EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

Enhancing role-based trust management with a reputation system for MANETs

Rehan Akbani and Turgay Korkmaz*

Abstract

We start with role-based trust management (RBTM) and address some of the challenges associated with using RBTM in mobile ad hoc networks (MANETs). We then enhance RBTM with reputation systems (RSs), and propose a new hybrid trust management system (HTMS). In HTMS, the privilege level of an entity is determined not only by its role in the system, but also by its reputation score, which in turn is based on its behavior. If a privileged node becomes compromised and conducts several malicious or risky transactions, then its privilege level is quickly reduced to limit its access to resources and minimize the damage it can inflict further. The system uses a global, network-wide perspective to thwart global attacks. Such fine-grained variations of access control and dynamically assigning privilege levels would be very difficult to accomplish manually. We evaluated HTMS by comparing an implementation of it against an ideal response. We show that HTMS performs very close to the ideal if we can accurately estimate the proportion of malicious nodes in the network. We suggest using sampling to estimate this proportion. However, even if this estimate is not accurate, the results are still much better than using RBTM by itself.

EDICS: SYS-ARCH; SYS-PROT; FOR-DETE; SYS-INTR.

Keywords: MANETs, trust management, access control, machine learning

I. Introduction

A typical organization may have many resources, and entities which want to access those resources. For example, in a military setting, there may be resources, such as battle plans, communication systems, surveillance equipment, and weapons systems that may need to be accessed by different personnel at different times. Not all the personnel are granted full access rights to every resource, and so there must be a trust management system (TMS) in place to perform *access control*. However, it is cumbersome to enforce access rights based merely on an entity's username. Every resource would need to have a database of usernames that are allowed to access it, along with authentication mechanisms to verify the authenticity of usernames provided by users. Making changes to a person's access rights would be a daunting task as every resource that the person accesses would need to have its database updated. Furthermore, having separate databases and authentication mechanisms for each resource makes an attacker's task easier by providing him or her with more potential points of entry.

In response to overcoming some of the challenges in access control, Li et al. introduced role-based trust management (RBTM) [1]. In essence, RBTM combines the merits of some earlier studies by merging the concept of Roles from RBAC (role based access control) [2] with trust management [3], so that entities are granted access to resources based on their "roles." In Section II, we give more background information about RBTM and discuss its merits. However, one of the drawbacks of RBTM is that if an entity is compromised and behaves badly, RBTM cannot distinguish it from the other entities with the same role, and keep granting access to compromised nodes.

To address this problem, we can use reputation systems (RSs) that grant access to entities based on their past behavior [4]. This form of trust management is based on the "reputation" of a node as judged by other entities in the system. Reputation is the opinion of one entity about another [5]. This model is inspired by PGP's (pretty good privacy's) web of trust model [6]. In

* Correspondence: korkmaz@cs.utsa.edu
Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249, USA

this model a member node can vouch for a client node that wants access to a resource. The model is based on prior information about the client node [7-9]. A node builds its reputation by previous positive experiences with other nodes. What constitutes a "positive experience" is left to the discretion of the node that vouches for it. In essence, RSs try to predict the future behavior of a node by analyzing its past behavior. Good behavior results in escalation of access rights, whereas bad behavior results in the reduction of rights. However, since there is no notion of roles in RSs, it cannot be readily used for the organizations consisting of designated roles, such as the military, or corporations.

Clearly, a possible combination of these two approaches will be more suitable to manage and control access rights in practical systems. In this article, we specifically investigate how to combine RBTM and RS in the context of *closed* mobile ad hoc networks (MANETs).[a] A typical MANET may have several resources, such as printers, file servers, databases, web servers, etc. In addition, many nodes may provide different services as part of a larger service-oriented architecture (SOA) [10] approach. In SOA, large applications are modularized into smaller services which run on heterogeneous devices. It especially makes sense to use SOA in MANETs so that large, computationally expensive applications can be implemented on resource-constrained devices in a distributed fashion. However, from a security standpoint, we need a mechanism to regulate access to these resources and services so that we can guard them against insiders or outsiders who would misuse these resources or launch attacks against them. Often, different mechanisms are needed to defend the underlying network against the insiders or outsiders. In this article, we focus on defending against *insiders* by combining RBTM and RS. Defending against outsiders has been extensively investigated in our related study [11] and references therein.

In the context of MANETs, researchers also considered monitoring-based TM to deal with attacks against insiders. In this approach, each node's wireless traffic is monitored by its neighbors, and conclusions are drawn based on its behavior [12,13]. Many monitoring-based systems have been proposed in the literature [14-18], and the principle behind their operation is that traffic from a node is classified as legitimate or illegitimate by its neighbors. Examples of illegitimate traffic include known attack signatures, viruses, and worms, or anomalous behavior. A node that is behaving well and communicating mostly legitimate traffic is deemed to be trustworthy. Such a node accumulates "good credit" points through its good behavior and slowly gains access to increasingly sensitive resources. This model has a very fast response time. A misbehaving node can quickly be detected, and its traffic can rapidly be blocked. This is because all the information gathering and decision making is done within a node's one-hop neighborhood. Any detected malicious traffic cannot pass beyond this one-hop neighborhood. However, the most serious disadvantage of using monitoring-based TMSs is that they have been shown to raise too many false positives because of noise. According to [19], traditional, simulated noise models do not mimic the behavior of observed noise patterns in an actual setting, leading to optimistic results in simulations. In experiments done on MANET test beds, it was shown that monitoring-based systems do not work well as they raise too many false alarms [20]. Therefore, in our research, we do not consider using monitoring-based systems. Instead we focus on reputation and role-based systems.

Using reputation or role-based TM in MANETs presents a new set of challenges. This has to do with factors such as there is no online central authority, many nodes are limited in their computational resources, nodes may go offline at any time necessitating redundancy, and nodes are not guaranteed to be completely trustworthy [21]. To address some of the challenges associated with MANETs, we first start with RBTM. We then enhance RBTM with the RS that we proposed in our related study [22]. As a result, we create a new hybrid trust management system (HTMS) that can effectively defend against insider attacks by taking the advantages of both RBTM and RSs.

The motivation of HTMS is derived from "real life." For example, a soldier with a long history of working with the army is more reliable than a brand new soldier even though both have the same role. HTMS takes into account the role of an entity, as well as its historical behavior and experiences in order to decide which access rights to be granted to an entity. This approach has the advantages of not granting complete access to a new role, but gradually transferring more rights as the entity becomes more experienced. The process is automatic and it grants rights in small steps, making the access control fine grained. On the other hand, if an entity is deemed to misbehave and abuse its rights, then its rights are reduced even though it still belongs to the same role. In addition, the system stores a history of past transactions and their feedbacks so that administrators/supervisors can review the performance of any entity, and promote/demote it.

The rest of this article is organized as follows. In Section II, we first discuss some background material about RBTMs. In Section III, we address some of challenges in utilizing RBTMs in MANETs. We then describe the proposed HTMS and discuss its merits in Section IV. We evaluate HTMS and present our findings in Section

V. Finally, we conclude this article and give some directions for future studies in Section VI.

II. RBTM: Background

As mentioned in previous section, RBTM [23] combines the concept of Roles from RBAC [2] with Trust Management [3,24,25]. RBTM assigns "roles" to entities/nodes and allows them to access resources based on their roles [26]. In essence, roles serve to characterize entities and represent their arbitrary attributes, such as student, faculty, or staff [27]. In a commercial organization, some personnel might belong to the role "manager," some might belong to "clerical staff," "executive," and so on. All users belonging to a given role are granted certain access rights using policy statements called "credentials." For example, all executives might have complete access to a file server, while all clerks might have "read-only" access. Each credential is issued by an "issuer" to grant access rights to a "subject." The issuer also digitally signs the credential to avoid counterfeiting, possibly using X.509 style certificates [28].

To make the above mentioned concepts more concrete, let us consider the following example. Suppose a university wants to grant all the enrolled students the right to check out books from its library and the right to purchase parking permits. A tedious way of doing this would be that enrollment services (ES) would give the librarian as well as the police department an updated list of enrolled students each semester. However, RBTM would make the job much easier. First, the librarian (issuer) issues a credential to ES (subject) and delegates to it the right to decide who gets access to the library. The police department also issues another credential to ES, delegating to it the right to decide who can purchase a parking permit. This is illustrated by the following notation.

$$\text{Library.Checkout} \rightarrow \text{EnrollmentServices.Student}$$
$$\text{PoliceDept.Permit} \rightarrow \text{EnrollmentServices.Student}$$

ES in turn, issues a credential to all the enrolled students each semester. The credential is set to expire at the end of the semester. For instance, the student 'Alice' would receive the following signed credential.

$$\text{EnrollmentServices.Student} \rightarrow \text{Alice}$$

If Alice needs to use the library, she only needs to show this credential to the librarian. The librarian can reconstruct the following "credential chain:"

$$\text{Library.Checkout} \rightarrow \text{EnrollmentServices.Student} \rightarrow \text{Alice}$$

and grant access to Alice. The police department can similarly reconstruct its credential chain and allow Alice to purchase a parking permit. In this way, every enrolled student can enjoy the same access rights easily, without the need for ES to send an updated list of students every semester to every single department at the university.

The rights granted to entities are directly related to the job they need to perform. This follows the real world example where employees/students have access rights based on their job title. The main advantage of RBTM is that it avoids requiring different passwords for each entity or resource. Instead, any entity validates itself to other entities simply by presenting to them its signed credential that assigns to it a given role. The kinds of access rights granted to a given role are fixed beforehand. However, roles may be dynamically assigned to individual entities, and so for instance, an external auditor might temporarily be assigned a "manager" role for a fixed period of time. The main disadvantage of RBTM is that it treats all entities within a role equally. Common observation tells us that not all managers can be equally trusted; for example: some are most trustworthy then others based on their past experience and duration of employment with the firm. It would not be practical to assign such fine-tuned roles as "managers with 10 years experience," "managers with 5 years experience," etc. because then each role would contain very few entities, perhaps even just one entity. To overcome this problem, we propose to use a RS along with RBTM, as discussed in detail later.

III. Using RBTM in Manets

To be able to use RBTM in MANAETs, we need to (a) find secure and efficient mechanisms to store credentials in a distributed manner and (b) collect/determine credential chains under the following problems in MANETs [29-31];

(1) Attacks on the authenticity of entities, such as impersonation and Sybil attacks, which are relatively easy to be done in MANETs.

(2) Ease of eavesdropping which exposes the identity of communicating entities. This may be a problem if we would like to keep the identity of a user confidential, such as in a military, or financial, or medical setting.

(3) Problem of selfish nodes which refuse to cooperate, such as not providing or storing credentials, not complying with rules regarding access rights, etc. Actually, this should not be an issue in close MANETs during the normal operation. However, compromised insider nodes may create this problem.

(4) Unauthorized alteration of distributed/stored resources or exchanged data. It is difficult to police nodes that are in charge of a distributed resource.

(5) Limited computational resources, especially on the client side, where devices such as PDAs and cell phones might request services.

A. Credential storage

MANET system administrators need to decide who should store credentials and whether they are stored redundantly or not. As defined in [23], a credential is a signed certificate from an issuer, X, that grants certain rights to the subject, Y, denoted bys

$$X \rightarrow Y$$

Hence X could grant Y access rights to a resource. X could also delegate authority to Y so that Y could authorize other nodes. A credential chain can thus be formed:

$$X \rightarrow Y \rightarrow Z$$

In this case, X authorizes Y and Y grants access rights to Z. The chain could be arbitrarily long. It is also possible to construct a web of credentials where multiple issuers issue credentials to multiple subjects:

$$\begin{array}{ccccc} X \rightarrow & Y \rightarrow & Z & \rightarrow U \rightarrow & V \\ & \downarrow & \uparrow & \downarrow & \\ & S \rightarrow & W & T & \end{array}$$

This creates the problem of who should store the credentials [32]. Should the issuer of a credential store it, or the subject, or a third party? A third party could, for instance, be a credential storage server, but this would create scalability and availability issues since, if the server goes down, the system would be incapacitated. It is therefore desirable to have distributed storage of credentials. The location of the credential has significant consequences when it comes to credential distribution and collection, which is discussed in the next subsection.

Another problem that we need to address is whether the credentials should be stored redundantly or not. In the chain $X \rightarrow Y \rightarrow Z$, consider what would happen if Y were to suddenly go offline. Regardless of whether the issuer or the subject chooses to store a credential, if Y goes offline, we cannot complete the chain since Y is both an issuer and a subject. In this case, Z will not be able to gain access rights even though it is entitled to it. Let us say, we choose to store a credential with both its issuer as well as its subject for the sake of redundancy. Then, the scheme still fails in cases such as $W \rightarrow X \rightarrow Y \rightarrow Z$ if both X and Y were to go offline simultaneously, since the middle credential would become inaccessible. There is a tradeoff between redundancy and storage requirement based on the degree of robustness a network prefers.

Redundancy

Ideally, a MANET must store credentials redundantly. Since the entire system is distributed, we cannot store all the credentials at a centralized server. Therefore, each issuer (or subject) stores its own credentials. However, if an issuer such as X goes offline, we would like Z to be able to contact some other node to obtain X's credentials. One possibility is that we designate another node, W, to store a duplicate of all of X's credentials. In case X is unreachable, Z would contact W to obtain the credentials. Howeer, if a malicious node wants to target X and remove all its credentials, then it only needs to attack X and W and disable them.

We propose that instead of storing all the credentials in one backup node, X would randomly divide all its credentials into r equal-sized groups and distribute each group to r different nodes to store. Since each credential is signed by X, it cannot be tampered with. If X becomes unreachable, Z can contact each of the r nodes in turn to obtain the complete set of X's credentials. Using this approach, we do not increase the storage or bandwidth requirements, but we make it harder for an attacker to disable access to all of X's credentials (the total number of nodes accessed by Z is increased, however). To disable access to all of X's credentials, the attacker must bring down all r nodes along with X. Further redundancy can be introduced by storing each group of credentials on more than one node. This would increase storage requirement though, so a balance needs to be reached between storage requirement and the preferred degree of redundancy.

Which group of nodes X chooses to store its credentials on depends on the implementation. One possibility is that X can simply choose the nodes with the next higher (or lower) IP addresses compared to its own. Another possibility is that X would take a hash of its own ID. It would then mod the hash value with the number of nodes in the network, n, to decide which node should store the credentials. X can hash its ID twice, or thrice to obtain the next nodes that store the credentials. When Z needs to access X's credentials and X is offline, Z can repeat the same procedure to determine which node to go to ask for X's credentials. The advantage of this approach is that it can distribute credentials in a pseudo-random fashion within the network and avoid possible "clustering," where a small group of nodes end up storing most of the credentials.

B. Credential chain distribution and collection

The problem of searching for credentials is directly correlated with where the credentials are stored. In [32], the authors discuss three approaches for constructing the chain of credentials going from the subject; the client requesting a service, to the issuer; and the server offering the service. The first approach is a top-down approach where one would begin constructing the chain starting from the issuer's end and spanning out until one would encounter the subject at the end of the chain. The second approach is a bottom-up approach in

which one would start at the subject's end and span out backward until the issuer is reached. The third approach is a meet-in-the-middle approach where one would start at both the issuer's and the subject's ends and meet in the middle. It is important to realize that a node storing a given credential could be located anywhere within the MANET, and so a credential such as $X \rightarrow Y$ does not imply that X and Y are neighbors.

Some practical considerations need to be kept in mind when designing the system. Consider the case where Z needs to use a service provided by X. Z must find and present to X a credential chain authorizing Z to use its services. We assume that the chain X $\rightarrow$ Y $\rightarrow$ Z exists. Suppose we choose to store the credential with the subject. In such a case, it would make sense to use the bottom-up approach. Since Z has the credential Y $\rightarrow$ Z, it knows it must contact Y to obtain the next link up in the chain. Z would contact Y and ask for all the credentials it possesses to see if a chain back to X can be constructed. Once Z receives the credential X $\rightarrow$ Y, it stops searching. If such a credential is not found, then it would in turn contact each of the issuers of Y's credentials and fan out from there until it reaches X, or until the maximum allowed search depth is reached. The maximum search depth should be so chosen as to allow the most of the chains to be found while not being so large as to place an excessive load on the network and on the node performing the search.

Conversely, if the credential is stored with the issuer, then we should use the top-down approach. In that case, Z would begin at X and know that it needs to contact Y next when it sees the credential X $\rightarrow$ Y. A meet-in-the-middle approach would be effective if both the issuer and the subject store the credential.

IV. Proposed HTMS

The proposed HTMS tries to take the advantages of both the Role based and Reputation-based TMSs by enhancing RBTM with RS. In HTMS, each entity is assigned a role beforehand by an administrative authority. However, all entities belonging to the same role do not necessarily have the same access rights. Within each role, a range of access rights is defined from a minimum privilege level, to a maximum privilege level. For instance, minimum privilege level access could mean that a soldier can only receive communications. Soldiers with higher privilege levels could receive as well as send communications, whereas soldiers with maximum privilege levels could access battle plans.

Privilege levels are real values between 0 and 1, with 0 being the least privileged, and 1 being the most privileged. Nodes with 0 privilege level cannot access any resource on the network, whereas nodes with privilege level 1 have full access to all the resources on the network. When the administrative authority assigns a role to a new node, it defines a minimum and a maximum privilege level that the node can possess. At any point of time, the exact privilege level of the node is determined by its reputation score, but it will always fall within this range. The administrator then issues a digitally signed certificate to the node, with the following credentials:

$$\langle \textit{Node ID},\ \textit{Role},\ \textit{MinPL},\ \textit{MaxPL},\ \textit{Expiration Time} \rangle$$

where *Node ID* is the ID of the node, and *Role* is the role title, such as private, major, colonel, or general, and is only meant for human consumption.

MinPL and *MaxPL* are the numbers between 0 and 1, and where $MinPL \leq MaxPL$, respectively, denote the minimum and maximum privilege levels of a given role. *Expiration time* is the time when the certificate becomes invalid and must be renewed.

This certificate is signed by the administrator and given to the user, say John, who is assigned a role of major. All the majors have the same maximum and minimum privilege levels. John can carry this certificate on a smart card, or this certificate can be linked to John's login credentials so that it becomes active whenever John logs in to a machine. That machine then becomes a node with the Node ID given by John's certificate. In this way, the certificate is not linked with any particular machine, rather with the user himself who is free to switch machines in the network.

Whenever the node needs to access any service on the network, it will present this certificate to the server. The server will check what is the minimum privilege level required to access that particular service. This required privilege level is decided beforehand by the server administrator. Then, the server will proceed as follows:

(1) Verify the signature to ensure that the certificate originated from the administrative authority.

(2) If the required privilege level for the service is greater than the maximum privilege level on the certificate, then deny service.

(3) If the required privilege level for the service is less than the minimum privilege level on the certificate, then grant service.

(4) If the required privilege level for the service is in between the minimum and the maximum privilege on the certificate, then obtain the reputation score of the node and compute its privilege level at that point of time (explained below).

(5) If node's privilege level is greater than or equal to the required privilege level, then grant service; otherwise, deny service.

A. Determining the Exact Privilege Level

As mentioned before, the exact privilege level of a node at a given point of time is determined by its reputation score, which is computed based on the feedbacks obtained from other nodes through the underlying RS. In general, we can summarize existing RSs [33-38] within the general framework shown in Figure 1. According to this framework, each node that interacts with another node stores some feedback about the interaction. The interaction is either classified as legitimate or suspicious (or some value in between). If, for instance, the interaction consisted of downloading a file, the client could determine if the downloaded file was indeed the one requested, or it was a Trojan or a worm. Based on the feedback of various nodes, a new node can decide whether to transact with a given node or not, even though they may never have interacted before. The underlying principle is that a node that has behaved well in the past is likely to behave well in the future, and so it gets high reputation score. On the other hand, a node that has misbehaved in the past is likely to misbehave in the future, and so it gets low reputation score. For example, eBay utilizes this form of RS where previous buyers leave feedback about the seller so that prospective buyers can decide whether to buy from the seller or not [39].

HTMS can actually use any RS [4,34,35,37]. However, we chose to use our previously proposed Machine Learning-based RS [22,40] for our experiments. The reasons for this choice are mentioned in Section V. In any case, whichever RS the network administrator decides to use, the only requirement is that the RS outputs a reputation score between 0 and 1, with 0 being the least reputable. The RS needs to guard against the possibility of malicious nodes giving incorrect feedback to malign another node. In case the node has no history, the RS outputs a preset default score, for example, 0.5.

Suppose a client wants to interact with a server. In HTMS, the server needs to first obtain the reputation score of the client. Accordingly, the server broadcasts a feedback request throughout the network. Any node that has transacted with the client before responds to the request and sends back encrypted and digitally signed feedbacks to the server. The server decrypts and verifies the signature of the feedback, and then computes a reputation score using any RS implementation. Once the reputation score is obtained, the final privilege level is computed using the following equation:

$$PL = (MaxPL - MinPL) * RScore + MinPL \quad (1)$$

where PL is the current privilege level, $MaxPL$ and $MinPL$ are the maximum and minimum privilege levels respectively in the given role certificate. $RScore$ is the reputation score, as outputed by the RS. Equation 1 simply normalizes the privilege level between the minimum and maximum levels. An $RScore$ of 0 yields the minimum privilege level, whereas an $RScore$ of 1 yields the maximum privilege level. A server will grant access to the requested service if $PL \geq MinPL$ for that service.

B. Merits of HTMS

Thwarting Global Attacks

The rationale behind HTMS is to find a solution to the problem where a privileged node is compromised by an attacker and then used to damage the network in some way. This could be by trying to attack other nodes, or by conducting malicious transactions with servers. One transaction on its own may not seem malicious, but when a combination of potentially harmful transactions occur globally throughout the network, the results could be disastrous. For instance, one update by a colonel to battle plans may not seem harmful, but a series of small updates over time may change the battle plans entirely. HTMS aims to thwart such attacks by collecting data from multiple parties that have transacted with the node before, and then deciding whether to grant access to this node using a global picture.

For this to be effective, each server assigns a feedback score between -1 and 1 to every transaction that takes place on the server. Potentially harmful transactions, such as updates to battle plans, or changes to passwords are assigned a negative feedback score, whereas harmless transactions, such as reading or printing non-classified files, are assigned positive feedback scores. Similarly, transactions deemed to be malicious, such as uploading viruses or worms, or trying to initiate a buffer overflow attack, will be given negative feedback scores. All of these feedbacks are then used by the RS to compute the

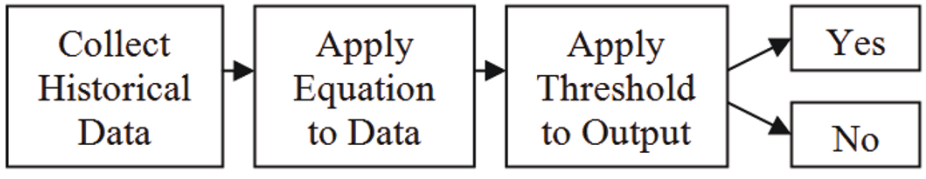

Figure 1 General framework of a RS that decides whether to transact with a given node or not.

overall reliability of the node, which in turn is used to determine access rights. A node that has conducted many potentially harmful transactions in the recent past will have a low reputation score, and service will be denied to it, even if it possesses a privileged role, thus thwarting any potential global attacks. At the very least, the attacker will have to slow down the attack, giving more time to the authorities to detect and defeat the attacker.

There might be some roles that routinely need to make mission-critical updates and these updates might give the role a bad reputation. Therefore, such crucial roles will have high minimum privilege levels so that they can do their job even with low reputation scores. However, extra precautions have to be taken to protect such roles from compromise.

Modeling Real World Scenario

The scheme models the real-world where an employee is given access rights based on his or her job title. At first, he or she is given a certain level of privilege. If he or she is observed to abuse his or her rights, some of his or her privileges are revoked. On the other hand, if he or she behaves well and gains the trust of he or her colleagues, he or she is rewarded with progressively higher access rights. This increase in rights may also be based on the fact that he or she will become more experienced over time and less likely to abuse the system inadvertently. Eventually, he or she may gain more and more trust and be promoted to a better position. Our scheme follows this model, and it is therefore intuitive.

Defense Against False Evidence

By maintaining a RS, we can deter a malicious node, M, from continuously presenting false evidence against X. Because the RS is based on evidence presented by several nodes, no single node can falsely incriminate X to significantly reduce its access rights. In any case, X's rights can never go below the minimum level within its role. Furthermore, if M repeatedly gives a low score to other nodes, it will hurt M's own reputation and diminish its own access rights.

Decentralized System

No single node holds all the evidence regarding any node. Therefore, if any one or a few nodes go offline then the system can still function well. The RS may be marginally affected since the contribution from the offline nodes will be missing, but it will usually not be critical if only a few nodes are offline.

Updating Roles

Periodically, the administrators responsible for role assignment can look at a node's past behavior and reconsider the role assigned to it. They could demote or promote a node to a different role (or job title) based on the behavior. Since each role credential carries an expiry date and time, temporary roles could also be assigned to guest entities for a predetermined amount of time.

Historical Record Keeping

This will encourage each node to always behave well so that it may retain or upgrade its access rights. This is similar to the credit score system maintained by credit agencies for individuals in the United States.

V. Evaluating HTMS

We use simulations to evaluate the efficacy of HTMS. Specifically, we focus on a node, X, within a large network of 1,000 nodes. Node X transacts with randomly selected nodes on the network. The behavior of node X varies randomly with time. Time is measured in units of days from 1 to 365 (i.e., one year). The *behavior* of a node on any given day is quantified as

$$\text{Proportion of Good Transactions} = \frac{\text{Number of Legitimate Transactions}}{\text{Total Number of Transactions}}.$$

To simulate the behavior of an actual node that may have been compromised, we use a randomly generated behavior (i.e., Proportion of Good Transactions) curve that fluctuates slowly over time, as illustrated by a dashed blue line in Figure 2.

Privilege levels are defined on a scale from 0 to 1, where 0 represents no access rights to any resource on the network, and 1 represents complete access to all available resources. In our simulation, the system administrator has assigned to node X a certain role, with a maximum privilege level of 0.8 and a minimum privilege level of 0.2. Other roles may have other corresponding minimum and maximum levels assigned by the administrator. The actual privilege level of node X depends on its behavior at any given point in time. In the ideal case, we would like the privilege level at any given time to be proportional to the behavior of the node at that time. If the behavior (i.e., Proportion of Good Transactions) is 1, then the privilege level should be maximum (i.e., 0.8). If the behavior is 0, then the privilege level should be minimum (i.e., 0.2). If the behavior is between 0 and 1, the privilege level will be computed using Equation 1. Hence, it will be any value between 0.2 and 0.8. Under no circumstances will the privilege level be higher than 0.8, or lower than 0.2. The ideal curve for the simulated behavior is illustrated by a solid red line in Figure 2.

The efficacy of a Trust Management (TM) mechanism will be measured by how well it can mimic the ideal curve as closely as possible. The ideal curve, therefore, represents the yardstick against which a TM implementation can be compared. Of course, in this simulation setup, we know the exact behavior of a node; but in an actual setting, we would need to estimate it using the

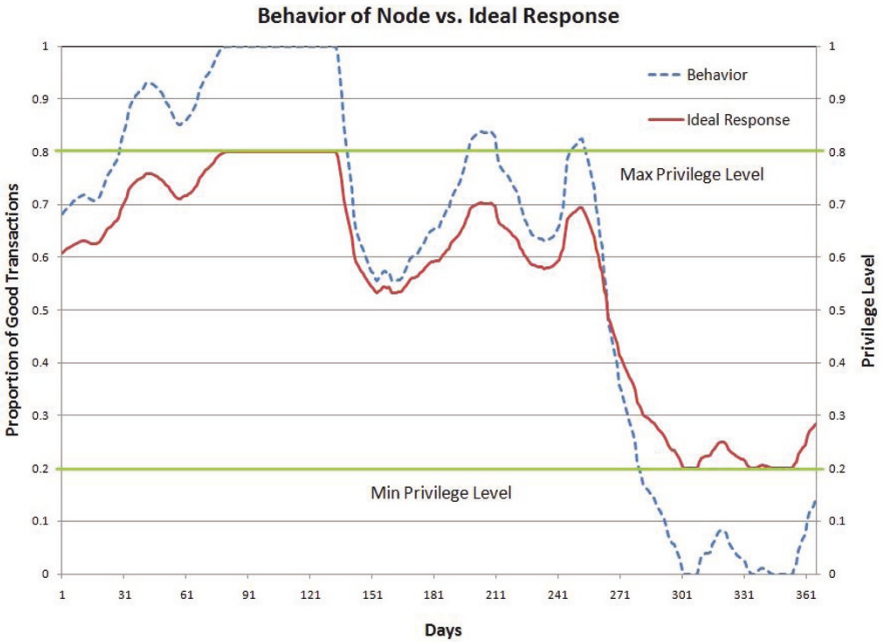

Figure 2 Randomly generated behavior of a node versus the corresponding ideal response curve, along with maximum and minimum privilege levels.

feedbacks obtained from other nodes through the underlying RS.

We used an implementation of the Support Vector Machine (SVM)-based RS that we proposed in a previous study [22] because of the following reasons:

(1) This RS has been shown to perform well with varying patterns of malicious behavior and varying proportions of malicious nodes.

(2) It protects against fake feedbacks about transactions that never really occurred.

(3) Since it is based on Machine Learning and SVM, it is easy to construct the RS model and automatically determine the model's parameters if there training data available [41].

We generated the training data using simulations on a different behavior curve. The proportion of malicious nodes in the network was varied in different simulations to obtain different training sets. A malicious node is defined as a node that lies and gives incorrect feedback about a node in an attempt to either decrease its reputation, or to increase it if the node is another colluding malicious node. The training sets were then used to train the SVM.

The test sets were generated using the behavior curve illustrated in Figure 2. Feedbacks were taken from the nodes in the network that had transacted with node X. The proportion of malicious nodes in the network was varied between 0% and 70%, so that the feedbacks were not always reliable. Each training and test instance consists of feedbacks obtained over the previous 7 days (i.e., one week). Accordingly, the privilege levels are automatically updated every 7 days so that a given privilege level is valid for one week.

In the first set of experiments, we used a training set consisting of 0% malicious nodes to train the SVM. Then, we tested this model against five different test sets consisting of 0-70% malicious nodes. The output of the model for each test set over time is plotted in Figure 3. The figure shows that the model closely mimics the ideal curve when the proportion of malicious nodes in the test set is also 0%, same as in the training set. For other proportions, the output deviates from the ideal curve, becoming almost a horizontal line at 50%. It becomes a mirror image of the ideal curve above 50%, increasing when the ideal curve decreases and vice versa. This is because after 50%, a majority of the nodes lie about the feedback, giving good feedback when the node is bad, and bad feedback when the node is good. This malicious majority overwhelms the feedback from the minority legitimate nodes, leading the SVM to reverse its output. At 50%, neither malicious nor legitimate nodes can overwhelm each other, and so the SVM produces a constant output of approximately 0.5.

We quantified the deviations of the test curves from the ideal curve by measuring the average overestimation of the curve, when the system grants more privilege than it should have, and underestimations of the curve, when the system grants fewer privileges than it should

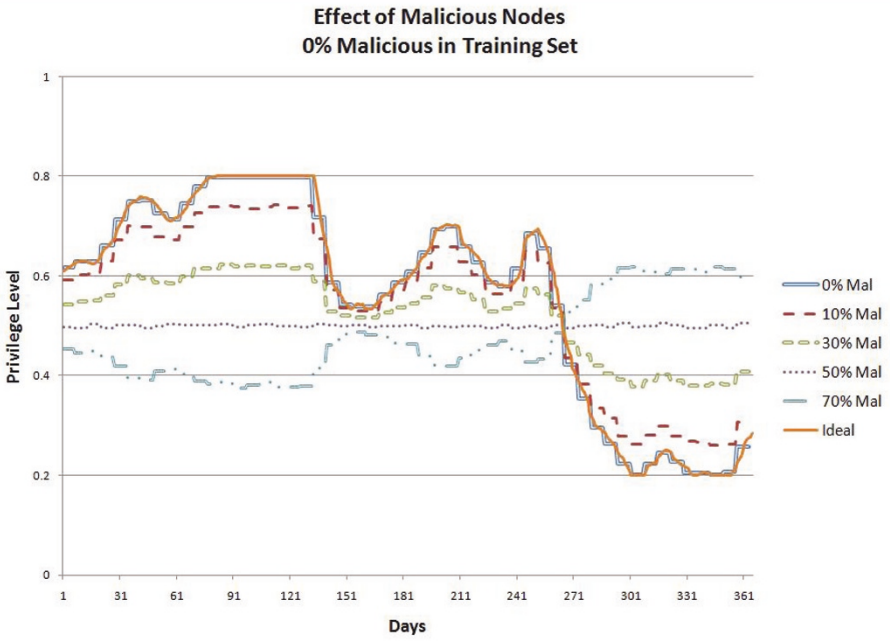

Figure 3 Effect of varying proportions of malicious nodes in test sets. Training set has 0% malicious nodes.

have. The daily averages of the over-and under-estimations are plotted in Figure 4. The sum of over-and under-estimation gives the overall average discrepancy from the ideal curve. As expected, the smallest discrepancy occurs when the percentage of malicious nodes in the test set is 0%, the same as in the training set. At 0%, the discrepancy is only about 0.008, which means that on average the system is off from the ideal privilege level by ±0.004. However, the discrepancy can be as high as 0.27, when 70% of the nodes are malicious.

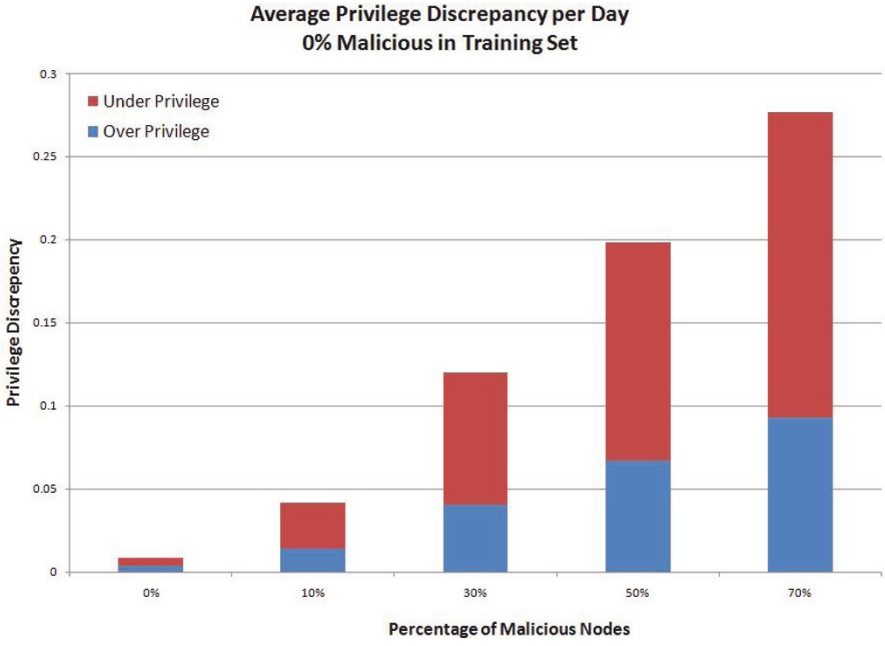

Figure 4 Average daily discrepancy from the ideal curve versus percentage of malicious nodes in the network (smaller is better). Training set has 0% malicious nodes.

We hypothesized that the discrepancy for a given test set would be minimized when the percentage of malicious nodes in the test set would be the same as in the training set. To test this hypothesis, we conducted further experiments, this time using different percentages of malicious nodes in the training sets. For the second set of experiments, we used 30% malicious nodes in the training set and repeated the experiments with the same test sets as before. The results are shown in Figures 5 and 6. Then, we repeated the experiments with 70% malicious nodes in the training set. The results are shown in Figures 7 and 8. The results support our hypothesis that the discrepancy is minimized when the percentage of malicious nodes in the test set is the same as in the training set.

A. Estimating Percentage of Malicious Nodes

Based on our results, we can conclude that to minimize the discrepancy, we need to train the SVM model using a training set that has approximately the same percentage of malicious nodes as the test set. Unfortunately, in a real-world setting, we do not know what the percentage of malicious nodes actually is, and it may vary with time. To overcome this problem, we propose generating several SVM models using different percentages of malicious nodes, for example, 0%, 10%, 20%, and so on. Then, we need to estimate the percentage of malicious nodes in the network, so that we can apply the appropriate model.

Estimation of the percentage of malicious nodes can be done by sampling. To begin with, a node uses a default SVM model, for instance, the one with 10% malicious nodes. Using this model, the node classifies other nodes that try to transact with it as legitimate or malicious and stores this sample. If the node ends up transacting with other nodes, then the outcomes of those transactions are also used for estimating the proportion of malicious nodes. When a sample of a certain size has been gathered in this way, the proportion of legitimate versus malicious nodes is estimated, and the SVM model corresponding to that estimate is used for future classifications. In this way, the proportion estimation is done dynamically, and may vary with time. We actually show next that a sample size of about 20-25 is sufficient for estimating the percentage of malicious nodes in a large network and selecting an SVM model with reasonable accuracy. However, even if we cannot accurately estimate this percentage, HTMS still performs better than just using RBTM which simply makes nodes have maximum privilege all the time.

To show the effectiveness of sampling method, we conducted some experiments to determine what a good sample size would be to estimate the percentage of malicious nodes. We ran simulations with various proportions of malicious nodes in the network. Then we randomly sampled the nodes and determined which of them were malicious. Based on the proportion of

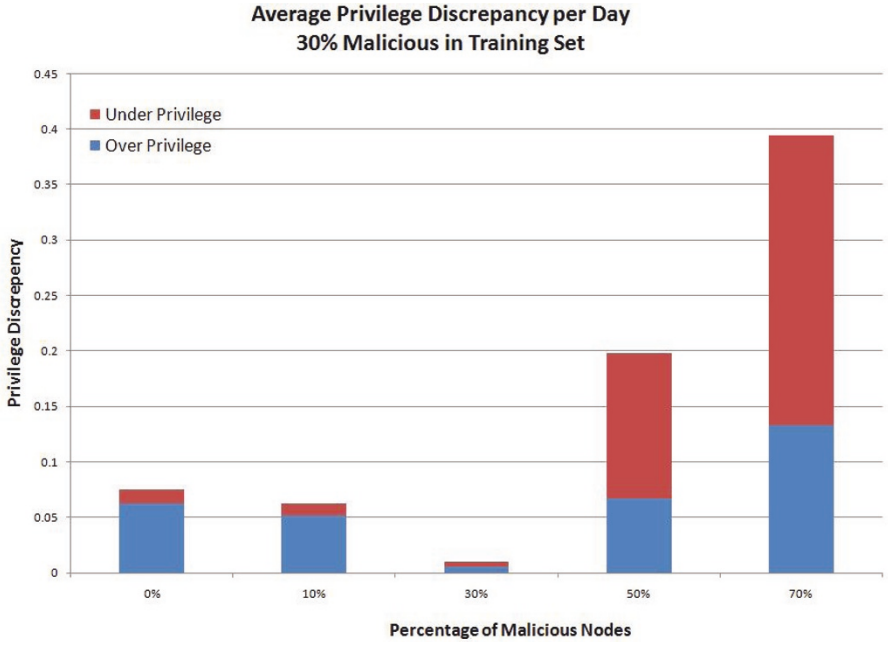

Figure 5 Effect of varying proportions of malicious nodes in test sets. Training set has 30% malicious nodes.

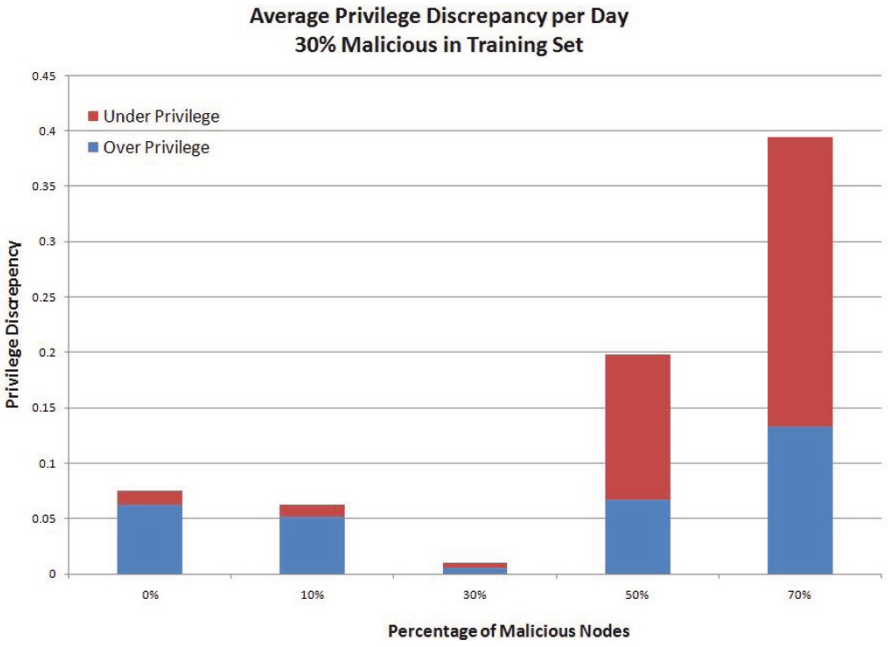

Figure 6 Average daily discrepancy from the ideal curve versus percentage of malicious nodes in the network. Training set has 30% malicious nodes.

malicious nodes in our random sample, we *estimated* the actual proportion in the entire network. Let A_{prop} denote the actual proportion of malicious nodes, and E_{prop} denote the *estimated* proportion of malicious nodes. Then, we can compute the absolute estimation error as $Error = E_{prop} - A_{prop}$.

Figure 9 shows the *Error* against various sample sizes. The results show that the error is erratic and unstable until a sample size of about 20. After 20 samples, a fairly good estimate of the actual proportion can be obtained. We therefore recommend that nodes should obtain a sample of at least 20 before adjusting their thresholds.

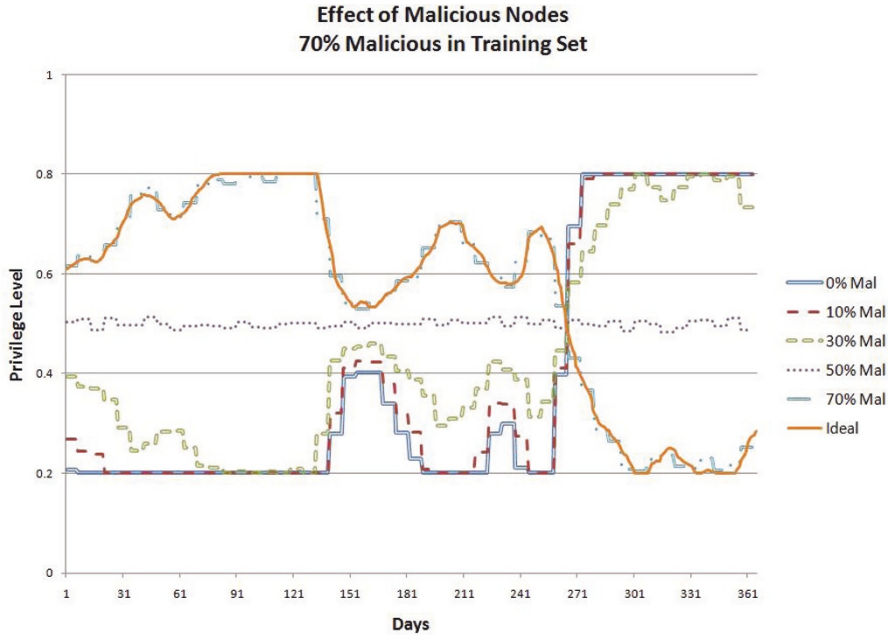

Figure 7 Effect of varying proportions of malicious nodes in test sets. Training set has 70% malicious nodes.

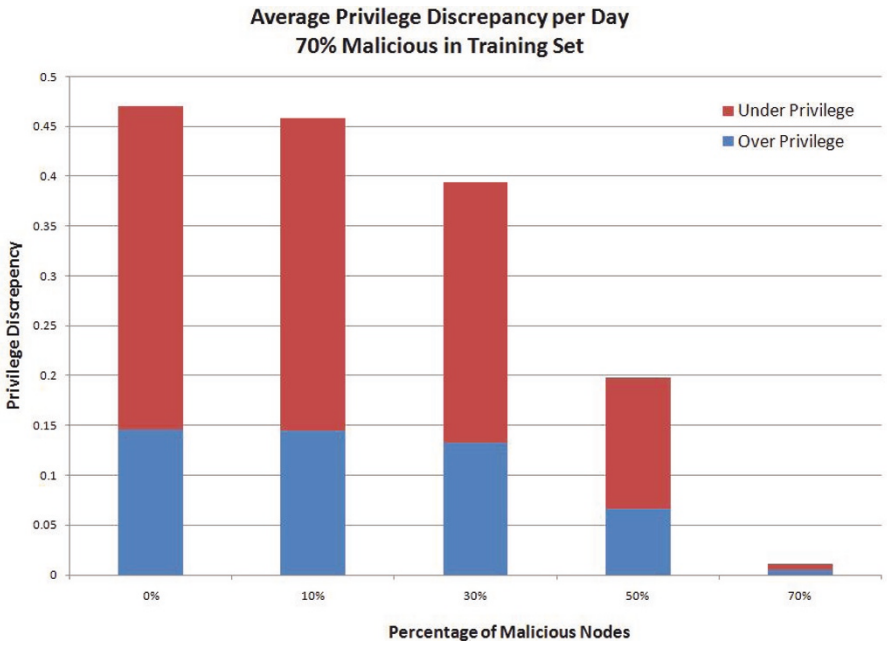

Figure 8 Average daily discrepancy from the ideal curve versus percentage of malicious nodes in the network. Training set has 70% malicious nodes.

VI. Conclusions and Future Work

In this article, we proposed a new HTMS that combines RBTM with RSs. At any given point of time, the privilege level of a node is determined not only by its role in the system, but also by its reputation score. The advantages of HTMS are that it allows automatic, fine-grained access control to network resources based on a node's behavior. If a privileged node becomes compromised and conducts several malicious or risky transactions, then its privilege level is quickly reduced to limit its access to resources and minimize the damage it can further inflict. This is accomplished by utilizing a global picture that is constructed by obtaining feedbacks from many sources on the network in order to determine

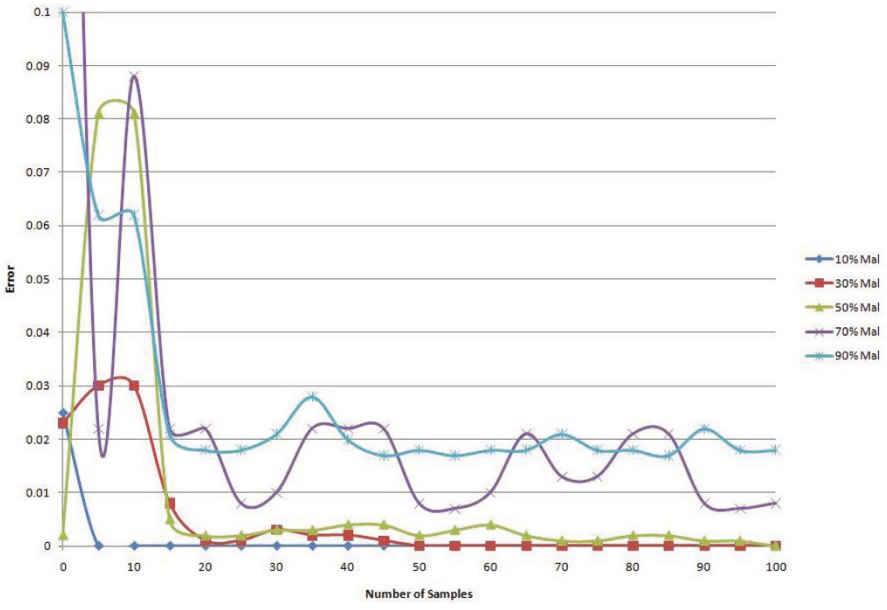

Figure 9 Absolute estimation error versus number of samples taken for different proportions of malicious nodes in the network.

access rights. If a node behaves well and conducts legitimate transactions, then more privileges are granted to it, providing an incentive to users to behave well. Such fine-grained access control and dynamically assigning privilege levels would be very difficult to accomplish manually.

Other advantages of the system are that it models real-world scenarios where experience and good behavior are rewarded with higher privileges, whereas novices and unreliable entities are only given limited rights. Furthermore, the system is decentralized and prevents a single node from damaging the reputation of others. It also keeps historical records of every node's behavior, making it easy for system administrators to monitor their behavior and promote or demote a node's role.

We evaluated the performance of a particular implementation of HTMS on a simulated network, using our previously proposed SVM-based RS [22]. We determined the ideal response that the system should have and compared the actual response with the ideal as a benchmark. The results show that the actual response is very close to the ideal response if the percentages of malicious nodes in the training and test sets are the same. Since we do not know this percentage in an actual network, we recommended using sampling to estimate this percentage and then using this estimate to determine which SVM model to use for generating maximum accuracy. Even if the estimate is not accurate, HTMS still performs better than just RBTM by itself, since RBTM would simply allow maximum privileges to the node all the time.

In future, we aim to devise better estimation mechanisms that can more accurately determine the percentage of malicious nodes in the network. We also aim to construct better SVM models that are more resilient to changes in the percentage of malicious nodes, and give us better results. This may be accomplished by trying different SVM kernels and varying its parameters.

Endnotes

[a]MANETs can be classified as open or closed. In an open MANET, anyone is free to enter or leave the network (e.g., in airports and university campuses), whereas in a closed MANET, only designated nodes are allowed to access the network (e.g., in a military setting). In general, it is more difficult to provide security in an open MANETs since there is no restriction on who may access the network. Fortunately, the security requirements of such networks are also not very demanding since users expect public networks to be insecure. By contrast, closed networks may have very strict security requirements, such as in the military or in the police department.

Acknowledgements
An abridged version of this article was presented in Proc. of MILCOM 2009 under the title of "A Hybrid Trust Management System For Automated Fine-Grained Access Control." This study is supported in part by the National Science Foundation and under grant CRI - 0551501 and by DoD Infrastructure Support Program for HBCU/MI under grant 54477-CI-ISP (UNCLASSIFIED).

Competing interests
The authors declare that they have no competing interests.

Received: 28 November 2010 Accepted: 6 September 2011
Published: 6 September 2011

References

1. N Li, JC Mitchell, WW Winsborough, Design of a role-based trust management framework, in *Proceedings of the 2002 IEEE Symposium on Security and Privacy* (Oakland), 1–17 (May 2002)
2. R Sandhu, E Coyne, H Feinstein, C Youman, Role-based access control models. IEEE Comput. **29**(2), 38–47 (1996)
3. M Blaze, J Feigenbaum, J Ioannidis, A Keromytis, The KeyNote trustmanagement system, version 2. IETF Tech. Rep. RFC 2704 (Sept 1999)
4. A Josang, R Ismail, C Boyd, A survey of trust and reputation systems for online service provision. Decis Support Sys. **2**(43), 618–644 (2007)
5. A Srinivasan, J Teitelbaum, H Liang, J Wu, M Cardei, *Reputation and Trust based System for Ad Hoc and Sensor Networks*, (Wiley & Sons, Chichester, 2006), pp. 1–36
6. N. A. Inc., An introduction to cryptography, Chap 1, 30–33 (1999) ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf
7. M Bhattacharyya, S Nandi, S Saha, A soft dynamic service specific trust management and authentication scheme for mobile ad hoc networks. 2006 IFIP International Conference on Wireless and Optical Communications Networks (Bangalore, India) 1–5 (April 2006)
8. G Theodorakopoulos, JS Baras, On trust models and trust evaluation metrics for ad hoc networks. IEEE J Selected Areas Commun. **24**(2), 318–328 (2006)
9. L Xiaoqi, Trust model based self-organized routing protocol for secure ad hoc networks, PhD term paper, (Chinese University of Hong Kong, 2003)
10. T Erl, in *Service-Oriented Architecture: Concepts, Technology, and Design*, (Prentice Hall PTR, Upper Saddle River, 2005)
11. R Akbani, T Korkmaz, G Raju, Heap: A packet authentication scheme for mobile ad hoc networks. Ad Hoc Netw. **6**(7), 1134–1150 (2008). doi:10.1016/j.adhoc.2007.11.002
12. JS Baras, T Jiang, Managing trust in self-organized mobile ad hoc networks, in *Proceedings of the 12th Annual Network and Distributed System Security Symposium Workshop*, (IEEE Computer Society, San Diego, CA, Feb. 2005), pp. 1–2
13. T Jiang, JS Baras, Ant-based adaptive trust evidence distribution in MANET, in *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops - W7: EC (ICDCSW04), ser. ICDCSW 04*, vol. 7. (IEEE Computer Society, Washington, DC, USA, 2004), pp. 588–593
14. A Srinivasan, J Wu, J Teitelbaum, Distributed reputation-based secure localization in sensor networks. Special Issue on Journal of Autonomic and Trusted Computing (JoATC), 3, 1–13 (2008)
15. S Marti, T Giuli, K Lai, M Baker, Mitigating routing misbehavior in mobile ad hoc networks, in *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, 255–265 (2000)
16. S Buchegger, J-Y Le Boudec, Performance analysis of the confidant protocol, in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing, ser. MobiHoc 02*, (ACM, New York, NY, 2002), pp. 226–236
17. P Michiardi, R Molva, Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks, in *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*, (Kluwer, B.V., Deventer, The Netherlands, 2002), pp. 107–121
18. S Bansal, M Baker, Observation-based cooperation enforcement in ad hoc networks (Standford University, Research Report cs. NI/0307012) (2003)
19. X Su, RV Boppana, On the impact of noise on mobile ad hoc networks, in *Proceedings of the 2007 international conference on Wireless communications and mobile computing, ser. IWCMC 07*, (ACM, New York, NY, 2007), pp. 208–213

20. RV Boppana, X Su, An analysis of monitoring based intrusion detection for ad hoc networks, in *Proceedings of IEEE Globecom: Computer and Communication Network Security Symposium* (IEEE, New Orleans, LA, Nov. 30-Dec. 4 2008), pp. 1–5
21. H Yang, H Luo, F Ye, S Lu, L Zhang, Security in mobile ad hoc networks: challenges and solutions. IEEE Wireless Commun. **11**(1), 38–47 (2004). doi:10.1109/MWC.2004.1269716
22. R Akbani, T Korkmaz, G Raju, Defending against malicious nodes using an SVM based reputation system, in *Proceedings of MILCOM, Military Communications Conference*. IEEE MILCOM 2008. (San Diego, CA 16-19 Nov. 2008), pp. 1–7 (2008)
23. N Li, JC Mitchell, W Winsborough, Design of a role-based trust management framework, in *Proceedings of the 2002 IEEE Symposium on Security and Privacy* (May 2002)
24. C Ellison, B Frantz, B Lampson, R Rivest, B Thomas, T Ylonen, The keynote trustmanagement system, version 2. IETF, Tech. Rep. RFC 2693 (Sept 1999)
25. D Clarke, J Elien, C Ellison, M Fredette, A Morcos, R Rivest, Certificate chain discovery in SPKI/SDSI. J Comput Secur. **9**(4), 285–322 (2001)
26. T Simon, M Zurko, Separation of duty in role-based environments, in *Proceedings of the 10th Computer Security Foundations Workshop*, (IEEE Computer Society Press, June 1997), pp. 183–194
27. A Herzberg, Y Mass, J Michaeli, D Naor, Y Ravid, Access control meets public key infrastructure, or: Assigning roles to strangers, in *Proceedings of the IEEE Symposium on Security and Privacy* (Berkeley), 214 (2000)
28. D Chadwick, A Otenko, E Ball, Role-based access control with x.509 attribute certificates. Internet Comput, **7**(2), 62–69. 2-3
29. Q Memon, S Akhtar, AA Aly, Role management in adhoc networks, in *Proceedings of the 10th Communications and Networking Simulation Symposium* (SCS and ACM, March 2007), pp. 131–137
30. V Tsetsos, GF Marias, S Paskalis, in *Trust Management Issues for Ad Hoc and Self-organized Networks, ser. Autonomic Communication, LNCS*, vol. 3854. (Springer Berlin, Heidelberg, 2006), pp. 153–164
31. S Keoh, E Lupu, M Sloman, PEACE: a policy-based establishment of ad-hoc communities, in *Proceedings of the 20th Annual Computer Security Applications Conference* (Tucson, AZ), pp. 386–395 (Dec. 2004)
32. N Li, W Winsborough, JC Mitchell, Distributed credential chain discovery in trust management. J Comput Secur. **11**(1), 35–86 (2003)
33. T Jiang, JS Baras, Trust evaluation in anarchy: a case study on autonomous networks, in *Proceedings of the 25th Conference on Computer Communications* (Barcelona, Spain), pp. 23–29 (April 2006)
34. M Srivatsa, L Xiong, L Liu, TrustGuard: countering vulnerabilities in reputation management for decentralized overlay networks, in *Proceedings of the 14th international conference on World Wide Web, ser. WWW 05*. (ACM, New York, NY, 2005), pp. 422–431
35. SD Kamvar, MT Schlosser, H Garcia-Molina, The eigentrust algorithm for reputation management in p2p networks, in *Proceedings of the 12th international conference on World Wide Web, ser. WWW 03*. (ACM, New York, NY, 2003), pp. 640–651
36. J Liu, V Issarny, Enhanced reputation mechanism for mobile ad hoc networks, in *Proceedings of iTrust 2004, ser. Springer-Verlag. Lecture Notes in Computer Science* (Oxford, UK), pp. 48–62 (March 2004)
37. A Josang, R Ismail, The beta reputation system, in *Proceedings of 15th Bled Electronic Commerce Conference* (Bled, Slovenia), pp. 1–14 (2002)
38. L Mui, M Mohtashemi, A Halberstadt, A computational model of trust and reputation, in *Proceedings of the 35th Hawaii International Conference on System Sciences* (IEEE Computer Society, Big Island, HI, 2002), pp. 188–196
39. EBay http://www.ebay.com
40. R Akbani, Defending against malicious nodes in closed MANETs through packet authentication and a hybrid trust management system, in Dissertation, (The University of Texas at San Antonio, August 2009) http://www.cs.utsa.edu/uploads/theses/rAkbani09.pdf
41. S Abe, in *Advances in pattern recognition*. Support Vector Machines for Pattern Classification, 2nd edn. (Springer-Verlag New York, LLC, 2010)

doi:10.1186/1687-1499-2011-90
Cite this article as: Akbani and Korkmaz: **Enhancing role-based trust management with a reputation system for MANETs.** *EURASIP Journal on Wireless Communications and Networking* 2011 **2011**:90.