Detection/Removal of Cooperative Black and Gray Hole Attack in Mobile Ad-Hoc Networks

Sukla Banerjee

***Abstract*—The inherent features (such as-open medium, dynamically changing network topology, lack of centralized monitoring and management point, and lack of a clear line of defense) of the MANET make it vulnerable to a wide range of attacks. There is no guarantee that a communication path is free from malicious or compromised nodes which deliberately wish to disrupt the network communication. So protecting the mobile ad-hoc network from malicious attacks is very important and challenging issue. In this paper we address the problem of packet forwarding misbehavior and propose a mechanism to detect and remove the black and gray hole attacks. Our technique is capable of finding chain of cooperating malicious nodes which drop a significant fraction of packets.**

***Index Terms*— Mobile ad-hoc network, Packet forwarding misbehavior, Black hole attack, Gray hole attack.**

I. Introduction

A mobile ad hoc network is a collection of wireless nodes that can dynamically be set up anywhere and anytime without using any pre-existing network infrastructure. It is an autonomous system in which mobile hosts connected by wireless links are free to move randomly and often act as routers at the same time; therefore, the limited wireless transmission range of each node gets extended by multi-hop packet forwarding. This kind of network is well suited for the mission critical applications such as- emergency relief, military operations, and terrorism response where no pre deployed infrastructure exits for communication. Due to its intrinsic nature of lacking of any centralized access control, secure boundaries (mobile nodes are free to join and leave and move inside the network) and limited resources mobile ad-hoc networks are vulnerable to several different types of passive and active attacks[1], [2]. Among these one of the most important security issues is the protection of the network layer from different active routing attacks.

In this paper we tackled two types of routing attacks namely Gray hole attack and Black hole attack which exhibits packet forwarding misbehavior. In a black hole attack malicious node (called black hole) replies to every route request by falsely claiming that it has a fresh enough route to the destination. In this way all the traffic of the network are redirected to that malicious node which then dumps them all. A gray hole attack is a variation of black hole attack, where an adversary first behave as an honest node during the route discovery process, and then silently drops some or all of the data packets sent to it for further forwarding even when no congestion occurs. Detection of gray hole attack is harder because nodes can drop packets partially not only due to its malicious nature but also due to overload, congestion or selfish nature. A selfish node is unwilling to spend its battery life, CPU cycles or available network bandwidth to forward packets not of direct interest to it, even though it expects others to forward packets on its behalf.

In this paper we present a mechanism capable of detecting and removing the malicious nodes launching these two types of attacks. Our approach consists of an algorithm which works as follows. Instead of sending the total data traffic at a time we divide the total traffic into some small sized blocks. So that malicious nodes can be detected and removed in between the transmission of two such blocks by ensuring an end-to-end checking. Source node sends a prelude message to the destination node before start of the sending any block to alert it about the incoming data block. Flow of the traffic is monitored by the neighbors of the each node in the route. After the end of the transmission destination node sends an acknowledgement via a postlude message containing the no of data packets received by destination node. Source node uses this information to check whether the data loss during transmission is within the tolerable range, if not then the source node initiate the process of detecting and removing malicious node by aggregating the response from the monitoring nodes and the network.

The rest of this paper is organized as follows. In section II, we discuss the related work. Network model and assumptions are discussed in section III. We present the methodology and relevant algorithms in section IV. Finally, the conclusion and discussion of future work in section V.

II. Related Work

Marti et al [3] proposed to trace malicious nodes by using watchdog/pathrater. In watchdog when a node forwards a packet, the node's watchdog verifies that the next node in the path also forwards the packet by promiscuously listening to the next node's transmissions. If the watchdog finds the next node does not forward the packet during a predefined threshold time, the watchdog will accuse the next node as a malicious node to the source node; The proposal has two shortcomings: 1) to monitor the behavior of nodes two or more hops away, one node has to trust the information from other nodes, which introduces the vulnerability that good nodes may be bypassed by malicious accusation; 2) The *watchdog* cannot differentiate the misbehavior from the ambiguous collisions, receiver collisions, controlled

Manuscript received July 22, 2008.

S. Banerjee is with the Computer Science & Engineering Department, RCC Institute of Information Technology, Kolkata 700015, WB, INDIA (mob: +919836565440; e-mail: sukla.banerjee@ gmail.com).

transmission power, collusion, false misbehavior and partial dropping. In *pathrater* algorithm each node uses the *watchdog's* monitored results to rate its one-hop neighbors. Further the nodes exchange their ratings, so that the *pathrater* can rate the paths and choose a path with highest rating for routing. Shortcoming of this algorithm is that the idea of exchanging ratings genuinely opens door for blackmail attack.

SCAN [4] exploits two ideas to protect the mobile ad hoc networks: 1) *local collaboration*: the neighboring nodes collectively monitor each other and sustain each other; and 2) *information cross-validation*: each node monitors its neighbors by cross-checking the overheard transmissions, and the monitoring results from different nodes are further cross validated. As a result, the security solution is self-organized, distributed, and fully localized. In SCAN once a malicious node is convicted by its neighbors, the network reacts by depriving its right to access the network by revoking its token. A powerful collusion among the attackers will break SCAN as it violates the assumption of the polynomial secret sharing scheme.

S. Ramaswamy et al presented an algorithm in [5] which claims to prevent the cooperative black hole attacks in ad-hoc network. In this algorithm each node maintains an additional Data Routing Information (DRI) table. Whenever a node (say IN) responded to a RREQ it send the id of its next hop neighbor (NHN) and DRI entry for NHN to the source. If IN is not a trustable node for source then source sends a further route request (FRq) to NHN. NHN in turn responds with FRp message including DRI entry for IN, the next hop node of current NHN, and the DRI entry for the current NHN's next hop. If NHN is trusted node then source checks whether IN is a black hole or not using the DRI entry for IN replied by NHN. If NHN is not trustable node then the same cross checking will be continued with the next hop node of NHN. This cross checking loop will be continued until a trusted node is found. Moreover, in the case when the network in not under the attack, the algorithm takes more time to complete. This algorithm is based on a trust relationship between the nodes, and hence it cannot tackle gray hole attacks.

Gonzalez et al [6] presents a methodology, for detecting packet forwarding misbehavior, which is based on the principle of flow conservation in a network. That states that if all neighbors of a node *vj* are queried for i) the amount of packets sent to *vj* to forward and ii) the amount of packets forwarded by *vj* to them, the total amount of packets sent to and received from *vj* must be equal. They assume a threshold value for non malicious packet drop. A node *vi* maintains a table with two metrics *Tij* and *Rij*, which contains an entry for each node *vj* to which *vi* has respectively transmitted packets to or received packets from. Node *vi* increments *Tij* on successful transmission of a packet to *vj* for *vj* to forward to another node, and increments *Rij* on successful receipt of a packet forwarded by *vj* that did not originate at *vj*. All nodes in the network continuously monitor their neighbors and update the list of those they have heard recently. This algorithm does not require many nodes to overhear each others' received and transmitted packets, but instead it uses statistics accumulated by each node as it transmits to and receives data from its neighbors. Since there is no collaborative consensus mechanism, such an algorithm may lead to false accusations against correctly behaving nodes.

Finally P. Agrawal et al [7] proposed a technique for detecting chain of cooperating malicious nodes (black and gray hole nodes) in ad hoc network. In this technique initially a backbone network of strong nodes (capable of tuning its antenna to short (normal) as well as to long ranges) is established over the ad hoc network. Each strong node is assumed to be a trustful one. These trustful strong nodes detect the regular nodes (having low power antenna) if they act maliciously. With the assistance of the backbone network of strong nodes, the source and the destination nodes carry out an end-to-end checking to determine whether the data packets have reached the destination or not. If the checking results in a failure then the backbone network initiates a protocol for detecting the malicious nodes. For detecting malicious node strong node associated with source node broadcast a find chain message to the network containing the id of the node replied to RREQ. On receiving find chain message strong node associated with destination node Initialize a list GrayHole Chain to contain the id of the node replied to RREQ. It then instructs all the neighbors of that node to vote for the next node to which it is forwarding packets. If the next node id is null then the node is a black hole node. Then the gray hole removal process is terminated and a broadcast message is sent across the network to alert all other nodes about the nodes in GrayHole Chain to be considered as malicious. Else strong node will elect the next node to which replied to RREQ is forwarding the packets based on reported reference counts. Then again broadcast the find chain message containing the id of the elected node. The main disadvantages of this algorithm are the difference between the regular node and backbone node in the network in terms of power, antenna range which makes it unsuitable for all types of mobile ad hoc network. Also it is not proved that backbone network is optimal in terms of minimality and coverage. Algorithm will fail if the intruder attacks strong nodes because it violates the assumption that strong nodes are always trusted node.

.

III. Network Model and Assumptions

We suggest to use a reliable MAC protocol such as IEEE 802.11, MACA (Multiple Access with Collision Avoidance) or MACAW (MACA for Wireless LANs) so that the problems (such as ambiguous collisions, receiver collisions, and the ability of a node to control its transmission power) that arise when overhearing other nodes' transmissions do not exist in our approach. .

The goal of our algorithm is to detect malicious dropping of data packets by an intruder node. In our approach each node in the route is monitored by its neighbors. Neighbors counts the no of data packets forwarded by the node (say *dataCount*) and on receiving query message from the source which contains no of packets actually sent by the source (say n_i) neighbors of each node check if ($dataCount \neq n_i$) then it replies to source via a result message. Now the problem is that mobile ad hoc networks are resource limited. So nodes may drop packets due to

overloaded, lack of CPU cycles, buffer space or bandwidth to forward packets. For these the above straight forward comparison cannot be applied in a rigorous manner. Therefore we assume a threshold probability of packets dropped by a node through no fault of its own.

Let μ be the threshold probability of non malicious packet drop by each node then each monitor node check if $(n_i(1-\mu) \le dataCount)$ then it is not a suspected node. In our algorithm source node will issue a query message to detect malicious node only when it found that no of packets received by destination (say d_count) is significantly less than the no packets actually sent. If the threshold probability of non malicious packet drop at source node is $\overline{\mu}$. Then source will start gray/black hole removal process only if $(d\_count < n_i(1-\overline{\mu}))$ can be estimated from μ as follows. If the non malicious data loss at first node in the route is μ then the volume of data actually forwarded by the node to the next node is $n_i(1-\mu)$. Similarly if at the next node data loss is μ then the next node actually forwards $n_i(1-\mu)(1-\mu)$ volume of data. So at the destination total data loss due to non malicious packet drop is $\left(n_i - n_i(1-\mu)^N\right)$, where N is the total number of nodes in the route. Therefore,

$$\overline{\mu} = 1-(1-\mu)^N \qquad (1)$$

IV. Methodology

The main idea behind this method is to formulate a list of malicious nodes locally at each node whenever they act as source node. The behavior of each node in the route is monitored by all the neighbors of that node. We employ the idea of dividing the total traffic volume into a set of small data blocks [7] so that the malicious nodes can be captured in between the transmission of two such blocks. We choose a window size *w* which is used to determine the total no. of such data blocks say k. Before starting the transmission of the data packets from the first block source node (say S) sends a prelude message to the destination node (say D). On receiving prelude message destination will be alert of the incoming data packets. So destination node sets a timer for the end of the incoming transmission and start counting the no. of the data packets received. After the timer expired it sends a postlude message to the source containing the no. of data packets received by it. On the other hand after sending prelude message source node broadcasts a monitor message to all its neighbors instructing them to monitor the action of the next node in the route and start transmitting data. After finishing the transmission source node sets a time out for the receiving of the postlude message. If source node received a postlude message before the timeout expire and the no. of the data packets received by destination is same as the no. of data packets sent by source or the data loss is within tolerable range then source starts the transmission of the next data block. Else it starts detection and removal of the malicious nodes in the route. Here we have assumed a threshold data loss rate μ at each node and total data loss rate threshold $\overline{\mu}$ which can be estimated from μ as shown in equation (1) of the previous section. Selection of the value of μ plays an important role in the detection power of our proposed algorithm, i.e. the capability of the algorithm to detect misbehaving nodes. The lower the μ is the more likely it is that our algorithm detects any malicious behavior. However, it also means that the probability of a false detection will increase with the lower value of μ . Also it should be taken into account the total data loss rate should not be higher otherwise source node will not invoke the process of malicious node detection at all. We suggest to assume the maximum value of $\overline{\mu}$ first, depending on the path length (which is the hop count for the route in AODV routing), then from $\overline{\mu}$ to estimate the value of μ .

On receiving the monitor message neighbors of the source node checks whether it is the neighbor of the next hop node in route or not. If it is neighbor of the hop node in route then it starts monitoring the action of the node. It first initializes a counter to count the no. of the data packets forwarded by the node also infer the id of the next node to which it is forwarding the data. To do so monitor nodes can maintains a copy of the neighbor's routing table and determines the next-hop node to which the neighbor should forward the packet; if the packet is not overheard as being forwarded, it is considered to have been dropped. Also the monitor nodes again broadcast a monitor message to all its neighbors containing the id of the next node to which this node is forwarding the data, instructing them to monitor the action of the next node. This process will continue until the next node is the destination node. If the receiving node of the monitor message is not the neighbor of the next hop node in route it simply forward the message to all its neighbors.

Whenever a source node wish to initiate the gray/black hole detection and removal process it broadcasts a query message to all its neighbors and sets a time out for the receipt of the result message from the monitoring nodes. When the timeout not expired each time a result message or the node is malicious message is received for any node source node will append that node in its findMalicious Table and initialize the voteCount as 1 if it is not already there, otherwise increments its voteCount by 1 and check if voteCount is greater than a predefined thresholdCount or not. If greater, then source node will remove that node from the findMalicious table and enter it into the Black/Gray Hole table. Broadcasts that the node is malicious to the network and modify the malicious status of that route by setting the findHoleStatus as true for that route in its routing table. When the timeout expired source node will start voting for the nodes left in the findMalicious table. It broadcasts vote request message to the network containing the id of each node in the findMalicious table one by one. Sets a timeout for the receipt of the vote reply and on receiving a reply voteCount is incremented by 1. Check if the voteCount is greater than a predefined thresholdCount remove that node from the findMalicious table and enter it into the Black/Gray Hole table. Also broadcasts that the node is malicious to the network and modify the malicious status of that route by setting the findHoleStatus as true for that route in its routing table. Finally the source node checks the findHoleStatus of the

route and if it is true then it terminates sending data until it finds a new route to the destination. If it is not true then it retries sending data of the same block.

In the above process source node actually elect the malicious node from the result messages sent by the neighbors based on the reference thresholdCount for both result if the node is voted as malicious by the neighbors or suspected as malicious by neighbors. By doing so we are avoiding the chance of accusing a legitimate node as malicious node by colluding neighbors. Also the vote method from the network enhances the possibility of detecting a really malicious node which is voted as legitimate by the colluding neighbors by not replying to the query message. Our methodology is based on the assumption that a neighborhood of any node in the ad hoc network has more trusted nodes than malicious nodes.

On receiving a query message monitoring nodes checks if the no. of data packets forwarded by the node under monitor is same as the no. of data sent to it or the data loss rate is within the tolerable range (determined by μ). If so then it simply broadcast the query message to all its neighbors by replacing the node id to be queried as the next node id to which the monitored node is forwarding the data packets and no. of data packets sent to next node by the data count of the monitoring node. Else monitoring nodes checks if the next id to which the monitored node is forwarding the data packets is NULL then it infers that the monitored node is a black hole node and replies to source as monitored node is malicious. If the next node id is not NULL monitoring nodes replied to the source that monitored node is suspected as malicious node by sending result message to the source. Also it again generate a further query message by replacing the node id to be queried as the next node id to which the monitored node is forwarding the data packets and no. of data packets sent to next node by the data count of the monitoring node and broadcast them to all its neighbors to check if there is any other cooperative malicious nodes exists or not. All the replies to the source are traversed through a reverse path of the query message; therefore, the need for broadcast messages will be minimized.

On receiving a vote request for any node a regular node in the network check their Black/Gray Hole table. If an entry for that node is found it replies to the source node (i.e. the generator of the vote request) via a vote reply message. Here we assume that if the node is not a newly joined node then there is a possibility that node has traversed from the different region of the network. So any other node in the network may have used this node for forwarding traffic and found it as malicious.

On receiving a node is malicious message all regular nodes in the network first check if they already have an entry for the node in their Black/Gray Hole table. If not then they make and entry for that node in their findMalicious table and initialize voteCount as 1. If the node already exists in any of the above tables ignore the message. We are doing so because if we black list the node or increment its voteCount then there is a chance of completely banning a legitimate node from the network by false probing.

Here in our method we propose to modify AODV protocol by introducing three more tables maintained at each node. First one is DRI (Data Routing Information) table maintained at each node for the purpose of monitoring each of its neighbors. Another table is the findMalicious table which keeps the track of the nodes suspected as malicious with their voteCount. And the Black/Gray hole table which keeps the track of the black listed nodes. We also modified the routing table of the AODV by adding a new field called findHoleStatus which is set as true if a malicious node s found in the route. With the help of the following Fig.1 which shows a current network topology each of the above tables are depicted below.

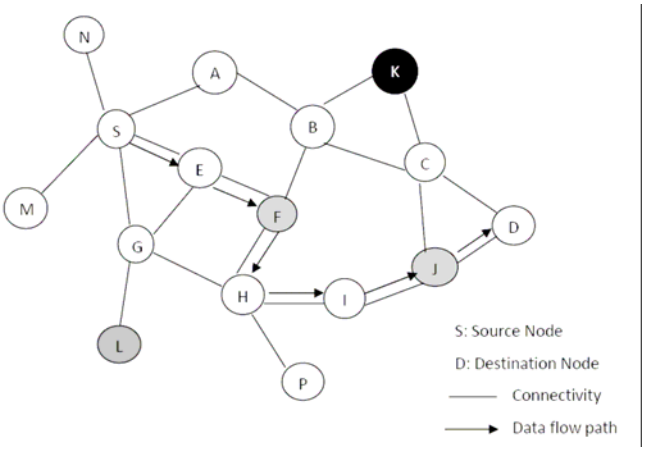

Fig.1: Current Network Topology

Table 1: Data routing table at S

DESTINATION NODE ID	ROUTE	findHoleStatus
D	E, F,H,I,J	False
P	A, B, F, H	False
J	G, H	False

Table 2: List of Neighbors maintained at S

NEIGHBOR NODE ID
E
G
A
M
N

Table 3: Data routing information table maintained at node B for monitoring neighbors

MONITORED NODE ID	NEXT NODE ID	DATACOUNT
F	H	5
K	NULL	0

Table 4: FindMalicious table maintained at S

NODE ID	VOTE COUNT
F	2
J	1

Table 5: Black/Grav Hole Table Maintained at S

NODE ID
L
K

Pseudo code of our algorithm is as follows.

Algorithm for Detecting Gray/Black Hole

Action by Source Node S

Step 1: Divides the data packets to be sent in **k** equal parts.
DATA [1,….,K];
Initialize **i = 1;**
Comment: Chose window size **w**, If total no of data packets **n** then **k = ceiling (n/w)**
Step 2: Send ***prelude(S,D,n_i)*** message to the destination node **D.** Where $\mathbf{n_i}$ is the no of data packets to be sent in current block.
Step 3: Broadcast ***monitor (S, D, NNR)*** message to all its neighbors. Instructing neighbors to monitor next node in the route (**NNR**).
Step 4: Starts transmitting data packets from the block **Data[i]** to **D.**
Step 5: Sets timeout $\mathbf{T_S}$ for the receipt of the ***postlude (D, S, d_count)*** message containing ***d_count***, no of data packets received by D.
Step 6: If $\mathbf{T_S}$ not expired and ***postlude*** message received, if $\left(n_i\left(1-\overline{\mu}\right)\le d\_count\right)$
Increment **i** by **1** and go to **Step 8**.
else Start Gray/Black hole removal process.
Comment: Where $\overline{\mu}$ is a threshold value ranges between 0 and 1 indicates the fraction of total packets gets lost due to error prone wireless channel. If we assume that μ is the permissible packet loss in each node in the route then $\overline{\mu}=1-(1-\mu)^N$, where N is the total no of nodes in the route (hop count).
Step 7: If $\mathbf{T_S}$ expired and ***postlude*** message not received then start Gray/Black hole removal process.
Step 8: Continues from **Step 2** when **i** less than equal to **k.**
Step 9: Terminates **S**'s action.

Action by Destination Node D

Step 1 On receiving ***prelude(S,D,n_i)*** message from **S** extracts $\mathbf{n_i}$
Initialize **d_count = 0.**
Step 2: Sets timeout $\mathbf{T_D}$ for the receipt of the current data sample and waits for the data packets.
Step 3: When $\mathbf{T_D}$ not expired and a data packet received
Update **d_count += 1**
Step 4: When $\mathbf{T_D}$ expired send ***postlude(D, S, d_count)*** message to **S.**
Step 9: Terminates **D**'s action.

Action by neighbors On receiving *monitor (S, D, NNR)* message

Step 1 On receiving ***monitor (S, D, NNR)*** message nodes extracts the id of the next node in the route **NNR**, source node id **S** and destination node id **D**.

Step 2: If the receiving node is neighbor of **NNR** then,
Step 2.1: Turn on Promiscus mode**.**
Step 2.2: Initialize $\mathbf{dataCount_{NNR} = 0}$.
Step 2.3: Find next node id $\mathbf{N_{next}}$ to which **NNR** is forwarding the data packets.
Step 2.4: start counting data packets by incrementing $\mathbf{dataCount_{NNR}}$ **+= 1.**
Step 2.5.: If $\mathbf{N_{next}}$ is not destination node **D** then
Step 2.5.1: Broadcast ***monitor (S, D, NNR)*** message to all its neighbors replacing **NNR** by $\mathbf{N_{next}}$.
Step 3: Else Rebroadcast ***monitor (S, D, NNR)*** message to all its neighbors.
Step 4: Terminates its action.

Gray/Black Hole Removal process

Action by Source Node S

Step 1: Broadcast ***query(S, D, N_{RREP}, n_i)*** message to all its neighbors. Where $\mathbf{N_{RREP}}$ is the id of the node sending route reply message to **S**.
Step 2: Sets timeout $\mathbf{T_{RES}}$ for the receipt of the ***result (MN, S, N_{RREP})*** message from the monitoring node **MN.**
Step 3: When $\mathbf{T_{RES}}$ not expired and ***result*** message received or ***"N_{RREP} Malicious"*** received then extracts $\mathbf{N_{RREP}}$.
Step 3.1 If $\mathbf{N_{RREP}}$ already exists in **FindMalicious** table
Step 3.1.1: Then increment **voteCount** for $\mathbf{N_{RREP}}$ by 1.
Step 3.1.2: If **votecount >= thresholdCount**
Step 3.1.2.1: Remove $\mathbf{N_{RREP}}$ from **FindMalicious** table and append $\mathbf{N_{RREP}}$ in **Gray/BlackHole** table.
Step 3.1.2.2: Broadcast ***"N_{RREP} Malicious"*** to the Network.
Step 3.1.2.3: Set **findHoleStatus = true** in the routing table of S for the route to **D.**
Step 3.2: Else
Step3.2.1: Append $\mathbf{N_{RREP}}$ in **FindMalicious.**
Step 3.2.2: Initialize **voteCount** = 1.
Step 4: Initialize **j = 1**.
Step 5: When **j** <= length of **FindMalicious** table
Step 5.1: Broadcast ***VREQ(S, N_j)*** to the network requesting other nodes in the network to vote for $\mathbf{N_j}$ if it is malicious.
Step 5.2: Sets timeout $\mathbf{T_{VREP}}$ for reply from the network ***VREP(RN, S, N_j)*** where **RN** is id of any regular node in the network.
Step 5.3: When $\mathbf{T_{VREP}}$ not expired and ***VREP*** message received then
Step 5.3.1: increment **voteCount** for $\mathbf{N_j}$ by **1.**
Step 5.4: If **voteCount >= thresholdCount**

Step 5.4.1: Remove $\mathbf{N_{RREP}}$ from **FindMalicious** table and append $\mathbf{N_{RREP}}$ in **Gray/BlackHole** table.
Step 5.4.2: Broadcast ***"N_{RREP} Malicious"*** to the Network.
Step 5.4.3: Set **findHoleStatus = true** in the routing table of S for the route to **D.**
Step 5.5: Increment **j** by **1.**
Step 6: If **findHoleStatus** is **True**
Step 6.1: Terminate sending data. Find new route to **D**.
Step 7: Resume its normal action.

Action by Neighbors on receiving on receiving *query(S, D, N_{RREP}, n_i)*message

Step 1: On receiving ***query(S, D, N_{RREP}, n_i)*** message nodes extracts $\mathbf{N_{RREP}}$ (id of the node sending route reply message to **D**), **S**, **D** and $\mathbf{n_i}$(no of data packets sent to D).
Step 2: If the receiving node is neighbor of $\mathbf{N_{RREP}}$ then,
Step 2.1: If $n_i(1-\mu) \le dataCount$
Step 2.1.1: when $\mathbf{N_{next}}$ is not **D**
Step 2.1.1.1: Broadcast ***query(S, D, N_{RREP}, n_i)*** message to all its neighbors replacing $\mathbf{N_{RREP}}$ by $\mathbf{N_{next}}$.
Step 2.2: Else
Step 2.2.1: If $\mathbf{N_{next}}$ equals to **NULL** then $\mathbf{N_{next}}$ itself droping all the packects
Step 2.2.1.1: Reply ***"N_{RREP} Malicious"*** to S.
Step 2.2.2: Else
Step 2.2.2.1: Reply ***result (MN, S, N_{RREP})*** to **S,** which means $\mathbf{N_{RREP}}$ may be malicious.
Step 2.2.2.2: Broadcast ***query(S, D, N_{RREP}, n_i)*** message to all its neighbors replacing $\mathbf{N_{RREP}}$ by $\mathbf{N_{next}}$ and $\mathbf{n_i}$ by **dataCount** for $\mathbf{N_{RREP}}$**.**
Step 3: If the receiving node is not neighbor of $\mathbf{N_{RREP}}$ then broadcast ***query(S, D, N_{RREP}, n_i)*** message to all its neighbors.
Step 4: Terminates its action.

Action by any regular nodes (RN) on receiving on receiving *VREQ(S, N_j)*message

Step 1 On receiving ***VREQ(S, N_j)*** message nodes extracts $\mathbf{N_j}$
Step 2: If $\mathbf{N_j}$ exists in **Gray/BlackHole** table
Step2.1: Reply ***VREP(RN, S, N_j)*** to **S**.
Step 3: Terminates its action.

Action by any regular nodes (RN) on receiving on receiving "N_{RREP} Malicious"

Step 1 On receiving **"$\mathbf{N_{RREP}}$ Malicious"** all regular nodes in the network check **Gray/BlackHole** table**.**
Step 2: If $\mathbf{N_{RREP}}$ not exists in **Gray/BlackHole** table, then
Step 2.1: If $\mathbf{N_{RREP}}$ not exists in **FindMalicious** table.
Step 2.1.1: Append $\mathbf{N_{RREP}}$ in **FindMalicious** table.
Step 2.2.2: Initialize **voteCount = 1**.
Step 3: Terminates its action.

V. Conclusion and Future Work

In this paper we have studied the work that attempt to detect black or gray hole or cooperative black and gray hole attack. Finally we proposed a feasible solution for detection and removal of chain of cooperative black and gray hole attack in AODV protocol. In our solution each node can locally maintain its own table of black listed nodes whenever it tries to send data to any destination node and it can also aware the network about the black listed nodes. This list of malicious nodes can be applied to discover secure paths from source to destination by avoiding multiple black/ gray hole nodes acting in cooperation. As future work, we intend to develop simulations to analyze the performance of the proposed solution based on the following metrics.

Throughput: This is the percentage of sent data packets to the actually received by the intended destination.

Overhead: This is the ratio of routing-related transmissions (ROUTE REQUEST, ROUTE REPLY, ROUTE ERROR, and QUERY, MONITOR, RESULT, VREQ, VREP) to data transmissions in a simulation. Some routing packets are more expensive to the network than other packets: ROUTE REQUEST, QUERY, MONITOR packets are broadcast to all neighbors which in turn broadcast to all of their neighbors, causing a tree of packet transmissions. Unicast ROUTE REPLY, ROUTE ERROR, RESULT, and VREP packets only travel along a single path.

Effects of the *false positives* on network throughput: False positives occur when the our proposed mechanism reports that a node is misbehaving when in fact it is not. We plan to study the impact of this on throughput.

References

[1] "Security Issues in Mobile Ad Hoc Networks- A Survey" Wenjia Li and Anupam Joshi, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County.

[2] Wireless/Mobile Network Security, Y. Xiao, X. Shen, and D.-Z. Du (Eds.) pp. 170 – 196,2006 Springer, "A Survey on Intrusion Detection in Mobile Ad Hoc Networks" Tiranuch Anantvalee.

[3] S. Marti, T. J. Giuli, K. Lai, and M. Baker, Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. *Proceedings of the 6th annual international conference on Mobile Computing and Networking (MOBICOM)*, Boston, Massachusetts, United States, 2000, 255-265.

[4] H. Yang, J. Shu, X. Meng, and S. Lu, "SCAN: Self-organized network-layer security in mobile ad hoc networks," IEEE Journal on Selected Areas in Communications, vol. 24, issue 2, pp. 261-273, February 2006.

[5] S. Ramaswamy, H. Fu, M. Sreekantaradhya, J. Dixon, and K. Nygard. Prevention of cooperative black hole attack in wireless ad hoc networks. In Proceedings of 2003 International Conference on Wireless Networks (ICWN'03), pages 570–575. Las Vegas, Nevada, USA, 2003.

[6] Oscar F. Gonzalez, Michael Howarth, and George Pavlou, Detection of Packet Forwarding Misbehavior in Mobile Ad-Hoc Networks. Center for Communications Systems Research, University of Surrey, Guildford, UK. Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium on May 21, 2007.

[7] Piyush Agrawal, R. K. Ghosh, Sajal K. Das, Cooperative Black and Gray Hole Attacks in Mobile Ad Hoc Networks In Proceedings of the 2nd international conference on Ubiquitous information management and communication, Pages 310-314, Suwon, Korea, 2008.