftp set interval_ 5

$ftp attach-agent $tcp
```

```
set udp [new Agent/UDP]

set null [new Agent/Null]

$ns attach-agent $node_(0) $udp

$ns attach-agent $node_(2) $null

$ns connect $udp $null


set cbr [new Application/Traffic/CBR]

$cbr set packetSize_ 500

$cbr set interval_ 5

$cbr attach-agent $udp


set tcp1 [new Agent/TCP]

set tcpsink1 [new Agent/TCPSink]

$ns attach-agent $node_(0) $tcp1

$ns attach-agent $node_(4) $tcpsink1

$ns connect $tcp1 $tcpsink1


set ftp1 [new Application/FTP]

$ftp1 set packetSize_ 500

$ftp1 set interval_ 5

$ftp1 attach-agent $tcp1


set udp1 [new Agent/UDP]

set null1 [new Agent/Null]

$ns attach-agent $node_(0) $udp1
```

```
$ns attach-agent $node_(5) $null1

$ns connect $udp1 $null1


set cbr1 [new Application/Traffic/CBR]

$cbr1 set packetSize_ 500

$cbr1 set interval_ 5

$cbr1 attach-agent $udp1


$ns at 5 "$cbr start"

$ns at 195 "$cbr stop"

$ns at 5 "$ftp start"

$ns at 195 "$ftp stop"

$ns at 5 "$ftp1 start"

$ns at 195 "$ftp1 stop"

$ns at 5 "$cbr1 start"

$ns at 195 "$cbr1 stop"

proc finish {} {

global ns nf namtrace

$ns flush-trace

close $nf

close $namtrace

exec nam Wireless.nam &

exit 0

}


$ns at 200 "finish"
```

```
$ns run
```

## Output



## Experiment No 6

# To Simulate transmission of ping messages over a network topology using ns-2

**Program**

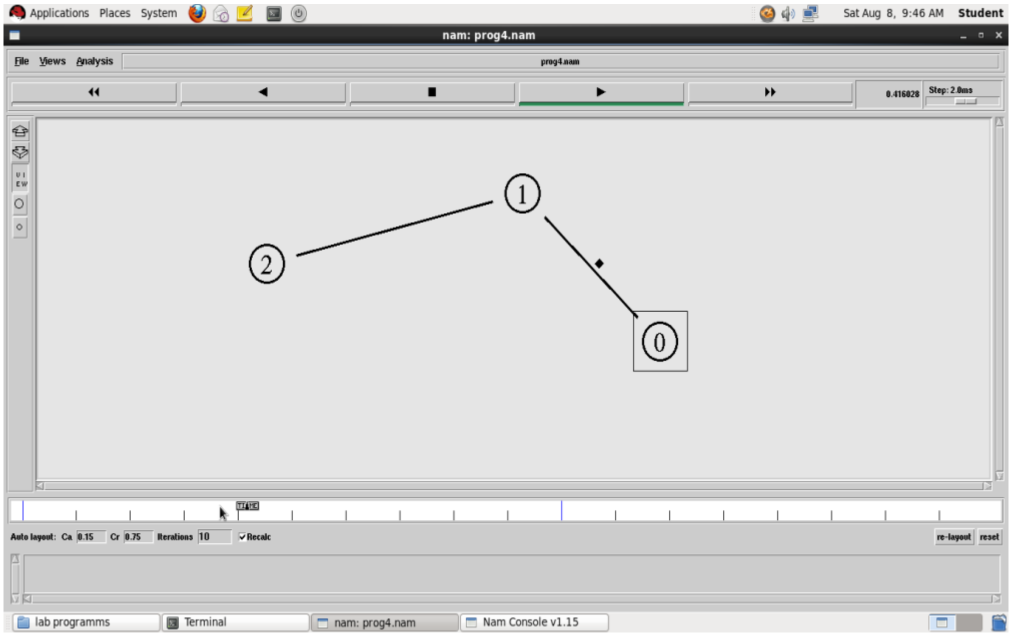
```
#Create Simulator
set ns [new Simulator]
#Open trace and NAM trace file
set ntrace [open prog4.tr w]
$ns trace-all $ntrace
set namfile [open prog4.nam w]
$ns namtrace-all $namfile
#Finish Procedure
proc Finish {} {
   global ns ntrace namfile
```

```
    #Dump all trace data and close the file
    $ns flush-trace
    close $ntrace
    close $namfile
    #Execute the nam animation file
    exec nam prog4.nam &
    exit 0
}
#Create 3 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail

$ns duplex-link $n1 $n2 1Mb 10ms DropTail
```

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


Agent/Ping instproc recv {from rtt} {

   $self instvar node_

   puts "$from received ping answer from node [$node_ id] with round trip time $rtt ms"

}

#Create two ping agents and attach them to n(0) and n(2)

set p0 [new Agent/Ping]

$ns attach-agent $n0 $p0

set p1 [new Agent/Ping]

$ns attach-agent $n2 $p1

$ns connect $p0 $p1
```

```
#Schedule events

$ns at 0.2 "$p0 send"

$ns at 0.4 "$p1 send"

$ns at 1.2 "$p0 send"

$ns at 1.7 "$p1 send"

$ns at 1.8 "Finish"

#Run the Simulation
$ns run
```

**Output**



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑ Recalc

## Experiment No 7

Develop a program that implements dynamic routing algorithm using 6 nodes

### Program

#Create a simulator object

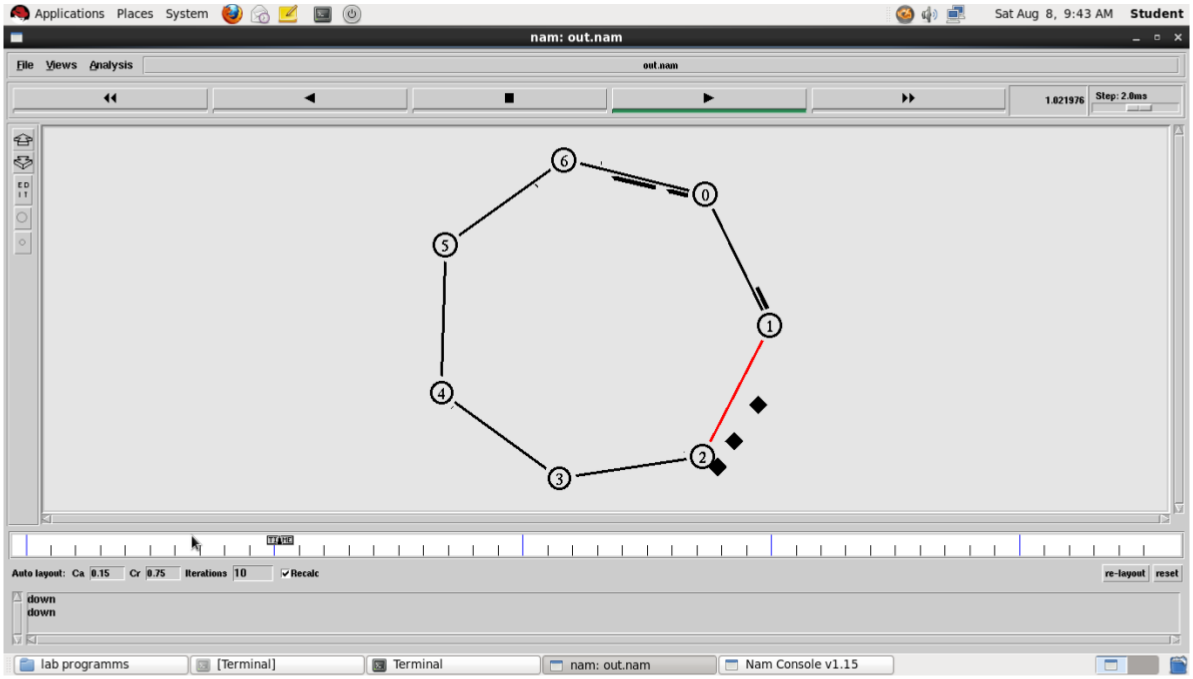
set ns [new Simulator]

```
#Tell the simulator to use dynamic routing

$ns rtproto DV



#Open the nam trace file

set nf [open out.nam w]

$ns namtrace-all $nf




#Define a 'finish' procedure

proc finish {} {

        global ns nf

        $ns flush-trace

      #Close the trace file

        close $nf

      #Execute nam on the trace file

        exec nam out.nam &

        exit 0

}
```

```
#Create seven nodes

for {set i 0} {$i < 7} {incr i} {

        set n($i) [$ns node]

}




#Create links between the nodes

for {set i 0} {$i < 7} {incr i} {

        $ns duplex-link $n($i) $n([expr ($i+1)%7]) 1Mb 10ms DropTail

}



#Create a UDP agent and attach it to node n(0)

set udp0 [new Agent/UDP]

$ns attach-agent $n(0) $udp0
# Create a CBR traffic source and attach it to udp0

set cbr0 [new Application/Traffic/CBR]

$cbr0 set packetSize_ 500

$cbr0 set interval_ 0.005

$cbr0 attach-agent $udp0
```

```
#Create a Null agent (a traffic sink) and attach it to node n(3)
set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0

#Connect the traffic source with the traffic sink
$ns connect $udp0 $null0

#Schedule events for the CBR agent and the network dynamics
$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(1) $n(2)
$ns rtmodel-at 2.0 up $n(1) $n(2)
$ns at 4.5 "$cbr0 stop"
#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Run the simulation
$ns run
```

## Output



```
down
down
```

## Experiment No 8

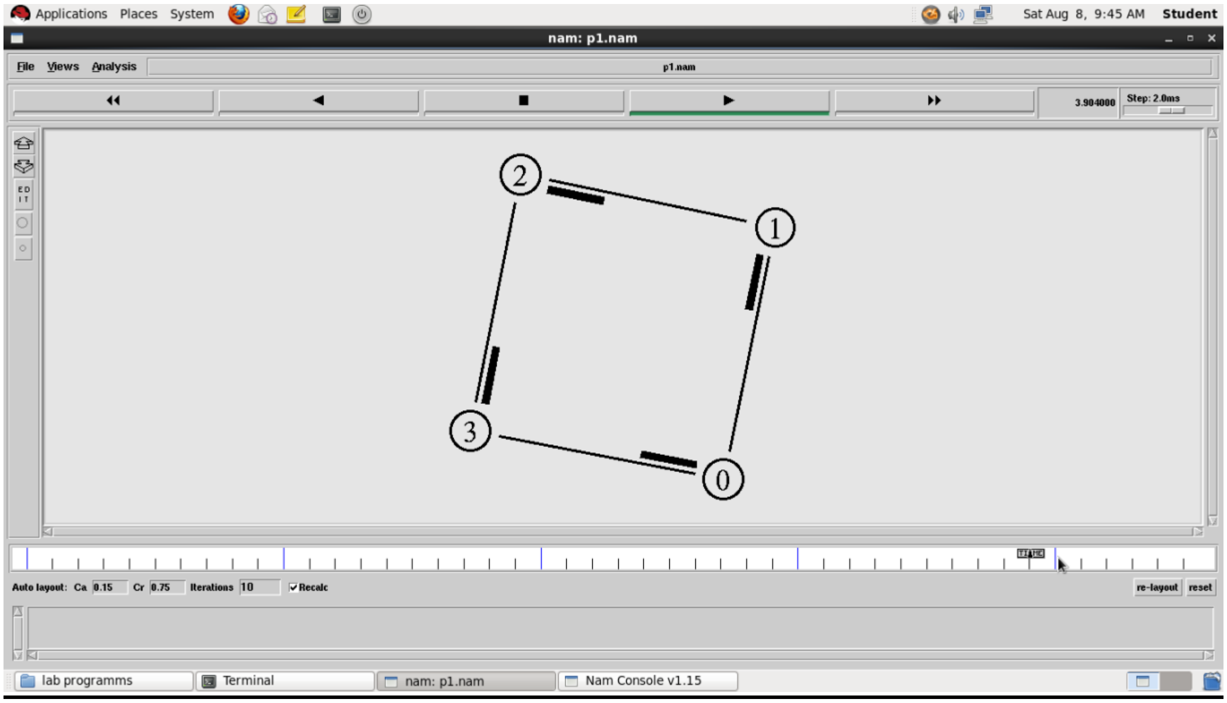
# Implement the method of cyclic data transmission using UDP protocol

```
set ns [new Simulator]

set nf [open p1.tr w]

$ns trace-all $nf
set ntrace [open p1.nam w]
$ns namtrace-all $ntrace

for {set i 0} { $i<4 } {incr i} {
   set n($i) [$ns node] }
for {set i 0} { $i<4 } {incr i} {
   $ns duplex-link $n($i) $n([expr ($i+1)%4]) 1Mb 10ms DropTail }


set udp [new Agent/UDP]
set null [new Agent/Null]
$ns attach-agent $n(0) $udp
$ns attach-agent $n(1) $null
$ns connect $udp $null

set cbr [new Application/Traffic/CBR]
$cbr set interval_ 0.005
$cbr set packetSize_ 500
$cbr attach-agent $udp


set udp1 [new Agent/UDP]
set null1 [new Agent/Null]
$ns attach-agent $n(1) $udp1
$ns attach-agent $n(2) $null1
$ns connect $udp1 $null1

set cbr1 [new Application/Traffic/CBR]
$cbr1 set interval_ 0.005
$cbr1 set packetSize_ 500
$cbr1 attach-agent $udp1
```

```
set udp2 [new Agent/UDP]
set null2 [new Agent/Null]
$ns attach-agent $n(2) $udp2
$ns attach-agent $n(3) $null2
$ns connect $udp2 $null2

set cbr2 [new Application/Traffic/CBR]
$cbr2 set interval_ 0.005
$cbr2 set packetSize_ 500
$cbr2 attach-agent $udp2


set udp3 [new Agent/UDP]
set null3 [new Agent/Null]
$ns attach-agent $n(3) $udp3
$ns attach-agent $n(0) $null3
$ns connect $udp3 $null3

set cbr3 [new Application/Traffic/CBR]
$cbr3 set interval_ 0.005
$cbr3 set packetSize_ 500
$cbr3 attach-agent $udp3

proc Finish { } {
global ns nf ntrace
$ns flush-trace
close $nf
close $ntrace
exec nam p1.nam &
exit 0
}

$ns at 0.5 "$cbr start"
$ns at 4.5 "$cbr stop"
$ns at 0.5 "$cbr1 start"
$ns at 4.5 "$cbr1 stop"
$ns at 0.5 "$cbr2 start"
$ns at 4.5 "$cbr2 stop"
```

```
$ns at 0.5 "$cbr3 start"
$ns at 4.5 "$cbr3 stop"
$ns at 5.0 "Finish"
$ns run
```

## Output



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑ Recalc

**Experiment No 9**

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

## Program

```
#include<stdio.h>
#define gen 0x11021
long int msb(long int temp);
long int checksum(long int frame);
int main() {
    long int opframe,inframe,flag,r_frame;
    printf("Enter the frame to be transmitted in hex:");
    scanf("%lx",&inframe);
    inframe=inframe<<16;
    opframe=inframe^(checksum(inframe));
    printf("The frame to be transmitted is %lx\n",opframe);
    printf("Enter the received frame:");
    scanf("%lx",&r_frame);
    printf("For the received frame\n ");
    flag=checksum(r_frame);
    if(flag>0)
            printf("\nError!!!!\n");
    else
            printf("\ndata received is error free\n");
}

long int checksum(long int frame)
{
    long int posit_frame,posit_gen,g=gen,g1,temp;
    posit_frame=msb(frame);
    posit_gen=msb(g);
    while(posit_gen<=posit_frame) {

            g1=g<<(posit_frame-posit_gen);
            frame=g1^frame;
            posit_frame=msb(frame);
```

```
        }
        printf("The checksum is %lx\n",frame);
        return(frame);
    }

    long int msb(long int temp){
        int i=0;
        if(temp==0)
        return 0;
        while(temp>0)
        {
                temp=temp<<1;
                i++;
        }
        return(32-i);
    }
```

**Output**

```
    Enter the frame to be transmitted in hex:2ab6
    The checksum is 2e30
    The frame to be trnsmitted is 2ab62e30
```

## Experiment No 10

Write a program for Hamming Code generation for error detection and correction

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
#include<stdio.h>

char data[5];
int encoded[8], edata[7], syndrome[3];
int hmatrix[3][7]= { 1,0,0,0,1,1,1,
                     0,1,0,1,0,1,1,
                         0,0,1,1,1,0,1};
char gmatrix[4][8]={ "0111000", "1010100", "1100010", "1110001"};
void main() {
  int i,j;
  clrscr();
  cout<<"Hamming Code --- Encoding\n";
  cout<<"Enter 4 bit data : ";
  cin>>data;
  cout<<"Generator Matrix\n";
  for(i=0;i<4;i++) cout<<"\t"<<gmatrix[i]<<"\n";
  cout<<"Encoded Data : ";
  for(i=0;i<7;i++) {
    for(j=0;j<4;j++)
```

```
      encoded[i]+=((data[j]- '0')*(gmatrix[j][i]- '0'));
    encoded[i]=encoded[i]%2;
    cout<<encoded[i]<<" ";
  }
  cout<<"\nHamming code --- Decoding\n";
  cout<<"Enter Encoded bits as received : ";
  for(i=0;i<7;i++) cin>>edata[i];
  for(i=0;i<3;i++) {
    for(j=0;j<7;j++)
      syndrome[i]=syndrome[i]+(edata[j]*hmatrix[i][j]);
    syndrome[i]=syndrome[i]%2;
  }
  for(j=0;j<7;j++)
    if ((syndrome[0]==hmatrix[0][j])&&(syndrome[1]==hmatrix[1][j])&&
            (syndrome[2]==hmatrix[2][j]))
      break;
  if(j==7)
    cout<<"Data is error free!!\n";
  else {
    cout<<"Error received at bit number "<<j+1<<" of the data\n";
    edata[j]=!edata[j];
    cout<<"The Correct data Should be : ";
    for(i=0;i<7;i++) cout<<edata[i]<<" ";
    }
  }
}
```

**Output**

```
Hamming Code --- Encoding
```

```
Enter 4 bit data : 1 0 1 0
Generator Matrix
      0111000
      1010100
      1100010
      1110001
Encoded Data : 1 0 1 1 0 1 0
Hamming code --- Decoding
Enter Encoded bits as received : 1 0 1 1 0 1 1
Error received at bit number 7 of the data
The Correct data Should be : 1 0 1 1 0 1 0
Hamming Code --- Encoding
Enter 4 bit data : 1 0 1 0
Generator Matrix
      0111000
      1010100
      1100010
      1110001
Encoded Data : 1 0 1 1 0 1 0
Hamming code --- Decoding
Enter Encoded bits as received : 1 0 1 1 0 1 0
Data is error free!!
```

## Experiment No 11

Write a program for plotting xgraph  using exponential traffic

```
#Create a simulator object
set ns [new Simulator]

#Open the output files
set f0 [open out0.tr w]
set f1 [open out1.tr w]
set f2 [open out2.tr w]

#Create 5 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]

#Connect the nodes
$ns duplex-link $n0 $n3 1Mb 100ms DropTail
$ns duplex-link $n1 $n3 1Mb 100ms DropTail
$ns duplex-link $n2 $n3 1Mb 100ms DropTail
$ns duplex-link $n3 $n4 1Mb 100ms DropTail

#Define a 'finish' procedure
proc finish {} {
```

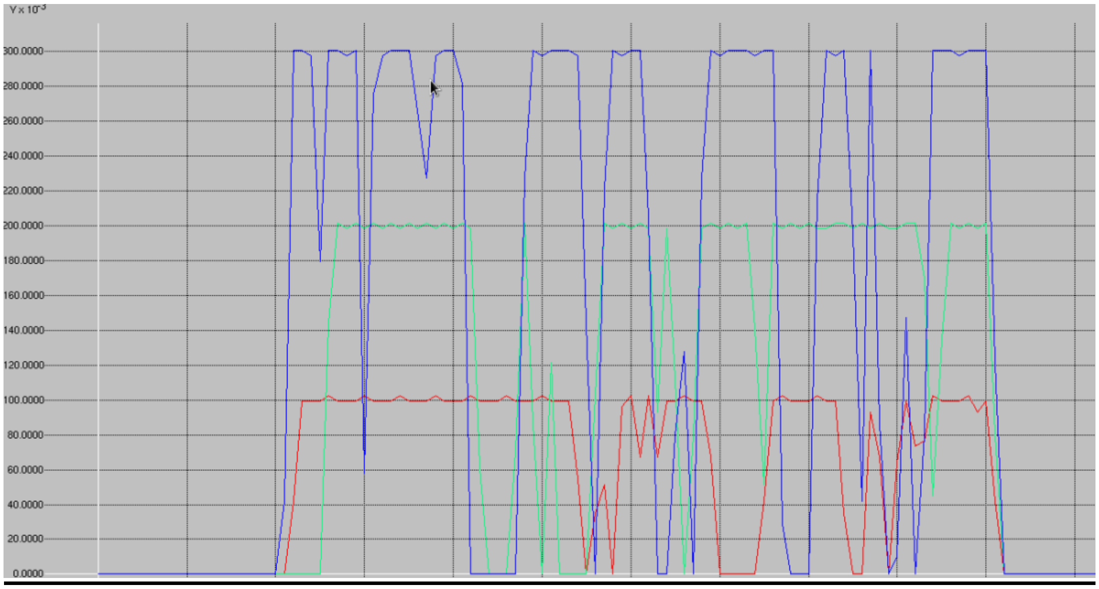
```
            global f0 f1 f2
            #Close the output files
            close $f0
            close $f1
            close $f2
            #Call xgraph to display the results
            exec xgraph out0.tr out1.tr out2.tr -geometry 800x400 &
        exit 0
}


#Define a procedure that attaches a UDP agent to a previously created node
#'node' and attaches an Expoo traffic generator to the agent with the
#characteristic values 'size' for packet size 'burst' for burst time,
#'idle' for idle time and 'rate' for burst peak rate. The procedure connects
#the source with the previously defined traffic sink 'sink' and returns the
#source object.
proc attach-expoo-traffic { node sink size burst idle rate } {
            #Get an instance of the simulator
            set ns [Simulator instance]

            #Create a UDP agent and attach it to the node
            set source [new Agent/UDP]
            $ns attach-agent $node $source

            #Create an Expoo traffic agent and set its configuration parameters
            set traffic [new Application/Traffic/Exponential]
            $traffic set packetSize_ $size
            $traffic set burst_time_ $burst
```

```
          $traffic set idle_time_ $idle
          $traffic set rate_ $rate

       # Attach traffic source to the traffic generator
       $traffic attach-agent $source
          #Connect the source and the sink
          $ns connect $source $sink
          return $traffic
}


#Define a procedure which periodically records the bandwidth received by the
#three traffic sinks sink0/1/2 and writes it to the three files f0/1/2.
proc record {} {
       global sink0 sink1 sink2 f0 f1 f2
          #Get an instance of the simulator
          set ns [Simulator instance]
          #Set the time after which the procedure should be called again
       set time 0.5
          #How many bytes have been received by the traffic sinks?
       set bw0 [$sink0 set bytes_]
       set bw1 [$sink1 set bytes_]
       set bw2 [$sink2 set bytes_]
          #Get the current time
       set now [$ns now]
          #Calculate the bandwidth (in MBit/s) and write it to the files
       puts $f0 "$now [expr $bw0/$time*8/1000000]"
       puts $f1 "$now [expr $bw1/$time*8/1000000]"
       puts $f2 "$now [expr $bw2/$time*8/1000000]"
```

```
        #Reset the bytes_ values on the traffic sinks
      $sink0 set bytes_ 0
      $sink1 set bytes_ 0
      $sink2 set bytes_ 0
        #Re-schedule the procedure
      $ns at [expr $now+$time] "record"
}


#Create three traffic sinks and attach them to the node n4
set sink0 [new Agent/LossMonitor]
set sink1 [new Agent/LossMonitor]
set sink2 [new Agent/LossMonitor]
$ns attach-agent $n4 $sink0
$ns attach-agent $n4 $sink1
$ns attach-agent $n4 $sink2

#Create three traffic sources
set source0 [attach-expoo-traffic $n0 $sink0 200 2s 1s 100k]
set source1 [attach-expoo-traffic $n1 $sink1 200 2s 1s 200k]
set source2 [attach-expoo-traffic $n2 $sink2 200 2s 1s 300k]

#Start logging the received bandwidth
$ns at 0.0 "record"
#Start the traffic sources
$ns at 10.0 "$source0 start"
$ns at 10.0 "$source1 start"
$ns at 10.0 "$source2 start"
#Stop the traffic sources
```

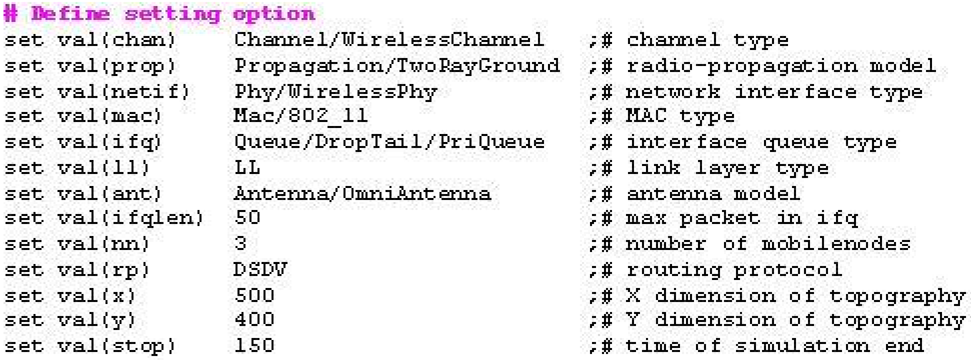
```
$ns at 50.0 "$source0 stop"
$ns at 50.0 "$source1 stop"
$ns at 50.0 "$source2 stop"
#Call the finish procedure after 60 seconds simulation time
$ns at 60.0 "finish"

#Run the simulation
$ns run
```

## Output



# Experiment-12

Write a program to create mobile nodes using Destination-Sequenced Distance-Vector Routing (DSDV) protocol

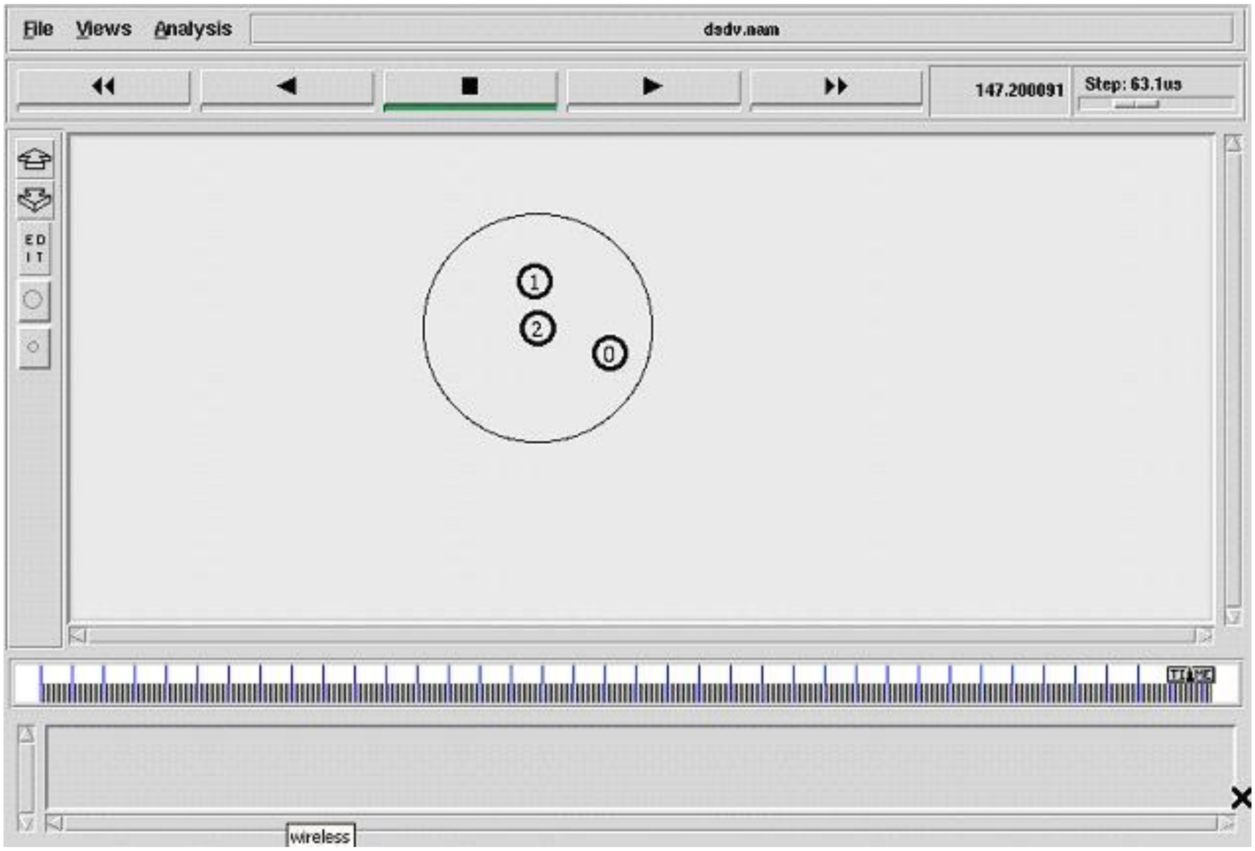


```
# Define setting option
set val(chan)     Channel/WirelessChannel   ;# channel type
set val(prop)     Propagation/TwoRayGround  ;# radio-propagation model
set val(netif)    Phy/WirelessPhy           ;# network interface type
set val(mac)      Mac/802_11                ;# MAC type
set val(ifq)      Queue/DropTail/PriQueue   ;# interface queue type
set val(ll)       LL                        ;# link layer type
set val(ant)      Antenna/OmniAntenna       ;# antenna model
set val(ifqlen)   50                        ;# max packet in ifq
set val(nn)       3                         ;# number of mobilenodes
set val(rp)       DSDV                      ;# routing protocol
set val(x)        500                       ;# X dimension of topography
set val(y)        400                       ;# Y dimension of topography
set val(stop)     150                       ;# time of simulation end
```

---

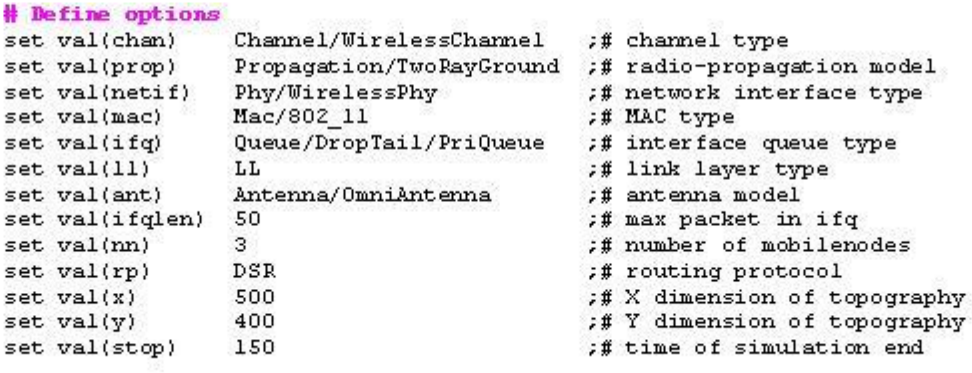
```
#Creating trace file and nam file
set tracefd       [open dsdv.tr w]
set windowVsTime2 [open win.tr w]
set namtrace      [open dsdv.nam w]

$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y)

# set up topography object
set topo       [new Topography]

$topo load_flatgrid $val(x) $val(y)

create-god $val(nn)

# configure the nodes
        $ns node-config -adhocRouting $val(rp) \
                   -llType $val(ll) \
                   -macType $val(mac) \
                   -ifqType $val(ifq) \
                   -ifqLen $val(ifqlen) \
                   -antType $val(ant) \
                   -propType $val(prop) \
                   -phyType $val(netif) \
                   -channelType $val(chan) \
                   -topoInstance $topo \
                   -agentTrace ON \
                   -routerTrace ON \
                   -macTrace OFF \
                   -movementTrace ON

         for {set i 0} {$i < $val(nn) } { incr i } {
             set node_($i) [$ns node]
         }

# Provide initial location of mobilenodes
$node_(0) set X_ 5.0
$node_(0) set Y_ 5.0
$node_(0) set Z_ 0.0

$node_(1) set X_ 490.0
$node_(1) set Y_ 285.0
$node_(1) set Z_ 0.0

$node_(2) set X_ 150.0
$node_(2) set Y_ 240.0
$node_(2) set Z_ 0.0

# Generation of movements
$ns at 10.0 "$node_(0) setdest 250.0 250.0 3.0"
$ns at 15.0 "$node_(1) setdest 45.0 285.0 5.0"
$ns at 110.0 "$node_(0) setdest 480.0 300.0 5.0"

# Set a TCP connection between node_(0) and node_(1)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $node_(0) $tcp
$ns attach-agent $node_(1) $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
```

```
$ftp attach-agent $tcp
$ns at 10.0 "$ftp start"

# Printing the window size
proc plotWindow {tcpSource file} {
global ns
set time 0.01
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file" }
$ns at 10.1 "plotWindow $tcp $windowVsTime2"

# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns initial_node_pos $node_($i) 30
}

# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
   $ns at $val(stop) "$node_($i) reset";
}

# ending nam and the simulation
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "stop"
$ns at 150.01 "puts \"end simulation\" ; $ns halt"
proc stop {} {
   global ns tracefd namtrace
   $ns flush-trace
   close $tracefd
   close $namtrace
exec nam dsdv.nam &
exit 0
}

$ns run

# How to run the program:

$ns dsdv.tcls
```


File
Views
Analysis
dsdv.nam
147.200091
Step: 63.1us
EDIT
1
2
0
TIME
wireless

## Experiment-13

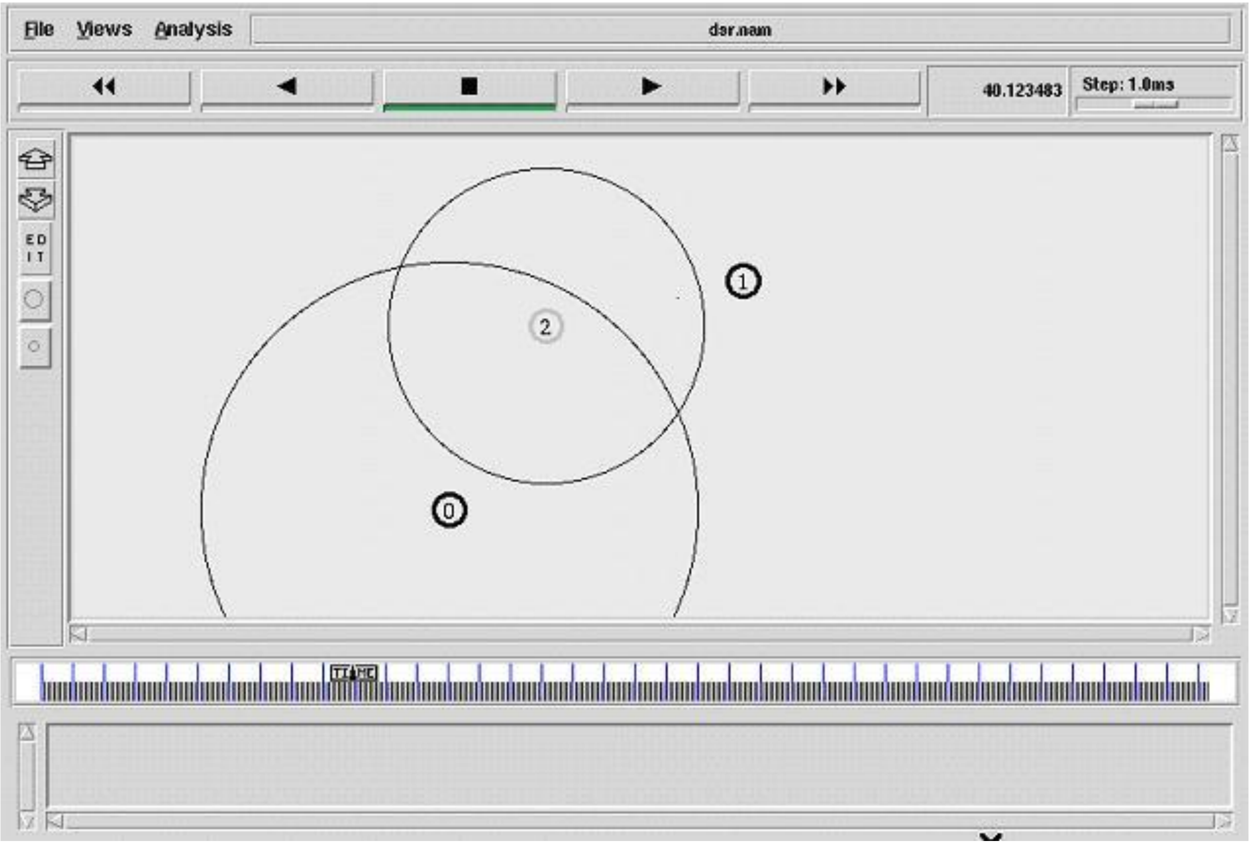
Write a program to create mobile nodes using Destination-Sequenced Dynamic Source Routing protocol (DSR) protocol



```
# Define options
set val(chan)     Channel/WirelessChannel   ;# channel type
set val(prop)     Propagation/TwoRayGround  ;# radio-propagation model
set val(netif)    Phy/WirelessPhy           ;# network interface type
set val(mac)      Mac/802_11                ;# MAC type
set val(ifq)      Queue/DropTail/PriQueue   ;# interface queue type
set val(ll)       LL                        ;# link layer type
set val(ant)      Antenna/OmniAntenna       ;# antenna model
set val(ifqlen)   50                        ;# max packet in ifq
set val(nn)       3                         ;# number of mobilenodes
set val(rp)       DSR                       ;# routing protocol
set val(x)        500                       ;# X dimension of topography
set val(y)        400                       ;# Y dimension of topography
set val(stop)     150                       ;# time of simulation end
```

```
#-------Event scheduler object creation--------#

set ns              [new Simulator]
#Creating trace file and nam file
set tracefd       [open dsr.tr w]
set windowVsTime2 [open win.tr w]
set namtrace      [open dsr.nam w]

$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y)

# set up topography object
set topo       [new Topography]

$topo load_flatgrid $val(x) $val(y)

create-god $val(nn)

# configure the nodes
        $ns node-config -adhocRouting $val(rp) \
                   -llType $val(ll) \
                   -macType $val(mac) \
                   -ifqType $val(ifq) \
                   -ifqLen $val(ifqlen) \
                   -antType $val(ant) \
                   -propType $val(prop) \
                   -phyType $val(netif) \
                   -channelType $val(chan) \
                   -topoInstance $topo \
                   -agentTrace ON \
                   -routerTrace ON \
                   -macTrace OFF \
                   -movementTrace ON

        for {set i 0} {$i < $val(nn) } { incr i } {
             set node_($i) [$ns node]
        }

# Provide initial location of mobilenodes
$node_(0) set X_ 5.0
$node_(0) set Y_ 5.0
$node_(0) set Z_ 0.0

$node_(1) set X_ 490.0
$node_(1) set Y_ 285.0
$node_(1) set Z_ 0.0

$node_(2) set X_ 150.0
$node_(2) set Y_ 240.0
$node_(2) set Z_ 0.0

# Generation of movements
$ns at 10.0 "$node_(0) setdest 250.0 250.0 3.0"
$ns at 15.0 "$node_(1) setdest 45.0 285.0 5.0"
$ns at 110.0 "$node_(0) setdest 480.0 300.0 5.0"

# Set a TCP connection between node_(0) and node_(1)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $node_(0) $tcp
$ns attach-agent $node_(1) $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 10.0 "$ftp start"
```

```
# Printing the window size
proc plotWindow {tcpSource file} {
global ns
set time 0.01
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file" }
$ns at 10.1 "plotWindow $tcp $windowVsTime2"

# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns initial_node_pos $node_($i) 30
}

# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
   $ns at $val(stop) "$node_($i) reset";
}

# ending nam and the simulation
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "stop"
$ns at 150.01 "puts \"end simulation\" ; $ns halt"
proc stop {} {
   global ns tracefd namtrace
   $ns flush-trace
   close $tracefd
   close $namtrace
exec nam dsr.nam &
exit 0
}

$ns run
```


File
Views
Analysis
dsr.nam
40.123483
Step: 1.0ms
EDIT
1
2
0
TIME

## Experiment-14

Write a program to create multicast network in ns2?

```
set ns [new Simulator -multicast on] ;# enable multicast routing
set trace [open test19.tr w]
$ns trace-all $trace
#$ns use-newtrace
set namtrace [open test19.nam w]
$ns namtrace-all $namtrace
set group [Node allocaddr] ;# allocate a multicast address
set node0 [$ns node] ;# create multicast capable nodes
set node1 [$ns node]
set node2 [$ns node]
$ns duplex-link $node0 $node1 1.5Mb 10ms DropTail
$ns duplex-link $node0 $node2 1.5Mb 10ms DropTail
set mproto DM ;# configure multicast protocol
set mrthandle [$ns mrtproto $mproto] ;
set udp [new Agent/UDP]
$ns attach-agent $node0 $udp
set src [new Application/Traffic/CBR]
$src attach-agent $udp
$udp set dst_addr_ $group
$udp set dst_port_ 0
set rcvr [new Agent/LossMonitor] ;
$ns attach-agent $node1 $rcvr
$ns at 0.3 "$node1 join-group $rcvr $group"
set rcvr2 [new Agent/LossMonitor]
$ns attach-agent $node2 $rcvr2
$ns at 0.3 "$node2 join-group $rcvr2 $group"
$ns at 3.3 "$node2 leave-group $rcvr2 $group"
$ns at 2.0 "$src start"
$ns at 5.0 "$src stop"
proc finish {} {
global ns namtrace trace
$ns flush-trace
close $namtrace ; close $trace
exec nam test19.nam &
exit 0
}
$ns at 10.0 "finish"
$ns run
```

## Experiment-15

NS2 simulation using Link State routing protocol

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tr [open out.tr w]
$ns trace-all $tr
proc finish {} {
        global nf ns tr
        $ns flush-trace
        close $tr
        exec nam out.nam &
        exit 0
        }
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n3 10Mb 10ms DropTail
$ns duplex-link $n2 $n1 10Mb 10ms DropTail
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n1 $n3 orient right
$ns duplex-link-op $n2 $n1 orient right-up
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
set udp [new Agent/UDP]
$ns attach-agent $n2 $udp

set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
set null [new Agent/Null]
```

```
$ns attach-agent $n3 $null
$ns connect $tcp $sink
$ns connect $udp $null
$ns rtmodel-at 1.0 down $n1 $n3
$ns rtmodel-at 2.0 up $n1 $n3
$ns rtproto LS
$ns at 0.0 "$ftp start"
$ns at 0.0 "$cbr start"
$ns at 5.0 "finish"
$ns run
```

## Experiment-16

NS2 simulation using Distance Vector routing protocol

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tr [open out.tr w]
$ns trace-all $tr
proc finish {} {
        global nf ns tr
        $ns flush-trace
        close $tr
        exec nam out.nam &
        exit 0
        }
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n3 10Mb 10ms DropTail
$ns duplex-link $n2 $n1 10Mb 10ms DropTail
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n1 $n3 orient right
$ns duplex-link-op $n2 $n1 orient right-up
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
set udp [new Agent/UDP]
$ns attach-agent $n2 $udp
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $tcp $sink
```

```
$ns connect $udp $null

$ns rtmodel-at 1.0 down $n1 $n3
$ns rtmodel-at 2.0 up $n1 $n3
$ns rtproto DV
$ns at 0.0 "$ftp start"
$ns at 0.0 "$cbr start"
$ns at 5.0 "finish"
$ns run
```

## Experiment-17

NS2 simulation for TCP packets in a network

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set nt [open out.tr w]
$ns trace-all $nt
proc finish {} {
        global ns nf
        $ns flush-trace
        close $nf
        exec nam out.nam &
        exit 0
        }
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n3 $n1 10Mb 10ms DropTail
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n1 $n2 orient right
$ns duplex-link-op $n3 $n1 orient right-up
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set ftp [new Application/FTP]
#$ftp set packet_size_ 4.5Mb
$ftp set interval_ 0.05
$ftp attach-agent $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
$ns at 0.3 "$ftp start"
```

```
$ns at 3.0 "finish"
$ns run
```

## Experiment-17

NS2 simulation for UDP packets in a network

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set nt [open out.tr w]
$ns trace-all $nt
proc finish {} {
        global ns nf nt
        $ns flush-trace
        close $nf
        close $nt
        exec nam out.nam &
        exit 0
        }
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link-op $n0 $n1 orient right-up
$ns duplex-link-op $n0 $n2 orient right
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
set null0 [new Agent/Null]
$ns attach-agent $n2 $null0
```

```
$ns connect $udp0 $null0
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns run
```

# VIVA QUESTIONS

1. **What are 10Base2, 10Base5 and 10BaseT Ethernet LANs**?

**2. What is the difference between an unspecified passive open and a fully specified passive open**?

**3**. **Explain the function of Transmission Control Block.**

**4. What is a Management Information Base (MIB)**?

**5. What is anonymous FTP and why would you use it ?**

**6. What is the front end and back end languages used in NS2**

**7. Which layer of the 7 layer model provides services to the Application layer over the Session layer connection?**

**8. What is full form OTCL ?**

**9. What is Point to Point communication.**

**10. Which OSI Reference Layer controls application to application communication?**

**11. What is a DNS resource record ?**

**12. What is the meaning of NAM.**

**13. What protocol is used by DNS name servers** ?

**14. What is the difference between interior and exterior neighbor gateways?**

**15. What is the HELLO protocol used for** ?

**16. What are the advantages and disadvantages of the three types of routing? tables**

**18. What is source route** ?

**19. What is RIP (Routing Information Protocol)?**

**20. What is SLIP (Serial Line Interface Protocol)?**

**21. What is Proxy ARP**?

**22. What is OSPF** ?

**23. What is Kerberos ?**

**24. What is a Multi-homed Host ?**

**25. What is NVT (Network Virtual Terminal) ?**

**26. What is Gateway-to-Gateway protocol ?**

**27. What is BGP (Border Gateway Protocol)?**

**28. What is autonomous system?**

**29. What is EGP (Exterior Gateway Protocol)?**

**30. What is IGP (Interior Gateway Protocol)?**

**31. What is Mail Gateway?**

**32. What is wide-mouth frog?**

**34. What is silly window syndrome ?**

**36. What is multicast routing?**

**37. What is traffic shaping?**

**38. What is packet filter?**

**39. What is virtual path?**

**40. What is virtual channel?**
**41. What is logical link control?**
**42. Why should you care about the OSI Reference Model?**
**43. What is the difference between routable and non- routable protocols?**
**44. Name the OS used in your lab to support NS2**
**45. Explain 5-4-3 rule**
**46. What is the difference between TFTP and FTP application layer protocols**
**47. What is the range of addresses in the classes of internet addresses**
**48. What is the minimum and maximum length of the header in the TCP segment and IP datagram**
**49. What is difference between ARP and RARP?**.
**50. What is ICMP ?**
**51. What are the data units at different layers of the TCP / IP protocol suite**
**52. What is Project 802?**
**53. What is Bandwidth?**
**54. Difference between bit rate and baud rate?**
**55. What is MAC address?**
**56. What is attenuation?**
**57. What is cladding?**
**58. Explain the five components of NS2**
**59. What is post processing in NS2**
**60. What is the command used to filter in trace file**
**61. What is Beaconing**?
**62. What is terminal emulation, in which layer it comes**?
**63. What is frame relay, in which layer it comes?**
**64. What do you meant by "triple X" in Networks**?
**65. What is SAP**?
**66. What is subnet**?
**67. What is Brouter?**
**68. How Gateway is different from Routers?**
**69. What are the different type of networking / internetworking devices?**
**70. What is mesh network?**
**71. What is passive topology?**
**72. What are the important topologies for networks?**
**73. What are major types of networks and explain?**
**74. What is Protocol Data Unit**?
**75. What is difference between baseband and broadband transmission**?
**76. What are the possible ways of data exchange**?
**77. What are the types of Transmission media?**
**78. Difference between the communication and transmission.**
**79.The Internet Control Message Protocol occurs at what layer of the seven layer mode**l?
**80.Which protocol resolves an IP address to a MAC address?**

**81. MPEG are examples of what layer of the OSI seven layer model?**
**82.What is the protocol number for UDP?**
**83.Which protocol is used for booting diskless workstations?**
**84.Which layer is responsible for putting 1s and 0s into a logical group?**
**85.What does 'P' mean when running a Trace?**
**86.UDP works at which layer of the DOD model?**
**87.What is the default encapsulation of Netware 3.12?**
**88.Ping uses which Internet layer protocol?**
**89.Which switching technology can reduce the size of a broadcast domain?**
**90.What is the first step in data encapsulation?**
**91.What is the protocol number for TCP?**
**92.What is the use of Xgraph plotting in NS2**
**93.Repeaters work at which layer of the OSI model?**
**94.WAN stands for which of the following?**
**95.LAN stands for which of the following?**
**96.DHCP stands for**
**97.What does the acronym ARP stand for?**
**98.Which layer is responsible for identifying and establishing the availability of the inte**nded communication partner?
**99.Which OSI layer provides mechanical, electrical, procedural for activating, maintaining physical link?**
**100. Define Network?**
**101. What is a Link?**
**102. What is a node?**
103**. What is a gateway or Router?**
**104. What is point-point link?**
**105. What is Multiple Access?**
**106. What is the essence of RSVP ? Explain the suitable example**
**107.What is the need of scheduling and policing techniques in multimedia networking ?**
**108. What is the need of RTCP protocol along with RTP protocol in multimedia communication ?**.
**109. Explain WAN architecture in detail**
**110. Explain email architecture and its services**
**112. Explain Bluetooth architecture with diagram**
**113. Discuss various layers used in ATM architecture**