Trust Management for Host-based Collaborative Intrusion Detection

Carol J Fung, Olga Baysal, Jie Zhang, Issam Aib, and Raouf Boutaba

David R. Cheriton School of Computer Science
University of Waterloo, Canada
{j22fung,obaysal,j44zhang,iaib,rboutaba}@uwaterloo.ca

Abstract. The accuracy of detecting an intrusion within a network of intrusion detection systems (IDSes) depends on the efficiency of collaboration between member IDSes. The security itself within this network is an additional concern that needs to be addressed. In this paper, we present a trust-based framework for secure and effective collaboration within an intrusion detection network (IDN). In particular, we define a trust model that allows each IDS to evaluate the trustworthiness of others based on personal experience. We prove the correctness of our approach in protecting the IDN. Additionally, experimental results demonstrate that our system yields a significant improvement in detecting intrusions. The trust model further improves the robustness of the collaborative system against malicious attacks.

Key words: Intrusion detection Network, Trust Management, Collaboration, Peer-to-Peer, Security

1 Introduction

Intrusions over the Internet are becoming more dynamic and sophisticated. Intrusion Detection Systems (IDSes) identify intrusions by comparing observable behavior against suspicious patterns. They can be network-based (NIDS) or host-based (HIDS). Traditional IDSes work in isolation and may be easily compromised by unknown or new threats. An Intrusion Detection Network (IDN) is a collaborative IDS network intended to overcome this weakness by having each members IDS benefit from the collective knowledge and experience shared by other member IDSes. This enhances their overall accuracy of intrusion assessment as well as the ability of detecting new intrusion types.

Intrusion types include worms, spamware, viruses, denial-of-service(DoS), malicious logins, etc. The potential damage of these intrusions can be significant if they are not detected promptly. An example is the Code Red worm that infected more than 350,000 systems in less than 14 hours in 2001 with a damage cost of more than two billion dollars [7]. IDS collaboration can also be an effective way to throttle or stop the spread of such contagious attacks.

Centralized collaboration of IDSes relies on a central server to gather alerts and analyze them. This technique suffers from the performance bottleneck problem. In addition the central server is a single point of failure and may become

the target of denial-of-service attacks. The distributed collaboration of IDSes can avoid these problems. However, in such a collaborative environment, a malicious IDS can degrade the performance of others by sending out false evaluations about intrusions. To protect an IDS collaborative network from malicious attacks, it is important to evaluate the trustworthiness of participating IDSes, especially when they are Host-based IDSes (HIDSes). Duma et al. [3] propose a simple trust management model to identify dishonest insiders. However, their model is vulnerable to some attacks which aim at compromising the trust model itself (see Sections 4 and 5 for more details).

In this work, we develop a robust trust management model that is suitable for distributed HIDS collaboration. Our model allows each HIDS to evaluate the trustworthiness of others based on its own experience with them. We also propose a framework for efficient HIDS collaboration using a peer-to-peer network. Our framework provides identity verification for participating HIDSes and creates incentives for collaboration amongst them.

We evaluate our system based on a simulated collaborative HIDS network. The HIDSes are distributed and may have different expertise levels in detecting intrusions. A HIDS may also become malicious in case it has been compromised (or the HIDS owner deliberately makes it malicious). We also simulate several potential threats. Our experimental results demonstrate that our system yields a significant improvement in detecting intrusions and is robust to various attacks as compared to that of [3].

The rest of the paper is organized as follows. Section 2 presents the collaborative IDS framework and collaboration management mechanisms. Section 3 formalizes our trust management model. Section 4 addresses common attacks on the collaboration and proves how our trust model cancels them. Section 5 presents the different simulation settings used and discusses the obtained results. Section 6 discusses related work. Finally, Section 7 summarizes our contributions and addresses future work directions.

2 HIDS Collaboration Framework

The purpose of this framework is to connect individual HIDSes so that they can securely communicate and cooperate with each other to achieve better intrusion detectability. Figure 1 illustrates the key components of our framework. Collaboration is ensured by trust-based cooperation and peer-to-peer communication. The trust management model allows a HIDS to evaluate the trustworthiness of its neighbors based on its own experience with them. The P2P component provides network organization, management and communication between the HIDSes. The collaboration is ensured by three processes, which are explained in the following.

2.1 Network Join Process

In our framework, each HIDS connects to other HIDSes over a peer-to-peer network. Before joining the network, a HIDS node needs to register to a trusted

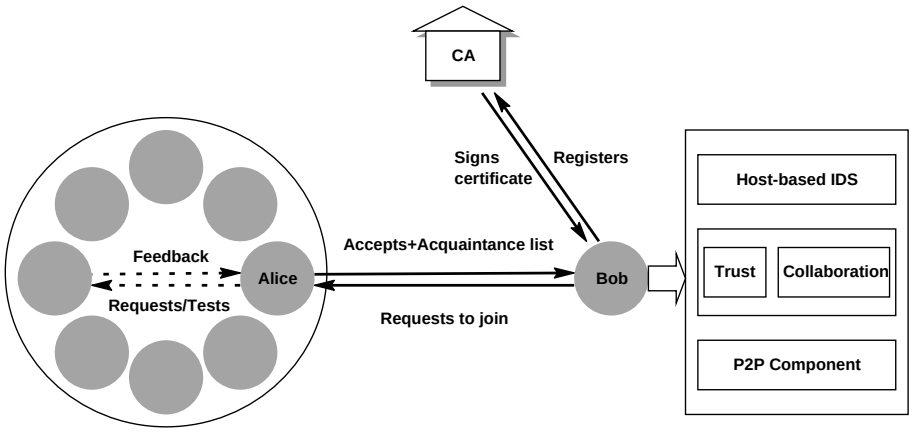

Fig. 1. IDS Collaboration Framework

digital certificate authority (Figure 1) and get a public and private key pair which uniquely identifies it. Note that we identify the (machine, user) tuple. This is because a different machine means a different HIDS instance. In addition, a different user of the same machine may have a different configuration of its HIDS. After a node joins the HIDS network, it is provided with a preliminary *acquaintance list*. This list is customizable and contains IDs (or public keys) of other nodes within the network along with their *trust* values and serves as the contact list for collaboration.

2.2 Test Messages

Each node sends out either *requests* for alert ranking (consultation), or *test messages*. A test message is a consultation request sent with the intention to evaluate the trustworthiness of another node in the acquaintance list. It is sent out in a way that makes it difficult to be distinguished from a real alert ranking request. The testing node knows the *severity* of the alert described in the test message and uses the received feedback to derive a trust value for the tested node. This technique helps in uncovering inexperienced and/or malicious nodes within the collaborative network.

2.3 Incentive Design

Our framework also provides *incentives* to motivate collaboration. Nodes that are asked for consultation will reply to only a number of requests in a certain period of time because of their limited bandwidth and computational resources. Thus, only highly trusted nodes will have higher priority of receiving help whenever needed. In this way, nodes are encouraged to build up their trust. In addition, our system accepts the reply of "*don't know*" to requests in order to encourage active collaboration in the network. This is explained in section 3.1.

3 Trust Management Model

This section describes the model we developed to establish trust relationships between the HIDSes in the collaborative environment. We first describe how we evaluate the trustworthiness of a HIDS and then present a method to aggregate feedback responses from trusted neighbours.

3.1 Evaluating the Trustworthiness of a node

The evaluation of the trustworthiness of a node is carried out using test messages sent out periodically using a random poisson process. After a node receives the feedback for an alert evaluation it assigns a satisfaction value to it, which can be "very satisfied" (1.0), "satisfied" (0.5), "neutral" (0.3), "unsatisfied" (0.1), or "very unsatisfied" (0).

The trust value of each node will be updated based on the satisfactory levels of its feedback. More specifically, the replies from a node i are ordered from the most recent to the oldest according to the time t_k at which they have been received by node j. The *trustworthiness* of node i according to node j can then be estimated as follows:

$$tw_i^j(n) = \frac{\sum_{k=0}^{n} S_k^{j,i} F^{t_k}}{\sum_{k=0}^{n} F^{t_k}} \tag{1}$$

where $S_k^{j,i} \in [0,1]$ is the satisfaction of the reply k and n is the total number of feedback. To deal with possible changes of the node behavior over time, we use a *forgetting factor* F ($0 \leq F \leq 1$) which helps in assigning less weight to older feedback responses [10]. Compared to Duma et al. [3], our model uses multiple satisfaction levels and forget old experiences exponentially, while [3] only uses two satisfaction levels (satisfied and unsatisfied) and all experiences have the same impact.

We also allow a node to send a "don't know" answer to a request if it has no experience with the alert or is not confident with its ranking decision. However, some nodes may take advantage of this option by always providing "don't know" feedback responses so as to maintain their trust values. In order to encourage nodes to provide satisfactory feedback responses whenever possible, the trust value will be slowly updated every time the node provides a "don't know" answer. The trustworthiness of a node i according to node j is then formulated as follows:

$$T_i^j = (tw_i^j - T_{stranger})(1-x)^m + T_{stranger}, \tag{2}$$

where x is the percentage of "don't know" answers from time t_0 to t_n. m is a positive incentive parameter (*forgetting sensitivity*) to control the severity of punishment to "don't know" replies, tw_i^j is the trust value without the integration of "don't know" answers (Equation 1), and $T_{stranger}$ is the default trust

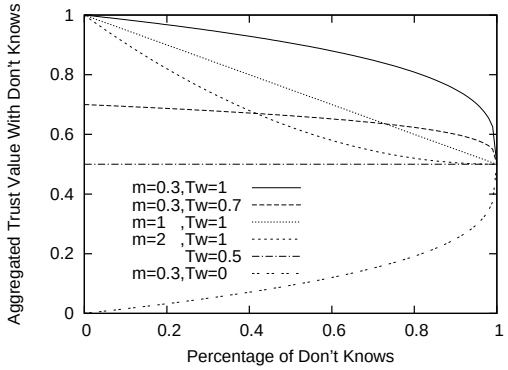

Fig. 2. Trust Convergence Curves with "don't know" answers

value of a stranger. As illustrated in Figure 2, Equation 2 causes trust values to converge to $T_{stranger}$ with the increase in the percentage of don't "know answers". Eventually, the trust value will become that of a stranger. This allows the trust value of an untrusted node to slowly increase up to the level of a stranger by providing "don't know" answers. In addition, nodes with little experience are motivated to provide "don't know" answers rather than incorrect alert rankings.

3.2 Feedback Aggregation

Based on the history of trustworthiness, each node i requests alert consulting only from those nodes in its acquaintance list whose trust values are greater than a threshold th_t^i. We also consider *proximity*, which is a measure of the physical distance between the node that provides feedback and the node that sends the request. Physical location can be an important parameter in intrusion detection. HIDSes that are located in the same or close by geographical region are likely to experience similar intrusions [1] and thus can help each other by broadcasting warnings of active threats. Feedback from nearby acquaintances is therefore more relevant than that from distant ones. We scale the proximity based on the region the node belongs to.

After receiving feedback from its acquaintances, node j aggregates the feedback using a *weighted majority* method as follows:

$$R_j(a) = \frac{\sum\limits_{T_i^j \geq th_t^j} T_i^j D_i^j R_i(a)}{\sum\limits_{T_i^j \geq th_t^j} T_i^j D_i^j}, \tag{3}$$

where $R_j(a)$ is the aggregated ranking of alert a from the feedback provided by each node belonging to the acquaintance list A_j of node j. T_i^j ($\in [0,1]$) is the trust value of node i according to node j. D_i^j ($\in [0,1]$) is the proximity weight of

node i. th_t^j is the trust threshold set by node j. $R_i(a)$ ($\in [0,1]$) is the feedback ranking of alert a by node i.

Compared to [3], our model only integrate feedback from trusted nodes while [3] integrates feedback from all neighbors.

4 Robustness against Common Threats

Trust management can effectively improve network collaboration and detect malicious HIDSes. However, the trust management itself may become the target of attacks and be compromised. In this section, we describe possible attacks and provide defense mechanisms against them.

Sybil attacks occur when a malicious node in the system creates a large amount of pseudonyms (fake identities) [2]. This malicious node uses fake identities to gain larger influence of the false alert ranking on others in the network. Our defense against sybil attacks relies on the design of the authentication mechanism. Authentication makes registering fake IDes difficult. In our model, the certificate issuing authority only allows one ID per IP address. In addition, our trust management model requires IDSes to first build up their trust before they can affect the decision of others, which is costly to do with many fake IDes. Thus, our security and trust mechanisms protect our collaborative network from sybil attacks.

Identity cloning attacks occur when a malicious node steals some node' identity and tries to communicate with others on its behalf. Our communication model is based on asymmetric cryptography, where each node has a pair of public and private keys. The certificate authority certifies the ownership of key pairs and in this way protects the authenticity of node identities.

Newcomer attacks occur when a malicious node can easily register as a new user [8]. Such a malicious node creates a new ID for the purpose of erasing its bad history with other nodes in the network. Our model handles this type of attack by assigning low trust values to all newcomers, so their feedback on the alerts is simply not considered by other nodes during the aggregation process.

Betrayal attacks occur when a trusted node suddenly turns into a malicious one and starts sending false alerts or even malware. A trust management system can be degraded dramatically because of this type of attacks. We employ a mechanism which is inspired by the social norm:

> *It takes a long-time interaction and consistent good behavior to build up a high trust, while only a few bad actions to ruin it.*

When a trustworthy node acts dishonestly, the forgetting factor (Eqn.1) causes its trust value to drop down quickly, hence making it difficult for this node to deceive others or gain back its previous trust within a short time.

Collusion attacks happen when a group of malicious nodes cooperate together by providing false alert rankings in order to compromise the network. In our system, nodes will not be adversely affected by collusion attacks. In our trust model each node relies on its own knowledge to detect dishonest nodes.

In addition, we use test messages to uncover malicious nodes. Since the test messages are sent in a random manner, it will be difficult for malicious nodes to distinguish them from actual requests.

5 Simulations and Experimental Results

In this section, we present the experiments used to evaluate the effectiveness and robustness of our trust-based IDS collaboration framework.

5.1 Simulation Setting

In our simulation model, we have n nodes in the collaboration network randomly distributed in a $s \times s$ grid region. The proximity weight of nodes is anti-proportional to the distance between the nodes in the number of grid steps. The expertise levels of the nodes can be low(0.1), medium(0.5) or high(0.95). In the beginning, each node builds an initial acquaintance list based on the communication cost (proximity). The initial trust values of all nodes in the acquaintance list are set to the stranger trust value ($T_{stranger}$). To test the trustworthiness of all the acquaintances in the list, each node sends out test messages following a Poisson process with average arrival rate λ_t. The intrusion detection expertise of a HIDS is modeled using a beta function. A honest HIDS always generates feedback based on its truthful judgment, while a dishonest HIDS always sends feedback opposite to its truthful judgment. The parameters used in the simulation are shown in Table 1.

To reflect a HIDS expertise level, we use a beta distribution for the decision model of HIDSes. The beta density function is expressed in Eqn.5:

$$f(p|\alpha,\beta) = \frac{1}{B(\alpha,\beta)} p^{\alpha-1}(1-p)^{\beta-1}, \tag{4}$$

$$B(\alpha,\beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt, \tag{5}$$

Table 1. Simulation Parameters for Experiments

Parameter	Value	Description
λ_t	5/day	Test messages frequency
F	0.9	Forgetting factor
$T_{stranger}$	0.5	Stranger trust value
th_t	0.8	Trust threshold
m	0.3	Forgetting sensitivity
Th_{DK}	1	Threshold of "don't know" replies
s	4	Size of grid region
n	30 (+10)	Number of HIDS

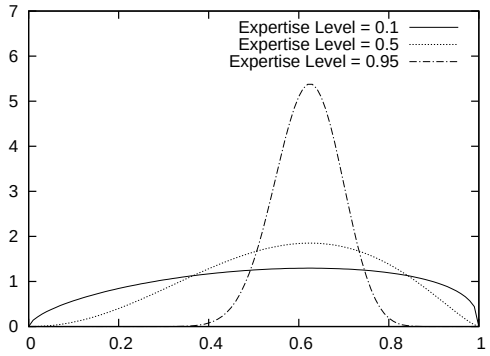

Fig. 3. Decision DF for Expertise Levels

We define α and β as:

$$\alpha = 1 + \frac{(1-D)}{D}\sqrt{\frac{E(b-1)}{1-E}}, \qquad \beta = 1 + \frac{(1-D)(1-E)}{DE}\sqrt{\frac{E(b-1)}{1-E}} \tag{6}$$

where E is the expected ranking of the alert. D ($\in [0,1]$) denotes the difficulty level of a alert. Higher values for D are associated to attacks that are difficult to detect, i.e. many HIDSes fail to identify them. $L(\in [0,1])$ denotes the expertise level of an IDS. A higher value for L reflects a higher probability of producing correct rankings for alerts. $f(p;\alpha,\beta)$ is the probability that the node with expertise level L answers with a value of p ($1 \geq p \geq 0$) to an alert of difficulty level D, $\alpha \geq 1$, $\beta \geq 1$, and $b = 1/(1-L)^2$. For a fixed difficulty level, this model assigns higher probabilities of producing correct ranking to nodes with higher levels of expertise. For a node with fixed expertise level, it has a lower probability of producing correct rankings for alerts with higher D values. A node with expertise level 1 or an alert with difficulty level 0 represents the extreme case that the node can rank the alert accurately with guarantee. This is reflected in the Beta distribution by parameters $\alpha = \infty$ and $\beta = \infty$. A node with expertise level 0 or an alert with difficulty level 1 represents the extreme case that the node ranks the alert by picking up answer randomly. This is reflected in the Beta distribution by parameters $\alpha = 1$ and $\beta = 1$ (Uniform distribution). Figure 3 shows the feedback probability distribution for IDSes with different expertise levels, where the expected risk level is fixed to 0.7 and the difficulty level of test messages is 0.5.

5.2 Results for a Honest Environment

The first experiment studies the effectiveness of IDS collaboration and the importance of trust management. In this experiment, all IDSes are honest. 30 IDSes are divided into three equally-sized groups, with expertise levels of 0.1, 0.5 and 0.95 respectively. We simulate the first 100 days to observe the trust values of

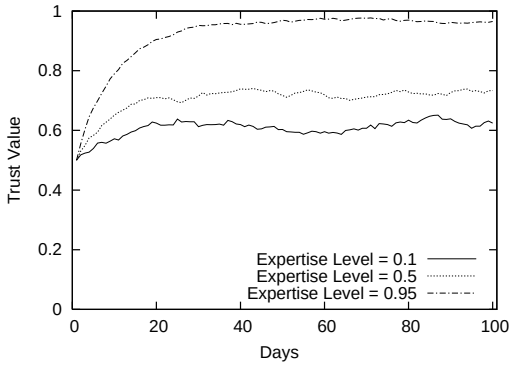

Fig. 4. Convergence of Trust Values for Different Expertise Levels during the learning period

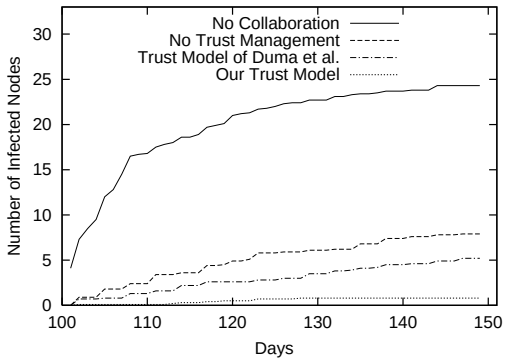

Fig. 5. Number of Infected Nodes for a Honest Environment during the attack period

nodes from each group, where nodes send only test messages to the others. Figure 4 shows the average trust values of nodes with different expertise levels. We can see that after 40-50 days, the trust values of all nodes converges to stable values.

Starting from day 101, we inject one random attack to all the nodes in the network in each day. The risk values of the attacks are uniformly generated from [low, medium, high]. The difficulty levels of attacks are fixed to 0.5. Each IDS in the network ranks the alert generated by the attack. If a node ranks "no risk" or "low risk" for a high-risk attack, then it is assumed to have been infected. We observe the total number of infected nodes in the network from day 101 to day 150 under different collaboration modes: no collaboration, collaboration without trust management, the trust management adapted from the model of Duma et al. [3] and our trust management method. The results of the total number of infected nodes are shown in Figure 5. In this figure, we can see that the network in collaboration mode is more resistant to attacks than the network in non-

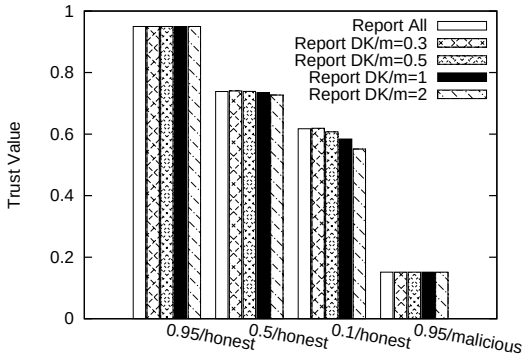

Fig. 6. Converged Trust Values for Different Expertise/Honesty Levels

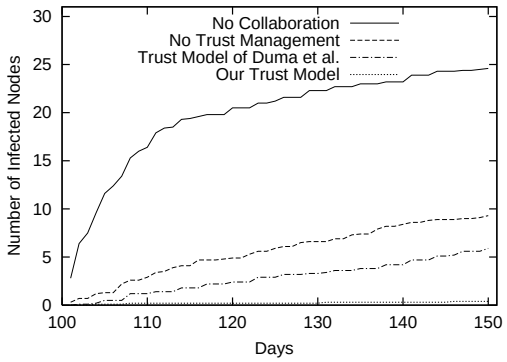

Fig. 7. Number of Infected Nodes for Dishonest Environment

collaborative mode. Trust management models further improve the effectiveness of the collaboration. Our trust model performs better than that of Duma et al. [3]. Almost all attacks are detected and almost no node is infected after 50 days.

5.3 Results for an Environment with some dishonest nodes

The purpose of the second experiment is to study the effectiveness of the collaboration model in a hazard situation where some nodes in the network are dishonest. We look at a special case where only some expert nodes are dishonest because malicious expert nodes have the largest impact on the system.

In this experiment, we have 10 expert nodes that are dishonest. There are two cases, without and with "don't know" replies. In the latter case, the percentages of "don't know" answers from nodes with expertise levels of 0.95, 0.5 and 0.1 are 0%, 4% and 45% respectively. The forgetting sensitiveness parameter (m) varies from 0.3 to 2. Figure 6 shows the converged trust values of nodes with

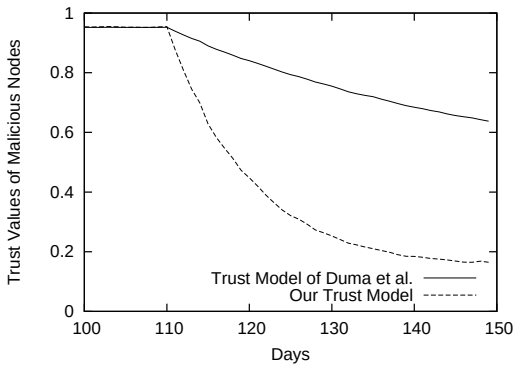

Fig. 8. Trust of Malicious Nodes

different expertise/honesty levels in "report all" case and "report don't know" case after 100 simulation days. When m is large, the punishment of reporting "don't know" is heavier. For $m = 0.3$, the nodes with medium (0.5) and low (0.1) expertise levels will be slightly rewarded. Therefore, we suggest using $m = 0.3$ to encourage non-expert nodes to report "don't know" when they are not confident about their replies.

After 100 days, we start injecting randomly generated attacks to all the nodes in the network at a rate of one attack per day. Figure 7 shows the total number of infected nodes under different collaboration modes. The number of infected nodes in the no-collaboration case is about the same as that in Figure 5 because the HIDSes make decisions independently. When there is no trust model or using the model of Duma et al. [3] to detect malicious nodes, the total number of infected nodes is larger than the corresponding case in Figure 5. The network hence suffers from malicious nodes. The number of infected nodes remains very small when using our trust model. This shows how important effective trust management is for a HIDS collaboration system.

5.4 Robustness of the Trust Model

The goal of this experiment is to study the robustness of our trust model against attacks. For the newcomer attack, malicious nodes white-wash their bad history and re-register as new users to the system. However, a newcomer attack is difficult to succeed in our system. This is because it takes a long time for a newcomer to gain trust over the trust threshold. In our experiment, it takes about 15 days for an expert node to gain trust of 0.8 to pass the threshold (as shown in Figure 4).

The second possible threat is the betrayal attack, where a malicious node gains a high trust value and then suddenly starts to act dishonestly. This scenario happens, for example, when a node is compromised. To demonstrate the robustness of our model against this attack type, we add 10 expert nodes which

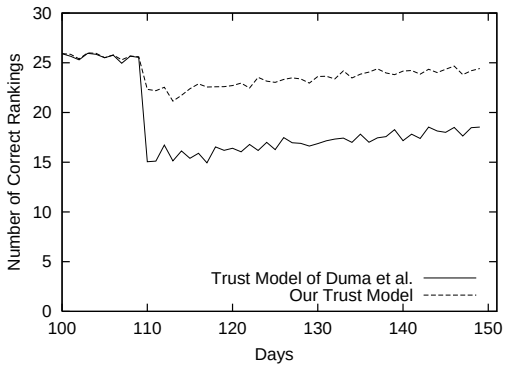

Fig. 9. The Impact of Betrayal Attack

spread opposite alert rankings on day 110. Figures 8 and 9 show the trust values of the betraying nodes and the number of correct attack rankings before and after the betrayal attack when using the trust model of Duma et al. and our trust model respectively. When using our trust model, we notice that the impact of the betrayal attack is smaller and the trust of malicious nodes drops down faster. This is because our trust model uses a forgetting factor. The trust values of dishonest nodes rely more on recent experience and therefore decrease more quickly. Our recovery phase is also shorter because we use a threshold to eliminate the impact of dishonest nodes. Once the trust values of malicious nodes drops under the trust threshold of 0.8, they are ignored in the alert consultation process and their impact is completely eliminated.

6 Related Work

Most of the existing work on distributed collaborative intrusion detection relies on the assumption that all IDSes are trusted and faithfully report intrusion events. For instance, [4] proposes a distributed information sharing scheme among trusted peers to guard against intrusions. and [6] proposes a distributed intrusion detection system based on the assumption that all peers are trusted. However, both systems can be easily compromised when some of the peers are (or become) dishonest. Duma et al. [3] address possibly malicious peer IDSes by introducing a trust-aware collaboration engine for correlating intrusion alerts. Their trust management scheme uses each past experience of a peer to predict the trustworthiness of other peers. However, their trust model is naive and does not address security issues within the collaborative network. For instance, in their system, the past experience of a peer has the same impact on its final trust values regardless of the age of its experience, therefore making it vulnerable to betrayal attacks. In our model, we use a forgetting factor when computing the trust to put more emphasis on the recent experience of the peer.

Different models have been proposed for trust management in distributed networks [5, 9]. [5] uses a global reputation management to evaluate distributed trust by aggregating votes from all peers in the network. Sun et al. [9] propose an entropy-based model and a probability-based model, which are used to calculate the indirect trust, propagation trust and multi-path trust. These models involve a lot of overhead and are not suitable for our system because IDSes can be easily compromised. They also suffer from collusion attacks.

Our model is also distinguished from the trust models developed for the application of e-marketplaces [10]. We introduce the concepts of expertise level and physical location to improve the accuracy of intrusion detection. We also allow IDSes to send test messages to establish better trust relationships with others. The alert risk ranking is categorized into multiple levels as well.

7 Conclusions and Future Work

In this paper, we presented a trust-based HIDS collaboration framework that enhances intrusion detection within a host-based IDN. The framework creates incentives for collaboration and we prove it is robust against common attacks on the collaborative network. The conducted simulations demonstrate the improved performance of our framework in detecting intrusions as well as its robustness against malicious attacks.

As future work, we will investigate the design of a communication protocol for the collaborative network, which takes privacy and efficiency issues into consideration. We will also design an automatic feedback to satisfaction level converting function, which takes risk levels of test messages, difficulty levels of test messages, and feedback from IDSes as inputs and generates satisfaction levels to the feedback as output.

Furthermore, we intend to extend our trust model to go beyond a generalized trust value for an HIDS. More specifically, since in practive HIDSes might have different expertise in detecting different types of intrusions, we would want to model the trustworthiness of a HIDS with respect to each individual type of intrusion. This will result in more effective trust management for assisting HIDSes to seek advice from truly helpful others. The subjectivity of HIDSes needs to be addressed when modeling their trustworthiness. HIDSes may have different subjective opinions on the risk levels of alerts. They can be more or less sensitive to certain intrusions.

Incentive design is another possible extension of our work. In the current protocol, the system may encounter free-rider problem such that some nodes forward the test messages to its neighbors and receive the rankings, then they forward the aggregated feedback from its neighbors to the tester. Free-riders can create unnecessary traffic in the network and degrade the inefficiency of the system. Honest users may be taken advantage and be deceived by dishonest "middle-agents". In our future work, we will investigate this problem and create corresponding incentive design to discourage free-riders and reward honest participants.

Finally, we also intend to evaluate the resistance of our framework against collusion attacks; as well as investigate its scalability in terms of number of HIDSes, rate and type of attacks.

References

1. J. Aycock. Painting the internet: A different kind of warhol worm. *Technical Report, TR2006-834-27, University of Calgary*, 2006.
2. J. Douceur. The sybil attack. *Peer-To-Peer Systems: First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002*, 2002.
3. C. Duma, M. Karresand, N. Shahmehri, and G. Caronni. A trust-aware, p2p-based overlay for intrusion detection. In *DEXA Workshops*, pages 692–697, 2006.
4. R. Janakiraman and M. Zhang. Indra: a peer-to-peer approach to network intrusion detection and prevention. *WET ICE 2003. Proceedings of the 12th IEEE International Workshops on Enabling Technologies*, pages 226–231, 2003.
5. T. Jiang and J. Baras. Trust evaluation in anarchy: A case study on autonomous networks. In *INFOCOM*. IEEE, 2006.
6. Z. Li, Y. Chen, and A. Beach. Towards scalable and robust distributed intrusion alert fusion with good load balancing. In *LSAD '06: SIGCOMM workshop on Large-scale attack defense*, pages 115–122, New York, NY, USA, 2006. ACM.
7. D. Moore, C. Shannon, and k claffy. Code-red: a case study on the spread and victims of an internet worm. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 273–284, New York, NY, 2002. ACM.
8. P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Commun. ACM*, 43(12):45–48, 2000.
9. Y. Sun, Z. Han, W. Yu, and K. Liu. A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks. In *INFOCOM*. IEEE, 2006.
10. J. Zhang and R. Cohen. Trusting advice from other buyers in e-marketplaces: the problem of unfair ratings. In *ICEC '06*, pages 225–234, New York, NY, 2006. ACM.