Practical Bayesian Poisoning Attacks on Challenge-Based Collaborative Intrusion Detection Networks

Meng, Weizhi; Li, Wenjuan; Jiang, Lijun; Choo, Kim-Kwang Raymond; Su, Chunhua

Published in:
European Symposium on Research in Computer Security

Link to article, DOI:
10.1007/978-3-030-29959-0_24

Publication date:
2019

Document Version
Peer reviewed version

Practical Bayesian Poisoning Attacks on Challenge-Based Collaborative Intrusion Detection Networks

Weizhi Meng[1(✉)], Wenjuan Li[1,2], Lijun Jiang[3], Kim-Kwang Raymond Choo[4], and Chunhua Su[5]

[1] Department of Applied Mathematics and Computer Science, Technical University of Denmark, Lyngby, Denmark
weme@dtu.dk
[2] Department of Computer Science, City University of Hong Kong, Kowloon Tong, Hong Kong
[3] Cyber Tree Research Center, Pokfulam, Hong Kong
[4] Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, USA
[5] Division of Computer Science, University of Aizu, Aizuwakamatsu, Japan

Abstract. As adversarial techniques constantly evolve to circumvent existing security measures, an isolated, stand-alone intrusion detection system (IDS) is unlikely to be efficient or effective. Hence, there has been a trend towards developing collaborative intrusion detection networks (CIDNs), where IDS nodes collaborate and communicate with each other. Such a distributed ecosystem can achieve improved detection accuracy, particularly for detecting emerging threats in a timely fashion (before the threat becomes common knowledge). However, there are inherent limitations due to malicious insiders who can seek to compromise and poison the ecosystem. A potential mitigation strategy is to introduce a challenge-based trust mechanism, in order to identify and penalize misbehaving nodes by evaluating the satisfaction between challenges and responses. While this mechanism has been shown to be robust against common insider attacks, it may still be vulnerable to advanced insider attacks in a real-world deployment. Therefore, in this paper, we develop a collusion attack, hereafter referred to as *Bayesian Poisoning Attack*, which enables a malicious node to model received messages and to craft a malicious response to those messages whose aggregated appearance probability of normal requests is above the defined threshold. In the evaluation, we explore the attack performance under both simulated and real network environments. Experimental results demonstrate that the malicious nodes under our attack can successfully craft and send untruthful feedback while maintaining their trust values.

Keywords: Intrusion detection · Collaborative network · Insider threat · Bayesian Poisoning Attack · Challenge-based trust mechanism

K. Sako et al. (Eds.): ESORICS 2019, LNCS 11735, pp. 493–511, 2019.
https://doi.org/10.1007/978-3-030-29959-0_24

1 Introduction

Intrusion detection/prevention systems (IDSs/IPSs; collectively referred to as IDSs in this paper) are widely deployed in computing networks, with the purpose of identifying and isolating intrusion attempts [11,26]. Traditionally, an IDS can be classified as either *network-based (NIDS)* or *host-based (HIDS)* [29]. As the importance of cyber security is increasingly recognized by both organizations and governments, so does the sophistication of cyber attackers. For example, an isolated IDS in any organization would easily be bypassed by zero-day attacks, since they (IDS and the organization) are not able to learn from ongoing attack campaigns faced by their peers or other industry sectors, either in the same jurisdiction or any part of the world. Thus, this maximizes the impact of a cyber attack in the sense that the same exploit or vulnerability can affect tens to hundreds or thousands of IDSs and organizations. However, if we are able to learn from an ongoing attack that is faced by organization X in country Y, then the entire ecosystem would be better prepared against attackers making use of the same exploit or vulnerability.

This gives rise to collaborative intrusion detection networks (CIDNs), so that IDS nodes can collaborate and communicate with each other [5,35]. Due to its distributed architecture, insider attacks are a key threat to the ecosystem [3]. For example, in a *collusion attack*, two or more malicious nodes can collude to provide untruthful information of alarm ranking and reduce the effectiveness of alarm aggregation. Thus, we need to establish some form of robust trust mechanisms to safeguard CIDNs against insider attacks.

In the literature, challenge-based trust mechanisms (shortly *challenge mechanisms*) are a promising solution to defend CIDNs against insider attacks, by identifying malicious nodes through evaluating the satisfaction between challenges and responses [8]. More specifically, a *challenge* can contain a set of alarms asking for the severity level, and can be sent to evaluate the trustworthiness of the suspected/tested nodes. Under this mechanism, the testing node knows the severity of the alarms; thus, it can utilize the received responses to derive a trust value (e.g., satisfaction level) for the target node. Studies, such as those in [5–7], have demonstrated that the challenge-based trust mechanism can mitigate common insider attacks like collusion attacks and betrayal attacks.

However, challenge mechanisms reply on two assumptions, namely: (*assumption A1*) challenges are sent out in a way that makes it toilsome for anyone to distinguish the challenges from normal messages; and (*assumption A2*) malicious nodes always send feedback contrary to its truthful judgment. In practice, however, malicious nodes may act more dynamically and have a complex behavior [8]. For example, malicious nodes may act faithfully most of the times and only untruthfully on some occasions (e.g., targeting specific events or systems). Therefore, existing challenge mechanisms may not be able to mitigate advanced insider attacks. For instance, Li et al. [16] developed the *passive message fingerprint attack (PMFA)* to distinguish challenges from normal requests; thus, circumventing the challenge-based trust mechanism. However, this attack can be mitigated by controlling the timing of sending normal requests.

Motivation and Contributions. Given the potential of challenge mechanisms to protect CIDNs against a range of attacks, including common insider attacks, we posit the importance of enhancing the robustness of such mechanisms against advanced attacks that are practical in nature. Focused on this issue, in this work, we develop an advanced collusion attack, coined *Bayesian Poisoning Attack*. In this attack, a malicious node can model the received messages and successfully send a untruthful response to messages that have a higher probability of being a normal request; thus, circumventing *assumption A1* of the challenge-based trust mechanism. Specifically, building on *PMFA*, we develop the *Bayesian Poisoning Attack*, where two or more malicious nodes can collude to collect messages and give untruthful answers to a normal request, without adversely affecting their trust values. Hopefully, the findings of this work will simulate further interest in designing more robust challenge-based CIDNs to deal with advanced insider attacks, as well as other practical attacks. In the end, we also discuss some countermeasures to defend our attack.

In the next section, we will revisit challenge-based CIDNs, including briefly introducing their key building blocks. In Sect. 3, we analyze the assumptions used in the existing challenge mechanisms and describe our *Bayesian poisoning attack*. In Sect. 4, we describe our evaluation setup and explain our findings under both simulated and real CIDN environments. Specifically, we show that our attack can help a malicious entity identify an appropriate timing for giving untruthful feedback, and it is effective to compromise the challenge-based trust mechanism in practical deployment. Related literature is reviewed in Sect. 5, and the last section concludes our work.

2 Challenge-Based CIDNs

Intuitively, challenge-based CIDNs employ the challenge-based trust mechanisms to defend against insider attacks. Figure 1 depicts the high-level architecture of a common challenge-based CIDN and its key building blocks. This architecture can be applied to network structure, such as wireless sensor networks (WSNs) and Internet of Things (IoT).

Network Interactions. In the architecture, each IDS node can choose its partners or collaborators, based on its own policies and experience. These nodes can be associated if they have a collaborative relationship (e.g., vendor, and organizations/entities within the same system). Each node can maintain a list of their collaborated nodes, called *partner list* (or *acquaintance list*). Such list is customizable and stores information of other nodes (e.g., public keys and their current trust values). Before a node can join the network, it has to register with a trusted certificate authority (CA) and obtain its unique proof of identity (e.g., a key pair with a public key and a private key). As shown in Fig. 1(a), if node C wishes to join the network, it needs to send an application to a network node, say node A. Then, node A makes a decision and sends back an initial *partner list*, if node C is accepted.

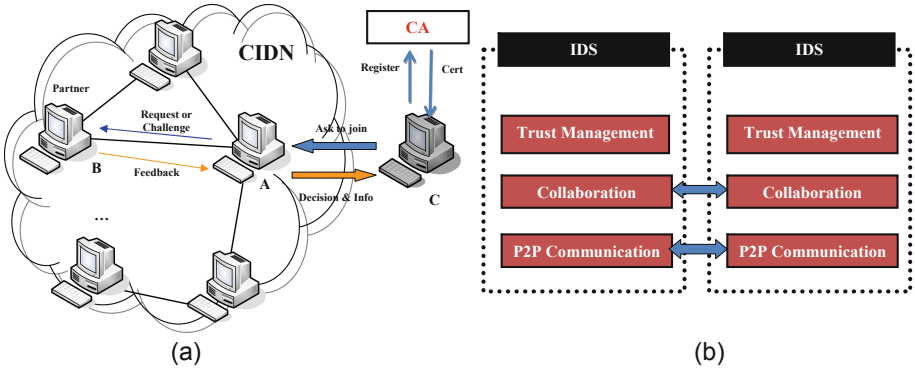

Fig. 1. (a) High-level architecture of a common challenge-based CIDN and (b) key building blocks.

CIDNs allow IDS nodes to exchange the necessary and required messages in-between to improve the performance. There are two major types of interactive messages, namely: challenges and normal requests.

- *Challenges.* A challenge contains a set of IDS alarms asking to label their severity. A testing node can send a challenge to other tested nodes and obtain the relevant feedback. As the testing node knows the severity of the sent alarms, it can use the received feedback to derive a trust value (e.g., satisfaction level) for the tested node.
- *Normal requests.* A normal request is sent by a node for alarm aggregation. Other IDS nodes should send back alarm ranking information as their feedback. Alarm aggregation is an important feature for CIDNs, which can help improve the detection performance, and it usually considers the feedback from trusted nodes.

Network Components. Figure 1(b) shows the key building blocks in a CIDN node, including *trust management component*, *collaboration component* and *P2P communication*.

- *Trust management component.* This component is responsible for evaluating the trustworthiness of other nodes. Under the challenge mechanism, the trustworthiness of other nodes is mainly computed by evaluating the received feedback. Each node can send out either normal requests or challenges for alert ranking (consultation). To protect challenges, it is worth noting that challenges should be sent out in a random manner and in a way that makes them difficult to be distinguished from a normal alarm ranking request.
- *Collaboration component.* This component is mainly responsible for assisting a node to evaluate the trustworthiness of other nodes by sending out *normal requests* and/or *challenges*, and upon receiving the relevant *feedback* to evaluate its truthfulness. As shown in Fig. 1, if node A sends a *request/challenge* to node B, then node B will send back the relevant feedback.

– *P2P communication.* This component is responsible for connecting with other IDS nodes and providing network organization, management and communication among IDS nodes.

Robustness. It has been shown that challenge-based trust mechanisms can enhance the CIDN's resilience in mitigating common insider attacks, such as Sybil, newcomer, betrayal and collusion attacks [5–8].

– *Sybil attack.* This attack occurs when a malicious node creates a large number of fake identities [2], with the aim of having an unfair influence on the alert aggregation. As shown in Fig. 1, an IDS node should register with a *CA* and obtain a unique proof identity; thus, mitigating such an attack. Clearly, if the *CA* is corrupted, then this attack will work. However, *CA* has a vested interest to ensure that they are not compromised or known to have laxed security practices, as this will affect their bottomline.
– *Newcomer (re-entry) attack.* This attack occurs when a malicious node registers as a new user, in order to erase its bad history. Challenge-based CIDNs begin by giving low initial trust values to all newcomers, so that the influence of new nodes on alarm aggregation is minimal. This is somewhat analogous to the credit history system, where one's creditworthiness is built over time (e.g., based on the factors like payment history and age of credit history).
– *Betrayal attack.* This attack occurs when a trusted node becomes malicious. To defeat such an attack, a high trust value should only be established after a lengthy interaction and consistently good behavior (again, similar to the credit history system), and only a few bad actions will ruin the trust value (in the context of the credit history system, bad activities like derogatory marks due to payment default, or hard credit inquiries). In particular, it employs a forgetting factor to give more credits to recent behaviors.
– *Collusion attack.* This attack happens when a group of malicious peers collude to provide false alarm rankings in order to compromise the network. Challenge-based trust mechanisms can uncover malicious peers via sending the challenges, where the trust values of malicious nodes can decrease rapidly if their untruthful feedback is detected.

3 Our Proposed Attack

In this section, we discuss the underlying assumptions (or threat model) made by challenge-based trust mechanisms, and describe our attack.

3.1 Threat Model and Assumption Analysis

As previously discussed, challenge-based mechanisms can be effective in defending against most common insider attacks, based on the following two assumptions.

– **A1.** Challenges are sent out in a random way, which is challenging to be distinguished from normal messages.
– **A2.** Malicious nodes always send feedback contrary to its true assessment (i.e., misreporting a malicious event as benign, and vice versa).

These two assumptions are key to protecting *challenges* and identifying malicious nodes. In particular, the first assumption implies two conditions: *a random manner* and *hard to distinguish*. These ensure that an IDS node cannot distinguish a challenge from normal requests. Thus, malicious nodes have a trivial possibility of identifying challenges, and have to respond to each message.

The second assumption implies a *maximal harm model*, where an adversary always chooses to report untruthful feedback with the intention to bring the most negative impact to the request sender [7]. As an example, whenever a malicious node receives a ranking request, it will reply with a 'no risk' for an alarm whose real risk level could be 'medium', because this feedback can maximize the impact at the sender side.

Are These Two Assumptions Realistic/Practical? These assumptions are reasonable in scenarios, where attackers (or *naive attackers*) choose a *maximal harm model*. In practical implementations, however, attackers can choose to go under the radar in order to avoid detection. For example, why would an attacker risk been identified as malicious by 'lying' all the times? Would it not make more sense to 'lie' only on events of importance (e.g., some sort of 'sleeper' node)? In other words, advanced attackers, including advanced persistent threat (APT) attackers, would likely behave normally/truthfully most of the time (referred to as 'advanced attack' where attackers can perform complex operations, unlike naive attacks in the *maximal harm model*, in this paper).

Thus, the existing challenge-based trust mechanisms that rely on the two assumptions will be insecure against such advanced attackers. This is the premise of our proposed attacks, to be described next.

3.2 Bayesian Poisoning Attacks

An example of an insider attack not captured in existing CIDN systems is the *passive message fingerprint attack (PMFA)* of Li et al. [16]. In such an attack, the attacker is able to distinguish normal requests from messages, based on the observation that a generic CIDN would send normal requests to trusted nodes at the same time in practice. In other words, if several nodes receive the same message (containing the same alarm set), then this message is very likely to be a normal request (and not a challenge).

However, this attack can be easily mitigated through controlling the timing of sending normal requests (i.e., sending the next request after getting a response from the last request). In this case, *PMFA* can be largely mitigated. In this work, we develop the *Bayesian Poisoning Attack*, in which malicious nodes can send untruthful feedback to those messages who have a high probability of being a normal request.

Main idea. The challenge mechanism can be compromised if malicious nodes can only send untruthful feedback to normal requests, but send truthful feedback to challenges. As a result, the key idea is to find an appropriate timing to deliver malicious feedback. Based on this key idea, our developed *Bayesian Poisoning Attack* aims to passively collect messages and send untruthful feedback in a

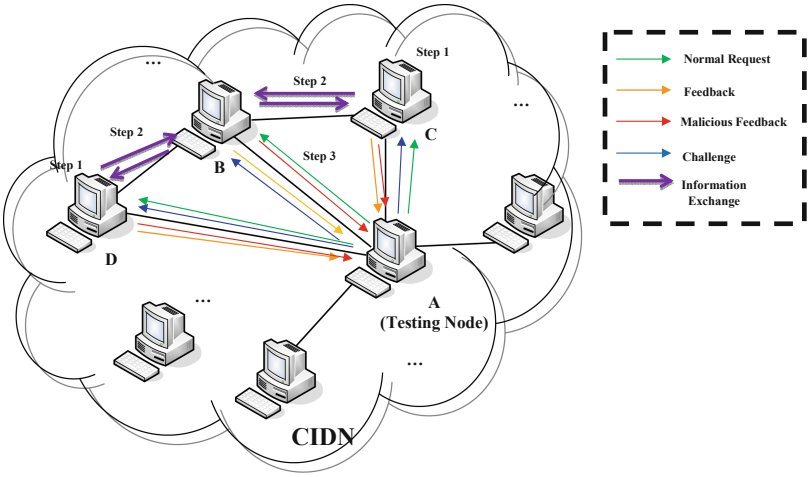

Fig. 2. Steps of Bayesian poisoning attacks on challenge-based CIDNs.

probability that is computed by the Bayesian inference model. That is, our attack can decide whether an incoming message has a high probability of being a normal request. Subsequently, suspicious nodes are able to send untruthful feedback only to normal requests, while providing truthful answers to other received messages.

Figure 2 provides an example to illustrate how such attack operates in practice. Suppose a testing node A delivers either requests or challenges to its partners. Under the mechanism, all tested nodes should provide feedback after receiving the messages. Assume nodes B, C and D to be suspicious/malicious, we explain our attack with detailed steps as follows.

- **Step 1.** At this stage, every suspicious node starts collecting and recording all received messages from the testing node. In this attack, we accept the first assumption that challenges are sent out in a random way, which is challenging to be distinguished from normal messages. Thus, malicious nodes have to passively collect data at this stage.
- **Step 2.** At this stage, suspicious nodes can collaborate with each other to exchange recorded messages. In practice, in order to rank alarms, normal requests have to be delivered to all trusted nodes. This opens a chance to distinguish normal request from messages [16]. Taking node B as an example, it can compare the recorded messages from nodes C and D. A message could be a normal request with a high probability, if a match is identified.
- **Step 3.** Based on the number of identified normal requests and the number of received messages, our attack builds a model and computes the probability of normal requests. By given a threshold, suspicious nodes can return untruthful feedback to the messages with a high probability of being a normal request. For other messages, malicious nodes can still return truthful answers.

Bayesian Inference Model. This is a statistical method of inference, by using the Bayes' rule to predict the probability for a hypothesis as additional evidence [33]. To compute the appearance probability of a normal request, suppose there are N messages received from the testing node, among which k messages are normal requests. Assume a Binomial distribution controls the probability of observing $n(N) = k$. The equation is shown below.

$$P(n(N) = k|p) = \binom{N}{k} p^k (1-p)^{N-k} \tag{1}$$

where $n(N)$ describes how many normal requests are received, and p describes how likely a message to be a normal request. Binomial distribution describes a distribution where there are two mutually exclusive outcomes to an event. It helps identify a sequence of n trials where each has the same probability of p. The ultimate goal of our model is to predict the possibility: $P(V_{N+1} = 1|n(N) = k)$; that is, measuring how likely the $(N+1)^{th}$ message can be a normal request. According to the Bayesian theorem, we can have the following:

$$P(V_{N+1} = 1|n(N) = k) = \frac{P(V_{N+1} = 1, n(N) = k)}{P(n(N) = k)} \tag{2}$$

where $P(V_{N+1} = 1|n(N) = k)$ describes how likely the $(N+1)^{th}$ message is a normal request, if we receive N messages which contain k normal requests. We further apply a marginal probability distribution[1] and can have the followings:

$$P(n(N) = k) = \int_0^1 P(n(N) = k|p) f(p) \cdot dp \tag{3}$$

$$P(V_{N+1} = 1, n(N) = k) = \int_0^1 P(n(N) = k|p) f(p) p \cdot dp \tag{4}$$

To estimate the prior information regarding $p \in [0, 1]$, it is reasonable to assume that it is decided by a uniform prior distribution $f(p) = 1$. According to Eqs. (2) to (4), we can obtain the following equation, which can describe the appearance possibility of a normal request, P_{req}, within a time period.

$$\begin{aligned} P_{req} = P(V_{N+1} = 1|n(N) = k) &= \frac{\int_0^1 P(n(N) = k|p) f(p) p \cdot dp}{\int_0^1 P(n(N) = k|p) f(p) \cdot dp} \\ &= \frac{k+1}{N+2} \end{aligned} \tag{5}$$

Bayesian Modeling of Normal Request Distribution. In a real-world deployment, the number of challenges is often pre-defined whereas the number of normal requests is dynamic based on the network traffic. The challenge mechanism assumes that the challenges are randomly sent, but a real-world environment can only achieve pseudo-randomness. To emphasize the identification

[1] Marginal distribution describes the possibility of various values of the variables in the subset, without considering the values of the other variables.

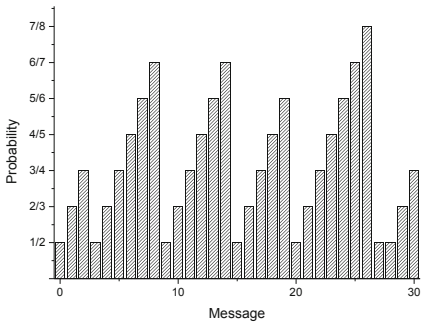

Fig. 3. An example of appearance probability of normal requests (P_{req}) for a node in one day.

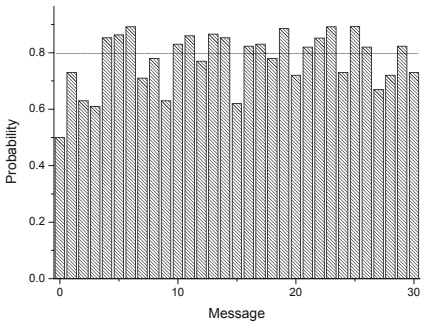

Fig. 4. Aggregated appearance probability of normal requests after one month.

of a challenge, we restore P_{req} to 1/2 when detecting a message is not a normal request. Subsequently, after collecting messages a period of time, it is feasible to model the appearance probability of normal requests based on Eq. (5). The aggregated appearance probability of normal requests for node I ($AP_{req}(I)$) and message j can be computed as below.

$$AP^{j}_{req}(I) = \frac{\sum_{1}^{DN} P^{j}_{req}}{DN} \tag{6}$$

In the above equation, DN denotes the number of days.

Figure 3 shows an example of P_{req} for a node in one day, where a probability of 1/2 means an identification of a message that is not a normal request. To model the appearance probability of normal requests, there is a need to monitor the data for a longer period of time. Thus, we continue collecting statistics of messages, and Fig. 4 depicts a Bayesian distribution of the normal request based on Eq. (6), after a month. It is shown that the appearance probability of normal requests varies with messages. For an attacker, it is critical to select a proper threshold, with the purpose of having a better chance to behave maliciously to normal requests. In this work, we select a threshold of 0.8 to strike a balance between the attack performance and the risk of being detected. For instance, given the threshold of 0.8, our attack allows one to send 16 untruthful feedback to corresponding messages, as shown in Fig. 4.

In summary, after selecting a threshold, suspicious nodes can decide whether to send malicious answers to those messages whose aggregated appearance probability of normal requests is above the threshold, and respond truthfully to other messages. This can circumvent the challenge-based trust mechanism; thus, malicious nodes can have a negative impact on the process of alarm aggregation while maintaining their reputation.

Table 1. Parameter settings in the experiment.

Parameters	Value	Description
ε_l	10/day	Low request frequency
ε_h	20/day	High request frequency
r	0.8	Trust threshold
λ	0.9	Forgetting factor
m	10	Lower limit of received feedback
d	0.3	Severity of punishment
T_s	0.5	Trust value for newcomers

4 Evaluation

We mainly perform two experiments in this section to investigate our attack performance with simulated settings and a real network, respectively.

- *Experiment-1.* We first evaluate our *Bayesian poisoning attack* in a simulated CIDN environment, in comparison with naive collusion attack and *PMFA*.
- *Experiment-2.* We then collaborate with an information center to exploit the impact of our attack in a practical CIDN, e.g., how the reputation of suspicious nodes changes and the influence on aggregating alarms.

4.1 CIDN Settings

A total of 25 nodes that are randomly distributed in the simulated CIDN environment with a 5×5 grid region. Snort [32] was deployed in each node as IDS component. A node can build a *partner list* by communicating with other nodes after a time period. To facilitate the comparison with previous work [6,7], we set the initial trust values of all nodes in the *partner list* as $T_s = 0.5$.

To measure the reputation of each partner node, a challenge can be delivered randomly with an average rate of ε. In this work, we adopted the same frequency levels: a low level of ε_l and a high level of ε_h. Intuitively, we should be confident about a highly trusted or untrusted node; thus, we can set a low request frequency for these nodes. By contrast, we should set a high request frequency to evaluate the nodes with a medium trust value around the threshold. To make a comparison with other competing approaches, we use the same settings as in [6,7,14]. Table 1 describes the parameter settings.

Node Expertise. Similar to previous work, we also adopted three levels of expertise to describe the detection capability of an IDS node, such as low (0.1), medium (0.5) and high (0.95). In particular, we can use the below beta function to measure the expertise of an IDS.

$$\begin{aligned} f(p'|\alpha,\beta) &= \frac{1}{B(\alpha,\beta)} p'^{\alpha-1}(1-p')^{\beta-1} \\ B(\alpha,\beta) &= \int_0^1 t^{\alpha-1}(1-t)^{\beta-1} dt \end{aligned} \tag{7}$$

In the above equation, $p'(\in [0,1])$ describes the possibility of an intrusion under the examination of an IDS. l means the expertise level, $d(\in [0,1])$ indicates the difficulty level, and $f(p'|\alpha,\beta)$ describes the possibility that under d, a node with l can identify an intrusion with p'. Intuitively, a bigger l reflects a better chance of correctly detecting an attack, whereas a bigger d indicates it is harder to identify an attack. The derivation of α and β can refer to the previous work [6].

$$\begin{aligned} \alpha &= 1 + \frac{l(1-d)}{d(1-l)}r \\ \beta &= 1 + \frac{l(1-d)}{d(1-l)}(1-r) \end{aligned} \tag{8}$$

In the above equation, $r \in \{0,1\}$ describes the desirable detection outcomes. Generally, given a fixed d, the node with higher expertise can have better detection performance. For instance, if $d = 0$, then a node with $l = 1$ can identify an attack without errors.

Node Trust Evaluation. To measure the reputation of a node, a randomly generated challenge can be delivered to the tested node. Then we can calculate the satisfaction level based on the received feedback. In particular, the reputation of a node i according to node j can be computed as below [5]:

$$T_i^j = (w_s \frac{\sum_{k=0}^{n} F_k^{j,i} \lambda^{tk}}{\sum_{k=0}^{n} \lambda^{tk}} - T_s)(1-x)^{d'} + T_s, \tag{9}$$

where n counts the number of received feedback, and $F_k^{j,i} \in [0,1]$ describes the satisfaction level regarding the received feedback k. The *forgetting factor*, denoted as λ, emphasizes more weights on the recent feedback. w_s means a *significant weight* varying with the received feedback. If the number of received feedback is smaller than m, then $w_s = \frac{\sum_{k=0}^{n} \lambda^{tk}}{m}$; otherwise, $w_s = 1$. In addition, x describes how many 'do not know' answers are received within a time period, d' is used to punish 'do not know' answers.

Satisfaction Level. Let $e \in [0,1]$ denote an expected feedback, $r \in [0,1]$ denote an actual received feedback, and F $(\in [0,1])$ denote the satisfaction level. Then we can have the followings based on [6,7]:

$$F = 1 - (\frac{e-r}{max(c_1 e, 1-e)})^{c_2} \quad e > r \tag{10}$$

$$F = 1 - (\frac{c_1(r-e)}{max(c_1 e, 1-e)})^{c_2} \quad e \leq r \tag{11}$$

In the above equations, c_1 indicates the penalty degree for errors, and c_2 indicates the sensitivity degree. To make the comparison workable with pervious studies like [6], we adopted $c_1 = 1.5$ and $c_2 = 1$ in the simulation.

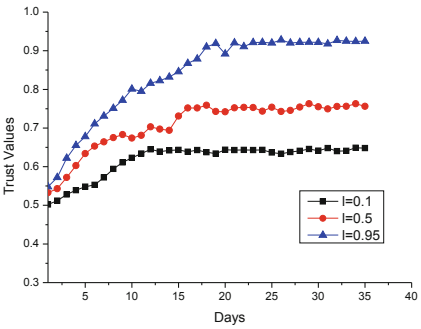

Fig. 5. Convergence of trust values of IDS nodes regarding three expertise levels.

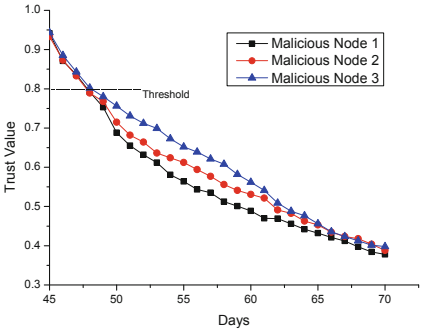

Fig. 6. Trust values of malicious nodes under naive collusion attacks.

4.2 Experiment-1

In this experiment, we attempt to evaluate the initial performance of our attack, naive collusion attack and *PMFA* [16]. Note that naive collusion attack adopts a maximal harm model, as discussed earlier. Under this attack, malicious nodes can cooperate to always respond with untruthful alarm ranking. Figures 5 and 6 depict the convergence of trust values and the reputation of malicious nodes under this attack, respectively.

Figure 5 shows the convergence of trust values for nodes with three expert levels. The observations echoed those of [5,6]; that is, a node with higher expertise can obtain better reputation. For example, the nodes with an expertise of high can reach a trust value above 0.9. In our settings, the reputation of all nodes started to be converged after a period of 15–20 days.

To launch naive collusion attack, we then choose three expert nodes ($I = 0.95$) randomly, which could behave maliciously from Day 45. We denote these nodes as *malicious node 1*, *malicious node 2* and *malicious node 3*. Figure 6 presents the malicious nodes' reputation under this attack. It is observed that within 2–3 days, the reputation of malicious nodes could decrease very fast to below the threshold of 0.8. This is because naive-collusion nodes always behave maliciously, and it is easy to be detected. Subsequently, challenge-based CIDNs can operate well under the native collusion attack, by decreasing malicios nodes' trust values in a short time period.

Advanced Attacks. We further investigate the performance of our attack, as well as those of the *passive message fingerprint attack (PMFA)*. Similarly, we adopted the same three expert nodes of *malicious node 1*, *malicious node 2* and *malicious node 3*. We further remark that the CIDN controls the timing of sending the normal request (i.e., send next request after getting a response from the last request). Figures 7 and 8 show the malicious nodes' reputation and the average errors, respectively.

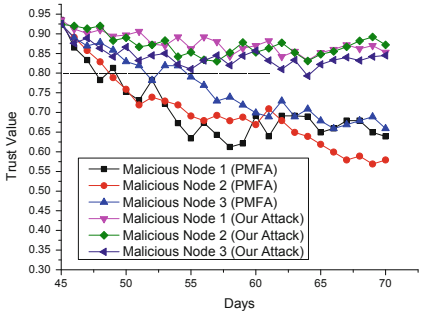

Fig. 7. Trust values of malicious nodes under our attack and PMFA.

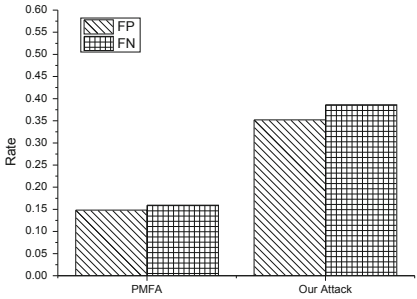

Fig. 8. Average false rates in alarm aggregation under our attack and PMFA.

Figure 7 shows that the reputation of malicious nodes in *PMFA* could drop to below the threshold of 0.8 gradually, because *PMFA* can only ensure that at most one malicious node can identify the normal request, while the other colluding nodes could be identified by challenges. It is found that the trustworthiness of the malicious nodes under our attack decreased slightly but still remained above the threshold. This is because the nodes under our attack only responded untruthfully occasionally.

Figure 8 shows the average alarm aggregation errors between PFMA and our attack. The errors could include both *false negatives (FN)* and *false positions (FP)*. It is easily observed that the alarm aggregation errors are about 14%–16% and 35%–38% under *PMFA* and our attack, respectively. Our attack can make a larger impact on the alarm aggregation process.

Thus, our results demonstrate the feasibility of our attack, where suspicious nodes can have a better chance of behaving maliciously to most normal requests, while providing a truthful response to the remaining messages. In this case, our attack enables malicious nodes acting maliciously without losing their reputation, thus can still make an impact on alarm aggregation.

4.3 Experiment-2

We further collaborated with an information center (including over 1000 personnel) and validate our attack performance in a practical CIDN environment. The information center deploys a wired CIDN, which contains up to 55 nodes, where the incoming network traffic can reach 1305 packets on average. We adopted the same setting as shown in Table 1. Before the experiment, we implemented the challenge mechanism and waited for the trust values to become stable. During this period, each node collected messages and established a Bayesian distribution of normal requests. Similarly, we also selected three expert nodes randomly to be malicious. The main results are depicted in Figs. 9 and 10.

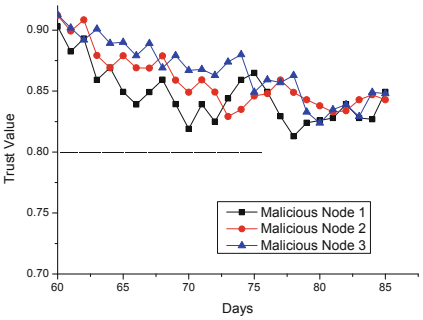

Fig. 9. Trust values of malicious nodes under our attack in a real CIDN.

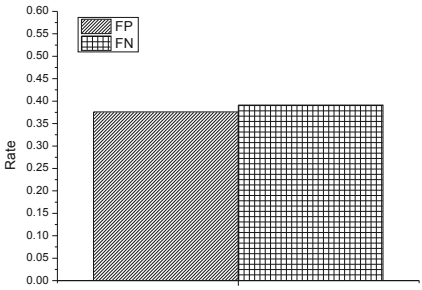

Fig. 10. Average false rates in alarm aggregation under our attack in a real CIDN.

- Figure 9 depicts that the malicious nodes' reputation under our attack could remain above the threshold of 0.8; thus, these nodes can make a negative impact on the alarm aggregation.
- Figure 10 shows that the caused average alarm aggregation errors could be around 37%–39% in the center environment. This is because the suspicious nodes under our attack without detection could keep behaving untruthfully to make an influence on the alarm aggregation.

Thus, our results validate the viability of our attack in a practical CIDN environment, and reveal the limitations of existing challenge-based trust mechanisms against such a practical attack.

4.4 Discussion and Countermeasures

As noted above, existing challenge-based CIDNs need to be redesigned to take into consideration practical attacks such as the attack we reveal in this paper. In other words, sending challenges in a random manner may not be good enough to protect the robustness of challenge-based CIDNs. To defeat our proposed attack, we discuss some potential countermeasures as below.

- One possible solution is to insert a special alarm in a normal request to validate the response. Such special alarm should be inserted in a random position every time, in order to raise the cracking difficulty.
- It is a promising way to combine other types of trust into challenge-based CIDNs. For instance, we can examine the packets that are sent by malicious nodes (denote as packet-level trust), as these nodes are most likely to deliver malicious traffic during an attack [24].
- There is also a good idea to involve some validation mechanisms, which can help check whether the received feedback is malicious or not. Due to the popularity of blockchain technology, it can be considered to help build a trusted feedback-chain that can be validated by all peers.

4.5 Limitations

In this work, our main purpose is to exploit the robustness of challenge-based CIDNs by designing a practical insider attack - *Bayesian Poisoning Attack*. Due to the scope, there are some limitations and open challenges on this topic.

- *CIDN deployment.* Distributed/collaborative intrusion detection has been proposed for decades, while these systems are mainly deployed at a small-scale, i.e., security-sensitive organizations, and security companies. With the current threats being more sophisticated, there is a trend towards developing a more practical, effective and robust DIDS/CIDN in various organizations. Our work is an effort to stimulate more research in this direction.
- *Existing security mechanisms.* There are many other security mechanisms in practice, like OSINT-based curated security feeds/platforms, and security information and event management (SIEM) system that correlates events coming from multiple sensors. Because of our focus, we did not discuss the existing security mechanisms, but intuitively, DIDS/CIDN is an alternative to collaborate with existing security solutions.

In the current literature, distributed/collaborative intrusion detection has been applied to many disciplines. For example, Shekari *et al.* [30] focused on supervisory control and data acquisition (SCADA) and proposed a radio frequency-based distributed intrusion detection system (RFDIDS), which uses radio frequency (RF) emissions to monitor the power grid substation activities. Hence, our work advocates the need of enhancing intrusion detection by enabling collaboration among various detectors, but also figures out the open challenge on how to design a practical, effective and robust DIDS/CIDN.

5 Related Work

Intuitively, it is challenging for an isolated IDS to learn about the evolving, real-time threat landscape. Thus, there is a need for sharing of information and threats, for example via a distributed system. Examples of such systems include *Centralized/Hierarchical systems* (e.g., Emerald [28] and DIDS [31]), *Publish/subscribe systems* (e.g., COSSACK [27] and DOMINO [36]), and *P2P Querying-based systems* (e.g., Netbait [1] and PIER [10]). CIDN is yet another example, and the focus of this paper. Specifically, in a CIDN [35], an IDS node can achieve better accuracy by collecting and communicating relevant information from other IDS nodes (i.e., collating the alerts from other nodes and synthesizing a global and aggregated alarm). However, insider attack is a major threat for such collaborative networks, as shown by researchers such as Li *et al.* [12]. Specifically, Li *et al.* [12] proposed a system based on the emerging decentralized location and routing infrastructure, and assumed that all peers are trusted. This is clearly vulnerable to insider attacks, such as betrayal attacks where some nodes become malicious suddenly (e.g., due to compromise).

The need to establish appropriate and strong trust models to defend against insider attacks is well-studied. Duma *et al.* [3], for example, proposed a P2P-based overlay for intrusion detection (Overlay IDS) that mitigated such insider threats, by using a trust-aware engine for correlating alerts and an adaptive scheme for managing trust. Tuan [34] utilized the game theory to model and analyze the processes of reporting and exclusion in a P2P network. They identified that if a reputation system is not incentive compatible, then peers in the system will be less inclined to report a malicious peer.

While challenge-based trust mechanisms are an effective approach of building trust among CIDN nodes (i.e., trustworthiness of a node depends on the received answers to the challenges), they are not infallible. Fung et al. [5] proposed a HIDS collaboration framework, which enables each HIDS to evaluate the trustworthiness of others based on its own experience by means of a forgetting factor. The forgetting factor can give more emphasis on the recent experience of the peer. Then, they improved their trust management model by using a Dirichlet-based model to measure the level of trustworthiness among IDS nodes, according to their mutual experience [6]. This model has strong scalability properties and is sufficiently robust against common insider threats, as demonstrated by their evalution findings. As feedback aggregation is a key component in a challenge mechanism, they further applied a Bayesian approach to feedback aggregation to minimize the combined costs of missed detection and false alarm [7].

To further improve the detection accuracy of challenge-based CIDNs, Li *et al.* [13] explained that different IDS nodes may have different levels of sensitivity in detecting different types of intrusions. They also proposed a notion of *intrusion sensitivity* that measures the detection sensitivity of an IDS in detecting different kinds of intrusions. For example, if a signature-based IDS node has more signatures (or rules) in detecting DoS attacks, then it should be considered to be more powerful in detecting such attacks, in comparison to other nodes with relatively fewer signatures. This notion is helpful when making decisions based on the collected information from different nodes, as it can help detect intrusions and correlate IDS alerts through emphasizing the impact of an *expert IDS*. Li *et al.* [14] further proposed an *intrusion sensitivity-based trust management model* for automating the allocation of *intrusion sensitivity*, using machine learning techniques such as knowledge-based KNN classifier [22]. The use of *intrusion sensitivity* could also be beneficial for alarm aggregation and defending against pollution attacks [15]. Experimental results demonstrated that *intrusion sensitivity* can decrease the trust values of malicious nodes promptly. Other related studies on intrusion detection enhancement include those of [4,9,18–21,24,25]

Challenge-based CIDNs are robust against common insider attacks, whereas some advanced insider threats are still feasible. Li *et al.* [16] showed an advanced insider attack named *passive message fingerprint attack (PMFA)*, where multiple suspicious nodes could work together to identify normal requests from the received messages. They further introduced another attack called Special On-Off Attack *SOOA* [17], which could send truthful answers to partial messages while behaving maliciously to other messages. The random poisoning attack [23] is a

special case of *SOOA*, in which malicious nodes can send a malicious response with a possibility of 1/2. The main difference between the above studies and our work is that we use a Bayesian approach to identify normal requests with a high possibility, hence each malicious node can have their own possibility list.

6 Conclusion

With the purpose of enhancing the robustness of challenge-based CIDNs against a broader range of attacks, we posit the importance of designing advanced and practical attacks. In this paper, we developed a type of collusion attack, *Bayesian Poisoning Attack*, which enables a malicious node to establish an appearance probability of normal requests based on Bayesian inference. In our attack, malicious nodes can respond untruthfully to messages, which have a higher possibility to be a normal request. In the evaluation, we compare our attack with naive collusion attack and PMFA under both simulated and practical environment, and experimental results demonstrated the utility of our attack (i.e., malicious nodes can respond untruthfully, while maintaining their trust values and causing errors in the alarm aggregation). We also discuss some potential countermeasures (e.g., combining other trust types) to defeat such type of attack, which could be one of our future work. Our work attempts to stimulate more research in building robust and practical challenge-based CIDNs.

Acknowledgments. We would like to thank all anonymous reviewers for their helpful comments in improving the paper. Weizhi Meng was partially supported by H2020 SU-ICT-03-2018 CyberSec4Europe.

References

1. Chun, B., Lee, J., Weatherspoon, H., Chun, B.N.: Netbait: a Distributed Worm Detection Service. Technical Report IRB-TR-03-033, Intel Research Berkeley (2003)
2. Douceur, J.R.: The sybil attack. In: Druschel, P., Kaashoek, F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45748-8_24
3. Duma, C., Karresand, M., Shahmehri, N., Caronni, G.: A trust-aware, P2P-based overlay for intrusion detection. In: DEXA Workshop, pp. 692–697 (2006)
4. Friedberg, I., Skopik, F., Settanni, G., Fiedler, R.: Combating advanced persistent threats: from network event correlation to incident detection. Comput. Secur. **48**, 35–57 (2015)
5. Fung, C.J., Baysal, O., Zhang, J., Aib, I., Boutaba, R.: Trust management for host-based collaborative intrusion detection. In: De Turck, F., Kellerer, W., Kormentzas, G. (eds.) DSOM 2008. LNCS, vol. 5273, pp. 109–122. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87353-2_9
6. Fung, C.J., Zhang. J., Aib, I., Boutaba, R.: Robust and scalable trust management for collaborative intrusion detection. In: Proceedings of the 11th IFIP/IEEE International Conference on Symposium on Integrated Network Management (IM), pp. 33–40 (2009)

7. Fung, C.J.; Zhu, Q., Boutaba, R., Basar, T.: Bayesian decision aggregation in collaborative intrusion detection networks. In: NOMS, pp. 349–356 (2010)
8. Fung, C.J., Boutaba, R.: Design and management of collaborative intrusion detection networks. In: Proceedings of the 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 955–961 (2013)
9. Gou, Z., Ahmadon, M.A.B., Yamaguchi, S., Gupta, B.B.: A petri net-based framework of intrusion detection systems. In: Proceedings of the 4th IEEE Global Conference on Consumer Electronics, pp. 579–583 (2015)
10. Huebsch, R., et al.: The architecture of PIER: an internet-scale query processor. In: Proceedings of the 2005 Conference on Innovative Data Systems Research (CIDR), pp. 28–43 (2005)
11. Kiennert, C., Ismail, Z., Debar, H., Leneutre, J.: A survey on game-theoretic approaches for intrusion detection and response optimization. ACM Comput. Surv. (CSUR) **51**(5), 90 (2018)
12. Li, Z., Chen, Y., Beach, A.: Towards scalable and robust distributed intrusion alert fusion with good load balancing. In: Proceedings of the 2006 SIGCOMM Workshop on Large-Scale Attack Defense (LSAD), pp. 115–122 (2006)
13. Li, W., Meng, Y., Kwok, L.-F.: Enhancing trust evaluation using intrusion sensitivity in collaborative intrusion detection networks: feasibility and challenges. In: Proceedings of the 9th International Conference on Computational Intelligence and Security (CIS), pp. 518–522. IEEE (2013)
14. Li, W., Meng, W., Kwok, L.-F.: Design of intrusion sensitivity-based trust management model for collaborative intrusion detection networks. In: Zhou, J., Gal-Oz, N., Zhang, J., Gudes, E. (eds.) IFIPTM 2014. IAICT, vol. 430, pp. 61–76. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43813-8_5
15. Li, W., Meng, W.: Enhancing collaborative intrusion detection networks using intrusion sensitivity in detecting pollution attacks. Inf. Comput. Secur. **24**(3), 265–276 (2016)
16. Li, W., Meng, W., Kwok, L.-F., Ip, H.H.S.: PMFA: toward passive message fingerprint attacks on challenge-based collaborative intrusion detection networks. In: Proceedings of the 10th International Conference on Network and System Security (NSS), pp. 433–449 (2016)
17. Li, W., Meng, W., Kwok, L.F.: SOOA: exploring special on-off attacks on challenge-based collaborative intrusion detection networks. In: Proceedings of GPC, pp. 402–415 (2017)
18. Meng, Y., Kwok, L.F.: Enhancing false alarm reduction using voted ensemble selection in intrusion detection. Int. J. Comput. Intell. Syst. **6**(4), 626–638 (2013)
19. Meng, Y., Li, W., Kwok, L.F.: Towards Adaptive character frequency-based exclusive signature matching scheme and its applications in distributed intrusion detection. Comput. Netw. **57**(17), 3630–3640 (2013)
20. Meng, W., Li, W., Kwok, L.-F.: An evaluation of single character frequency-based exclusive signature matching in distinct IDS environments. In: Proceedings of the 17th International Conference on Information Security (ISC), pp. 465–476 (2014)
21. Meng, W., Li, W., Kwok, L.-F.: EFM: enhancing the performance of signature-based network intrusion detection systems using enhanced filter mechanism. Comput. Secur. **43**, 189–204 (2014)
22. Meng, W., Li, W., Kwok, L.-F.: Design of intelligent KNN-based alarm filter using knowledge-based alert verification in intrusion detection. Secur. Commun. Netw. **8**(18), 3883–3895 (2015)

23. Meng, W., Luo, X., Li, W., Li, Y.: Design and evaluation of advanced collusion attacks on collaborative intrusion detection networks in practice. In: Proceedings of the 15th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1061–1068 (2016)
24. Meng, W., Li, W., Kwok, L.-F.: Towards effective trust-based packet filtering in collaborative network environments. IEEE Trans. Netw. Serv. Manage. **14**(1), 233–245 (2017)
25. Mishra, A., Gupta, B.B., Joshi, R.C.: A comparative study of distributed denial of service attacks, intrusion tolerance and mitigation techniques. In: Proceedings of the 2011 European Intelligence and Security Informatics Conference, pp. 286–289 (2011)
26. Nisioti, A., Mylonas, A., Yoo, P.D., Katos, V.: From intrusion detection to attacker attribution: a comprehensive survey of unsupervised methods. IEEE Commun. Surv. Tutorials **20**(4), 3369–3388 (2018)
27. Papadopoulos, C., Lindell, R., Mehringer, J., Hussain, A., Govindan, R.: COSSACK: coordinated suppression of simultaneous attacks. In: Proceedings of the 2003 DARPA Information Survivability Conference and Exposition (DISCEX), pp. 94–96 (2003)
28. Porras, P.A., Neumann, P.G.: Emerald: event monitoring enabling responses to anomalous live disturbances. In: Proceedings of the 20th National Information Systems Security Conference, pp. 353–365 (1997)
29. Scarfone, K., Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS). NIST Special Publication 800-94 (2007)
30. Shekari, T., Bayens, C., Cohen, M., Graber, L., Beyah, R.: RFDIDS: radio frequency-based distributed intrusion detection system for the power grid. In: Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS) (2019)
31. Snapp, S.R., et al.: DIDS (Distributed Intrusion Detection System) - motivation, architecture, and an early prototype. In: Proceedings of the 14th National Computer Security Conference, pp. 167–176 (1991)
32. Snort: An an open source network intrusion prevention and detection system (IDS/IPS). http://www.snort.org/
33. Sun, Y., Yu, W., Han, Z., Liu, K.: Information theoretic framework of trust modeling and evaluation for ad hoc networks. IEEE J. Sel. Areas Commun. **24**(2), 305–317 (2006)
34. Tuan, T.A.: A game-theoretic analysis of trust management in P2P systems. In: Proceedings of ICCE, pp. 130–134 (2006)
35. Wu, Y.-S., Foo, B., Mei, Y., Bagchi, S.: Collaborative Intrusion Detection System (CIDS): a framework for accurate and efficient IDS. In: Proceedings of the 2003 Annual Computer Security Applications Conference (ACSAC), pp. 234–244 (2003)
36. Yegneswaran, V., Barford, P., Jha, S.: Global intrusion detection in the DOMINO overlay system. In: Proceedings of the 2004 Network and Distributed System Security Symposium (NDSS), pp. 1–17 (2004)