**COMPUTER NETWORKS ASSIGNMENT - NS2**

Name- Ashutosh Soni

Id- 2018ucp1505

Ques 1 :

Ans:

.**tcl script:-**

#create a ns simulator

set ns [new Simulator]

#open the ns trace file and nam file

set nf [open out1.nam w]

$ns namtrace-all $nf

set nr [open out1.tr w]

$ns trace-all $nr

set f0 [open x1.tr w]

proc record {} {

global f0 sink3

set ns [Simulator instance]

set time 0.01

set bw0 [$sink3 set bytes\_]

set now [$ns now]

puts $f0 “$now $bw0”

$sink3 set bytes\_ 0

$ns at [expr $now+$time] “record”

}

#NODES DEFINATION

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 n8 [$ns node]

set n9 [$ns node]

set n10 [$ns node]

set n11 [$ns node]

set n12 [$ns node]

set n13 [$ns node]

#LINKS DEFINATION

$ns duplex-link $n0 $n2 2Mb 10ms RED

$ns duplex-link $n1 $n2 1Mb 10ms FQ

$ns duplex-link $n2 $n3 2Mb 10ms RED

$ns duplex-link $n2 $n4 1Mb 10ms FQ

$ns duplex-link $n3 $n4 1Mb 10ms FQ

$ns duplex-link $n3 $n5 2Mb 10ms RED

$ns duplex-link $n3 $n6 2Mb 10ms RED

$ns duplex-link $n4 $n11 1Mb 10ms FQ

$ns duplex-link $n4 $n5 2Mb 10ms RED

$ns duplex-link $n6 $n5 2Mb 10ms RED

$ns duplex-link $n6 $n7 2Mb 10ms RED

$ns duplex-link $n7 $n8 2Mb 10ms RED

$ns duplex-link $n7 $n9 2Mb 10ms RED

$ns duplex-link $n7 $n10 2Mb 10ms RED

$ns duplex-link $n11 $n12 2Mb 10ms RED

$ns duplex-link $n11 $n13 2Mb 10ms RED

#NODE POSITION

$ns duplex-link-op $n0 $n2 orient right-down

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

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

$ns duplex-link-op $n2 $n4 orient right-down

$ns duplex-link-op $n3 $n4 orient right-up

$ns duplex-link-op $n3 $n5 orient right-down

$ns duplex-link-op $n3 $n6 orient down

$ns duplex-link-op $n4 $n11 orient right-up

$ns duplex-link-op $n4 $n5 orient down

$ns duplex-link-op $n6 $n5 orient right-up

$ns duplex-link-op $n6 $n7 orient down

$ns duplex-link-op $n7 $n8 orient left-down

$ns duplex-link-op $n7 $n9 orient right-down

$ns duplex-link-op $n7 $n10 orient down

$ns duplex-link-op $n11 $n12 orient right-up

$ns duplex-link-op $n11 $n13 orient right-down

#Agent Definition

set tcp0 [new Agent/TCP]

$ns attach-agent $n0 $tcp0

set ftp0 [new Application/FTP]

$ftp0 attach-agent $tcp0

set sink0 [new Agent/TCPSink]

$ns attach-agent $n8 $sink0

$ns connect $tcp0 $sink0

set tcp1 [new Agent/TCP]

$ns attach-agent $n0 $tcp1

set ftp1 [new Application/FTP]

$ftp1 attach-agent $tcp1

set sink1 [new Agent/TCPSink]

$ns attach-agent $n9 $sink1

$ns connect $tcp1 $sink1

set tcp2 [new Agent/TCP]

$ns attach-agent $n0 $tcp2

set ftp2 [new Application/FTP]

$ftp2 attach-agent $tcp2

set sink2 [new Agent/TCPSink]

$ns attach-agent $n10 $sink2

$ns connect $tcp2 $sink2

set tcp3 [new Agent/TCP]

$ns attach-agent $n0 $tcp3

set ftp3 [new Application/FTP]

$ftp3 attach-agent $tcp0

set sink3 [new Agent/TCPSink]

$ns attach-agent $n13 $sink3

$ns connect $tcp3 $sink3

set tcp4 [new Agent/TCP]

$ns attach-agent $n1 $tcp4

set ftp4 [new Application/FTP]

$ftp4 attach-agent $tcp4

set sink4 [new Agent/TCPSink]

$ns attach-agent $n8 $sink4

$ns connect $tcp4 $sink4

set tcp5 [new Agent/TCP]

$ns attach-agent $n1 $tcp5

set ftp5 [new Application/FTP]

$ftp5 attach-agent $tcp5

set sink5 [new Agent/TCPSink]

$ns attach-agent $n9 $sink5

$ns connect $tcp5 $sink5

set tcp6 [new Agent/TCP]

$ns attach-agent $n1 $tcp6

set ftp6 [new Application/FTP]

$ftp6 attach-agent $tcp6

set sink6 [new Agent/TCPSink]

$ns attach-agent $n10 $sink6

$ns connect $tcp6 $sink6

set tcp7 [new Agent/TCP]

$ns attach-agent $n1 $tcp7

set ftp7 [new Application/FTP]

$ftp5 attach-agent $tcp7

set sink7 [new Agent/TCPSink]

$ns attach-agent $n13 $sink7

$ns connect $tcp7 $sink7

$ns at 0.1 “$ftp0 start”

$ns at 3.0 “$ftp0 stop”

$ns at 0.3 “$ftp1 start”

$ns at 3.0 “$ftp1 stop”

$ns at 0.5 “$ftp2 start”

$ns at 3.0 “$ftp2 stop”

$ns at 0.8 “$ftp3 start”

$ns at 3.0 “$ftp3 stop”

$ns rtproto DV

$ns rtmodel-at 0.1 down $n2 $n3

$ns rtmodel-at 0.15 down $n6 $n7

$ns rtmodel-at 0.20 up $n2 $n3

$ns rtmodel-at 0.30 up $n6 $n7

$ns at 0.01 “record”

$ns at 1.2 “$ftp4 start”

$ns at 4.0 “$ftp4 stop”

$ns at 1.5 “$ftp5 start”

$ns at 4.0 “$ftp5 stop”

$ns at 1.7 “$ftp6 start”

$ns at 4.0 “$ftp6 stop”

$ns at 5.0 “finish”

$ns run

# Termination

proc finish {} {

global ns nr nf

global f0

$ns flush-trace

close $nf

close $nr

close $f0

exec nam out1.nam &

exec xgraph x1.tr -geometry 800\*400 -x bandwidth -y time &

exit 0

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*AWK File\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**BEGIN {**

**rec=0;**

**rec1=0;**

**rec2=0;**

**send=0;**

**pr=0;**

**pr1=0;**

**pr2=0;**

**}**

**{**

**if($1==”+” && $5==”tcp” && $3==0 || $1==”+” && $5==”tcp” && $3==1)**

**send++;**

**if($1==”r” && $5==”tcp” && $4==8)**

**rec1++;**

**if($1==”r” && $5==”tcp” && $4==9)**

**rec++;**

**if($1==”r” && $5==”tcp” && $4==10)**

**rec2++;**

**}**

**END {**

**pr=rec/send;**

**pr1=rec1/send;**

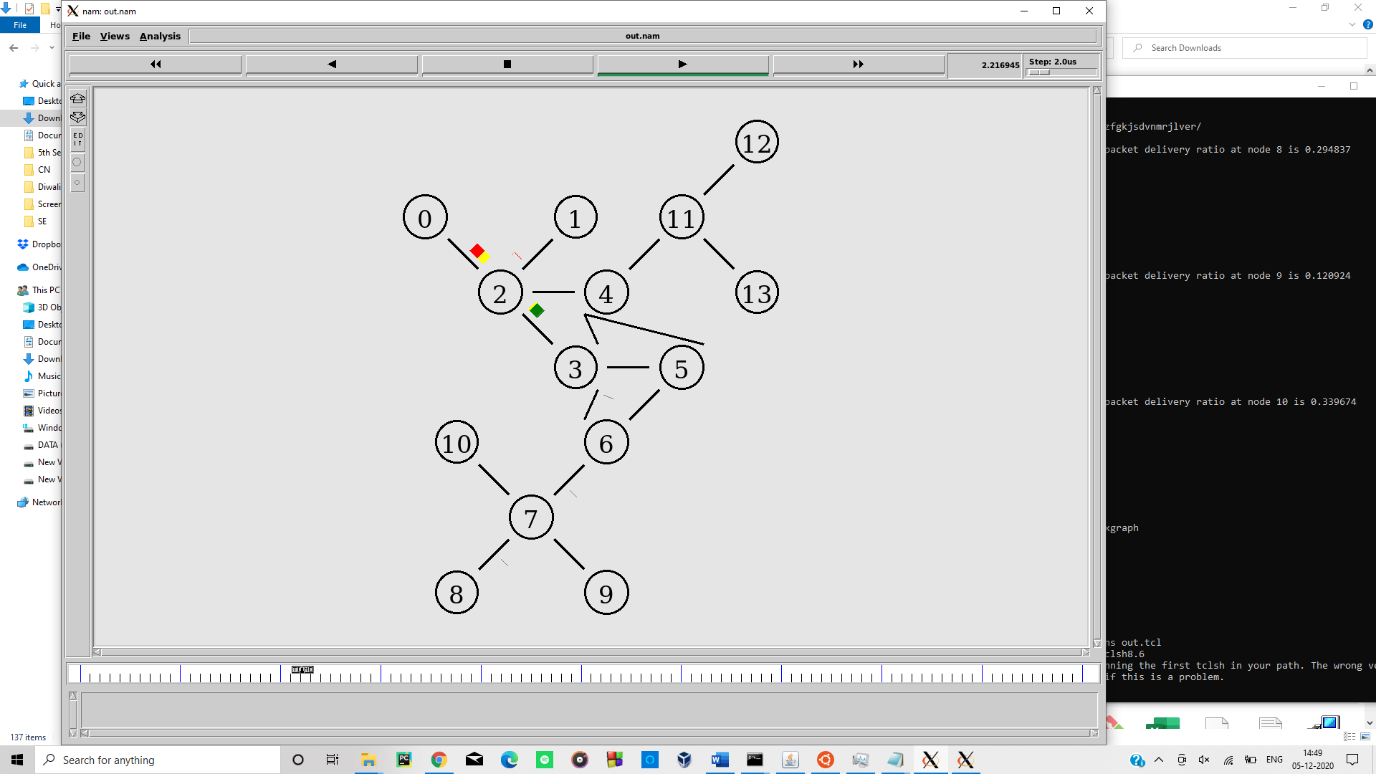
**pr2=rec2/send;**

**printf(“packet delivery ratio at node 8 is %f\n”,pr);**

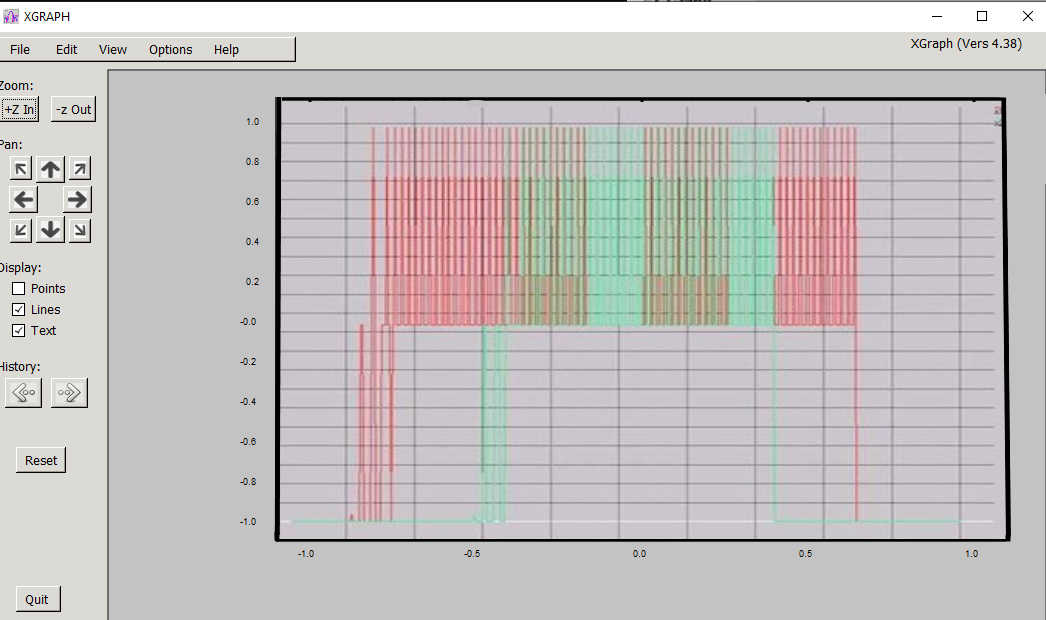
**printf(“packet delivery ratio at node 9 is %f\n”,pr1);**

**printf(“packet delivery ratio at node 10 is %f\n”,pr2);**

**}**

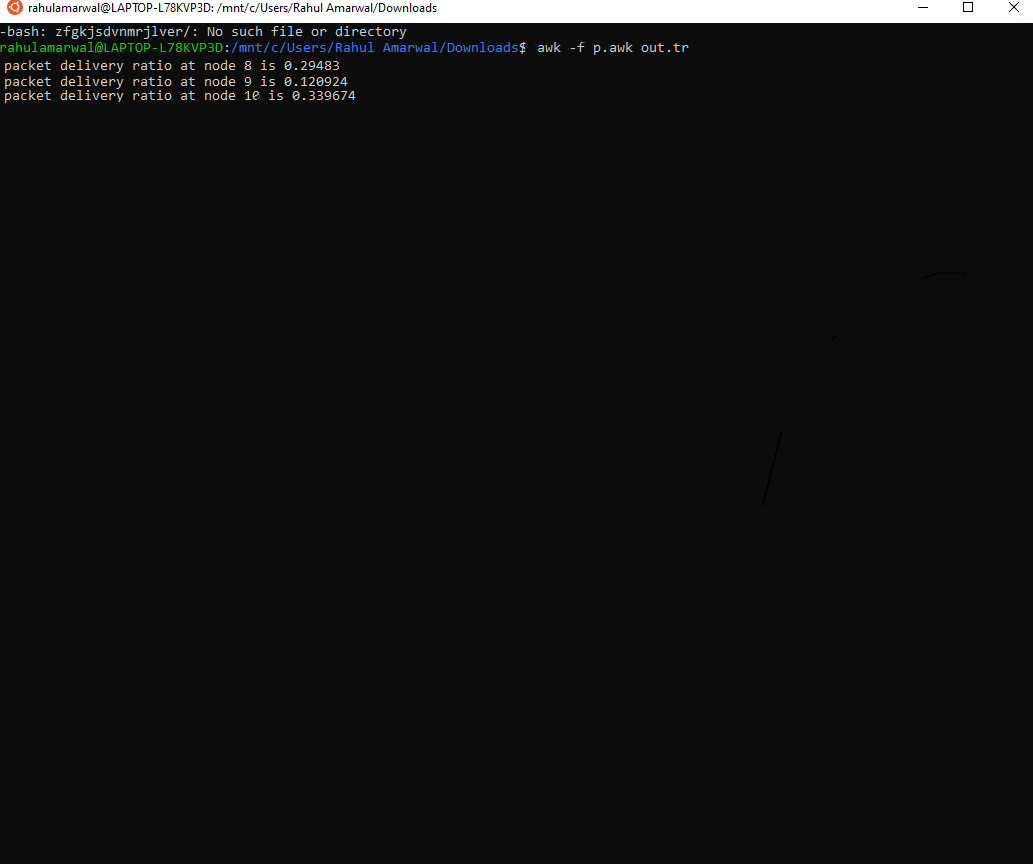


Bandwidth V/S Time Graph



The packet delivery ratio for 8 9 and 10:

* The packet delivery ratio of node 8 is 0.29483
* The packet delivery ratio of node 9 is 0.120924
* The packet delivery ratio of node 10 is 0.339674



Ques 2: -

Ans:

.**tcl script:-**

# Set Channels

set opt(chan) Channel/WirelessChannel ;# channel type

set opt(prop) Propagation/TwoRayGround ;# radio-propagation model

set opt(netif) Phy/WirelessPhy

set opt(mac) Mac/802\_11

set opt(ifq) Queue/DropTail/PriQueue

set opt(ll) LL

set opt(ant) Antenna/OmniAntenna

set opt(ifqlen) 50

set opt(nn) 1

set opt(adhocRouting) DSDV

set opt(cp) ""

set opt(sc) ""

set opt(x) 670

set opt(y) 670

set opt(seed) 0.0

set opt(stop) 250

set opt(ftp1-start) 100.0

set num\_wired\_nodes 2

#set num\_bs\_nodes 2 ; this is not really used here.

# check for boundary parameters and random seed

if { $opt(x) == 0 || $opt(y) == 0 } {

puts "No X-Y boundary values given for wireless topology\n"

}

if {$opt(seed) > 0} {

puts "Seeding Random number generator with $opt(seed)\n"

ns-random $opt(seed)

}

# create simulator instance

set ns\_ [new Simulator]

# set up for hierarchical routing

$ns\_ node-config -addressType hierarchical

AddrParams set domain\_num\_ 3 ;# number of domains

lappend cluster\_num 2 1 1 ;# number of clusters in each domain

AddrParams set cluster\_num\_ $cluster\_num

lappend eilastlevel 1 1 2 1 ;# number of nodes in each cluster

AddrParams set nodes\_num\_ $eilastlevel ;# of each domain

set tracefd [open wireless3-out.tr w]

set namtrace [open wireless3-out.nam w]

$ns\_ trace-all $tracefd

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

# Create topography object

set topo [new Topography]

# define topology

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

# create God

# 2 for HA and FA

create-god [expr $opt(nn) + 2]

#create wired nodes

set temp {0.0.0 0.1.0} ;# hierarchical addresses

for {set i 0} {$i < $num\_wired\_nodes} {incr i} {

set W($i) [$ns\_ node [lindex $temp $i]]

}

# Configure for ForeignAgent and HomeAgent nodes

$ns\_ node-config -mobileIP ON \

-adhocRouting $opt(adhocRouting) \

-llType $opt(ll) \

-macType $opt(mac) \

-ifqType $opt(ifq) \

-ifqLen $opt(ifqlen) \

-antType $opt(ant) \

-propType $opt(prop) \

-phyType $opt(netif) \

-channelType $opt(chan) \

-topoInstance $topo \

-wiredRouting ON \

-agentTrace ON \

-routerTrace OFF \

-macTrace OFF

# Create HA and FA

set HA [$ns\_ node 1.0.0]

set FA [$ns\_ node 2.0.0]

$HA random-motion 0

$FA random-motion 0

# Position (fixed) for base-station nodes (HA & FA).

$HA set X\_ 1.000000000000

$HA set Y\_ 2.000000000000

$HA set Z\_ 0.000000000000

$FA set X\_ 650.000000000000

$FA set Y\_ 600.000000000000

$FA set Z\_ 0.000000000000

# create a mobilenode that would be moving between HA and FA.

# note address of MH indicates its in the same domain as HA.

$ns\_ node-config -wiredRouting OFF

set MH [$ns\_ node 1.0.1]

set node\_(0) $MH

set HAaddress [AddrParams addr2id [$HA node-addr]]

[$MH set regagent\_] set home\_agent\_ $HAaddress

# movement of the MH

$MH set Z\_ 0.000000000000

$MH set Y\_ 2.000000000000

$MH set X\_ 2.000000000000

# MH starts to move towards FA

$ns\_ at 100.000000000000 "$MH setdest 640.000000000000 610.000000000000 20.000000000000"

# goes back to HA

$ns\_ at 200.000000000000 "$MH setdest 2.000000000000 2.000000000000 20.000000000000"

# create links between wired and BaseStation nodes

$ns\_ duplex-link $W(0) $W(1) 5Mb 2ms DropTail

$ns\_ duplex-link $W(1) $HA 5Mb 2ms DropTail

$ns\_ duplex-link $W(1) $FA 5Mb 2ms DropTail

$ns\_ duplex-link-op $W(0) $W(1) orient down

$ns\_ duplex-link-op $W(1) $HA orient left-down

$ns\_ duplex-link-op $W(1) $FA orient right-down

# setup TCP connections between a wired node and the MobileHost

set tcp1 [new Agent/TCP]

$tcp1 set class\_ 2

set sink1 [new Agent/TCPSink]

$ns\_ attach-agent $W(0) $tcp1

$ns\_ attach-agent $MH $sink1

$ns\_ connect $tcp1 $sink1

set ftp1 [new Application/FTP]

$ftp1 attach-agent $tcp1

$ns\_ at $opt(ftp1-start) "$ftp1 start"

# source connection-pattern and node-movement scripts

if { $opt(cp) == "" } {

puts "\*\*\* NOTE: no connection pattern specified."

set opt(cp) "none"

} else {

puts "Loading connection pattern..."

source $opt(cp)

}

if { $opt(sc) == "" } {

puts "\*\*\* NOTE: no scenario file specified."

set opt(sc) "none"

} else {

puts "Loading scenario file..."

source $opt(sc)

puts "Load complete..."

}

# Define initial node position in nam

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

# 20 defines the node size in nam, must adjust it according to your

# scenario

# The function must be called after mobility model is defined

$ns\_ initial\_node\_pos $node\_($i) 20

}

# Tell all nodes when the siulation ends

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

$ns\_ at $opt(stop).0 "$node\_($i) reset";

}

$ns\_ at $opt(stop).0 "$HA reset";

$ns\_ at $opt(stop).0 "$FA reset";

$ns\_ at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns\_ halt"

$ns\_ at $opt(stop).0001 "stop"

proc stop {} {

global ns\_ tracefd namtrace

close $tracefd

close $namtrace

}

# some useful headers for tracefile

puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp \

$opt(adhocRouting)"

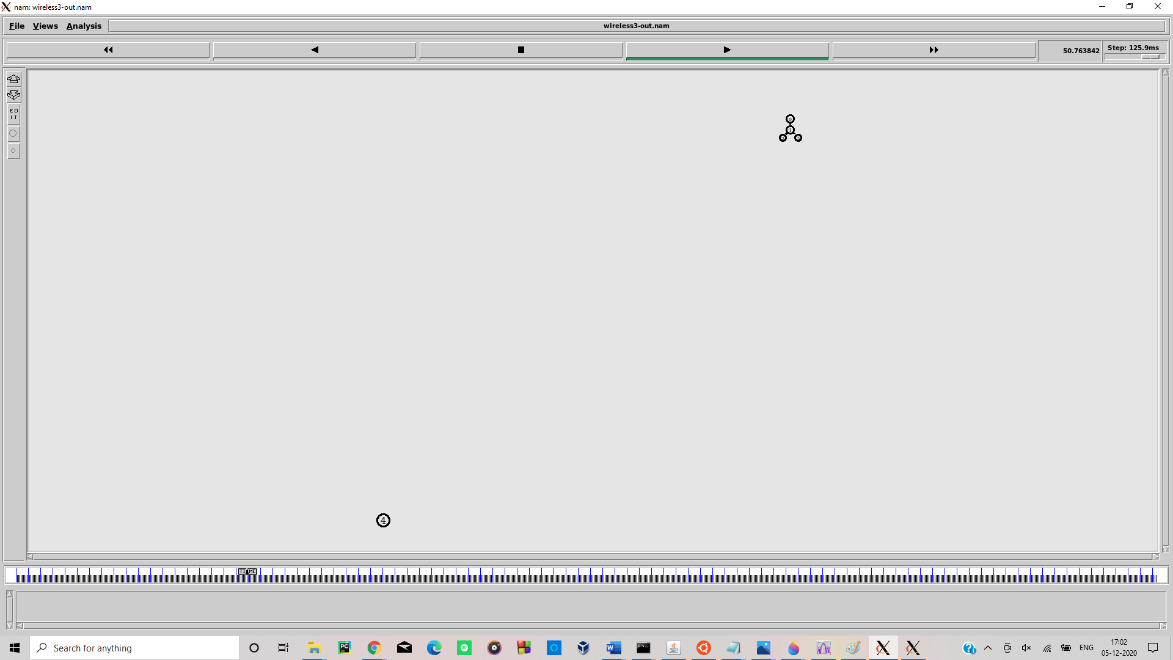
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed $opt(seed)"

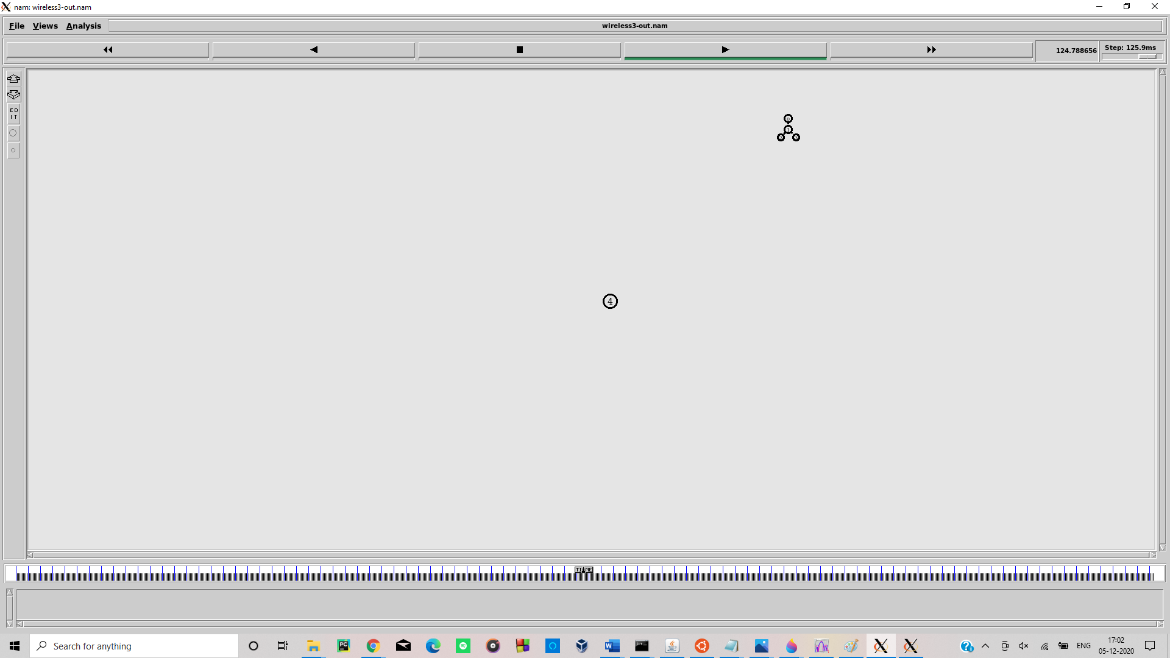
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

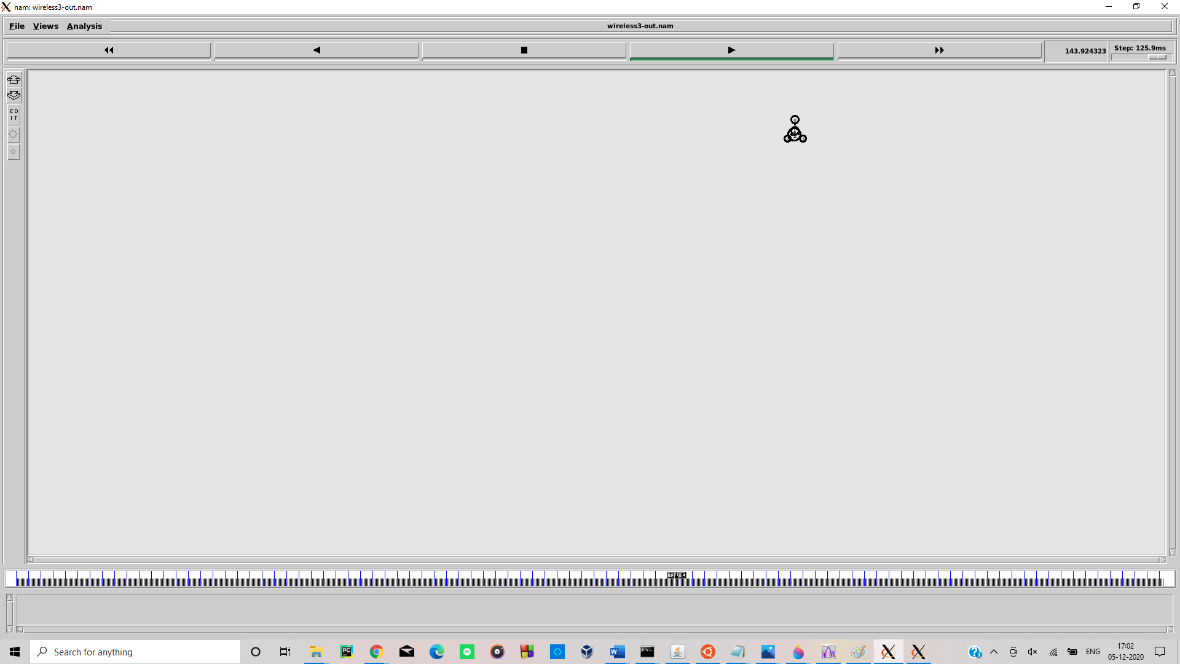
puts "Starting Simulation..."

$ns\_ run

**the below pictures are showing that the 4th node is mobile node and the other three are wireless immobile nodes.**







**Here we can see the node is moving. And then passing the packets to other three.**

Ques 3:-

Ans:

**A )**

. **tcl script: -**

set ns [new Simulator]

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

$ns at 0.0 "$n0 label Sender"  
$ns at 0.0 "$n1 label Receiver"

set nf [open A1-stop-n-wait.nam w]  
$ns namtrace-all $nf  
set f [open A1-stop-n-wait.tr w]  
$ns trace-all $f

$ns duplex-link $n0 $n1 0.2Mb 200ms DropTail  
$ns duplex-link-op $n0 $n1 orient right  
$ns queue-limit $n0 $n1 10

Agent/TCP set nam\_tracevar\_ true

set tcp [new Agent/TCP]  
$tcp set window\_ 1  
$tcp set maxcwnd\_ 1  
$ns attach-agent $n0 $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 add-agent-trace $tcp tcp  
$ns monitor-agent-trace $tcp  
$tcp tracevar cwnd\_

$ns at 0.1 "$ftp start"  
$ns at 3.0 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n1 $sink"  
$ns at 3.5 "finish"

$ns at 0.0 "$ns trace-annotate \"Stop and Wait with normal operation\""

$ns at 0.05 "$ns trace-annotate \"FTP starts at 0.1\""

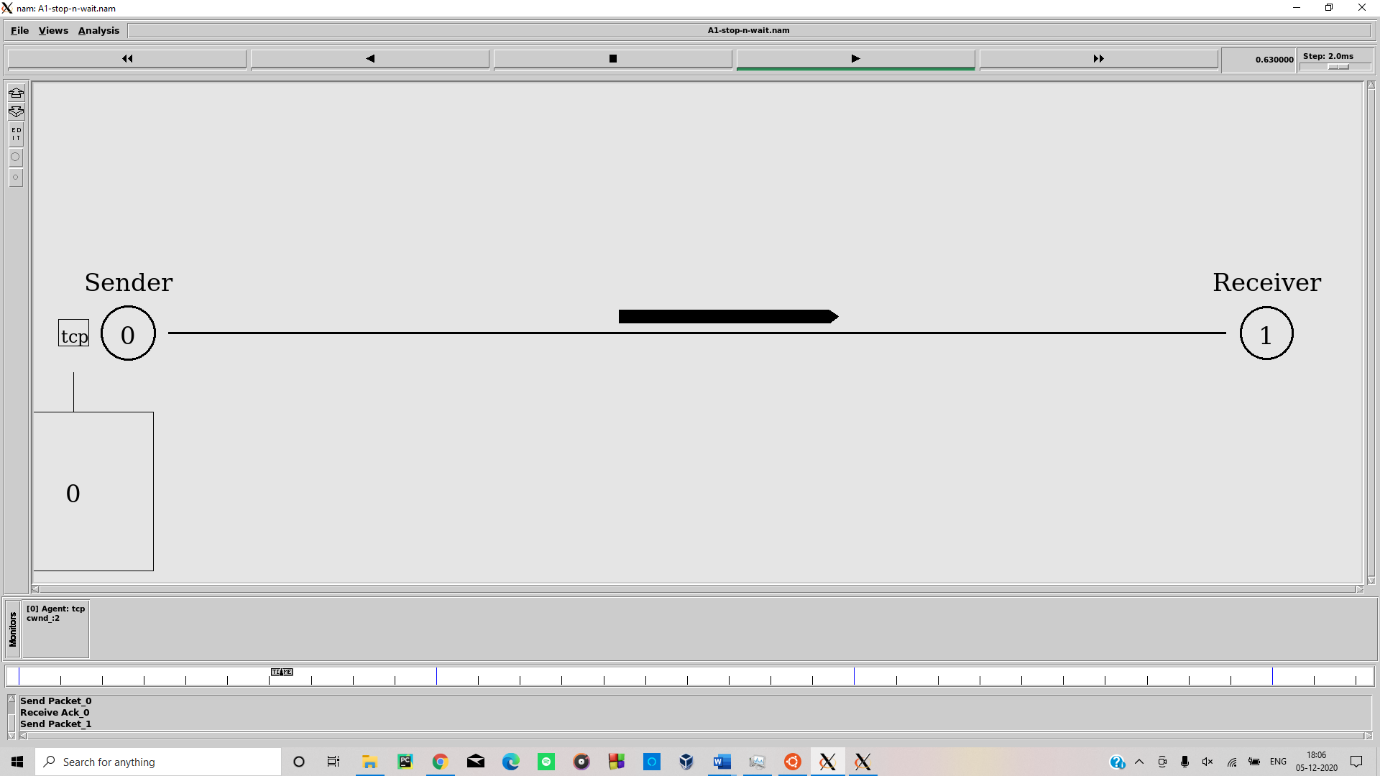
$ns at 0.11 "$ns trace-annotate \"Send Packet\_0\""  
$ns at 0.35 "$ns trace-annotate \"Receive Ack\_0\""  
$ns at 0.56 "$ns trace-annotate \"Send Packet\_1\""  
$ns at 0.79 "$ns trace-annotate \"Receive Ack\_1\""  
$ns at 0.99 "$ns trace-annotate \"Send Packet\_2\""  
$ns at 1.23 "$ns trace-annotate \"Receive Ack\_2  \""  
$ns at 1.43 "$ns trace-annotate \"Send Packet\_3\""  
$ns at 1.67 "$ns trace-annotate \"Receive Ack\_3\""  
$ns at 1.88 "$ns trace-annotate \"Send Packet\_4\""  
$ns at 2.11 "$ns trace-annotate \"Receive Ack\_4\""  
$ns at 2.32 "$ns trace-annotate \"Send Packet\_5\""  
$ns at 2.55 "$ns trace-annotate \"Receive Ack\_5    \""  
$ns at 2.75 "$ns trace-annotate \"Send Packet\_6\""  
$ns at 2.99 "$ns trace-annotate \"Receive Ack\_6\""

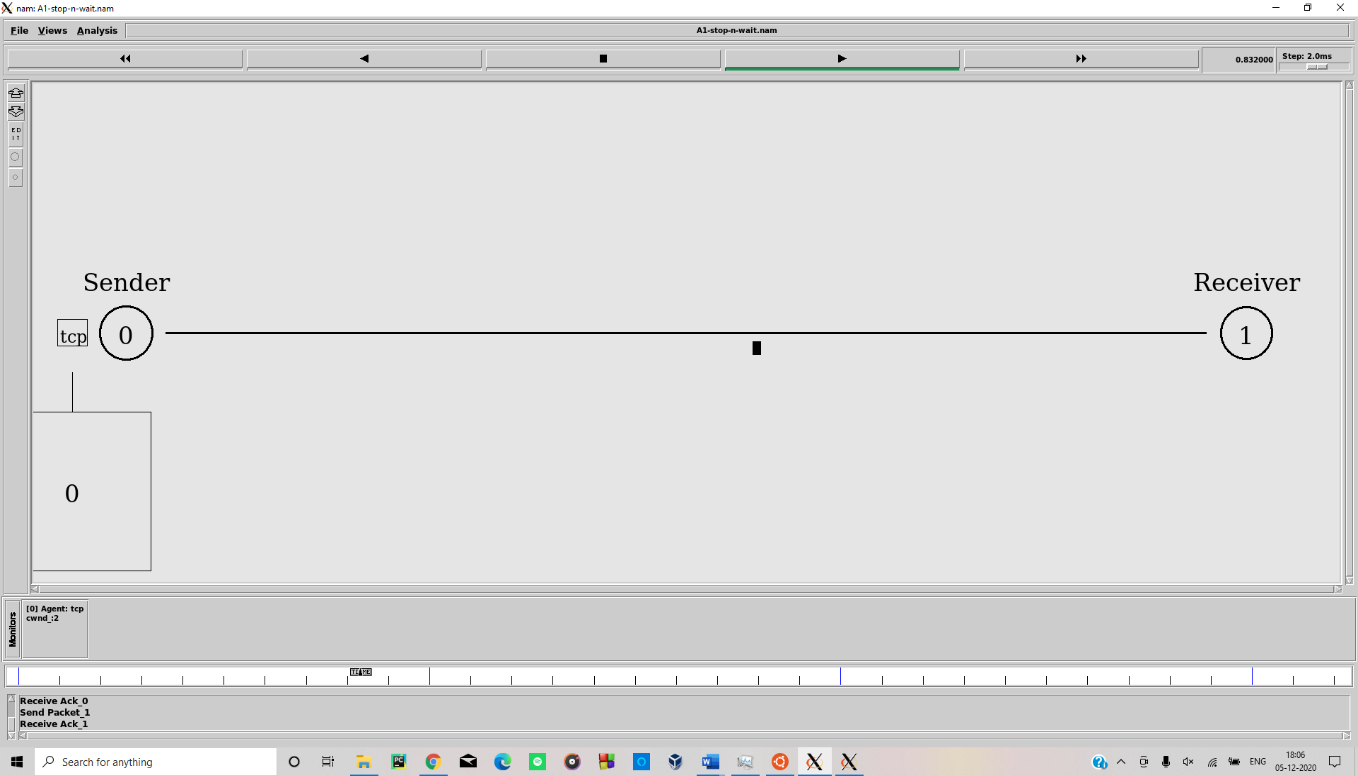
$ns at 3.1 "$ns trace-annotate \"FTP stops\""

proc finish {} {  
 global ns nf  
 $ns flush-trace  
 close $nf

puts "filtering..."  
 exec tclsh ../ns-allinone-2.1b5/nam-1.0a7/bin/namfilter.tcl A1-stop-n-wait.nam  
        puts "running nam..."  
        exec nam A1-stop-n-wait.nam &  
 exit 0}  
$ns run

**Screen Shot of the nam output file**

****

****

**B )**

.**tcl script:-**

**set ns [new Simulator]**

**$ns color 1 red**

**$ns trace-all [open B3-sliding-window.tr w]**

**$ns namtrace-all [open B3-sliding-window.nam w]**

**### build topology with 6 nodes**

**proc build\_topology { ns } {**

**global node\_**

**set node\_(s1) [$ns node]**

**set node\_(s2) [$ns node]**

**set node\_(r1) [$ns node]**

**set node\_(r2) [$ns node]**

**set node\_(s3) [$ns node]**

**set node\_(s4) [$ns node]**

**$node\_(s2) color "red"**

**$node\_(s4) color "red"**

**$node\_(r1) color "blue"**

**$node\_(r2) color "blue"**

**$node\_(r1) shape "rectangular"**

**$node\_(r2) shape "rectangular"**

**$ns at 0.0 "$node\_(s1) label Sliding-W-sender"**

**$ns at 0.0 "$node\_(s2) label CBR-sender"**

**$ns at 0.0 "$node\_(s3) label Sliding-W-receiver"**

**$ns at 0.0 "$node\_(s4) label CBR-receiver"**

**$ns duplex-link $node\_(s1) $node\_(r1) 0.5Mb 50ms DropTail**

**$ns duplex-link $node\_(s2) $node\_(r1) 0.5Mb 50ms DropTail**

**$ns duplex-link $node\_(r1) $node\_(r2) 0.5Mb 50ms DropTail**

**$ns duplex-link $node\_(r2) $node\_(s3) 0.5Mb 50ms DropTail**

**$ns duplex-link $node\_(r2) $node\_(s4) 0.5Mb 50ms DropTail**

**$ns queue-limit $node\_(r1) $node\_(r2) 100**

**$ns queue-limit $node\_(r2) $node\_(r1) 100**

**$ns duplex-link-op $node\_(s1) $node\_(r1) orient right-down**

**$ns duplex-link-op $node\_(s2) $node\_(r1) orient right-up**

**$ns duplex-link-op $node\_(r1) $node\_(r2) orient right**

**$ns duplex-link-op $node\_(r2) $node\_(s3) orient right-up**

**$ns duplex-link-op $node\_(r2) $node\_(s4) orient right-down**

**$ns duplex-link-op $node\_(r1) $node\_(r2) queuePos 0.5**

**$ns duplex-link-op $node\_(r2) $node\_(r1) queuePos 0.5**

**}**

**build\_topology $ns**

**Agent/TCP set nam\_tracevar\_ true**

**### sliding-window protocol between s1 and s3 (Black)**

**set tcp [$ns create-connection TCP $node\_(s1) TCPSink $node\_(s3) 0]**

**$tcp set windowInit\_ 4**

**$tcp set maxcwnd\_ 4**

**$tcp set class\_ 0**

**set ftp [$tcp attach-app FTP]**

**$ns add-agent-trace $tcp tcp**

**$ns monitor-agent-trace $tcp**

**$tcp tracevar cwnd\_**

**### CBR traffic between s2 and s4 (Red)**

**set cbr [$ns create-connection CBR $node\_(s2) Null $node\_(s4) 1]**

**$cbr set packetSize\_ 500**

**$cbr set interval\_ 0.05**

**$cbr set class\_ 1**

**proc finish {} {**

**global ns**

**$ns flush-trace**

**puts "filtering..."**

**exec tclsh ../ns-allinone-2.1b5/nam-1.0a7/bin/namfilter.tcl B3-sliding-window.nam**

**puts "running nam..."**

**exec nam B3-sliding-window.nam &**

**exit 0**

**}**

**### set operations**

**$ns at 0.1 "$ftp start"**

**$ns at 1.7 "$ftp stop"**

**$ns at 0.1 "$cbr start"**

**$ns at 1.7 "$cbr stop"**

**$ns at 2.0 "finish"**

**### add annotations**

**$ns at 0.0 "$ns trace-annotate \"Normal operation of <Sliding Window> with window size, 4\""**

**$ns at 0.1 "$ns trace-annotate \"FTP starts at 0.1\""**

**$ns at 0.1 "$ns trace-annotate \"CBR starts at 0.1\""**

**$ns at 0.11 "$ns trace-annotate \"Send Packet\_0,1,2,3 : window size, 4\""**

**$ns at 0.32 "$ns trace-annotate \"Ack\_0,1,2,3\""**

**$ns at 0.46 "$ns trace-annotate \"Send Packet\_4,5,6,7 : window size, 4\""**

**$ns at 0.66 "$ns trace-annotate \"Ack\_4,5,6,7\""**

**$ns at 0.81 "$ns trace-annotate \"Send Packet\_8,9,10,11 : window size, 4\""**

**$ns at 1.00 "$ns trace-annotate \"Ack\_8,9,10,11\""**

**$ns at 1.16 "$ns trace-annotate \"Send Packet\_12,13,14,15 : window size, 4\""**

**$ns at 1.35 "$ns trace-annotate \"Ack\_12,13,14,15\""**

**$ns at 1.50 "$ns trace-annotate \"Send Packet\_16,17,18,19 : window size, 4\""**

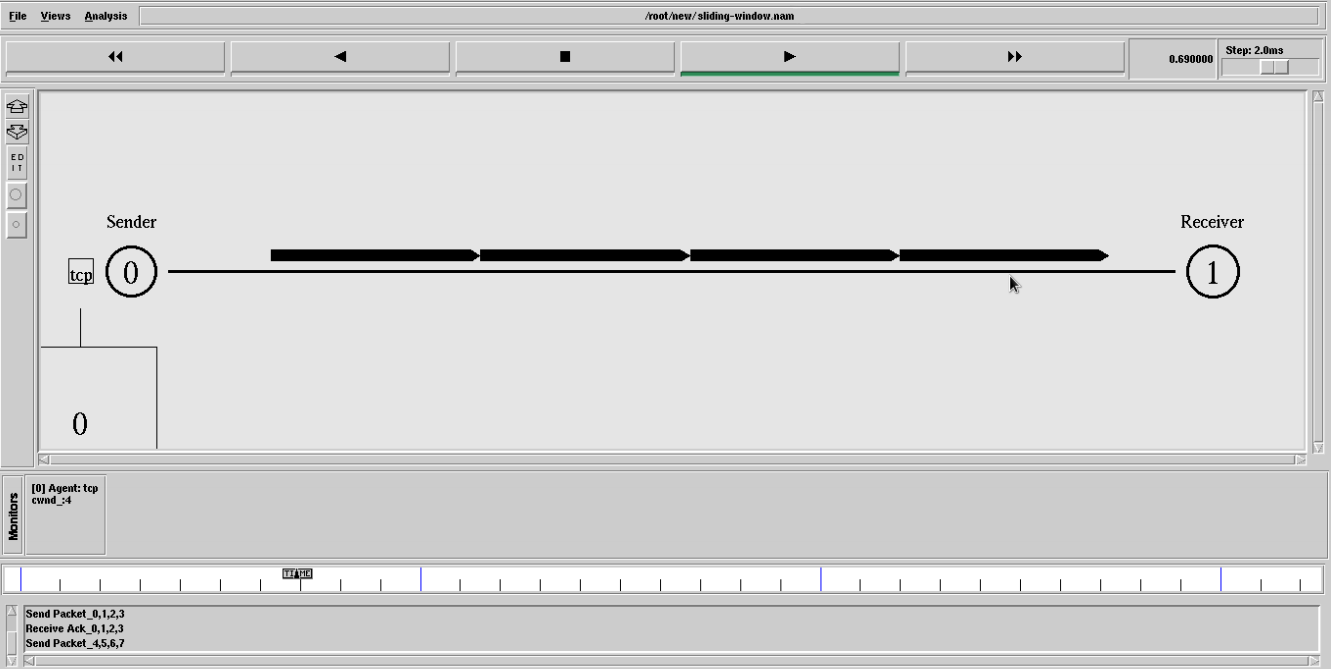
**$ns at 1.71 "$ns trace-annotate \"Ack\_16,17,18,19\""**

**$ns at 1.8 "$ns trace-annotate \"FTP stops at 1.7\""**

**$ns at 1.8 "$ns trace-annotate \"CBR stops at 1.7\""**

**$ns run**

**Screen Shot of the nam output file**



Ques 4: -

Ans:

**Script:-**

**## Setting The wireless Channels..**

**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) 5 ;# max packet in ifq**

**set val(nn) 8 ;# number of mobilenodes**

**set val(rp) DSR ;# routing protocol**

**set val(x) 750 ;# X dimension of topography**

**set val(y) 500 ;# Y dimension of topography**

**set val(stop) 9.95 ;# time of simulation end**

**set ns [new Simulator]**

**set tracefd [open dsr.tr w]**

**set namtrace [open dsr.nam w]**

**## Trace the nam and trace details from the main simulation..**

**$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)**

**## Color Descriptions..**

**$ns color 1 dodgerblue**

**$ns color 2 blue**

**$ns color 3 cyan**

**$ns color 4 green**

**$ns color 5 yellow**

**$ns color 6 black**

**$ns color 7 magenta**

**$ns color 8 gold**

**$ns color 9 red**

**$ns color 10 cornflowerblue**

**$ns color 11 deepskyblue**

**$ns color 12 steelblue**

**$ns color 13 navy**

**#$ns color 14 darkolivergreen**

**$ns color 15 brown**

**$ns color 16 darkorange**

**$ns color 17 orange**

**$ns color 18 darksalmon**

**$ns color 19 salmon**

**$ns color 20 greenyellow**

**$ns color 21 darkslategray**

**$ns color 22 darkkhaki**

**$ns color 23 darkorchid**

**$ns color 24 darkviolet**

**$ns color 25 darkcyan**

**$ns color 26 darkmagenta**

**## Setting The Distance Variables..**

**# For model 'TwoRayGround'**

**set dist(5m) 7.69113e-06**

**set dist(9m) 2.37381e-06**

**set dist(10m) 1.92278e-06**

**set dist(11m) 1.58908e-06**

**set dist(12m) 1.33527e-06**

**set dist(13m) 1.13774e-06**

**set dist(14m) 9.81011e-07**

**set dist(15m) 8.54570e-07**

**set dist(16m) 7.51087e-07**

**set dist(20m) 4.80696e-07**

**set dist(25m) 3.07645e-07**

**set dist(30m) 2.13643e-07**

**set dist(35m) 1.56962e-07**

**set dist(40m) 1.56962e-10**

**set dist(45m) 1.56962e-11**

**set dist(50m) 1.20174e-13**

**#Phy/WirelessPhy set CSThresh\_ $dist(50m)**

**#Phy/WirelessPhy set RXThresh\_ $dist(50m)**

**## Setting node config event with set of inputs..**

**puts "Node Configuration Started here...\n \**

**-channel $val(chan) \n \**

**-adhocRouting $val(rp) \n \**

**-llType $val(ll) \n \**

**-macType $val(mac) \n \**

**-ifqType $val(ifq) \n \**

**-ifqLen $val(ifqlen) \n \**

**-antType $val(ant) \n \**

**-propType $val(prop) \n \**

**-phyType $val(netif) \n"**

**$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**

**## Creating node objects..**

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

**#set node\_($i) [$ns node]**

**#}**

**set node\_(0) [$ns node]**

**set node\_(1) [$ns node]**

**$ns node-config -ifqLen 2**

**set node\_(2) [$ns node]**

**$ns node-config -ifqLen 50**

**set node\_(3) [$ns node]**

**set node\_(4) [$ns node]**

**set node\_(5) [$ns node]**

**set node\_(6) [$ns node]**

**set node\_(7) [$ns node]**

**## Provide initial location of mobilenodes..**

**$node\_(0) set X\_ 27.0**

**$node\_(0) set Y\_ 260.0**

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

**$node\_(1) set X\_ 137.0**

**$node\_(1) set Y\_ 348.0**

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

**$node\_(2) set X\_ 294.0**

**$node\_(2) set Y\_ 235.0**

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

**$node\_(3) set X\_ 414.0**

**$node\_(3) set Y\_ 342.0**

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

**$node\_(4) set X\_ 562.0**

**$node\_(4) set Y\_ 267.0**

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

**$node\_(5) set X\_ 279.0**

**$node\_(5) set Y\_ 447.0**

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

**$node\_(6) set X\_ -128.0**

**$node\_(6) set Y\_ 260.0**

**$node\_(6) set Z\_ 0.0**

**$node\_(7) set X\_ 727.0**

**$node\_(7) set Y\_ 269.0**

**$node\_(7) set Z\_ 0.0**

**## 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**

**}**

**## Set a TCP connection between node\_(0) and node\_(1)..**

**$ns at 1.0 "commn"**

**proc commn {} {**

**global ns node\_ rootPath**

**set now [$ns now]**

**set time 0.6**

**set time1 4.1**

**source "$rootPath/commn1.tcl";$ns at [expr $now+$time1] "commn"**

**}**

**$ns at 1.0 "$ns trace-annotate \"Node1 Dropping some packets\""**

**$ns at 2.0 "$ns trace-annotate \"Again node1 dropping some packets\""**

**$ns at 3.0 "$ns trace-annotate \"Node1 Make the Periodicall data dropping Attack\""**

**$ns at 3.0 "$ns trace-annotate \"Now the Node1 activity is called Blackhole\""**

**$node\_(0) color black**

**$node\_(1) color black**

**$ns at 3.0 "$node\_(1) color red"**

**$ns at 3.0 "$node\_(1) label Per\_drop\_attack"**

**## Generation of movements-Destination setting..**

**$ns at 0.0 "Mobility"**

**proc Mobility {} {**

**global ns node\_ rootPath**

**set ns\_ [Simulator instance]**

**set time 1.0**

**set moveTime 0.5**

**set now [$ns now]**

**source "$rootPath/setdest.tcl"**

**$ns\_ at [expr $now+$time] "Mobility"**

**}**

**#source "$rootPath/annotate.tcl"**

**## Telling nodes when the simulation ends..**

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

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

**}**

**## Procedure for pktDeliveryRatio graph..**

**set PktDelRatio 0.98**

**set MisBehRatio 0**

**set pdrgraph [open pdratio(DSR).tr w]**

**$ns at 0.1 "PktDelRatio"**

**proc PktDelRatio {} {**

**global ns rootPath pdrgraph PktDelRatio MisBehRatio pktdelratio misbehratio**

**set ns\_ [Simulator instance]**

**set time 1.2**

**set now [$ns\_ now]**

**source "$rootPath/PDRatio.tcl"**

**if {$now<1.2} {set PktDelRatio 0.98;set MisBehRatio 0}**

**if {$now>1.2} {**

**set PktDelRatio [expr $PktDelRatio-$pktdelratio]**

**set MisBehRatio [expr $MisBehRatio+$misbehratio]**

**}**

**puts $pdrgraph "$MisBehRatio $PktDelRatio"**

**$ns\_ at [expr $now+$time] "PktDelRatio"**

**}**

**## Procedure for RouteOverhead graph..**

**set RouteOverhead 0.98**

**set MisBehRatio 0**

**set rohgraph [open routeOH(DSR).tr w]**

**$ns at 0.1 "Overhead\_Cal"**

**proc Overhead\_Cal {} {**

**global ns rootPath rohgraph RouteOverhead MisBehRatio routeoverhead misbehratio**

**set ns\_ [Simulator instance]**

**set time 1.2**

**set now [$ns\_ now]**

**source "$rootPath/RouteOverHead.tcl"**

**if {$now<1.2} {set RouteOverhead 0.01;set MisBehRatio 0}**

**if {$now>1.2}**

**set RouteOverhead [expr $RouteOverhead+$routeoverhead]**

**set MisBehRatio [expr $MisBehRatio+$misbehratio]**

**}**

**puts $rohgraph "$MisBehRatio $RouteOverhead"**

**$ns\_ at [expr $now+$time] "Overhead\_Cal"**

**}$ns at $val(stop) "$ns nam-end-wireless $val(stop)"**

**$ns at $val(stop) "stop"**

**$ns at 10.01 "puts \"end simulation\" ; $ns halt"**

**## Setting Animation Rate..**

**$ns at 0.0 "$ns set-animation-rate 6.50ms"**

**## Stop procedure..**

**proc stop {} {**

**global ns tracefd namtrace**

**$ns flush-trace**

**close $tracefd**

**close $namtrace**

**exec nam dsr.nam &**

**exit 0**

**}**

**the Following screenshot for packet dropping**

