LOVELY PROFESSIONAL UNIVERSITY

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

ETE REPORT

ECE 396

PROJECT TITLE

IMPLEMENTATION OF DSR PROTOCOL AND CALCULATE VARIOUS QoS USING XGRAPH

SUBMITTED TO: Dr. KRISHAN KUMAR

GROUP MEMBERS(G2 B5)

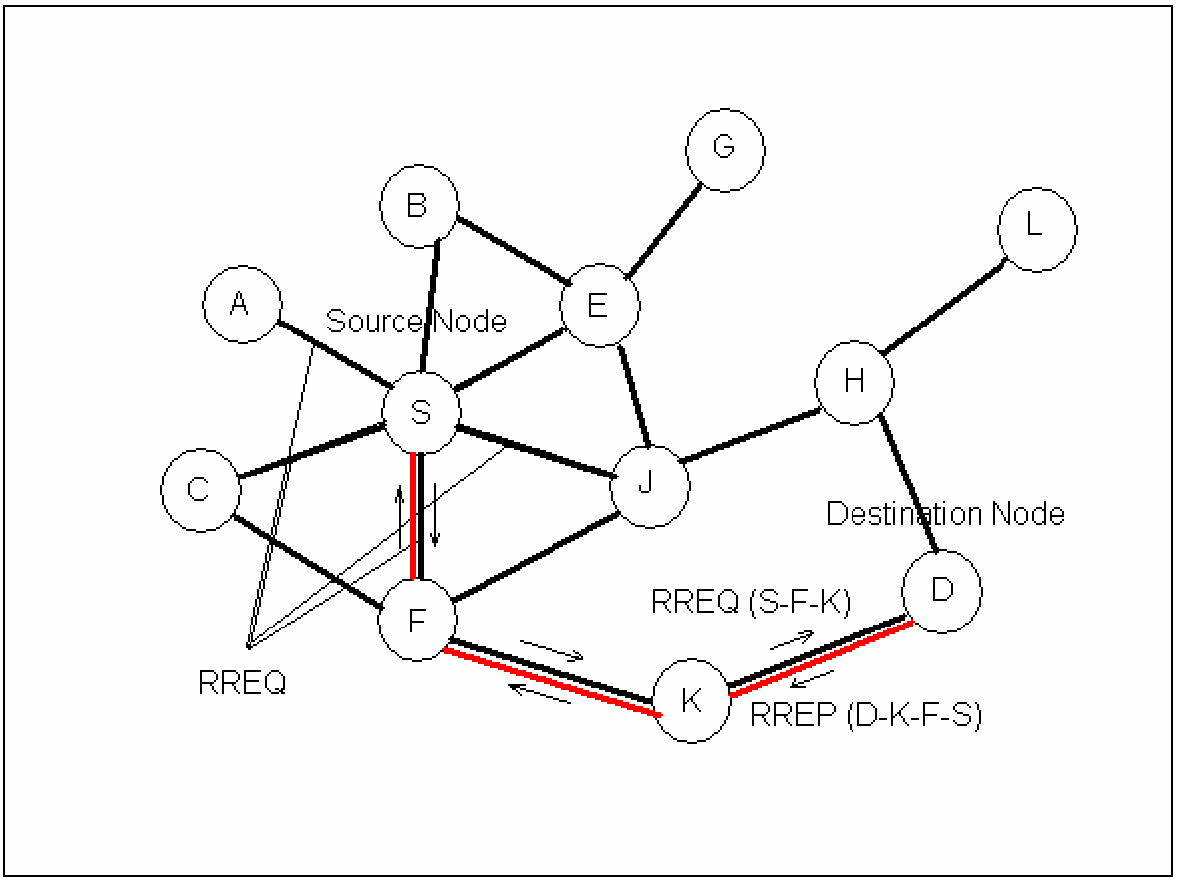
NAME	ROLL NO	REGISTRATION NO
ANUSHI CHAUHAN	36	11806957
Md. ARIFUL ISLAM	42	11809175

INTRODUCTION

The Dynamic Source Routing protocol (DSR) is a simple and efficient routing protocol designed specifically for use in multi-hop wireless ad hoc networks of mobile nodes. DSR allows the network to be completely self-organizing and self-configuring, without the need for any existing network infrastructure or administration.

Dynamic source routing protocol (DSR) is an on-request convention intended to limit the data transmission devoured by control parcels in specially appointed remote organizations by dispensing with the occasional table-update messages needed in the table driven methodology. The significant distinction among this and the other on-request directing conventions is that it is reference point less and thus doesn't need occasional hi parcel (guide) transmissions, which are utilized by a hub to illuminate its neighbours regarding its quality. The essential methodology of this convention (and any remaining on-request steering conventions) during the course development stage is to set up a course by flooding Route Request bundles in the organization.

The objective hub, on accepting a Route Request parcel, reacts by sending a Route Reply bundle back to the source, which conveys the course crossed by the Route Request parcel got. Dynamic Source Routing (DSR) is a self-keeping up directing convention for remote organizations. In Dynamic Source Routing, each source decides the course be utilized in communicating its bundles to chosen objections.



NS-2

NS2 stands for Network Simulator Version. It is an open-source event-driven simulator designed specifically for research in computer communication networks. NS2 comprises of two key dialects: C++ and Object-situated Tool Command Language (OTcl). While the C++ characterizes the interior system (i.e., a backend) of the reproduction protests, the OTcl sets up reenactment by amassing and designing the articles just as booking discrete occasions. The C++ and the OTcl are connected together utilizing TclCL.

Features of NS-2:

- It is a discrete occasion test system for systems administration research.
- It offers significant help to recreate bundle of conventions like TCP, FTP, UDP, https and DSR.

- It recreates wired and remote organization.
- It is basically Unix based.
- Utilizations TCL as its scripting language.
- Otcl: Object arranged help.
- Tclcl: C++ and otcl linkage.
- Discrete occasion scheduler

QoS

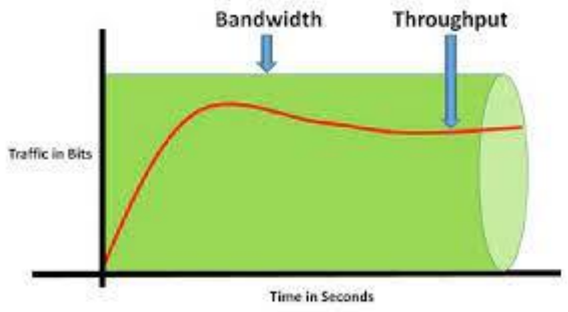
Quality of administration (QoS) is the portrayal or estimation of the general exhibition of a help, for example, a communication or PC organization or a distributed computing administration, especially the presentation seen by the clients of the organization. To quantitatively quantify nature of administration, a few related parts of the organization administration are frequently thought to be, like bundle misfortune, bit rate, throughput, transmission delay, accessibility, jitter etc.

In the field of networking and other bundle exchanged media transmission organizations, nature of administration eludes to traffic prioritization and asset reservation control components as opposed to the accomplished help quality. Nature of administration is the capacity to give various needs to various applications, clients, or information streams, or to ensure a specific degree of execution to an information stream.

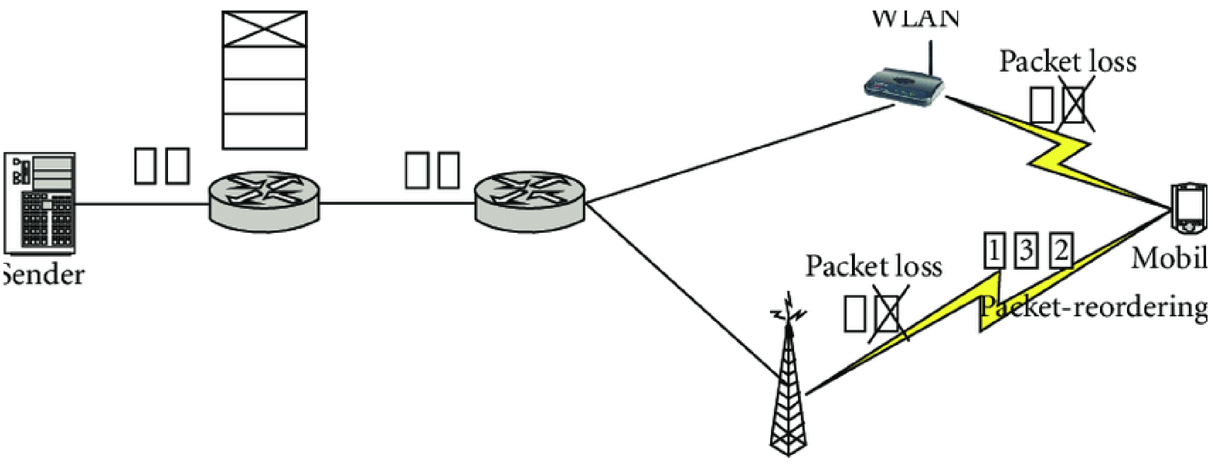
Quality of service is especially significant for the vehicle of traffic with uncommon necessities. Specifically, engineers have acquainted Voice over IP innovation with permit PC organizations to become as helpful as phone networks for sound discussions, just as supporting new applications with much stricter organization execution necessities.

Selection of Parameters:

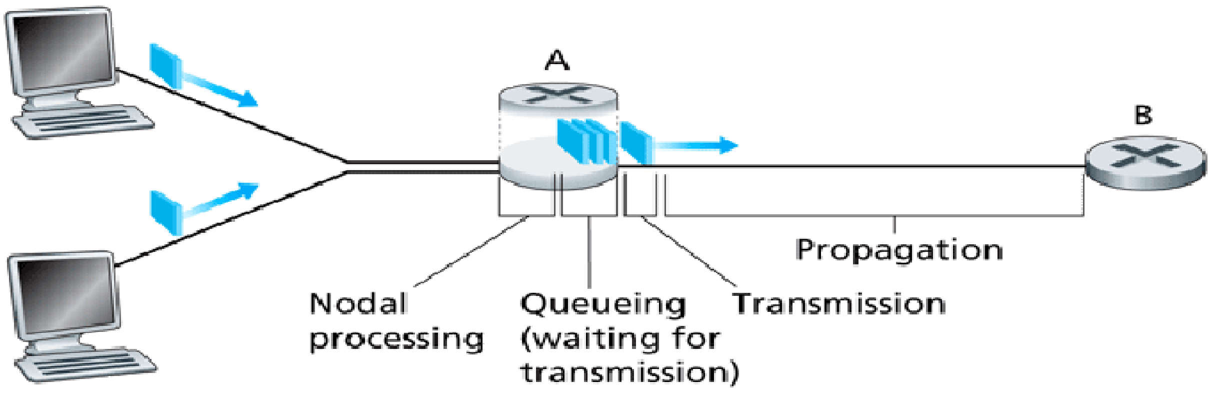
• Throughput: Throughput discloses to you how much information was moved from a source at some random time and transmission capacity reveals to you how much information could hypothetically be moved from a source at some random time. Knowing how both throughput and data transmission are performing is urgent for heads expecting to get a reasonable image of their organization's presentation. Throughput estimates the number of bundles show up at their objections effectively. Generally, throughput limit is estimated in bits each second, yet it can likewise be estimated in information each second.



• Packet Loss: Path loss is the decrease in influence thickness (weakening) of an electromagnetic wave as it spreads through space. This term is generally utilized in remote correspondences and sign proliferation. Path loss models depict the sign weakening between a communication and a get receiving wire as a component of the engendering distance and different boundaries. A few models incorporate numerous subtleties of the territory profile to gauge the sign constriction, though others simply think about transporter recurrence and distance. Radio wire statures are other basic boundaries.



• Packet Delay: Packet delay is the time delay between when a packet of data goes from one place to another, also referred to as latency. In packet switched networks, there are four types of commonly identified delays – processing, queuing, transmission and propagation delays. Processing delay is the CPU cycles needed to look at the packet headers and decide what to do with the packet, and do it – basically the time needed to process the packet.



XGRAPH

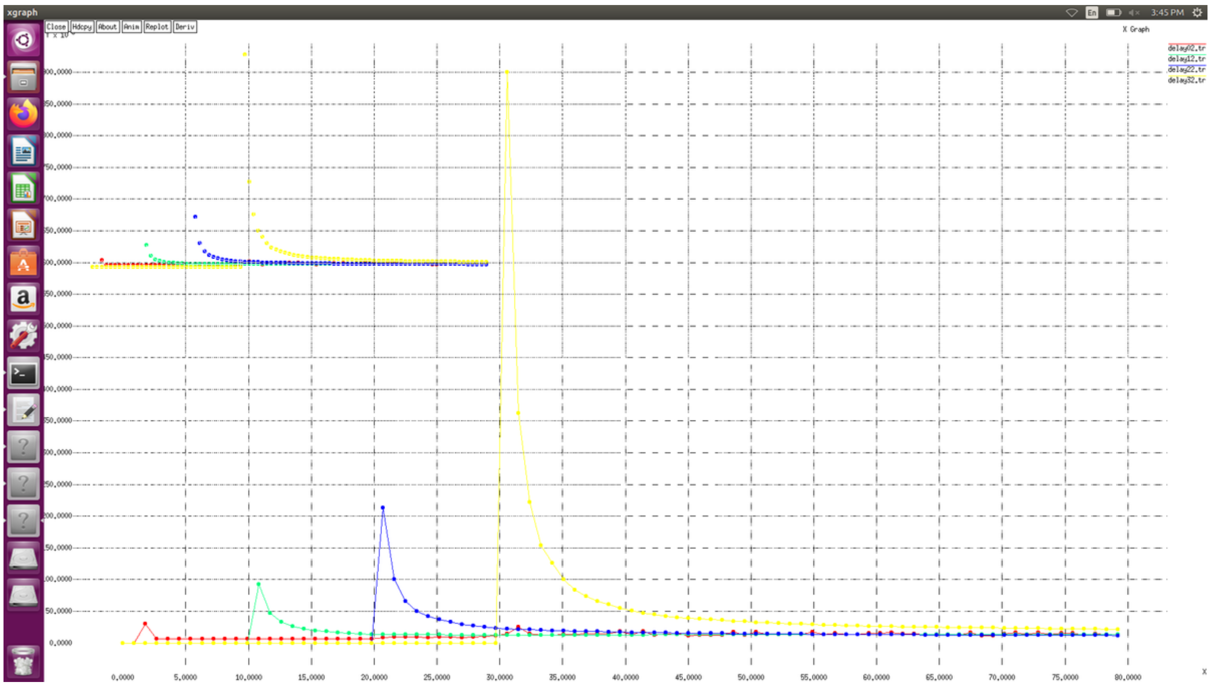
Xgraph in ns2 is used to plot the network parameter characteristics like throughput, delay, jitter, latency etc.

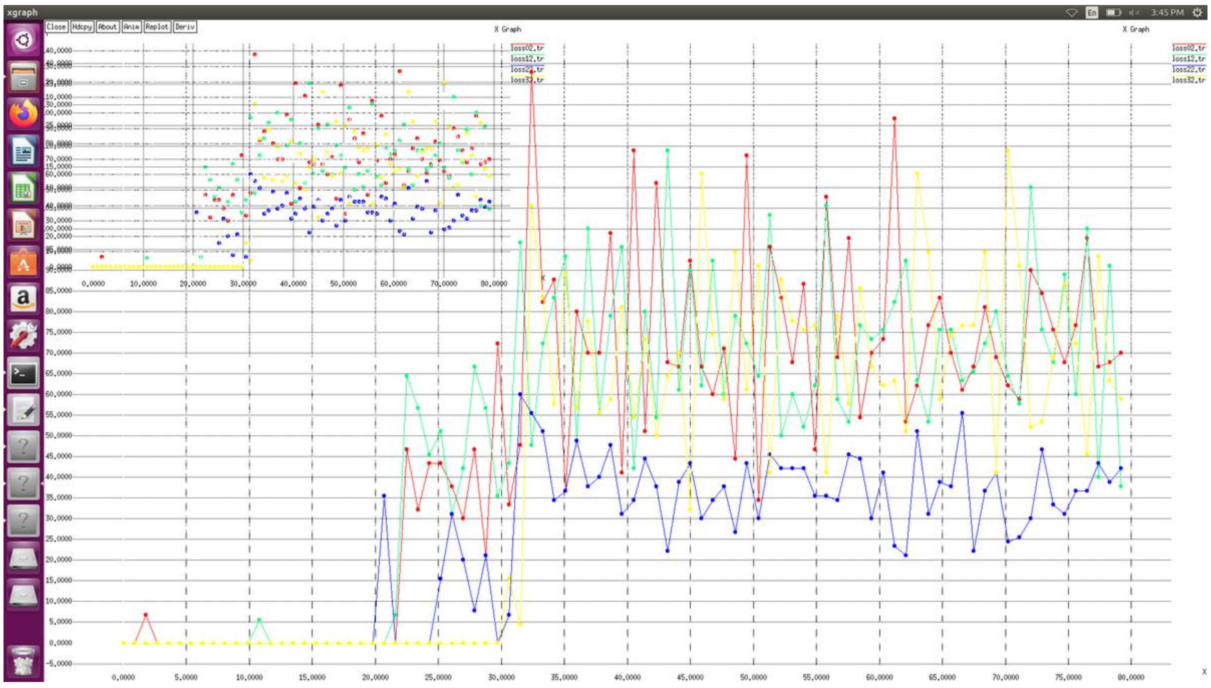
Xgraph is an X-Windows application that includes:

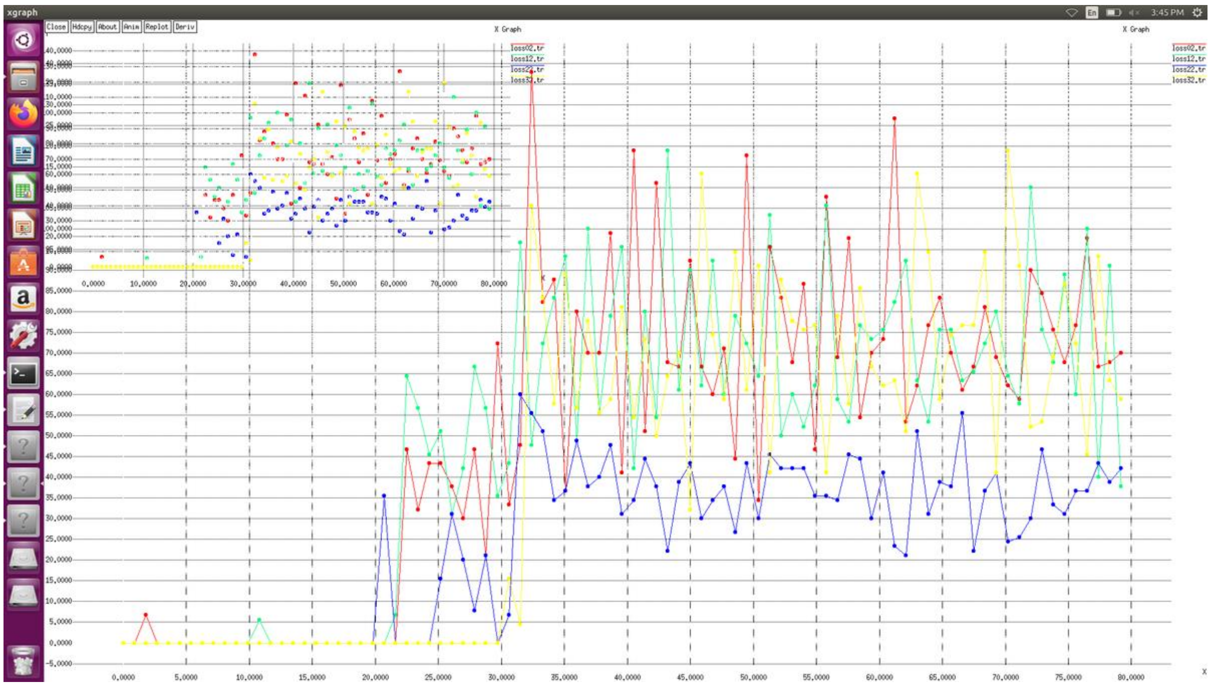
- Interactive plotting and graphing.
- Animation and directives.

- Portability and bug fixes.

GRAPHS







CODE

```
# Define Node Configuration Parameters


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) 8; # number of mobilenodes

set val(rp) DSR; # routing protocol

set val(x) 1200 ; # X dimension of topography
set val(y) 1200 ; # Y dimension of topography
set val(stop) 10.0 ;

if { $val(rp) == "DSR" } { set val(ifq) CMUPriQueue
} else { set val(ifq) Queue/DropTail/PriQueue
}
# time of simulation end
Mac/802_11 set RTSThreshold_ 3000
Mac/802_11 set basicRate_ 1Mb
Mac/802_11 set dataRate_ 2Mb


# ***Throughput file ***

set f0 [open out02.tr w]
set f1 [open out12.tr w]
set f2 [open out22.tr w]
set f3 [open out32.tr w]



#***Packet Loss file***

set f4 [open loss02.tr w]
set f5 [open loss12.tr w]
set f6 [open loss22.tr w]
set f7 [open loss32.tr w]
```

```
# ***Packet Delay file ***

set f8 [open delay02.tr w]
set f9 [open delay12.tr w]
set f10 [open delay22.tr w]
set f11 [open delay32.tr w]


# Initialize Simulator
set ns_ [new Simulator]


# Initialize Trace file
set tracefd [open tracer.tr w]
$ns_ trace-all $tracefd

#Initialize NAM
set namtrace [open sim12.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)


# set up topography object
set topo [new Topography]
$topo load_flatgrid 500 500



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 OFF




# Node Creation

for {set i 0} {$i < $val(nn) } { incr i } {
set node_($i) [$ns_ node]
$node_($i) random-motion 0;
}

# Set up Node Coordinates
```

```
$node_(0) set X_ 5.0
$node_(0) set Y_ 5.0
$node_(0) set Z_ 0.0

$node_(1) set X_ 200.0
$node_(1) set Y_ 5.0
$node_(1) set Z_ 0.0

$node_(2) set X_ 5.0
$node_(2) set Y_ 50.0
$node_(2) set Z_ 0.0

$node_(3) set X_ 200.0
$node_(3) set Y_ 50.0
$node_(3) set Z_ 0.0

$node_(4) set X_ 5.0
$node_(4) set Y_ 100.0
$node_(4) set Z_ 0.0

$node_(5) set X_ 5.0
$node_(5) set Y_ 100.0
$node_(5) set Z_ 0.0
```

```
$node_(6) set X_ 2.0
$node_(6) set Y_ 150.0
$node_(6) set Z_ 0.0



$node_(7) set X_ 200.0
$node_(7) set Y_ 150.0
$node_(7) set Z_ 0.0



# Setup TCP connection between nodes

set agent1 [new Agent/UDP];
$agent1 set prio_ 0;
set sink [new Agent/LossMonitor];
$ns_ attach-agent $node_(0) $agent1;
$ns_ attach-agent $node_(1) $sink;
$ns_ connect $agent1 $sink;
set app1 [new Application/Traffic/CBR];
$app1 set packetSize_ 512;
$app1 set rate_ 600kb;
$app1 attach-agent $agent1;

set agent2 [new Agent/UDP];
$agent2 set prio_ 1;
set sink2 [new Agent/LossMonitor];
```

```
$ns_ attach-agent $node_(2) $agent2;
$ns_ attach-agent $node_(3) $sink2
$ns_ connect $agent2 $sink2;
set app2 [new Application/Traffic/CBR];
$app2 set packetSize_ 512;
$app2 set rate_ 600kb;
$app2 attach-agent $agent2;



set agent3 [new Agent/UDP];
$agent3 set prio_ 2;
set sink3 [new Agent/LossMonitor];
$ns_ attach-agent $node_(4) $agent3;
$ns_ attach-agent $node_(5) $sink3;
$ns_ connect $agent3 $sink3;
set app3 [new Application/Traffic/CBR];
$app3 set packetSize_ 512;
$app3 set rate_ 600kb;
$app3 attach-agent $agent3;


set agent4 [new Agent/UDP];
$agent4 set prio_ 3;
set sink4 [new Agent/LossMonitor];
$ns_ attach-agent $node_(6) $agent4;
$ns_ attach-agent $node_(7) $sink4;
$ns_ connect $agent4 $sink4;
set app4 [new Application/Traffic/CBR];
```

```
$app4 set packetSize_ 512;
$app4 set rate_ 600kb;
$app4 attach-agent $agent4;




# Define node size in nam
for {set i 0} {$i < $val(nn)} { incr i } {
$ns_ initial_node_pos $node_($i) 20
}



#Initialize Flags

set holdtime 0
set holdseq 0

set holdtime1 0
set holdseq1 0

set holdtime2 0
set holdseq2 0

set holdtime3 0
set holdseq3 0

set holdrate1 0
set holdrate2 0
set holdrate3 0
set holdrate4 0



# Record statistic

proc record {} {

global sink sink2 sink3 sink4 f0 f1 f2 f3 f4 f5 f6 f7 holdtime holdseq holdtime1 holdseq1
holdtime2 holdseq2 holdtime3 holdseq3 f8 f9 f10 f11 holdrate1 holdrate2 holdrate3
holdrate4



set ns [Simulator instance]

set time 0.9; #Sampling time


set bw0 [$sink set bytes_]

set bw1 [$sink2 set bytes_]

set bw2 [$sink3 set bytes_]

set bw3 [$sink4 set bytes_]


set bw4 [$sink set nlost_]

set bw5 [$sink2 set nlost_]

set bw6 [$sink3 set nlost_]

set bw7 [$sink4 set nlost_]


set bw8 [$sink set lastPktTime_]

set bw9 [$sink set npkts_]


set bw10 [$sink2 set lastPktTime_]

set bw11 [$sink2 set npkts_]


set bw12 [$sink3 set lastPktTime_]
set bw13 [$sink3 set npkts_]

set bw14 [$sink4 set lastPktTime_]
set bw15 [$sink4 set npkts_]

set now [$ns now]

# Record Bit Rate



puts $f0 "$now [expr (($bw0+$holdrate1)*8)/(2*$time*1000000)]"
puts $f1 "$now [expr (($bw1+$holdrate2)*8)/(2*$time*1000000)]"
puts $f2 "$now [expr (($bw2+$holdrate3)*8)/(2*$time*1000000)]"
puts $f3 "$now [expr (($bw3+$holdrate4)*8)/(2*$time*1000000)]"

# Record Packet Loss
puts $f4 "$now [expr $bw4/$time]"
puts $f5 "$now [expr $bw5/$time]"
puts $f6 "$now [expr $bw6/$time]"
puts $f7 "$now [expr $bw7/$time]"

# Record Packet Delay
if { $bw9 > $holdseq} {
puts $f8 "$now [expr ($bw8 -$holdtime)/($bw9 - $holdseq)]"
} else {puts $f8 "$now [expr ($bw9 - $holdseq)]"
}



if { $bw11 > $holdseq1} {
```

```
puts $f9 "$now [expr ($bw10 -$holdtime1)/($bw11 - $holdseq1)]"

} else { puts $f9 "$now [expr ($bw11 - $holdseq1)]"

}


if { $bw13 > $holdseq2} {

puts $f10 "$now [expr ($bw12 -$holdtime2)/($bw13 - $holdseq2)]"

} else { puts $f10 "$now [expr ($bw13 - $holdseq2)]"

}


if { $bw15 > $holdseq3} {

puts $f11 "$now [expr ($bw14 -$holdtime3)/($bw15 - $holdseq3)]"

} else { puts $f11 "$now [expr ($bw15 - $holdseq3)]"

}



# Reset Variables

$sink set bytes_ 0

$sink2 set bytes_ 0

$sink3 set bytes_ 0

$sink4 set bytes_ 0



$sink set nlost_ 0

$sink2 set nlost_ 0

$sink3 set nlost_ 0

$sink4 set nlost_ 0



set holdtime $bw8
set holdseq $bw9

set holdrate1 $bw0
set holdrate2 $bw1
set holdrate3 $bw2
set holdrate4 $bw3



$ns at [expr $now+$time] "record";
}

#Time
$ns_ at 0.0 "record"
$ns_ at 1.4 "$app1 start";
$ns_ at 10.0 "$app2 start";
$ns_ at 20.0 "$app3 start";
$ns_ at 30.0 "$app4 start";

# Stop Simulation

$ns_ at 80.0 "stop"

# Reset Nodes
for { set i 0} {$i < $val(nn) } {incr i} {
$ns_ at 80.0 "$node_($i) reset";
}

# Exit simulation
$ns_ at 80.01 "puts \"NS EXITING....\"; $ns_ halt"
```

```
proc stop {} {
global ns_ tracefd f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 f10 f11

# Close Trace file



close $f0
close $f1
close $f2
close $f3



close $f4
close $f5
close $f6
close $f7



close $f8
close $f9
close $f10
close $f11

#XGRAPH
exec xgraph out02.tr out12.tr out22.tr out32.tr -geometry 800x400 -P -bg white &
exec xgraph loss02.tr loss12.tr loss22.tr loss32.tr -geometry 800x400 -P -bg white &
exec xgraph delay02.tr delay12.tr delay22.tr delay32.tr -geometry 800x400 -P -bg white &
$ns_ flush-trace
close $tracefd
exit 0
}
puts "Strating Simulation....."

$ns_ run
```

Conclusion

From this project, we learnt the basic operation of NS-2 network simulator. Got to learn about the Mobile Ad-hoc Network (MANET); how it works and also had a detailed study of different routing protocols. We have simulated DSR protocol using NS-2 and have compared the DSR routing protocol with ZRP for some parameters like PDR, End-to-end delay, throughput and packet loss using X-Graph. With that we have also compared the parameters like Delay, PDR and Throughput for normal DSR using X-graph.

References:

- S.S.Dhenakaran and A.Parvathavarthini. An Overview of Routing Protocols in Mobile Ad-Hoc Network. International Journal of Advanced Research in Computer Science and Software Engineering; Volume 3, Issue 2, February 2013
- Neha Jain and Yogesh Chaba. Simulation based Performance Analysis of Zone Routing Protocol in Manet. International Journal of Computer Applications; Volume 88 – No.4, February 2014
- www.wikipedia.org
- www.ijarcet.org/wp-content/uploads/IJARCET-VOL-1-ISSUE-4-525-529.pdf
- www.ijcst.org/Volume3/Issue7/p10_3_7.pdf

- www.slogix.in
- www.ijcsits.org/papers/Vol2no32012/27vol2no3.pdf
- www.ripublication.com/irph/ijict_spl/ijictv4n4spl_07.pdf
- www.arxiv.org/ftp/arxiv/papers/0909/0909.2371.pdf
- www.youtube.com/watch?v=8B_of6FK9L