2012 26th International Conference on Advanced Information Networking and Applications Workshops

A Distributed IDS for Ad Hoc Networks

Paulo M. Mafra, Joni da Silva Fraga
Departamento de Automacao e Sistemas (DAS)
Universidade Federal de Santa Catarina (UFSC)
Florianopolis – SC – Brazil
Email: mafra,fraga@das.ufsc.br

Altair Olivo Santin
Pontificia Universidade Catolica do Parana (PUC-PR)
Curitiba – PR – Brazil
Email: santin@ppgia.pucpr.br

***Abstract*—The intrusion detection systems (IDS) are usually designed to work on local networks. However, with the development of mobile networks and their applications, it became necessary to develop new architectures for IDSs to act on these networks in order to detect problems and ensure the correct operation of data communications and its applications. This paper presents a distributed IDS model for mobile ad hoc networks that can identify and punish those network nodes that have malicious behavior. In this paper we describe the proposed model, making a comparison with major efforts in the literature on distributed intrusion detection systems for mobile ad hoc networks.**

***Keywords*-Mobile Ad Hoc Networks; Intrusion Detection System; Distributed Systems; Internet Security;**

I. Introduction

The past decade has witnessed great evolution in communication technologies. This evolution has permitted that communications have become both faster and cheaper. Among such new technologies are Mobile Ad Hoc Networks (MANETs) [1], which form highly dynamic environments without the presences of concentrated units. However, such technology is also vulnerable to diverse types of attacks from with faulty or malicious nodes which evolve as well, given the mobility of such environments. The key question in such systems is the guarantee that applications in them may always evolve, even in the face of the failures, attacks from malicious entities, or the mobility of the entities.

In evidence of the difficulty of avoiding malicious actions, there is the need for mechanisms which at least deal with minimizing the effects of such activities. Among these mechanisms, we can include intrusion detection systems (IDS). The use of monitoring communications in MANETs among the activities which make up part of this network and punishing entities which present faulty or malicious behavior (their exclusion from the network, for example) improves the security conditions in these environments. This paper proposes a secure IDS model which can be distributed in mobile ad hoc networks which applies the concepts of distributed and dependable systems. The use of these concepts permits - within certain limits - a model which is less subject to the restrictions of other proposed in corresponding literature. In section II, we describe the organization of these entities, as well as the roles that define the algorithmic base which supports our secure IDS. Next, section III presents key research present in literature, comparing them with our propositions. We finalize this paper with some conclusions.

II. Architecture Description

In MANETs, the communication protocols that involve them, such as message routing, depend upon collaboration from equipment which form these networks' nodes. As such, we also assume this collaborative environment with all the participating intrusion detection nodes in our IDS model. The proposed IDS model is distributed and should assume a hierarchical stratification in order to attend the diverse IDS functions. The differentiation of functions and their distribution among nodes in this collaborative environment establish two classes of nodes: the "leader nodes", which perform higher level functions such as analysis; and the "collector nodes", which assume lower level functionalities such as sensors, collecting data for future analysis.

It is also important to highlight that any node, contrary to the majority of related studies, may assume any IDS function, performing both the collecting or leading role in the network during different periods within the system. We also assume that all the network nodes possess at least $2f+1$ neighbors, where f is the failure or intrusion limit that our algorithms should support[1].

A. Topology Description

The hierarchical topology of the model introduces the idea of clusters, as well as in systems proposed by [2], [3], [4]. However, in our model, we consider each cluster to have various "leaders". This various leaders form what we denominate as "leadership". The leading nodes which form a leadership in a cluster use "secure channels" to exchange information and alerts among themselves. The collecting nodes send a summary of the data collected to the leaders

[1]This limit f will be used in our approach of a threshold for the occurrance of anormalities which, once they are not surpassed, services and protocols in the system will continue to supply the correct and expected behavior. At this limit, malicious and faulty nodes present in the cluster and further, the outputs or churn during a predefined period are accounted for at a given instant named *epoch_time*

978-0-7695-4652-0/12 $26.00 © 2012 IEEE
DOI 10.1109/WAINA.2012.55

also through "secure channels". The leadership which define the domains of a cluster are instituted based on $2f+1$ leaders. Leadership and, consequently, the corresponding cluster no longer exist as of the moment in which a leadership has less than $2f+1$ leaders.

B. Component Description

The collecting nodes constitute the largest part of the cluster. They are principally formed by the newest nodes in the network, those with energy restrictions or few neighbors, etc. Beyond data collection, these sensory nodes apply some filters and periodically send data summaries to the leaders for analysis. In order to belong to a cluster, the collecting nodes need to possess secure channels with at least f +1 leaders of the cluster. It is also assumed that communication among the collecting nodes and messages which do not belong to the IDS occur in an insecure manner. The IDS messages are always encrypted and authenticated in secure channels (the "collector - leader" and "leader - leader" channels). The cryptographic mechanisms employed are elaborated in section II-C. The data collection by the collectors always considers its neighborhood. In other words, the collecting nodes capture the data from each neighbor and store the information in them in a table. This information may be, for example, the quantity of packages received and sent by the neighboring node. Each column of this table represents a neighbor and the rows describe the information obtained concerning the traffic monitored from these neighbors. These collections and reports to leadership are made with a defined frequency, during precise periods.

The leader nodes analyze the data sent by the collecting nodes related to the same cluster, or even from those obtained through own analysis. Based on the analysis of this monitoring information, they make their decisions concerning malicious nodes. These decisions are in turn registered in local lists (ex. malicious nodes list) and later shared, compared, and synchronized with the remaining leader nodes which make up the same leadership which determines the existence of the corresponding cluster. The values collected by the sensors (collecting nodes) are sent to each leader during established periods of time, defined in detail in section II-D. At the end of each of these times, each leader initiates its analysis. Such analysis takes into account the available data based on node's neighbors and is realized based on the system denominated Octopus-IIDS [5]. If analysis from data supplied by a collector indicates that a neighbor i is suspect, the leader will analyze the data supplied by all the collectors which neighbor the suspected node i. If the data analysis from $f+1$ nodes of its neighborhood point out node i as suspect, then the corresponding leader considers it malicious. The Algorithm 1 describes this procedure.

Analysis concerning malicious activities (Algorithm 1) is done by all the leader nodes present in a leadership.

Algorithm 1 Data_Analysis(Collected_data,i)

```
1: suspect ← 0
2: ∀ j ∈ Neighbor(i)                                  % for all j neighbor of i
3:    if OctopusAnalysis(i, Collected_data) then      % i is suspect by j
4:       suspect ← suspect + 1
5:    end if
6: if suspect > f + 1 then                            % the majority considers i malicious
7:    return ← malicious
8: end if
```

The comparison (through the exchange of encrypted and authenticated messages) among the results obtained in analysis from these leaders is made concrete as well in the leadership. The results obtained by these comparisons will be considered by the cluster as a whole. With $f+1$ leaders agreeing upon the analysis results, the leadership decision is always assumed as the results of these leaders' analysis results. If there are at least $2f+1$ leading nodes, the cluster will always have to decide upon any analysis result, even in the presence of f malicious leaders.

Each leader will only be connected to a cluster if it possesses routes to at least $f+1$ leaders for that cluster, just as any node in the network. In order to the messages reach leadership, there must always be dissemination in leadership based on the correct leader. The algorithm 2, based on [6], describes the steps of the dissemination protocol. In this algorithm, a node j disseminates a message msg_j to the $Leadership_j$ to what corresponds to its knowledge about the leaders which form the cluster leadership. Upon receiving the message msg_j, each leader in turn sends it to its respective leadership knowledge. In this algorithm, at least one correct leader is reached with each resend, which returns to disseminate the message once again, using the recursion of the protocol $Disseminate()$. As each leader is connected to at least one correct leader and if the cluster leadership does not form disjointed graphs, then all the leaders will be reached through such dissemination.

Algorithm 2 Disseminate Protocol

```
1: {On Initialization}
2:    Received_k ← {};

3: {On Disseminate(msg_j, Leadership_j) at node j}
4:    for all l_k ∈ Leadership_j do
   %    msg_j is send to leaders l_k known by node j
5:       send < DISSEMINATION, msg_j > to l_k
6:    end for

7: {On Receive(< DISSEMINATION, msg_j >) at l_k}
8:    if (< DISSEMINATION, msg_j > ∉ Received_k) then
9:       Received_k ← Received_k ∪ {< DISSEMINATION, msg_j >
   };
10:      Disseminate(msg_j, Leadership_k)
11:      deliverDissemination(msg_j);   % msg_j locally delivered in l_k
12:   end if
```

Once the leadership has been agreed concerning a malicious node (decision by $f+1$ leaders), a global alert is generated and sent by leadership to the cluster itself and to neighboring cluster leaderships, if they are present. The

clusters are organized still on another higher level of the network which exchanges information among its leadership. Communication among leadership utilizes the communication infrastructure of the collecting and leading nodes, sending authenticated and encrypted messages through these nodes. Upon receiving the global alert message, leaders must disseminate such messages, first in their clusters and later to other leadership which neighbors the receptor cluster. It will be necessary to authenticate the leadership in various leadership actions within the proposed model.

C. Cryptographic Mechanisms

IDS communication, which occurs among (collector - leader) and (leader - leader) cluster nodes makes use of cryptography. Encrypting messages is done with the use of session keys and symmetric encryption. Key distribution involves the public keys (pair of asymmetric keys) for each participating node. These asymmetric cryptography keys are one of the prerequisites for any participating node in the system and in the IDS. Leadership also authenticates its messages. Such leadership authentications are founded in the threshold signature scheme (TSS) [7].

The public leadership key (E_l) will always been known by all the cluster nodes and is used in verifying leadership signatures. The corresponding private key (D_l), used to generate the leadership signatures, possesses guaranteed sanctity through threshold cryptography. In other words, the key D_l is not available in any moment in the system. Its use is through a K set of partial keys ($|K| = m, K = SK_1, SK_2, ..., SK_m$) derived from D_l using threshold cryptography. The threshold signature scheme used is based on the RSA [8] algorithm, i.e., the combination of the partial signatures generates an RSA signature. In this model, generating and verifying partial signatures is completely non-interactive, without needing messages exchanges to execute these operations. Beyond this, the size of a partial signature is limited by the size of the RSA module.

Table I
KEYS IN THE DISTRIBUTED IDS MODEL

Cluster Entities	**Session Keys (k_s)**	**Asymmetric Key Pair (E, D)**	**Threshold Scheme**
Collector Node	Yes, for secure channels	Yes, for authentication and establishment of secure channels	No
Leader Node	Yes, for secure channels	Yes, for authentication and establishment of secure channels	Participate with partial key SK_i
Leadership	No	E_l and D_l for leadership's authentication	Signature with D_l is only reached with the combination of at least t partial valid signatures $(a_1; ...; a_k)$

In the proposed distributed IDS, each one of these m partial keys is delivered to a specific leader from leadership. Any operation with D_l is only possible through the participation of at least t leaders and their partial keys. These keys are generated during the activation of a cluster, executing an algorithm for distributed key generation [9], [10]. At the end of executing the algorithm, each leadership member will possess the public key E_l of leadership and its derived partial key D_l. The public leadership key, used to verify the signatures generated, is available in any cluster node. In order to guarantee the sanctity of the leadership authentication scheme, it is necessary that the f limit is not greater than the t threshold from the threshold scheme ($f < t < m$). On the other hand, the number t must not be very large in order not to make the leadership authentication protocol difficult. We thus assume $t = f + 1$, which guarantees that the malicious nodes would need a correct node for a signature, which is not possible. Table I summarizes the use of crytographic keys in a cluster.

D. Synchronizing periods, epoch times, and round updates

In order to deal with the dynamic aspects of the network and collect data for the detection process, it was necessary to define the times which determine the synchronization of the actions distributed throughout the system. As we work essentially with time periods, synchronizing the clocks is not necessary for the nodes to initiate synchronized operations. Using their local clocks, the periods are controlled with their respective deadlines with timers which aid corresponding operational activation. In order to initiate a common activity synchronized among pairs, the routine presented in Algorithm 3 is used. It depends upon the course of the stipulated time and reception of at least $f + 1$ corresponding sync messages. Two periods were defined for synchronization: transmission time and epoch. The epoch corresponds to the periods in which each cluster "freezes" its composition. The changes which occur in the system during this period are not updated, i.e. possible system changes (such as faults, node entrances or exits, etc.) are not taken into consideration in composing the cluster during that time. At the conclusion of each epoch, the cluster must synchronize itself. Thus, the updating round (UR) is initiated. These updating rounds, also present in [11], [12], define a time period where cluster leaders exchange information in order to update their knowledge concerning the present state of the cluster. During an updating round, new roles are also defined within the cluster and global alerts are sent. These decisions will always depend upon the agreement of $f + 1$ leaders.

In Section II-B, we described that the clusters send data monitoring summaries to leadership. Such data is sent upon concluding a time period called the transmission time (Tt). The leaders analyze the data sent from collectors at the end of each Tt. Comparison of such analysis is made at the end of each epoch. Transmission time occurs n times in each epoch ($Tt = epoch/n$). The network administrator, upon configuration, determines the $epoch$ and n values. The ratio of these periods of time is illustrated in Figure 1.

Algorithm 3 Synchronization()

1: **Require** : $receive < sync_j >$ *or period* d *is elapsed in* N_i
2: **Init** :
3: $\delta \leftarrow \mathbf{time}()$ % starts a new period counting d

4: **upon** $((\mathbf{time}() - \delta) \geq d)$ **do** % at the end of d
5: $send < sync_j > to\ Leadership_i$ % send of messages *sync* by i

6: **upon** $receive(sync_j)$ **do**
7: $Sync_i^d \leftarrow Sync_i^d \cup \{sync_j\}$ % sync messages received by i
8: **if**$(\mid Sync_i^d \mid \geq f+1)$**then** % number of *syncs* received by i overcomes the threshold f
9: $send < sync_j > to\ Leadership_i$ % call other leaders to start synchronization
10: $UR()$ % a new syncronized UR is started
11: **end if**

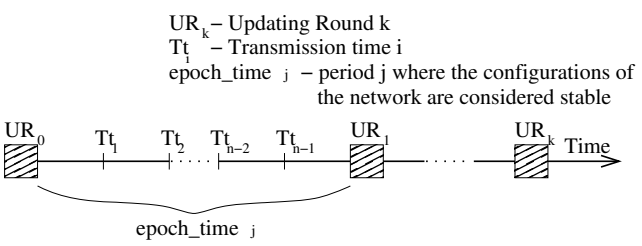

Figure 1. Temporal relation among updating rounds

In the case of including or removing leaders from a cluster, it is necessary to revoke and generate new partial keys for leadership. In order to generate keys, in the threshold scheme used, we employ the protocols proposed in [13]. In the event the leadership is left with less than $2f+1$ leaders, the leadership will be undone, no longer existing. In this case, the remaining leaders and all the collecting nodes of such a cluster will no longer participate in the IDS until they are associated with other leadership, during an updating round. At the end of the UR, a new epoch is begun (new period epoch_time). We have developed algoritms for both Tt and UR but because of lack of space we will not show them here.

E. Identification, entrance, and communication among nodes

The node identification process in the network should be secure enough so that it cannot create multiple identities. Thus, it is possible to prevent attacks like the sybil [14]. In our model, node identification is carried out with the use of certificates. A certifying authority (CA) considered to be known and reliable by the network nodes generates a certificate for the public key for each node when it joins the system. The role of the CA in our model will be managed by the network administrator in this initial step[2]. With this model, we may define the user and his/her equipment. There will not be users using multiple equipment on the network.

[2]This certifying entity will not necessarily need to be an official PKI. It may be a system management commission, an administrador, etc.

A certificate will be attributed to each node that participates in the network. The entrance process for a node involves exchanging diverse messages. Table II presents a summary of the messages exchanged in this process.

Table II
EXCHANGED MESSAGES IN THE ENTRANCE OF A NODE

Message	Sender	Destination	Content
REQCOMP	Node$_i$	Leadership	(node_id, public_key, energy, id_neighbors, num_steps)
REPCOMP	Leadership	Node$_i$	(*sign*(*encrypt*(public_key, num_leaders_leadership, role, UR, NETWORKCONF)))
DISSCOMP	Leadership	Leaders and Collectors of Leadership	(*sign*(node_id, public_key, UR, role))

Algorithm 4 presents the steps involved to insert a new node within a network cluster. Upon entrance, a new node i must send an entrance request message (REQCOMP), starting with its neighbors. This message, which should reach the cluster leadership, informs its identification (node_id), its credentials (for example, its public key in certificate form), energy, etc., (lines 7-9 of the algorithm 4). If a neighboring node is already part of the IDS and is not a leader, that node must possess a path to some leadership and will send this received REQCOMP message directly to its leadership (line 14). Upon receiving the REQCOMP message, a leader disseminates the message to the remaining cluster leaders (line 17). Leadership then defines the role of the new node and generates a REPCOMP message, signed with the leadership response and sent to the new node i (lines 18-23). Following, a DISSCOMP message containing the identification of the new node i and its role in the cluster is sent to the remaining cluster nodes (lines 24-26). The new node i, upon receiving the REPCOMP message, verifies its signature and updates its view of the cluster, assuming the role attributed to it by leadership (lines 30-41).

In lines 19, 20, and 21 of the algorithm 4, we consider that the threshold cryptography algorithms and the decision of roles are centralized in a coordinator. Detecting whether this coordinator were malicious will be easy through the redundancy of information and public keys involved. It will receive partial information signed from leadership participation. The need to disseminate responses in leadership limits malicious actions. The fact of having more than one coordinator does not provoke serious problems, as the results are always the same.

It is important to express that the same REQCOMP message should arrive to leaders various times. This is due to the initial flood activated by the node i and also the use of the $Disseminate()$ protocol by the first leader, in order to instigate that it reaches all of leadership's nodes.

Algorithm 4 Entrance of a node i into the cluster

1: **Var**
2: $Collectors\_List \leftarrow$ {Collectors in last UR}; % list agreed by f+1 leaders in last UR
3: $Leaders\_List \leftarrow$ {Leaders in last UR}; % list agreed by f+1 leaders in last UR
4: $Malicious\_List \leftarrow$ {Malicious in last UR}; % list agreed by f+1 leaders in UR
5: $leadership_i \leftarrow \{\ \}$ % view of i about the list of leaders

6: {node$_i$}
7: **for all** $j \in Neighbor(i)$ **do** % for all j neighbor of i
8: Send$< REQCOMP, id_i, cred_i >$ **to** j % send the REQCOMP message to j
9: **end for**

10: {node$_j$}
11: **upon receive** $< REQCOMP, id_i, cred_i >$ **do**
12: $msgValid \leftarrow VerifiedSignature(cred_i, admin)$ % node j verify the sender credentials i
13: **if** $(j \in Collectors\_List) \wedge (i \notin Malicious\_List)$ **then** % if j is Collector and i is not malicious
14: Disseminate($< REQCOMP, id_i, cred_i >, leadership_j$) % j disseminate REQCOMP
15: **end if**
16: **if** $(j \in Leaders\_List) \wedge (i \notin Malicious\_List)$ **then** % if j is leader and i is not malicious
17: Disseminate($< REQCOMP, id_i, cred_i >, leadership_j$) % j disseminate REQCOMP
18: **if** $(\forall\ id_k : l_k \in Leaders\_List \wedge id_j < id_k)$ **then**
19: $coordinator \leftarrow id_j$ % leadership signature: l_j is the Coordinator
20: $role_i \leftarrow$ defineRole$< REQCOMP, i >$ % definition of the role of the node i by the leadership
21: $sign \leftarrow$ signature($< REPCOMP, role_i >, leadership_j$) % signature of the response
22: Disseminate($< REPCOMP, role_i, sign, E_l, j >, leadership_j$)
23: Send$< REPCOMP, role_i, sign, E_l, j >$ **to** i% send REPCOMP to the new node i
24: **for all** $q \in neighbors(j)$ **do**
25: Send$< DISSCOMP, j >$ **to** q % send DISSCOMP to all q nodes of the cluster
26: **end for**
27: **end if**
28: **end if**

29: $\{node_i\}$
30: **upon receive** $< REPCOMP, role_i, sign, E_l, j >$ **do**
31: $msgValid \leftarrow$ VerifiedSignature($< REPCOMP, role_i >, leadership_j$) % valid signature
32: **if** msgValid **then**
33: $leadership_i \leftarrow leadership_i + \{l_j\}$; % update of the leaders view
34: **end if**
35: **if** $\mid msg_i \mid \geq f + 1$ **then** % if receive REPCOMP of f +1 leaders of the same leadership
36: **if** $role_i = initiator$ **then** % the chosen role was to create a new leadership and cluster
37: CreateNewLeadership($REPCOMP, i$) % node i must create a new leadership and cluster
38: **else**
39: registry informations and save UR +1 to be inserted in the network
40: **end if**
41: **end if**

F. Analysis Techniques and Network Node Punishment

Analyzing the data from the IDS may be based on signatures or anomalies. We have chosen to apply an analysis technique based on anomalies, as it is then possible to identify variations of attacks. This technique also permits the evolution and updating of the system over time in a non-supervised manner. In our model, such analysis is done through the leaders, defined in algorithm 1 through the OctopusAnalysis function, making use of the already-implemented system, described in paper [5].

The technique utilized is based on the use of two layers. One classifies the data and uses Kohonen neural networks. Its function is to identify the type of attack at that moment. A second, more specialized layer was implemented utilizing support vector machines (SVM). Its function is to precisely identify the attacks.

In the event that leadership identifies malicious activity in a network node, leadership must send an alert message to neighboring cluster leadership and to the remaining collecting nodes of their cluster. This alert message will be sent in the next updating round as a consequence of the agreement among the $f+1$ cluster leaders. The malicious node will be punished through its inclusion on the list of malicious nodes for a pre-determined period (quarantine_time).

The network nodes (collectors or leaders) will exclude the malicious node from the routing table and include its identification in a list of malicious nodes. This list, present in each node, is composed of the MAC address of the malicious node, its public key, and its insertion time in the list. As such, all the legitimate network nodes will not send messages to the malicious node and will not retransmit messages originating from the malicious node during the established quarantine time. This solution makes it possible to detect various malicious nodes at the same time, given that each node possesses at least 2f +1 neighbors.

III. Related Studies

Intrusion detection systems for MANETs were proposed in [3], [2]. These systems use clusters to collaboratively detect intrusions. Each cluster possesses a leader which monitors all the traffic within its cluster. These studies have not used cryptography in message exchange, thus making it possible for various types of attacks to occur in the intrusion detection process. Nor have these systems assumed their leaders were alone in their clusters, with malicious behavior.

In another IDS [15], the node which detects suspect activity requests opinions from its neighbors concerning this suspect activity. After analyzing each neighbor's vote, the node makes a decision and informs it to the participating nodes who voted. However, this voting mechanism is vulnerable to message violation from and collusion with malicious nodes. In another study [16], a node hierarchy organizational model was developed on various levels, where the lowest level collects the data and the higher levels correlate the data sent to them. This study, to the contrary of the others cited here, permits the detection of several malicious nodes at the same time. However, the malicious nodes may only belong to the lower levels of the proposed hierarchy. In our proposal, any node may have malicious behavior, whether leaders or collectors. The only limitation to guarantee efficiency in our model is that the number of malicious nodes cannot exceed the f limit.

Studies concerning IDS for MANETs show that the majority of the systems proposed are capable of identifying few types of attacks or some routing protocol problems for these networks. In our proposal, we adopt a detection model based on anomalies. Thus we are able to identify and neutralize a large set of types of attacks and routing problems described in literature. Just as the architectures presented in [2], [3], [16], our model also assumes a hierarchical stratification. In these models, the hierarchical topology introduces the idea of clusters. The majority of studies in literature do not deal with the entrance aspects, departure aspects, or node mobility within the network. In no related study were we able to find simulated test results or real environment test results. In [3] a time period was established for the network to reorganize itself, in which the leaders could be re-elected through a voting process. Merely some of the IDS presented ([17], [16]), indicate the use of cryptographic mechanisms to secure property such as authenticity, confidentiality, and the integrity of messages exchanged between the IDS nodes.

The greatest contribution of this study, separating it from others present in literature, is the use of distributed systems concepts and dependability concepts applied to an IDS model for MANETs. The use of these concepts permitted - within certain limits - the development of a model less subject to restrictions. The proposed system is able to deal with various faulty or malicious nodes without there being interference in the network's normal behavior. Beyond this, our system is able to identify a large number of different attacks or variations of known attacks.

IV. Conclusions

In this paper, we presented our efforts to develop an IDS model for dynamic environments. This proposal is centered on a hierarchical malicious behavior detection model for MANETs. This model follows the concepts of dynamic distributed systems, permitting the presence of various non-malicious entities. The proposed model permits the correct functioning of the network while the faulty or malicious node limit is not exceeded. However, even with the f limit exceeded, the system continues to function. In such a case, there is no guarantee that our algorithms always work correctly.

References

[1] D. Djenouri, L. Khelladi, and A. Badache, "A survey of security issues in mobile ad hoc and sensor networks," in *Proceedings of Communications Surveys & Tutorials IEEE*, vol. 7, 2005, pp. 2–28.

[2] O. Kachirski and R. Guha, "Effective intrusion detection using multiple sensors in wireless ad hoc networks," in *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS)*, January 2003, pp. 1–8.

[3] E. Ahmed, K. Samad, and W. Mahmood, "Cluster-based intrusion detection (cbid) architecture for mobile ad hoc networks," in *AusCERT R&D Stream Program, Information Technology Security Conference*, Australia, May 2006, pp. 1–11.

[4] L. Bononi and C. Tacconi, "Intrusion detection for secure clustering and routing in mobile multi-hop wireless networks," *Int. J. Inf. Secur.*, vol. 6, no. 6, pp. 379–392, 2007.

[5] P. M. Mafra, V. Moll, J. da Silva Fraga, and A. O. Santin, "Octopus-iids: An anomaly based intelligent intrusion detection system," in *IEEE Symposium on Computers and Communications ISCC*, Riccione, Italy, 2010, pp. 405–410.

[6] T. Chandra and S. Toueg, "Unreliable failure detectors for reliable distributed systems," *J. ACM*, vol. 43 2, pp. 225–267, 1996.

[7] V. Shoup, "Practical threshold signatures." Springer-Verlag, 1999, pp. 207–220.

[8] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, 1978.

[9] T. P. Pedersen, "A Threshold Cryptosystem without a Trusted Party," *Advances in Cryptology - EUROCRYPT'91*, vol. 547, pp. 522–526, 1991.

[10] R. Gennaro, S. Jarecki, and H. Krawczyk, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems," *Journal of Cryptology*, vol. 20, no. 1, pp. 51–83, 2007.

[11] B. Liskov and R. Rodrigues, "Tolerating byzantine faulty clients in a quorum system," in *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2006, p. 34.

[12] F. C. Pereira, J. da Silva Fraga, A. E. Notoya, and R. F. Custdio, "Autoridade certificadora dinmica para redes ad hoc mveis," in *Anais do 25o Simpsio Brasileiro de Redes de Computedores (SBRC)*. SBC, 2007, pp. 191–204.

[13] Y. Desmedt, "Some Recent Research Aspects of Threshold Cryptography," in *ISW '97: Proceedings of the First International Workshop on Information Security*. London, UK: Springer-Verlag, 1998, pp. 158–173.

[14] A. Vora, M. Nesterenko, S. Tixeuil, and S. Delaët, "Universe detectors for sybil defense in ad hoc wireless networks," *CoRR*, vol. abs/0805.0087, 2008.

[15] S. A. Razak, S. M. Furnell, N. L. Clarke, and P. J. Brooke, "Friend-assisted intrusion detection and response mechanisms for mobile ad hoc networks," *Ad Hoc Netw.*, vol. 6, no. 7, pp. 1151–1167, 2008.

[16] D. Sterne and G. Lawler, "A dynamic intrusion detection hierarchy for manets," in *Sarnoff Symposium, 2009. SARNOFF '09. IEEE*, april 2009, pp. 1 –8.

[17] Y. Zhang, W. Lee, and Y. Huang, "Intrusion detection techniques for mobile wireless networks," *ACM/Kluwer Wireless Networks Journal*, vol. 9, no. 5, September 2003.