Dirichlet-Based Trust Management for Effective Collaborative Intrusion Detection Networks

Carol J Fung, *Member, IEEE,* Jie Zhang, *Member, IEEE,*
Issam Aib, *Member, IEEE,* and Raouf Boutaba, *Senior Member, IEEE*

***Abstract*—The accuracy of detecting intrusions within a Collaborative Intrusion Detection Network (CIDN) depends on the efficiency of collaboration between peer Intrusion Detection Systems (IDSes) as well as the security itself of the CIDN. In this paper, we propose Dirichlet-based trust management to measure the level of trust among IDSes according to their mutual experience. An acquaintance management algorithm is also proposed to allow each IDS to manage its acquaintances according to their trustworthiness. Our approach achieves strong scalability properties and is robust against common insider threats, resulting in an effective CIDN.**

We evaluate our approach based on a simulated CIDN, demonstrating its improved robustness, efficiency and scalability for collaborative intrusion detection in comparison with other existing models.

***Index Terms*—Collaborative intrusion detection system, trust management, admission control, computer security, security management.**

I. Introduction

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. A Collaborative Intrusion Detection Network (CIDN) is an IDS network intended to overcome this weakness by having each peer IDS benefit from the collective information, knowledge and experience shared by other peers. This enhances the overall accuracy of intrusion assessment as well as the ability of detecting new classes of intrusions.

Intrusions can have several forms such as worms, spamware, viruses, denial-of-service attacks (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 [1]. Another example is the Conflicker worm which infected more than 3.5 million Microsoft server systems during the year of 2008 to 2009, with the estimated economic loss of 9.1 billion dollars [2]. IDS collaboration can be an effective way to throttle or stop the spread of such contagious attacks.

The centralized collaboration of IDSes relies on a central server to gather and analyze alerts. This technique suffers from the classical performance bottleneck and single point of failure problems. The distributed collaboration of IDSes can avoid these problems. However, in such collaborative environments, a malicious (or malfunctioning) IDS can degrade the performance of others by sending false intrusion assessments. It is common in practice that a large number of nodes are compromised to form Botnets [3]. For example, the percentage of compromised nodes in 2009 is estimated to be over 10% across all Internet computers [4]. If some nodes are controlled by the same Bot-master, they can then easily collude and send false intrusion assessments. Besides, the problem of malicious IDSes is not the only concern in a distributed CIDN. IDSes may have different levels of expertise in intrusion assessments. To protect a CIDN from malicious attacks as well as find expert IDSes to consult about intrusion assessments, it is important to evaluate the trustworthiness of participating IDSes, especially when they are host-based. Since the trust model itself may also be the target of malicious attacks, robustness is a desired feature of the trust management scheme in collaborative intrusion detection networks.

In this paper, we propose a Bayesian trust management model that is robust, scalable, and suitable for distributed HIDS collaboration. More specifically, we adopt the Dirichlet family of probability density functions in our trust management for estimating the likely future behavior of a HIDS based on its past history. This theoretical model allows us to track the uncertainty in estimating the trustworthiness of the HIDS. An acquaintance management algorithm based on the estimated trust values is also proposed to enhance intrusion detection accuracy and reduce the overall overhead. The algorithm dynamically improves the list of peers a HIDS uses for collaboration and controls the frequency and type of communication between peers. Together our proposals can be used to deploy a secure and scalable CIDN where effective collaboration can be established between HIDSes.

We evaluate our approach based on a simulated collaborative HIDS network. The HIDSes are distributed and may have different expertise levels in detecting intrusions. A HIDS may also turn malicious due to runtime bugs, having been

Manuscript received April 28, 2010; revised October 15, 2010 and January 8, 2011. The associate editor coordinating the review of this paper and approving it for publication was M. Sloman.

C. J. Fung and I. Aib are with the Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada (e-mail: {j22fung, iaib}@uwaterloo.ca).

R. Boutaba is with the Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, and the Division of IT Convergence Engineering, POSTECH, Republic of South Korea (e-mail: rboutaba@uwaterloo.ca).

J. Zhang is with the School of Computer Engineering, Nanyang Technological University, Singapore (e-mail: zhangj@ntu.edu.sg).

Digital Object Identifier 10.1109/TNSM.2011.050311.100028

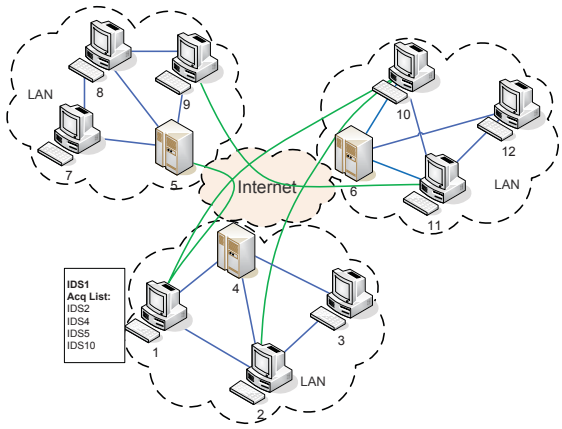


Fig. 1. The overlay design of a collaborative intrusion detection network.

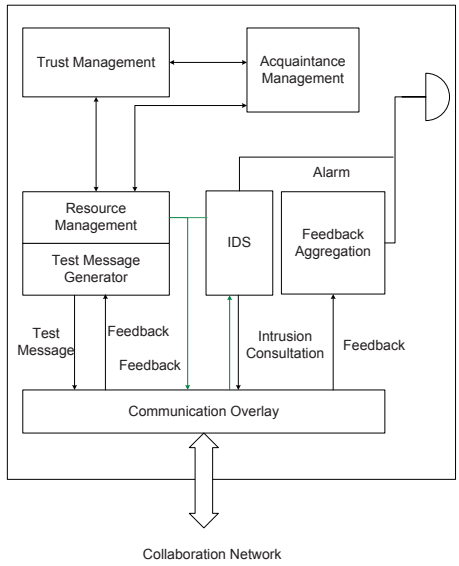


Fig. 2. CIDN architecture design.

compromised, having been updated with a faulty new configuration, or having been deliberately made malicious by its owner. We also simulate several potential threats, e.g., betrayal attacks where malicious HIDSes masquerade as honest ones to gain trust then suddenly act dishonestly. Our experimental results demonstrate that our trust management yields a significant improvement in detecting intrusions, is robust against various attacks, and improves the scalability of the system, as compared to existing collaborative HIDS systems. The proposed acquaintance management algorithm is also shown to be effective, fair, and able to create incentives for HIDSes to be honest in collaborating with others.

The CIDN framework is presented in Section II, the trust management model in Section III, and the acquaintance management algorithm in Section IV. The scalability of our approach is discussed in Section V and its robustness against common threats in Section VI. Section VII provides experimental evidence of the efficiency, robustness and scalability of our approach. Section VIII surveys related work. Section IX summarizes our contributions and outlines future work directions.

II. Collaboration Framework

Our CIDN framework connects HIDSes to form a collaborative network, in which each HIDS is free to choose whom to collaborate with. For example, it may choose peers with whom it has a sustained good experience. Peers may have different expertise levels of detecting intrusions. They may also act dishonestly or selfishly. Our framework is proposed to have several features: 1) An effective trust management model to reduce the negative impact of low expertise HIDSes, dishonest HIDSes and discover compromised ones; 2) An effective, fair and incentive algorithm for HIDSes to manage their acquaintances from which they can ask advice about intrusions; 3) Robustness against malicious insiders; 4) Scalability in terms of CIDN size, trust evaluation, and intrusion assessement.

Figure 1 shows a CIDN as an overlay network of collaborating HIDSes. Note that these HIDSes can be produced by different vendors with different intrusion detection techniques and knowledge. Links between HIDSes indicate their collaborative relationships. Each node maintains a list of *acquaintances* whom it trusts the most and collaborates with. Nodes communicate by means of intrusion evaluation requests and corresponding feedback. There are two types of requests: *intrusion consultation messages* and *test messages*. These messages are passed among HIDSes through the communication overlay as shown in the architecture design (Figure 2). The collaboration framework is built around the IDS component. The purpose of the surrounding components is to enable the communication and collaboration among different IDSes, and to archive collaborative intrusion detection. The main components of trust management and acquaintance management will be described in Sections III and IV respectively. The other components are outlined as follows.

A. Intrusion Consultation

When a HIDS detects suspicious behavior but lacks expertise to make a decision whether it should raise an alarm, it may send requests to its acquainted HIDSes for consultation. Feedback from the acquaintances is aggregated and a final decision can be made based on the aggregated results. The alert information provided to acquaintances depends on the trust level of each acquaintance. For example, a node may want to share complete alert information including data payload with other nodes inside its local network. Some part of the intrusion information might be digested or even removed when sent to "less trusted" acquaintances.

B. Test Messages

We propose that nodes in the CIDN use test messages to evaluate the trustworthiness of each other. Test messages are "bogus" consultation requests, sent out in a way that makes them difficult to be distinguished from real consultation requests. The content of test messages varies depending on the type of the intrusion. For example, a behavior graph can be used to describe activities from suspicious software. The testing node needs to know beforehand the true diagnosis result of the test message and uses the received feedback to derive a trust value for the other nodes. This technique helps in identifying inexperienced and/or malicious nodes within the collaborative network.

A test message can be generated artificially by a HIDS using a knowledge database. The database is downloaded from a central server and contains information such as the signature or behavior pattern of known malware. The latter can also be a real trace of a past intrusion that the HIDS has encountered and that has been verified by the administrator, or the latest malicious traffic collected through Honeypots [5].

C. Resource Management

The resource management policy is to decide whether a host IDS should allocate resources to respond to a consultation request. This helps in several cases, such as preventing denial of service attacks launched by sending a large number of consultation messages to a targeted HIDS. Our CIDN framework uses an incentive-compatible resource management policy [6] to assist a HIDS in allocating resources to its acquaintances. This ensures a fair treatment of HIDSes based on their past assistance. In this way, a HIDS that abusively uses the collaboration resources will be penalized by receiving fewer responses from others. The resource management policy also controls the rate of test messages in order to avoid network as well as peer overloading.

D. Communication Overlay

It is the component which handles all communication with other peers in the collaborative network. Messages passing through the communication overlay include: test messages, consultation requests to or from neighbors, and feedback from or to acquaintances. It enables IDSes from different vendors to communicate through a common protocol.

III. Trust Management Model

In this section, we propose a robust and scalable trust model which uses a Bayesian approach to evaluate the trustworthiness between each pair of HIDSes. Specifically, we use a Dirichlet family of probability density functions to estimate the likely future behavior of a HIDS based on its past history. A weighted majority method is used to aggregate feedback to make intrusion decisions.

A. Satisfaction Mapping

In our model, a HIDS sends requests to its peers and evaluates the satisfaction level of received feedback. Note that the request can be a test message or a real request. The true answer of a test message is known beforehand while that of a real request is verified by administrators after some delay through the observed impact of the corresponding alert.

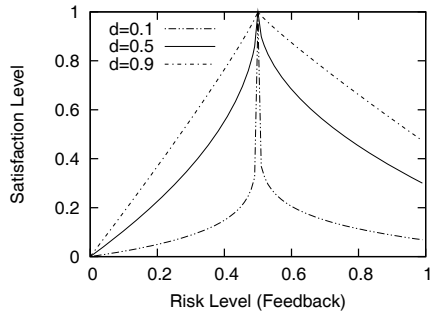


Fig. 3. Satisfaction level for feedback (r=0.5, $c_1 = 2$, $c_2 = 1$).

HIDSes may have different metrics to rank alerts. Snort for example uses three levels (low, medium, high), while Bro allows up to 100 different levels. We assume the existence of a function H, which maps a HIDS alert ranking onto the [0,1] interval where 0 denotes benign traffic and 1 highly dangerous intrusions. H preserves the "more severe than" partial order relationship. That is, if alert a_j is more severe than alert a_i then H preserves that relationship by having $H(a_j) > H(a_i)$.

The satisfaction level of feedback is determined by three factors: the expected answer ($r \in [0,1]$), the received answer ($a \in [0,1]$) and the difficulty level of the test message ($d \in [0,1]$). The larger is d the more difficult it is to correctly answer the request. Note that the difficulty of the test message can be roughly estimated by the age of the corresponding signatures or knowledge. For example, the difficulty level is low for test messages generated from old signatures; medium difficulty is for test messages generated from new signatures; high difficulty for malicious traffic taken from Honeypots and no local signature is able to detect it.

To quantitively measure the quality of feedback, we use a function $Sat(r,a,d)$ ($\in [0,1]$) to represent the level of satisfaction of the received answer based on its distance to the expected answer and the difficulty of the test message, as follows:

$$Sat(r,a,d) = \begin{cases} 1 - \left(\frac{a-r}{\max(c_1 r, 1-r)}\right)^{d/c_2} & a > r \\ 1 - \left(\frac{c_1(r-a)}{\max(c_1 r, 1-r)}\right)^{d/c_2} & a \leq r \end{cases} \quad (1)$$

where c_1 controls the extent of penalty for wrong estimates. It is set > 1 to reflect that estimates lower than the exact answer get stronger penalty than those that are higher. Parameter $c_2 \in R^+$ controls satisfaction sensitivity, with larger values reflecting more sensitivity to the distance between the correct and received answers. The equation also ensures that low difficulty level tests are more severe in their penalty to incorrect answers. The shape of the satisfaction function is depicted in Figure 3.

B. Dirichlet-based Model

Bayesian statistics provides a theoretical foundation for measuring the uncertainty in a decision that is based on a

collection of observations. We are interested in knowing the distribution of satisfaction levels of the answers from each peer HIDS and, particularly, using this information to estimate the satisfaction level of future consultations. For the case of a binary satisfaction level {satisfied,¬satisfied}, a Beta distribution can be used as appeared in [7]. For multi-valued satisfaction levels, Dirichlet distributions are more appropriate.

A Dirichlet distribution [8] is based on initial beliefs about an unknown event represented by a prior distribution. The initial beliefs combined with collected sample data can be represented by a posterior distribution. The posterior distribution well suits our trust management model since the trust is updated based on the history of interactions.

Let X be the discrete random variable denoting the satisfaction level of the feedback from a peer HIDS. X takes values in the set $\mathcal{X} = \{x_1, x_2, ..., x_k\}$ ($x_i \in [0,1]$, $x_{i+1} > x_i$) of the supported levels of satisfaction. Let $\vec{p} = \{p_1, p_2, ..., p_k\}$ ($\sum_{i=1}^{k} p_i = 1$) be the probability distribution vector of X, i.e. $P\{X = x_i\} = p_i$. Also, let $\vec{\gamma} = \{\gamma_1, \gamma_2, ..., \gamma_k\}$ denote the vector of cumulative observations and initial beliefs of X. Then we can model $\vec{p}$ using a posterior Dirichlet distribution as follows:

$$f(\vec{p}|\xi) = Dir(\vec{p}|\vec{\gamma}) = \frac{\Gamma(\sum_{i=1}^{k}\gamma_i)}{\prod_{i=1}^{k}\Gamma(\gamma_i)} \prod_{i=1}^{k} p_i^{\gamma_i - 1} \tag{2}$$

where ξ denotes the background knowledge, which in here is summarized by $\vec{\gamma}$.

Let

$$\gamma_0 = \sum_{i=1}^{k} \gamma_i \tag{3}$$

The expected value of the probability of X to be x_i given the history of observations $\vec{\gamma}$ is given by:

$$E(p_i|\vec{\gamma}) = \frac{\gamma_i}{\gamma_0} \tag{4}$$

In order to give more weight to recent observations over old ones, we embed a *forgetting factor* λ in the Dirichlet background knowledge vector $\vec{\gamma}$ as follows:

$$\vec{\gamma}^{(n)} = \sum_{i=1}^{n} \lambda^{t_i} \times \vec{S^i} + c_0 \lambda^{t_0} \vec{S^0} \tag{5}$$

where n is the number of observations; $\vec{S^0}$ is the initial beliefs vector. If no additional information is available, all outcomes have an equal probability making $S_j^0 = 1/k$ for all $j \in \{1, .., k\}$. Parameter $c_0 > 0$ is a priori constant, which puts a weight on the initial beliefs. Vector $\vec{S^i}$ denotes the satisfaction level of the i^{th} evidence, which is a tuple containing $k-1$ elements set to zero and only one element set to 1, corresponding to the selected satisfaction level for that evidence. Parameter $\lambda \in [0,1]$ is the forgetting factor. A small λ makes old observations quickly forgettable. Parameter t_i denotes the time elapsed (age) since the i^{th} evidence $\vec{S^i}$ was observed. Let $\Delta t_i = t_i - t_{i+1}$. For the purpose of scalability, the $\vec{\gamma}^{(n)}$ in Equation 5 can be rewritten in terms of $\vec{\gamma}^{(n-1)}$, $\vec{S^n}$ and Δt_n as follows:

$$\vec{\gamma}^{(n)} = \begin{cases} c_0 \vec{S^0} & n = 0 \\ \lambda^{\Delta t_n} \times \vec{\gamma}^{(n-1)} + \vec{S^n} & n > 0 \end{cases} \tag{6}$$

C. Evaluating the Trustworthiness of a Peer

After a peer receives the feedback for an alert evaluation, it assigns a satisfaction value to the feedback according to Equation 1. This satisfaction value is assigned with one of the satisfaction levels in the set $\mathcal{X} = \{x_1, x_2, ..., x_k\}$ that has the closest value. Each satisfaction level x_i also has a weight w_i.

Let p_i^{uv} denote the probability that peer v provides answers to the requests sent by peer u with satisfaction level x_i. Let $\vec{p}^{uv} = (p_i^{uv})_{i=1...k} \mid \sum_{i=1}^{k} p_i^{uv} = 1$. We model $\vec{p}^{uv}$ using Equation 2. Let Y^{uv} be the random variable denoting the weighted average of the probability of each satisfaction level in $\vec{p}^{uv}$.

$$Y^{uv} = \sum_{i=1}^{k} p_i^{uv} w_i \tag{7}$$

The *trustworthiness* of peer v as noticed by peer u is then calculated as:

$$T^{uv} = E[Y^{uv}] = \sum_{i=1}^{k} w_i E[p_i^{uv}] = \frac{1}{\gamma_0^{uv}} \sum_{i=1}^{k} w_i \gamma_i^{uv} \tag{8}$$

where γ_i^{uv} is the cumulated evidence that v has replied to u with satisfaction level x_i. The variance of Y^{uv} is equal to (superscript uv is omitted for clarity):

$$\sigma^2[Y] = \sum_{i=1}^{k}\sum_{j=1}^{k} w_i w_j cov[p_i, p_j] \tag{9}$$

Knowing that the covariance of p_i and p_j (i $\neq$ j) is given by:

$$cov(p_i, p_j) = \frac{-\gamma_i \gamma_j}{\gamma_0^2(\gamma_0 + 1)} \tag{10}$$

We get:

$$\begin{aligned} \sigma^2[Y] &= \sum_{i=1}^{k} w_i^2 \sigma^2[p_i] + 2\sum_{i=1}^{k}\sum_{j=i+1}^{k} w_i w_j cov[p_i, p_j] \\ &= \sum_{i=1}^{k} w_i^2 \frac{\gamma_i(\gamma_0 - \gamma_i)}{\gamma_0^2(\gamma_0+1)} + 2\sum_{i=1}^{k}\sum_{j=i+1}^{k} w_i w_j \frac{-\gamma_i\gamma_j}{\gamma_0^2(\gamma_0+1)} \\ &= \frac{1}{\gamma_0^3 + \gamma_0^2} \sum_{i=1}^{k} w_i \gamma_i \left(w_i(\gamma_0 - \gamma_i) - 2\sum_{j=i+1}^{k} w_j \gamma_j \right) \end{aligned} \tag{11}$$

Let $C^{uv} \in (-1, 1]$ be the confidence level for the value of T^{uv}, and we describe it as:

$$C^{uv} = 1 - 4\,\sigma[Y^{uv}] \tag{12}$$

where $4\,\sigma[Y^{uv}]$ is roughly the 95% confidence interval.

Lemma 3.1: The confidence level C^{uv} formulated by Equation 12 lies in bound (-1, 1].

Proof:

From Equation 12 and Equation 11, we have,

$$C^{uv} = 1 - \frac{4}{\sqrt{1+\gamma_0}} \sqrt{\sum_{i=1}^{k} w_i^2 \frac{\gamma_i}{\gamma_0} - (\sum_{i=1}^{k} w_i \frac{\gamma_i}{\gamma_0})^2} \tag{13}$$

where $w_i \in [0,1], \forall i$ is the weight of the satisfaction level i, and $\gamma_0 = \sum_{i=1}^{k} \gamma_i > 0$. To prove the boundary of C^{uv}, we construct a discrete random variable $Z \in \{w_1, w_2, ..., w_k\}$, where $w_1 \leq w_2 \leq ... \leq w_k$ and $\mathbb{P}[Z = w_i] = \frac{\gamma_i}{\gamma_0}, \forall i$. Then we have,

$$\sigma^2[Z] = \mathbb{E}(Z^2) - \mathbb{E}^2(Z) = \sum_{i=1}^{k} w_i^2 \frac{\gamma_i}{\gamma_0} - (\sum_{i=1}^{k} w_i \frac{\gamma_i}{\gamma_0})^2 \quad (14)$$

We can see that the variation of Z is the major component of C^{uv}. It is not difficult to see that $\sigma^2[Z]$ reaches its maximum when $\mathbb{P}[Z = w_1] = \mathbb{P}[Z = w_k] = 0.5$ and $\mathbb{P}[Z = w_j] = 0, \forall j (1 < j < k)$. Therefore, we have $0 \leq \sigma^2[Z] \leq \frac{1}{4}$. After replacing Equation 14 back into Equation 13, we have $-1 < C^{uv} \leq 1$. ∎

D. Feedback Aggregation

Based on their estimated trustworthiness, each peer requests alert consultation only from those peers in its acquaintance list whose trust values are greater than a threshold. We introduce the 95% confidence interval lower-bound T_l and upper bound T_h as follows:

$$T_l^{uv} = E[Y^{uv}] - 2\sigma[Y^{uv}] = T^{uv} + C^{uv}/2 - 1/2 \quad (15)$$
$$T_h^{uv} = E[Y^{uv}] + 2\sigma[Y^{uv}] = T^{uv} - C^{uv}/2 + 1/2 \quad (16)$$

Note that for a node with a low confidence value, the lower-bound trust T_l is low compared to its expected trust and the upper-bound T_h is high. In the rest of this paper, we use T_l and T_h when a confident judgment is needed.

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

$$\overline{a}_i^u = \frac{\sum_{T_l^{uv} \geq th^u, v \in A^u} T^{uv} a_i^{uv}}{\sum_{T_l^{uv} \geq th^u, v \in A^u} T^{uv}}, \quad (17)$$

where $\overline{a}_i^u$ is the aggregated ranking of alert i from the feedback provided by each peer belonging to the acquaintance list A^u of peer u maintained using our acquaintance management model in the next section; th^u is the trust threshold set by u; $a_i^{uv} \in [0,1]$ is the feedback ranking of alert i from v to u.

IV. Acquaintance Management

In this section, we propose an algorithm for a HIDS in the CIDN to maintain a list of acquaintances from which it can ask for consultation of intrusions. The HIDS sends test or consultation messages to its acquaintances and updates the trust values based on the satisfaction level it gets from their feedback (see Section III for detailed computation). It is not recommended for a node to keep all the other nodes in its acquaintance list because the communication overhead to maintain a long acquaintance list makes it not scalable. We propose that each HIDS maintains an acquaintance list with adapted length (the number of acquaintances in the list) based on its resource capacity. Each node communicates with its acquaintances regularly to keep their trust values up to date. Less trusted acquaintances will be replaced by new nodes with higher trust values. It also responds to requests from its acquaintances to maintain its own trustworthiness.

Since it takes some time to learn the trust value of a new node, we propose that each HIDS also maintains a *probation list*, where a new node can stay in probation for some period of time before it becomes an acquaintance. A node also communicates with the nodes in its probation list periodically to learn their trust values. The purpose of the probation list is to explore potential collaborators and keep introducing new trustworthy nodes to the acquaintance list.

Suppose that node i has a list of n_i acquaintances, and let S_a^i and S_p^i be the acquaintance list and the probation list of the node. The corresponding trust values for the acquaintances are $T^i = \{T_1^i, T_2^i, ..., T_{n_i}^i\}$, and the confidence of estimated trust is $C^i = \{C_1^i, C_2^i, ..., C_{n^i}^i\}$. Let l_a^i and l_p^i be the length of the acquaintance list and probation list, respectively. Let l_{max}^i be the maximum number of nodes in the acquaintance and probation lists for HIDS i. l_{max}^i is determined by the resource capacity of the HIDS. Let q be the parameter that controls the length of the probation list compared to l_{max}^i, and we have $l_p^i \leq q l_{max}^i$. If we choose q too small, the updating of the acquaintance list may be too slow and less adaptive to the changes of nodes' behavior. On the other hand, if the probation list is too large, a lot of resources are wasted on testing new nodes. l_p^i is chosen by $l_p^i = max(q l_{max}^i, l_{max}^i - l_a^i)$.

The acquaintance management procedure for each node is shown in Algorithm 1. For convenience, we drop super-script i. The acquaintance list S_a is initially empty and the probation list S_p is filled by l_{max} random nodes. An acquaintance list updating event is triggered every t_u time units. S_a is updated by including new trusted nodes from S_p. A node that stays at least $t_p > t_u$ time units in probation is called a *mature node*. Some nodes are discarded early if their upper-bound trust values formulated by Equation 16 drop below a low threshold (lines 13-17). Only mature nodes are allowed to join the acquaintance list (lines 19-24). The purpose of this requirement is to give time for an IDS to learn its potential collaborators and set obstacles for newcomer attack (see section VI-2). Then, S_p is refilled with new randomly chosen nodes (lines 27-30). After that, the nodes whose lower-bound trust values formulated by Equation 15 are the smallest are removed from S_a if the total number of nodes $|S_a| + |S_p|$ is higher than l_{max} (lines 32-35).

We use an upper and lower bound to protect new nodes in the probation list and penalize those which do not respond to consultations frequently enough. For example, if two nodes in the probation list have the same bad trust values then the one with higher confidence level will be removed first. Similarly, if two nodes in the acquaintance list have the same low trust values, then the node having lower confidence will be removed first. We choose to remove the bad nodes entirely and repopulate probation list so that the system keeps learning about new nodes and tries to find better collaborators over time.

Several properties are desirable for the acquaintance management algorithm, including convergence, fairness, and incentive for collaboration. When our acquaintance management

Algorithm 1 Managing Acquaintance & Probation Lists

Initialization :
$S_a \Leftarrow \emptyset$ //Acquaintance list.
$l_p = max(l_{max} - |S_a|, ql_{max})$ //Probation list.
//Fill S_p with randomly selected nodes
while $|S_p| < l_p$ **do**
  $e \Leftarrow$ select random node
  $S_p \Leftarrow S_p \cup e$
end while
set new timer event(t_u, "**SpUpdate**")
Periodic Maintenance:
at timer event ev of type "**SpUpdate**" **do**
//Remove from probation list nodes with bad record
for all $e \in S_p$ **do**
  if $T(e) - C(e)/2 + 1/2 < th_p$ **then**
    $S_p \Leftarrow S_p \setminus e$
  end if
end for
//Merge mature nodes into the acquaintance list.
for all e in S_p **do**
  if $t_e > t_p$ //$t_e =$ age of node e in probation list **then**
    $S_p \Leftarrow S_p \setminus e$
    $S_a \Leftarrow S_a \cup e$
  end if
end for
$l_p = max(l_{max} - |S_a|, ql_{max})$
//Refill S_p with randomly selected nodes
while $|S_p| < l_p$ **do**
  $e \Leftarrow$ random node not in S_a or S_p
  $S_p \Leftarrow S_p \cup e$
end while
//Remove least trusted nodes from acquaintance list till $|S_a| + |S_p| \leq l_{max}$
while $|S_a| + |S_p| > l_{max}$ **do**
  $a \Leftarrow$ select $e \in S_a$ where $T(e)$+$C(e)/2$ is minimum
  $S_a \Leftarrow S_a \setminus a$
end while
set new timer event(t_u, "**SpUpdate**")
end timer event

is in place, we expect the acquaintance list of each node to converge to a stable state. This is important since cooperation between HIDSes is generally long-term in nature. Frequently changing collaborators is costly because HIDSes need to spend considerable amount of time to learn the trustworthiness of new collaborators. Fairness and incentive for collaboration are also important properties, and are the foundation for forming sustainable long-term collaborative relationships. Section VII-I evaluates our acquaintance management algorithm, to demonstrate that our algorithm achieves these three properties.

V. Test Message Exchange Rate and Scalability of Our System

Each HIDS u in our system maintains an acquaintance list and a probation list with maximum length l^u_{max}. This length can be fixed according to the resource capacity of node u or slightly updated with the changes in CIDN size.

TABLE I
Acquaintance Categorization

Peer category	Criterion	Rate
Highly Trustworthy	$0 < th \leq T_l$	R_l
Trustworthy	$T_l < th \leq T$	R_h
Untrustworthy	$T < th \leq T_h$	R_m
Highly Untrustworthy	$T_h < th \leq 1$	R_l

However, it is always set to a value small enough to account for scalability. Equation 6 ensures that the process of updating the trustworthiness of a peer after the reception of a response is performed with only three operations, making it linear with respect to the number of answers.

There is a trade-off to be resolved in order to account for scalability in the number of messages exchanged in the CIDN. On one hand, the forgetting factor in Equation 6 decays the importance given to existing highly trusted peers. This implies that their corresponding test message rates need to be above a certain minimal value. On the other hand, sending too many requests to other peers may compromise scalability. To solve this issue, we adapt the rate of test messages to a given peer according to its estimated trustworthiness. The adaptation policy is provided in Table I, where acquaintances are categorized into highly trustworthy, trustworthy, untrustworthy, and highly untrustworthy. There are three levels of test message rates: $R_l < R_m < R_h$. We can see in Table I that the test message rate to highly trustworthy or highly untrustworthy peers is low. This is because we are confident about our decision of including or not their feedback into the aggregation. A higher test message rate is assigned to trustworthy or untrustworthy peers because their trust values are close to the threshold and hence need to be kept under close surveillance.

Each peer in the system needs to actively respond to others' requests in order to keep up its trustworthiness and be able to receive prompt help when needed. However, actively responding to every other peer may cause bandwidth and/or CPU overloading. Therefore, as a consultant to others, a peer would like to limit the rate of answers it provides. In this regard, each peer in our system would respond to requests with a priority proportional to the amount of trust it places on the source of the request [6]. It will give higher priority to highly trusted friends. This obeys the social norm: "Be nice to others who are nice to you", and also provides incentives for encouraging peers to act honestly in order to receive prompt help in times of need.

VI. Robustness against Common Threats

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

1) Sybil attacks: occur when a malicious peer in the system creates a large amount of pseudonyms (fake identities) [9]. Such a malicious peer uses fake identities to gain larger influence over the false alert ranking on others in the network. Our defense against sybil attacks can rely on the design of authentication mechanism as well as the acquaintance management system. Authentication makes registering fake identities

difficult. Our model can use a certificate issuing authority which only allows one identity per (user, machine) tuple. In addition, our trust management model requires HIDSes to first build up their trust before they can affect the decision of others, which is costly to do with many fake identities. This way, our security and trust mechanisms protect our collaborative network from sybil attacks.

2) Newcomer attacks: occur when a malicious peer can easily register as a new user [10]. Such a malicious peer creates a new ID for the purpose of erasing its bad history with other peers in the network and create immediate damages. Our model handles this type of attacks by assigning low trust values to all newcomers and enforcing the probation period for each new node. In this way, their feedback on the alerts is simply not considered by other peers during the aggregation process. Newcomers may gain more trust over time and eventually move to acquaintance list of they behave consistently well.

3) Betrayal attacks: occur when a trusted peer suddenly turns into a malicious one and starts sending false feedbacks. 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 peer acts dishonestly, the forgetting factor (Equation 6) causes its trust value to drop down quickly, hence making it difficult for this peer to deceive others or gain back its previous trust within a short time.

4) Collusion attacks: happen when a group of malicious peers cooperate together by providing false alert rankings in order to compromise the network. In our system, peers will not be adversely affected by collusion attacks. In our trust model each peer relies on its own knowledge to detect dishonest peers. In addition, we use test messages to uncover malicious peers. Since the test messages are sent in a random manner, it will be difficult for malicious peers to distinguish them from actual requests.

5) Inconsistency attacks: happen when a malicious peer repeatedly changes its behavior from honest to dishonest in order to degrade the efficiency of the CIDN. Inconsistency attacks are harder to succeed in the Dirichlet-based model because of the use of the forgetting factor and the dynamic test message rate, which makes trust values easy to lose and hard to gain. This ensures that the trust values of peers with inconsistent behavior remain low and hence have little impact.

VII. Simulations and Experimental Results

In this section, we present a set of experiments used to evaluate the efficiency, scalability and robustness of our trust management model in comparison with existing ones [11][12]. Experiments are also carried out to demonstrate the desirable properties of our acquaintance management algorithm. Each experimental result presented in this section is derived from the average of a large number of replications with an overall negligible confidence interval.

TABLE II
Simulations Parameters

Parameter	Value	Description
R_l	2/day	Low test message rate
R_m	10/day	Medium test message rate
R_h	20/day	High test message rate
λ	0.9	Forgetting factor
th	0.8	Trust threshold for aggregation
c_0	10	Priori Constant
c_1	1.5	Cost rate of low estimate to high estimate
c_2	1	Satisfaction sensitivity factor
s	4	Size of grid region
k	10	Number of satisfaction levels

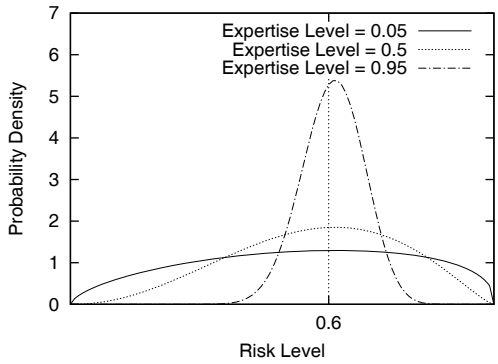


Fig. 4. Decision density function for expertise levels.

A. Simulation Setting

We simulate a CIDN environment with n HIDS peers randomly distributed over an $s \times s$ grid region. The proximity distance is given by the minimum number of square steps between each two peers. The expertise level of a peer can be low (0.05), medium (0.5) or high (0.95). In the beginning, each peer receives an initial acquaintance list containing neighbour nodes based on proximity. The initial trust value of every peer in the acquaintance list is 0.5. To test the trustworthiness of acquaintances, each peer sends out test messages following a Poisson process with rates according to Table I. The parameters we used are shown in Table II.

B. Modeling the Expertise Level of a Peer

To reflect the expertise level of each peer, we use a Beta distribution to simulate the decision model of answering requests. A Beta density function is given by:

$$\begin{aligned} f(p|\alpha,\beta) &= \frac{1}{B(\alpha,\beta)} p^{\alpha-1}(1-p)^{\beta-1} \\ B(\alpha,\beta) &= \int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt \end{aligned} \tag{18}$$

where $f(p|\alpha,\beta)$ is the probability that a peer with expertise level l answers with a value of $p \in [0,1]$ to an alert of difficulty level $d \in [0,1]$. Higher values for d are associated to attacks that are difficult to detect, i.e. many peers fail to identify them. Higher values of l imply a higher probability of producing correct alert rankings.

Let r be the expected ranking of an alert. We define α and β as follows:

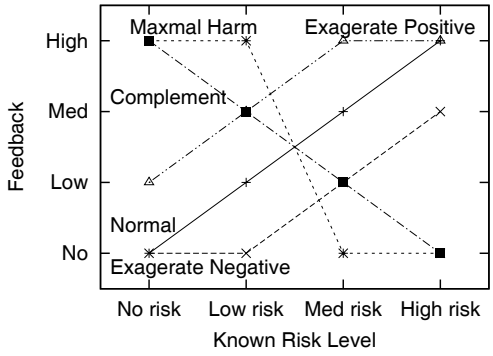


Fig. 5. Feedback curves for different deception strategies.

$$\begin{aligned} \alpha &= 1+\frac{l(1-d)}{d(1-l)}\sqrt{\frac{r}{1-r}}\sqrt{\frac{2}{l}-1} \\ \beta &= 1+\frac{l(1-d)}{d(1-l)}\sqrt{\frac{1-r}{r}}\sqrt{\frac{2}{l}-1} \end{aligned} \tag{19}$$

For a fixed difficulty level, the above model has the property of assigning higher probabilities of producing correct rankings to peers with higher levels of expertise. A peer with expertise level l has a lower probability of producing correct rankings for alerts of higher difficulty ($d > l$). $l = 1$ or $d = 0$ represent the extreme cases where the peer can always accurately rank the alert. This is reflected in the Beta distribution by $\alpha, \beta \to \infty$. Figure 4 shows the feedback probability distribution for peers with different expertise levels, where we fix the expected risk level to 0.6 and the difficulty level of test messages to 0.5.

C. Deception Models

A dishonest peer may adopt one of the four deception models: *complementary*, *exaggerate positive*, *exaggerate negative*, and *maximal harm*. The first three deception models are described in [13], where an adversary may choose to send feedback about the risk level of an alert that is respectively opposite to, higher, or lower than the true risk level. We propose a maximal harm model where an adversary always chooses to report false feedback with the intention to bring the most negative impact to the request sender. Figure 5 shows the feedback curve for the different deception strategies. For instance, when a deceptive peer using the maximal harm strategy receives a ranking request and detects that the risk level of the request is "medium", it sends feedback "no risk" because this feedback can maximally deviate the aggregated result at the sender side.

D. Trust Values and Confidence Levels for Honest Peers

The first experiment studies the effectiveness of the collaboration and the importance of our trust management. In this experiment, all peers are honest. We simulate the scenario where each peer u has a fixed size N^u of its acquaintance list. The peers are divided into three equally-sized groups of *low*, *medium* and *high* expertise levels respectively. The first phase of the simulation is a learning period (50 days), during which peers learn about each other's expertise levels by sending out test messages. Figure 6 shows the resulting average trust values of the 30 acquaintances of peer u. The trust values converge after 30 days of simulation and the actual expertise levels of the peers are able to be effectively identified by our trust model.

To study the impact of different test message rates on the confidence level of trust estimation (Equation 12), we conduct a second experiment to let u use a fixed test message rate in every simulation round. The rate of sending test messages starts with one message per day and increases by five for every simulation round. We plot the confidence level of trust evaluation for each test message rate in Figure 7. We can observe that the confidence level increases with the increase of the test message rate. This confirms our argument that sending more test messages improves the confidence of trust estimation. We also observe that the confidence levels increase with the expertise levels. This is because peers with higher expertise levels tend to perform more consistently.

E. Trust Values for Dishonest Peers

The purpose of this experiment is to study the impact of dishonest peers using the four different deception strategies described in Section VII-C. To study the maximum impact of these deception strategies, we only use peers with a *high* expertise level as deceptive adversaries since they are more likely to know the true answers and can perform the deception strategies more accurately.

In this experiment, we let peer u have an acquaintance list of 40 dishonest peers divided into four groups. Each group uses one of the four deception models: complimentary, exaggerate positive, exaggerate negative, and maximal harm. We use a dynamic test message rate and observe the convergence curve of the average trust value for each group of deceptive peers. Results are plotted in Figure 8.

We notice that the trust values of all adversary peers converge to stable values after 30 days of the learning phase. It is not surprising that adversary peers using the maximal harm strategy have the lowest trust values, while adversary peers using the complimentary strategy have the second lowest ones. The converged trust values of adversary peers using exaggerate positives are higher than those using exaggerate negatives. This is because we use an asymmetric penalization mechanism for inaccurate replies ($c_1 > 1$ in Equation 1). We penalize more heavily peers that untruthfully report lower risks than those which untruthfully report higher risks.

F. Robustness of Our Trust Model

The goal of this experiment is to study the robustness of our trust model against various insider attacks. For the newcomer attack, malicious peers white-wash their bad history and re-register as new users to the system. If the trust value of a newcomer can increase quickly based on its short term good behavior, the system is then vulnerable to newcomer attacks. However, a newcomer attack is difficult to succeed in our model. In our model, we use parameter c_0 in Equation 6 to control the trust value increasing rate. When c_0 is larger, it takes longer for a newcomer to gain a trust value above the trust threshold.

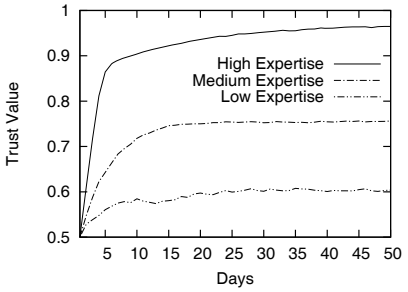


Fig. 6. Convergence of trust values for different expertise levels.

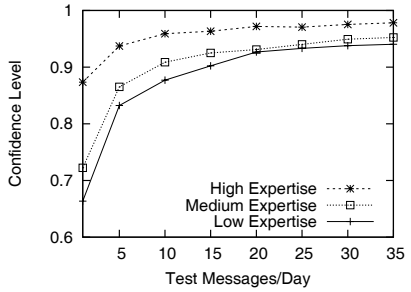


Fig. 7. Confidence levels of estimation for different test message rates.

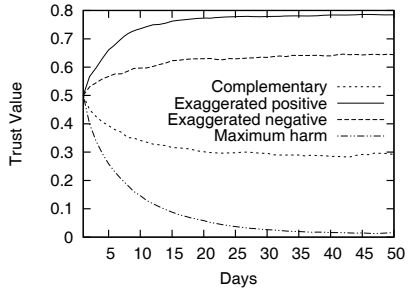


Fig. 8. Trust values of deceptive peers with different deception strategies.

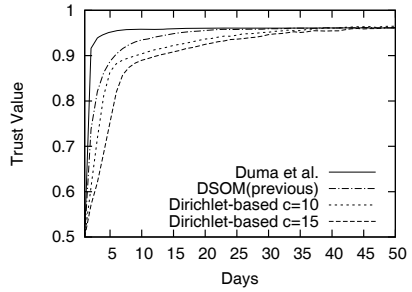


Fig. 9. Trust values of newcomers under different trust models.

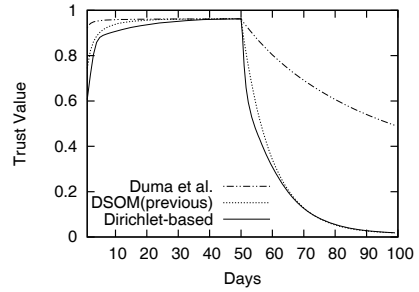


Fig. 10. Trust of malicious peers under betrayal attack.

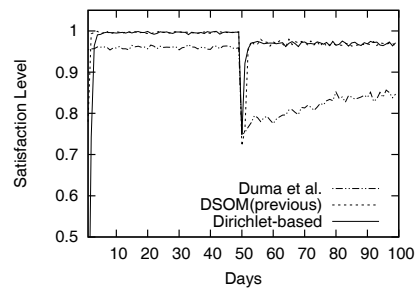


Fig. 11. Impact on accuracy of betrayal attack.

We compare our Dirichlet-based model with our previous model [11] and the model of Duma et al. [12] in Figure 9. We observe that in the Duma et al. model, the trust values of new users increase very fast and reach the aggregation trust threshold (0.8) in the first day, which reveals a high vulnerability to newcomer attacks. The reason for this is that their model does not have an initial trust to new peers and therefore their trust values change fast in the beginning. In the model we developed in [11], the trust values increase in a slower manner and reach the trust threshold after three days. However, that model is not flexible in that it does not offer control over the trust increase speed. In the Dirichlet-based model, the trust increase speed is controlled by the priori constant c_0. For $c_0 = 10$, it takes a newcomer four to five days of consistent good behavior to reach the same trust value. Larger values of c_0 make it even slower to reach high trust, hence offering robustness against newcomer attacks.

The second possible threat is the betrayal attack, where a malicious peer first gain a high trust value and then suddenly starts to act dishonestly. This scenario can happen, for example, when a peer is compromised. To demonstrate the robustness of our model against this attack type, we set up a scenario where u has seven peers in its acquaintance list, of which six are honest with an expertise level evenly divided between low, medium, and high. The malicious one has high expertise and behaves honestly in the first 50 days. After that, it launches a betrayal attack by adopting a maximal harm deceptive strategy. We observe the trust value of the betraying peer and the satisfaction levels of aggregated feedback in each day with respect to u.

Figure 10 shows the trust value of the betraying peer before and after the launching of the betrayal attack when respectively using Duma et al., our previous and our current trust models. For the Duma et al. model, the trust value of the malicious peer slowly drops after the betrayal attack. This is because their model does not use a forgetting factor, hence providing the previous honest behavior of a malicious peer with a heavy impact on the trust calculation for a considerable amount of time. The trust value of the betraying peer drops much faster using our previous model, while the fastest rate is observed when using our Dirichlet-based model. This is because both models use a forgetting factor to pay more attention to the more recent behavior of peers.

We also notice that the Dirichlet-based model has a slight improvement over our previous model. The Dirichlet-based model adopts the dynamic test message rate and can react more swiftly. The rate of sending messages to malicious peers increases as soon as they start behaving dishonestly. Higher rates of test messages help in faster detection of dishonest behavior. However, in our previous model, the test message rate remains the same. This phenomenon can be further observed in Figure 12.

The results for the satisfaction levels of aggregated feedback with respect to u before and after the betrayal attack are shown in Figure 11. We notice that the satisfaction level of u for the aggregated feedback drops down drastically in the first day following the learning period and recovers after that in all three models. The recovery period is however much shorter for the Dirichlet-based and our previous models. This is again attributed to the use of the forgetting factor. The Dirichlet-based model has a slight improvement in the recovering speed over our previous model. This is because in the Dirichlet-based model, the trust values of betraying peers drop under the aggregation threshold faster than our previous model.

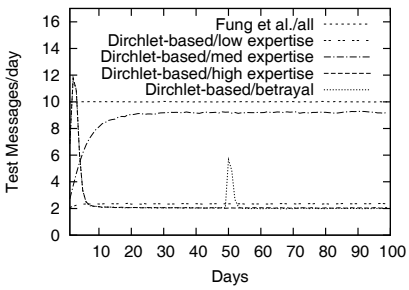


Fig. 12. Comparison of average test message rates under different models.

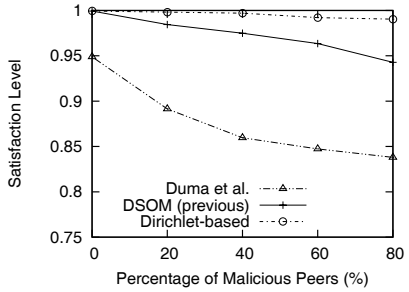


Fig. 13. Aggregated feedback under inconsistency attack.

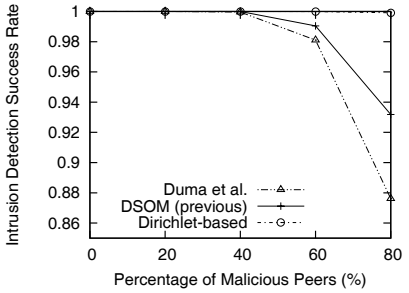


Fig. 14. Intrusion detection success rate under inconsistency attack.

Therefore, the impact of betraying peers is eliminated earlier than that in the previous model.

G. Scalability of our Trust Model

The result of test message rates under betrayal attack is shown in Figure 12. We notice that in our Dirichlet-based model, the average test message rates for highly trustworthy as well as highly untrustworthy peers are the lowest. The average test message sending rate to peers with the medium expertise level is higher but still below the medium rate (R_m). Compared to our previous model, the average message sending rate is much lower, which demonstrates the improved scalability of our Dirichlet-based model. Note that the spike from the betraying group on around day 50 is caused by the drastic increment of the test message rate. The sudden change of a highly trusted peer behavior will cause the trust confidence level to drop down quickly. The rate of sending messages to this peer then switches to R_h accordingly.

H. Efficiency of our Trust Model

To demonstrate the efficiency of our Dirichlet-based trust model, we conduct another experiment to evaluate the intrusion detection accuracy. In this experiment, we let peer u have 15 acquaintances, which are evenly divided into low, medium, and high expertise groups. Among the expert peers, some are malicious and launch inconsistency attacks synchronously to degrade the efficiency of the CIDN. More specifically, in each round of behavior changing, these malicious peers adopt the maximal harm deception strategy for two days followed by six days of honest behavior.

In Figure 13, we vary the percentages of malicious peers from 0% to 80%. We inject daily intrusions to peer u with medium difficulty (0.5) and random risk levels. We then plot the average satisfaction level for the aggregated feedback. We observe that our Dirichlet-based model outperforms the others. This is because the dynamic test message rate in Dirichlet-based model causes the trust of malicious peers to drop faster and increase more slowly, hence minimizing the impact of dishonest behavior. Among the three models, Duma et al. has the least satisfaction level because of its slow response to sudden changes in peer behavior and its aggregation of all feedback from even untrustworthy peers.

Figure 14 shows the success rate of peer u in detecting intrusions. We notice that both our previous model and the Duma et al. model cannot effectively detect intrusions when the majority of peers are malicious. Our Dirichlet-based model shows excellent efficiency in intrusion detection even in the situation of a dishonest majority.

I. Properties of our Acquaintance Management Algorithm

1) Convergence: In this experiment, we study the convergence of our proposed acquaintance management algorithm. First, we simulate a network of 30 HIDSes divided into 10 equally-sized groups with expertise levels of $0.1, 0.2, ..., 1.0$, respectively. We set the maximum acquaintance/probation list length $l_{max} = 8$ and probation list ratio $p = 0.25$. The updating interval is 5 days and the probation period is 30 days. Figure 15 plots the average expertise levels of the acquaintances for nodes with different expertise levels. We observe that in the first three rounds, the average expertise levels of acquaintances for all nodes are close (0.55). After that, they diverge and converge to stable values depending on the expertise levels of the host nodes.

We also count the number N of nodes that include a given node into their acquaintance lists. We then compute the average N for each group of nodes with similar expertise levels as shown in Figure 16. We can see that nodes with higher expertise levels are more likely to stay in the acquaintance list of others nodes and hence have a larger number of collaborators.

2) Fairness: In this experiment, we create a random CIDN containing 100 nodes with random expertise levels uniformly distributed in $[0, 1]$. All nodes are honest. We observe the converged average expertise level of nodes in the acquaintance list for all nodes in the network. The result is shown in Figure 17. Nodes with higher expertise levels end up having collaborators whose expertise levels are close by. This reflects a good fairness in the collaboration.

3) Incentive for Collaboration: We add another 50 nodes which are dishonest and have random expertise levels uniformly distributed between $[0, 1]$. We observe the distribution of the acquaintances for all nodes after convergence. Figure 18 shows the distribution of acquaintances' expertise levels for all nodes in the network, where a negative expertise represents a dishonest node. Notice that honest nodes end up having few dishonest nodes in their acquaintance lists, while dishonest nodes tend to collaborate with other dishonest nodes. This reflects the incentive for collaboration induced by our

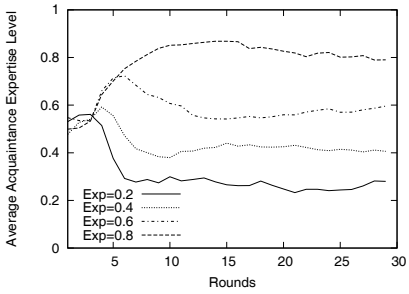


Fig. 15. Convergence of the acquaintance list.

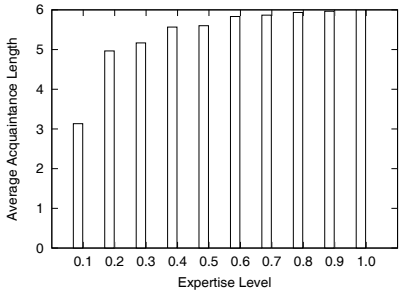


Fig. 16. The average acquaintance length.

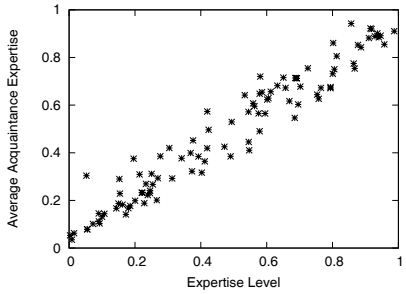


Fig. 17. Expertise levels of acquaintances.

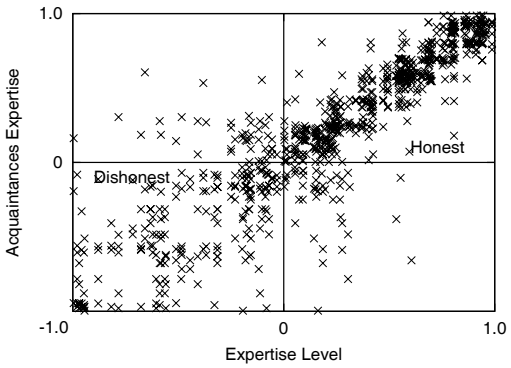


Fig. 18. Acquaintances' expertise levels for a CIDN with dishonest nodes.

acquaintance management algorithm. Nodes are encouraged to behave honestly no matter what their expertise levels are. A dishonest node ends up not receiving help from honest ones.

VIII. Related Work

CIDNs can be divided into information-based CIDNs and experience-based CIDNs. In information-based CIDNs, intrusion information such as intrusion alerts, intrusion traffic samples, firewall logs, are shared in the network and aggregated to achieve better network-wide intrusion decisions. Many information-based CIDNs, such as [14]-[15], have been proposed in the past few years. Information-based CIDNs are especially effective in detecting epidemic worms or attacks, and most of them require homogeneous participant IDSes. While in experience-based CIDNs, suspicious data samples are sent to collaborators for diagnosis. Feedback from collaborators are then aggregated to help the sender IDS make intrusion decisions. Examples of such CIDNs include [11], [12], [16], and [17]. Experience-based CIDNs may involve heterogeneous IDSes and are effective in detecting many intrusion types including worms, malware, port scannings, and buffer-overflows.

From the perspective of collaboration topology, many CIDNs are centralized, such as [14], [18], and [19]. A centralized CIDN requires a consistently on-site central server and it suffers from the single point of failure problem. A decentralized CIDN such as [15] can alleviate the workload of the central server by means of clustering where the cluster heads partially process data they collect. In a fully distributed CIDN [11], [16], [20], [21], all IDSs are equally responsible for collecting/processing information and therefore is the most scalable and flexible topology.

Most of the existing work on distributed collaborative intrusion detection relies on the assumption that all HIDSes are trustworthy and faithfully report intrusion events. The Indra system [22] distributes among peers information about attack attempts on different machines so as to proactively react and increase the chance of detecting an attack. This system also allows peer neighbors to share information about intrusion attempts in order to enhance the overall system security. Another example is the distributed intrusion alert fusion system called Cyber Disease Distributed Hash Table (CDDHT) [23]. The CDDHT system provides several load balancing schemes to evenly distribute intrusion alarms among the sensor fusion centers in order to increase the scalability, fault-tolerance and robustness of the system. However, the systems mentioned above are all vulnerable to malicious IDS attacks. False information about intrusion events sent by malicious IDSes may heavily degrade the performance of these CIDNs.

To protect a CIDN, it is important to evaluate the trustworthiness of participating IDSes. ABDIAS [24] is a community based CIDN where IDSes are organized into groups and exchange intrusion information to gain better intrusion detection accuracy. A simple majority-based voting system was proposed to detect compromised nodes. However, such system is vulnerable to colluded voting. Duma et al. [12] propose to address possibly malicious IDSes (peers) by introducing a trust-aware collaboration engine for correlating intrusion alerts. Their trust management scheme uses each peer's past experience to predict others' trustworthiness. However, their trust model is simplistic and does not address security issues within the collaborative network. For instance, in their system, the peer's past experience has the same impact on the final trust values of others, and therefore is vulnerable to betrayal attacks where compromised peers suddenly change their behavior. In our model, we use a forgetting factor when calculating trust, in order to rely more on the peer's recent experience and be robust to the changes of other peers' behavior. Our previous work [11] proposed a robust trust management model that uses test messages to gain personal experience and a forgetting factor to emphasize most recent experiences. However, this model needs to repeatedly aggregate all past experience with a peer when updating its trust, which makes it not scalable over time. It uses a linear model to calculate the average

satisfaction levels of past interactions, and lacks a theoretical foundation. Also this approach does not capture trust modeling uncertainties or provide statistical confidence information on intrusion decisions. Our new model uses Dirichlet distributions to model peer trustworthiness. It makes use of dynamic test message rates in order to allow for better scalability. Also, our new model further improves robustness over our previous one through the use of flexible test message rates.

Researchers in multi-agent systems have also been developing trust models to evaluate the trustworthiness of buying and selling agents in e-marketplaces [7]. One of the earliest trust models developed by Marsh [25] computes the trustworthiness of selling agents by taking into account direct interactions between buying and selling agents. The trust-oriented learning strategy proposed by Tran and Cohen [26] uses reinforcement learning to determine the trustworthiness of selling agents, after the true value of delivered goods is evaluated and compared to the buying agent's expected value for the goods. Selling agents can be classified as untrustworthy if their trust values fall below a certain threshold and buying agents try to select the trustworthy selling agent with the highest expected value for the goods. The Beta Reputation System (BRS) of Whitby et al. [27] and the TRAVOS model of Teacy et al. [28] estimate the trustworthiness of a selling agent by employing a Beta probability density function representing a probability distribution of a continuous variable. The work of Zhang and Cohen [7] focuses on coping with inaccurate reputation information about selling agents shared by malicious buying agents in e-marketplaces. The REGRET model of Sabater et al. [29] offers a multi-dimensional view of trust that includes a social dimension taking into consideration the social relationships among agents. However, it is difficult to clearly determine social relationships among HIDSes in CIDNs.

Our model is different from the above trust models in several aspects. First, our model is focused on long-term collaboration trust. Repetitive direct interactions between two agents are common in CIDN environment. Second, the cost of experience in CIDN is much lower than in e-commerce and it allows HIDSes to send test messages to better establish trust relationships with others. Third, our model uses fine-grained experience quality rather than a binary measurement such as "good" or "bad". Instead, it is categorized into multiple levels. Finally, our model uses direct trust modeling rather than reputation models. It is because the latter may suffer from collusion attacks where a group of malicious IDSes cooperate together by providing false reputation information about some IDSes to bad-mouth these targets for example.

Different reputation models were proposed in distributed systems [30], [31]. These reputation models allow peers to get advice when evaluating the trustworthiness of other peers. For example, [30] uses a global reputation management to evaluate distributed trust by aggregating votes from all peers in the network. Sun et al. [31] propose for the communication in distributed networks an entropy-based model and a probability-based one. The models are used to calculate indirect trust, propagation trust and multi-path trust. They however involve a lot of overhead which limits their scalability. Another important concern is that IDSes can be easily compromised and become deceptive when reporting the trustworthiness of others. The reputation models for peer-to-peer networks, such as PowerTrust [32], TrustGuard [33], Malicious detector [34], and Fine-Grained reputation [35] are capable of detecting malicious peers. However, they are purposed to detect deceiving nodes in a P2P network and can not be directly used in IDNs to improve the intrusion detection accuracy. A trust model in CIDN should not only detect malicious nodes but also improve the intrusion detection accuracy as well as contain good levels of robustness and scalability.

IX. Conclusions and Future Work

In this paper, we presented a trust management solution for evaluating trustworthiness of Host-based Intrusion Detection Systems in a Collaborative Intrusion Detection Network. Our trust management adapts Dirichlet density functions as its theoretical foundation, and is accordingly able to measure the uncertainty in estimating the likely future behavior of HIDSes. The measured uncertainty allows our trust management to employ an adaptive message exchange rate, resulting in good scalability. Equipped with a forgetting factor, it is also robust against some common threats. The effectiveness, robustness and scalability of our trust management have been further demonstrated through experiments carried out in a simulated Collaborative Intrusion Detection Network. HIDSes in the conducted experiments have different levels of expertise in detecting intrusions and adopt different deception strategies. The results show that our trust management is more effective compared to previous trust models. Our acquaintance management algorithm is also demonstrated to have the properties of fairness and convergence, and to provide incentive for collaboration. Putting it all together, our work serves as an important step toward effective trust management necessary for the deployment of a secure CIDN capable to enhance networks and systems security.

One possible direction for future work is to incorporate reputation models in our trust management. In this case, the important issues of inaccurate reputation information, scalability and collusion attacks should be addressed.

For future work, we also plan to deploy a real CIDN using existing intrusion detection systems, and use it for further experiments.

Acknowledgment

This work was supported in part by the Natural Science and Engineering Council of Canada (NSERC) under its Strategic program, and in part by the WCU (World Class University) program through the Korea Science and Engineering Foundation funded by the Ministry of Education, Science and Technology (Project No. R31-2008-000-10100-0).

References

[1] D. Moore, C. Shannon, and k claffy, "Code-red: a case study on the spread and victims of an Internet worm," in *Proc. 2nd ACM SIGCOMM Workshop Internet Measurment*, 2002.

[2] "ZDnet." Available: http://www.zdnet.com/blog/security/conficker-estimated-economic-cost-91-billion/3207 [Last accessed Oct. 15, 2010].

[3] R. Vogt, J. Aycock, and M. Jacobson, "Army of botnets," in *Netw. Distributed Syst. Security Symp.*, 2007.

[4] M. Ahamad, D. Amster, M. Barrett, T. Cross, G. Heron, D. Jackson, J. King, W. Lee, R. Naraine, G. Ollmann *et al.*, "Emerging cyber threats report for 2009," 2008.

[5] D. Dagon, X. Qin, G. Gu, W. Lee, J. Grizzard, J. Levine, and H. Owen, "Honeystat: local worm detection using honeypots," *Recent Advances in Intrusion Detection*. Springer, 2004, pp. 39-58.

[6] Q. Zhu, C. Fung, R. Boutaba, and T. Başar, "A game-theoretical approach to incentive design in collaborative intrusion detection networks," in *Proc. International Symp. Game Theory Netw.*, May 2009.

[7] J. Zhang and R. Cohen, "Trusting advice from other buyers in e-marketplaces: the problem of unfair ratings," in *Proc. 8th International Conf. Electron. Commerce: The New e-commerce: Innovations for Conquering Current Barriers, Obstacles and Limitations to Conducting Successful Business on the Internet*, 2006.

[8] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd edition. Prentice Hall, 2002.

[9] J. Douceur, "The sybil attack," in *Proc. 1st International Workshop Peer-to-Peer Syst.*, 2002.

[10] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, "Reputation systems," *Commun. ACM*, vol. 43, no. 12, pp. 45-48, 2000.

[11] C. Fung, O. Baysal, J. Zhang, I. Aib, and R. Boutaba, "Trust management for host-based collaborative intrusion detection," in *19th IFIP/IEEE International Workshop Distributed Syst.*, 2008.

[12] C. Duma, M. Karresand, N. Shahmehri, and G. Caronni, "A trust-aware, p2p-based overlay for intrusion detection," in *International Conf. Database Expert Syst. Appl.*, 2006.

[13] B. Yu and M. Singh, "Detecting deception in reputation management," in *Proc. Second International Joint Conf. Autonomous Agents Multiagent Syst.*, 2003.

[14] "SANS Internet Storm Center (ISC)" Available: http://isc.sans.org/.

[15] V. Yegneswaran, P. Barford, and S. Jha, "Global intrusion detection in the domino overlay system," in *Proc. Netw. Distributed Syst. Security Symp.*, 2004.

[16] C. Fung, J. Zhang, I. Aib, and R. Boutaba, "Robust and scalable trust management for collaborative intrusion detection," in *Proc. Eleventh IFIP/IEEE International Symp. Integrated Netw. Management*, 2009.

[17] J. Oberheide, E. Cooke, and F. Jahanian, "Cloudav: N-version antivirus in the network cloud," in *Proc. 17th USENIX Security Symp.*, 2008.

[18] F. Cuppens and A. Miege, "Alert correlation in a cooperative intrusion detection framework," in *Proc. IEEE Symp. Security Privacy*, 2002.

[19] J. Ullrich, "DShield." Available: http://www.dshield.org/indexd.html.

[20] M. Cai, K. Hwang, Y. Kwok, S. Song, and Y. Chen, "Collaborative Internet worm containment," *IEEE Security Privacy*, vol. 3, no. 3, pp. 25-33, 2005.

[21] Z. Zhong, L. Ramaswamy, and K. Li, "ALPACAS: a large-scale privacy-aware collaborative anti-spam system," in *Proc. IEEE INFOCOM*, 2008.

[22] R. Janakiraman and M. Zhang, "Indra: a peer-to-peer approach to network intrusion detection and prevention," in *Proc. 12th IEEE International Workshops Enabling Technol.*, 2003.

[23] Z. Li, Y. Chen, and A. Beach, "Towards scalable and robust distributed intrusion alert fusion with good load balancing," in *Proc. 2006 SIGCOMM Workshop Large-Scale Attack Defense*, 2006.

[24] A. Ghosh and S. Sen, "Agent-based distributed intrusion alert system," in *Proc. 6th International Workshop Distributed Comput*, 2004.

[25] S. Marsh, "Formalising trust as a computational concept," Ph.D. thesis, Department of Mathematics and Computer Science, University of Stirling, 1994.

[26] T. Tran and R. Cohen, "Improving user satisfaction in agent-based electronic marketplaces by reputation modeling and adjustable product quality," in *Proc. Third International Joint Conf. Autonomous Agents Multiagent Syst.*, 2004.

[27] A. Whitby, A. Jøsang, and J. Indulska, "Filtering out unfair ratings in Bayesian reputation systems," *Icfain J. Management Research*, pp. 48-64, Feb. 2005.

[28] W. T. L. Teacy, J. Patel, N. R. Jennings, and M. Luck, "Coping with inaccurate reputation sources: experimental analysis of a probabilistic trust model," in *Proc. Fourth International Autonomous Agents Multiagent Syst.*, 2005.

[29] J. Sabater and C. Sierra, "Regret: a reputation model for gregarious societies," in *Proc. Fifth International Conf. Autonomous Agents Workshop Deception, Fraud Trust Agent Societies*, 2001.

[30] T. Jiang and J. Baras, "Trust evaluation in anarchy: a case study on autonomous networks," in *IEEE INFOCOM*, 2006.

[31] Y. Sun, Z. Han, W. Yu, and K. Liu, "A trust evaluation framework in distributed networks: vulnerability analysis and defense against attacks," in *IEEE INFOCOM*, 2006.

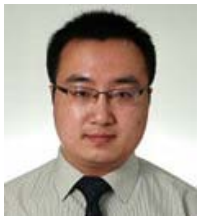
[32] A. Rahbar and O. Yang, "Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Trans. Parallel Distributed Syst.*, vol. 18, no. 4, pp. 460-473, 2007.

[33] M. Srivatsa, L. Xiong, and L. Liu, "TrustGuard: countering vulnerabilities in reputation management for decentralized overlay networks," in *Proc. 14th International Conf. World Wide Web*, 2005.

[34] L. Mekouar, Y. Iraqi, and R. Boutaba, "Peer-to-peer's most wanted: malicious peers," *Comput. Netw.*, vol. 50, no. 4, pp. 545-562, 2006.

[35] Y. Zhang and Y. Fang, "A fine-grained reputation system for reliable service selection in peer-to-peer networks," *IEEE Trans. Parallel Distributed Syst.*, pp. 1134-1145, 2007.

Carol Fung is a Ph.D. student in University of Waterloo (Canada). She received her BSc and Msc from University of Manitoba, Canada. Her research topic is collaborative Intrusion Detection networks, which includes trust management, collaborative decision, resource management, and incentive design of such a system. She is also interested in the security problems in wireless networks and social networks.



Jie Zhang joined Nanyang Technological University, Singapore as an assistant professor, working on trust modeling and incentive mechanism design. He got Ph.D. in the School of Computer Science at the University of Waterloo, and was the recipient of the Alumni Gold Medal, awarded to honour the top PhD graduate.

Issam Aib is currently a research fellow in the School of Computer Science at the University of Waterloo (Canada). He received Ph.D. from the University of Pierre et Marie Curie. His research interests include policy-based management, security policies, intrusion detection systems, and management of WLANs.

Raouf Boutaba received the MSc and Ph.D. degrees in computer science from the University Pierre & Marie Curie, Paris, in 1990 and 1994, respectively. He is currently a professor of computer science at the University of Waterloo. His research interests include network, resource and service management in wired, and wireless networks. He has received several best paper awards and other recognitions such as the Premier's Research Excellence Award, the IEEE Hal Sobol Award in 2007, the Fred W. Ellersick Prize in 2008, the Joe LociCero Award and the Dan Stokesbury in 2009.