TABLE OF CONTENTS:

1. Summary
2. Introduction
3. Downloading and Installing Ns-2 And Nam In Ubuntu 14.04
4. Software tools used with ns-2
5. Trace data Analyzing Applications
6. MANET – Mobile Ad-hoc network
7. Routing protocols used in MANET
8. Comparison of AODV with ZRP for End-to-End Delay, Throughput and Packet Delivery Ratio using X-graph
9. Black Hole Attack on Ad-hoc network (working on AODV protocol)
10. Black Hole Attack prevention
11. Simulation for system under Black Hole attack and preventing it
12. Comparison of AODV routing protocol with the system under Black Hole attack and with prevention of Black Hole attack (for End-to-End Delay, PDR and Throughput)
13. Conclusion
   - References
1. SUMMARY:
   Today, wired and wireless communication system has become very important in transferring data from one device to other. And amongst them, wireless communication is used more in most of the applications. Our project deals with complete understanding of NS2 network simulator; what is NS2, how does it work and how can we use it. Then we have studied different routing protocols used in Mobile Ad-hoc network and have tried to implement them using NS2 and have compared parameters like end-to-end delay, throughput, and Packet Delivery Ratio for AODV and ZRP. We then have implemented Black Hole attack on the AODV protocol operated system using NS2 and have given the proposed solution for it. And finally concludes with the comparative graph for the three systems like AODV, AODV under attack and AODV with solution.

2. INTRODUCTION:

2.1. History of NS2:
Ns started as a variant of the important network simulator software in 1989 and has progressed considerably over the past few years. In 1995 ns development was encouraged by agency called DARPA through the VINT project at LBL, Xerox PARC, UCB, and USC/ISI. Currently ns development is sustained through Defense Advanced Research Projects Agency (DARPA) with SAMAN and thru National Science Foundation with CONSER, each united with alternative researchers like ACIRI. Ns has invariably enclosed substantial contributions from alternative researchers, that includes wireless code from the UCB Daedelus and CMU Monarch comes and Sun Microsystems.

2.2. What is NS2?
NS2 is an open-source simulation tool that works on Linux and Windows (using Cygwin). NS2 is a discrete event simulator. It maintains the queue of events and each event is associated with the time. It is focused at networking analysis and supplies substantial pillar for simulation of routing, multicast protocols and Internet protocols such as UDP, TCP, RTP and SRM over wired and wireless (local and satellite) networks.

   At each loop takes in an event, executes it and moves on to the next event executes it and moves on. NS2 is implemented in OTcl and C++. A class can be installed entirely in C++ or in OTcl. Classes that implement in C++ are typically of the lower functionality and the classes that implement in OTcl provide the flexibility of gluing different objects. It also provides hooks to configure the parameters in the C++ objects.

2.3. Design of NS2:
Two different working language of NS2 : (1) an object oriented simulator that’s written in C++, and (2) a OTcl(Object oriented extension of Tcl) interpreter, accustomed to execute user’s command scripts. NS contains a reach library of network and protocol objects. There are 2 class hierarchies: the compiled C++ hierarchy and the depicted OTcl one, with one on one correspondence between them.

The compiled C++ hierarchy permits the user to get efficiency in the simulation and quicker execution times. This is helpful for the elaborated definition and operation of protocols. So the processing time for the packet and the event gets reduced. Then within the OTcl script provided by the user, one can define a network topology, the particular protocols and applications that we want to simulate and the kind of the output that we expect from the simulator. The OTcl uses the objects which are compiled in C++ through an OTcl linkage that makes a matching of OTcl object for each of the C++ codes.
2.4. Tool Command Language (TCL):

The inventor of TCL programing language is John Ousterhout. It is a very dynamic as well as powerful programing language that is easy to learn. It can be used widely for web and desktop applications, testing, networking and much more. It is an open source programing language that can work with different platforms (allows a very easy integration with other languages) and can be deployed easily. It has a graphical user interface development toolkit called a tk; which is used to make desktop applications very easily then the conventional approaches.

3. DOWNLOADING AND INSTALLING NS-2 AND NAM IN UBUNTU 14.04:

NS-2 works on Linux platform but you can also run it on windows by using Cygwin software.

3.1. Installation steps:

First, download the ns all in one package (ns-allin-one) for ns2 from the link https://sourceforge.net/projects/nsnam/files/latest/download. The name of the downloaded package will be "ns-allinone-2.35.tar.gz". Copy it to the home folder. Then open terminal and use the following two commands to expand the contents of the package.

```bash
cd ~/
tar -xvzf ns-allinone-2.35.tar.gz
```

Using this command all the files will be extracted into a folder called "ns-allinone-2.35". You can also extract the files directly without using this command.

It is important for the NS2 that a few packages to be preinstalled. It also requires the GCC version 4.3 to work properly. So all the package in the NS2 will be installed by using the command mentioned below:

```bash
sudo apt-get install build-essential autoconf automake libxml2-dev
```

One of the reliance declared is the compiler GCC-4.3, which is not available for a long time, and because of that we must need to install GCC-4.4 version. The version 4.4 is the previous version we can get. To install it we need to use the following command:

```bash
sudo apt-get install gcc-4.4
```

Navigate to the folder "link-state", use the following command. Here it is supposed that the expanded ns folder is in the home folder of your system.

```bash
cd ~/ns-allinone-2.35/ns-2.35/linkstate
```

Now we inform the ns which version of GCC will be used. For informing ns we need to go to your ns folder where it is located and type the following command:

```bash
sudo gedit ns-allinone-2.34/otcl-1.13/Makefile.in
```

In the file, change CC= @CC@ to CC=gcc-4.4, as shown in the image below.
Now to install ns2, we first need origin privileges and then the install script can be run. For that use the following two commands:

```
# last line of output
```

The last producer is to inform the system, where the files for ns2 need to be installed or present. For completing the above producer, we have to set the environment path using the ".bashrc" file. In that file, it is very valuable to add a few lines at the bottom. The things to be included are given below.

```
sudo gedit ~/.bashrc
```

Lines to be added:

```
# LD_LIBRARY_PATH
OTCL_LIB=/home/akshay/ns-allinone-2.35/otcl-1.14
NS2_LIB=/home/akshay/ns-allinone-2.35/lib
X11_LIB=/usr/X11R6/lib
USR_LOCAL_LIB=/usr/local/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$X11_LIB:$USR_LOCAL_LIB
# TCL_LIBRARY
TCL_LIB=/home/akshay/ns-allinone-2.35/tcl8.5.10/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIBRARY:$USR_LIB
# PATH
XGRAPH=/home/akshay/ns-allinone-2.35/bin:/home/akshay/ns-allinone-2.35/tcl8.5.10/unix:/home/akshay/ns-allinone-2.35/0/unix
# the above two lines beginning from xgraph and ending with unix should come on the same line
NS=/home/akshay/ns-allinone-2.35/ns-2.35/
NAM=/home/akshay/ns-allinone-2.35/nam-1.15/
PATH=$PATH:$XGRAPH:$NS:$NAM
```

Once the changes have been made, save the file and restart the system. Once the system has restarted, open a terminal and start NS2 by using the following command:

```
akshay@akshay-UBPC:~$ ns
```

Now on executing ns if you get % sign, it means that NS is running and is installed perfectly.

4. SOFTWARE TOOLS USED WITH NS-2:

4.1. NAM:

Nam gives a clear to be seen interpretation of the network configuration developed. As a part of the VINT project, a new application has been developed.

NAM’s features are as follows:

- It gives a visual Elucidation of the network created.
- NAM can be executed directly from the TCL script
- We can control the simulation by different buttons like play, stop, rewind, fast forward, pause, display speed controller and a packet monitor facility.
- It displays information such as throughput and number of packets on each link.
- We can just use drag and drop function to create network topology.
The following figure shows the NAM application window and its components.

4.2. **Nscript:**
Nscript is developed in Java. It is a GUI used for building NS-TCL scripts. The network topology can be built by simply drawing it in the edit screen. The Nodes & agents can be added by using the simple drag and drop function in the edit screen. Once we create the network topology, the TCL code for this topology is automatically generated in the TCL script screen.

The Nscript is used to create different network topologies by simply adding nodes and their respective links. Transport agents like UDP, TCP can be created and added in the topology and the simulation events can be scheduled. The scripts created using Nscript can be exported and executed in NS-2.

5. **TRACE DATA ANALYZING APPLICATIONS:**
The applications used for analyzing trace files produced from the simulation are XGraph and TraceGraph.

5.1. **X-Graph:**
The XGraph is used for lucrative plotting and generating graphs. To use XGraph in NS-2, it should be called within a TCL Script. It will load a graph showing the visual information of the trace file produced in the simulation.

For opening the NAM trace file and the Trace file, following code is executed,
```
#Open the NAM trace file
Set nf [open (Name of your .nam file ) w]
$ ns namtrace-all $nf
```
5.2. Trace Graph:
Trace graph is a good application that comes very easy to NS2 user. It removes the coding of awk scripts that is needed to configure and run over the trace file. Trace graph makes analysis very simple. Trace graph seems to have been developed using MATLAB and therefore supporting codes are needed to make it run in Linux.

Steps for installing Trace Graph:
- Download tracegraph202linux.tar.gz and extract it in home.
- Download mglinstaller.gz and extract it in the tracegraph202 folder.
- We need to run the following command to execute this.
  - $sudo chmod 777 mglinstaller
  - $ ./mglinstaller
  - Open the dependency file from trace graph folder.
  - Run $gedit ~/.bash command in terminal
  - Export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:NS2_LIB:$X11_LIB:your location
  - Type above line in .bash file and save it.
  - Make a terminal as tracegraph202. In our case cd /home/ydjoshi/tracegraph202
  - Type ./trgraph to start trace graph.

6. MANET – Mobile Ad-hoc network:
A MANET has no fixed infrastructure, configured by itself network of wirelessly connected mobile devices. Nowadays, mobile devices, laptops and PDAs are widely used and it is a big market over the time it has been introduced.

As we know that MANET is a multi-hop wireless network. In this network all nodes are connected to each other all the time to make a communication. Their communication and connection is done with one another with the limited bandwidth availability. These type of network is widely used in the area where temporary network is needed to be established and also in the areas where there is no prefixed infrastructure. For example, areas which are affected by natural calamities, where current infrastructure is destroyed. It can be also be deployed in the military applications where temporary strategic network is required.

In MANET, each device has a freedom to move freely in any way and thus will often Switch its links to other nearby devices. Every device must forward the packets that are not for their use so it should be a router. The main problem in building a MANET network is how to equip each router with the information necessary to appropriately route the traffic. These kinds of networks sometimes operate independently or sometimes will be connected to a larger network. Above the link layer, Wireless Ad-Hoc network has a routing networking environment. Peer-to-peer, self-forming, self-healing network are the characteristics of the MANETS.

It is not possible to depend on fixed infrastructure in the battlefield ground. In these situations, setting up the new infrastructure is not practical things in terms of cost and time consumed. Hence, providing the necessary connection between nodes and network services becomes a major issue. Break connection or Links failures may occur due to communication on the wireless medium, or due to mobility of the nodes, when two or more routers move out of the range or if any obstacles come in between them. These network don’t require any centralized administration so can be called as self-configuring.
After 1990, there is a huge research and new products are introduced in the market as a mobile devices and Wi-Fi 802.11. Now-a-days this is a hot and most popular topic for research. There are many academic papers available that assess different available protocols and their capabilities, by varying mobility speed of the nodes within a fix area. Evaluation and comparison of different protocol are based on their performance evaluation such as the packet drop rate, the overhead introduced by the routing protocol, end-to-end packet delays, network throughput, ability to scale, etc.

7. ROUTING PROTOCOLS USED IN MANET:
Routing protocols are used to decide the path of the message packets from source to destination in a network by some set of rules. In MANET, various routing protocols are available and each of them is implemented as per the network circumstances.

7.1. Proactive Routing Protocol:
Proactive routing protocols are likewise called as table driven routing protocols. In this each node have directing table which contains data about the system topology. This component although helpful for datagram activity, causes significant movement and power utilization. The routing tables are updated occasionally at whatever point the system topology changes. Proactive protocols are not proper for large sized network as they must keep the node entries for each and every node in the routing table of every node. These protocols keep various number routing tables changing from protocol to protocol. There are different types of proactive protocols like WRP, DSDV, OLSR etc.

7.2. Reactive Routing Protocols:
Reactive routing protocol is also termed as on demand routing protocol. In this protocol, route is found at whatever point it is required. Route discovery fully rely on the nodes and it is used as per requirement. Source node checks its route cache for the possible routes from the source node to the destination node. If no route is determined, it starts route discovery process. Examples are AODV, LMR, DSR and TORA. It has two major components:
7.2.1. **Route discovery:**
During this phase the source initiates the process of route discovery only when there is a demand. The source node examines its route cache to authenticates which routes are available from source to the destination. If no route is determined, it starts a route discovery process. The packet sent by the source consists of destination node address and the address of the intermediate nodes to the destination.

7.2.2. **Route maintenance:**
As the network has a dynamic topology, sometimes there is a condition of route failure due to the link breakage. So for that route maintenance needs to be done. These protocols consists of acknowledgement mechanism which helps in route maintenance. But because of this mechanism latency gets added in the network. Every node that is involved in the route maintenance mechanism adds latency in the network. So they are generally preferred in the situations where low there is a requirement of low routing overhead.

7.3. **Dynamic Source Routing (DSR):**
DSR is also lies in the reactive type protocol and working methodology is source route method. It works on the principle of the link state algorithm where the source initiates the route discovery only when there is a demand. First the sender will establish the route from source to destination and it also keeps a note of the address of the intermediate nodes and will save these details to the route record of the packet. This protocol was mainly designed for the multi node network having small diameter.

7.4. **Ad Hoc On-Demand Distance Vector Routing (AODV):**
Ad Hoc On-Demand Distance Vector Routing Protocol is working on the principle of fulfilling the demand on order. This will reduce the route broadcast messages as it will provide route on request. When source node requires transferring the packet to the destination it will simply broadcast the route-finding message to its neighbor. Once neighbor receives the broadcasted message it will do the same for transferring the packet to the destination until the packet does not reach the destination. Once the route is defined all nodes saves the route in their route table for acknowledgement to the source node. That node will then reply to the source node. Once source node will get reply from neighboring node, any route-reply after that will be removed by the source node. If network topology changes during the simulation, source node will copy the producer and again broadcast the route request message to carry on the communication in the network. If any link fault is detected, node of that link will simply send an error message to their neighbor node and the process of finding route will start again.

7.5. **Simulation for AODV:**
7.6. Hybrid routing protocol:
As we have seen, the proactive protocols have less latency and large overhead, while the reactive protocols have more latency and less overhead. So to overcome the drawback of both the protocols, hybrid protocol is implemented. It works by using the table maintenance system of the proactive protocol and the route discovery mechanism of the reactive protocol. Thus the problem of large overhead and more latency gets solved. This kind of routing approach is suitable for system having more number of nodes. Here the system is divided into two zones; (1) Inside Zone Routing routing which is of proactive type and (2) Outside Zone Routing which is of reactive type. Zone routing Protocol is commonly known as ZRP. This is one of the Hybrid Routing Protocol.

7.7. Zone Routing Protocol:
As per the name Zone Routing Protocol, routing of the packet or data is done according to the concept of the zones. It is Hybrid Type Protocol which includes the characteristics of both protocol such as Proactive and Reactive Routing Protocol. The main goal behind developing this protocol is to decrease the control overhead of the Proactive Routing Protocol and also reduce the delay of the Reactive Routing Protocol. For all nodes in the network the routing zone is defined separately. The following node is overlapped for the zone of the main node. The zone radius of the node is expressed in terms of the hope to the next node. Zone Routing Protocol is used for the network with large complexity such as large area network with more number of nodes and constantly varying mobility pattern. Let us consider the one example of the Zone Routing Protocol to clearly understand the working pattern of the protocol.

Here node A wants to send packets to L. The radius of each node is set to 2. If the destination node is in the radius of the source node, the source node directly sends the packets to the destination. If the destination is not in the radius then the source node is broadcasting the message of route request to their neighbors. Similarly, the neighbor nodes also follows the same process, if destination is in the radius it forwards the message otherwise it also broadcasts the route request message. As seen from the above figure, L is not in the range of A. So A simply broadcast the route request message. So according to radius declared as 2, A’s requested packets go to node D,
G, F and E. As node L is in the zone of node G, node G broadcast the route request message to node E, A, M and K. Now destination node L is in the radius of node K. So the path for sending packets from the Node A to Node L is A-G-K-L. So route from A-G is done by the proactive routing algorithm as it is in the same zone and path G-K-L is selected by reactive routing protocol as it is in the different routing protocol. The local component of ZRP i.e. a proactive routing component is called as IARP (IntrA-Zone Routing Protocol). And the other reactive routing component is called IERP(IntEr-Zone Routing Protocol). IARP consists of proactive link state protocols and it has only information of the nodes that lies within its routing zone. Similarly, IERP consists of reactive routing protocols and it helps in route discovery and maintenance mechanism on the basis of local connectivity offered by IARP.

7.8. Steps for Patching ZRP file in NS 2.33:
1. After installed ns-allinone-2.33 in ns-allinone-2.33/install file
2.replace line number 546
   ./configure -- with–
   otcl=./otcl-$OTCLVER --with-tclcl=./tclcl-$TCLCLVER || die "Ns configuration failed! Exiting ...");
3. fi
   -if make
   +if make -k
4. then
   echo " Ns has been installed successfully."
5. else
   Paste zrp folder in ns-2.33 folder
6. Paste packet.h in ns-2.33/common folder
8. Paste cmu-trace.cc and cmu-trace.h in ns-2.33/trace folder
9. Paste Makefile.in in ns-2.33 folder
   ./configure
10. In terminal, folder ns-2.35, give command make clean
11. In terminal, folder ns-2.35, give command make
12. In terminal, folder ns-2.35, give command make install
7.9. Simulation of 10 Nodes system using ZRP:

Here radius is set as 3. The path for transferring the packets from source to destination is 7-1-0-9 and 7-1-4-9.
Here radius is set as 1. The path for transferring the packets from source to destination is 7-8-0-9 and 7-1-5-9.
7.10. Comparison of ZRP for 10 and 20 nodes Mobile and Static system using X-Graph:

7.10.1. PDR:
Here for ZRP protocol, we have simulated two systems i.e. a 10 node and a 20 node system having mobile as well as static nature. Using X-Graph, we have compared PDR and End-to-end Delay parameters for 10 node static and mobile and the same for the 20 node system. For each system, we have considered the radius as 1, 3 and 5. From the above graph we can see that PDR is better and delay is less for static system with 10 nodes as well as 20 nodes.
8. COMPARISON OF AODV WITH ZRP FOR END TO END DELAY, THROUGHPUT AND PACKET DELIVERY RATIO USING X-GRAFH:
9. BLACK HOLE ATTACK ON AD-HOC NETWORK (WORKING ON AODV PROTOCOL):

Wireless Ad-hoc Network has lack of infrastructure; it is easily attacked by various hackers or viruses. There are various type of attacks identified in Ad-hoc network. One of the attack is Black-Hole Attack.

In black hole attack hacker or attacker target one of the nodes in the network and try to mess-up the whole communication. In this type of attack, the malicious node absorbs all the data directed towards it; starts dropping the packets and does not allow the data packets to pass through it. Black hole attacker node tries to pretend that only it has the fresh route to transfer data packets. But actually it is a hacker who hacks all packet data and starts dropping it in the network. As route discovery process in Ad-hoc network, source node is finding fresh path to destination with shortest distance and fastest route. At this time attacker node immediately gives response to the source node that it has the fastest and shortest path to destination with secure transmission. By pretending like this in front of source node, source node believes that it has the fresh path and trusts that node. Then it starts transferring all data packets to attacker node and assumes that it will deliver them to destination node as it has a true path for communication. This node intercept all the packet data as well as damage the node interface. In this case other nodes constantly try to find the possible route to destination which makes them to use their battery in addition to dropping packet data.

To perform the black hole attack in the Mobile Ad-Hoc network malicious node waits for the route requested (RREQ) message from the nearby node. As malicious node getting RREQ message, without checking their availability or routing table it will send a fake route reply (RREP) that contains the information about routing to destination over itself by giving a high sequence number to make in the routing table of the source node before any node which is not malicious send their true routing table to source. So source node stop their process of finding suitable route to destination as it assumes that it finds the route for forwarding packets to destination. It
will ignore the RREP message of other node and will start forwarding data to malicious node. So all packets are been delivered to that attacker node from where all the packets are sucks in and they are not being transfered to another node. This is called black hole attack.

In this we are implementing black hole attack in wireless Ad-hoc network that runs on AODV routing protocol. To implement Black Hole Attack we are using Network Simulator 2. We do some coding in TCL language to implement Black Hole scenario. As Network Simulator 2 has inbuilt module for wireless Ad-hoc network, we can easily simulate wireless scenario by coding some program in TCL language. But to simulate black hole attack scenario, we need to have knowledge about C++ because it does not have any module to simulate malicious node. So for changing the node properties, we have to use C++ coding. We modifying the Ad-hoc file in NS2 which is in C++ with .cc extension to simulate Black Hole Attack. We can get adhoc.cc file in the NS2 package.

10. BLACK HOLE ATTACK PREVENTION:
There are many different techniques are used for preventing and detecting the Black-Hole Attack in the AODV based mobile Ad-Hoc network. Some of the common and most popular techniques are as follows:

10.1. DPRAODV Scheme:
The full form of the DPRAODV is Detection, prevention and reactive AODV scheme. In this with AODV, the node which is received the route reply message it will first check the sequence number in the routing table, if the sequence number is higher than the number in routing table it will accept the packets. In DPRAODV solution, if the sequence number is higher than the threshold value it will just consider that node as malicious node. This node is now going in the blacklist. As it is detected working abnormally it sends the alarm packet to its neighbor. As soon as it is detected as a malicious it is neither updates its routing table nor forward the data packets through that node. The threshold value is determined by the average of that routing table entry sequence number for certain period of time. The threshold value is also updated as per the overhead of packets in the network. The threshold value is increased as new node is introduced in the network. The packet delivery ratio is noticeably increased by 80 to 85% with comparing network under attack. In this method as the alarm message is sent in the network the malicious node is isolated or just ignore by the other node in the network.

10.2. Sequence Number Comparison:
This method is similar to the Detection, Prevention and Reactive AODV method. In this method, it will check the sequence number of the source node and the sequence number of the intermediate node who will send the route reply message to the source node first. Here a system is developed for comparing the sequence number. The comparison is done in between the sequence number of the route request method and the sequence number of the route reply method. If there is a huge difference is detected or measured the method considered it as the route reply is coming from the malicious node. This method will just remove that node from the routing table and make it isolated from the network. In this presentation we are using this method to detect the Black-Hole attack. This method is very simple and most effective. This method is not only detecting the black-hole node but also preventing the network from the malicious node.

10.3. Anti-Black-Hole Mechanism:
This method is used to detect and separate the malicious node from the other node and from the network. The node which is selectively performed the black-hole attacks by deploying IDS in the mobile Ad-Hoc network. All Intrusion Detection System nodes execute an ABM which evaluates the doubted value of the node. If the intermediate nodes are not able to reply to the requested node or we can say that the intermediate node which is neither broadcasting a request nor a destination of the packet and sends the route reply message to the source, it will increase its doubted value by 1 in the suspicious node table. There is a limit for this doubted value if this limit is exceeds beyond the threshold, a banned message is broadcasted by the IDS node to all nodes in the network.
network. This is because to isolate the doubted node in the network and make network secured. But we required security mechanism in order to make communication safe between special IDS nodes.

10.4. ERDA (Enhancement Route Discovery for AODV) Scheme:
This method is simply based on the route discovery mechanism with the help of Route reply function. This proposed method is reduced the issues by implementing new conditions in the routing table updating process. This will also include the simple malicious node detection and separation process to the AODV route discovery mechanism. This method does not intercept in the original AODV routing protocol but it is just modified it. It is just introduced some elements in the routing protocol. There are few elements introduced in the modified AODV. (1) Route reply table (2) Keeping the malicious list (3) Route Updating. In route reply table, it will store all the route reply messages. While mali_list is used to keep the detected malicious nodes entries, identities and parameters. Rt_udt is used to updating the route entries every after some period of time. As malicious node immediately reply to the route request done by the source node, source node made a connection with it but before transmitting a packets it will wait for another reply from the another node. It will comparing the both and if the second node has low sequence number compare to malicious node, the new entry in the routing table will update and overwrite on the previous entry. It is very simple method for detecting and preventing black hole attack by simply removing the false entry and replaces it with the new and later reply entry.

10.5. Intrusion Detection System (IDS):
This is one of the primary techniques for the wireless Ad-Hoc network. This technique is used to prevent attack against security threats. This technique is classified in two categories: (1) Network Based (2) Host Based. As in the Mobile Ad-Hoc network there is no any central device or mechanism that monitors the whole network traffic flow. So intrusion detection using anomaly detection use host based intrusion detection system. In IDID it will monitor the system and identify the abnormal activities in the network by comparing it with the normal activities. For this method we need a pre-collected data which behaves like what we expected. Now this data send to IDID system mechanism for comparing the network behavior. The set of expected data is called an audit data. If the audit data is not matched with the data of the activity occur in the network, IDID mechanism simply isolate the node with abnormal behavior from the network. This mechanism is used as the route request is broadcasted by the source and as it is received the route reply message it will compare this reply message with the audit data. If this will match it will keep the route entry in the route table otherwise discarding this entry. Here in our project we have used Sequence Number Comparison method for prevention of Black Hole attack.

10.6. The C++ code for implementing the Scenario with Black hole attack:
```
else if(malicious==1000) {
    seqno=2147483600;
    sendReply(rq->rq_src, // IP Destination
              1,        // Hop Count
              rq->rq_dst,
              seqno,
              MY_ROUTE_TIMEOUT,
              rq->rq_timestamp); // timestamp
    printf("Attacker-node---%d sends seq no is---%d\n",index,seqno);
    //rt-&gt;pc_insert(rt0-&gt;rt_nexthop);
    Packet::free(p);
}
```
As we talked in Black hole attack methodology, the malicious node will send the route reply (RREP) with highest sequence number. So here in this code they have set the seqno of 2147483600.

10.7. Addition of the C++ code for the proposed method for solving Blackhole attack:

```c++
if ( ((rp->rp_dst_seqno - rt->rt_seqno < 100) && (rp->rp_dst_seqno != rt->rt_seqno )) ||
    ((rt->rt_seqno == rp->rp_dst_seqno) && (rp->rp_dst_seqno < 100) && (rp->rp_dst_seqno - rt->rt_seqno < 100)) ||
    ((rp->rp_dst_seqno - rt->rt_seqno < 100) && (rt->rt_hops > rp->rp_hop_count)) )
{
    printf("Node--%d--source seq no--%d dest seq no is %d\n",index,rt->rt_seqno,rp->rp_dst_seqno); //Update the routing table
    rt_update(rt, rp->rp_dst_seqno, rp->rp_hop_count, rp->rp_src, CURRENT_TIME + rp->rp_lifetime);
}
```

Here when each node receives the route reply, it checks src seq no and dst seq no. If the difference is larger than 100, that means the packets come from an attacker node. So, the routing table won’t get updated. The node having difference in the sequence number of less than 100, that node will be involved in routing.

11. SIMULATION FOR SYSTEM UNDER BLACK HOLE ATTACK AND PREVENTING IT:

11.1. With Black-hole attack:

Initially the positions of the nodes are as above. Node 4 is a source node; node 2 is a destination node and node 1 acts as the attacker node called as malicious node.
Then the attacker node sends a RREP with highest sequence number to the sender node 4. So source node starts transmitting data packets to the attacker node and the attacker node sucks in all of the data packets resulting into improper functioning of the network. When the attacker node moves away from the range of source node, packet gets dropped. For this simulation the range of the node is fixed to 100 m. This kind of misbehavior of the system is called Black-hole attack.

11.2. Prevention of Black-hole attack:
Initially the positions of the nodes are same as in the previous scenario.
By using our proposed method, the attacker node gets ignored and the data packets gets transferred from source node 4 to the destination node 2 through node 3 and node 0. Thus, the black hole attack can be prevented by using this method.

12. COMPARISON OF AODV ROUTING PROTOCOL WITH THE SYSTEM UNDER BLACK HOLE ATTACK AND WITH PREVENTION OF BLACKHOLEATTACK (FOR END-TO-END DELAY, PDR AND THROUGHPUT):
Here we have taken three scenarios with 8, 10 and 20 nodes. From the above graphs we can see that delay is maximum, PDR and throughput is minimum for the system under attack. While Delay, PDR and Throughput is almost same for AODV and the system with proposed solution to blackhole attack.
13. 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 AODV and ZRP protocol using NS-2 and have compared the AODV routing protocol with ZRP for some parameters like PDR, End-to-end delay and throughput using X-Graph. Then we have studied the Black hole attack in MANET; have implemented it using AODV routing protocol in NS-2 and have also proposed a solution for preventing it using Sequence Number Comparison method with a simulation in NS-2. With that we have also compared the parameters like Delay, PDR and Throughput for normal AODV, AODV under attack and AODV with prevention of Blackhole using X-Graph.

14. REFERENCES:

- Progress Report 1,2,3,4 and the midterm report for Mobile Computing Fall-2016. Cleveland State University.
- www.wikipedia.org
- www.slogix.in
- www.ripublication.com/irph/ijictSpl/ijictv4n4spl_07.pdf
- www.youtube.com/watch?v=8B_ofFK9Lc