TITLE OF THE PROJECT BASED WORK

BLACK HOLE ATTACKS IN WIRELESS SENSOR NETWORKS

Report submitted to the SASTRA Deemed to be University
as the requirement for the course

CSE302: COMPUTER NETWORKS

Submitted by

NAME: VARSHITH SAI NARAGAM
(Reg.No.:224003157, III B.Tech CSE)

December 2022

DEPARTMENT OF CSE/SRC

KUMBAKONAM, TAMIL NADU, INDIA – 612001

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act,1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

DEPARTMENT OF CSE/SRC

KUMBAKONAM 612001.

Bonafide Certificate

This is to certify that the report titled “**black hole attacks in wireless sensor networks**” submitted as a requirement for the course, **CSE302: COMPUTER NETWORKS** for B.Tech. is a bonafide record of the work done by **Shri. VARSHITH SAI NARAGAM (Reg. No.224003157, III B.Tech CSE**) during the academic year 2022-23, in the School of Computing.

Project Based Work *Viva voic*e held on ___________________

Examiner 1 **Examiner 2**

LIST OF FIGURES

ABBREVIATIONS

WSN	:	Wireless Sensor Network
NAM	:	Network Animator
RREQ	:	Route Request
RREP	:	Route Reply
BS	:	Base Station
E2ED	:	End -to -end delay
PDR	:	Packet Delivery Ratio
AODV	:	Ad-hoc On-demand Distance Vector
TCP	:	Transmission Control Protocol

ABSTRACT

A Wireless sensor network is a network of devices that can communicate the information gathered from a monitored field through wireless links. The data is forwarded through multiple nodes, and with a gateway, the data is connected to other networks like wireless Ethernet. There are different types of attacks against wireless sensor networks. Black hole is one of the possible attacks in wireless sensor networks.

These networks are used to monitor physical or environmental conditions like sound, pressure, temperature, and co-operatively pass data through the network to the main location.

A mobile ad hoc network MANET is a collection of mobile nodes in which the nodes can communicate without the need of any access point. The mobile hosts are free to move dynamically and act as routers. Security is a highly challenging issue in ad hoc networks. The presence of malicious nodes will affect the performance and reliability of the network.

The routing protocol is a set of rules and conventions that govern the movement of data within the network. In black hole attack, a malicious node uses its routing protocol to publicize itself for having the shortest route to the destination node.

Layer: Network Layer

Protocol: AODV

Frontend and Backend: NS-2

Keywords: MANET, Black Hole Attack, WSN, AODV, Routing Protocol.

NETWORK SIMULATOR (Ns2) AND NAM OVERVIEW

Network Simulator-2

NS2 stands for Network Simulator Version 2. It is an open-source event-driven simulator designed specifically for research in computer communication networks. NS-2 is a packet —level simulator and essentially a centric discrete event scheduler to schedule the events such as packet and timer expiration. Centric event scheduler cannot accurately emulate "events handled at the same time" in real world, that is, events are handled one by one. This is not a serious problem in most network simulations, because the events here are often transitory. Beyond the event scheduler, ns-2 implements a variety of network component and protocols. Notably, the wireless extension, derived from CMU Monarch Project, has 2 assumptions simplifying the physical world: Nodes do not move significantly over the length of time they transmit or receive a packet.

This assumption holds only for mobile nodes of high-rate and low-speed. Consider a node with the sending rate of 10kbps and moving speed of 10m/s, during its significantly and cause reception failure. Node velocity is insignificant compared to the speed of light. None of the provided models include Doppler effects, although they could.

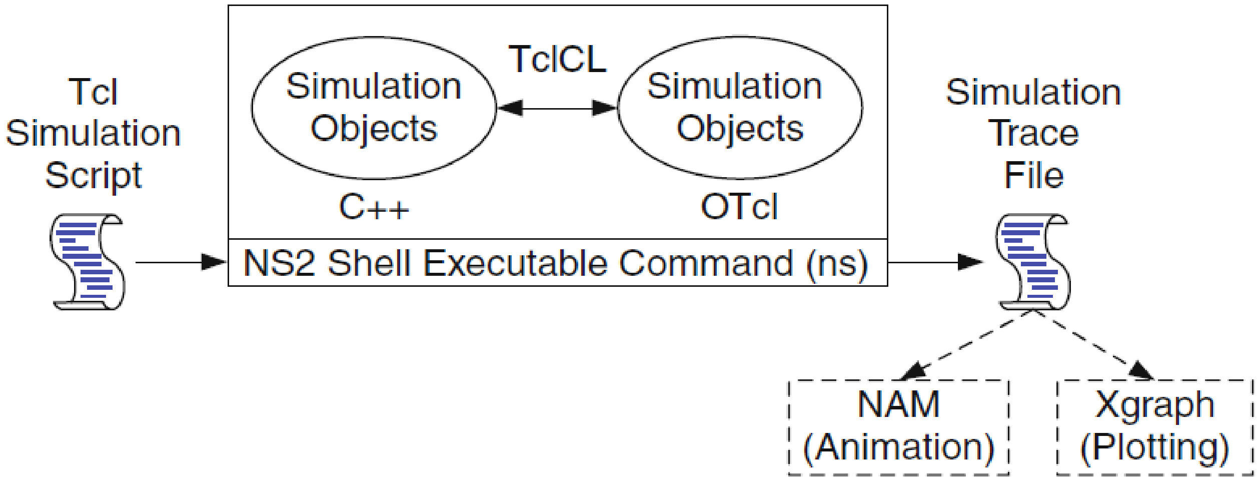
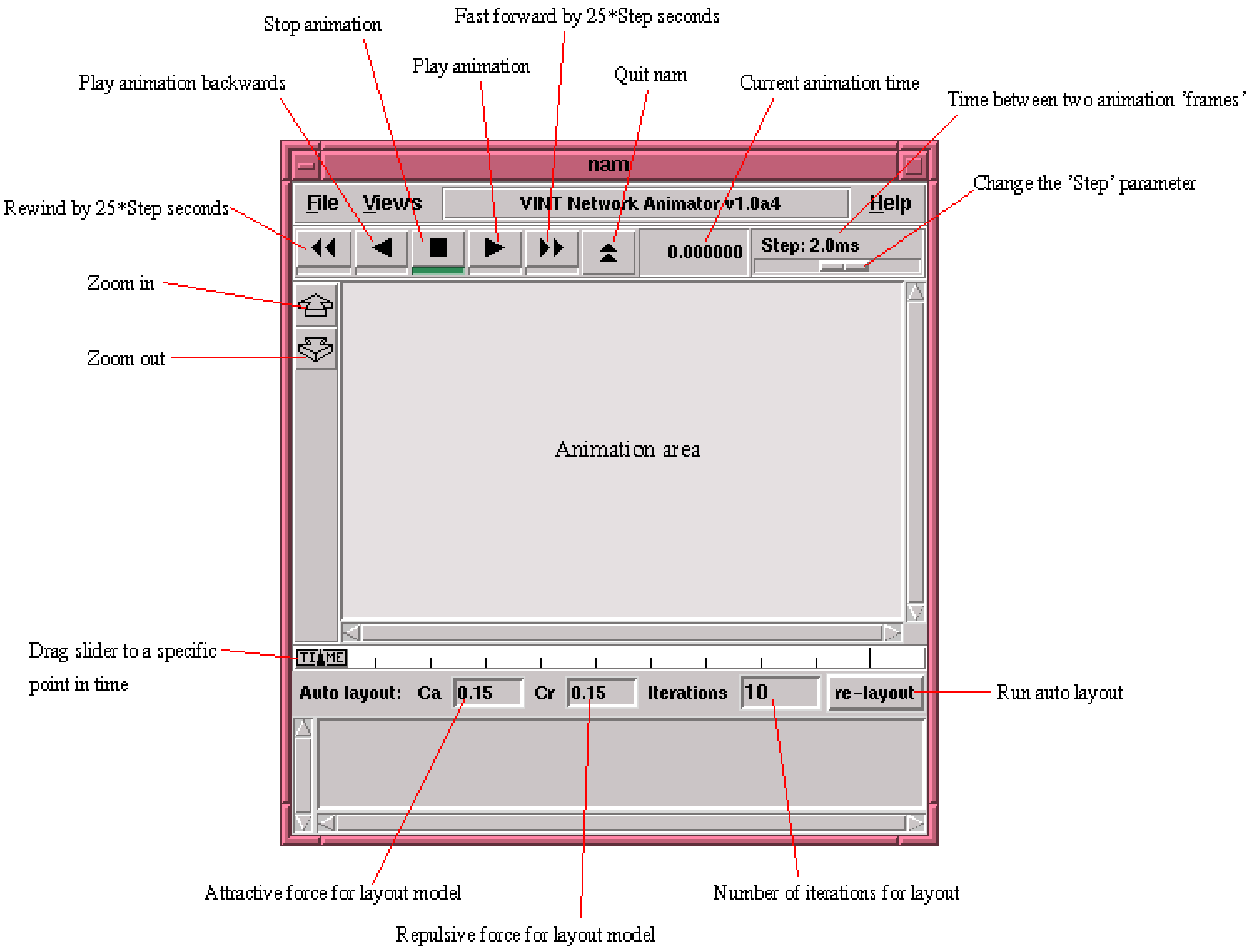


Fig 1 Working of NS2

NAM

NAM offers a visual depiction of the constructed network architecture. The following are its features shows the NAM application and all its elements. gives a visual representation of the built-in network. direct execution from a tcl script is possible. Play, stop, ff, rw, pause, a display speed controller, and a packet monitor facility are among the controls. It displays data on throughput and the quantity of packets sent across each link. gives topologies a drag-and-drop interface to be created.



Stop animation
Fast forward by 25*Step seconds
Play animation
Quit nam
Current animation time
Time between two animation 'frames'
Play animation backwards
Change the 'Step' parameter
Rewind by 25*Step seconds
Zoom in
Zoom out
nam
File
Views
VINT Network Animator v1.0a4
Help
0.000000
Step: 2.0ms
Animation area
TIME
Drag slider to a specific point in time
Auto layout:
Ca
0.15
Cr
0.15
Iterations
10
re-layout
Run auto layout
Attractive force for layout model
Number of iterations for layout
Repulsive force for layout model

Fig 2. NAM User interface

ADVANTAGES

1.Cheap- Does not require costly equipment.
2.Complex scenarios can be easily tested.
3.Results can be quickly obtained-more ideas can be tested in a smaller time frame.

DIASADVANTAGES

1.Real system too complex to model i.e. complicated structure.
2.Bugs are unreliable

SYSTEM REQUIREMENTS

Software Requirements

Operating system – UBUNTU
Simulation tool – NS-2
Other tools – NAM, Xgraph, Ns2 Trace File Analyzer, Virtual Box

Hardware Requirements

CPU type - Intel Pentium 4
RAM size – 512MB
Hard disk capacity– 80GB
Clock speed– 3.0 GHz

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Wireless Sensor Networks (WSN) has wide application in data gathering and data transmission as per the user’s requirement and it consist of number of nodes. These nodes have limited computational power, resources, and battery life. All these things together increase the security risks that WSN faces.

The black hole attack is one of the well-known, significant security hazards to wireless mobile ad hoc networks. Because the route-finding procedure is required and unavoidable, hackers exploit the Black hole to carry out their nefarious activities. Numerous researchers have used various detection methods to suggest various kinds of detection approaches. The trust connection between nodes is important in isolating the malicious nodes that root a black hole attack in the network. Even if it lacks accurate routing data, a malicious node (black hole node) may always reply positively to route requests. All packets that are routed to the black hole node may be dropped. In other words, Black Hole attack is one of the attacks that advertise it for having the shortest path to destination node and drops the entire packet that is coming from source node.

Malicious nodes will drop the packet during a black hole attack rather than sending it to its target destination. As a result, a black hole attack reduces the network's efficiency.

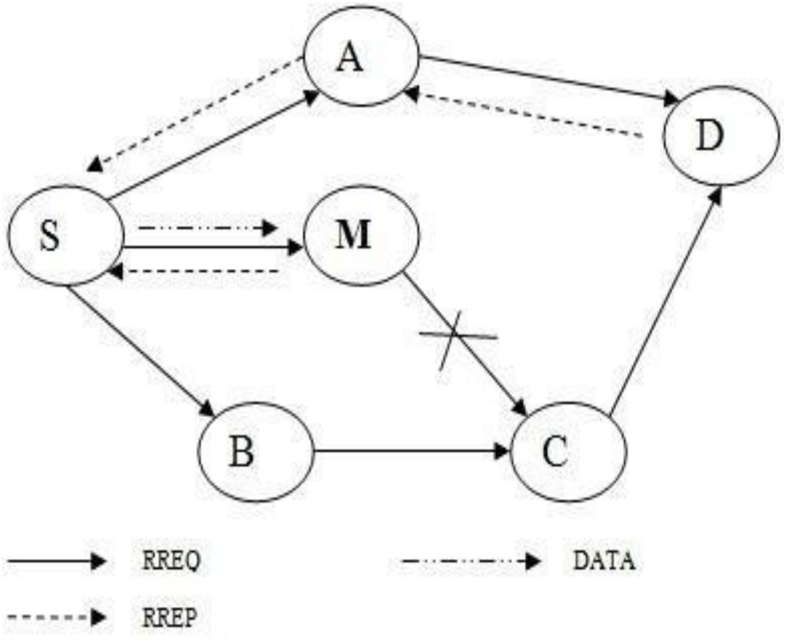


Fig 1.1 Black hole attack

1.1 AODV Routing Protocol

The AODV (Ad-Hoc On-Demand Distance Vector) is frequently used protocol in Wireless Sensor Network. According to demand, an automatic route is built. A node uses its Routing Table while sending a data packet to another node. If a new route is found, it will deliver data packets from source to destination. If it does not get a new route, the node initiates the Route Discovery Process.

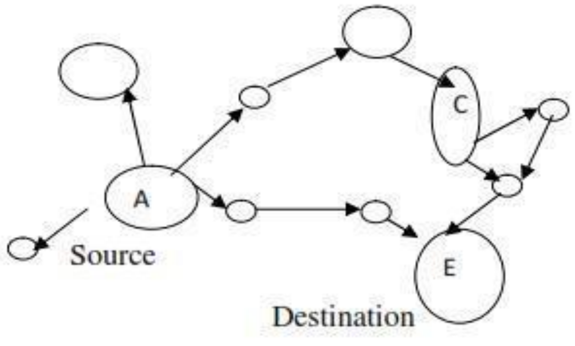
Route Request (RREQ) and Route Reply are the two control messages used in the AODV route finding process (RREP). Both control messages are used to identify the new route. The source node and destination node can exchange data packets after the route discovery process is finished.

1.2 Route Discovery Process

Route discovery is carried out by broadcasting the RREQ message. Every time a node wants to deliver data packets to a destination, it first examines the routing table to see if it already has a route. If not, the source node will start an RREQ and broadcast the request to all neighbours. Following that, nearby nodes will update their routing tables in response with the message received. The destination node will produce an RREP in response to RREQ once it reaches the destination. In order to update the route, the RREP will be routed back to the source of the RREQ. A gratuitous route reply, or RREP, is what an intermediate node can send in response to an RREQ if it has an active route to the destination. The RREP will be sent in reverse route of RREQ if a bidirectional link exists

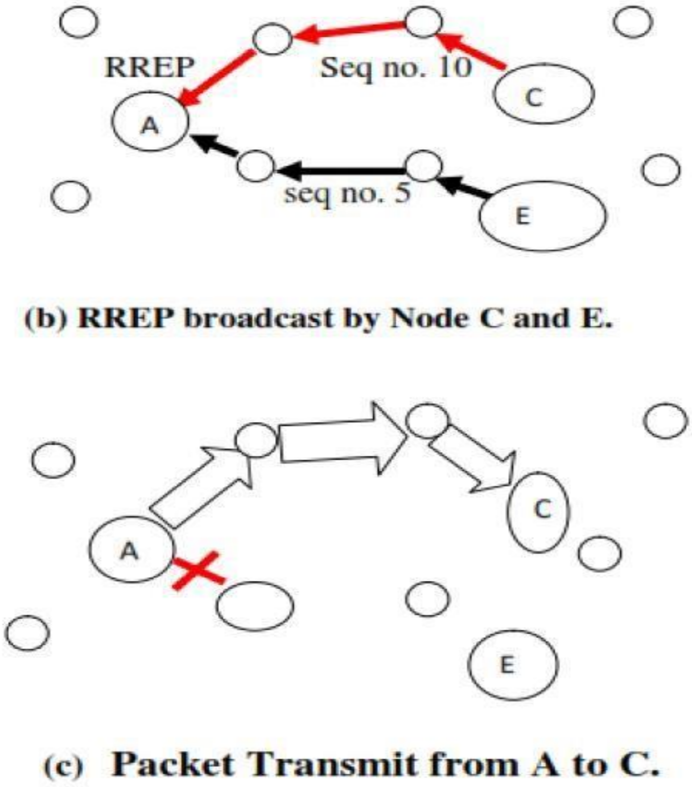
1.3 Black Hole Attacks

In black hole attack, a malicious node uses its routing protocol in order to publicize itself for having the shortest route to the destination node and attracts all the data traffic towards itself. It absorbs all packets and does not send them to their destination. The route discovery procedure is started by the source node broadcasting a Route Request (RREQ) packet to its neighbour. The next neighbour who receives the RREQ adds their address to it and forwards it further in the direction of the destination. The adversary node sends fictitious Route Reply (RREP) packet (with highest sequence number and least hop count) as a response to source node to pretend as a destination node. When the source node receives multiple responses, the sequence number of the RREPs received is compared. It chooses the path with the highest sequence number. If two RREPs have the same sequence number, the one with least hop count is used. The source node delivers all data packets to the adversary node because its RREP has the longest sequence. As a result, the source and destination nodes are unable to communicate. The fig 1(a) shows the Route discovery process initiated by node A. Node C is assumed as an adversary node and Node E as the destination node. In (b) shows the RREP as the response from destination node E as well as from adversary node C. All the data packets from A transmit to C as shown in (c).



RREQ

(a) RREQ broadcast by A



(b) RREP broadcast by Node C and E.

(c) Packet Transmit from A to C.

Fig 1.2 Process of black hole attack

In figure 1(a), the Node ‘A’ is the Source Node and ‘E’ is the Destination Node. When ‘A’ sends the data packets to ‘E’, it begins the route discovery process by broadcasting Route Request (RREQ) messages to the neighbouring nodes [8]. So that this message is received by the other nodes in the above picture.

In figure 1(b), assumed that Node ‘C’ is a malicious node. It directly sends out fictious Route Reply (RREP) message to Node ‘A’ with highest sequence number as well as the Node ‘C’ and ‘E’ also sends an actual Route Reply (RREP) message to Source Node ’A’ with sequence number.

In figure 1(c), malicious node drops the entire data packet instead of sending them to proper destination Node ‘E’. This describes in the figure 1(c).

This is the entire Black hole attack scenario as explain above in Figure 1(a), (b) and (c).

1.3.1 Steps Involved In Simulation Of Black Hole Attack

- Create a MANET.
- AODV routing protocol implementation.
- Insert nodes into the network.
- Introduce malicious(attacker) node into the network.
- Send packets from source to destination.
- Display simulation of nodes and packets in NAM.
- Evaluate performance metrics using analyzer.
- Generate graphs using Xgraph.

RSSI Approach

The received signal strength indicator (RSSI) measures the amount of power in a received radio transmission. The strength of the received signal is proportional to the distance between the sender and receiving nodes, which fluctuates due to various in-path interferences. This approach is useful in determining the path between sender and receiver or source and destination using signal strength.

1.3.2 Algorithm To Prevent Blackhole Attack In MANET

Notations:
SN: Source Node
IN: Intermediate Node
DN: Destination Node NHN: Next Hop Noda
FRq: Further Request FRp: Further Reply
Reliable Node: The mode through which the SN has routed data
DRI: Data Routing Information
ID: Identity of the node

```
SN broadcasts RREQ
SN receives RREP
IF (RREP is from DN or a reliable mode)
{
    Route data packets (Secure Route)
ELSE {
    Do {
        Send FRq and ID of IN to NHN
        Receive FRp, NHN of current NHN, DRI entry for
        NHN's next hop, DRI entry for current IN
        IF (NHN is a reliable node) {
            Check IN for black hole using DRI entry
            IF (IN is not a black hole)
                Route data packets (Secure Route)
            ELSE {
                Insecure Route
                IN is a black hole
                All the nodes along the reverse path from IN to the node
                that generated RREP are black holes
            }
        }
        ELSE
            Current IN= NHN
    }
}
While (IN is NOT a reliable node)
```

1.3.3 Scenario Of Packets In Black Hole Attack

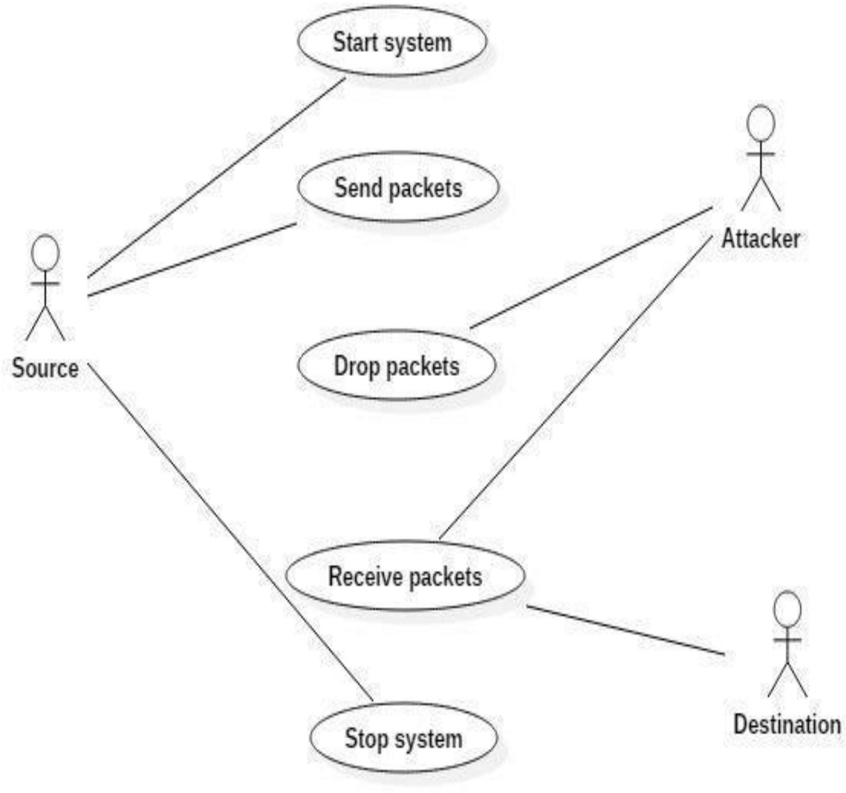
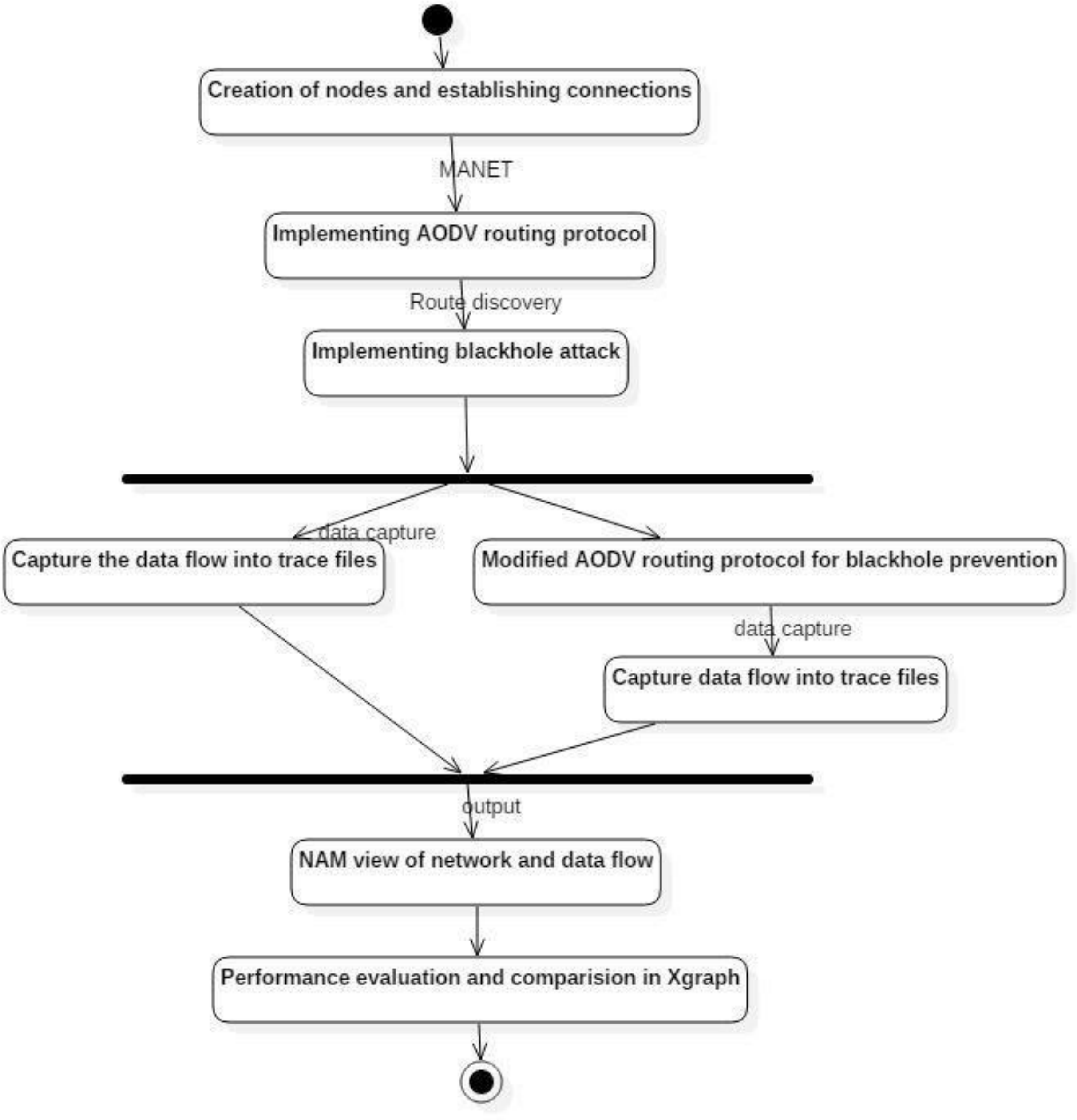


Fig1.3. Scenario of packets in Black hole attack

The flow diagram above uses three main factors: Source, Destination, and Attacker. The source initiates the system by sending RREQ packets to neighbouring nodes to determine the shortest path to the destination and then transmits packets to the destination. When the communication is over, the stop system is used to shut off the communication channel. Malicious node is the attacker in this network; it pretends to have the lowest hop count and the highest sequence number, which causes the source to accept its route reply message before it begins discarding packets and this results in starting a blackhole attack.

1.4 FLOW CHART



Creation of nodes and establishing connections
MANET
Implementing AODV routing protocol
Route discovery
Implementing blackhole attack
data capture
Capture the data flow into trace files
Modified AODV routing protocol for blackhole prevention
data capture
Capture data flow into trace files
output
NAM view of network and data flow
Performance evaluation and comparision in Xgraph

Fig 1.4. Activity diagram

The above flow chart shows that the start action is performed, followed by the creation of nodes and the establishment of connections between them, and the AODV routing protocol is implemented, followed by a blackhole attack and the generation of trace files in one phase of a fork, and the blackhole is prevented by modifying the AODV routing protocol and the generation of trace files in another phase of the fork. Finally, the findings are displayed in NAM and graphs are used to compare them.

CHAPTER 2

SOURCE CODE

```
#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) 50 ;# number of mobilenodes
#set val(r) 400 ;
set val(rp) AODV ;# routing protocol
#set val(rp) DSR ;# routing protocol
set val(x) 1000 ;
set val(y) 1000 ;
set val(stop) 150 ;

set ns [new Simulator]

set tf [open tf.tr w]
set ntf [open ntf.nam w]

$ns trace-all $tf
$ns namtrace-all-wireless $ntf 600 600

set topo [new Topography]

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

create-god $val(nn)
# Create node(0) "attached" to channel #1
# configure node, please note the change below.
$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
#-channelType $val(chan) \

#$ns_ initial_node_pos $n($i) 20
#for {set i 0} {$i <  [expr$val(nn)]} { incr i } {
for {set i 0} {$i <  $val(nn)} { incr i } {
set n($i) [$ns node]
 $n($i) color blue


}

#Initial Location of mobilenodes

$n(0) color red

$n(7) set X_ 30.0
$n(7) set Y_ 4.0
$n(7) set Z_ 0.0
$n(7) color red

$n(2) set X_ 91.6678
$n(2) set Y_ 280.592
$n(2) set Z_ 0.0

$n(6) set X_ 24.0326
$n(6) set Y_ 168.474
$n(6) set Z_ 0.0

$n(3) set X_ -65.9721
$n(3) set Y_ 79.1654
$n(3) set Z_ 0.0

$n(4) set X_ 171.98
$n(4) set Y_ 114.221
$n(4) set Z_ 0.0

$n(8) set X_ 98.0155
$n(8) set Y_ 206.64
$n(8) set Z_ 0.0

$n(10) set X_ 69.0
$n(10) set Y_ 6.0
$n(10) set Z_ 0.0

$n(11) set X_ 62.1082
$n(11) set Y_ 76.3292
```

```
$n(11) set Z_ 0.0

$n(12) set X_ 414.142
$n(12) set Y_ 165.638
$n(12) set Z_ 0.0

$n(13) set X_ 299.688
$n(13) set Y_ 120.55
$n(13) set Z_ 0.0

$n(14) set X_ -210.154
$n(14) set Y_ 168.24
$n(14) set Z_ 0.0

$n(15) set X_ -237.901
$n(15) set Y_ 127.487
$n(15) set Z_ 0.0

$n(16) set X_ 617.039
$n(16) set Y_ 107.544
$n(16) set Z_ 0.0

$n(17) set X_ 499.983
$n(17) set Y_ 135.291
$n(17) set Z_ 0.0

$n(18) set X_ -299.464
$n(18) set Y_ 232.403
$n(18) set Z_ 0.0

$n(19) set X_ -230.542
$n(19) set Y_ 13.03255
$n(19) set Z_ 0.0

$n(20) set X_ 141.879
$n(20) set Y_ 26.0387
$n(20) set Z_ 0.0

$n(21) set X_ -32.4034
$n(21) set Y_ 143.094
$n(21) set Z_ 0.0

$n(22) set X_ 998.4034
$n(22) set Y_ 243.094
$n(22) set Z_ 0.0

$n(32) set X_ 998.4034
$n(32) set Y_ 193.094
$n(32) set Z_ 0.0
```

```
$n(34) set X_ 98.4034
$n(34) set Y_ 151.094
$n(34) set Z_ 0.0

$n(33) set X_ 998.4034
$n(33) set Y_ 293.094
$n(33) set Z_ 0.0

$n(23) set X_ 538.134
$n(23) set Y_ 140.493
$n(23) set Z_ 0.0

$n(24) set X_ 247.633
$n(24) set Y_ 247.144
$n(24) set Z_ 0.0

$n(25) set X_ -105.0
$n(25) set Y_ 230.669
$n(25) set Z_ 0.0

$n(26) set X_ -15.0618
$n(26) set Y_ 191.651
$n(26) set Z_ 0.0

$n(27) set X_ 191.303
$n(27) set Y_ 220.264
$n(27) set Z_ 0.0

$n(28) set X_ 501.717
$n(28) set Y_ 71.1267
$n(28) set Z_ 0.0

$n(29) set X_ 268.473
$n(29) set Y_ 199.454
$n(29) set Z_ 0.0

$n(30) set X_ 307.491
$n(30) set Y_ 28.5873
$n(30) set Z_ 0.0

$n(31) set X_ 28.2921
$n(31) set Y_ 50.3169
$n(31) set Z_ 0.0

$n(35) set X_ -208.42
$n(35) set Y_ 22.5177
$n(35) set Z_ 0.0
```

```
$n(36) set X_ -232.699
$n(36) set Y_ 275.757
$n(36) set Z_ 0.0

$n(37) set X_ -276.052
$n(37) set Y_ 173.442
$n(37) set Z_ 0.0

$n(38) set X_ -192.813
$n(38) set Y_ 235.005
$n(38) set Z_ 0.0

$n(40) set X_ -96.5672
$n(40) set Y_ 147.43
$n(40) set Z_ 0.0

$n(41) set X_ -149.459
$n(41) set Y_ 8.69709
$n(41) set Z_ 0.0

$n(42) set X_ 104.595
$n(42) set Y_ 27.7728
$n(42) set Z_ 0.0

$n(44) set X_ -148.592
$n(44) set Y_ 53.7852
$n(44) set Z_ 0.0

$n(45) set X_ 306.624
$n(45) set Y_ 37.3107
$n(45) set Z_ 0.0

$n(46) set X_ 225.119
$n(46) set Y_ 203.79
$n(46) set Z_ 0.0

$n(47) set X_ -27.2009
$n(47) set Y_ 237.606
$n(47) set Z_ 0.0

$n(49) set X_ -118.79
$n(49) set Y_ 37.2844
$n(49) set Z_ 0.0

$n(1) set X_ -18.79
$n(1) set Y_ 37.2844
$n(1) set Z_ 0.0

$n(5) set X_ -28.79
```

```
$n(5) set Y_ 37.2844
$n(5) set Z_ 0.0

$n(9) set X_ -38.79
$n(9) set Y_ 37.2844
$n(9) set Z_ 0.0

$n(39) set X_ -67.79
$n(39) set Y_ 37.2844
$n(39) set Z_ 0.0

$n(43) set X_ 998.40
$n(43) set Y_ 151.094
$n(43) set Z_ 0.0

$n(48) set X_ 222.79
$n(48) set Y_ 37.2844
$n(48) set Z_ 0.0




$ns at 100.0 "$n(6) setdest 270.0 160.0 5.0"
$ns at 100.0 "$n(42) setdest 270.0 160.0 5.0"
$ns at 100.0 "$n(36) setdest 270.0 160.0 5.0"

$ns at 100.0 "$n(35) setdest 70.0 160.0 5.0"
$ns at 100.0 "$n(16) setdest 70.0 160.0 5.0"
$ns at 100.0 "$n(25) setdest 70.0 160.0 5.0"

$ns at 100.0 "$n(47) setdest 10.0 160.0 5.0"
$ns at 100.0 "$n(36) setdest 10.0 160.0 5.0"
$ns at 100.0 "$n(28) setdest 10.0 160.0 5.0"
$ns at 100.0 "$n(28) setdest 10.0 160.0 5.0"




set udp [new Agent/UDP]
set sink [new Agent/LossMonitor]
set vbr [new Application/Traffic/Exponential]

$ns attach-agent $n(6) $udp
$ns attach-agent $n(1) $sink
$vbr attach-agent $udp
$vbr set packetsize_ 2000
$vbr set idle_time_ 12ms
$vbr set burst time_ 200ms
```

```
$vbr set rate_ 100k

$ns connect $udp $sink
$ns at 30.0 "$vbr start"

##################################################################################
##MALICIOUS DETECTION AT MAC LAYER
##################################################################################
set hopcount 0

proc algorithm {} {

        global h val node_ r ns hopcount router

set now [$ns now]

set time 1.0


set count($hopcount) 0

        for {set j 0} {$j < $val(nn) } { incr j } {
                # $h($hopcount-$j)=intrusion detection
                if {$h($hopcount-$j) < 12} {

                        set count($hopcount) [expr $count($hopcount)]
                        set n($hopcount-$count($hopcount)) $j
}
# $h($hopcount-$j)=intrusion prevention
                if {$h($hopcount-$j) >12} {
                        set count($hopcount) [expr $count($hopcount)]
                        set n($hopcount-$count ($hopcount)) $j
}
}
}

set tcp15 [new Agent/TCP/Newreno]
$tcp15 set maxcwnd_ 15
$tcp15 set fid_ 4
set sink15 [new Agent/TCPSink]
$ns attach-agent $n(26) $tcp15
$ns attach-agent $n(8) $sink15
$ns connect $tcp15 $sink15
set ftp15 [new Application/FTP]
$ftp15 attach-agent $tcp15
$n(46) color red
# $ns at 15.0 "$n(46) color red"
$ns at 0.0 "$n(8) label ATTACKER"
$ns at 0.0 "$ftp15 start"
```

```
$ns at 20.0 "$ftp15 stop"
#$ns at 110.0 "$n(6) setdest 283.0 160.0 5.0"

# Set a TCP connection between n(26) and n(21)

set tcp14 [new Agent/TCP/Newreno]
$tcp14 set maxcwnd_ 16
$tcp14 set fid_ 4
set sink14 [new Agent/TCPSink]
$ns attach-agent $n(26) $tcp14
$ns attach-agent $n(2) $sink14
$ns connect $tcp14 $sink14
set ftp14 [new Application/FTP]
$ftp14 attach-agent $tcp14
$ns at 21.0 "$ftp14 start"
$ns at 130.0 "$ftp14 stop"
$ns at 2.0 "$n(21) color cyan"
$ns at 2.0 "$n(21) label GWReady"

#######################################################################
## MALICIOUS DETECTION AT NETWORK LAYER
#######################################################################


set pathselection 0


proc algorithm {} {

        global h val node_r ns pathselection router

set now [$ns now]

set time 1.0


set count ($pathselection) 0

        for {set j 0) {$j < $val(nn) } { incr j } {
                # Sh($pathselection-$j)=intrusion detection
                if ($h($pathselection-$j)< j} {

                        set count ($pathselection) [expr $count ($pathselection)]
                        set n($pathselection-$count($pathselection)) $j


al=q1
a2=q2
a3=q3
N=h
No packets counted =count ($pathselection);
Node=nn
S=pathselection
Q=qx
B=p
L=L1


W=a1*N+(-a2*stability+a3*load/N
}
        #$h($pathselection-$j)=intrusion prevention
                if {$h($pathselection-$j) >j} {

                        set count($pathselection) [expr $count($pathselection)]
                        set n($pathselection-$count ($pathselection)) $j
}
}
}
set tcp21 [new Agent/TCP/Newreno]
$tcp21 set maxcwnd_ 21
$tcp21 set fid_ 4
set sink21 [new Agent/TCPSink]
$ns attach-agent $n(27) $tcp21
$ns attach-agent $n(8) $sink21
$ns connect $tcp21 $sink21
set ftp21 [new Application/FTP]
$ftp21 attach-agent $tcp21
$ns at 0.0 "$ftp21 start"
$ns at 20.0 "$ftp21 stop"
$ns at 2.0 "$n(8) color cyan"
$ns at 2.0 "$n(8) label ATTACKER"
$ns at 21.0 "$n(8) color red"

# Set a TCP connection between n(26) and n(2)

set tcp16 [new Agent/TCP/Newreno]
$tcp16 set naxcwnd_ 16
$tcp16 set fid_ 4
set sink16 [new Agent/TCPSink]
$ns attach-agent $n(27) $tcp16
$ns attach-agent $n(2) $sink16
$ns connect $tcp16 $sink16
set ftp16 [new Application/FTP]
$ftp16 attach-agent $tcp16
$ns at 20.0 "$ftp16 start"
$ns at 130.0 "$ftp16 stop"

# Set a TCP connection between n(46) and n(2)
```

```
set tcp17 [new Agent/TCP/Newreno]
$tcp17 set maxcwnd_ 16
$tcp17 set fid_ 4
set sink17 [new Agent/TCPSink]
$ns attach-agent $n(23) $tcp17
$ns attach-agent $n(28) $sink17
$ns connect $tcp17 $sink17
set ftp17 [new Application/FTP]
$ftp17 attach-agent $tcp17
$ns at 0.0 "$ftp17 start"
$ns at 20.0 "$ftp17 stop"
$ns at 2.0 "$n(23) color cyan"
$ns at 2.0 "$n(23) label ATTACKER"
$ns at 21.0 "$n(23) color red"

##################################################################
## MALICIOUS DETECTION AT PHYSICAL LAYER
##################################################################

set RSSI 1


proc algorithm {} {

        global h val node_ r ns RSSI

set now [$ns now]

set time 1.0


set count($RSSI) O

        for {set j 0} {$j < $val(nn) } { incr j} {
                # $h($RSSI-$j)=intrusion detection
                if {$h($RSSI-$j) < q} {

                        set count ($RSSI) [expr $count($RSSI)]
                        set n($RSSI-$count($RSSI)) $j

txpower=j;
dist bw sen and rec=RSSI ;
n=1;
rxd power=txpower*(1/dist bw sen and rec)^n;


}
        #$h($RSSI -$j)=intrusion prevention
        if {$h($RSSI-$j) >q} {

                set count($RSSI) [expr $count($RSSI)]
                set n($RSSI-$count($RSSI)) $j
}
}
}

set tcp18 [new Agent/TCP/Newreno]
$tcp18 set maxcwnd_ 18
$tcp18 set fid_ 4
set sink18 [new Agent/TCPSink]
$ns attach-agent $n(16) $tcp18
$ns attach-agent $n(23) $sink18
$ns connect $tcp18 $sink18
set ftp18 [new Application/FTP]
$ftp18 attach-agent $tcp18
$ns at 0.0 "$ftp18 start"
$ns at 20.0 "$ftp18 stop"

set tcp19 [new Agent/TCP/Newreno]
$tcp19 set maxcwnd_ 19
$tcp19 set fid_ 4
set sink19 [new Agent/TCPSink]
$ns attach-agent $n(17) $tcp19
$ns attach-agent $n(28) $sink19
$ns connect $tcp19 $sink19
set ftp19 [new Application/FTP]
$ftp19 attach-agent $tcp19
$ns at 20.0 "$ftp19 start"
$ns at 130.0 "$ftp19 stop"

set tcp20 [new Agent/TCP/Newreno]
$tcp20 set maxcwnd_ 20
$tcp20 set fid_ 4
set sink20 [new Agent/TCPSink]
$ns attach-agent $n(17) $tcp20
$ns attach-agent $n(16) $sink20
$ns connect $tcp20 $sink20
set ftp20 [new Application/FTP]
$ftp20 attach-agent $tcp20
$ns at 22.0 "$ftp20 start"
$ns at 130.0 "$ftp20 stop"

set tcp22 [new Agent/TCP/Newreno]
$tcp22 set maxcwnd_ 22
$tcp22 set fid_ 4
set sink22 [new Agent/TCPSink]
$ns attach-agent $n(35) $tcp22
```

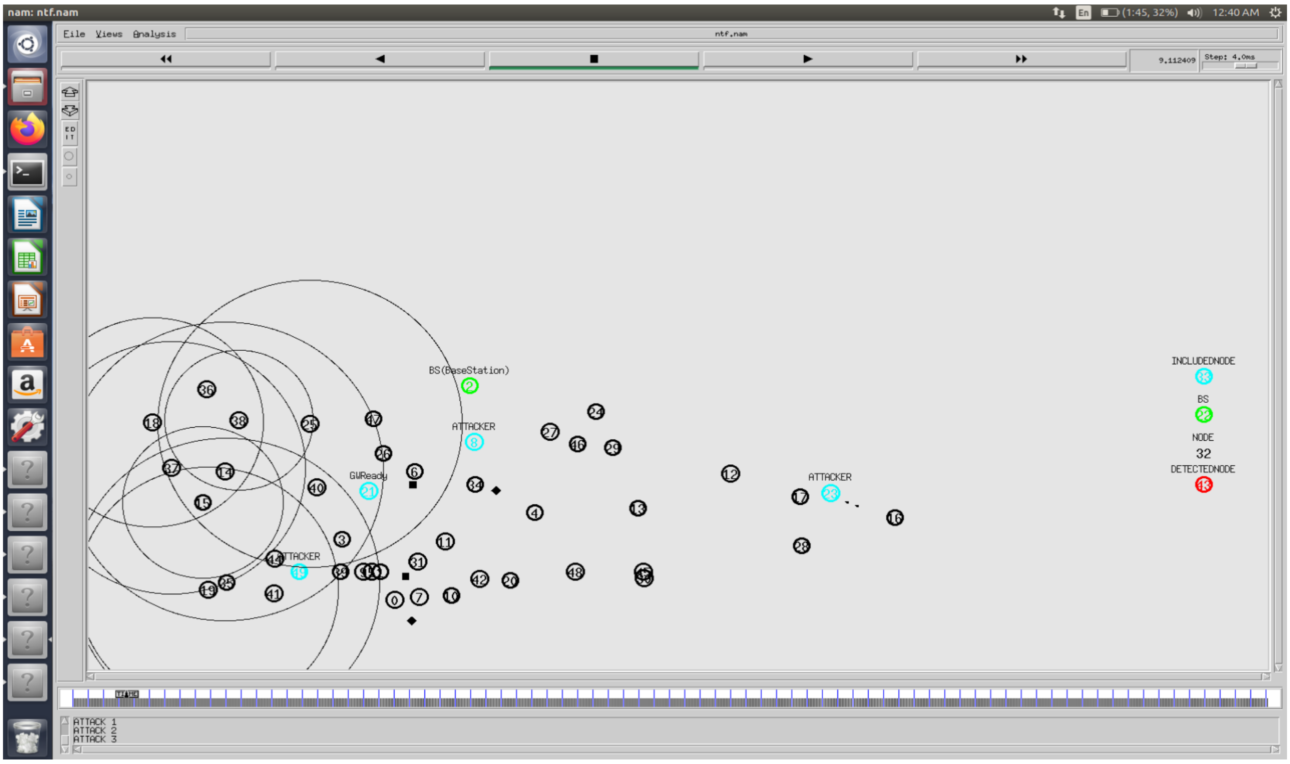
```
$ns attach-agent $n(49) $sink22
$ns connect $tcp22 $sink22
set ftp22 [new Application/FTP]
$ftp22 attach-agent $tcp22
$ns at 0.0 "$ftp22 start"
$ns at 20.0 "$ftp22 stop"

set tcp23 [new Agent/TCP/Newreno]
$tcp23 set maxcwnd_ 23
$tcp23 set fid_ 4
set sink23 [new Agent/TCPSink]
$ns attach-agent $n(41) $tcp23
$ns attach-agent $n(49) $sink23
$ns connect $tcp23 $sink23
set ftp23 [new Application/FTP]
$ftp23 attach-agent $tcp23
$ns at 0.0 "$ftp23 start"
$ns at 20.0 "$ftp23 stop"

set tcp24 [new Agent/TCP/Newreno]
$tcp24 set maxcwnd_ 24
$tcp24 set fid_ 4
set sink24 [new Agent/TCPSink]
$ns attach-agent $n(41) $tcp24
$ns attach-agent $n(44) $sink24
$ns connect $tcp24 $sink24
set ftp24 [new Application/FTP]
$ftp24 attach-agent $tcp24
$ns at 21.0 "$ftp24 start"
$ns at 130.0 "$ftp24 stop"
$ns at 2.0 "$n(49) color cyan"
$ns at 2.0 "$n(49) label ATTACKER"
$ns at 21.0 "$n(49) color red"

set tcp25 [new Agent/TCP/Newreno]
$tcp25 set maxcwnd_ 25
$tcp25 set fid_ 4
set sink25 [new Agent/TCPSink]
$ns attach-agent $n(15) $tcp25
$ns attach-agent $n(35) $sink25
$ns connect $tcp25 $sink25
set ftp25 [new Application/FTP]
$ftp25 attach-agent $tcp25
$ns at 0.0 "$ftp25 start"
$ns at 130.0 "$ftp25 stop"

#$ns at 0.03 "$ns trace-annotate \"NODE2 ACTS AS BASE STATION\""
#$ns at 0.05 "$ns trace-annotate\"CLUSTER FORMATION\""
$ns at 0.203 "$ns trace-annotate \"NODE 6 TRANSFERS DATA TO NODE 30\""
```

```
$ns at 1.6 "$ns trace-annotate \"NODE 35 TRANSFERS DATA TO NODE 49\""
$ns at 2.1 "$ns trace-annotate \"NODE 8,23 AND 40 ARE THE ATTACKER NODES\""
$ns at 2.4 "$ns trace-annotate \"ATTACK 1\""
$ns at 2.6 "$ns trace-annotate \"ATTACK 2\""
$ns at 2.7 "$ns trace-annotate \"ATTACK 3\""
$ns at 22 "$ns trace-annotate \"ATTACKER NODES ARE DETECTED AND PREVENTED\""

$ns at 0.0 "$n(22) label BS"
$ns at 0.0 "$n(22) color green"
$ns at 0.0 "$n(32) label NODE"
$ns at 0.0 "$n(32) color block"
$ns at 0.0 "$n(33) label INCLUDEDNODE"
$ns at 0.0 "$n(33) color cyan"
$ns at 0.0 "$n(43) label DETECTEDNODE"
$ns at 0.0 "$n(43) color red"


#Defining heads
$n(2) color green
        $ns at 0.0 "$n(2) color green"

$ns at 1.0 "$n(2) label BS(BaseStation)"

#####################################################################
## ##ENERGY CONSUMPTION
#####################################################################

proc findEnergyConsumption {} {
        global ns node_ val IE energyConsumption r FE
set energy1 [open ACTIVENODE_$val(nn).tr w]
set energy2 [open ACTIVENODE REDUCE.xg w]

        for {set i 0} {$i < $val(nn) } { incr i } {
                 set FE($i) [$node_($i) energy]
          puts $r "FE($i)=$FE($i)"

        }
        for {set i 0} {$i < $val(nn) } { incr i } {
                 set CE($i) [expr $IE($i) - $FE($i)]
        }

        set energyConsumption 0
        for {set i 0} {$t < $val(nn) } { incr i } {
                 set energyConsumption [expr $CE($i) + $energyConsumption]
        }
        puts $energy1 "Energy Consumption = $energyConsumption J"
        puts $energy2 "$val(nn) $energyConsumption"
}
```

```
# Define node initial position in nam
# 20 defines the node size for nam
for {set i 0} {$i < $val(nn)} { incr i } {

$ns initial_node_pos $n($i) 20
}

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

set f1 [open f1.tr w]
set f2 [open f2.tr w]
set f3 [open f3.tr w]
set f4 [open f4.tr w]
set f5 [open f5.tr w]
set f6 [open f6.tr w]
set f7 [open f7.tr w]

set a1 [open a1.tr w]
set a2 [open a2.tr w]
set a3 [open a3.tr w]
set a4 [open a4.tr w]
set a5 [open a5.tr w]
set a6 [open a6.tr w]
set a7 [open a7.tr w]

proc record {} {

global ns f1 f2 f3 f4 f5 f6 f7 sink
global f1 f2 f3 f4 f5 f6 f7 a1 a2 a3 a4 a5 a6 a7
#global ns tf ntf bs1 bs2 pt pt1 p p1 d d1 q q2
set time 1
set now [$ns now]

set bw1 [$sink set bytes_]
set bw2 [$sink set npkts_]
set bw3 [$sink set lastPktTime_]

puts $a1 "$now [expr $bw1/$time*.01/5]"
puts $a2 "$now [expr $bw1/$time*.05/5]"
puts $a5 "$now [expr $bw1/$time*.09/5]"
puts $a6 "$now [expr $bw3/$time*.001/5]"
puts $a3 "$now [expr $bw3/$time*.005/5]"
puts $a4 "$now [expr $bw3+$time*.003/5]"
puts $a7 "$now [expr $bw3+$time*6/5]"

$sink set bytes_ 1
$sink set npkts_ 0
$sink set nlost_ 0

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

}

proc finish {} {

global ns tf ntf a1 a2 a3 a4 a5 a6 a7
#global ns tf ntf bs1 bs2 pt pti p pi d di q q2
$ns flush-trace
close $tf
close $ntf
close $a1
close $a2
close $a3
close $a4
close $a5
close $a6
close $a7
exec nam ntf.nam &

#exec xgraph a1.tr a2.tr a3.tr a4.tr -t "energy consumption over the percentage of detected
intruder"-x "percentage of detected intruder" -y "energy Consumption"-m &
#exec xgraph  eab.tr esb.tr ehb.tr ecb.tr -t  &
exec xgraph  dab.tr dsb.tr dhb.tr dcb.tr &
exec xgraph  oab.tr osb.tr ohb.tr ocb.tr &
exec xgraph  bab.tr bsb.tr bhb.tr bcb.tr &
exec xgraph  rab.tr rsb.tr rhb.tr rcb.tr &
exit 0
}

$ns at 8.0 "$vbr start"
$ns at 0.3 "record"

$ns at 200.0 "$vbr stop"
$ns at 200.1 "finish"
$ns run
```

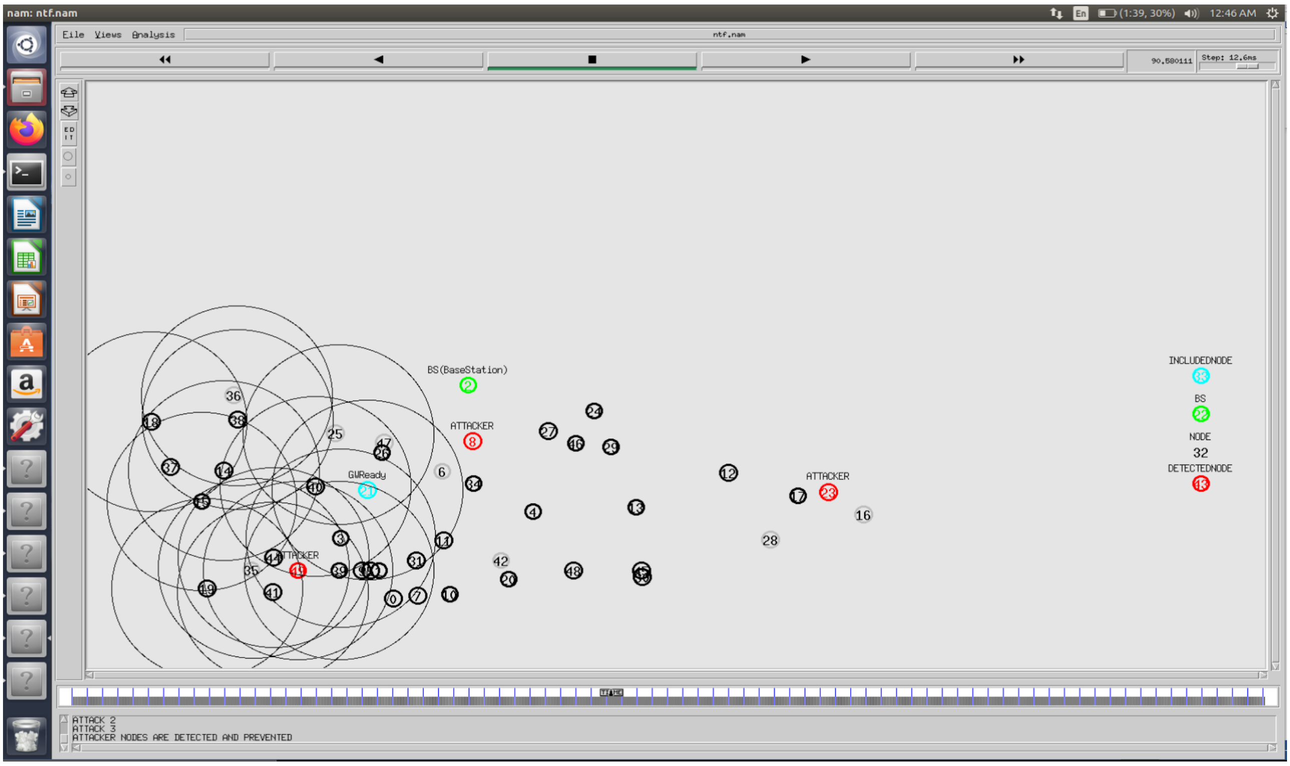
CHAPTER 3

SIMULATION RESULTS



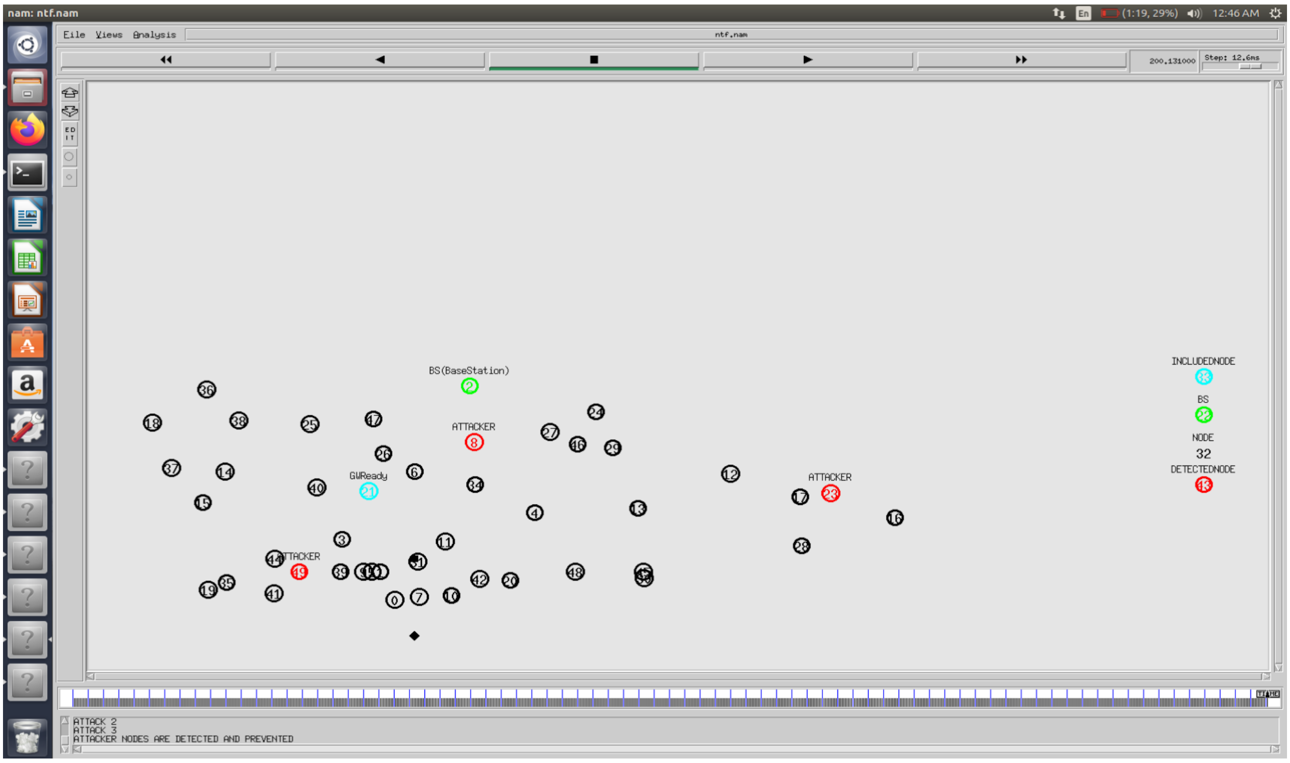
```
ATTACK 1
ATTACK 2
ATTACK 3
```

Fig 3.1 Simulation of nodes at time interval 9.1 seconds



```
ATTACK 2
ATTACK 3
ATTACKER NODES ARE DETECTED AND PREVENTED
```

Fig 3.2 Simulation of nodes at time interval 90.6 seconds



```
ATTACK 2
ATTACK 3
ATTACKER NODES ARE DETECTED AND PREVENTED
```

Fig 3.3 Simulation of nodes at time interval 200 seconds

CHAPTER 4

PERFORMANCE EVALUATION METRICS

4.1 Graphs

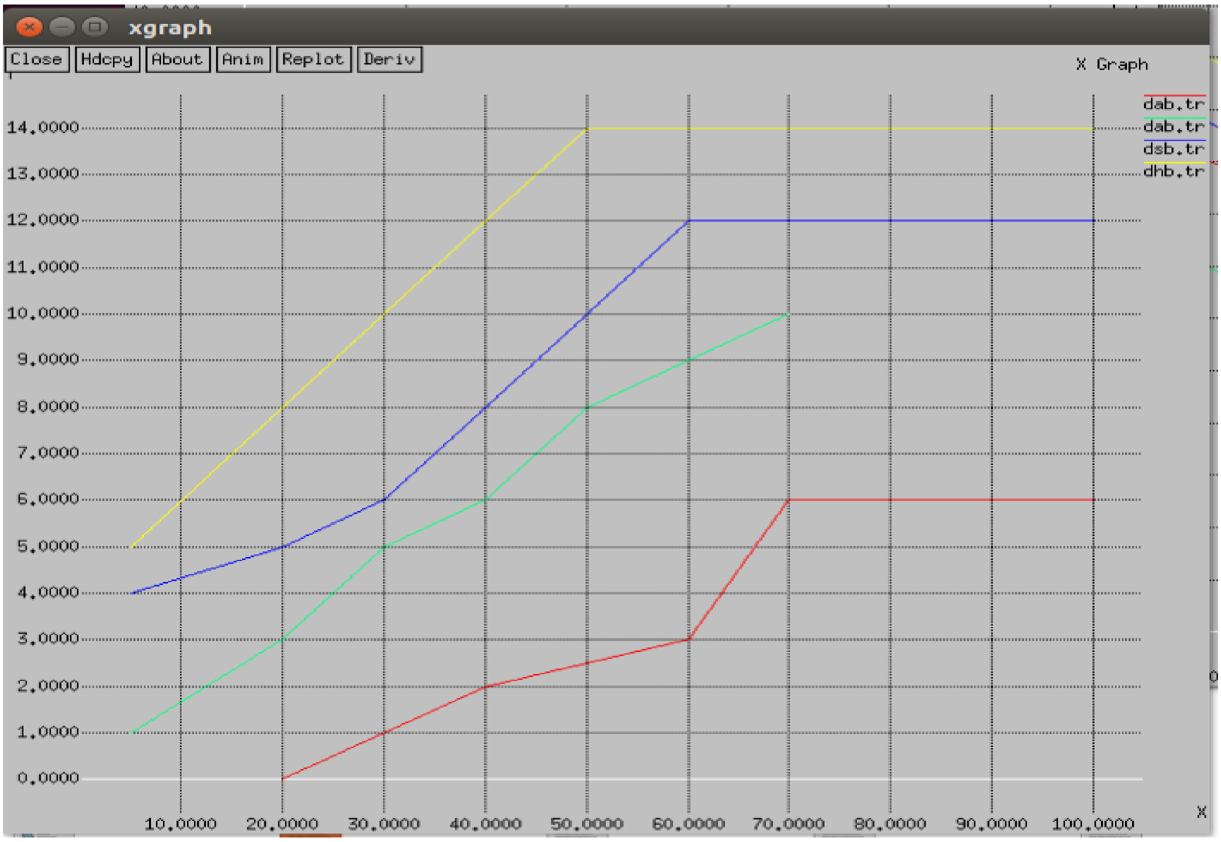
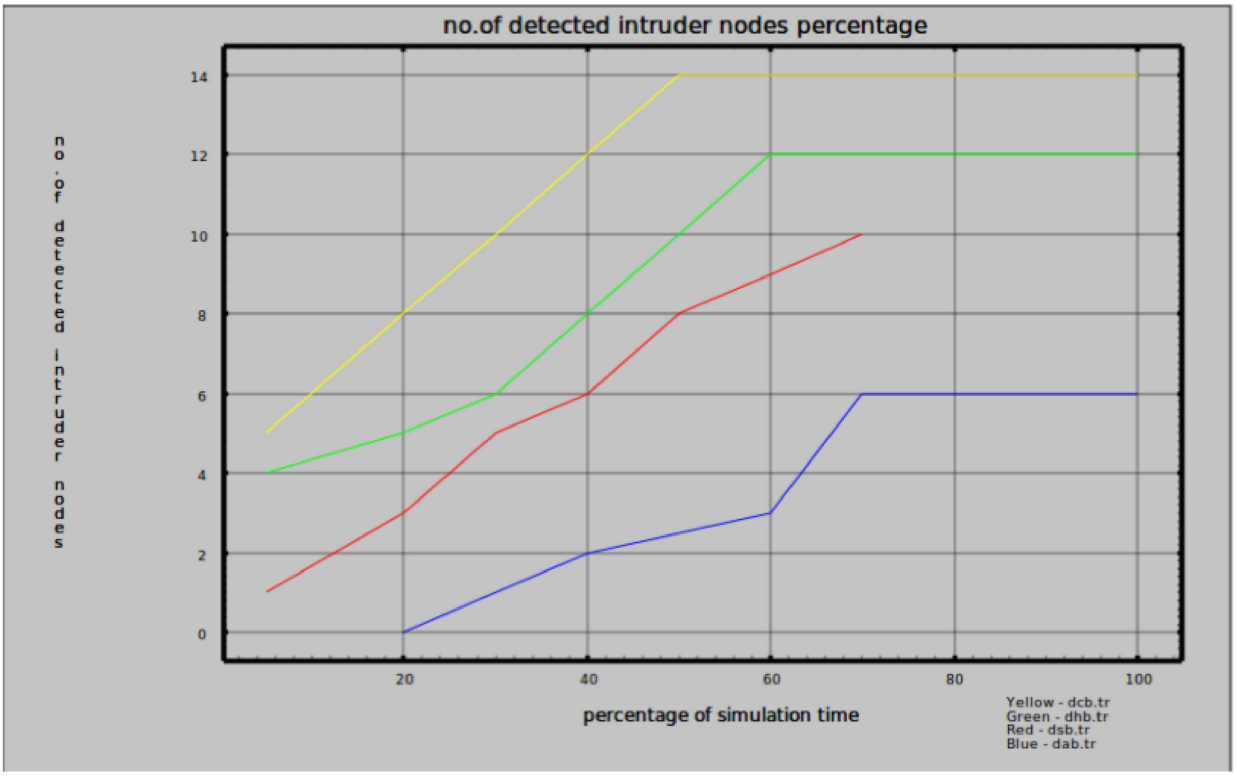


Fig 4.1 Graph for Simulation time vs No.of detected intruder nodes percentage



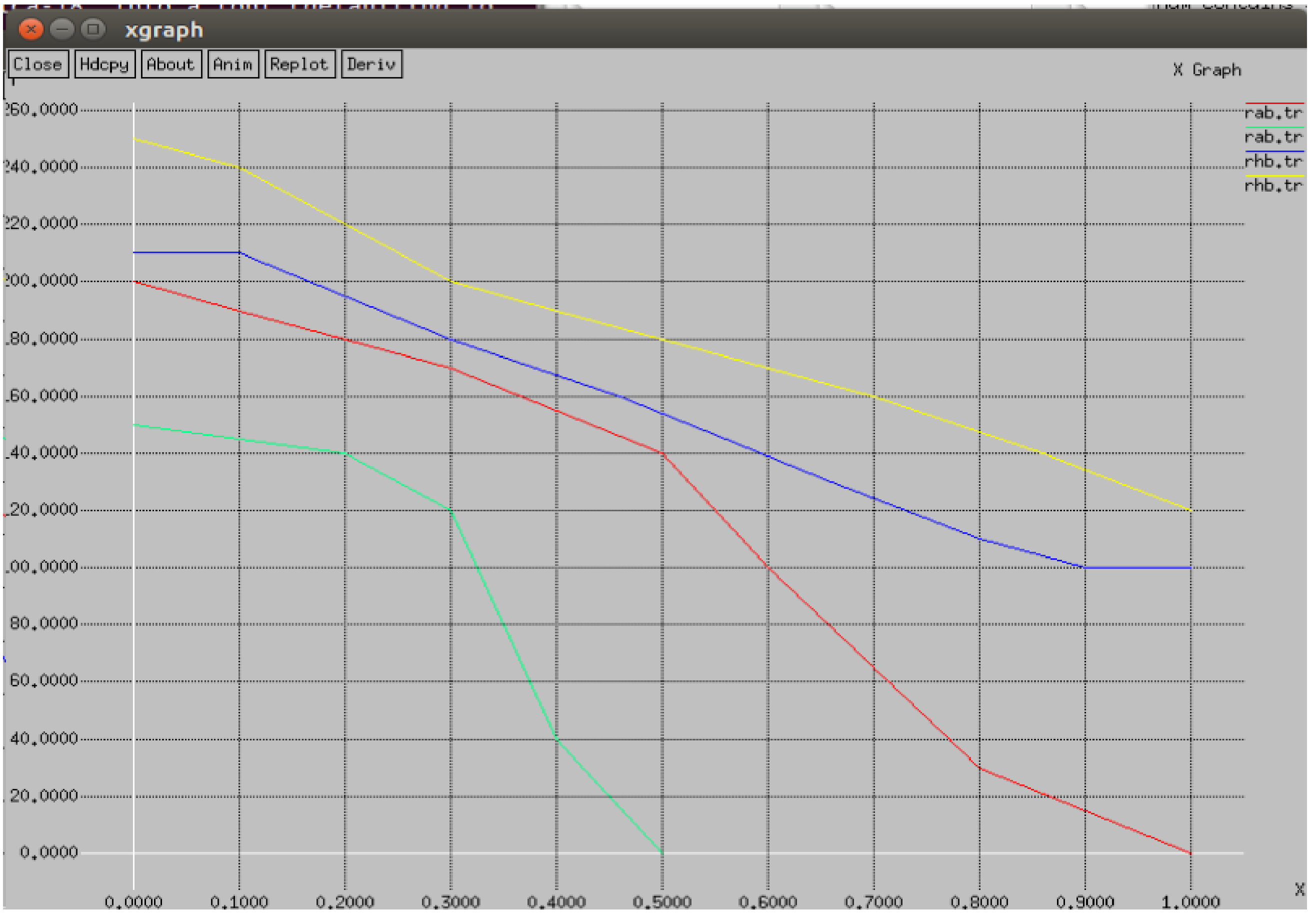
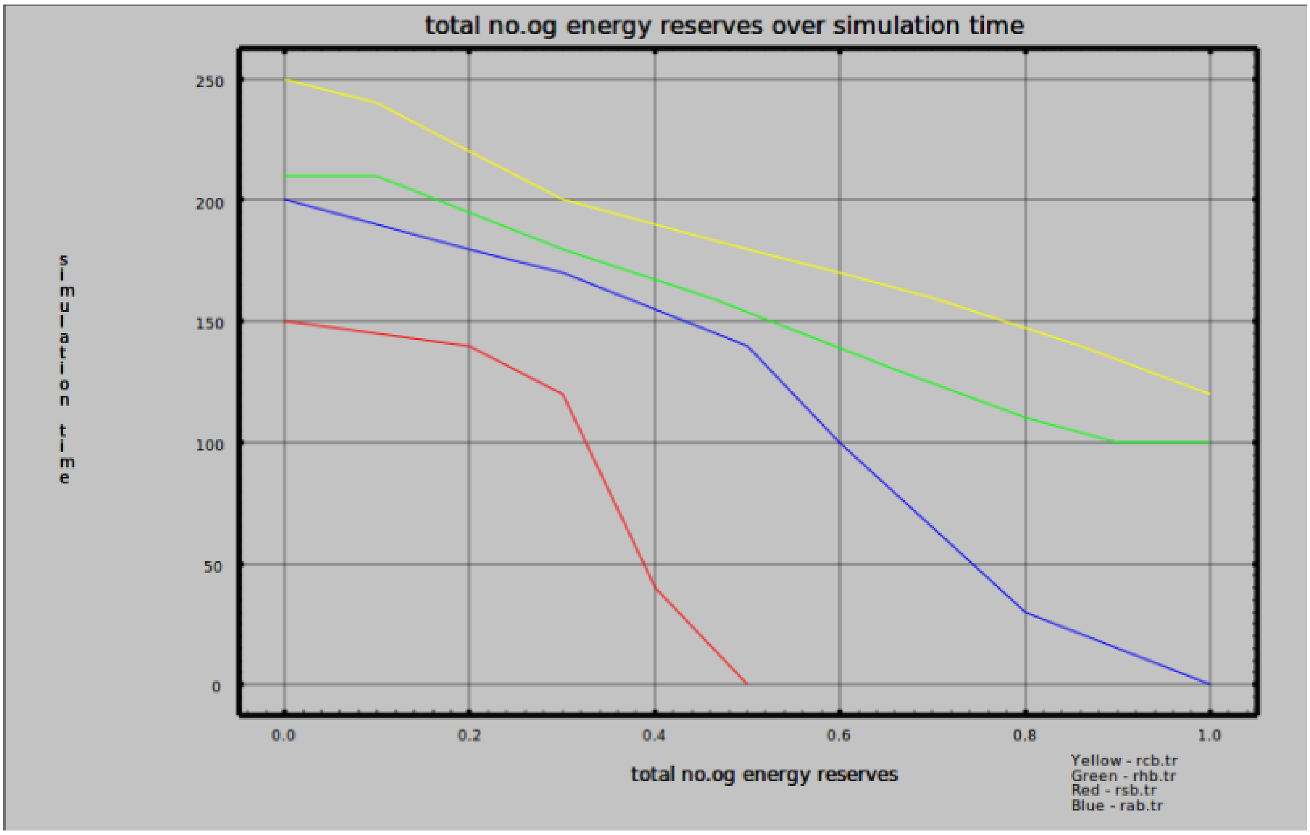


Fig 4.2 Graph for Total no.of energy reserves vs Simulation time



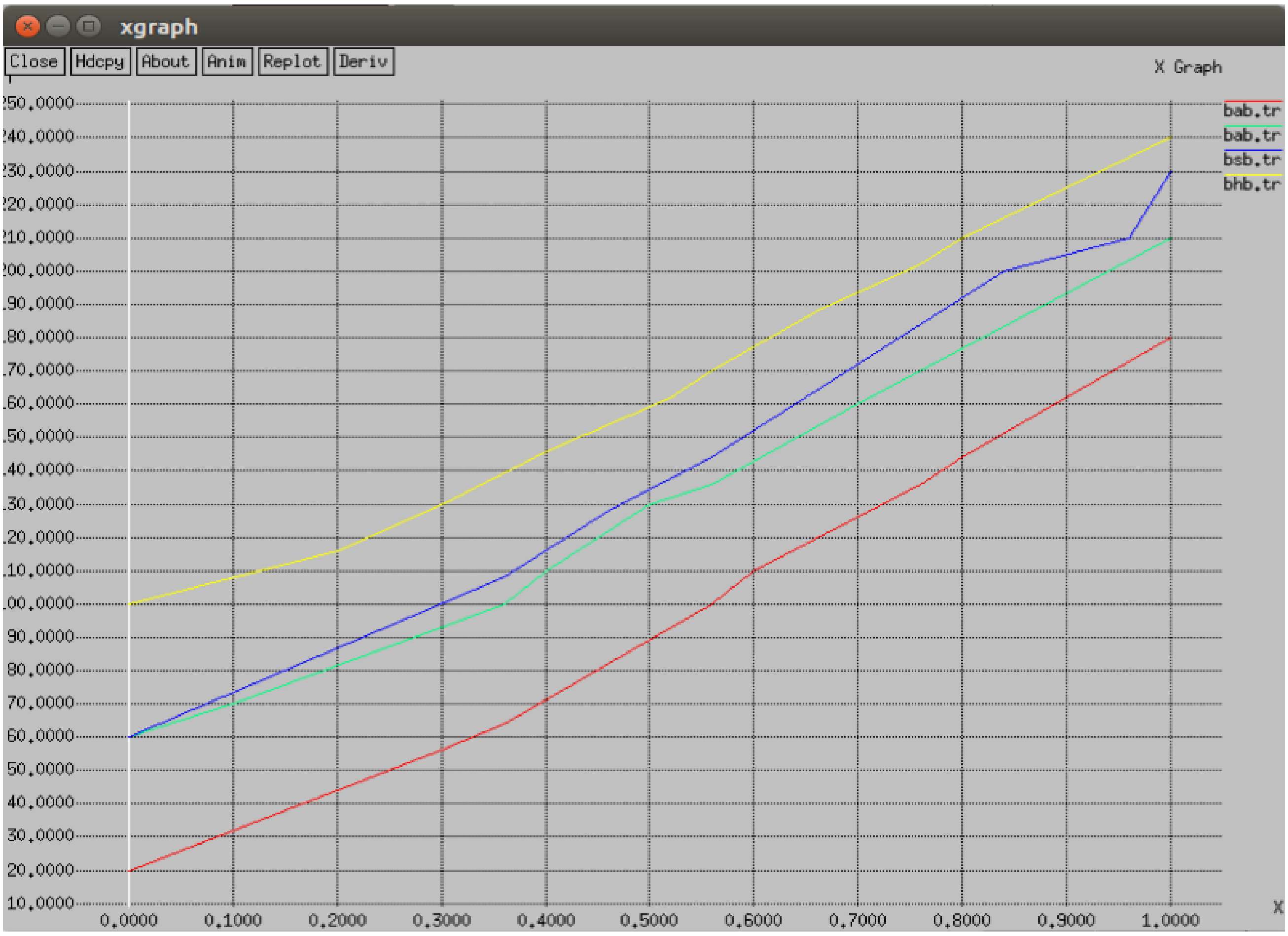
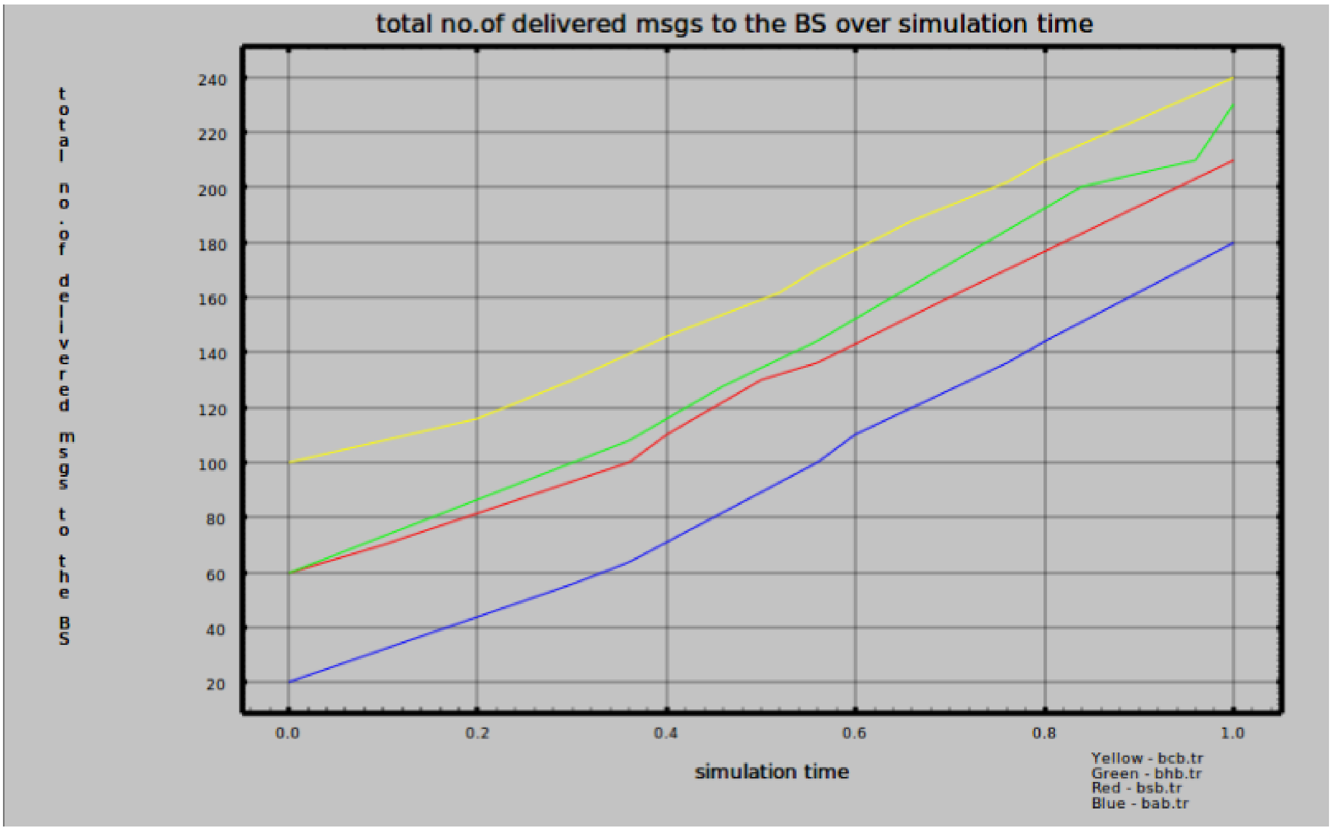


Fig 4.3 Graph for Simulation time vs Total of delivered message to the Base Station



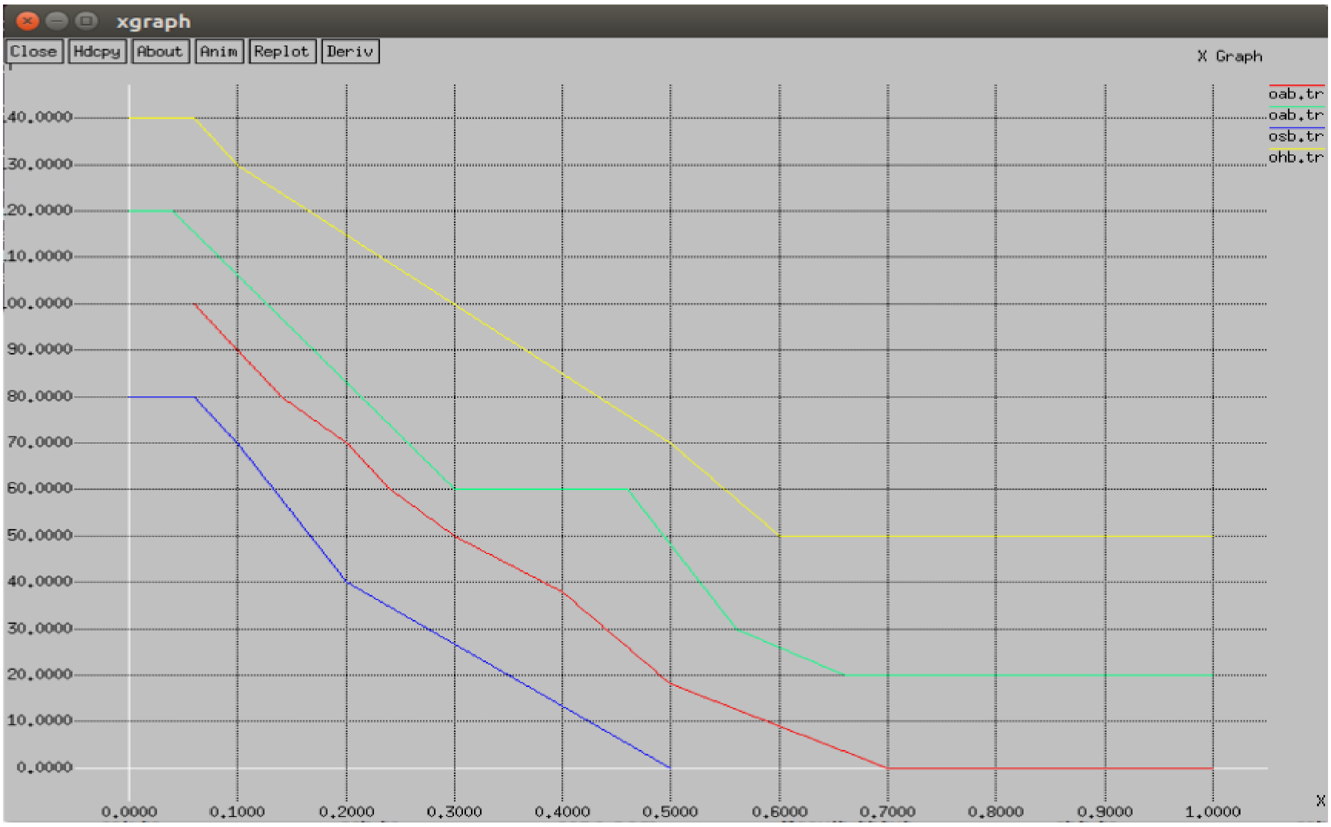
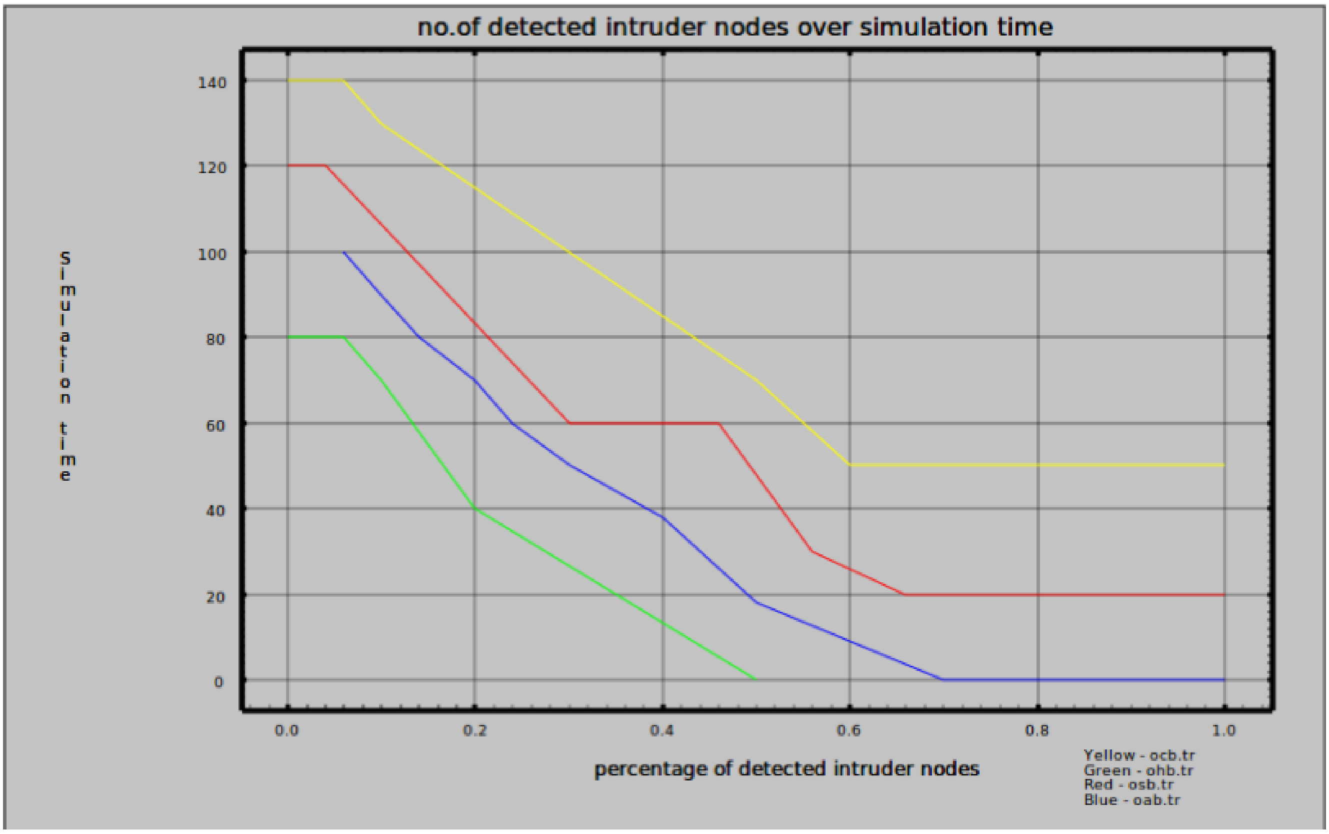
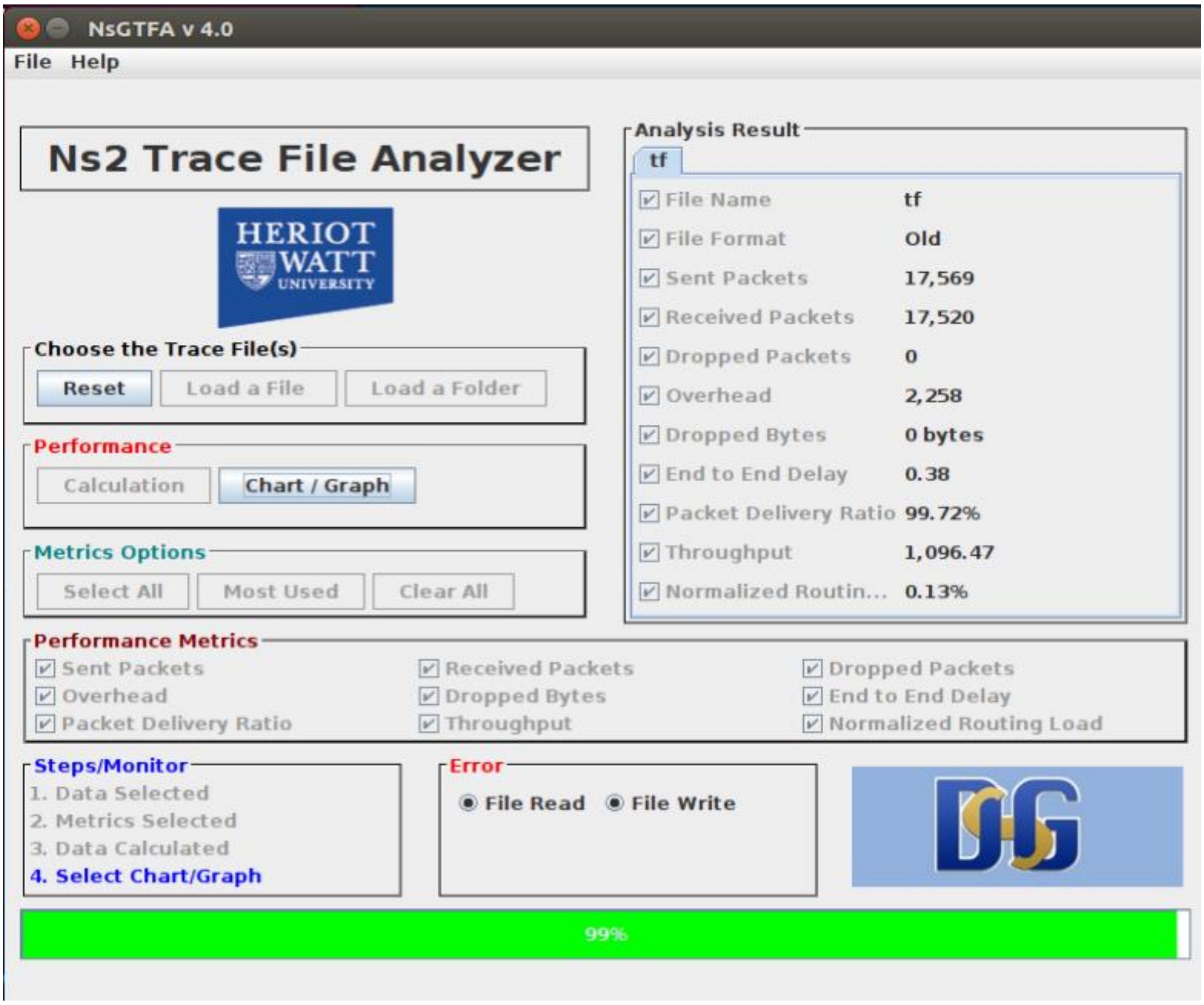


Fig 4.4. Graph for Percentage of detected intruder nodes vs Simulation time



4.2 Calculations



Analysis Result

☑ File Name	tf
☑ File Format	Old
☑ Sent Packets	17,569
☑ Received Packets	17,520
☑ Dropped Packets	0
☑ Overhead	2,258
☑ Dropped Bytes	0 bytes
☑ End to End Delay	0.38
☑ Packet Delivery Ratio	99.72%
☑ Throughput	1,096.47
☑ Normalized Routin...	0.13%

Performance Metrics

☑ Sent Packets ☑ Received Packets ☑ Dropped Packets
☑ Overhead ☑ Dropped Bytes ☑ End to End Delay
☑ Packet Delivery Ratio ☑ Throughput ☑ Normalized Routing Load

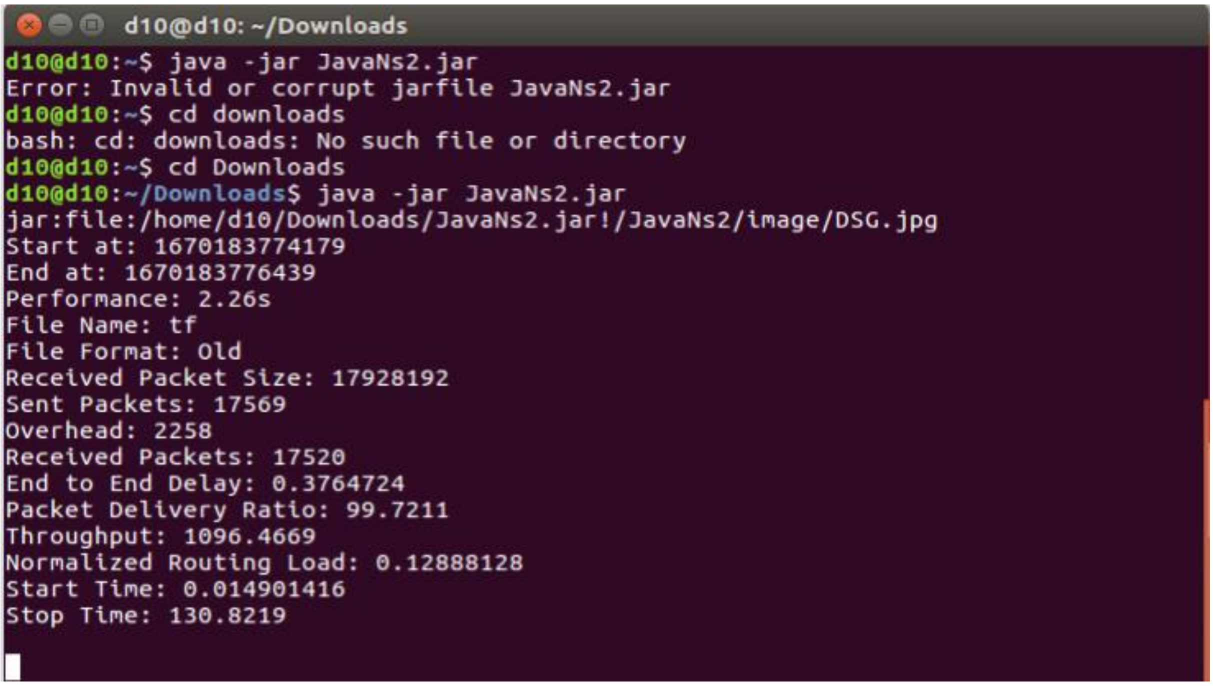
Steps/Monitor

1. Data Selected
2. Metrics Selected
3. Data Calculated
4. Select Chart/Graph

Error

● File Read ● File Write

Fig 4.5 Calculations using trace file



```
d10@d10:~$ java -jar JavaNs2.jar
Error: Invalid or corrupt jarfile JavaNs2.jar
d10@d10:~$ cd downloads
bash: cd: downloads: No such file or directory
d10@d10:~$ cd Downloads
d10@d10:~/Downloads$ java -jar JavaNs2.jar
jar:file:/home/d10/Downloads/JavaNs2.jar!/JavaNs2/image/DSG.jpg
Start at: 1670183774179
End at: 1670183776439
Performance: 2.26s
File Name: tf
File Format: Old
Received Packet Size: 17928192
Sent Packets: 17569
Overhead: 2258
Received Packets: 17520
End to End Delay: 0.3764724
Packet Delivery Ratio: 99.7211
Throughput: 1096.4669
Normalized Routing Load: 0.12888128
Start Time: 0.014901416
Stop Time: 130.8219
```

Fig 4.6 Data in the form of numericals

CHAPTER 5

5.1 CONCLUSION

The wireless sensor network is vulnerable to several types of attacks. We talked about the blackhole attack in this report. We looked into the causes and detection methods. The detection and eradication of a blackhole attacker has been demonstrated. This approach increases the packet delivery ratio while decreasing the delay. By avoiding the attacker nodes, the proposed mechanism finds a different route to the destination. To identify black hole attacks in wireless sensor networks, we used one of the routing methods called AODV. All of the techniques presented will definitely be used to detect black holes, but they can function more effectively when combined. We also observed the performance of several factors such as throughput, end-to-end delay, and packet delivery ratio in numerical numbers, which allows us to readily evaluate the performance of the black hole attack.

5.2 FUTURE ENHANCEMENTS

Future blackhole attack solutions improve security and prevent single and multiple black hole attacks. By integrating more modern and updated routing protocols, we can demonstrate significant improvements in characteristics such as throughput, energy usage, and end-to-end delay. There is still room to increase network parameter values under blackhole attack by employing additional strategies to identify and avoid blackhole attack in wireless sensor networks. The current study will be enhanced in the future to include various topologies and attack numbers.

By creating a new protocol, MAODV, which is a modified version of the AODV routing protocol, we can improve the performance of black hole attacks. It adds the following features to the AODV. The black hole node is validated as valid or invalid by inserting a verified field in RREP format and delivering the two RREP packets. The routing process ignores incorrect nodes. This solution improves security by preventing both single and multiple black hole attacks. The suggested protocol's simulation indicates a considerable improvement in throughput, energy consumption, and end-to-end delay. Here in this report, we used only AODV routing protocol which involves a process called Route Discovery Process to detect the blackhole attack. The modified version can enhance the secure routing mechanism.

CHAPTER 6

REFERENCES

[1] “A Survey of Various Algorithms to Detect Black Hole Attack in Wireless Sensor Network”,Soni Rani1, Charanjit Singh2,International Journal of Engineering Development and Research,IJEDR 2016 (www.ijedr.org),volume 4,Issue 3

[2] “Blackhole Attack Detection And Prevention In Wireless Sensor Networks:A Study”,1 Bindu Rani 2 Harkesh Sehrawat, www.jetir.org,JETIR March 2018, Volume 5, Issue 3.

[3] “Performance Evaluation of Routing Protocol for FANET using Ns2”,1 Megha N,2 Dr. Mallikarjun B C,International Journal of Engineering Research & Technology (IJERT),Vol. 9 Issue 07, July-2020

[4] Chandeep Singh, Vishal Walia, Dr.Rahul Malhotra, “Genetic Optimization Based Adaptive approach for the determination of Black Hole Attack in aodv protocol” ,2nd international conference on science, technology and management, pp 2742-2753, 2017.

[5] Mohit Kumar, Rashmi Mishra, “An Overview of MANET: History, Challenges and Application”,Indian Journal of Computer Science and Engineering (IJCSE), 2012,Vol.3.

[6] Jiahong Weng, “Security Issues in Mobile Ad Hoc Networks-A Survey”