Experiment No: 8

ABHISHEK SINGH

119A3004

SE-IT(E1).

Aim : Comparative analysis of routing protocols with respect to QoS parameters using Xgraph/gnuplot

for different load conditions.

Theory : The experiment two parts A. Physical layer configuration, and B. Setting of Network Topology

XGRAPH :

The xgraph program draws a graph on an x-display given data read from either data file or from standard

input if no files are specified. It can display up to 64 independent data sets using different colours and line

styles for each set. It annotates the graph with a title, axis labels, grid lines or tick marks, grid labels and a

legend.

Syntax:

Xgraph [options] file-name

Options are listed here

/-bd <color> (Border)

This specifies the border color of the xgraph window.

/-bg <color> (Background)

This specifies the background color of the xgraph window.

/-fg<color> (Foreground)

This specifies the foreground color of the xgraph window.

/-lf <fontname> (LabelFont)

All axis labels and grid labels are drawn using this font.

/-t<string> (Title Text)

This string is centered at the top of the graph.This is the unit name for the x-axis. Its default is “X”.

/-y <unit name> (YunitText)

This is the unit name for the y-axis. Its default is “Y”.

Steps to create and execute tcl script:

Step 1: Open any text editor (vi, nano)

Step 2: Write the program using ns2 tcl script and save with extension as filename.tcl

Step 3: Execute tcl script as “ns filename.tcl”

Step 4: Observe the output.

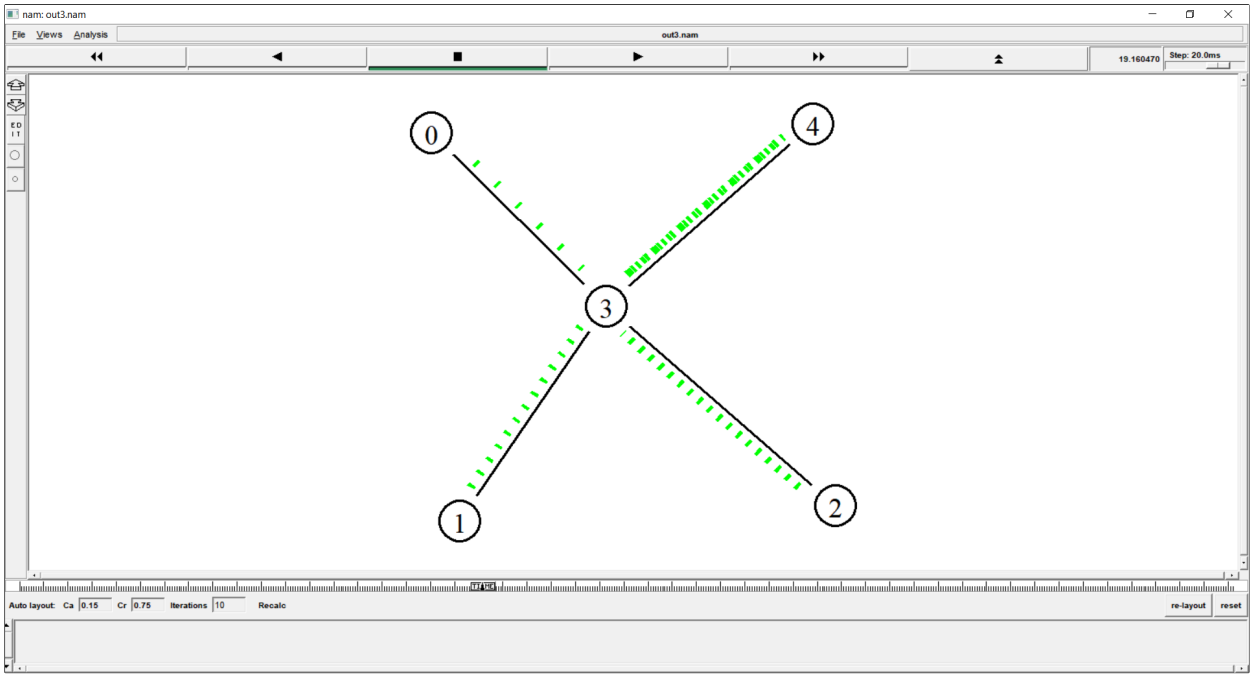
Code:

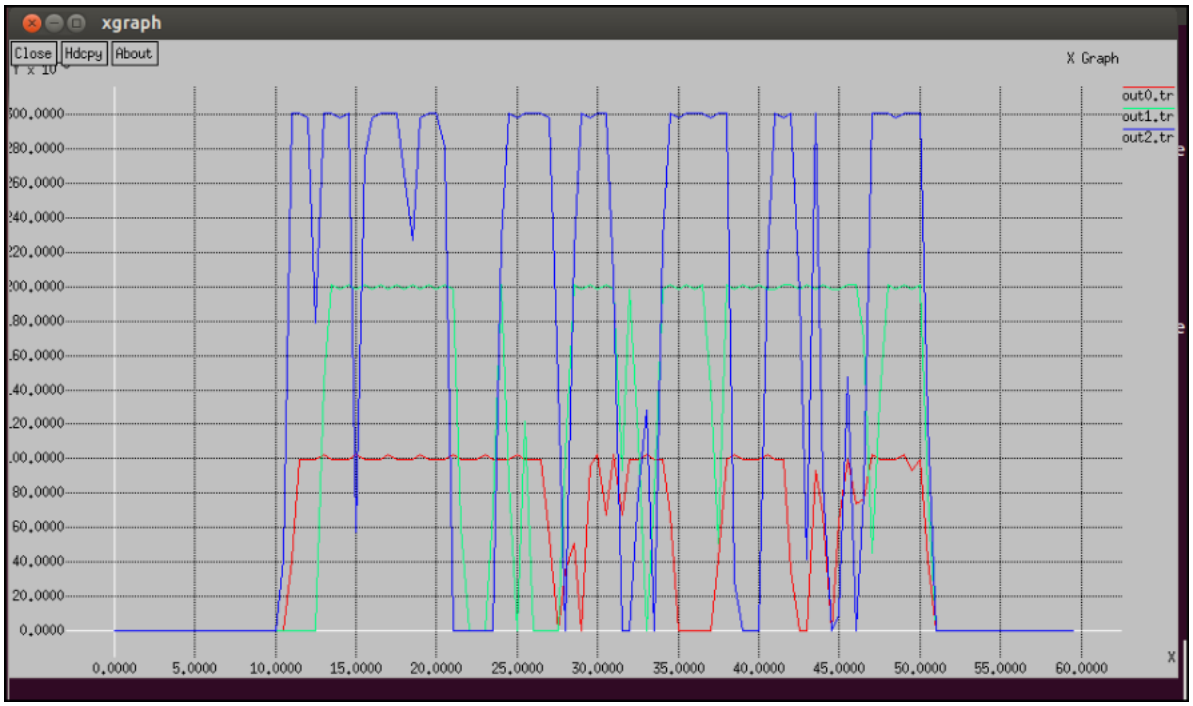
```
set ns [new Simulator]
$ns color 0 green
set f0 [open out0.tr w]
set f1 [open out1.tr w]
set f2 [open out2.tr w]
set f3 [open out3.nam w]
$ns namtrace-all $f3
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
$ns duplex-link $n0 $n3 1Mb 100ms DropTail
$ns duplex-link $n1 $n3 1Mb 100ms DropTail
$ns duplex-link $n2 $n3 1Mb 100ms DropTail
$ns duplex-link $n3 $n4 1Mb 100ms DropTailproc finish {} {
global ns f0 f1 f2 f3
$ns flush-trace
close $f0
close $f1
close $f2
close $f3
```

```
exec nam out3.nam &
exec xgraph out0.tr out1.tr out2.tr -geometry 800x400 &
exit 0
}
proc attach-expoo-traffic { node sink size burst idle rate } {
set ns [Simulator instance]
set source [new Agent/UDP]
$ns attach-agent $node $source
set traffic [new Application/Traffic/Exponential]
$traffic set packetSize_ $size
$traffic set burst_time_ $burst
$traffic set idle_time_ $idle
$traffic set rate_ $rate
$traffic attach-agent $source
$ns connect $source $sink
return $traffic
}
proc record {} {
global sink0 sink1 sink2 f0 f1 f2
set ns [Simulator instance]
set time 0.5
set bw0 [$sink0 set bytes_]
set bw1 [$sink1 set bytes_]
set bw2 [$sink2 set bytes_]
set now [$ns now]
puts $f0 "$now [expr $bw0/$time*8/1000000]"
puts $f1 "$now [expr $bw1/$time*8/1000000]"
puts $f2 "$now [expr $bw2/$time*8/1000000]"
$sink0 set bytes_ 0
$sink1 set bytes_ 0
$sink2 set bytes_ 0
$ns at [expr $now+$time] "record"
}
set sink0 [new Agent/LossMonitor]
set sink1 [new Agent/LossMonitor]
Xgraph command
to plot graph
Procedure to attach
traffic to source node
Procedure to record
different load conditions
in trace filesset sink2 [new Agent/LossMonitor]
$ns attach-agent $n4 $sink0
$ns attach-agent $n4 $sink1
$ns attach-agent $n4 $sink2
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]
$ns at 0.0 "record"
$ns at 10.0 "$source0 start"
$ns at 10.0 "$source1 start"
$ns at 10.0 "$source2 start"
$ns at 50.0 "$source0 stop"
$ns at 50.0 "$source1 stop"
$ns at 50.0 "$source2 stop"
```

```
$ns at 60.0 "finish"
$ns run
```

Output:





Conclusion: Thus we successfully plotted graph for different load conditions using xgraph for routing protocol.