Wired programs:

1. **Simulate a point-to-point network with duplex link as follows: n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3 and UDP agent between n1-n3. Apply relevant applications over TCP and UDP agents. Set the queue size to 5 and vary the bandwidth to find number of packets dropped and received by TCP and UDP agents using awk script and grep command.**

set ns [new Simulator]

set tf [open ex1.tr w]

$ns trace-all $tf

set nf [open ex1.nam w]

$ns namtrace-all $nf

set n0 [$ns node]

set n1 [$ns node]

set n2 [$ns node]

set n3 [$ns node]

$ns duplex-link $n0 $n2 2Mb 2ms DropTail

$ns duplex-link $n1 $n2 2Mb 2ms DropTail

$ns duplex-link $n2 $n3 0.4Mb 10ms DropTail

$ns queue-limit $n1 $n2 5

set udp1 [new Agent/UDP]

$ns attach-agent $n0 $udp1

set null1 [new Agent/Null]

$ns attach-agent $n3 $null1

$ns connect $udp1 $null1

set cbr1 [new Application/Traffic/CBR]

$cbr1 attach-agent $udp1

$ns at 1.1 "$cbr1 start"

set tcp [new Agent/TCP]

$ns attach-agent $n3 $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n1 $sink

$ns connect $tcp $sink

set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ns at 0.1 "$ftp start"

$ns at 10.0 "finish"

proc finish {} {

global ns tf nf

$ns flush-trace

close $tf

close $nf

puts "running nam..."

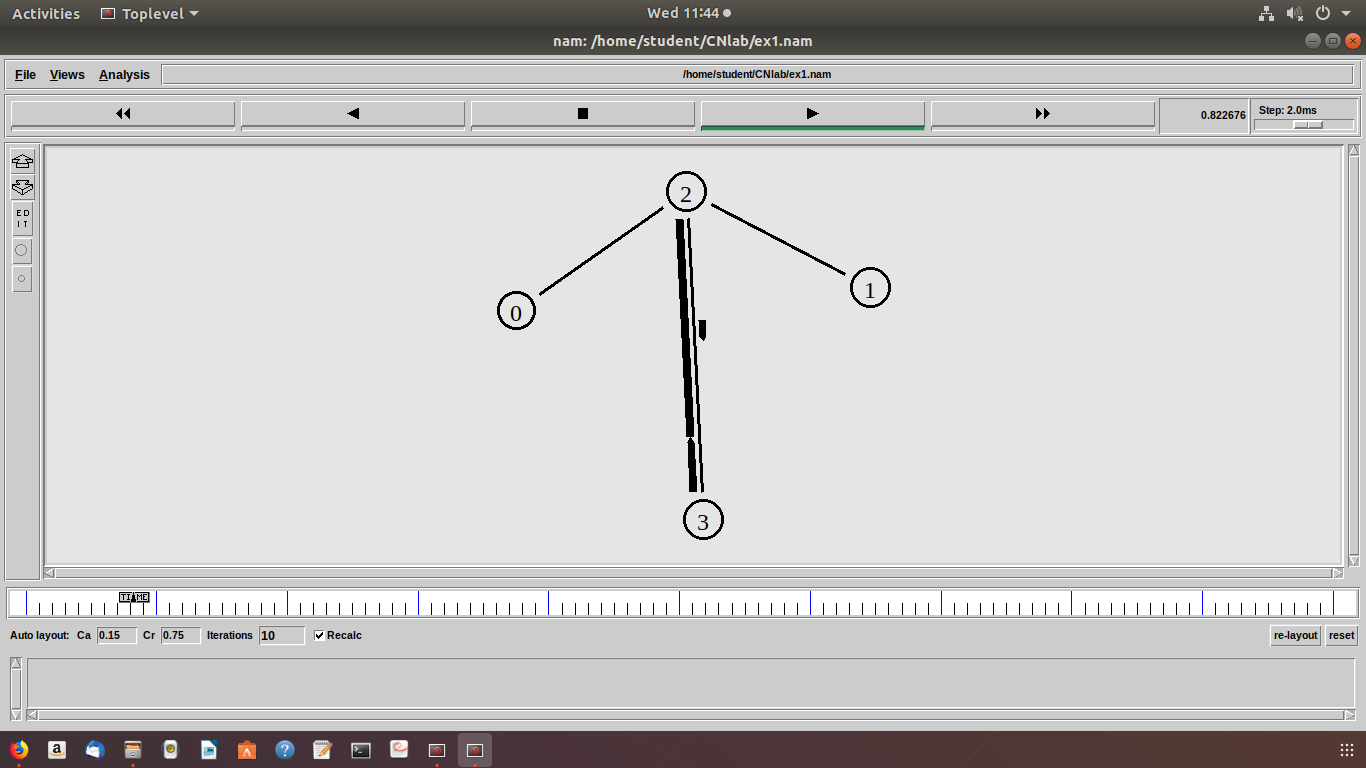
exec nam ex1.nam &

exit 0

}

$ns run

***Expected output:*** Animated 4 node structure is displayed. We need to see the trace file to understand what has happened to the data flow.

****

Grep

***grep “^r” ex1.tr*   #packets received**

To calculate number of packet dropped by TCP and udp we need to write awk script.

Save the below program as ex1.awk

BEGIN {

tcp\_count=0;

udp\_count=0;

}

{

if ( $1 == "d" && $5 == "tcp")

tcp\_count ++;

if ( $1 == "d" && $5 == "cbr")

udp\_count ++;

}

END {

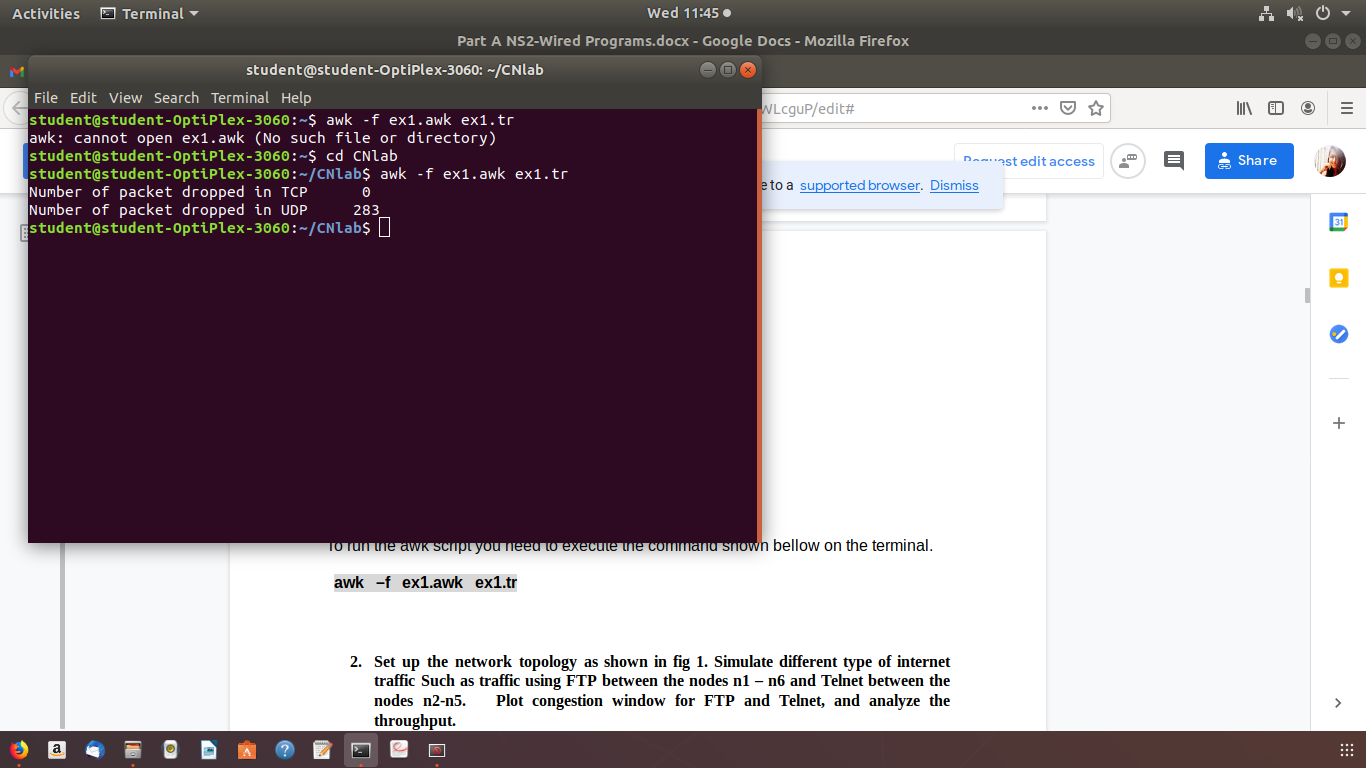
printf("Number of packet dropped in TCP %d\n", tcp\_count);

printf("Number of packet dropped in UDP %d\n", udp\_count);

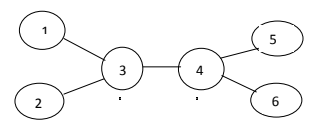
}

To run the awk script you need to execute the command shown bellow on the terminal.

**awk –f ex1.awk ex1.tr**

****

1. **Set up the network topology as shown in fig 1. Simulate different type of internet traffic Such as traffic using FTP between the nodes n1 – n6 and Telnet between the nodes n2-n5. Plot congestion window for FTP and Telnet, and analyze the throughput.**

****

**Fig. 1: Network Topology**

set ns [new Simulator]

set tf [open ex2.tr w]

$ns trace-all $tf

set nf [open ex2.nam w]

$ns namtrace-all $nf

set cwind [open win2.tr w]

set n0 [$ns node]

set n1 [$ns node]

set n2 [$ns node]

set n3 [$ns node]

$ns duplex-link $n0 $n2 5Mb 2ms DropTail

$ns duplex-link $n1 $n2 5Mb 2ms DropTail

$ns duplex-link $n2 $n3 1.5Mb 10ms 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 ftp [new Application/FTP]

$ftp attach-agent $tcp0

$ns at 1.2 "$ftp start"

set tcp1 [new Agent/TCP]

$ns attach-agent $n1 $tcp1

set sink1 [new Agent/TCPSink]

$ns attach-agent $n0 $sink1

$ns connect $tcp1 $sink1

set telnet [new Application/Telnet]

$telnet attach-agent $tcp1

$ns at 1.5 "$telnet start"

$ns at 10.0 "finish"

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 2.0 "plotWindow $tcp0 $cwind"

$ns at 5.5 "plotWindow $tcp1 $cwind"

proc finish {} {

global ns tf nf cwind

$ns flush-trace

close $tf

close $nf

close $cwind

puts "running nam..."

puts "FTP PACKETS.."

puts "Telnet PACKETS.."

exec nam ex2.nam &

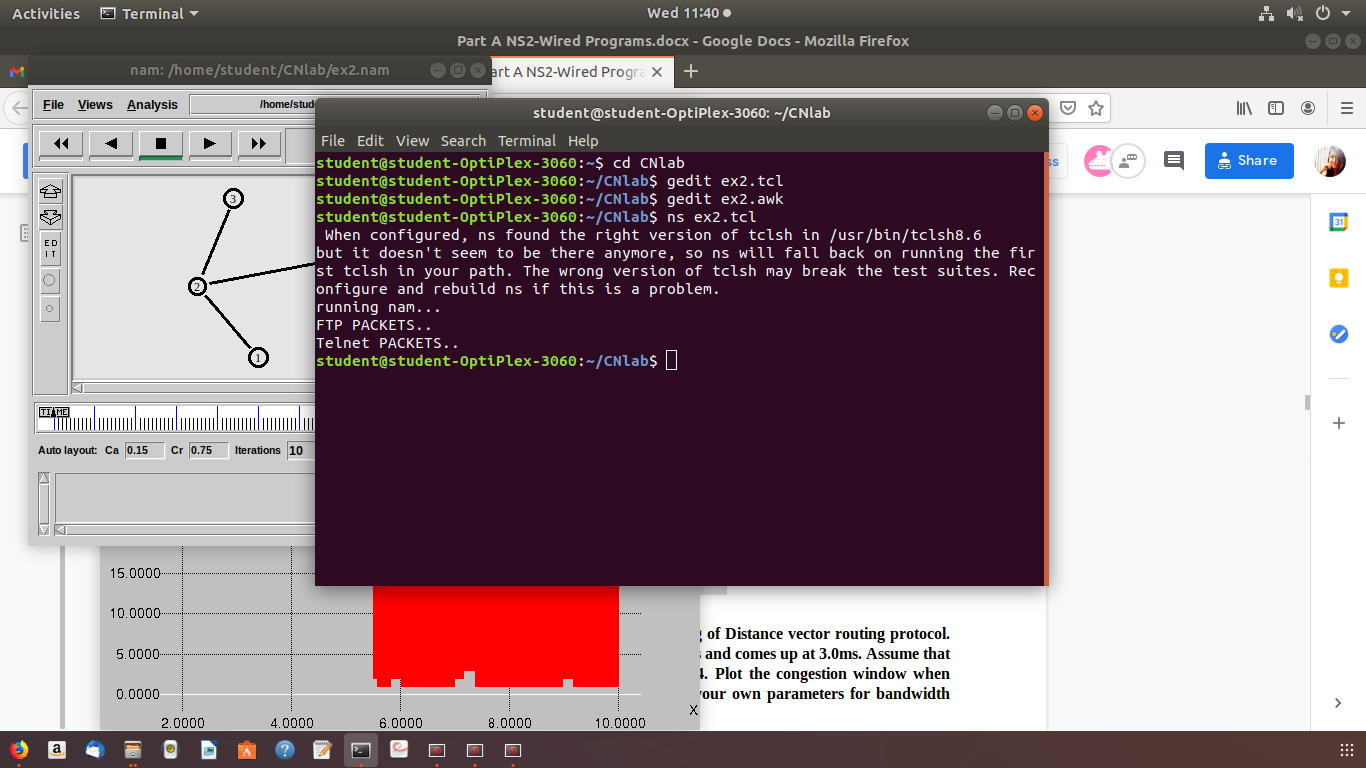
exec xgraph win2.tr &

exit 0

}

$ns run

***Expected output:*** Animated 4 node structure is displayed. We need to see the trace file to understand what has happened to the data flow depending on the application used.



Awk script to calculate throughput

BEGIN {

last = 0

tcp\_sz = 0

cbr\_sz = 0

total\_sz = 0

}

{

action = $1;

time = $2;

from = $3;

to = $4;

type = $5;

pktsize = $6;

flow\_id = $8;

src = $9;

dst = $10;

seq\_no = $11;

packet\_id = $12;

if (type == "tcp" && action == "r" && to == "3" )

tcp\_sz += pktsize

if (type == "cbr" && action == "r" && to == "3" )

cbr\_sz += pktsize

total\_sz += pktsize

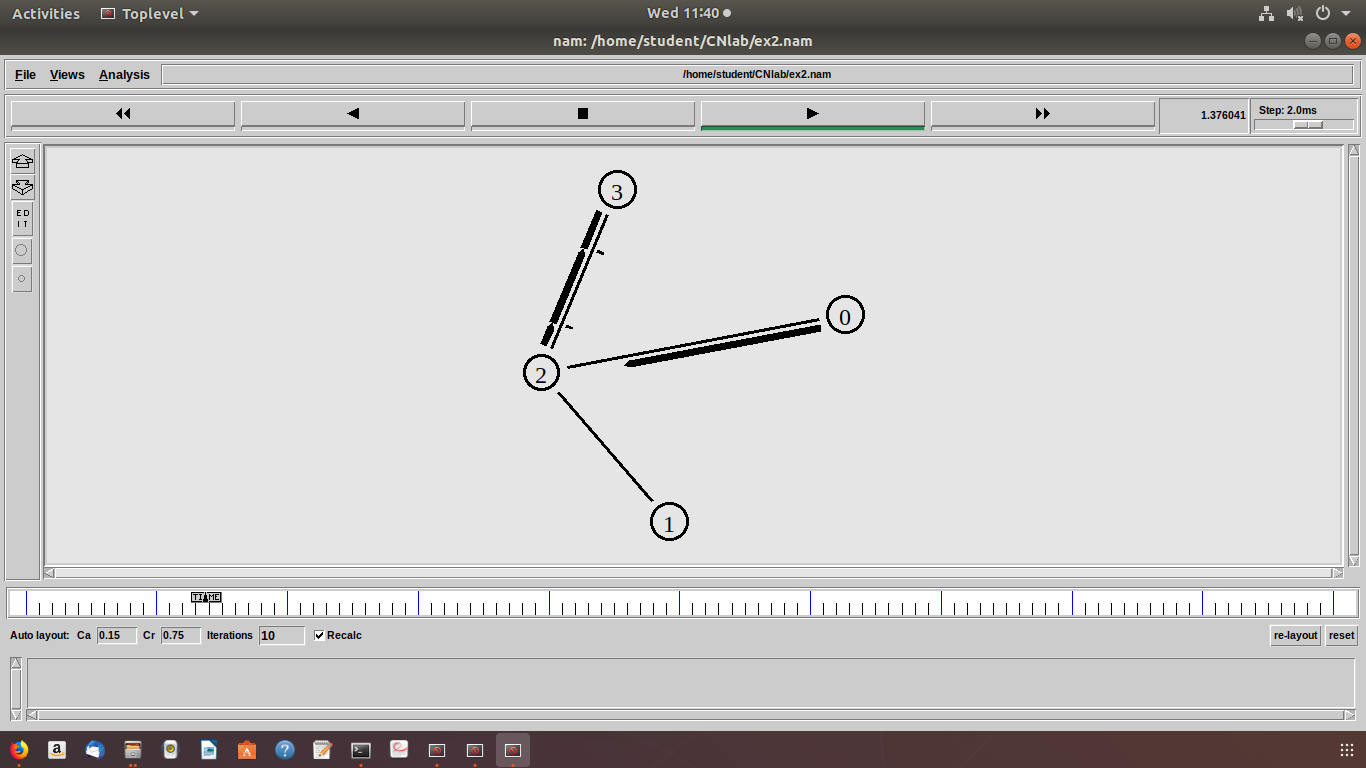
}

END {

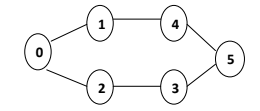
print time, ( tcp\_sz \* 8 / 1000000)

print time , (tcp\_sz \* 8 / 1000000 ), ( total\_sz \* 8 / 1000000)

}



1. **Design networks that demonstrate the working of Distance vector routing protocol. The link between node 1 and 4 breaks at 1.0ms and comes up at 3.0ms. Assume that the source node 0 transmits packets to node 4. Plot the congestion window when TCP sends packets via other nodes. Assume your own parameters for bandwidth and delay.**

****

**Fig 2: Network Topology**

set ns [new Simulator]

set tf [open ex4.tr w]

$ns trace-all $tf

set nf [open ex4.nam w]

$ns namtrace-all $nf

set cwind [open win4.tr w]

$ns color 1 Blue

$ns color 2 Red

$ns rtproto DV

set n0 [$ns node]

set n1 [$ns node]

set n2 [$ns node]

set n3 [$ns node]

set n4 [$ns node]

set n5 [$ns node]

$ns duplex-link $n0 $n1 0.3Mb 10ms DropTail

$ns duplex-link $n1 $n2 0.3Mb 10ms DropTail

$ns duplex-link $n2 $n3 0.3Mb 10ms DropTail

$ns duplex-link $n1 $n4 0.3Mb 10ms DropTail

$ns duplex-link $n3 $n5 0.5Mb 10ms DropTail

$ns duplex-link $n4 $n5 0.5Mb 10ms DropTail

$ns duplex-link-op $n0 $n1 orient right

$ns duplex-link-op $n1 $n2 orient right

$ns duplex-link-op $n2 $n3 orient up

$ns duplex-link-op $n1 $n4 orient up-left

$ns duplex-link-op $n3 $n5 orient up-left

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

set tcp [new Agent/TCP]

$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n5 $sink

$ns connect $tcp $sink

$tcp set fid\_ 1

set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ns rtmodel-at 1.0 down $n1 $n4

$ns rtmodel-at 3.0 up $n1 $n4

$ns at 0.1 "$ftp start"

$ns at 12.0 "finish"

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 1.0 "plotWindow $tcp $cwind"

proc finish {} {

global ns tf nf cwind

$ns flush-trace

close $tf

close $nf

exec nam ex4.nam &

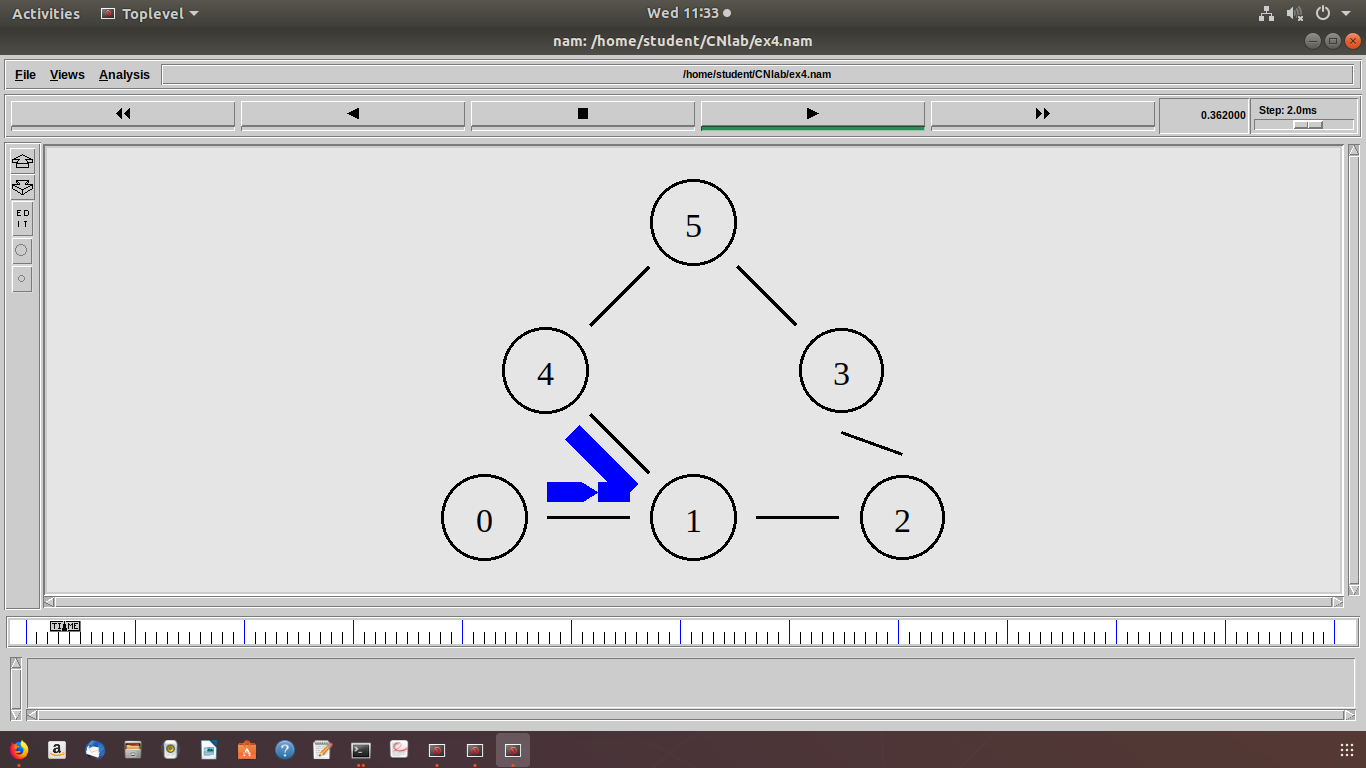
exec xgraph win4.tr &

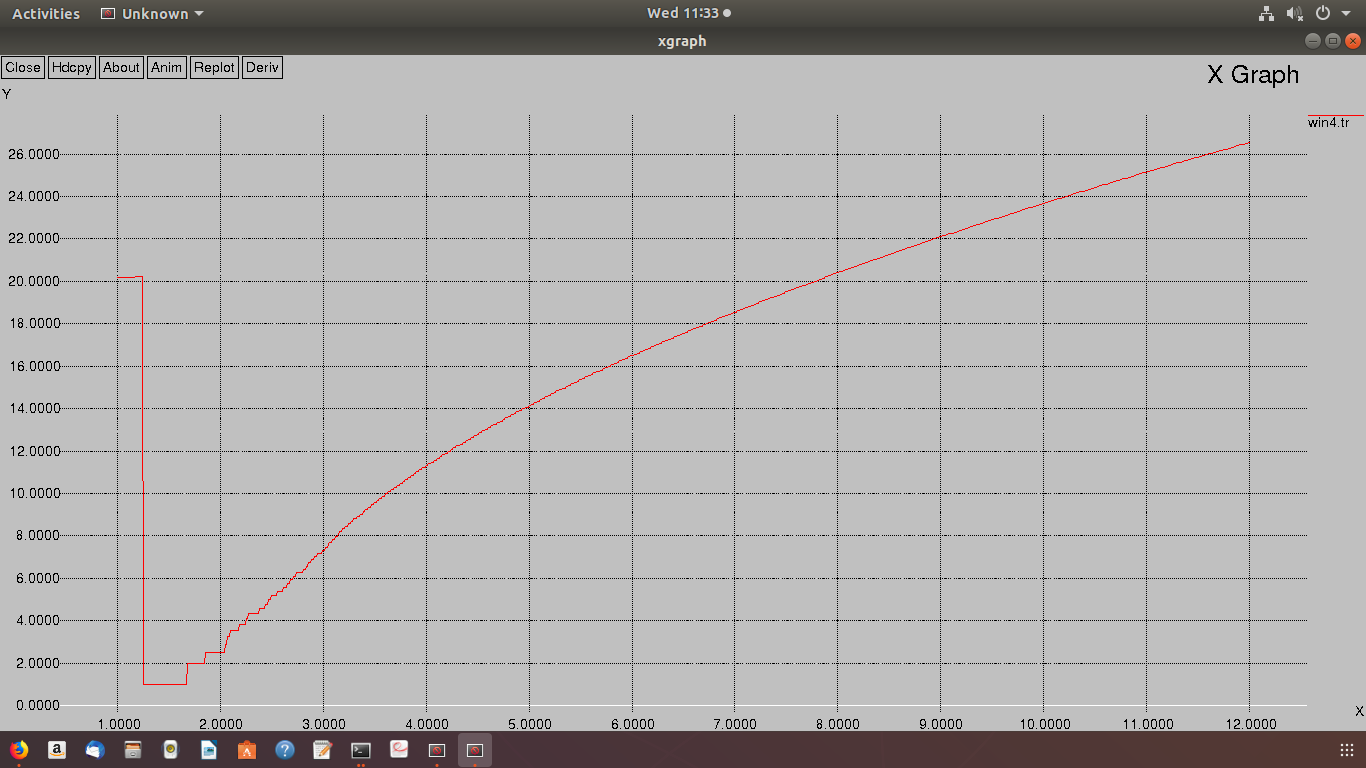
exit 0

}

$ns run

***Expected output:*** Animated 6 node structure is displayed. We need to see the nam file as well as the trace file to understand what has happened to the data flow.





1. **Consider a client and a server. The server is running a FTP application over TCP. The client sends a request to download a file of size 10 MB from the server. Write a TCL script to simulate this scenario. Let node n0 be the server and node n1 be the client. TCP packet size is 1500 Bytes.**

#Create a ns simulator

set ns [new Simulator]

#Open the NS trace file

set tracefile [open ex5.tr w]

$ns trace-all $tracefile

#Open the NAM trace file

set namfile [open ex5.nam w]

$ns namtrace-all $namfile

#Create 2 nodes

set s [$ns node]

set c [$ns node]

$ns color 1 Blue

#Create labels for nodes

$s label "Server"

$c label "Client"

#Create links between nodes

$ns duplex-link $s $c 10Mb 22ms DropTail

#Give node position (for NAM)

$ns duplex-link-op $s $c orient right

#Setup a TCP connection for node s(server)

set tcp0 [new Agent/TCP]

$ns attach-agent $s $tcp0

$tcp0 set packetSize\_ 1500

#Setup a TCPSink connection for node c(client)

set sink0 [new Agent/TCPSink]

$ns attach-agent $c $sink0

$ns connect $tcp0 $sink0

#Setup a FTP Application over TCP connection

set ftp0 [new Application/FTP]

$ftp0 attach-agent $tcp0

$tcp0 set fid\_ 1

proc finish { } {

global ns tracefile namfile

$ns flush-trace

close $tracefile

close $namfile

exec nam ex5.nam &

}

$ns at 0.01 "$ftp0 start"

$ns at 15.0 "$ftp0 stop"

$ns at 15.1 "finish"

$ns run

***Expected output:*** Animated 2 node structure is displayed with the node labeled as client and server. We need to make use of the awk script to calculate the time required to transfer the 10 MB file from the server to client and duration for converting downloaded file into MB.

Save the following awk script as ex5transfer.awk

# AWK script to calulate the time required to transfer the 10 MB file from the server to client

BEGIN {

count=0;

time=0;

total\_bytes\_sent =0;

total\_bytes\_received=0;

}

{

if ( $1 == "r" && $4 == 1 && $5 == "tcp")

total\_bytes\_received += $6;

if($1 == "+" && $3 == 0 && $5 == "tcp")

total\_bytes\_sent += $6;

}

END {

system("clear");

printf("\n Transmission time required to transfer the file is %f",$2);

printf("\n Actual data sent from the server is %f Mbps",(total\_bytes\_sent)/1000000);

printf("\n Data Received by the client is %f Mbps\n",(total\_bytes\_received)/1000000);

}

Save the following awk script as ex5convert.awk

# AWK Script to convert the downloaded file into MB

BEGIN {

count=0;

time=0;

}

{

if ( $1 == "r" && $4 == 1 && $5 == "tcp")

{

count += $6;

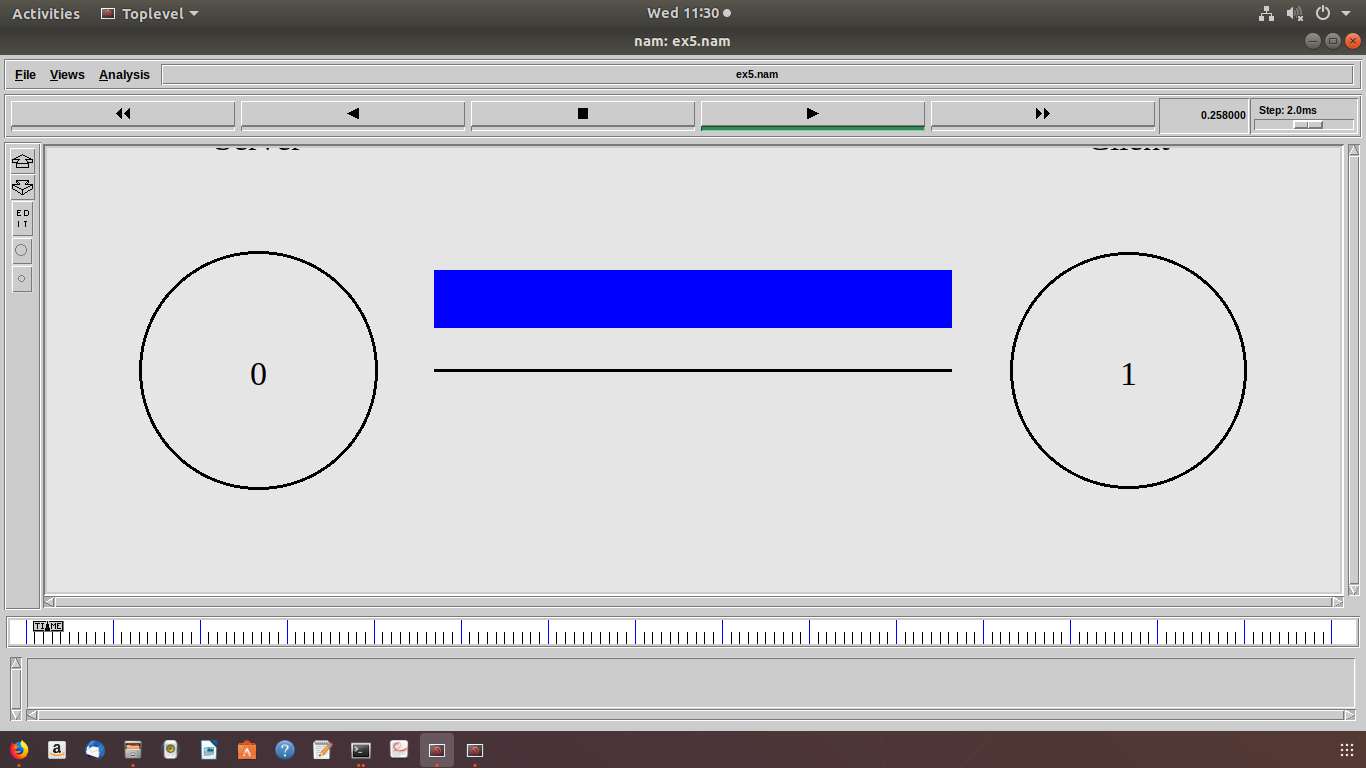
time=$2;

printf("\n%f\t%f",time,(count)/1000000);

}

}

END {

} 

1. **Demonstrate the working of multicast routing protocol. Assume your own parameters for bandwidth and delay.**

#Create an event scheduler wit multicast turned on

set ns [new Simulator -multicast on]

#$ns multicast

#Turn on Tracing

set tf [open mcast.tr w]

$ns trace-all $tf

# Turn on nam Tracing

set fd [open mcast.nam w]

$ns namtrace-all $fd

# 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 n6 [$ns node]

set n7 [$ns node]

# Create links

$ns duplex-link $n0 $n2 1.5Mb 10ms DropTail

$ns duplex-link $n1 $n2 1.5Mb 10ms DropTail

$ns duplex-link $n2 $n3 1.5Mb 10ms DropTail

$ns duplex-link $n3 $n4 1.5Mb 10ms DropTail

$ns duplex-link $n3 $n7 1.5Mb 10ms DropTail

$ns duplex-link $n4 $n5 1.5Mb 10ms DropTail

$ns duplex-link $n4 $n6 1.5Mb 10ms DropTail

# Routing protocol: say distance vector

#Protocols: CtrMcast, DM, ST, BST

set mproto DM

set mrthandle [$ns mrtproto $mproto {}]

# Allocate group addresses

set group1 [Node allocaddr]

set group2 [Node allocaddr]

# UDP Transport agent for the traffic source

set udp0 [new Agent/UDP]

$ns attach-agent $n0 $udp0

$udp0 set dst\_addr\_ $group1

$udp0 set dst\_port\_ 0

set cbr1 [new Application/Traffic/CBR]

$cbr1 attach-agent $udp0

# Transport agent for the traffic source

set udp1 [new Agent/UDP]

$ns attach-agent $n1 $udp1

$udp1 set dst\_addr\_ $group2

$udp1 set dst\_port\_ 0

set cbr2 [new Application/Traffic/CBR]

$cbr2 attach-agent $udp1

# Create receiver

set rcvr1 [new Agent/Null]

$ns attach-agent $n2 $rcvr1

$ns at 1.0 "$n5 join-group $rcvr1 $group1"

set rcvr2 [new Agent/Null]

$ns attach-agent $n3 $rcvr2

$ns at 1.5 "$n6 join-group $rcvr2 $group1"

set rcvr3 [new Agent/Null]

$ns attach-agent $n4 $rcvr3

$ns at 2.0 "$n7 join-group $rcvr3 $group1"

set rcvr4 [new Agent/Null]

$ns attach-agent $n5 $rcvr1

$ns at 2.5 "$n5 join-group $rcvr4 $group2"

set rcvr5 [new Agent/Null]

$ns attach-agent $n6 $rcvr2

$ns at 3.0 "$n6 join-group $rcvr5 $group2"

set rcvr6 [new Agent/Null]

$ns attach-agent $n7 $rcvr3

$ns at 3.5 "$n7 join-group $rcvr6 $group2"

$ns at 4.0 "$n2 leave-group $rcvr1 $group1"

$ns at 4.5 "$n3 leave-group $rcvr2 $group1"

$ns at 5.0 "$n4 leave-group $rcvr3 $group1"

$ns at 5.5 "$n5 leave-group $rcvr4 $group2"

$ns at 6.0 "$n6 leave-group $rcvr5 $group2"

$ns at 6.5 "$n7 leave-group $rcvr6 $group2"

# Schedule events

$ns at 0.5 "$cbr1 start"

$ns at 9.5 "$cbr1 stop"

$ns at 0.5 "$cbr2 start"

$ns at 9.5 "$cbr2 stop"

$ns at 10.0 "finish"

proc finish {} {

global ns tf fd

$ns flush-trace

close $tf

close $fd

exec nam mcast.nam &

exit 0

}

# For nam

# Group 0 source

#$udp0 set fid\_ 1

#$n0 color red

$n0 label "Source 1"

# Group 1 source

#$udp1 set fid\_ 2

#$n1 color green

$n1 label "Source 2"

#Colors for packets from two mcast groups

$ns color 1 red

$ns color 2 green

$n5 label "Receiver 1"

$n5 color blue

$n6 label "Receiver 2"

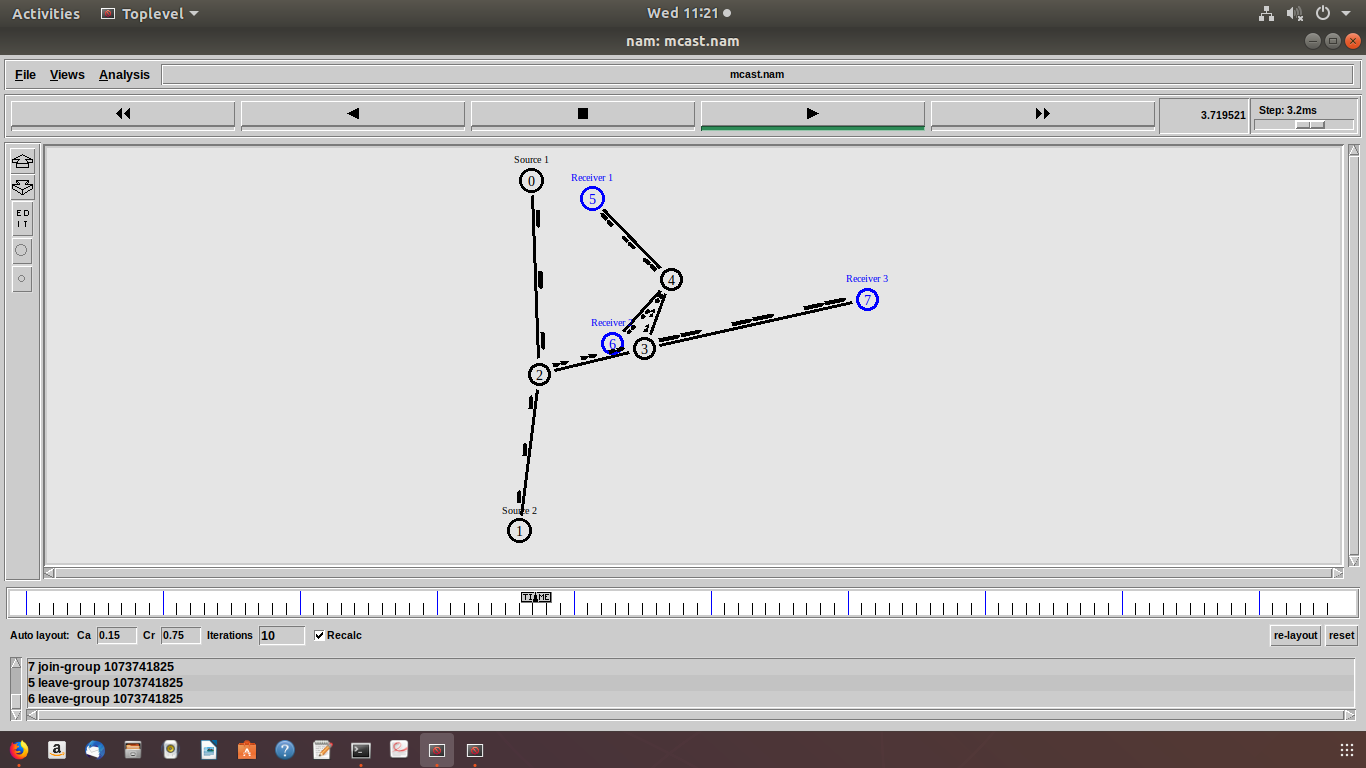
$n6 color blue

$n7 label "Receiver 3"

$n7 color blue

$ns run

***Expected output:*** Animated node structure is displayed with single sender multicasting data to specific receiver group.

****

**Wireless programs:**

1. Set up a 2-node wireless network. Analyze TCP performance for this scenario with DSDV as routing protocol.

**#Setting the Default Parameters**

##################################################################

#     Setting the Default Parameters #

##################################################################

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)       2

set val(stop) 20.0

set val(rp)             DSDV

set ns\_ [new Simulator]

set tracefd [open 001.tr w]

$ns\_ trace-all $tracefd

set namtrace [open 001.nam w]

$ns\_ namtrace-all-wireless $namtrace $val(x) $val(y)

set prop [new $val(prop)]

set topo [new Topography]

$topo load\_flatgrid $val(x) $val(y)

create-god $val(nn)

#Node Configuration

        $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

#Creating Nodes

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

     set node\_($i) [$ns\_ node]

#Initial Positions of Nodes

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

$ns\_ initial\_node\_pos $node\_($i) 40

}

#Topology Design

$ns\_ at 1.1 "$node\_(0) setdest 310.0 10.0 20.0"

$ns\_ at 1.1 "$node\_(1) setdest 10.0 310.0 20.0"

#Generating Traffic

   set tcp0 [new Agent/TCP]

      set sink0 [new Agent/TCPSink]

   $ns\_ attach-agent $node\_(0) $tcp0

       $ns\_ attach-agent $node\_(1) $sink0

   $ns\_ connect $tcp0 $sink0

   set ftp0 [new Application/FTP]

   $ftp0 attach-agent $tcp0

   $ns\_ at 1.0 "$ftp0 start"

       $ns\_ at 18.0 "$ftp0 stop"

#Simulation Termination

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

    $ns\_ at $val(stop) "$node\_($i) reset";

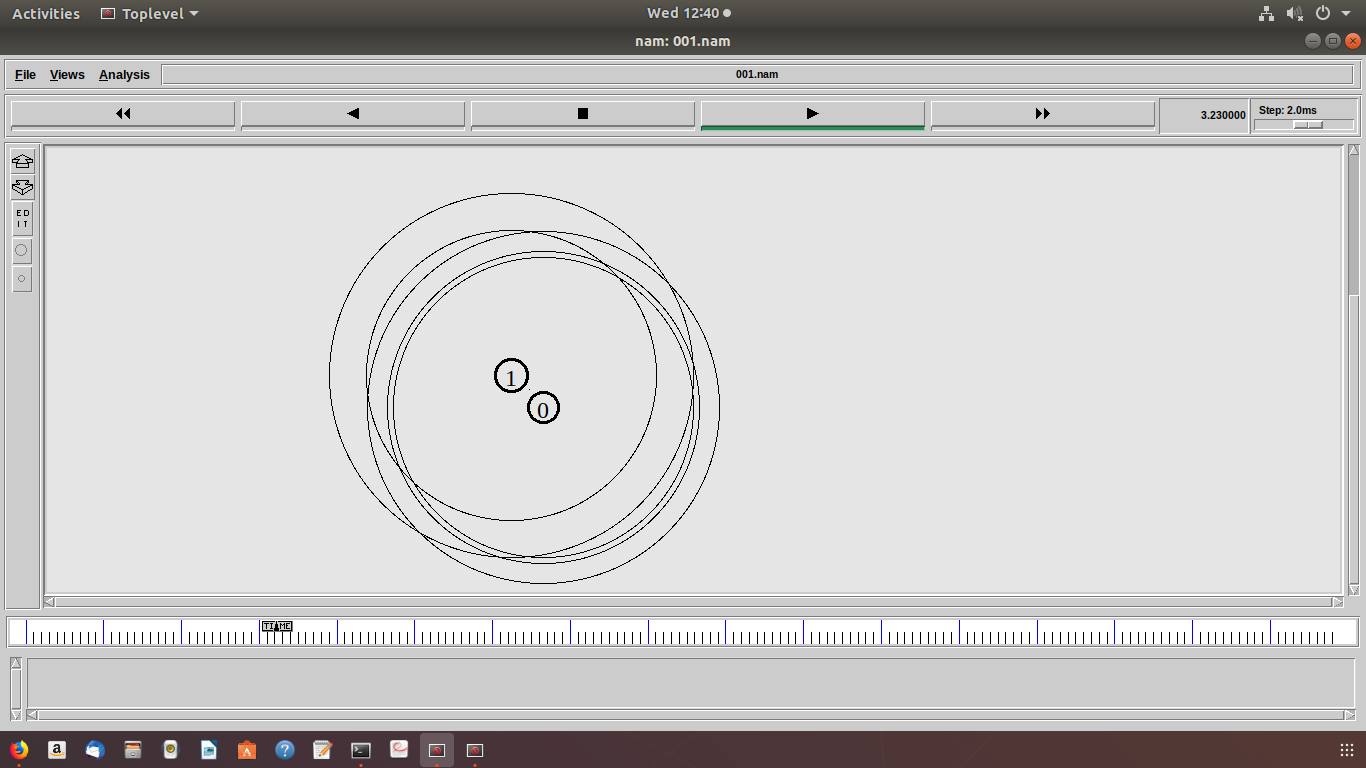
}

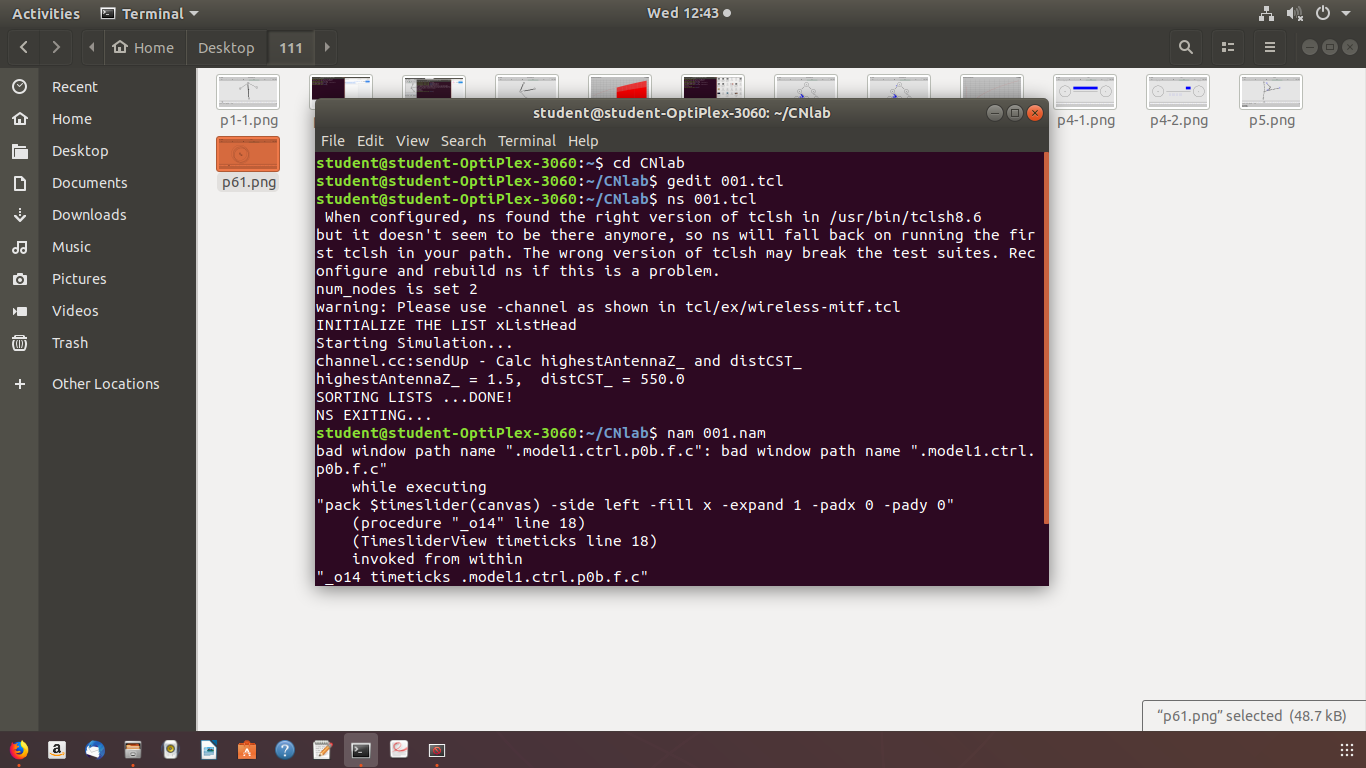
$ns\_ at $val(stop) "puts \"NS EXITING...\" ; $ns\_ halt"

puts "Starting Simulation..."

$ns\_ run

**Expected Output:** Demonstration of wireless network, Analysis of DSDV routing protocol.





1. Set up 3-node wireless network with node N1 between N0 and N2. As the nodes N0 and N2 moves towards each other they exchange packets. As they move out of each other‘s range they drop some packets. Analyze TCP performance for this scenario with AODV routing protocol.

**#Setting the Default Parameters**

set val(chan) Channel/WirelessChannel

set val(prop) Propagation/TwoRayGround

set val(netif) Phy/WirelessPhy

set val(mac)          Mac/802\_11

 #set val(ifq)           CMUPriQueue

set val(ifq)   Queue/DropTail/PriQueue

set val(ll) LL

set val(ant)          Antenna/OmniAntenna

set val(x)       500

set val(y)       400

set val(ifqlen) 50

set val(nn) 3

set val(stop) 60.0

 set val(rp)              AODV

set ns\_ [new Simulator]

set tracefd [open 002.tr w]

$ns\_ trace-all $tracefd

set namtrace [open 002.nam w]

$ns\_ namtrace-all-wireless $namtrace $val(x) $val(y)

set prop [new $val(prop)]

set topo [new Topography]

$topo load\_flatgrid $val(x) $val(y)

create-god $val(nn)

#Node Configuration

        $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

#Creating Nodes

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

 set node\_($i) [$ns\_ node]

     $node\_($i) random-motion 0

     }

#Initial Positions of Nodes

$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

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

$ns\_ initial\_node\_pos $node\_($i) 40

}

#Topology Design

$ns\_ at 0.0 "$node\_(0) setdest 450.0 285.0  30.0"

$ns\_ at 0.0 "$node\_(1) setdest 200.0 285.0 30.0"

$ns\_ at 0.0 "$node\_(2) setdest 1.0 285.0 30.0"

$ns\_ at 25.0 "$node\_(0) setdest 300.0 285.0 10.0"

$ns\_ at 25.0 "$node\_(2) setdest 100.0 285.0 10.0"

$ns\_ at 40.0 "$node\_(0) setdest 490.0 285.0  5.0"

$ns\_ at 40.0 "$node\_(2) setdest 1.0 285.0 5.0"

#Generating Traffic

   set tcp0 [new Agent/TCP]

      set sink0 [new Agent/TCPSink]

   $ns\_ attach-agent $node\_(0) $tcp0

       $ns\_ attach-agent $node\_(2) $sink0

  $ns\_ connect $tcp0 $sink0

   set ftp0 [new Application/FTP]

   $ftp0 attach-agent $tcp0

   $ns\_ at 10.0 "$ftp0 start"

#Simulation Termination

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

    $ns\_ at $val(stop) "$node\_($i) reset";

    }

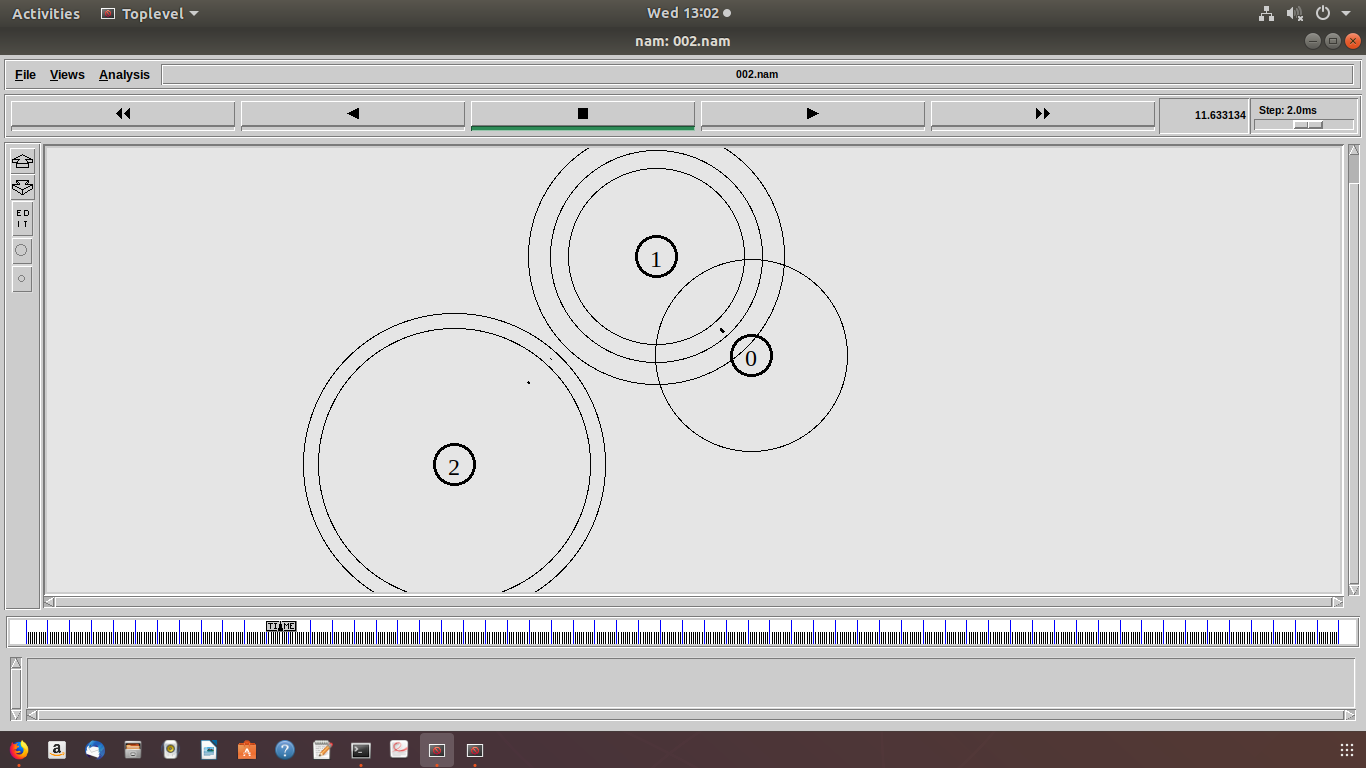
    $ns\_ at $val(stop) "puts \"NS EXITING...\" ; $ns\_ halt"

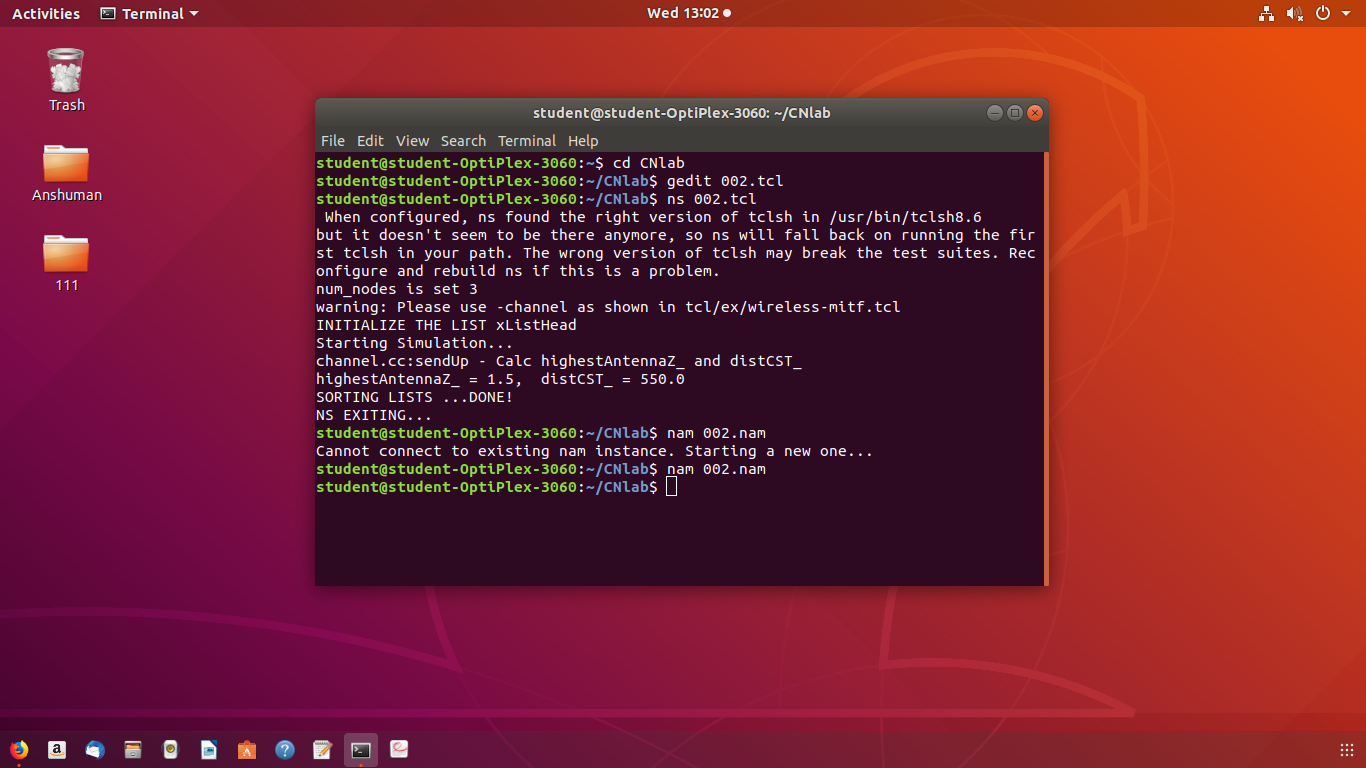
puts "Starting Simulation..."

     $ns\_ run

Repeat the simulation for AODV and DSR Routing protocols.

**Output**: Demonstration of wireless network, Analysis of routing protocols.





1. Set up a 6-node wireless network; analyze TCP performance when nodes are static and mobile.

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) 25

set val(stop) 100.0

     set val(rp)            AODV

 #set val(sc)               "mob-25-50"

set val(cp)               "tcp-25-8"

set ns\_ [new Simulator]

set tracefd [open 003.tr w]

$ns\_ trace-all $tracefd

set namtrace [open 003.nam w]

$ns\_ namtrace-all-wireless $namtrace $val(x) $val(y)

set prop [new $val(prop)]

set topo [new Topography]

$topo load\_flatgrid $val(x) $val(y)

set god\_ [create-god $val(nn)]

     #Node Configuration

        $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

#Creating Nodes

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 } {

                  set xx [expr rand()\*500]

                  set yy [expr rand()\*400]

  $node\_($i) set X\_ $xx

                  $node\_($i) set Y\_ $yy

            }

#Initial Positions of Nodes

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

$ns\_ initial\_node\_pos $node\_($i) 40

}

#puts "Loading scenario file..."

#source $val(sc)

puts "Loading connection file..."

source $val(cp)

#Simulation Termination

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

    $ns\_ at $val(stop) "$node\_($i) reset";

    }

    $ns\_ at $val(stop) "puts \"NS EXITING...\" ; $ns\_ halt"

puts "Starting Simulation..."

     $ns\_ run

**Expected Output:** Setup of wireless network and Performance analysis of static and mobile nodes.

1. Write a TCL script to simulate the following scenario. Consider six nodes, (as shown in the figure below) moving within a flat topology of 700m x 700m. The initial positions of nodes are: n0 (150, 300), n1 (300, 500), n2(500, 500), n3 (300, 100), n4(500, 100) and n5(650, 300) respectively. A TCP connection is initiated between n0 (source) and n5 (destination) through n3 and n4 i.e., the route is 0- 3-4-5. At time t = 3 seconds, the FTP application runs over it. After time t = 4 seconds, n3 (300,100) moves towards n1 (300, 500) with a speed of 5.0m/sec and after some time the path breaks. The data is then transmitted with a new path via n1 and n2 i.e., the new route is 0-1-2-5. The simulation lasts for 60 secs. In the above said case both the routes have equal cost. Use DSR as the routing protocol And the IEEE 802.11 MAC protocol.

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(ifq)           CMUPriQueue

set val(ll) LL

set val(ant)          Antenna/OmniAntenna

set val(x)       700

set val(y)       700

set val(ifqlen) 50

set val(nn) 6

set val(stop) 60.0

     set val(rp)            DSR

set ns\_ [new Simulator]

set tracefd [open 004.tr w]

$ns\_ trace-all $tracefd

set namtrace [open 004.nam w]

$ns\_ namtrace-all-wireless $namtrace $val(x) $val(y)

set prop [new $val(prop)]

set topo [new Topography]

$topo load\_flatgrid $val(x) $val(y)

set god\_ [create-god $val(nn)]

     #Node Configuration

        $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

#Creating Nodes

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

     set node\_($i) [$ns\_ node]

     $node\_($i) random-motion 0

}

#Initial Positions of Nodes

$node\_(0) set X\_ 150.0

$node\_(0) set Y\_ 300.0

$node\_(0) set Z\_ 0.0

$node\_(1) set X\_ 300.0

$node\_(1) set Y\_ 500.0

$node\_(1) set Z\_ 0.0

$node\_(2) set X\_ 500.0

$node\_(2) set Y\_ 500.0

$node\_(2) set Z\_ 0.0

$node\_(3) set X\_ 300.0

$node\_(3) set Y\_ 100.0

$node\_(3) set Z\_ 0.0

$node\_(4) set X\_ 500.0

$node\_(4) set Y\_ 100.0

$node\_(4) set Z\_ 0.0

$node\_(5) set X\_ 650.0

$node\_(5) set Y\_ 300.0

$node\_(5) set Z\_ 0.0

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

$ns\_ initial\_node\_pos $node\_($i) 40

}

#Topology Design

$ns\_ at 1.0 "$node\_(0) setdest 160.0 300.0 2.0"

$ns\_ at 1.0 "$node\_(1) setdest 310.0 150.0 2.0"

$ns\_ at 1.0 "$node\_(2) setdest 490.0 490.0 2.0"

$ns\_ at 1.0 "$node\_(3) setdest 300.0 120.0 2.0"

$ns\_ at 1.0 "$node\_(4) setdest 510.0 90.0 2.0"

$ns\_ at 1.0 "$node\_(5) setdest 640.0 290.0 2.0"

$ns\_ at 4.0 "$node\_(3) setdest 300.0 500.0 5.0"

#Generating Traffic

   set tcp0 [new Agent/TCP]

      set sink0 [new Agent/TCPSink]

   $ns\_ attach-agent $node\_(0) $tcp0

       $ns\_ attach-agent $node\_(5) $sink0

   $ns\_ connect $tcp0 $sink0

   set ftp0 [new Application/FTP]

   $ftp0 attach-agent $tcp0

   $ns\_ at 5.0 "$ftp0 start"

       $ns\_ at 60.0 "$ftp0 stop"

#Simulation Termination

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

    $ns\_ at $val(stop) "$node\_($i) reset";

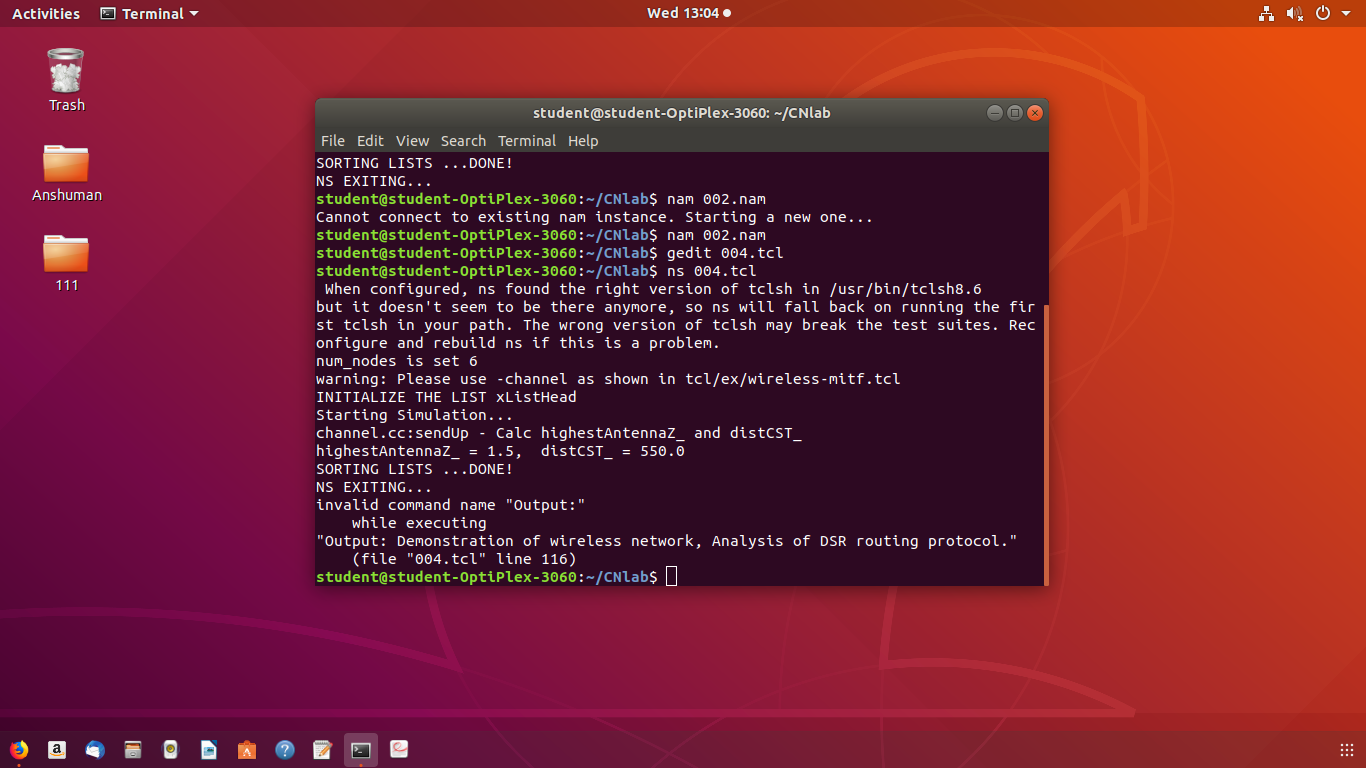
    }

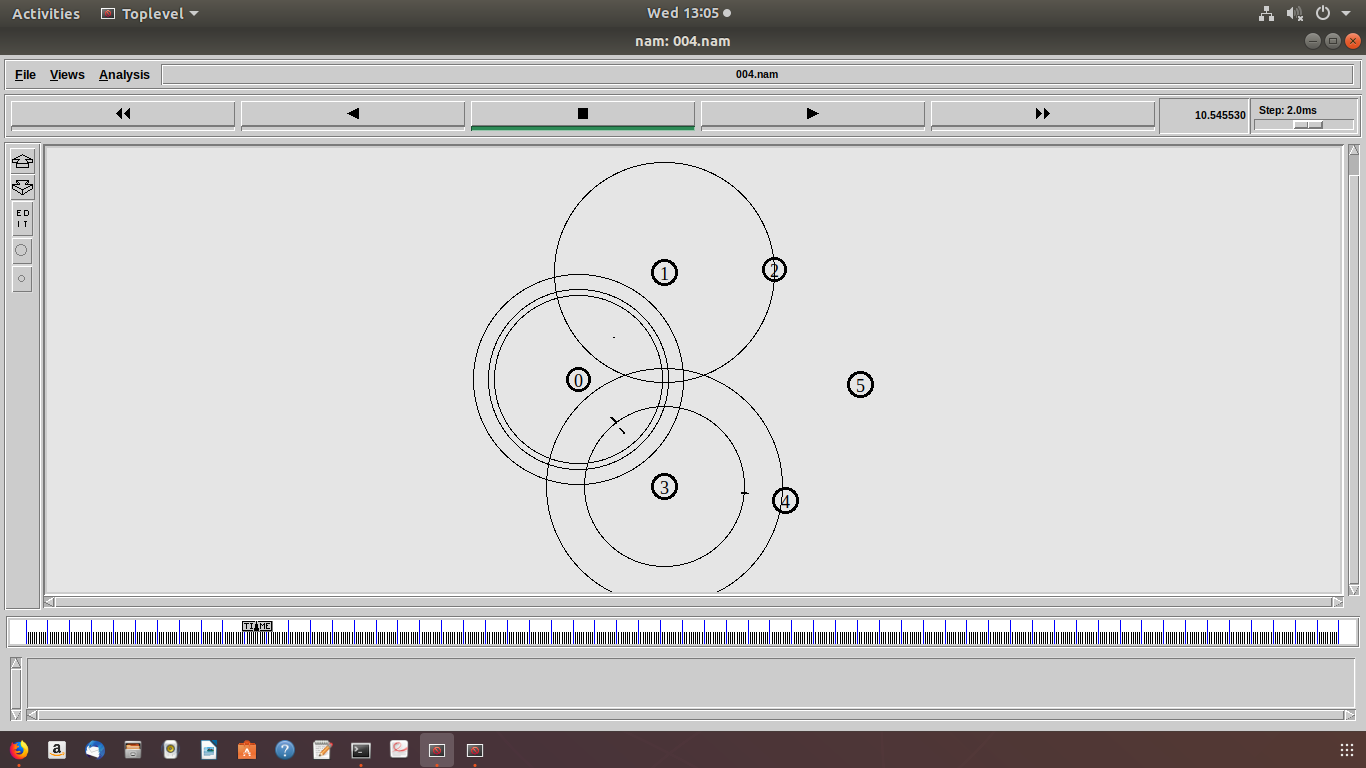
    $ns\_ at $val(stop) "puts \"NS EXITING...\" ; $ns\_ halt"

puts "Starting Simulation..."

     $ns\_ run

**Output:** Demonstration of wireless network, Analysis of DSR routing protocol.





1. Set up a wireless network with mobile nodes, induce 1 to 10% error to the network using uniform error model. Plot the congestion window for TCP connections. Write your observation on TCP performance as error increases in the network.

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) 5

set val(stop) 50.0

     set val(rp)            AODV

set ns\_ [new Simulator]

set tracefd [open 006.tr w]

$ns\_ trace-all $tracefd

set namtrace [open 006.nam w]

$ns\_ namtrace-all-wireless $namtrace $val(x) $val(y)

set prop [new $val(prop)]

set topo [new Topography]

$topo load\_flatgrid $val(x) $val(y)

create-god $val(nn)

     #Node Configuration

        $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 \

                        -IncomingErrProc "uniformErr" \

-OutgoingErrProc "uniformErr"

proc uniformErr {} {

set err [new ErrorModel]

$err unit pkt

$err set rate\_ 0.01

return $err

}

#Creating Nodes

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

     set node\_($i) [$ns\_ node]

     $node\_($i) random-motion 0

}

#Initial Positions of Nodes

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

$ns\_ initial\_node\_pos $node\_($i) 40

}

#Topology Design

$ns\_ at 1.0 "$node\_(0) setdest 10.0 10.0 50.0"

$ns\_ at 1.0 "$node\_(1) setdest 10.0 100.0 50.0"

$ns\_ at 1.0 "$node\_(4) setdest 50.0 50.0 50.0"

$ns\_ at 1.0 "$node\_(2) setdest 100.0 100.0 50.0"

$ns\_ at 1.0 "$node\_(3) setdest 100.0 10.0 50.0"

#Generating Traffic

   set tcp0 [new Agent/TCP]

      set sink0 [new Agent/TCPSink]

   $ns\_ attach-agent $node\_(0) $tcp0

       $ns\_ attach-agent $node\_(2) $sink0

   $ns\_ connect $tcp0 $sink0

   set ftp0 [new Application/FTP]

   $ftp0 attach-agent $tcp0

   $ns\_ at 1.0 "$ftp0 start"

       $ns\_ at 50.0 "$ftp0 stop"

         set tcp1 [new Agent/TCP]

      set sink1 [new Agent/TCPSink]

   $ns\_ attach-agent $node\_(1) $tcp1

       $ns\_ attach-agent $node\_(2) $sink1

   $ns\_ connect $tcp1 $sink1

   set ftp1 [new Application/FTP]

   $ftp1 attach-agent $tcp1

   $ns\_ at 1.0 "$ftp1 start"

       $ns\_ at 50.0 "$ftp1 stop"

#Simulation Termination

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

    $ns\_ at $val(stop) "$node\_($i) reset";

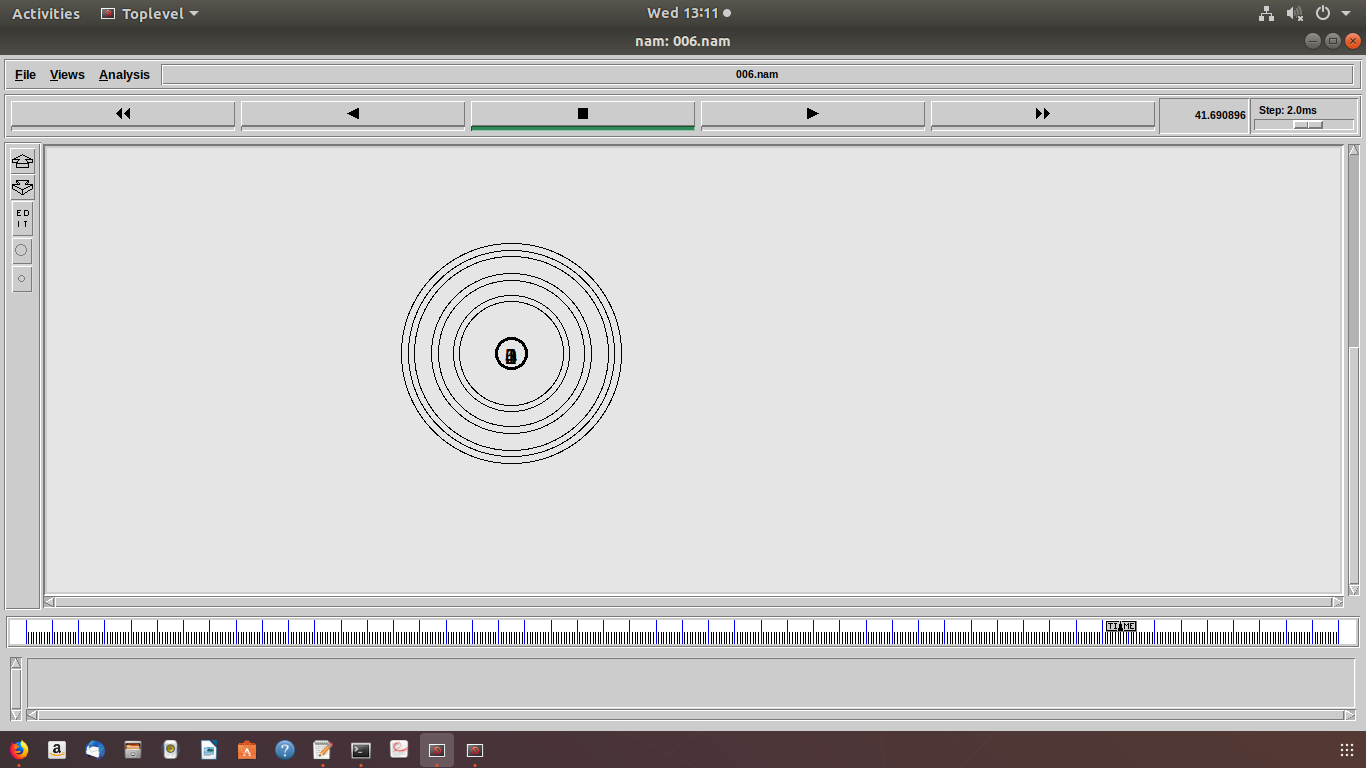
    }

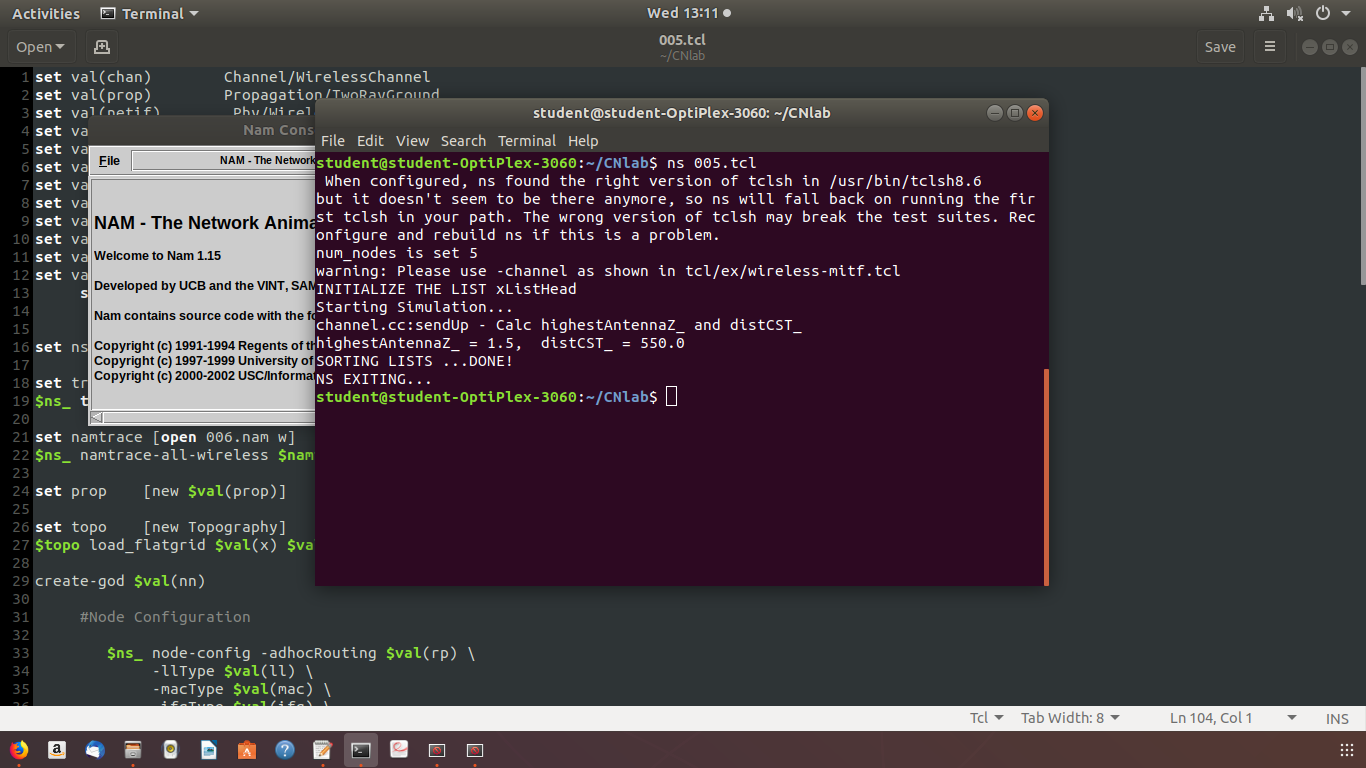
    $ns\_ at $val(stop) "puts \"NS EXITING...\" ; $ns\_ halt"

puts "Starting Simulation..."

     $ns\_ run

***Expected output:*** Animated nodes structure is displayed. We need to see the performance of the network with a varying error rate.

****

****