Journal of Information Assurance and Security.
ISSN 1554-1010 Volume 7 (2012) pp. 89-101

A Cooperative Intrusion Respond for Clustered MANETs

Hajar Al-Hujailan, Mznah Al-Rodhaan and Abdullah Al-Dhelaan

Computer Science Department, College of Computer & Information Sciences, King Saud University
Riyadh, Saudi Arabia
{halhujailan, rodhaan, dhelaan}@ksu.edu.sa

***Abstract*: One of the security techniques is intrusion detection system which provides a second line of defense. This system concerns with detecting malicious nodes and performing the response which could be on detected node's level or on network level. In this paper, we propose an efficient response system based on a cooperative scheme to deal with intrusions in clustered mobile ad hoc networks and work in the two levels. Our proposed system provides security against all network attacks that can be detected by any node in the network, in particular detects the actor. It is simple, reliable, and effective with no affect of channel status on the performance. The results of simulation illustrate that the scheme works well and there is a remarkable decrease in false negative and positive rates.**

***Keywords*:** Wireless network, MANET, cluster, security, intrusion detection, IDS, attack, respond.

I. Introduction

The Internet has been evolved nearly half a century [1]. A Mobile Ad hoc NETwork (MANET) is a collection of wireless mobile devices communicating with each other and forming a temporary network, without any pre-deployed infrastructure. Number of applications benefit from this kind of networking, such as military or police exercises, Urgent Business meetings, disaster relief operations, personal area network, conferences and mine site operations [2, 3]. But the security and communication reliability is a main requirement for such applications especially in the military field.

Security in MANET is a main and important element for the basic functions of a network such as routing, packet forwarding, and network management. In fact, network operation can be endangered because of the nature of networks. In mobile ad hoc network the basic functions are done by every participant node in the network, unlike networks that use special nodes to support the basic functions. This difference causes many important problems of the security, which are specific to this type of networks. So, we cannot insure the behaviors and the intentions of ad hoc network nodes when they execute critical network functions such as routing on the contrary in classical networks.

There are questions about the reason for the existence of intrusion detection. There is an opinion considers that the intrusion prevention [4, 5] is enough. But the logical opinion is "intrusion preventive measures such as encryption and authentication can reduce intrusion but not eliminate them" [6]. The nature of MANETs is vulnerable, and malicious nodes may compromise the network by carrying private keys or dropping the packets [7], and make the network more vulnerable. These types of attack cannot be prevented by encryption and authentication. Hence if intrusion prevention is the first line of defense, intrusion detection provides a second line of it.

There are two major parts of processes fall under the intrusion detection system (IDS). They are *detection process* and *response process*. Every one of them consists of three agents as shown in Figure 1. Our scheme includes response process with all agents under it. We add additional agent which is "*cooperative collection*". The new agent is for reducing a False Positive Rate. There are number of response types, they depend on the type of attack [8]. For every attack, the network needs one or more of the response types to recover from it. For example, here are two common responses [8]:

- Reinitializing communication channels.
- Reorganizing the network to isolate the malicious nodes.

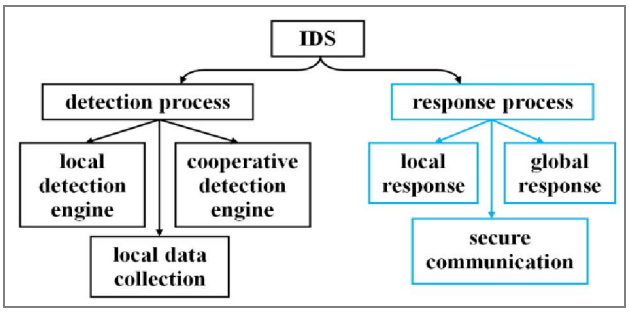


Figure 1. Structure of Intrusion Detection System

Local response: The response done by each node that knows about the misbehavior.

Global response: The response done by whole network against a malicious node.

Secure communication: The actions done to keep the malicious node isolated and does not affect the network.

Cooperative collection: The actions done cooperatively to make an exact decision against a misbehaving node.

Some detection schemes in the literature [9, 10, 11, 12] do not emphases on the isolation process, no alarming mechanism, specific to limit number of attacks, depend on judgment from one node which might be an attacker, or not designed to clustered networks. This problem motivates us to provide scheme that outdo most the weaknesses. We are aiming at simple, reliable and effective scheme that provides security against almost all network attacks that can be detected by any node in the network, in particular detects the actor. Its performance not affected immediately by status of channel. It starts when the detection process ends, and does not end until a malicious node is isolated. It uses a warning message to avoid and punish the misbehaving nodes. Margin of error in judgment that the node is guilty and considers a malicious node (False Positive Rate) is very rare; because we put regulations and standards to control this process.

The remainder of this paper is organized as follows: In section II, we review the related work of security in mobile ad hoc networks. Section III presents proposed scheme. Section IV clarifies the experimental results. We conclude the paper in section V.

II. Related Works

The IDS in MANETs is a hot research field. In this section we present some of the proposed systems. We classify them into two subsections: IDSs with and without clustering.

A. IDSs with clustering

There isn't enough research about IDS in clustered MANETs. In this subsection, we find and present the researches that use clustering or similar to clustering concept.

Y. Yao, L. Zhe, and L. Jun design a detection scheme [13] by using two detection techniques to the anomalous basic actions. The techniques are specification violation detection and statistical violation detection. The authors combine these techniques to achieve the advantages of both. They apply their scheme on the Weighted Clustering Algorithm (WCA) to test the scheme. The test results clarify the improvement in the network performance and the veracity in detection with the limitations of working with a hierarchical network structure, and securing the process of clustering only i.e., does not include communication security.

J. Parker, J. Undercoffer, J. Pinkston, and A. Joshi proposed a scheme [14] to enhance the Watchdog technique and their scheme contains two response mechanisms. They are passive and active response modes. In the first mechanism, each node acts independently and eventually the malicious node will be prevented from dealing with any resources available in the network. While the second one, a cluster-head decides and suspects a node. Then the cluster-head creates a voting process. The decision based on a majority decision. If they decide the suspected node is malicious, then an alert will be broadcasted to all nodes the network and the malicious node will be prevented from dealing with any resources available in the network. The simulation results indicate that the performance is improved in both passive and active response mode.

M. Shao, J. Lin, and Y. Lee proposed a defense system [15] which detects intrusions in AODV-based MANETs. It based on a cooperative scheme and uses clustering technique and Back Propagation Network (BPN) [16]. The benefits of a clustering architecture are scalability and fault tolerance. Also, it benefits from back-propagation neural networks in anomaly and intrusion detection. In this system, the response is divided into two types: local response and total partial response. In local response, when the intrusion detection system detects a malicious node, it will write the malicious node id in the field in the hello message. When any node receives the hello message, it omits the malicious node from its routing table to isolate it. Then, the hello message updates the route itself. In total response, there is one node that applies intrusions detection system, and when it finds a malicious node, it will tell them. Simulation results clarify the effectiveness of the proposed scheme. This is reflected in the comparison between it and finite state machine (FSM).

N. marching and R. Datta proposed a collaborative technique for intrusion detection system in MANET [17], which consists of two intrusion detection techniques. The difference between them is whether or not each pair of nodes is within transmission range of each other. In both techniques, the neighbors collaborate to detect a malicious node. In this scheme, the neighborhood is similar to a cluster. There is a monitoring node which concerned with receiving any notification about suspected node. It uses voting messages to collects the information. This technique does not care about response stage, it just detects malicious node, but does not alarm the others. The results present that the malicious nodes detection is very successful and the average false detection is minimal. But when the number of malicious nodes exceeds k (predefined value), often the detection process will collapse. The advantage of this technique is that it is independent of any routing protocol. In contrast, the disadvantages are the amount of exchanged messages in this technique also its detection correctness depends on connection reliability. The correctness of the first technique is mathematically proven. In simulation, the performance of two techniques is good.

B. IDSs without clustering

In this subsection, we present the researches about IDS in MANETs which are related to our subject; even though no clustering is used.

S. Bhargava and D. P. Agrawal design a detection scheme [18] and they use two modules an Intrusion Detection Module (IDM) and an Intrusion Response Module (IRM). The IRM uses a counter $\boldsymbol{C}$ between every two nodes $\boldsymbol{i}$ and $\boldsymbol{j}$, $\boldsymbol{C_{i,j}}$ is for node $\boldsymbol{i}$ and increments when any malicious behavior appears on node $\boldsymbol{j}$. If value of $\boldsymbol{C_{i,j}}$ reaches a predefined threshold value, node $\boldsymbol{i}$ spreads the alarm about the node $\boldsymbol{j}$. The implementation results clarify the effect of this scheme in performance increasing and overhead decreasing.

L. Bononi and C. Tacconi proposed IDS [19] which contains two main components: detection and reaction. In the detection of malicious behavior, it is based on the assistance of neighboring nodes, with passive reactions. The passive reaction means there is no alarm or warning about detected malicious node in the reaction. The node that discovered the corruption is counting a malicious behavior as IRM of [18]. When this counter exceeds a threshold value, the discovering node excludes the malicious node. But it does not warn the others. Simulation results clarify the effect of the proposed

scheme in isolating malicious nodes.

M. Su and K. Chiang design an approach for detecting and isolating wormhole nodes [5] by deploying a node implementing IDS in MANETs. When IDS detects a wormhole node, it broadcasts a block message to all MANET nodes. The IDS node only can broadcast a block message. Every node has a block table contains the malicious nodes that will be isolated. The simulation results show that the scheme can rapidly and correctly block the wormhole nodes but the correctness is affected when the false positives increase.

JML. Manickam and S. Shanmugavel propose a Resiliency Oriented Secure (ROS) routing protocol for MANET [20]. This protocol adds security to Ad hoc On-demand Distance Vector (AODV) routing protocol. Every node suspects a routing packet confirms with its neighbors. It uses update time interval that stored in its Route Table to discover any misbehaver. If it is sure that the node is malicious, it will add its identifier to its Malicious List without any alarm to other nodes in the network. The results clarify that the proposed protocol has better performance than AODV routing protocol. It has been tested under attack and it became clear its performance is higher in terms of routing overhead ratio, delivery ratio, average route acquisition latency, and average end-to-end delay.

S. Madhavi and T. H. Kim designed a scheme [21] that contains a monitoring node whose responsibilities as in [17]. It also detects the packet dropping and packet delaying attack. In this scheme, there are two types of the responses: global response and end-host response. In the global response, the malicious node is isolated in the network. While in the end-host response, every node makes its own reaction based on well-behaviors and misbehaviors of the malicious node.

A. Al-Roubaiey, T. Sheltami, A. Mahmoud, E. Shakshuki, and H. Mouftah designed a new scheme, named: Adaptive ACKnowledgment (AACK) [22], which is an enhancement to the TWOACK scheme. The TWOACK depends on the Watchdog technique as are most of the current intrusion detection systems for mobile ad hoc networks. If the detection system discovers a malicious node, it tells the response system. Each node can detect a misbehaving node and deal with response procedures. In fact, each node has an array to register a behavior type of nodes that deal with it, either good or bad behavior. When any node in the path routing discovers a malicious node, it sends an alarm to the source. The alarm contains a malicious node id. When the alarm passes through a path routing, it tells the visited node about the malicious node. The results show that this scheme is better than both of the TWOACK and Watchdog schemes. Its overhead is less than theirs and it achieves more packet delivery ratio.

S. Ganapathy, P. Yogesh, and A. Kannan proposed a system [23] that uses a combination of tree classifier and a multiclass Support Vector Machines (SVM) algorithm for detecting the intruders in MANETs. The SVM used to classify the attacks effectively and to detect them. The system can classify the four types of attacks: Probing, Denial of Service (DoS), User to Root (U2R), and Root to Location (R2L) attacks. In this work, the focus was on increasing the detection accuracy and reducing in false positive rates. So, it stops at the detection phase and does not address the response phase. The result is amazing; the classification accuracy for all attacks is very high.

S. Bu, F. R. Yu, P. X. Liu, H. Tang, and P. Mason proposed a distributed scheme [24] which combines authentication and intrusion detection, i.e., combines prevention and detection. The scheme chooses the appropriate biosensors or intrusion detection system based on the security needs and energy states; some biometric authentication computation needs high energy. The distributed multimodal biometrics and intrusion detection system scheduling process have two parts: off-line and on-line. The purpose of the partition is reducing the computational complexity. The sensor which collects and computes the indices will broadcast the new computed indices to the other nodes whether these indices indicate a malicious node existence or not. The results present an improvement in network security.

All previous work can be one of these cases, either they built their IDS on a cluster-less network, or build the IDS based on specific type of clustering algorithm i.e., the proposed IDS suits one of clustering algorithms and does not applicable to all or at least some of them. Also, the proposed IDS has been built sometime on clustering basis, but the clustering is used just for security and does not benefit from it in other functions. Another case, some IDS detects just one type of attack. In the most previous works only one alarm (or notification) was taken seriously, while the malicious node itself maybe notifies about normal node as misbehaving. The last case named the false misbehaving problem. We summarized the previous works in Table 1.

III. The Proposed Scheme

In this section we present our work. The first subsection clarifies the data structure that we use in the proposed scheme. Second subsection explains our assumptions. In the third subsection, we present our scheme.

At the beginning, we will clarify important points about our scheme. It overcomes some of the drawbacks that founded in the literature. It is general and not constrained a specific attack. It uses cooperative concept in decision making and in alarm. It allows all nodes to participate in the decision, but the final decision is assigned to the heads.

A. Data Structures

The proposed scheme needs some tables and packets. In two subsections below, we explain them in detail.

1) Tables

We use three types of tables which are "PACKET_TABLE", "HEAD_MALICIOUS_TABLE" and "MALICIOUS_TABLE", as shown below.

(a) *HEAD_MALICIOUS_TABLE:* Every HEAD of a cluster in the network has this table which consists of four columns as shown in Figure 2(a). The sender_id column contains identifier of a sender node which sends a notification to the head. The malicious_id column contains identifier of

Reference No.	Feature					
	clustering	***generality***	***cooperative***	***alarm***	***detection***	***affect by***
13	yes	hierarchical structure	Yes	no	monitor	
14	yes	Watchdog	Yes	yes	all nodes	
15	yes	AODV	Yes	yes	all nodes	
17	similar	general	Yes	no	monitor	1- no. of malicious node 2- connection reliability
18	no	general	Yes	yes	all nodes	
19	no	general	Yes	no	all nodes	
5	no	Wormhole attack	No	yes	monitor	increasing in false positives
20	no	AODV	Yes	no	all nodes	
21	no	general	Yes	yes	monitor	
22	no	TWOACK	No	some of node	all nodes	
23	no	Probing, DoS, U2R, and R2L attacks	No	no	all nodes	
24	no	sensors	Yes	yes	all nodes	
25 “our scheme”	yes	general	Yes	yes	all nodes	

Table 1. Summary of the related works features.

suspected node. The attack column contains type of attack that suspected node do. The dangerous column contains the degree of dangerous.

(b) MALICIOUS_TABLE: Every node (head or not) in the network has this table, and the node does avoid any node recorded in this table, i.e., ignores all nodes in the table as they leave the network. It consists of three columns as shown in Figure 2(b). The malicious_id column contains identifier of malicious node. The sure column is Boolean, '0' means the node itself decides but the HEAD does not confirm yet (the default), while '1' means the HEAD decides as a collective decision. The time column contains the current time when the record is written and may contains 'null' value.

(c) PACKET_TABLE: Every node (head or not) in the network has this table which consists of two columns as shown in Figure 2(c). The packet_id column contains identifier of the packet. The type column contains the type of the packet. When the node receives any packet, it determines its type from this table.

2) Packets

We use six types of packets, "*notification*", "*warning*", "*acknowledgment*", "*new_request*", "*new_info*" and "*new_complete*" as shown below.

(a) "*notification*": This packet is used to notify the HEAD about a malicious node, it shown in Figure 3 (a). When any node detects a malicious node, it writes the ID of a malicious node in malicious_id field, and type of its attack in attack field. The first field is used to identify type of the packet. The node that generates this packet writes its ID in sender_id field; to receive the *acknowledgement* from the HEAD and the notification is recorded in HEAD's table with sender ID.

(b) "*warning*": This packet is used to warn the nodes from a malicious node, it shown in Figure 3(b). When any HEAD makes sure that the suspected node is a malicious node, it writes the ID of a malicious node in malicious_id field. Then it writes number of attacks in number_of_attack field; to let the receiver know how many remainder fields. The attack_# field contains the type of attacks done by this malicious node. The first field is used to identify type of the packet.

(c) "*acknowledgement*": This packet is used to acknowledge *notification* or *warning* packets, it shown in Figure 3(c). When the HEAD receives the notification from any node, the HEAD sends this packet to the node. And when any node receives the *warning* from a HEAD, the node sends this packet to the HEAD. The malicious_id field is used when a node sends *notification* (or a HEAD sends *warning*) about two or more malicious knows, by this field, any *notification* (or *warning*) is arrived. The first field is used to identify type of the packet.

(a) HEAD_MALICIOUS_TABLE

sender_id	malicious_id	attack	dangerous

(b) MALICIOUS_TABLE

malicious_id	Sure	time

(c) PACKET_TABLE

packet_id	type
1	notification
2	warning
3	acknowledgment
4	new_request
5	new_info
6	new_complete

Figure 2. Scheme tables

(d) "*new_request*": This packet is used to request the HEAD of previous cluster of a node that want to join a new cluster. The HEAD of the host (new) cluster sends this packet to the HEAD of previous cluster; to request it to send all notifications about an arrival node, it shown in Figure 3(d). The ID of an arrival node is written in new_id field. The first field is used to identify type of the packet.

(e) "*new_info*": This packet is used to tell a HEAD about any notification of a specific node, it shown in Figure 3(e). The ID of this specific node is written in new_id field. When any HEAD receives *new_request*, it generates this packet. The number_of_record field contains number of notifications (rows); to let the receiver know how many remainder fields. The sender_id_# field corresponds to sender_id column in *HEAD_MALICIOUS_TABLE*. The attack_# field corresponds to attack column in *HEAD_MALICIOUS_TABLE*. The dangerous_# field corresponds to dangerous column in *HEAD_MALICIOUS_TABLE*. The first field is used to identify type of the packet.

(f) "*new_complete*": This packet is used to confirm arrival of *new_info* packet, it shown in Figure 3(f). When the HEAD receives the *new_info* from another HEAD, the receiver sends this packet to the sender. The new_id field is used when a HEAD sends *new_info* about two or more nodes, the *new_complete* packet receiver knows from this field any *new_info* of them is arrived. The first field is used to identify type of the packet.

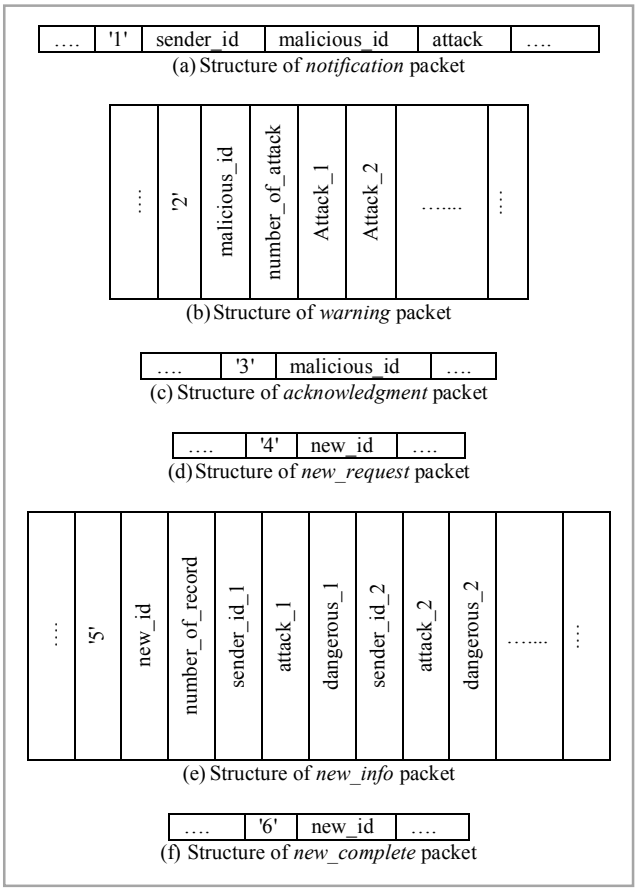


Figure 3. Scheme packets

B. Assumptions

Our scheme is designed for clustered wireless networks where every cluster has a HEAD. We assume that all HEADs are trusted. Our proposed scheme works as reaction of detection process, and the prevention and detection systems are exist in the network. Therefore, we assume that any attack makes node(s) needs a special case of recovery, this recovery should be known. All HEADs have these *recovery process(s)*. We assume every node in a network has an identifier (node_id). A node cannot change its identifier even if it leaves the network and joins it again.

C. The Algorithm

The proposed algorithm consists of ten parts, as illustrated in an earlier version of our work [25], below an updated version of it. Hence, the used thresholds depend on the cluster size and/or the security degree.

1- When any node (here: node A) decides a node (here: node X) is a malicious node
{
- Node A writes id of node X and the current time in its *MALICIOUS_TABLE* // sure will be '0' by default
- It creates a *notification* packet by writing its id, id of X, and type of attack
- It sends the notification packet to HEAD of the malicious node
- It waits for the *acknowledgment* from the HEAD ***t1*** unit of time
{
- If does not receive *acknowledgment*, it resends *notification* again (because the previous *notification* is lost)
}}

2- In *MALICIOUS_TABLE* of every node, after exceeds a threshold ***threshold1*** (i.e., predefined timeout expire) of any record (using time column)
{
- The record is deleted // because the decision to consider it a malicious node is wrong
}

3- When the HEAD receives any *notification,* it takes malicious_id from it
{
- If the id is an id of node not within its cluster, the *notification* will ignored
- Else, the HEAD checks its HEAD_MALICIOUS_TABLE
{
- If information of the *notification* exists, it resends *acknowledgment* again (because the previous *acknowledgment* is lost) and exits this part
- Else, the HEAD checks its MALICIOUS_TABLE
{
- If malicious exists, it resends *acknowledgment* again (because the previous *acknowledgment* is lost) and exits this part
- Else, if the sender is a head, go to part 5
{

- Else, the HEAD
{
- Writes the information of the *notification* in its HEAD_MALICIOUS_TABLE
- Call dangerous_assigner procedure for assigning the degree of dangerous to the new record
- Sends the *acknowledgment* to node which sends the *notification* (here: node A)
}}}}}

4- When dangerous_assigner procedure called, it takes (sender_id, malicious_id, and attack) from the new record
{
- It collects all records which have a same sender_id and attack in *HEAD_MALICIOUS_TABLE*
- If the collection (with the new record) contains all ordinary nodes in the network (using malicious_id column) // there is a probability that the sender is a malicious node
{
- It reduces the degree of dangerous of all these records by ***reduce_SA_low***
- It assigns ***low_degree*** of dangerous to the new record
}
- Else, if number of records in the collection exceeds a predefined threshold ***threshold2***
{
- It reduces the degree of dangerous of all these records by ***reduce_SA_moderate***
- It assigns ***moderate_degree*** of dangerous to the new record
}
- Else, it assigns ***high_degree*** of dangerous to the new record
- It collects all records which have a same sender_id and malicious_id in *HEAD_MALICIOUS_TABLE*
- If number of records in the collection exceeds a predefined threshold ***threshold3***
{
- It reduces the degree of dangerous of all these records by ***reduce_SM***
}}

5- Every HEAD after predefined threshold (or percentage) ***threshold4*** of dangerous degree of specific node (here: node X) in *HEAD_MALICIOUS_TABLE*
{
- It writes malicious_id (here: node X), writes '1' in the sure column and 'null' in time column in its *MALICIOUS_TABLE*
- It writes malicious_id field in *warning* packet
- It writes number of attacks and lists these attack(s) in warning packet
- It sends the *warning* to all nodes in its cluster (except the malicious) and all heads in the network
- It waits an *acknowledgment* from every node ***t2*** unit of time, and from every head ***t3*** unit of time // heads busier than others and might be far
{
- If does not receive *acknowledgment* from any node within its cluster or head, it resends *warning* again to this node(s) or head(s) (because the previous *warning* is lost)
- If receives *acknowledgment* from all nodes and heads, OK
}
- It organizes and prepares all reactions after the attack(s) (some of attacks need special reaction rather than isolation)
- It executes the organized reactions one by one
- It deletes all rows that contain malicious_id (here: node X) in malicious_id column from *HEAD_MALICIOUS_TABLE*
}

6- When a HEAD decides a node is a malicious node
{
- It writes id of the malicious node and the current time in its *MALICIOUS_TABLE* // sure will be '0' by default
- If the malicious node within its cluster
{
- It writes the information of the node in its *HEAD_MALICIOUS_TABLE*
- It assigns ***high_degree*** of dangerous to the new record
}
- Else, It creates a *notification* packet by writing its id, malicious_id, and type of attack
{
- It sends the *notification* packet to HEAD of the malicious node
- It waits for the *acknowledgment* from the HEAD ***t1*** unit of time
- If does not receive *acknowledgment*, it resends *notification* again (because the previous *notification* is lost)
}}

7- When any node (HEAD or not) receives a *warning,* it checks its *MALICIOUS_TABLE*
{
- If malicious_id of the *warning* exists
{
- If sure column contains '0'
{
- Changes it to '1' and writes 'null' in time column
- It organizes and prepares all reactions after the attacks
- It executes the organized reactions one by one
- Else, it resends *acknowledgment* again (because the previous *acknowledgment* is lost) and exits this part
}}
- Else
{
- It writes malicious_id in its *MALICIOUS_TABLE* and writes '1' in the sure column and 'null' in the time column
- It organizes and prepares all reactions after the attacks
- It executes the organized reactions one by one
}
- If it is a HEAD
{
- It forwards *warning* to all nodes within its cluster
- It waits for the *acknowledgment* from every node ***t2*** unit of time
{
- If does not receive *acknowledgment* from every node, it resends *warning* again to this node(s) (because the previous *warning* is lost)
}}
- It sends *acknowledgment* to the HEAD which sends the *warning*
}

8- When any node moves from one cluster to another, the HEAD of the host cluster checks its *MALICIOUS_TABLE*
{

- If the new node exists, exits this part
- It writes node_id of a new node in new_id field in *new_request* packet
- It gets the id of a previous HEAD of a new node from Routing Table
- It sends the *new_request* to the previous HEAD (to request the previous HEAD to send all *notification*s that in its *HEAD_MALICIOUS_TABLE* where malicious_id column is equal to new_id)
- It waits for the *new_info* from the previous HEAD ***t4*** unit of time
{
- If does not receive *new_info*, it resends *new_request* again (because the previous *new_request* is lost)
- Else, it checks number_of_record field, if it more than zero
{
- It checks its *HEAD_MALICIOUS_TABLE*
{
- If the information of the *new_info* or some of them exists, it ignores the old info. (because the old info. does not deleted)
- It records the new information in its *HEAD_MALICIOUS_TABLE*
}
- It writes new_id in *new_complete* packet and sends it to the previous HEAD (here we do not make sure if or not this packet arrives to the previous HEAD, because the importance of bandwidth is bigger than memory)
- It accepts the new node in the cluster
}}}

9- When the HEAD receives *new_request*, it checks its *HEAD_MALICIOUS_TABLE*
{
- If there is at least one row in the *table* which its malicious_id is equal to new_id
{
- It writes all rows that malicious_id column is equal to new_id from *HEAD_MALICIOUS_TABLE* in *new_info* packet
- It writes the number of these rows in *new_info* packet
}
- Else, it writes '0' in *new_info* packet
- It writes new_id in *new_info* packet
- It sends to the sender the *new_info*
}

10- When the HEAD receives *new_complete,* it checks its *HEAD_MALICIOUS_TABLE*
{
- If there is a row in the *table* which its malicious_id is equal to new_id
{
- It deletes all rows that contain new_id in malicious_id column from *HEAD_MALICIOUS_TABLE*
}}

IV. Experimental Results

To study the feasibility of our security protocol, we have implemented many scenarios in a network simulator and conducted a series of experiments to evaluate its effectiveness. We build a discrete event simulator using Java language by NetBeans 7.0.1 integrated development environment. We conduct our experiments by using proactive routing protocol which is Destination-Sequenced Distance Vector routing (DSDV) [26] with little modifications. In clustering protocol, we used Virtual Dynamic Backbone Protocol (VDBP) [27] with some changes. It flat topology structure, mobile, single-hop, and Location-based.

A. Parameters

Every simulation has its parameters that determine an experiment countenance. In our work, there are two parts of the parameters: parameters related to the environment and parameters related to the algorithm itself. So, we clarify them in two subsections separately, as below.

1) System Parameters

As known in the literature the simulation model consists of several important parameters such as simulation time, simulation area, mobility model, node speed, pause time, number of nodes, number of clusters, number of malicious nodes, and number of attacks that done by every malicious node, Each run was simulated in an area of 500m x 500m, and 1800 seconds of simulation time. Every node participated in the experiment is identical, mobile, and has 250m as a transmission range value. The transmission range value chosen carefully to reduce route-path; because our interest is information arriving, not how it arrive. Table 2 summarizes simulation parameter values.

Parameter	**Value**
Simulation time	1800s
Simulation area	500m x 500m
Transmission range	250m
Routing protocol	DSDV
Clustering protocol	VDBP
Mobility model	RWP
Node speed	7m/s
Pause time	50s
Number of nodes	50,120,200
Number of clusters	5
Number of malicious nodes	5%, 20%, 35% (of nodes)
Ranges of attacks/malicious node	(1-7), (10-15), (20-30)

Table 2. Summary of simulation parameters.

2) Algorithm Parameters

Our algorithm covers a wide range of the security degrees, and this was satisfied by number of parameters and thresholds included in it. We test several values of these parameters, and then we reach these values that give us a good algorithm behavior. As is evident in the algorithm (section III "part C"), there are fourteen parameters and Table 3 clarify their values.

B. Performance metrics

To evaluate the quality of our algorithm, we must use some measurements that show the performance of the scheme. In our work, we chose these metrics and use them in all experiments [13, 15,28]:

1) False Negatives (FN)

It represents the system detects a normal behavior, but _in fact_ it is an attack behavior. So, this situation is considered as

systematic missing. On the other hand, True Positives (TP) represents the system detects an attack behavior, and _in fact_ it is an attack behavior. Therefore, this situation considered as systematic correct. The equation of FN shown below:

(1)

2) False Positive (FP)

It represents the system detects an attack behavior, but _in fact_ it is not an attack behavior. So, this situation is considered as systematic error. On the other hand, True Negatives (TN) represents the system detects a normal behavior, and _in fact_ it is a normal behavior. Therefore, this situation is considered as systematic correct. The equation of FP shown below:

(2)

False Negatives Rate (FNR) is the percentage of FN in the system; False Positive Rate (FPR) is the percentage of FP in the system.

Parameter	Value
reduce_SA_low	10
reduce_SA_moderate	6
reduce_SM	15
low_degree	10
moderate_degree	15
high_degree	20
threshold1	900s
threshold2	20% (of ordinary nodes)
threshold3	3
threshold4	(*high_degree*)*5% (of nodes)
t1	60ms
t2	60ms
t3	80ms
t4	80ms

Table 3. Summary of algorithm parameters.

C. Scenarios

In this subsection, we will present thirty scenarios. There are three of them depend on security degree, and the remainder are used to show the performance of the proposed scheme under the chosen security degree. So, we clarify them in two separate subsections, as shown later.

In every scenario, we choose ~5% of nodes to form couples (saboteur and victim); the saboteur claims _falsity_ that the victim is a malicious and notifies about it multiple times; to let the network ignores and isolates it. The number of notifications that the saboteur sends it ranges from 5 to 20 notifications. Also, we choose 2% and 10% of nodes as colluding nodes which form a zombie; to the same reason that clarified above. Colluding nodes are those nodes who have agreed with each other to notify about a chosen victim. These actions clarify the behavior of the algorithm in prospective cases. All these nodes are ordinary _not malicious_ nodes; to see the behavior of the algorithm when it faces complex cases.

1) Security degree scenarios

As can be seen, the efficiency of our algorithm hinges on the security degree which depends on some parameter's and threshold's values. Thus values of these thresholds have to be set judiciously. Too high value of security degree would lead to the algorithm being too strict and hence increasing false positive rate. On the other hand, too low value of them would result in the algorithm ending up increasing false negative rate. Therefore we run different scenarios with a lot of combination's values to reach the best degree that can be named *moderate degree*. In all scenarios in this section, we fix the parameters that not related to security directly. We select three of scenarios _including the best one_; to clarify the effect of a security degree on the matrices, as illustrate below:

(a) High security degree: in this scenario, all values are assigned to give a very secure network. Parameters which responsible of reducing values of questionable notifications are given low values, while others which responsible of increasing a notification importance are given high values. Table 4 shows these values.

Parameter	Value
reduce_SA_low	4
reduce_SA_moderate	3
reduce_SM	5
low_degree	23
moderate_degree	26
high_degree	30
threshold1	1800s
threshold2	90% (of ordinary nodes)
threshold3	20
threshold4	(*high_degree*)*1% (of nodes)

Table 4. Parameters of high security degree's scenario.

(b) Moderate security degree: in this scenario, values are chosen to give a secure network. Balancing between false positive rate and false negative rate is taking into account. So, these parameters have been chosen to be fixed and general to all experiments (as shown previously in section IV "part 2 in A"). These values shown in Table 5.

Parameter	Value
reduce_SA_low	10
reduce_SA_moderate	6
reduce_SM	15
low_degree	10
moderate_degree	15
high_degree	20
threshold1	900s
threshold2	20% (of ordinary nodes)
threshold3	3
threshold4	(*high_degree*)*5% (of nodes)

Table 5. Parameters of moderate security degree's scenario.

(c) *Low security degree:* in this scenario, all values are assigned to give a low secure network. Parameters which responsible of reducing values of questionable notifications are given high values, while others which responsible of increasing a notification importance are given low values. Table 6 presents these values.

Parameter	Value
reduce_SA_low	25
reduce_SA_moderate	15
reduce_SM	30
low_degree	3
moderate_degree	7
high_degree	10
threshold1	100s
threshold2	5% (of ordinary nodes)
threshold3	1
threshold4	(*high_degree*)*50% (of nodes)

Table 6. Parameters of low security degree's scenario.

2) Algorithm scenarios

In our work, we had chosen some parameters to be variable; to study the system behavior in different cases. The scenarios consist of combination of these values. In every scenario, we change one parameter and fix the rests. The variable parameters are: number of nodes, number of malicious nodes, and range of attacks that done by each malicious node. We let the scenario's names clarify the parameter's values. Every name consists of three digits, first one is capital letter either *A, B,* or *C* which represents number of nodes *50,120,* or *200* respectively. The second digit is number *1, 2,* or *3* that represents number of malicious nodes which are *5%, 20%,* or *35%* respectively. Ranges of attacks that done by each malicious node are *(1-7), (10-15),* or *(20-30)* which is represented by third digit that is small letter *a, b,* or *c* respectively. Table 7 presents the twenty-seven scenarios with their names and parameters.

D. Results and Discussions

In this subsection, the simulation results are presented then a detailed discussion was done. Also, we proposed some improvements in the scheme. For each result, ten runs were averaged and reported with 95% confidence interval *(CI)*.

The results of the security degree scenarios appeared as we expected and predicated previously (section IV "part 1 in C"), the high security degree gives an effective false negative rate, but the false positive rate is very poor, as shown in Figure 4. On the other hand, low security degree lets the false negative rate extremely huge, even if the false positive rate is efficient.

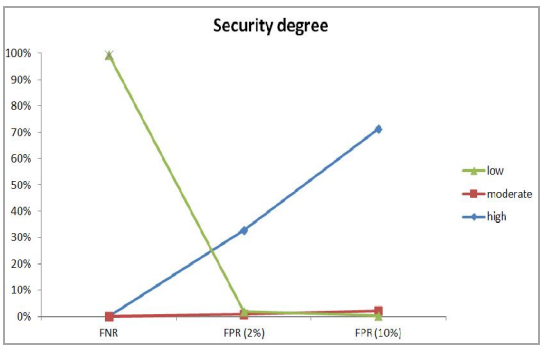


Figure 4. Security degree scenarios

We use parameters of *B-2-b* scenario (explained in Table 7) to run the security degree scenarios. The results illustrated in Table 8.

scenario's name	node's number	malicious nodes (of nodes)	attacks/malicious node
A-1-a	50	5%	(1-7)
A-1-b			(10-15)
A-1-c			(20-30)
A-2-a		20%	(1-7)
A-2-b			(10-15)
A-2-c			(20-30)
A-3-a		35%	(1-7)
A-3-b			(10-15)
A-3-c			(20-30)
B-1-a	120	5%	(1-7)
B-1-b			(10-15)
B-1-c			(20-30)
B-2-a		20%	(1-7)
B-2-b			(10-15)
B-2-c			(20-30)
B-3-a		35%	(1-7)
B-3-b			(10-15)
B-3-c			(20-30)
C-1-a	200	5%	(1-7)
C-1-b			(10-15)
C-1-c			(20-30)
C-2-a		20%	(1-7)
C-2-b			(10-15)
C-2-c			(20-30)
C-3-a		35%	(1-7)
C-3-b			(10-15)
C-3-c			(20-30)

Table 7. Scenario's names and parameters.

The results of the remainder twenty-seven scenarios reveal the importance and effect of the three variable parameters. They illustrated in Table 9. First of all, as is clear, all scenarios are not affected by colluding nodes when they were just 2% of nodes, but after they become 10% of nodes, their affect appears. So, when we discuss the false positive rate, we mean the values under 10% colluding nodes and we will ignore 2%.

security degree	FNR	FPR	
		colluding node	
		2%	10%
high	0%	32.8%	71.2%

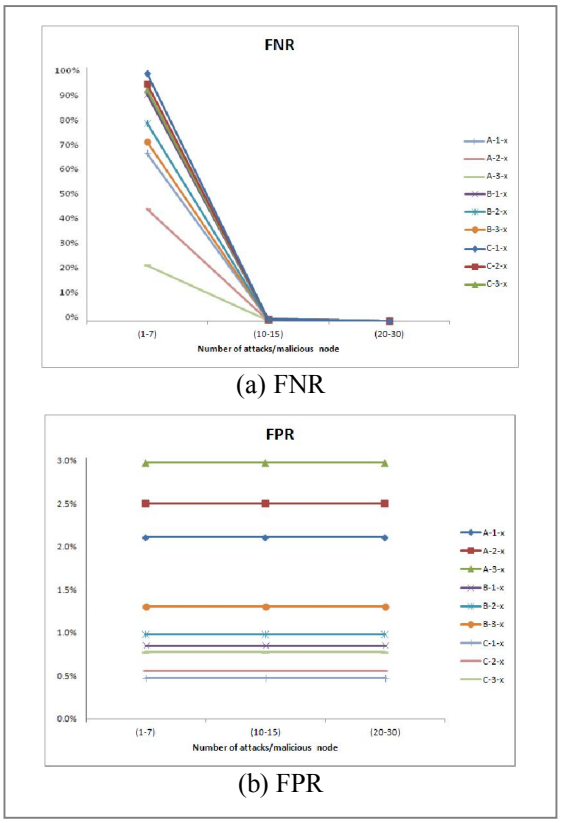
moderate	0%	0%	1%
low	99.3%	0%	0%

Table 8. Security degrees scenario's results.

Let us start with "range of attacks that done by each malicious node" parameter which is the most influential parameter, the results is concluded in Figure 5(a) to false negative rate and Figure 5(b) to false positive rate. Figure 5(a) clarifies that when the value of this parameter is small, the malicious node will be not isolated. So, every scenario has "*a*" in its name, gives a high and remarkable values of the false negative rate, and these values increases dramatically according to the increasing in the network size. While the false positive rate was not affected by this parameter at all, it is stabile in all ranges as it clear in Figure 5(b).

scenario's name	FNR	FPR	
		colluding node	
		2%	10%
A-1-a	18.2%	0%	2.1%
A-1-b	0%	0%	2.1%
A-1-c	0%	0%	2.1%
A-2-a	10.5%	0%	2.5%
A-2-b	0%	0%	2.5%
A-2-c	0%	0%	2.5%
A-3-a	7.8%	0%	3%
A-3-b	0%	0%	3%
A-3-c	0%	0%	3%
B-1-a	51.1%	0%	0.9%
B-1-b	0.2%	0%	0.9%
B-1-c	0%	0%	0.9%
B-2-a	33.2%	0%	1%
B-2-b	0%	0%	1%
B-2-c	0%	0%	1%
B-3-a	20.1%	0%	1.3%
B-3-b	0.1%	0%	1.3%
B-3-c	0%	0%	1.3%
C-1-a	98.6%	0%	0.5%
C-1-b	1.6%	0%	0.5%
C-1-c	0%	0%	0.5%
C-2-a	98.7%	0%	0.6%
C-2-b	0.8%	0%	0.6%
C-2-c	0%	0%	0.6%
C-3-a	98.3%	0%	0.8%
C-3-b	0.5%	0%	0.8%
C-3-c	0%	0%	0.8%

Table 9. Scenario's results.

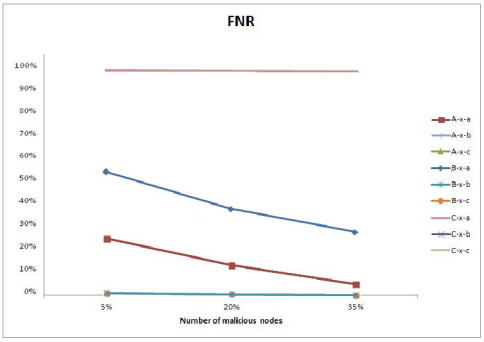


(a) FNR

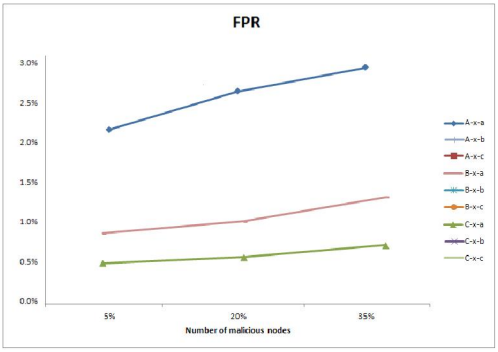
(b) FPR

Figure 5. Metrics with range of attacks/malicious node parameter.

Now, we will focus on "number of malicious nodes" parameter and its results that shown in Figure 6(a) to false negative rate and Figure 6(b) to false positive rate. By a deep slant on these figures, we find this parameter affect slightly on the results. While false positive rate increases tardily when this parameter increased, the false negative rate is decreases.



(a) FNR



(b) FPR

Figure 6. Metrics with number of malicious nodes parameter.

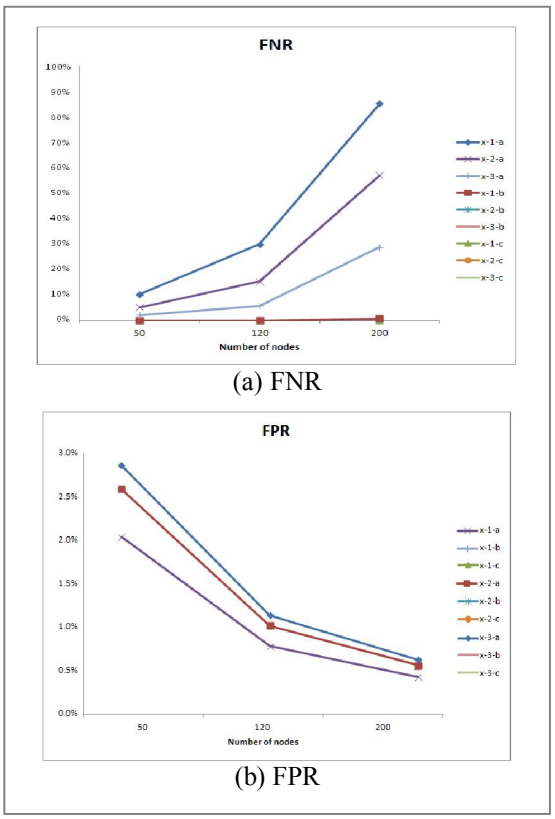


Figure 7. Metrics with network size parameter.

The last measured parameter is network size (number of nodes) that its results shown in Figure 7(a) to false negative rate and Figure 7(b) to false positive rate. These figures clarify that when the network size increases, there is a little decreasing in false positive rate, and a dramatic increasing in false negative rate.

On the subject of number of the control packets that sent in the network, we count them in each scenario and calculate the average. It was nearly 50 packets/ node during the simulation time which is 30 minutes.

By a wide look on the previous table and figures, we find the algorithm work well in different scenarios except those that have a in their name. So, let us discuss this point. When number of attacks that are done by one malicious node was less than that considered by ***threshold4***, the alarm does not happen. On the other hand, when we try to consider this case, the other cases affected negatively. So, we propose a solution to this point and we will work in it in the future, which is adding a new agent to our scheme as "*monitoring*," that monitors each suspected node after predefined number of notifications about it. Theoretically, this solution will reduce the false negative rate dramatically. Also, false positive rate will be affected positively, especially in colluding nodes existence. We suppose _after this addition_ the algorithm will work well in all cases.

V. Conclusion

In this work, we described the importance of security in MANETs. We explained the processes of intrusion detection system which is one of technologies within security space. Also, we presented some of intrusion detection systems that exist in the literature. We clarified their limitations and drawbacks.

After that, we proposed a new intrusion detection scheme for clustered mobile ad hoc networks which overcomes the drawbacks and eliminates the limitations. Our scheme warns all nodes in the network from a malicious node provided to be sure that the node is really malicious. The network nodes cooperate to get rid of the attack effects. We simulated the scheme and get great results. We reduced the false positive rate remarkably, which becomes close to zero. The false negative rate is reduced unless in one case and we are working on it right now.

In the future we will implement the "*monitoring*" agent idea (for details see section IV "part D"), and we will balance between reducing a false negative rate and increasing the overhead of node(s) that run this agent. Also, we will concatenate several alarms with each other when they generated in the same time. On the same way, we will concatenate and delayed acknowledgement to reduce the number of packets sent.

References

[1] A. Goel and A. Sharma, "Performance Analysis of Mobile Ad-hoc Network Using AODV Protocol," *International Journal of Computer Science and Security*, 3 (5), pp. 334-343, 2009.

[2] Y. Hu, A. Perrig and D. B. Johnson, "Ariadne: a secure on-demand routing protocol for ad hoc networks," *Wireless Networks Journal*, 11 (1-2), pp. 21-38, 2005.

[3] C. Rajabhushanam and A. Kathirvel, "Survey of Wireless MANET Application in Battlefield Operations," *International Journal of Advanced Computer Science and Applications,* 2 (1), pp. 50-58, 2011.

[4] Y. Fu, J. He, L. Luan, G. Li, and W. Rong, "A Key Management Scheme Combined with Intrusion Detection for Mobile Ad Hoc Networks". In *Proc. KES-AMSTA'08*, pp.584-593, 2008.

[5] M. Su and K. Chiang, "Prevention of Wormhole Attacks in mobile ad hoc networks by Intrusion Detection Nodes". In *Proc. WASA'10*, pp. 253-260, 2010.

[6] F. H. Wai, Y. N. Aye, and N. H. James, "Intrusion Detection in Wireless Ad-Hoc Networks". In *Proc. MobiCom'00*, pp. 275-283, 2000.

[7] W. Wang, B. Bhargava, and L. Mark, "Defending against Collaborative Packet Drop Attacks on MANETs". In *Proc. DNCMS'09,* 2009.

[8] Y. Zhang and W. Lee, "Intrusion Detection in Wireless Ad-Hoc Networks". In *Proc. MobiCom'00*, 2000.

[9] F. Nait-Abdesselam, "Detecting and avoiding wormhole attacks in wireless ad hoc networks," *IEEE Commun. Mag.*, 46 (4), pp. 127-133, 2008.

[10] D. B. Roy and R. Chaki, "Baids: Detection of Blackhole Attack in Manet by Specialized Mobile Agent", *International Journal of Computer Applications*, 40 (13), pp. 1-6, 2012.

[11] Shi-rui, Liu, and Z. Li, "The intrusion detection model based on the fuzzy judgment in ad hoc network." In *Proc. WiCom'09*, pp. 1-4, 2009.

[12] S. Hirnwal, K. Chauhan, and A. Gupta, "Intrusion Detection Technique in Mobile Adhoc Network Based on Quantitative Approach", International Journal of Computer Applications, 37 (8), pp. 22-27, 2012.

[13] Y. Yao, L. Zhe, and L. Jun, "Research on the Security Scheme of Clustering in Mobile Ad Hoc Networks". In *Proc. ITCS'09*, pp. 518-521, 2009.

[14] J. Parker, J. Undercoffer, J. Pinkston, and A. Joshi, "On intrusion detection and response for mobile ad-hoc networks". In *Proc. IPCCC'04*, 2004.

[15] M. Shao, J. Lin, and Y. Lee, "Cluster-based Cooperative Back Propagation Network Approach for Intrusion Detection in MANET". In *Proc. CIT'10*, 2010.

[16] R. Hecht-Nielsen, "Theory of the back-propagation neural network". In *Proc. IJCNN*, pp. 593-605, 1989.

[17] N. Marching and R. Datta, "Collaborative Technique for Intrusion Detection in Mobile Ad hoc Network," *Ad hoc Networks*, 6 (4), pp. 508-523, 2008.

[18] S. Bhargava and D. P. Agrawal., "Security Enhancements in AODV protocol for Wireless Ad Hoc Networks". In *Proc. VTC'01*, vol. 4, pp. 2143-2147, 2001.

[19] L. Bononi and C. Tacconi, "A Wireless Intrusion Detection System for Secure Clustering and Routing in Ad Hoc Networks". In *Proc. ISC'06*, vol. 4176, pp. 398-414, 2006.

[20] JML. Manickam and S. Shanmugavel, "Providing Routing Security Using ROS Protocol in MANET and Performance Comparison with AODV," *Inform. Technol. J.,* 6 (5), pp. 656-663, 2007.

[21] S. Madhavi and T. H. Kim, "An Intrusion Detection System in Mobile Ad hoc Networks," *International Journal of Security and its Application*, 2 (3), pp. 1-16, 2008.

[22] A. Al-Roubaiey, T. Sheltami, A. Mahmoud, E. Shakshuki, and H. Mouftah, "AACK: Adaptive Acknowledgment Intrusion Detection for MANET with Node Detection Enhancement". In *Proc. AINA'10*, pp. 634-640*,* 2010.

[23] S. Ganapathy, P. Yogesh, and A. Kannan, "An Intelligent Intrusion Detection System for Mobile Ad-Hoc Networks Using Classification Techniques". In *Proc. PEIE'11*, 148 (3), pp. 117-122, 2011.

[24] S. Bu, F. R. Yu, P. X. Liu, H. Tang, and P. Mason, "Distributed Combined Authentication and Intrusion Detection with Data Fusion in High Security Mobile Ad-hoc Networks," *Vehicular Technology Journal*, 60 (3), pp. 1025-1036, 2011.

[25] Hajar Al-Hujailan, Mznah Al-Rodhaan and Abdullah Al-Dhelaan "A Cooperative Intrusion Detection Scheme for Clustered Mobile Ad Hoc Networks". In *Proc. IAS'11*, pp. 179-185, 2011.

[26] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," *ACM SIGCOMM Computer Communication*, 24 (4), pp. 234-244 , 1994.

[27] U. C. Kozat, G. Kondylis, B. Ryu, and M. K. Marina, "Virtual dynamic backbone for mobile ad hoc networks". In *Proc. ICC'01*, 2001.

[28] H.–Y. Chang, S.F. Wu and Y.F. Jou, "Real-Time Protocol Analysis for Detecting Link-State Routing Protocol Attacks," *ACM Tran. Inf. Sys.Sec.*, 1, pp. 1-36, 2001.

Author Biographies

Hajar Al-Hujailan received the BS degree in information technology from King Saud University, Riyadh, Saudi Arabia, in 2009, and will receive MS degrees in computer science from King Saud University, Riyadh, Saudi Arabia, in the spring of 2012. Since 2011, Hajar Al-Hujailan has been a faculty member in the Department of Computer Science, College of Computer and Information Sciences at King Saud University, Riyadh, Saudi Arabia. Hajar Al-Hujailan's research interests include information and networks security, mobile ad hoc networks, sensors networks, interconnection networks, high performance computing and parallel processing.

Mznah Al-Rodhaan received her BS in Computer Applications (Hon) and MS in Computer Science both from King Saud University, in 1999 and 2003 respectively. In 2009, she received her Ph.D. in Computer Science from the University of Glasgow, Scotland, UK. She is currently working as an assistant professor and vice chair of the Computer Science department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia. Moreover, she served in the editorial board of the *Ad Hoc Journal* (Elsevier) and has participated in several international

conferences. Her current research interest includes: Mobile Ad Hoc Networks, Wireless Sensor Networks, Cognitive Networks, Network Security, and High Performance Computing.

Abdullah Al-Dhelaan received his BS in Statistics (Hon) from King Saud University, in 1982, and his MS and Ph.D. in Computer Science from Oregon State University in 1986 and 1989 respectively. He is currently a Professor of Computer Science, Vice Dean for Graduate Studies and Research, Chairman of the join Ph.D. program, and Director General for the Center for International Collaboration and Visiting Professors, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia. He has guest edited several special issues for the *Telecommunication Journal* (Springer), and the *International Journal for Computers and Their Applications* (ISCA). Moreover, he is currently on the editorial boards of several journals such as *Computer Network* (Elsevier) and the *International Journal of Computers and Their Applications*. His current research interest includes: Mobile Ad Hoc Networks, Sensor Networks, Cognitive Networks, Network Security, and High Performance Computing.