Simulation of Blackhole attack in Network Simulator 3

A Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Bachelor of Technology
in
Computer Science & Engineering

by
Deepak Bulani(20153026)
Gaurav Agarwal(20154097)
Bonthu Harsha Vardhan Reddy(20154148)
Avichal Verma(20154085)
Amit Ranjan(20154101)

to the
COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY
ALLAHABAD
April, 2018

UNDERTAKING

I declare that the work presented in this report titled "*Simulation of Blackhole attack in Network Simulator 3*", submitted to the Computer Science and Engineering Department, Motilal Nehru National Institute of Technology, Allahabad, for the award of the **Bachelor of Technology** degree in **Computer Science & Engineering**, is my original work. I have not plagiarized or submitted the same work for the award of any other degree. In case this undertaking is found incorrect, I accept that my degree may be unconditionally withdrawn.

April, 2018
Allahabad

(Deepak Bulani, 20153026)

(Gaurav Agarwal, 20154097)

(Bonthu Harsha Vardhan Reddy, 20154148)

(Avichal Verma, 20154085)

(Amit Ranjan, 20154101)

'

CERTIFICATE

Certified that the work contained in the report titled "*Simulation of Blackhole attack in Network Simulator 3*", by
Deepak Bulani(20153026)
Gaurav Agarwal(20154097)
Bonthu Harsha Vardhan Reddy(20154148)
Avichal Verma(20154085)
Amit Ranjan(20154101)
has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Dr. Dinesh Singh
(Assistant Professor)
Computer Science and Engineering Dept.
M.N.N.I.T, Allahabad

April, 2018

Preface

Mobile devices form an integral part of our lives and their importance cannot be understated,also they can construct network proactively to exchange information where the conventional communication infrastructure are not exist. We call this kind of network environment as Ad Hoc Network. However, the Ad Hoc Network has vulnerability in data security due to its characteristics of network protocol. The Blackhole Attack is one of the major contributor in the risks vulnerable to the Ad Hoc Network as an attacker makes faulty route by responding fake network information to the information source, and intercepts data through faulty route they made.
In this project we first analyzed various Vehicular Adhoc Network protocols like AODV,DSR and DSDV and performed real-time simulation by integrating SUMO and NS-3.Also,we scrutinized the application of routing protocols in the above scenario.
The working of Blackhole attack was simulated on NS-3 and the analysis was made using PyViz visualizer which helped us to construct an in-depth understanding of the flaws in the routing protocols.

Acknowledgements

We would like to express our sincere gratitude towards Mr. Dinesh Singh, for his valuable guidance and advice. Without his co-operation and support, this project could not have been completed. From aiding us with purposeful software and also patiently resolving our doubts, Dinesh Sir played a significant role in making the project happen. We also extend our thanks to our colleagues who provided valuable insight and expertise that greatly assisted in making this research work comprehensive and precise.
We would also like to show our gratitude to Mr. Shashank Srivastav for sharing his pearls of wisdom on how wireless networking works and helping us to understand the inner workings of network protocols and the attacks associated with them.

Contents

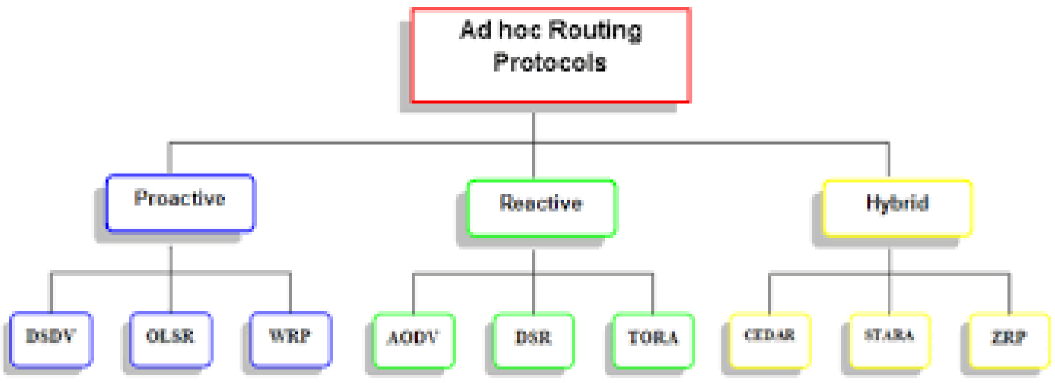
Chapter 1

Introduction

MANET is considered a collection of wireless mobile nodes that communicate with each other without the use of a network infrastructure or central base station.In such a network, each mobile node operates not only as a host but also as a router, forwarding packets for other mobile nodes in the network that may not be within direct wireless transmission range of each other. These infrastructure-less mobile nodes in ad hoc networks dynamically create routes among themselves to form own wireless network on the fly. Different applications of MANET are sensor networks, vehicle to vehicle communication, military battlefields and emergency services. These wireless links makes the MANETs more susceptible to attacks, which make it easier for the attacker to go inside the network and get access to the ongoing communication. Blackhole attack is one kind of attack that a MANET suffers from.

MANET Routing Protocols

For deployment of MANETs several routing protocols have been proposed. The protocols differ in terms of routing methodologies and the information used to make routing decisions. On the behalf of their different working methodologies, these routing protocols are divided into three different categories:



- Proactive Protocols - Proactive Protocols are also known as Table Driven Protocols.These protocols maintain constantly updated topology of the network. Every node in the network knows about the other nodes in advance the routing information is usually kept in number of different tables. These tables are updated according to the changes in the network.

- Reactive Protocols - Reactive Protocols are also known as, On Demand Routing Protocols because they establish routes between nodes only when they are required to route data packets.

- Hybrid Protocols - Hybrid Routing Protocols combine proactive protocols with reactive protocols. To provide the best path to destination network it uses the distance-vectors techniques.

In this project, we will mainly focus on the Ad Hoc On Demand Distance Vector Routing protocol (AODV) and how it is susceptible to Blackhole Attack.

AODV Protocol

AODV is a reactive routing protocol in which the network generates routes at the start of communication. Each node has its own sequence number and this number increases when links change. Each node judges whether the channel information is new according to sequence numbers. Figure 1 illustrates the route discovery process in AODV. In this figure, node S is trying to establish a connection to destination D. First, the source node S refers to the route map at the start of communication. In case where there is no route to destination node D, it sends a Route Request (RREQ) message using broadcasting. RREQ ID increases one every time node sends a RREQ. Node A and B which have received RREQ generate and renew the route to its previous hop. They also judge if this is a repeated RREQ. If such RREQ is received, it will be discarded. If A and B has a valid route to the destination D, they send a RREP message to node S. By contrast, in case where the node has no valid route, they send a RREQ using broadcasting. The exchange of route information will be repeated until a RREQ reaches at node D. When node D receives the RREQ, it sends a RREP to node S. When node S receives the RREP, then a route is established. In case a node receives multiple RREPs, it will select a RREP whos the destination sequence number (Dst Seq) is the largest amongst all previously received RREPs. But if Dst Seq were same, it will select the RREP whose hop count is the smallest.

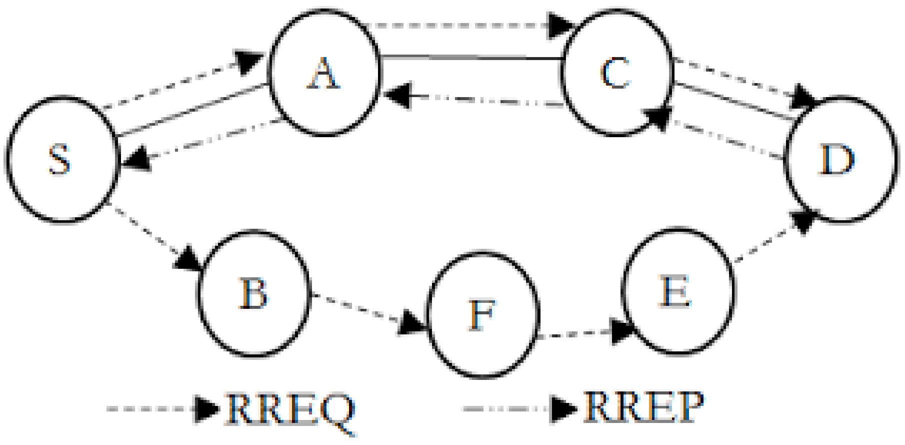


Figure. 1 Route discovery process

In Figure 2, when node B detects disconnection of route, it generates Route Error (RERR) messages and puts the invalidated address of node E into list, then sends it to the node A. When node A receives the RERR, it refers to its route map and the current list of RERR messages. If there was a route to destination for node E included in its map, and the next hop in the routing table is a neighbouring node B, it invalidates the route and sends a RERR message to node S. In this way, the RERR message can be finally sent to the source node S.

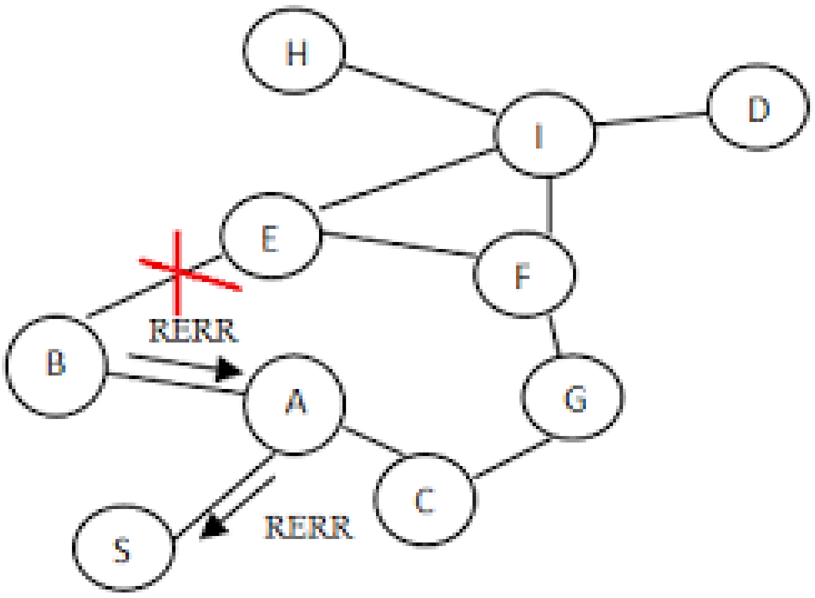


Figure. 2 Transferring route error messages

Security Issues in AODV

AODV besides being an efficient routing algorithm possesses some limitations due to which it is easily attacked by the external intruders This protocol is exposed to a variety of attacks, the impact of these attacks on AODV protocol are not the same. Some of these attacks can cause a breakdown of the network connectivity, increasing the end-to-end delay, increasing the number of the loss packets, or shutting down some nodes by consuming all the energy left in there batteries.Blackhole attack is an attack in which malicious node uses its routing protocol to advertise itself for having the shortest path with minimum hops to the destination node whose data packet it wants to take away. In this way attacker node is always available to the nodes whose packets it wants to retain.

Chapter 2

Related Work

The problem of security and cooperation enforcement has received considerable attention by researchers in the ad hoc network community. In this section, some of these contributions are presented.

Mohammad Al-Shurman [1] has proposed solution in which host node sends a packet that includes the packet id and sequence number to destination node through three different paths. When any intermediate node or malicious node has a route to the destination node will reply to that packet. Once the source node collects all the reply from the intermediate nodes, source node will check for the secure route to the destination.
Hesiri Weerasinghe [2] studied the problem of cooperative blackhole attack in MANET. He has compared the performance of the network in terms of throughput, end to end delay, packet loss and packet overhead by varying the number of blackhole nodes, number of mobile nodes, mobility speed and the size of terrain area. Simulation results show greater reduction in terms of throughput and packet loss.
Dinesh [3] analyzed the behavior of malicious node in different routing protocols in MANET. He proposed a solution for finding a safe route by waiting and checking the replies received from the intermediate nodes. He observed the network under varying network mobility with maximum speed of 10m/s. He concluded that AODV routing protocol results in a better performance with blackhole nodesthan DSR in terms of throughput.

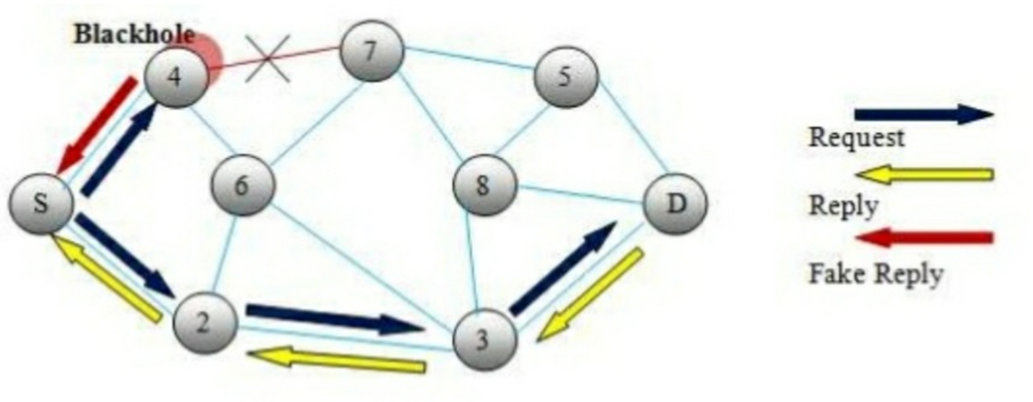
Chapter 3

Proposed Work

3.1 Analysis of BlackHole Attack:

Blackhole Attack is a simple but certainly effective Denial of Service attack in which a malicious node, through its routing protocol, advertises itself for having the shortest path to the destination node or to the node whose packets it wants to intercept. It pretends to have enough of fresh routes for a certain destination. The source node assumes it to be true and the data packets are forwarded to a node which actually does not exist, causing the data packets to be lost. When a source node wants to initiate the communication, it broadcasts a RREQ message for route discovery. As soon as the malicious node receives this RREQ packet, it immediately responds with a false RREP message to the respective node advertising itself as the destination or having the shortest path for that destination. Since the malicious node needs not to check its routing table before responding to a routing request, it is often the first one to reply compared to other nodes. When the requesting node receives this RREP, it terminates its routing discovery process and ignores all other RREP messages coming from other nodes. Thus the data packets are sent to such a hole from where they are not sent anywhere and absorbed by the malicious node. Often many nodes send RREQ simultaneously; the attacker node is still able to respond immediately with false RREP to all requesting nodes and thus easily takes access to all the routes. In this way source nodes are bluffed by malicious node which gulps a lot of network traffic to itself resulting severe loss of data. BlackHole nodes may also work as a group in a network. This kind of attack is called Collaborative BlackHole attack or BlackHole Attack with multiple malicious nodes.

The main objective of blackhole attack is to drape packets and break communications between nodes, all the network's traffic is redirected to a specific node which does not exist at all. Blackhole node work with two scenarios, in the first one the node exploits all the vulnerability that exists in an ad hoc network such as announcing itself having a valid route to a destination node; the Second one, the node drupes and controls all the intercepted packets. The Blackhole attack in AODV protocol can be classified into two categories: blackhole attack caused by RREP and blackhole attack caused by RREQ.



Blackhole attack

3.1.1 Blackhole attack caused by RREQ

This attack work by sending fakes RREQ messages, an attacker can form a blackhole attack as follows:

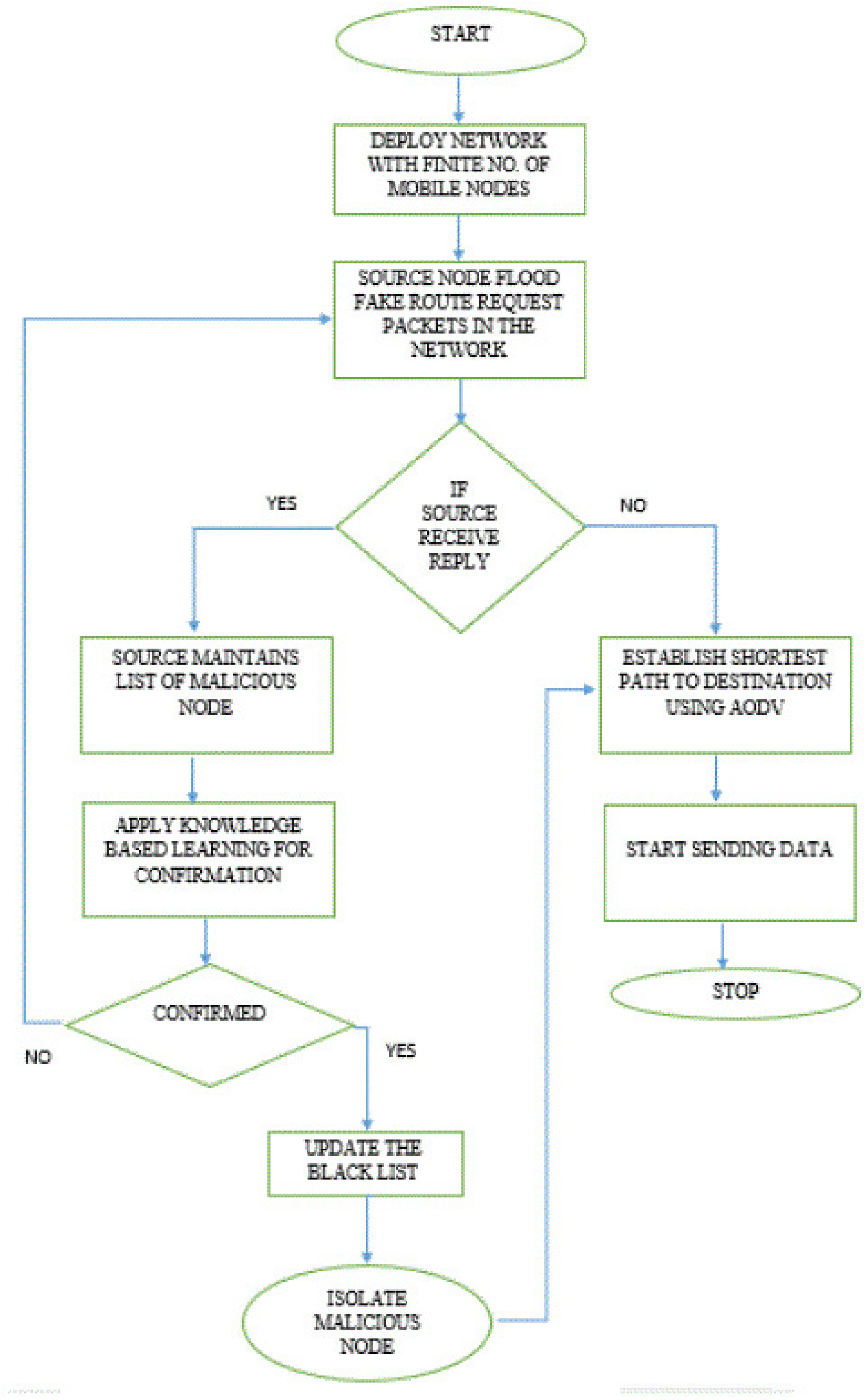
- Set the originator IP address in RREQ.
- Set the destination IP address in RREQ.
- Set the source IP address of the IP header to its own IP address.
- Set the destination IP address of the IP header to broadcast address or to a nonexistent IP address.
- Increase the sequence number and declaring a low hop count and put them in the related fields in RREQ.

False information about source node is inserted to the routing table of nodes that get the fake RREQ, if these nodes want to send data to the source, at first step they send it to the malicious node.

3.1.2 Blackhole attack caused by RREP

This attack work by sending fakes RREP messages after receiving RREQ from source node, a malicious node can generate blackhole attack by sending RREP as follow:

- Set the originator IP address in RREP to the originator nodes IP address.
- Set the destination IP address in RREP to the destination nodes IP address.
- Set the source IP address of the IP header to its own IP address.
- Set the destination IP address of the IP header to the IP address of the node that RREQ has been received from it.


START
DEPLOY NETWORK WITH FINITE NO. OF MOBILE NODES
SOURCE NODE FLOOD FAKE ROUTE REQUEST PACKETS IN THE NETWORK
IF SOURCE RECEIVE REPLY
YES
NO
SOURCE MAINTAINS LIST OF MALICIOUS NODE
APPLY KNOWLEDGE BASED LEARNING FOR CONFIRMATION
CONFIRMED
NO
YES
UPDATE THE BLACK LIST
ISOLATE MALICIOUS NODE
ESTABLISH SHORTEST PATH TO DESTINATION USING AODV
START SENDING DATA
STOP

Proposed theoretical solution of Blackhole attack

Chapter 4

Experimental Setup and Results Analysis

In VANET, two types of simulation are required for smooth functioning, Network simulation and Traffic simulation. Network simulators are used to evaluate the network protocols or say routing protocols for issues related to the communication among vehicles ,and application in a variety of conditions whereas traffic simulators are used for Traffic engineering and transportation.

4.1 Network Simulator 3(NS-3)

NS-3 is a discrete-event network simulator and a free software which succeeds popular network simulator NS-2, licensed under the GNU GPLv2 license and is publicly available for research, development and use.The goal of the NS-3 project is to develop a preferred, open simulation environment for networking research. NS-3 is available for Linux, Mac OS and MS Windows using cygwin. A simple NS-3 script can be written in either C++ or Python language. NS-3 is built on both-C++ and Python bindings. We're using C++ scripts in our simulation as we're more familiar with C++.

Import Statements: Which modules we need to add for our simulation. NS-3 provides a set of HelperClasses which are very helpful to newbies.A good example of HelperClass

is "aodv-helper.h". Statements can be imported in the following way:
#include "ns3/aodv-module.h"
#include "ns3/aodv-helper.h"

Main Function:
This is the core of our simulation. We can configure the scenario and implement protocols in this function. There are some basic necessities which must be configured in a simulation.

1. **WiFi Channel:** For VANET Simulation, the first thing to do is create a WiFi channel. There are various link layer models support like WiFi-802.11, WiMax-802.16, Point To Point Channels, CSMA links, etc. This models can be configured according to the needs. WiFiHelper class is used to set standards.
   WifiHelper wifi;
   wifi.SetStandard (WIFI_PHY_STANDARD_80211b);
   Next step is to set a propogation model for the given channel and then set Propogation Delay and Propogation Loss model. We've selected YansWiFiChannelHelper to configure the WiFi channel and FriisPropagationLossModel as propagation loss model.The reason to select FriisPropagationLossModel is because this model is more suitable for VANET because in this model we can set a minimum distance for the node which will be it's maximum transmission capacity.
   YansWifiChannelHelper wifiChannel;
   wifiChannel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");
   wifiChannel.AddPropagationLoss ("ns3::FriisPropagationLossModel","MinDistance"
   ,DoubleValue (250));
   wifiPhy.SetChannel (wifiChannel.Create ());

2. **Node Creation:** So we've created channel. Now we need create nodes and add mobility to the nodes. NS-3 provides us with ns2mobilityhelper class which lets the user use external mobility files from sources like sumo. It accepts .tcl files as inputs. NetDeviceContainer class is used to configure all the devices used in simulation.
   NoeContainer c;
   c.Create (nNodes);

Ns2MobilityHelper ns2 = Ns2MobilityHelper("scratch/g50_8mob.tcl");
ns2.Install ();
NetDeviceContainer devices = wifi.Install (wifiPhy, wifiMac, c);

3. **Routing:** This is how we import mobility from external source in ns3 provided that it follows ns2mobility file structure. NodeContainer class creates c number of nodes for our simulation. The number of nodes in simulation must be equal to number of vehicles in sumo configuration. Next we have to assign a routing protocol for the simulation followed by a relevent Internet protocol. For routing protocol, there are some HelperClass available like AodvHelper. The following set of code implements aodv routing protocol for the simulation. InternetStackHelper is used to install an internet stack on each node.
   AodvHelper aodv;
   InternetStackHelper internet;
   internet.SetRoutingHelper (aodv);
   internet.Install (c);

4. **IP Addressing:** NS-3 supports both IPv4 and IPv6 addressing modes. User can select any one of them as per the requirement. Furthermore, user can also add headers in IPv4 and IPv6 packets. For VANET scenarios, generally IPv4 addressing is used.
   Ipv4AddressHelper ipv4;
   ipv4.SetBase ("10.1.1.0", "255.255.255.0");
   Ipv4InterfaceContainer i = ipv4.Assign (devices);

5. **Running Simulation:** Coming back to the simulation; once the channels, devices, nodes and applications are configured, the simulation can be started. It must be managed by the user by providing ending time of the simulation otherwise simulation will never be completed.
   Simulator::Run ();
   Simulator::Stop(Seconds (120));

6. **Visualization:** To run the simulation with visualization, we use following command:

./waf –run filename –visualize

4.2 Simulation of Urban Mobility

Simulation of Urban Mobility (SUMO) is an open source traffic simulation package including net import and demand modelling components. SUMO helps to investigate several research topics e.g. route choice and traffic light algorithm or simulating vehicular communication. Therefore the framework is used in different projects to simulate automatic driving or traffic management strategies. Simulations has: space-continuous and time-discrete vehicle movement, different vehicle types, multi-lane streets with lane changing, different right-of-way rules, traffic lights, a fast open GL graphical user interface, manages networks with several edges, fast execution speed, interoperability with other application at run-time, network-wide, edge-based, vehicle-based, and detector-based outputs, supports person-based inter-modal trips, high portability and high interoperability.

Steps of converting a Real World Map into SUMO network file

Step 1: Steps to create a Traffic

1. Open browser and type http://www.openstreetmap.org and search a particular area and click export in the top.
2. Click "Manually select an area" and select the area and click export again (it will download a file called map.osm, rename this file as per your convenience)
3. Enter these commands on terminal:
   (a) netconvert –osm-files mnnit.osm -o mnnit.net.xml
   (b) polyconvert –osm-files mnnit.osm –net-file mnnit.net.xml –type-file osmPolyconvert.typ.xml -o mnnit.poly.xml
   (c) python SUMO_HOME/tools/randomTrips.py -n mnnit.net.xml -r mnnit.rou.xml -e 100 -l

4. Now create a configuration file with the following code

   $\langle configuration \rangle$

   $\langle input \rangle$

   $\langle net - filevalue = "mnnit.net.xml" / \rangle$

   $\langle route - filesvalue = "mnnit.rou.xml" / \rangle$

   $\langle additional - filesvalue = "mnnit.poly.xml" / \rangle$

   $\langle / input \rangle$

   $\langle time \rangle$

   $\langle beginvalue = "0" / \rangle$

   $\langle endvalue = "100" / \rangle$

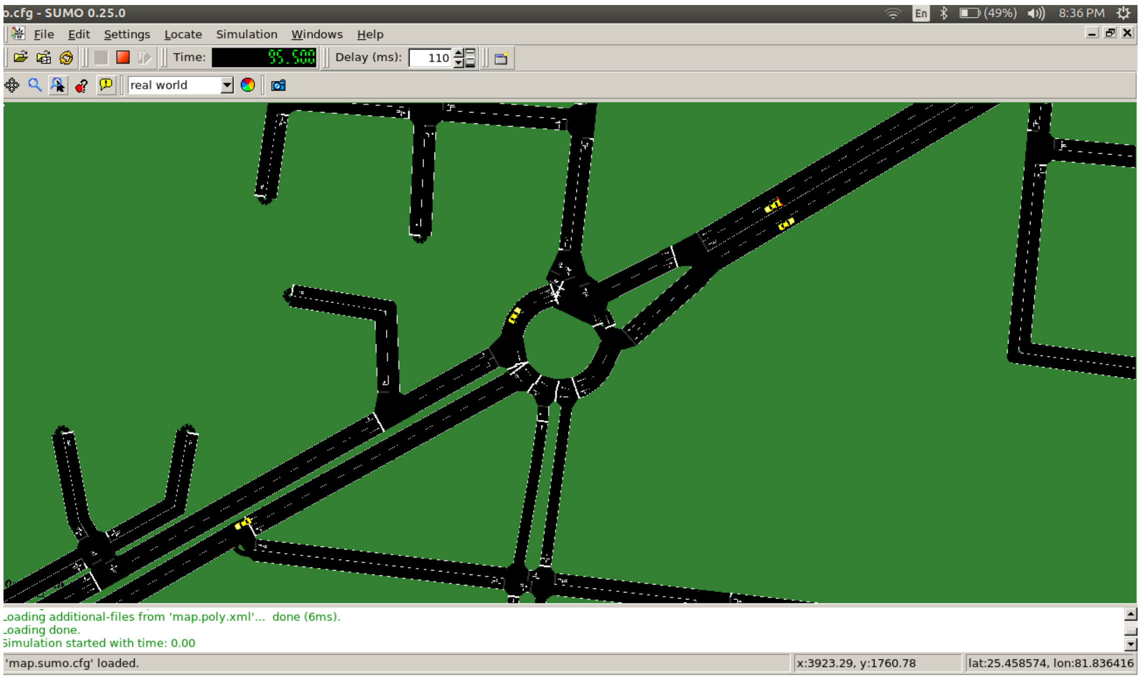
   $\langle step - lengthvalue = "0.1" / \rangle$

   $\langle / time \rangle$

   $\langle / configuration \rangle$

5.Visualization on SUMO GUI:

SUMO GUI is basically the same application as SUMO just extended by a graphical user interface which is supported to Open GL library.



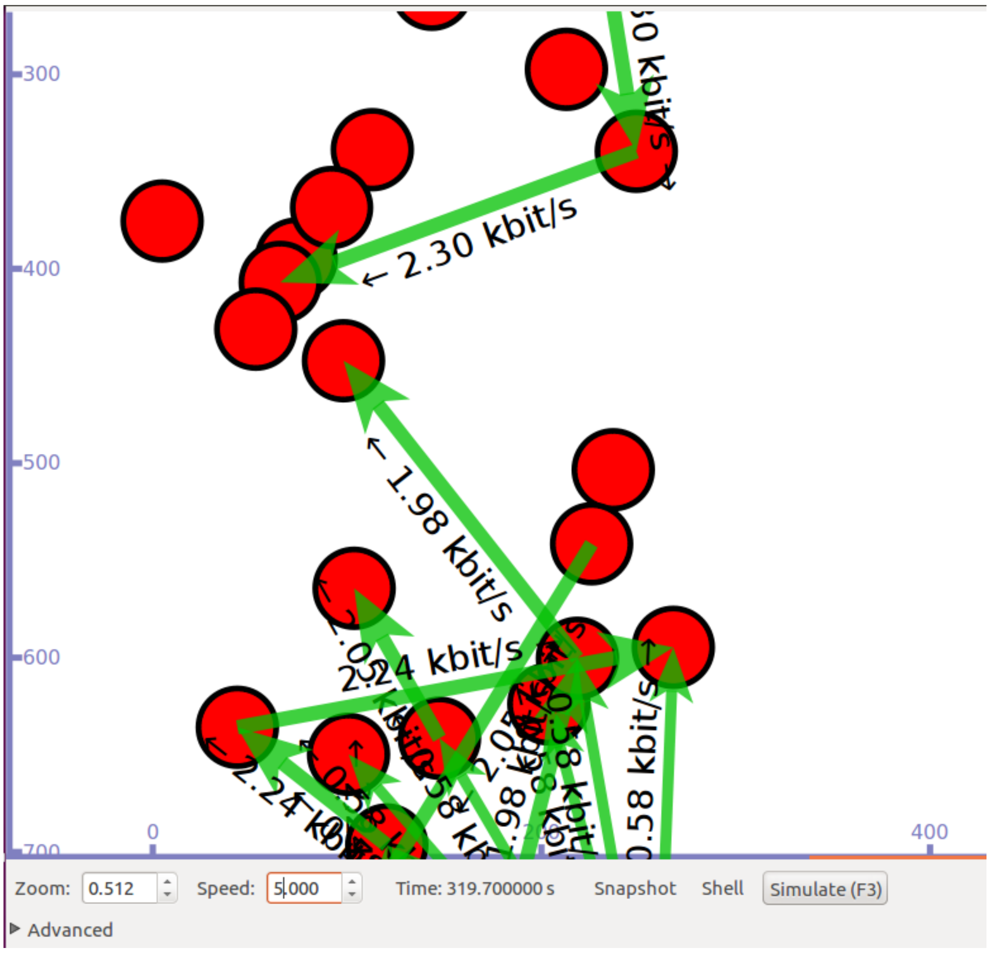
```
Loading additional-files from 'map.poly.xml'...  done (6ms).
Loading done.
Simulation started with time: 0.00
'map.sumo.cfg' loaded.    x:3923.29, y:1760.78    lat:25.458574, lon:81.836416
```

SUMO GUI output for Civil Lines,Allahabd Open Street Map

Step 2: Exporting to NS3

Open terminal and type the following commands

- sumo -c mnnit.sumo.cfg –fcd-output mnnit.sumo.xml
- python /home/deepak/sumo-0.26.0/tools/traceExporter.py –fcd-input mnnit.sumo.xml –ns2config-output mnnit.tcl –ns2activity-output activity.tcl –ns2mobility-output mobility.tcl
- ./waf –run "vanet-routing-compare –traceFile=scratch/mobility.tcl –scenario=2 –protocol=2 –logFile=aodvlogging.log" –vis



Simulation output of "vanet-routing-compare" using SUMO mobility real time trace with ns3 and applying aodv routing protocol on mobile nodes

4.3 Simulation of Blackhole Attack

4.3.1 Blackhole ns-3.27 patch

Using the patch we modified the standard aodv-routing-protocol.cc file to visualize it's behaviour in case of Blackhole attack.

- Check if the node is suppose to behave maliciously
  ```
  if(IsMalicious) {
  //When malicious node receives packet it drops the packet.
  std ::cout<<"Launching Blackhole Attack! Packet dropped . . . ";
  return false;
  }
  ```
- Adding attribute for Malicious node
  ```
  .AddAttribute ("IsMalicious", "Is the node malicious", BooleanValue (false), Make-
  BooleanAccessor (&RoutingProtocol::SetMaliciousEnable, &RoutingProtocol::GetMaliciousEnable),
  MakeBooleanChecker ()) ;
  ```
- Attract node to set up path through malicious node
  ```
  if(IsMalicious)
  {
  rrepHeader.SetHopCount(1);
  }
  ```
- If node is malicious, it creates false routing table entry having sequence number much higher than that in RREQ message and hop count as 1.Malicious node itself sends the RREP message,so that the route will be established through malicious node
  ```
  if(IsMalicious)
  {
  Ptr<NetDevice>dev = m_ipv4- >GetNetDevice (m_ipv4->GetInterfaceForAddress
  (receiver));
  RoutingTableEntry falseToDst(dev,dst,true,rreqHeader.GetDstSeqno()+100,m_ipv4-
  >GetAddress (m_ipv4->GetInterfaceForAddress (receiver),0),1,dst,m_activeRouteTimeout);
  SendReplyByIntermediateNode (falseToDst, toOrigin, rreqHeader.GetGratuitousRrep
  ```

());
return; }

4.3.2 Blackhole attack Simulation code

Network topology

n0 ————>n1 ————>n2 ————->n3

Each node is in the range of its immediate adjacent.

Source Node:**n1** Destination Node:**n3** Malicious Node:**n0**

- Modules Import:
  ```
  #include "ns3/aodv-module.h"
  #include "ns3/netanim-module.h"
  #include "ns3/core-module.h"
  #include "ns3/network-module.h"
  #include "ns3/internet-module.h"
  #include "ns3/applications-module.h"
  #include "ns3/mobility-module.h"
  #include "ns3/wifi-module.h"
  #include "ns3/netanim-module.h"
  #include "ns3/flow-monitor-module.h"
  #include "ns3/mobility-module.h"
  #include "myapp.h"
  ```
- Creating malicious and non-malicious nodes:
  ```
  NodeContainer c;
  NodeContainer not_malicious;
  NodeContainer malicious;
  c.Create(4);
  not_malicious.Add(c.Get(1));
  not_malicious.Add(c.Get(2));
  not_malicious.Add(c.Get(3));
  malicious.Add(c.Get(0));
  ```

- Creating wifi network topology
  WifiHelper wifi;

  YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default ();
  wifiPhy.SetPcapDataLinkType (YansWifiPhyHelper::DLT_IEEE802_11);

  YansWifiChannelHelper wifiChannel ;
  wifiChannel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");
  wifiChannel.AddPropagationLoss ("ns3::TwoRayGroundPropagationLossModel", "SystemLoss", DoubleValue(1), "HeightAboveZ", DoubleValue(1.5));

  // For range near 250m wifiPhy.Set ("TxPowerStart", DoubleValue(33));
  wifiPhy.Set ("TxPowerEnd", DoubleValue(33));
  wifiPhy.Set ("TxPowerLevels", UintegerValue(1));
  wifiPhy.Set ("TxGain", DoubleValue(0));
  wifiPhy.Set ("RxGain", DoubleValue(0));
  wifiPhy.Set ("EnergyDetectionThreshold", DoubleValue(-61.8));
  wifiPhy.Set ("CcaMode1Threshold", DoubleValue(-64.8)); wifiPhy.SetChannel (wifiChannel.Create ());

  // Add a non-QoS upper mac
  NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default ();
  wifiMac.SetType ("ns3::AdhocWifiMac");

  // Set 802.11b standard
  wifi.SetStandard (WIFI_PHY_STANDARD_80211b);
  wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager", "DataMode", StringValue(phyMode),"ControlMode",StringValue(phyMode));
  NetDeviceContainer devices;
  devices = wifi.Install (wifiPhy, wifiMac, c);

- Enable AODV
  AodvHelper aodv;
  AodvHelper malicious_aodv;

- Setting up internet stack on nodes
  InternetStackHelper internet;
  internet.SetRoutingHelper (aodv);
  internet.Install (not_malicious);

  malicious_aodv.Set("IsMalicious",BooleanValue(true)); // putting *false* instead of *true* would disable the malicious behavior of the node
  internet.SetRoutingHelper (malicious_aodv);
  internet.Install (malicious);

- Setting mobility for nodes using ns3::ConstantPositionMobilityModel
  MobilityHelper mobility;
  Ptr<ListPositionAllocator>positionAlloc = CreateObject <ListPositionAllocator>();
  positionAlloc ->Add(Vector(100, 0, 0)); // node0
  positionAlloc ->Add(Vector(200, 0, 0)); // node1
  positionAlloc ->Add(Vector(450, 0, 0)); // node2
  positionAlloc ->Add(Vector(550, 0, 0)); // node3
  mobility.SetPositionAllocator(positionAlloc);
  mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
  mobility.Install(c);

- Create udp connection from n1 to n3
  uint16_t sinkPort = 6;
  Address sinkAddress (InetSocketAddress (ifcont.GetAddress (3), sinkPort)); // interface of n3
  PacketSinkHelper packetSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress (Ipv4Address::GetAny (), sinkPort)); ApplicationContainer sinkApps = packetSinkHelper.Install (c.Get (3)); //n3 as sink
  sinkApps.Start (Seconds (0.));
  sinkApps.Stop (Seconds (100.));

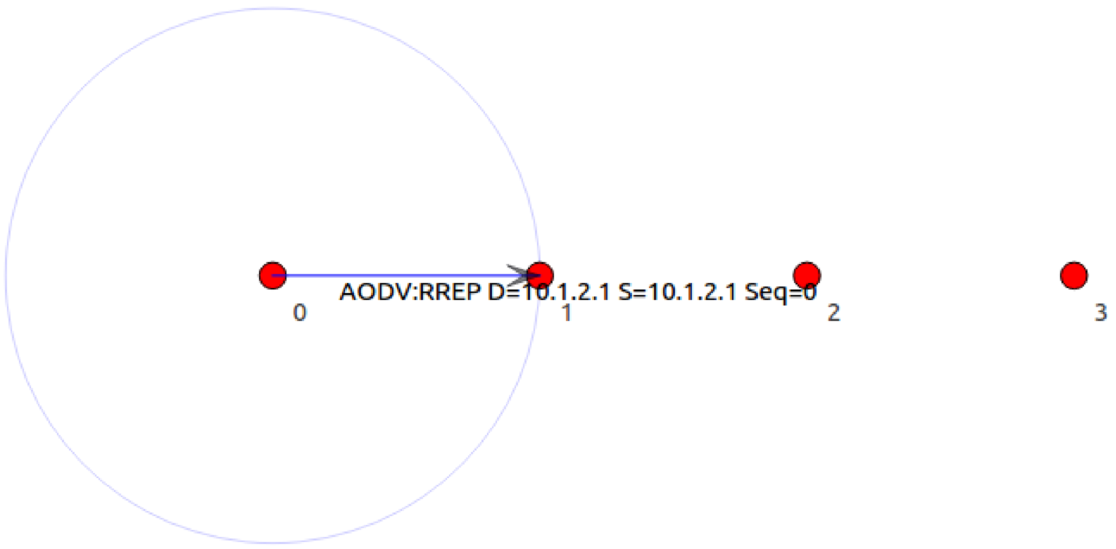
```
Ptr<Socket>ns3UdpSocket = Socket::CreateSocket (c.Get (1), UdpSocketFac-
tory::GetTypeId ()); //source at n1

// Create UDP application at n1
Ptr<MyApp>app = CreateObject<MyApp>();
app->Setup (ns3UdpSocket, sinkAddress, 1040, 5, DataRate ("250Kbps"));
c.Get (1)->AddApplication (app);
app->SetStartTime (Seconds (40.));
app->SetStopTime (Seconds (100.));
```

- Calculating throughput through Flowmonitor

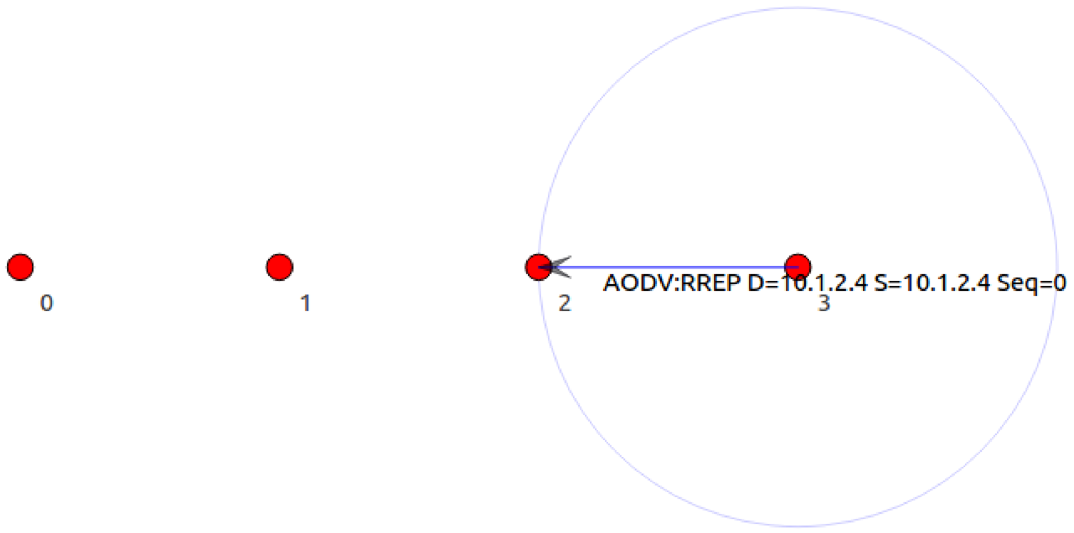
```
FlowMonitorHelper flowmon;
Ptr<FlowMonitor>monitor = flowmon.InstallAll();
monitor->CheckForLostPackets ();

Ptr<Ipv4FlowClassifier>classifier = DynamicCast<Ipv4FlowClassifier>(flowmon.GetClassifier
());
std::map<FlowId, FlowMonitor::FlowStats>stats = monitor->GetFlowStats ();
for (std::map, FlowMonitor::FlowStats>::const_iterator i = stats.begin (); i !=
stats.end (); ++i)
{
Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow (i->first);
if ((t.sourceAddress=="10.1.2.2" && t.destinationAddress == "10.1.2.4"))
{
std::cout <<"Flow" <<i-<<" (" <<t.sourceAddress <<" ->" <<t.destinationAddress
<<")";
std::cout <<" Tx Bytes: " <<i->second.txBytes <<"";
std::cout <<" Rx Bytes: " <<i->second.rxBytes <<"";
std::cout <<" Throughput: " <<i->second.rxBytes * 8.0 / (i->second.timeLastRxPacket.
GetSeconds() - i->second.timeFirstTxPacket.GetSeconds())/1024/1024 <<" Mbps";
}
}
```

4.4 Simulation results

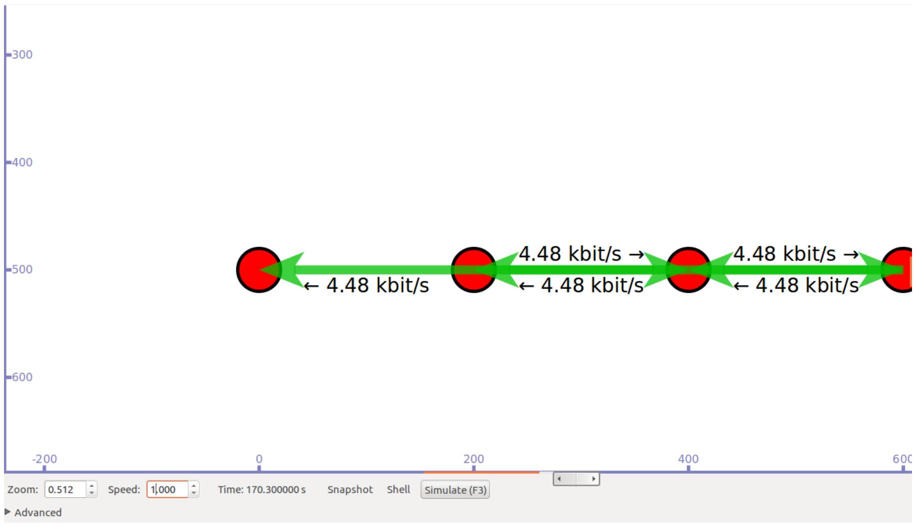
4.4.1 Blackhole attack



malicious node replies first with route reply packet

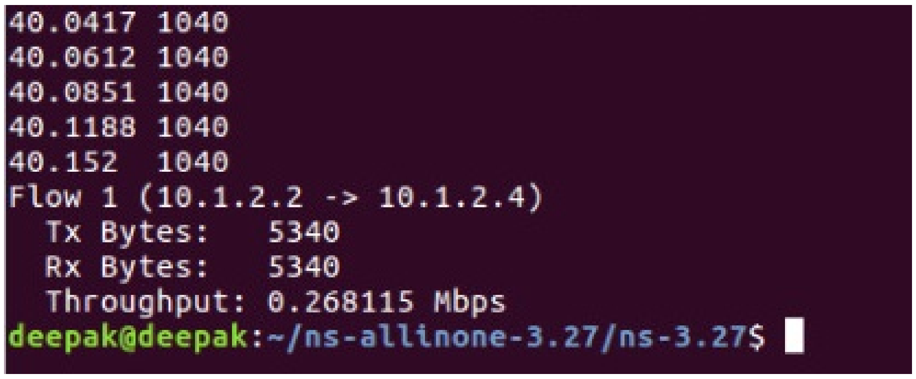


now non malicious node replies with route reply packet



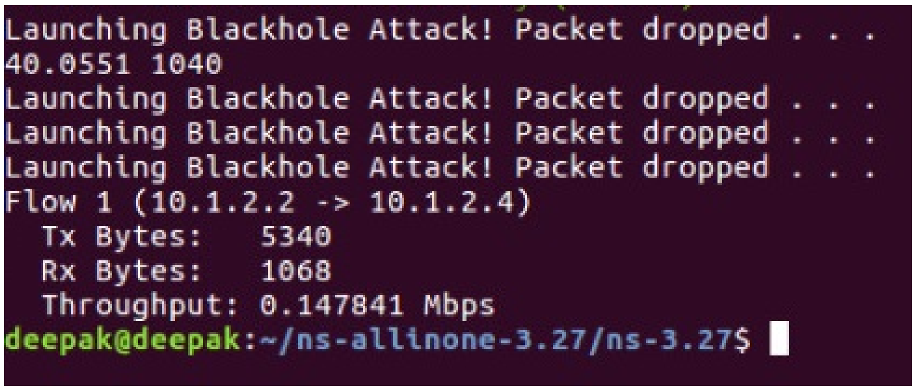
source node is forced to send packets to malicious node

4.4.2 Throughput results



```
40.0417 1040
40.0612 1040
40.0851 1040
40.1188 1040
40.152  1040
Flow 1 (10.1.2.2 -> 10.1.2.4)
  Tx Bytes:   5340
  Rx Bytes:   5340
  Throughput: 0.268115 Mbps
deepak@deepak:~/ns-allinone-3.27/ns-3.27$
```

Throughput without malicious(Blackhole) node



```
Launching Blackhole Attack! Packet dropped . . .
40.0551 1040
Launching Blackhole Attack! Packet dropped . . .
Launching Blackhole Attack! Packet dropped . . .
Launching Blackhole Attack! Packet dropped . . .
Flow 1 (10.1.2.2 -> 10.1.2.4)
  Tx Bytes:   5340
  Rx Bytes:   1068
  Throughput: 0.147841 Mbps
deepak@deepak:~/ns-allinone-3.27/ns-3.27$
```

Throughput with malicious(Blackhole) node

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In our paper we simulated Blackhole attack in NS3 on AODV VANET Routing protocol.This simuation helped us gain insight in the vulnerabilities of AODV protocol and how well BlackHole attack being one of the most powerful attacks on an Ad hoc network can cause a complete failure of the network by dropping all the traffic specially when the nodes are non-mobile. Additionally, we also integrated NS3 with SUMO where we generated real time traffic whose mobility was controlled by the sumo file mobility.tcl and the interaction of the mobile nodes was regulated by AODV protocol.

5.2 Future Work

We intend to analyse the working of BlackHole attack with the help of Wireshark and determine how exactly it affects the performance in the network for a given scenario. Also we would strive to obtain a solution which can try to mitigate and counter the obstacles posed by the blackhole attack. We also intend to perform an in-depth analysis of the performance of AODV under blackhole attack using performance metrics like average end to end delay, packet delivery ratio and routing overhead.In addition to it,we will compare the degradation of performance with other routing protocols in terms of variable node mobility, pause time and number of transactions.

References

[1] Mohammad Al-Shurman, and Seong Moo Yoo, "Black Hole Attack in Mobile Ad Hoc Networks," *2004 42nd Annual South east Regional Conference*.

[2] Hesiri Weerasinghe and Huirong Fu,Member of IEEE, "Preventing Co-operative Blackhole Attacks in Mobile Adhoc Network," *IJSEA,Vol 2,No 3,2008*.

[3] Dinesh Mishra, Yogendra Kumar Jain and Sudhir Agrawal, "Behaviour Analysis of Malicious Node in different routing algorithms in MANET," *International Conference on Advances in Computing Control and Telecommunications Technologies,2009*