Towards Trust-Aware Collaborative Intrusion Detection: Challenges and Solutions

Emmanouil Vasilomanolakis(✉), Sheikh Mahbub Habib, Pavlos Milaszewicz, Rabee Sohail Malik, and Max Mühlhäuser

Telecooperation Group, Technische Universität Darmstadt, Darmstadt, Germany
{vasilomano,habib,max}@tk.tu-darmstadt.de,
{pavlos.milaszewicz,rabeesohail.malik}@stud.tu-darmstadt.de

Abstract. Collaborative Intrusion Detection Systems (CIDSs) are an emerging field in cyber-security. In such an approach, multiple sensors collaborate by exchanging alert data with the goal of generating a complete picture of the monitored network. This can provide significant improvements in intrusion detection and especially in the identification of sophisticated attacks. However, the challenge of deciding to which extend a sensor can trust others, has not yet been holistically addressed in related work. In this paper, we firstly propose a set of requirements for reliable trust management in CIDSs. Afterwards, we carefully investigate the most dominant CIDS trust schemes. The main contribution of the paper is mapping the results of the analysis to the aforementioned requirements, along with a comparison of the state of the art. Furthermore, this paper identifies and discusses the research gaps and challenges with regard to trust and CIDSs.

1 Introduction

With the continuous growth of cyber-attacks, Intrusion Detection Systems (IDSs) are nowadays considered a mandatory line of defense for any type of network [6]. However, as isolated IDSs do not scale and are not capable of detecting distributed and highly sophisticated attacks, more collaborative approaches have emerged. The term Collaborative IDS (CIDS) describes systems that exhibit such a cooperative approach [10]. In a CIDS, a plethora of different sensors (e.g., honeypots, firewalls, IDSs, etc.) collaborate by exchanging alert data with the scope of creating a holistic picture of the monitored network. As sensors exchange data and correlate information, it becomes feasible to detect a larger portion of attacks. Moreover, in contrast to isolated IDSs that do not scale, these systems can monitor very large networks.

However, a big challenge in CIDSs is the ability to manage the various sensors in an efficient and productive manner. In this context, the aspect of trust is of high importance for CIDSs. First, with the usage of computational trust it is possible to deal with insider attacks [3]. Such attacks refer to cases in which a number of sensors, inside the CIDS, are infected or compromised. In such an

J.-P. Steghöfer and B. Esfandiari (Eds.): IFIPTM 2017, IFIP AICT 505, pp. 94–109, 2017.
DOI: 10.1007/978-3-319-59171-1_8

event, rogue sensors can significantly reduce the accuracy of the overall system by contaminating the alert exchange process with fake alerts. Second, apart from insider attacks, trust mechanisms are valuable for assessing the quality and thus the weight of importance that different sensors ought to have. For instance, in a large CIDS, a multitude of heterogeneous sensors is to be expected; from highly trusted IDSs to honeypots and/or to third party untrustworthy sources of alert data. In all cases, the CIDS needs to be able to assess which sources are more relevant and/or reliable.

In this paper, we attempt to bridge the areas of computational trust and collaborative intrusion detection, discuss the state of the art, and identify the respective research gaps. We firstly propose a number of requirements for reliable Trust Management (TM) in CIDSs. Afterwards, we carefully investigate the related work for the most dominant and promising CIDS trust schemes. The trust components of the identified systems are discussed separately on the basis of the aforesaid requirements. Furthermore, we compare all the trust mechanisms by mapping them to the requirements. In addition, based on our analysis, we identify and discuss research gaps and challenges with regard to trust and CIDSs.

The remainder of this paper is organized as follows. In Sect. 2, we propose a number of requirements for trust mechanisms in CIDSs. On this basis, Sect. 3 provides a brief description and analysis of the most prominent CIDS trust mechanisms. Furthermore, Sect. 4 contains a detailed comparison of these mechanisms by mapping them to the requirements and provides future directions. Lastly, Sect. 5 concludes the paper.

2 Requirements

Managing trust in a CIDS is a complex problem which has many conflicting requirements for which an acceptable tradeoff has to be chosen. In our previous work, we have examined the related work in both CIDSs [10] and computational trust [5]. On this basis, along with an additional study of the state of the art in trust mechanisms for CIDSs, we propose the following requirements. These will be utilized along the discussion of the different approaches in Sect. 3 and will be more extensively analyzed in Sect. 4.

- **Global view:** Some approaches for managing trust require a *global view* of the monitored network, in which an administrator has full control over the sensors or sensors have full-fledged information about the entire network. This is not always realistic; for instance, fully distributed CIDSs usually cannot guarantee such a global view. Hence, approaches that do not require global view can be applied to a larger variety of CIDSs.
- **Minimum overhead:** The overhead associated with the computing and managing of trust in a CIDS should be kept to a minimum. In particular, the overhead can be either communicational or computational. *Communicational overhead* refers to the need of the trust mechanism to generate additional messages. *Computational overhead* is associated with the computational power required to compute the various trust values.

- **Incentive mechanism:** An *incentive mechanism* refers to the ability of a trust system to motivate and reward sensors for behaving in a trustworthy manner. An example of such an incentive can be the ability to give alert data feedback only to sensors with high trust values.
- **Initial trust:** Assigning a reasonable *initial trust* value to a sensor that has recently joined the CIDS is a challenging task [8]. As historical data do not always exist for newcomers, the trust mechanism has to choose between assigning random values, a probation period approach or the assignment of high/low initial trust values. Each approach has certain advantages and disadvantages that will be discussed in the following sections.
- **Forgetting factor:** The *forgetting factor* (or aging) is a parameter that ensures that the most recent feedback, given by nodes, carries more weight than less recent feedback. This is desirable as it allows a more accurate and up-to-date calculation of trust values.
- **Performance history:** The *performance history* describes how a sensor has performed based on historical data (e.g., old transactions).

3 CIDS Trust Management

In this section, we analyze and discuss four trust approaches for CIDSs. The selected systems were identified by analyzing the state of the art. In particular, the emphasis of our analysis lies on the collaboration framework or architecture, the TM mechanisms, and the utilized evaluation methods.

3.1 Dirichlet-Based Trust Management

Fung et al. proposed a TM model to facilitate an effective trust-aware CIDS [4]. The system consists of three main components: the Collaboration component, the TM component and Acquaintance Management component. The Collaboration Framework connects different hosts in a network and allows them to communicate in a fair and scalable manner. The TM framework leverages the collaboration framework to establish trust among networked hosts based on the history of their performance. It uses Bayesian statistics to calculate the trustworthiness of hosts. Finally, the Acquaintance Management is used to manage a list of trustworthy acquaintances using test messages. Each of these components is described in the following.

Collaboration Component. This component has an incentive mechanism for hosts to share information and manage their acquaintances. Each host maintains a list of acquaintances, peers (i.e., other sensors in the CIDS) that it trusts and collaborates with. Each sensor sends two types of requests to its peers, *intrusion consultation messages* and *test messages*. Intrusion consultation messages are sent when a host needs feedback to determine whether an alarm should be raised or not. The amount of information that a host shares with a peer depends

on the trustworthiness of that peer; more trustworthy peers receive more information than less trustworthy ones. Additionally, each host sends test messages periodically. The nature of the test message is known to the sender beforehand. Test messages are used by a sensor to establish the trust levels of its peers. Such messages can be generated artificially using a knowledge database.

Trust Management Component. To establish trust, hosts send requests to peers and evaluate the satisfaction levels of the reply. The alert ranking raised by a host lies in the interval [0,1] where 0 is harmless and 1 is highly dangerous. The satisfaction level of a reply from an acquaintance is a function of three parameters, the expected answer (r), the received answer (a) and the difficulty level (d) of the test message. The values of these three parameters also lie in the interval [0,1]. The function $Sat(r, a, d)(\in [0, 1])$ (see Eq. 1) represents the satisfaction level of the given feedback. The value of c_1 determines the level of penalization of a wrong estimate. Parameter c_2 controls sensitivity of the satisfaction level to the distance between the expected and received answer.

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

Bayesian statistics, and specifically the Dirichlet distribution, is used to model the distribution of past satisfaction levels from the acquaintances of each peer. The Dirichlet distribution is utilized since the authors are modeling multi-valued satisfaction levels; it is a continuous, multivariate probability distribution, which is a generalization of the Beta Distribution for multivariate values. This prior distribution is then used to estimate the posterior distributions, i.e. satisfaction levels of future answers.

If X is the random variable representing the satisfaction level of feedback from a peer, then X can take values $\chi = \{x_1, x_2...x_k\}$ of the supported levels of satisfaction where each value lies in the interval [0,1]. $\overrightarrow{P} = \{p_1, p_2...p_k\}$ is the probability distribution vector of X such that $P\{X = x_i\} = p_i$. The cumulative observations and beliefs of X are represented by $\overrightarrow{\gamma} = \{\gamma_1, \gamma_2...\gamma_k\}$. Using the Dirichlet distribution, the vector $\overrightarrow{p}$ is modeled as:

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

where ξ is the background information which is represented by $\overrightarrow{\gamma}$. Let $\gamma_0 = \sum_{i=1}^{k} \gamma_i$, then the expected value of probability of X to be x_i is then given by: $E(p_i|\overrightarrow{\gamma}) = \frac{\gamma_i}{\gamma_0}$. Moreover, a forgetting factor λ can be used to give recent observations more importance which leads to: $\overrightarrow{\gamma}^{(n)} = \sum_{i=1}^{n} \lambda^{t_i} \times \overrightarrow{S}^i + c_0 \lambda^{t_0} \overrightarrow{S}^0$.

S_0 is the initial beliefs vector and c_0 is the constant which puts weight on the initial beliefs. t_i represents the time elapsed since the i^{th} evidence. When feedback is received from a peer, it is given a score according to Eq. 1. p_i^{uv} denotes the probability that feedback from peer v to peer u has the satisfaction value x_i. The sum of P^{uv} over all i is equal to 1. $\overrightarrow{P}^{uv}$ is modeled using Eq. 2. Y^{uv} is the random variable such that: $Y^{uv} = \sum_{i=1}^{k} p_i^{uv} w_i$. The following equation then gives the trustworthiness of a peer:

$$T^{uv} = E[Y^{uv}] = \sum_{i=1}^{k} w_i E[p_i^{uv}] = \frac{1}{\gamma_0^{uv}} \sum_{i=1}^{k} w_i \gamma_i^{uv} \tag{3}$$

Where γ_i^{uv} is the cumulative evidence that v has replied to u with satisfaction level x_i. As soon as the trustworthiness has been calculated, feedback only from peers whose trustworthiness levels exceed a certain level is considered. An upper bound for the trust level is calculated using the covariance of p_i and p_j. Once feedback from acquaintances has been collected, it is aggregated using the following weighted majority formula:

$$\bar{a}_i^u = \frac{\sum_{T_l^{uv} \geq th^u, v \in A^u} T^{uv} a_i^{uv}}{\sum_{T_l^{uv} \geq th^u, v \in A^u} T^{uv}} \tag{4}$$

Where $\bar{a}^u$ is the aggregated ranking of alert i. T^{uv} is the trustworthiness of peer v to peer u. This formula is applied only to feedback from peers with trustworthiness higher than a certain threshold. a_i^{uv} is the ranking of the alert i given from u to v.

Acquaintance Management. The authors contributed an algorithm to maintain a list of acquaintances in the proposed system. Maintaining such a list is necessary since it is not scalable for the host to keep records for all the peers in the network. Each host maintains a list of trusted nodes, with the length of the list depending on the available resources. Since it takes time to determine the trustworthiness of a peer, a probation list should also be maintained. The host communicates with peers in its probation list to determine their trust levels and if the levels exceed a certain threshold, the relevant nodes can be added to the acquaintance list.

Experiments and Results. The authors simulate an environment in which a host is allowed to have an acquaintance list with 40 dishonest peers, divided equally into 4 groups. Each group uses a different strategies for its dishonesty - complimentary, exaggerate positive, exaggerate negative and maximal harm. Complimentary simply inverts the alert level of a message. For example, an actual alert level of 0.7 will be converted to a 0.3. Exaggerate positives and exaggerate negatives convert low positives and negatives to high positives and

negatives [11]. In maximal harm, the peer reports false feedback with the intention of causing the most harm to the host. The trust values of the peers converge after 30 days and, as expected, the trust value of the peers using the maximal harm scheme is the lowest. Fung et al. also conduct an experiment in which a peer behaves honestly for 50 days and then launches a betrayal attack using a maximal harm scheme. The results indicate that the trust value of that peer decreases rapidly due to the forgetting factor used to associate more weight on recent messages. Additionally, once a peer is downgraded from "highly trustworthy" to "trustworthy" the rate of test messages sent to it is increased.

3.2 Trust Diversity

Perez et al. [9] proposed the notion of Trust Diversity (TD) to maximize the information quality and resilience against dishonest behavior of CIDS sensors. TD is defined as the measurement of the dispersion of the trust values of sensors in a given domain. For instance, a low diversity would indicate that the sensors exhibit similar trustworthiness. The goal is to find a placement of sensors such that TD is maximized in all given domains. Thus, all domains will have a roughly equal distribution of trustworthy and untrustworthy sensors, leaving no domain unprotected. Furthermore, when there is high TD, the more reliable sensors can help identifying untrustworthy sensors, making the system more resilient to insider attacks. Finally, higher TD also leads to a system which is more resilient to external attacks.

System Model. The services and resources of a CIDS can be divided into different *domains* (D). Each domain has a set of *requirements* (R) for the proper functioning of the entire system. Each sensor has certain *properties* (P) to monitor certain requirements, denoted by $P(S_j)$. For each sensor, a reputation value ($Rep(S_j)$) is also maintained. This value is based on the assessment of the sensor's past behavior. The CIDS can be configured to deploy sensors as required in different domains to reach a desired goal.

Sensor Placement. Reconfiguring the placement of sensors is important to increase TD which reduces the uncertainty about the nature of events, that is, if they are malicious or not. It also allows the reassessment of trustworthiness in sensors, as the feedback from more trustworthy sensors can be compared with the feedback given from less trustworthy sensors. The authors propose a *Trust and Reputation module* which uses the past behavior of the sensors to monitor the quality of each domain. The module uses three metrics, the past behavior of a sensor, the past behavior of a sensor's neighbors and the sensor's capabilities. The final result is the trustworthiness value, calculated for a sensor considering these metrics. The value is then used to compute the TD in a given domain. Once the TD of all the domains has been computed, an optimization algorithm is used to generate the best possible reconfiguration of the sensors.

Trust and Reputation Management System. To compute the trustworthiness of sensors, Perez et al. make use of a trust and reputation management system. The computed trustworthiness can then be used to maximize TD. Computing the inter-quartile range, mean difference and arithmetic difference are some examples of how this diversity can be quantified. TD is computed at three different levels: at the requirement level such that diversity is maximized among sensors assigned to fulfill a certain requirement; at the domain level to ensure that there is a diverse spread of sensors' reputation levels in all domains; and lastly, at the global system level such that no domain is left unprotected. TD at the requirement level, for a requirement R_k for a domain Ω, denoted by $TD_\Omega \in [0, 1]$ can be calculated by:

$$TD_\Omega(R_k) = max\{Rep_\Omega(S_{R_k})\}.\psi(Rep_\Omega(S_{j,R_k}).\mu_\Omega(R_{k,S_j})), \forall S_j \in SP(\Omega) \quad (5)$$

where ψ is the dispersion among the sensors' reputation, $max\{Rep_\Omega(S_{Rk})\}$ denotes the highest reputation value among all sensors in the given domain Ω, $Rep_\Omega(S_{j,R_k})$ is the reputation of the j^{th} sensor in Ω fulfilling requirements R_k and $\mu_\Omega(R_{k,S_j})$ is the risk incurred when R_k is not satisfied. Once the TD has been calculated at the requirement level, it can then be calculated at the domain level as: $TD_\Omega = \oplus_{k=1}^{\phi_\Omega(R)} TD_\Omega(R_k)$. $TD_\Omega(R_k)$ is the TD of the k^{th} requirement calculated in (5). $\phi_\Omega(R)$ represents the total number of requirements in Ω. $\oplus$ is an aggregation operation, for example, arithmetic mean or harmonic mean. Finally, TD at the CIDS level can be calculated as: $TD_{CIDS} = \oplus_{i=1}^{\phi_{CIDS}(D)} TD_{D_i}$ where $\phi_{CIDS}(D)$ is the number of domains in the CIDS and TD_{D_i} is the TD of each domain.

Upon receiving the alert of a new event, the monitoring system first assesses the trust in the event being true and then updates the reputation levels of all relevant sensors which are configured to report such an event. The trust of an event is the confidence the system places on an event being true. Three factors are used to compute the trust level in an event: the agreement level of all relevant sensors; the number of domains in which the event was detected and the TD in all such domains. Using these three factors, the trust level of an alert can be calculated as follows:

$$T(E_{R_k}) = \oplus_{i=1}^{\phi_D(E_{R_k})} |\delta_{D_i}(E_{R_k})|.TD_{D_i}(R_k) \quad (6)$$

where $\phi_D(E_{R_k})$ is the number of domains from which the event alert has been issued. $\delta_{D_i}(E_{R_k})$ is the level of agreement of the relevant sensors in the ith domain D_i, relevant to the event E_{R_k}. $TD_{D_i}(R_k)$ is the TD of each domain where the event happened. The agreement of relevant sensors on a given event is calculated using a voting scheme and can be computed as follows:

$$\delta_\Omega(E_{R_k}) = \frac{\sum_{j=1}^{\phi_{S_\Omega}(E_{R_k})} Rep_\Omega(S_{j,R_k})}{\phi_{S_\Omega}(E_{R_k})} - \frac{\sum_{j=1}^{\phi_{S_\Omega}(E_{R_k})} Rep_\Omega(S_{j,R_k})}{\phi_{S_\Omega}(\neg E_{R_k})} \quad (7)$$

Here, $Rep(S_{j,R_k})$ is the reputation of the j^{th} sensor fulfilling requirement R_k, $\phi_{S_\Omega}(E_{R_k})$ is the number of relevant sensors in domain Ω that have issued

a notification for the event, and $\phi_{S_\Omega}(\neg E_{R_k})$ is the number of sensors that did not report it. A neutral agreement with the computed value of 0 indicates total uncertainty about whether a given event is true or not. A value of 1 indicates full confidence that is true and a value of -1 indicates that it is false. Once a trust value is assigned to an event it can be used to update the reputation of the involved sensors as follows:

$$RepS_j{}^{(t)} = \omega RepS_j{}^{(t-1)} + (1-\omega)\frac{\sum_{k=1}^{\phi_{S_j}(E)} Sat(S_j, E_k)\mu(R_{E_k S_j})\xi(E_k)}{\phi_{S_j}(E)} \quad (8)$$

where $\phi_{S_j}(E)$ is the total number of events the sensor j has been involved in. $Sat(S_j, E_k)$ is the calculated satisfaction level of the behavior shown by S_j with regards to the event E_k. Moreover, $RepS_j{}^{(t-1)}$ is the last reputation value of S_j, while $\mu(R_{E_k S_j})$ is the associated risk of the requirement affected by E_k, and $\xi(E_k)$ is a forgetting factor. Lastly, ω is the weight on each term which determines importance of past behavior. The satisfaction level of the behavior of a sensor with regards to an event is dependent on the trust value of that event and the action of the given sensor. It can be computed as follows:

$$Sat(S_j, E_k) \quad = \quad \begin{cases} |\delta(E_k)| \;\; if(T(E_k) \geq T_\sigma \wedge S_j \subseteq G_s(E_k)) \\ \qquad\qquad \vee (T(E_k) < T_\sigma \wedge S_j \nsubseteq G_s(E_k)) \\ |-\delta(E_k)| \;\; otherwise \end{cases} \quad (9)$$

Here, T_σ is the threshold which decides whether an event is trustworthy or not. $G_s(E_k)$ is the set of all sensors which have issued a notification for the event E_k. Therefore, if an event is trustworthy, and its notification has been issued by the sensor, the reputation of that sensor will increase. The opposite is true for when the event is considered not true. When the TD of a requirement, domain or the entire CIDS falls below a certain level, then a new configuration could be found which increases it once again.

Experiments and Results. The authors experiment with a simulation that includes 500 sensors, 20 domains and 10 requirements. The initial reputation of each sensor is assigned a random value. The first experiment assesses events in a domain with higher TD compared to a domain with lower TD. The authors use an optimization algorithm to search for a placement with high TD. They simulated over 2000 events and assessed the trust levels of these events (see Eq. 7). The value of agreement level is between 1 and -1. A value close to 1 means that the sensors are in agreement that an event is true while a value close to -1 means that the sensors agree that an event is bogus. For domains with lower TD, the agreement level for honest events and bogus events were 0.3139 and -0.3102 respectively. For the domains with higher TD, the agreement level for the honest and bogus events were 0.7253 and -0.7098 respectively, a significant improvement. Another experiment was to test the effect TD has on the resilience of the system to malicious sensors. The authors incrementally increase

the number of malicious sensors in the system and observe the effect this has on the trust values of the malicious sensors. With higher TD, these trust values decrease more rapidly than in domains with lower TD. This can give a clear indication that the system is being compromised by malicious sensors.

3.3 A Trust-Aware P2P-Based CIDS

Duma et al. [1] proposed a trust-aware Peer-to-Peer (P2P)-based CIDS. The proposed system consists of a trust-aware correlation engine and an adaptive TM scheme. The correlation engine is used to filter data sent by untrustworthy peers. The TM scheme works by using past experience to decide if peers are trustworthy or not. The sensors form a P2P network, in which they are interconnected and communicate to detect and prevent attacks.

Trust Management. To calculate trust among sensors, each peer has a list of peers that it trusts, and checks for peer trustworthiness before taking any decision regarding a possible threat. To adjust the trustworthiness of a peer, the local peer will evaluate any event according to if it was an incident or not, and adjust its trust regarding other peers. In more details, each peer P_i has a list of acquaintance peers, which consists of other peers that P_i has interacted with and their trust value. For each peer P_j which is present in the acquaintance list, P_i keeps two variables: the first one is s_{ij} which represents the number of successful experiences that i had with j. The second one is u_{ij} which represents the number of unsuccessful experiences that i had with j. Having these, peer i computes the trustworthiness of peer j as: $t_{ij} = w_s \frac{s_{ij}-u_{ij}}{s_{ij}+u_{ij}}$. The w_s parameter, is called significance weight and depends on the total number of experiences that are available for the computations regarding trust. If the number of experiences available are too less, then a peer's trustworthiness cannot be computed by this formula. That means that if the total number of experiences $s_{ij}+u_{ij}$ is below a certain minimum number n, then $w_s = (s_{ij} + u_{ij})/n$, otherwise $w_s = 1$.

A *trust threshold* also exists, which is a minimum value of trust that the peers in a list need to have, so that their warnings are taken into consideration. Peers that are below the threshold are marked with a probation flag and have a certain probation period to pass the threshold. If the peer manages to pass the threshold in time, the flag is removed. If not, the peer is removed from the acquaintance list and some new randomly chosen peer will take its place. The new peer will also be flagged and given a probation period to pass the threshold. If it does not, then the aforesaid procedure will take place. This means that for every peer P_j in the list, P_i has a probation flag pf_{ij} that shows if P_j is flagged or not, and a probation time pt_{ij} that shows the time passed since P_j was flagged.

To ensure that the acquaintance list is dynamically built (and managed), making sure that only trustworthy peers remain in the list, four different cases can be distinguished:

– If an attack occurred and P_j sent a warning then $s_{ij} = s_{ij} + 1$, $u_{ij} = u_{ij}$ and $pt_{ij} = pt_{ij} + 1$.

- If an attack took place and P_j did not send a warning then $s_{ij} = s_{ij}$, $u_{ij} = u_{ij} + 1$ and $pt_{ij} = pt_{ij} + 1$.
- If no attack occurred but P_j sent a warning then $s_{ij} = s_{ij}$, $u_{ij} = u_{ij} + 1$ and $pt_{ij} = pt_{ij} + 1$.
- If no attack took place and P_j did not send any warning then s_{ij}, u_{ij} and pt_{ij} remain the same.

Alert Correlation. A peer can utilize the knowledge of the trustworthiness of others to perform alert correlation. The confidence level of a correlated alert is computed as follows:

$$C_i = w_{dir} \cdot c_i + w_{ind} \cdot \frac{1}{N} \sum_{j=1}^{N} c_j \cdot t_{ij} \tag{10}$$

Here, c_i is the confidence in the correlated alert as correlated by P_i, c_j is the confidence of the alert received from peer P_j, and N is the number of peers that have not been flagged and have sent alerts used in the correlation. w_{dir} and w_{ind} are the direct and indirect weights (direct for locally generated alerts and indirect for received alerts). Regarding N it is required to be above a certain threshold N_{min}, and if it is lower, then the weight w_{ind} is decreased by N/N_{min}. Hence, dependence on only a low number of peers is avoided. However, even with a high N_{min} problems might also appear, when the number of truthful warnings available for correlation is very small. In the end, if the confidence of a certain correlated alert is above a certain threshold, the peer will activate the incidence response module which will take passive or active action towards the threat.

Experiments and Results. The authors conducted experiments for a virtual network (consisting of 36 clients) that was being attacked by a worm. The clients were grouped in 6 sub-networks. A survival rate was defined, as the number of nodes resisting the worm divided by the number of all nodes in the network. The survival rates for the case when the clients were part of the CIDS and when they were not, were compared. The results showed a significant increase of the average survival rate when the CIDS was utilized. According to the experiments, the more peers are in the CIDS the higher the probability is that the worm will be detected. On the one hand, the survival rate decreases by increasing the number of trusted peers needed for correlation (N_{min}). On the other hand, the resilience of the network increases with N_{min} as the impact of a malicious peers is diminished (which also means that the false alarm rate decreases). Thus, by configuring N_{min} one of the following can be achieved: either a faster detection system with a higher survival rate but prone to false alarms, or a more robust system with a lower level of false alarms but which (due to lack of trust) could miss some of the real alarms.

3.4 A Reputation-Based Bootstrapping Mechanism for CIDSs

Perez et al. [8] proposed a reputation management system that addresses the newcomer (bootstrapping) problem in CIDSs. Bootstrapping is a common issue in P2P networks where newcomers join the networks for the first time. Similarly, reputation bootstrapping is a common issue in reputation systems.

System Model. The CIDS is divided into *security domains* (D_1,D_2,..., D_n), each of which defines a *Collaborative Intrusion Detection Network* (CIDN) [2]. Each security domain is composed of a multitude of IDSs, one of which is chosen to be its leader (*Domain Leader* (DL)) acting as the representative of the CIDN in the CIDS. The DL's purpose is to share alerts detected by its CIDN with the CIDNs of other domains, and request recommendations from other CIDNs about a newcomer to compute its initial reputation (trust) score [7]. The newcomers can be *static* IDSs (permanently placed), *mobile* IDSs belonging to users who want to collaborate with security domains, or a *security domain* that wants to improve its accuracy in detecting distributed threats by exchanging alert data. Trust is computed on the basis of the initial absence of historical data and on the fact that IDSs join and leave the system regularly.

Reputation Management System. This section describes the reputation management system focusing on the aforesaid three possible newcomers. Note that the model also depends on the *detection skills*, that is the usefulness and willingness of a newcomer, as well as the similarity between two domains. The usefulness function of a newcomer's detection unit (DU_m) is denoted as $\Phi_x(DU_m)$ and its computation can be found in [8]. Similarly, the computations regarding the willingness of the new detection unit which is expressed as $\omega_x(DU_m)$, and the similarity between two domains (D_x and D_y) which is denoted as $\lambda(D_x, D_y)$ can also be found in [8].

Recommendations from the CIDS about a newcomer. Whenever a newcomer (NC) (either a mobile IDS or a security domain), joins a domain D_x, the latter can query other trusted domains within the CIDS asking for recommendations regarding the newcomer. The aim is to find the most reputable path leading to the most trustworthy domain having recommendations about the NC's behavior in sharing alerts. The best trust path (T_{tp_i}) built up to the domain D_x, which maximizes the confidence that D_x can have on the most trustworthy domain D_y (which will return its recommendation $Rec_y(NC)$), is computed as:

$$T_{tp_i}(D_x, D_y) = \frac{1}{|tp_i|} \cdot \sum_{D_f, D_k \in tp_i}^{|tp_i|} \frac{T(D_j, D_k) \cdot (\frac{1}{\Delta t+1}) \cdot \delta(D_j, D_k)}{|tp_i|} \tag{11}$$

and

$$\delta(D_j, D_k) = \begin{cases} 1, \text{if } D_j, D_k \in AD_z \\ \epsilon_j \in [0, 1], \text{otherwise} \end{cases}$$

where (D_j, D_k) represents each consecutive pair of domains in the trust path tp_i, $|tp_i|$ is the length of such a trust path built up to D_x and Δt expresses the amount of time elapsed since the last interaction between D_j and D_k. Essentially, this computes a weighted average of the direct trust values of each subsequent pair of domains, $\mathrm{T}(D_j,D_k)$, along trust path tp_i. If NC is a *mobile* IDS joining the domain D_x, $mIDS_i$, this D_x can also query other mobile IDSs that are currently collaborating with D_x ($mIDS_j$s), about their recommendations on $mIDS_i$. Thus, the final recommendation for $mIDS_i$ is:

$$Rec_{mIDS}(D_x, mIDS_i) = \sum_{mIDS_j \in D_x} \frac{Rec_{mIDS_j}(mIDS_i) \cdot (\frac{1}{\Delta t+1})}{|mIDS_j \in D_x|} \tag{12}$$

Moreover, for computing the confidence that D_x has, on a recommendation gathered from the CIDS, the following formula can be used:

$$T_{mIDS}(D_x, mIDS_i) = \sum_{mIDS_j \in D_x} \frac{Rep_{D_x}(mIDS_j)}{|mIDS_j \in D_x|} \tag{13}$$

Finally, if it is required to compute a single recommendation score by taking into consideration all the recommendations gathered from those sources that maintain behavioral-based information about the newcomer, then the following equation is used:

$$Rec_{CIDS,mIDS}(D_x, NC) = \theta_{tp_i} \cdot Rec_{CIDS}(NC) + \theta_{mIDS} \cdot Rec_{mIDS}(D_x, NC) \tag{14}$$

where θ_{tp_i} and θ_{mIDS} are the trust that D_x has on the domains providing the NC's recommendation score and those provided by the mobile IDSs currently collaborating with D_x. The equations computing them can both be found in the original paper [8]. The proposal for bootstrapping the reputation of a newcomer in a CIDN will be presented bellow, by distinguishing three cases: the reputation bootstrapping model for a *static* IDS, a *mobile* IDS, or a *new security domain*.

Static IDS: The proposed equation to compute the initial reputation score of a newcomer static IDS, namely $sIDS_i$, when it joins the domain D_x at time t is:

$$Rep^{(t)}_{D_x}(sIDS_i) = (\frac{1}{\Delta t+1}) \cdot Rep^{\Delta t}_{D_x}(sIDS_i) + (\frac{\Delta t}{\Delta t+1}) \cdot \Phi_{D_x}(sIDS_i)^{\tau(sIDS_i)} \tag{15}$$

where Δt represents the time elapsed since the last time $sIDS_i$ participated in D_x, $Rep^{\Delta t}_{D_x}(sIDS_i)$ indicates the last $sIDS_i$'s reputation score that D_x has stored, and $\Phi_{D_x}(sIDS_i)$ is the usefulness of $sIDS_i$ from the perspective of D_x.

Mobile IDS: The proposed equation to compute the initial reputation score of a newcomer mobile IDS, namely $mIDS_i$, when it joins the domain D_x at time t is (a formal definition for f'_m can be found in [8]):

$$Rep^{(t)}_{D_x}(mIDS_i) = (\frac{1}{\Delta t+1}) \cdot Rep^{\Delta t}_{D_x}(mIDS_i) + (\frac{\Delta t}{\Delta t+1}) \cdot f'_m(\Phi, \omega, \tau, Rec_{CIDS,mIDS}) \tag{16}$$

Security Domain: The proposed equation to compute the initial reputation score of a newcomer security domain D_y, that wishes to collaborate with the domain D_x at time t is (a formal definition for f'_d can be found in [8]):

$$Rep^{(t)}_{D_x}(D_y) = (\frac{1}{\Delta t + 1}) \cdot Rep^{\Delta t}_{D_x}(D_y) + (\frac{\Delta t}{\Delta t + 1}) \cdot f'_d(\lambda, \tau, Rec_{CIDS}) \quad (17)$$

Experiments and Results. The authors firstly examined the benefits of including mobile IDSs in the system. Their results suggest an improvement of the detection capabilities required by a CIDN to detect distributed threats. Further testing showed that the reputation bootstrapping model can support around 20% of malicious mobile IDSs before being discarded as valuable detection units. In addition, Perez et al. analyzed the variance of the reputation scores of static and mobile IDSs over time with regard to compromise and misbehavior. It was found that reputation scores are rapidly decremented when there are less than 5% of malicious IDSs. This finding was interesting for mobile IDSs, as they follow a similar pattern with static IDSs although mobile IDSs reputation is computed in each movement across the domains. This accuracy is due to the use of recommendations provided by other trusted parties of the CIDS. Further testing showed that this reputation bootstrapping model maintains its robustness for up to around 20% of malicious IDSs without losing its detection accuracy.

4 Discussion

This section begins with a comparison overview of the surveyed approaches with respect to fulfillment of the requirements discussed in Sect. 2. This comparison is also summarized in Table 1. Subsequently, we discuss the lessons learned from the current status of the state of the art and future directions that will advance the intersection of trust and CIDSs.

Table 1. Comparison of surveyed systems with the proposed requirements

Requirement	Dirichlet-based approach [4]	Trust diversity [9]	Trust-aware CIDS [1]	Bootstraping approach [8]
Global view	✖	✔	✖	✖
Overhead	●●○	●●●	●○○	●●●
Incentive	✔	✖	✖	✖
Initial trust	✖	✖	✖	✔
Forgetting factor	✔	✔	✖	✔
Performance history	✔	✔	✔	✔

4.1 Comparison

In the following we discuss each requirement and how the different approaches fulfill it or not.

- **Global view:** The first [4] and third [1] surveyed systems do not require global knowledge as each node computes its peers' trust values independently. In the second approach [9], a global view is required as sensors are moved between domains to increase trust diversity. This implies that an administrator who can control where the sensors are deployed is required. For the last approach [8], the entire system is divided into security domains. For each domain, the domain leader must have knowledge of the topology and behavior of the nodes in its domains. For this reason, only partial view of the network is required by domain leaders.
- **Minimum overhead:** The first approach [4] utilizes test messages, which increase the communication overhead. In fact, the confidence of the trust value is dependent to the number of the sent test messages. The second approach [9] also depends on the sensors being assigned trust values according to their past behavior which also requires the dispatching of test messages. Moreover, the overhead of re-configuring sensors to increase trust diversity also has to be considered. The third approach [1] has lower overhead than the previous ones as it does not require test messages for the computation of trust values. Instead it only uses the knowledge gained from past interactions to calculate trust values. The last approach [8] has significant overhead, due to the number of steps a newcomer has to take when joining the CIDS.
- **Incentive mechanism:** In the first approach [4], an incentive mechanism is proposed where nodes give more feedback to trustworthy peers while not giving as much to untrustworthy peers. This incentive mechanism is important as it ensures that peers which behave in a trustworthy manner are rewarded while untrustworthy peers are ignored. This also reduces overhead communication and computation overhead as peers do not spend time responding to untrustworthy peers. The rest of the approaches, however, do not make use of any incentive mechanism.
- **Initial trust:** The first system [4] assigns the newcomers with random trust values. Nevertheless, the CIDS mitigates the risk by placing newcomer nodes in a probation list for a certain period of time. The second approach [9] also assigns completely random values and takes no steps to mitigate the risk associated with this scheme. The third system [1] keeps newcomers under probation for a fixed period of time. The newcomer is not assigned an initial trust value. After the probation time period has elapsed, if the newcomer's trust value is above a certain threshold, it is considered trustworthy. The last approach [8] exhibits the most sophisticated approach for assigning trust values to newcomers. It takes into account many factors and gathers background information about nodes to solve the newcomer problem.
- **Forgetting factor:** The first [4], the second [9] and the fourth [8] approaches, all make of use of forgetting factors in the calculation of trust, so that more

recent messages are given more weight. However, the third approach [1], does not make use of such a technique.
– **Performance history:** All of the surveyed approaches make use of historical data, in different ways, during the trust calculations.

4.2 Challenges and Steps Ahead

The analysis of the state of the art suggests a plethora of novel ways for managing trust. As we have already described in the previous section, each of the aforesaid mechanisms introduces various advantages. Combining the benefits of each approach is one way towards the fulfillment of the requirements. In addition, we argue that a common methodology for the evaluation of trust in CIDSs is required. A basis for this can be the requirements proposed in Sect. 2.

Furthermore, we envision the following research questions for which we argue that, when answered properly, can improve the output quality of CIDSs:

– Which additional *parameters*, inside the alert data, can be utilized for the computation of trust?
– Is it possible to include more *social* attributes/parameters inside the overall trust calculations? For example, it might be that two organizations, inside the CIDS, have some special long-term connection that cannot easily depicted formally.
– How important is the *timeliness* of the exchanged alert data? It can be that some sensors in a CIDS do not publish their alerts immediately due to internal security policies or due to the need for anonymization of the data.
– How important is the *relevance* of the received alert data for a sensor? Can it be that a sensor receives valid data from a highly trustworthy sensor which however are irrelevant? For instance, what about an organization which obtains information about a port-specific attack, which however is completely banned from that particular organization's network.
– How can *uncertainty* be included in the trust model? Would it be of benefit to include *certainty* scores for the trust values? For instance, an approach used in [5] can consider the volume of data utilized for generating a trust score; when sufficient data is not available, the certainty score would be low.

5 Conclusion

With the continuous growth in both the numbers and the sophistication of cyber-attacks, CIDSs are becoming increasingly important. Introducing computational trust techniques in the field of CIDSs can provide substantial benefits for the detection of insider attacks as well as for the creation of highly tailored threat awareness of the monitored network. We proposed requirements for TM in the context of CIDSs. Moreover, we analyzed the most prominent systems with a focus on how they calculate and manage trust in such a context. The paper also provides an overall discussion of the requirements, with respect to their fulfillment in the related work, and highlights the research challenges and gaps that need to be tackled in the future.

Acknowledgments. This work has received funding from the European Union's Horizon 2020 Research and Innovation Program, PROTECTIVE, under Grant Agreement No 700071.

References

1. Duma, C., Karresand, M., Shahmehri, N., Caronni, G.: A trust-aware, P2P-based overlay for intrusion detection. In: 17th International Workshop on Database and Expert Systems Applications, DEXA 2006, September 2006
2. Fung, C., Zhang, J., Aib, I., Boutaba, R.: Trust management and admission control for host-based collaborative intrusion detection. J. Netw. Syst. Manag. **19**, 257–277 (2011)
3. Fung, C.: Collaborative intrusion detection networks and insider attacks. J. Wireless Mob. Netw. Ubiquit. Comput. Dependable Appl. **2**(1), 63–74 (2011)
4. Fung, C.J., Zhang, J., Aib, I., Boutaba, R.: Dirichlet-based trust management for effective collaborative intrusion detection networks. IEEE Trans. Netw. Serv. Manag. **8**(2), 79–91 (2011)
5. Habib, S.M., Volk, F., Hauke, S., Mühlhäuser, M.: Computational trust methods for security quantification in the cloud ecosystem. In: The Cloud Security Ecosystem - Technical, Legal, Business and Management Issues, pp. 463–493. Syngress (2015)
6. Mitchell, R., Chen, I.R.: A survey of intrusion detection techniques for cyber-physical systems. ACM Comput. Surv. (CSUR) **46**(4), 55 (2014)
7. Ortega, F.J., Troyano, J.A., Cruz, F.L., Vallejo, C.G., Enríquez, F.: Propagation of trust and distrust for the detection of trolls in a social network. Comput. Netw. **56**(12), 2884–2895 (2012)
8. Pérez, M.G., Mármol, F.G., Pérez, G.M., Skarmeta Gómez, A.F.: Building a reputation-based bootstrapping mechanism for newcomers in collaborative alert systems. J. Comput. Syst. Sci. **80**, 571–590 (2014)
9. Pérez, M.G., Tapiador, J.E., Clark, J.A., Pérez, G.M., Skarmeta Gómez, A.F.: Trustworthy placements: Improving quality and resilience in collaborative attack detection. Comput. Netw. **58**, 70–86 (2014)
10. Vasilomanolakis, E., Karuppayah, S., Mühlhäuser, M., Fischer, M.: Taxonomy and survey of collaborative intrusion detection. ACM Comput. Surv. **47**(4), 33 (2015)
11. Yu, B., Singh, M.: Detecting deception in reputation management. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (2003)