Trustworthy Placements: Improving Quality and Resilience in Collaborative Attack Detection

Manuel Gil Pérez[a,*], Juan E. Tapiador[b], John A. Clark[c], Gregorio Martínez Pérez[a], Antonio F. Skarmeta Gómez[a]

[a] *Departamento de Ingeniería de la Información y las Comunicaciones, University of Murcia, 30071 Murcia, Spain*
[b] *Computer Science Department, Carlos III University of Madrid, 28911 Leganés, Madrid, Spain*
[c] *Department of Computer Science, University of York, York YO10 5GH, United Kingdom*

Abstract

In distributed and collaborative attack detection systems decisions are made on the basis of the events reported by many sensors, e.g., Intrusion Detection Systems placed across various network locations. In some cases such events originate at locations over which we have little control, for example because they belong to an organisation that shares information with us. Blindly accepting such reports as real encompasses several risks, as sensors might be dishonest, unreliable or simply compromised. In these situations trust plays an important role in deciding whether alerts should be believed or not. In this work we present an approach to maximise the quality of the information gathered in such systems and the resilience against dishonest behaviours. We introduce the notion of *trust diversity* amongst sensors and argue that detection configurations with such a property perform much better in many respects. Using reputation as a proxy for trust, we introduce an adaptive scheme to dynamically reconfigure the network of detection sensors. Experiments confirm an overall increase both in detection quality and resilience against compromise and misbehaviour.

Keywords: Trust diversity, reputation, adaptive self-configuration, detection quality, collaborative attack detection

1. Introduction

Many security threats in current computing environments can only be detected by gathering and correlating evidence obtained at different locations [1, 2].

*Corresponding author. Phone: +34 868 887645, Fax: +34 868 884151.
Email addresses: mgilperez@um.es (Manuel Gil Pérez), jestevez@inf.uc3m.es (Juan E. Tapiador), jac@cs.york.ac.uk (John A. Clark), gregorio@um.es (Gregorio Martínez Pérez), skarmeta@um.es (Antonio F. Skarmeta Gómez)

In some cases, evidence may come from sources over which we have little control. This is, for example, the case when organisations choose to share information about detected security events. In other cases, the integrity of the source may be questioned, for example if there is evidence that it may be malfunctioning, exhibiting a dishonest behaviour, or simply compromised.

Whatever the case, gaining such evidence in a prompt and timely manner is essential for minimising risk exposure. A major challenge in such Collaborative Intrusion Detection Networks (CIDN) [3, 4] is the ability to properly assess how much trust we can place in each piece of information and what risks we incur by believing or not believing it. Blindly accepting each report as truth is certainly dangerous. Consider, for example, an intelligent adversary who, after compromising a few detection sensors, forces them to stop sending alerts related to his intended malicious activities. Conversely, sensors may report too many false alerts, either to undermine our confidence in them or divert attention from a more serious event taking place. Ultimately, deciding what to believe depends on available information about the source and its operational context. This topic has received too little attention so far, despite being crucial for a proper functioning of collaborative detection efforts where there is some degree of distrust amongst parties or regarding the resilience of the sensors against attack.

In this paper, we propose a scheme aimed at increasing the *quality* of the decisions made about pieces of evidence in which we place different degrees of trust. In our proposal, we first quantify the confidence we place in sensors behaviour according to how they have behaved in the past, normally referred to as reputation [5]. However, the use of reputation alone has shortcomings, and systems based solely on assessments of past behaviour have severe limitations. We address this issue by introducing the notion of *trust diversity*. In essence, trust diversity measures the dispersion among the trust values of a population of detection sensors placed in a given domain, with low diversity values indicating that sensors have similar trustworthiness, and vice versa.

We then propose to quantify the quality of a particular sensor placement through its trust diversity, and to dynamically search for high quality placements. Informally speaking, a high quality placement is one where trusted sensors are deployed in the vicinity of others we have doubts about. Similarly, low quality placements are those where all sensors have roughly the same trustworthiness. We pursue two main goals with the use of trust diversity as a measure of placement quality:

1. Firstly, by simultaneously maximising trust diversity across all network domains we guarantee that no domain is left poorly protected (i.e., monitored exclusively by untrusted sensors).
2. Secondly, since each domain will have sensors with varying trust values, the most reliable among them can contribute to the assessment of the reputation of others, for example by identifying those that behave differently when presented with the same events.

The overall result is that sensor placements with a sufficient amount of trust

diversity exhibit two interesting properties: (i) they facilitate the early identification of misbehaving sensors, a fact that implicitly contributes to a better assessment on the truth of the events they generate; and (ii) they are more resilient to compromise by external attackers.

In our scheme, trust diversity is used to dynamically place sensors when and where they are most needed, both to lessen uncertainty about what is actually happening in the network infrastructure and also to make defences more resilient to threats. Assume, for example, that contradictory events are reported by sensors in the same network area. If our confidence in all these sources is similar, we face a problem when deciding what is actually going on. Even if at present time there is little we can do about this, a careful deployment of alternative or additional sensors will help to resolve such conflicts in the future and, indirectly, to re-assess our confidence on the sensors currently deployed there.

As we discuss later, such re-deployments (or re-configurations of the monitoring infrastructure) can be triggered under many circumstances, posssibly in a fully automated reconfigation. This allows defences to react to the presence of uncertain information by seeking ways of improving their function. We believe this is a very interesting property for adaptive security systems and a prerequisite for self-healing networks.

The remainder of this paper is organised as follows. Section 2 presents some definitions and outlines our system model. Section 3 describes the adaptive model designed to reduce the uncertainty arising from discrepancies. Section 4 introduces both the reputation system used to assess sensors' behaviour and the quantification of trust diversity. Section 5 reports some experimental results to illustrate how the system can obtain better evidence by maximising trust diversity. Section 6 discusses related work in this area and, finally, Section 7 summarises our contributions and identifies future research directions.

2. Definitions and system model

Any *information system* (IS), regardless of the services it offers, can be modelled on the basis of all the elements tha make up its underlying network. In this sense, the information system administrator can internally structure services and resources within a well-defined set of *domains* (D). Such a grouping may be based on the site's security policy and/or some modelling of the services, e.g., with respect to their type, so as to allow a seamless scalability as the number of services increases [6]. Governance functions could also be expected to monitor the proper operation of all services deployed in the information system, thereby defining a *monitoring system* as a surveillance centre.

Each domain imposes a set of *requirements* (R), or liabilities, for the proper operation of its services and, consequently, for the proper operation of the entire information system. As these requirements are given by the needs of each service, each domain can automatically extract its requirements from those of the services it contains. Requirements may differ in importance and so, consequently, may their monitoring. The administrator should provide a weight for

each requirement $R_k \in R$, $Imp(R_k) \in [0,1]$, indicating the impact or importance on the information system if R_k is compromised.

Using the notation defined earlier, we formally define an information system with a total number of u requirements that can be demanded by m domains as follows:

$$\begin{aligned} IS &= \{D_1, D_2, ..., D_m\} \\ R &= \{R_1, R_2, ..., R_u\} \\ R(D_i) &= \{R_{i_1}, R_{i_2}, ..., R_{i_x}\} \end{aligned}$$

where each domain $D_i \in IS$ $(1 \leq i \leq m)$ is defined in accordance with its needs with x requirements $(x \leq u)$ for the proper operation of its services.

An example of a generic information system is depicted in Figure 1. Note that this figure includes some other concepts not yet defined, belonging to the monitoring system, that will be formally introduced later.

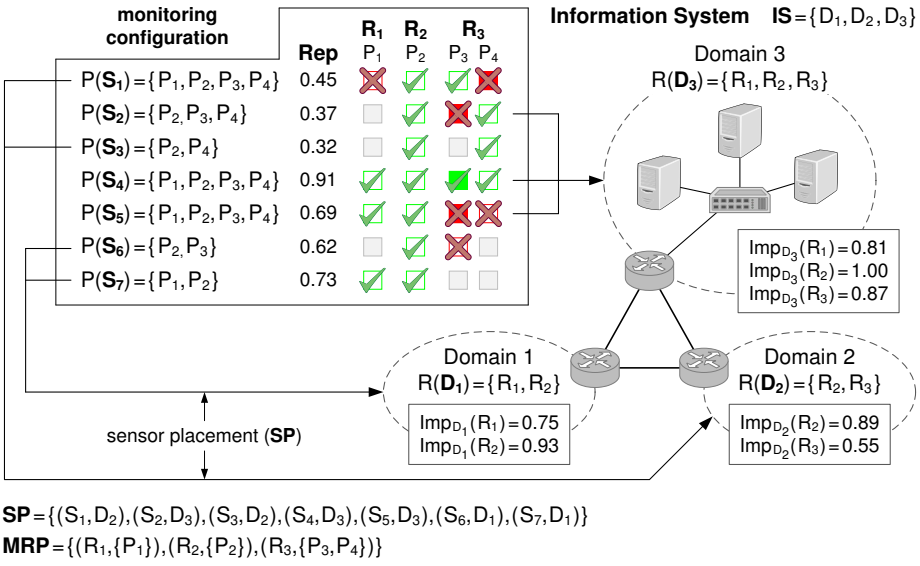

Figure 1: Example of a monitoring configuration for a generic information system

This information system illustrates a distribution of the services in three domains, $IS = \{D_1, D_2, D_3\}$. The information system administrator has defined three requirements, $R = \{R_1, R_2, R_3\}$, that have to be monitored to determine the proper operation of the system, with $R(D_1) = \{R_1, R_2\}$, $R(D_2) = \{R_2, R_3\}$ and $R(D_3) = \{R_1, R_2, R_3\}$. In turn, each domain specifies the impact associated (Imp) with each requirement if these are not satisfied.

For the detection of security events, we assume that the information system has at its disposal a limited set S of n *sensors*, with $n \gg m$ so as to have a great variety of monitoring points. Such sensors will be configured and deployed into domains in order to check if services are properly working and if they are under

attack. Thus, their main activity is to send events –alerts, notifications– upon malfunctioning or attack attempts, covering both *dependability* and *security* functions (i.e., to ensure that users can make use of all services offered, and that this is done securely).

Each sensor is characterised by a set of *properties*, or capabilities, that it can offer to monitor requirements. For example, an application-layer Intrusion Detection System (IDS) might be capable of detecting HTTP-based attacks only, while others could detect threats based on network and transport protocols. For each sensor $S_j \in S$ a reputation value is also maintained, $Rep(S_j) \in [0, 1]$, that models its behaviour according to the assessment of all the information sent (or not) by the sensor so far. The higher the reputation, the higher the belief that the information provided by the sensor is true. Section 4.3 further discusses how to compute and update this reputation.

We now formally define the set of v supported properties that the n sensors can provide to the information system as follows:

$$\begin{aligned} S &= \{S_1, S_2, ..., S_n\} \\ P &= \{P_1, P_2, ..., P_v\} \\ P(S_j) &= \{P_{j_1}, P_{j_1}, ..., P_{j_y}\} \end{aligned}$$

where each sensor $S_j \in S$ $(1 \leq j \leq n)$ can be modelled according to the y properties $(y \leq v)$ it supports at a given time to monitor some requirements.

In Figure 1, seven sensors for monitoring purposes have been defined, i.e., $S = \{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$, including the definition of their properties extracted from the total set of properties defined by the system administrator as $P = \{P_1, P_2, P_3, P_4\}$. Note that sensors are configured by activating or deactivating the specified properties as the system considers appropriate.

We can notice in Figure 1 that there is a direct relationship between each property offered by a sensor and the monitoring requirements demanded by the system, which has also been included in Figure 1. The *mapping amongst requirements and properties* (MRP) defines which sensor properties can monitor the proper operation of a requirement:

$$MRP = \{(R_k, P_w) \mid R_k \Leftrightarrow P_w,\ P_w = \{P_1, P_2, ..., P_l\}\}$$

where $P_w \subseteq P$ $(1 \leq l \leq v)$ is the set of properties needed to satisfy the k-th requirement, with $R_k \in R$ $(1 \leq k \leq u)$.

In order to illustrate the use of the concepts introduced above, consider the following scenario where a collaborative intrusion detection network will be deployed in a system:

- A *domain* is defined by each network segment in the information system.
- A *requirement* corresponds to an attack class that each service deployed in a network may suffer, for example a privilege escalation attack or a denial of service (DoS) attempt.

- A *sensor* is an IDS capable of analysing the traffic that flows through a network segment –domain– or the activity produced in a host.
- *Properties* are particular configurations of an IDS to detect attack classes, depending on its detection techniques. These can be either signature-based detection rules (*misuse detection*) or policies that model normal system behaviour (*anomaly detection*). In both cases, a set of rules –properties– are used to monitor some of the demanded attack types and to provide alerts about the compromise of a requirement.

2.1. Monitoring system

As sensors are services too, with a special purpose in the information system, their operation needs also to be supervised to check if they are behaving properly and according to their claimed capabilities [7]. Otherwise, a malfunction of any of them (or, even worse, the reporting of wrong information due to malicious behaviour) can compromise the perceived situation about the system security state. The dangers of undetected attacks are well known. Alternatively, the incorrect suspicion that there is an ongoing security breach may trigger response mechanisms that will interfere with the normal operation of the system.

Apart from simple malfunctioning, which can be also regarded to as unintentional but improper behaviour, any sensor might maliciously misbehave, for example after being compromised by an adversary. Sensors may thus feed the system with bad (bogus) information, or fail to report something that has actually happened. We distinguish both cases from the usual *false positives* and *false negatives* [8], as they clearly have an intentional nature. These two cases are widely explained in Section 2.2.

The information system, guided by its monitoring needs, can configure and deploy sensors at will in different domains to reach the desired goal. This requires having a placement model where sensors are configured and deployed in a given domain with the properties enabled for each of them (obviously, each sensor can only be configured with the properties that it supports). Thus, a *sensor placement* (SP) is given by a mapping:

$$SP = \{(S_j, D_i) \mid S_j \rightarrow D_i\}$$

where the j-th sensor, $S_j \in S$ $(1 \leq j \leq n)$, is deployed in the i-th domain, $D_i \in D$ $(1 \leq i \leq m)$.

In summary, a monitoring system is defined by making use of: the information system components (i.e., IS, R and $R(D_i)$); its specific sets of monitoring components S and P; the set of relationships amongst requirements and properties (MRP); and the sensor placement (SP). Establishing which properties of a sensor are used to satisfy some of the requirements demanded by the domain where it is placed, determined by $P(S_j)$, together with all previous information about the monitoring system, defines a possible *monitoring configuration* that the monitoring system can enforce in the underlying network.

Following the example of Figure 1, two relationships of a requirement to a single property have been defined for simplicity, although R_3 includes two

properties to be monitored, P_3 and P_4, for completeness. Thus, the monitoring configuration defines the following sensor placement: $\{S_6, S_7\} \rightarrow D_1$; $\{S_1, S_3\} \rightarrow D_2$; and $\{S_2, S_4, S_5\} \rightarrow D_3$. As a real example of these components, in the collaborative intrusion detection network scenario introduced above, R_3 could be a critical *network trojan* attack class through which attackers can gain unauthorised privileges in the victim to execute any desired remote code on it. This attack can be achieved by exploiting new vulnerabilities against the Java Runtime Environment (JRE), discovered in recent Oracle Java SE versions. For example, by launching a remote DoS attack to cause the Java Virtual Machine to crash or bypass Java sandbox restrictions (vulnerability identifier: CVE-2012-0507), or by exploiting its successor (vulnerability identifier: CVE-2012-1723) to reach the same objectives. The latter vulnerability can be easily exploited by using Black Hole, one of the most widely used exploit toolkit amongst hackers nowadays. Thus, IDSs must be configured to detect both vulnerabilities, which can be identified as P_3 and P_4 in the example shown in Figure 1. Note that Snort is capable of detecting both vulnerabilities [9].

2.2. Sensor behaviour

Sensors can exhibit malicious behaviour in providing information about the monitoring of some requirements for which they are responsible. Table 1 shows a classification of the types of event that sensors can report.

Type	**Description**	**Behaviour**
True positive	The information has been properly classified, as it is truthful and the event has actually happened	Honest
True negative	There is no evidence to offer any information, and the sensor has not properly sent anything	Honest
False positive (FP)	It is a real false positive due to a limitation or an internal failure in the sensor's monitoring algorithms	Honest (**HFP**)
	The sensor has reported on an event that it has not actually occurred in the information system	Malicious (**MFP**)
False negative (FN)	The sensor has not reported about a real event, but it is beyond its detection capabilities to do so	Honest (**HFN**)
	Intentional denial of information, as the event has occurred and the sensor is configured to detect it	Malicious (**MFN**)

Table 1: Possible sensor behaviour

A true event, either a *true positive* or a *true negative*, indicates a good functioning of the sensor, with respect to both the proper way of monitoring and the behaviour exhibited by the sensor. Instead, a false event can take different meanings depending on why it has been produced. On the one hand, a *false positive* is an alert generated by one or more sensors that does not actually correspond to an existing threat attempt. This situation can be due

to an inherent limitation in the sensor's capabilities, or perhaps a failure in the detection algorithms (e.g., because of too much data to be analysed or detection rules at an inappropriate level of specificity required to detect more or less threats [10]). This type of alert is classified as an *Honest False Positive* (HFP), since it is a detection error but the sensor exhibits an honest behaviour.

Quite a different type of false positive is the one generated by sensors with a malicious purpose. We refer to these as *Malicious False Positives* (MFP). Dishonest or compromised sensors may try to disturb the proper operation of the monitoring system in order to cause intended errors or generate confusion. For example, a goal pursued by generating massive numbers of MFPs can be to inject some *noise* to capture the attention of the administrators while camouflaging a more important attack. Another potential objective sought by generating MFPs is to misguide the attribution of an attack; that is, the deliberate intent of accusing a legitimate entity of doing something that it did not do. Malicious sensors can generate critical events from scratch using some user's IP address as the threat source. As a consequence, the monitoring system may trigger some responses to fix the *presumed* damage caused by such a user (e.g., filtering his IP address to deny him further access to the network). If such a user happens to be another sensor, the system may well choose to disable it after considering that it is not properly working, thus facilitating subsequent attacks.

A *false negative* is a type of event that has not been reported by the sensors but it actually corresponds to a threat attempt. For example, this may occur due to an *evasion attack*, where the malicious entity exploits some weakness in the particular detection technology used by the sensor so that it cannot recognise an ongoing attack [11]. Alternatively, it may simply happen that the sensor does not have the proper capabilities to detect such an attack (e.g., an IDS missing the required signatures). In either case, both correspond to *Honest False Negatives* (HFN). Again, an entirely different situation occurs when the sensor is properly configured to detect an event but fails to do so due to malicious behaviour. We refer to such cases as *Malicious False Negatives* (MFN).

As an example of these behaviours, let us suppose that the information system of Figure 1 receives an event about an attack related to the compromise of R_2. All sensors are configured to monitor R_2 (with P_2 activated). If all sensors report it, this event can be considered as truth; that is, it is a true positive and not an MFP. Otherwise, if there is a disagreement, e.g., some sensors report the event while the rest do not, the following two situations can arise when assessing the event:

- It is classified as a true positive. This implies that the sensors that did not report on the event have misbehaved by generating an MFN; or
- Some sensors have reported about a situation that has not actually happened, thus exhibiting an improper behaviour by producing an MFP. The remaining sensors do exhibit a good behaviour, thereby classifying the event as a true negative.

Both alternatives, each in a separate way, can be true. The final decision

about which one is actually true is difficult to make. In this work, we base such decisions on the *reputation* of the sensors, this being a measure of how they behaved in the past. Furthermore, other factors will also be taken into consideration, such as the trust diversity (defined later) currently existing in the domain and the potential impact of the presumed attack. Note also that sensors can "misbehave" due to inherent limitations in their monitoring algorithms, thus producing an honest mistake (i.e., an HFN in the first case and an HFP in the latter). In both cases, sensors will be accused of malicious behaviour and their reputation will be unfairly decreased. As a possible solution, the monitoring system may assume sensors' internal failures if all sensors configured with the same detection software behave the same, provided that the population of sensors is sufficiently high.

Assume now that an event originating at D_3 in Figure 1 has been sent by one, or two, of the three sensors deployed in such a domain. If only S_4 has sent the event, the system may consider that it is believable because the sensor has a high reputation: $Rep(S_4) = 0.91$. Instead, the event may be considered as an MFP if it is sent by S_2 *only*, whose reputation is quite low: $Rep(S_2) = 0.37$. Alternatively, if two out of the three sensors send the event, the decision can be made according to the aggregation value obtained from the reputation of both sensors. In this case, the confidence assigned to such an event E –compromise of R_2– might be $T(E_{R_2}) = 0.8$, if the event has been sent by S_4 and S_5. This means that the event may be considered believable because $T(E_{R_2}) \geq 0.5$, provided that a threshold of 0.5 is set. Both sensors have exposed an honest behaviour, while the lack of an alert from S_2 is classified as an MFN.

Now, if the event was sent by S_2 and S_5, the confidence that the system can place in it is $T(E_{R_2}) = 0.53$. As in the previous case, the event is trusted because $T(E_{R_2}) \geq 0.5$. Nevertheless, there are other sensors in the same domain that did not send the event, despite having a reputation higher than the two sensors that did it. In this case, we need to take into account other factors when assessing this situation. In this work, we propose to use a measure of the *trust diversity* in the domain. Roughly speaking, this is a measure of the variability in trust values of the deployed sensors. Consider, for example, that the event comes from D_1. As both sensors have similar trust values, the trust diversity in such a domain is quite low. Thus, it is reasonable to expect that both sensors behave similarly. If only one of them reports the event, the system will not be able to discern what is going on. Although each sensor's reputation has a medium/high value ($Rep(S_6) = 0.64$ and $Rep(S_7) = 0.73$), it will not be easy to determine if it is an MFN or not. Subsequently in this paper, we suggest to reconfigure the system with a new sensor placement that increases the trust diversity while ensuring that all requirements are sufficiently covered.

3. Adaptive model: dynamically reconfiguring sensors

This section describes an adaptive model aimed at lessening potential uncertainty in the events received from the monitoring system. One of the major goals pursued here is to improve the quality of the decision-making process by

correctly determining the truth of the information reported by sensors. Furthermore, the conclusions drawn from this analysis can be also used to re-assess our trust in sensors' behaviour, notably by using a reputation system. However, as discussed in the example above, in some situations the only way to increase our discerning power is to modify the location and/or configuration of some sensors to ensure that a better decision will be made.

3.1. State matrix

Each state in which the system is (or was in the past) corresponds to the monitoring configuration deployed at each moment. For our purposes, the definition of a state is given by the sensor placement –which sensors are deployed in which domain– and the properties used in each sensor to satisfy the requirements demanded by each domain. Thus, a *state* is modelled by an $m \times u$ matrix, where rows denote the domains defined in the information system, $\forall D_i \in D$ $(1 \leq i \leq m)$, and columns represent the requirements that need to be monitored, $\forall R_k \in R$ $(1 \leq k \leq u)$, to ensure the proper functioning of the system.

Cell (i, k) in the state matrix defines the set of properties that R_k satisfies as demanded by domain D_i. All these properties are provided by the sensors deployed in D_i. Thus, each cell (i, k) in the state matrix, with $1 \leq i \leq m$ and $1 \leq k \leq u$, is defined as a set of pairs $<Sensor, Property>$:

$$State(D_i, R_k) = \{(S_j, P_l)\} = \{S_j P_l\}$$

where $S_j \in S$ is the sensor configured with the l-th property $P_l \in P$. For example, the state associated with the example shown in Figure 1 would be:

$$\begin{pmatrix} \{S_7P_1\} & \{S_6P_2, S_7P_2\} & \{\} \\ \{\} & \{S_1P_2, S_3P_2\} & \{S_1P_3, S_3P_4\} \\ \{S_4P_1, S_5P_1\} & \{S_2P_2, S_4P_2, S_5P_2\} & \{S_2P_4, S_4P_3, S_4P_4\} \end{pmatrix}$$

Note that the number of states in the monitoring system is bounded by $|S| \cdot |D| \cdot 2^{|P(S)|} - 1$. In the previous example, the system may move to one of the $21 \cdot 2^{21} - 1$ (around 44 million) states and check if the new one is somehow better. We next elaborate on this.

3.2. Reconfiguration architecture

The selection of a new state must be carried out so as to maximise the information gain when identifying threats against the information system. In our proposal, this translates into finding the best possible sensor placement (SP) along with the appropriate (re-)configuration for each sensor. Ideally, this decision should be made in a fully automated way, as human involvement might not always be available, e.g., in autonomous systems, or would not be appropriate if a fast response is required. With this aim in mind, we propose the architecture shown in Figure 2 to automatically choose a state to move to considering all the historical information received so far.

In our scheme, we assume that sensors' behaviour is assessed by analysing the events they generate. The *Trust & Reputation* module uses this information

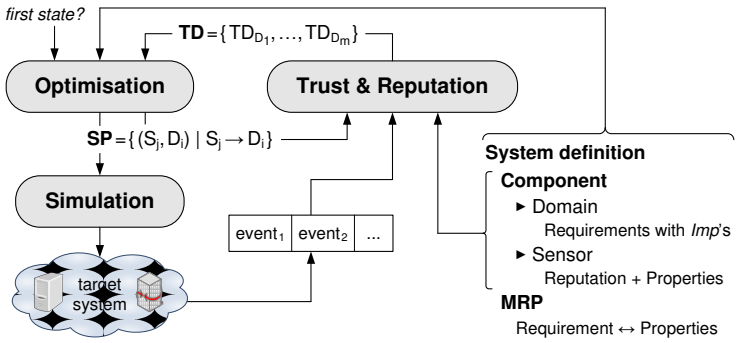

Figure 2: Architecture of the sensor reconfiguration system.

to derive a monitoring quality measure (trust diversity) for each domain. Such an assessment is obtained by analysing three main factors for each sensor S_j: 1) the alerts reported by S_j; 2) the alerts reported by other sensors in the same domain as S_j (i.e., its neighbours); and 3) the capabilities of S_j and its neighbours. The key outcome of this analysis is a trust value for each sensor, which is used both to assess the trustworthiness of its alerts and also to measure the trust diversity in the domain. This module is described in detail in Section 4.

Once each domain's trust diversity is updated, the *Optimisation* module decides if a reconfiguration is required and, if so, generates the best possible sensor placement. Finding such a placement becomes essentially an optimisation problem because of the potentially huge number of states that need to be explored. Furthermore, search proceeds over valid states only, i.e., those that satisfy the requirements demanded by domains, as not all sensors might be appropriate for all domains. This is already taken into account in Figure 2 by explicitly using the system definition as an input to the optimisation module. Other non-functional factors could be taken into account too. For example, in many situations solutions that imply the smallest possible number of changes with respect to the current deployment will be preferred, as reconfiguration may entail some non-negligible costs. For the reasons given above, we propose to use automated optimisation techniques, such as the ones presented for example in [12, 13], to solve the following problem:

> Find a sensor placement $SP = \{(S_j, D_i)\}$ that maximises $Q(SP)$, minimises $Cost(SP)$, and such that $V(SP) = \texttt{true}$.

where $Q(SP)$ is some quality metric associated with SP; $Cost(SP)$ measures all the costs involved in applying it; and $V(SP) = \texttt{true}$ if SP is a valid solution.

In principle, the particular choice of one optimisation technique or another is relatively unimportant as long as it provides a good enough solution. Whatever the procedure chosen, we suggest that the trust diversity (quantified as discussed in the next section) should be the criterion to be maximised as function $Q()$. The remaining important decision is choosing *when* to search for a better configuration. We will address this point later on in Section 4.4.

Once a new sensor placement is chosen, the *Simulation* module will proceed to enforce the required sensor changes in the physical target system. Different alternatives can be used to carry out dynamic reconfigurations of the monitoring infrastructure. We next describe two possibilites:

- *Reconfiguration*: Each network domain keeps a copy of all sensors, but at each given time only a subset of them is activated and properly configured. When a new placement needs to be enforced, the domain receives instructions about the new state and configuration of each sensor. This is an easy to implement solution with current technologies. For example, [35, 36] suggest the implementation of a Component Management Interface (CMI) to remotely manage, configure and turn on/off software components by making use of standards like Web-Based Enterprise Management (WBEM) [37]. The main advantage of this solution is its low overhead in terms of communication bandwidth, which makes reconfigurations rather inexpensive. However, even though this avoids the need to physically relocate sensors, it forces each location to maintain a pool of available sensors.

- *Reallocation*: Alternatively, reallocation strategies entail a more complex process, as sensors will be remotely installed, configured and turned on at each location where they are needed. This is feasible with well-known technologies such as *mobile agents*, where NIDSs are autonomous systems with mobility capabilities capable of being configured at runtime. For example, [38] proposes a Snort-based mobile agent approach.

4. Trust and reputation management system

This section describes the trust and reputation management system designed to model the behaviour of monitoring sensors. Such a system plays a key role in two different tasks. First, it is used to assess sensors' honesty and, indirectly, to quantify how trustworthy the events they generate are. In addition, trust will be the most important factor in determining how sensors should be placed on the system in order to maximise the overall detection quality. To do this, we introduce one remarkably effective heuristic: *trust diversity*.

Roughly speaking, the idea is to guarantee, whenever possible, that sensors with quite different trust values are jointly deployed. This fact has several advantages. For example, it ensures that unreliable sensors are never left alone, thus improving the quality of evidences gathered at each domain. Furthermore, it helps to rapidly identify misbehaving sensors (e.g., by contrasting the alerts, or the lack of them, issued by different sensors located at the same spot).

Figure 3 illustrates how monitoring evolves over time.

The state $State^{(t)}$ at a given time t is given by its sensor placement, included in the monitoring configuration, and the events received. Each event E_k is subsequently assigned a trust value $T(E_k)$ indicating the likelihood of it being true (see Section 4.2 for details). After this, the reputation of each sensor is updated

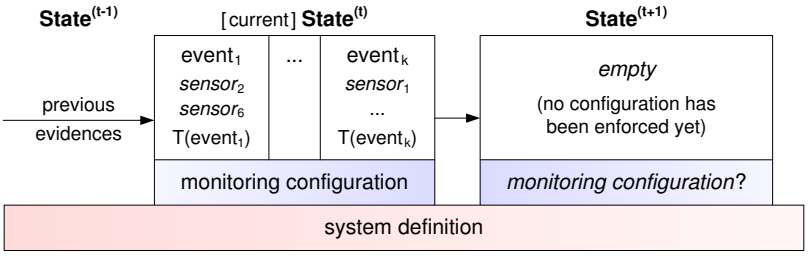

Figure 3: State evolution over time

according to the processed events. Changes in sensors' reputation could make the current monitoring configuration inadequate (e.g., with domains left protected only by a few unreliable sensors). When this occurs, a new configuration is derived and applied.

4.1. Computing trust diversity

We quantify "diversity" in a population of reputation values through a measure of dispersion [14], such as the range, the interquartile range (IQR), the mean difference or the standard deviation, amongst others. Whatever the specific measure used, trust diversity is computed amongst the sensors' reputation at three different levels: 1) at the requirement level, to guarantee the truthfulness of the events reported by sensors that have been assigned to monitor a given requirement; 2) at the domain level, to ensure that all its requirements are being monitored by a sufficiently capable set of sensors; and 3) globally at the information system level, to guarantee that the sensor placement is good enough to lessen the uncertainty about the current situational picture. We next discuss in detail each one of them.

4.1.1. Trust diversity at the requirement level

Besides measuring diversity in sensors' reputation values, one major aspect at this level is the risk (or impact) associated with each monitored requirement. The trust diversity of a requirement $R_k \in R$, demanded by a generic domain Ω and denoted $TD_\Omega(R_k) \in [0,1]$, is obtained as:

$$TD_\Omega(R_k) = max\{Rep_\Omega(S_{R_k})\} \cdot \psi(Rep_\Omega(S_{j,R_k}) \cdot \mu_\Omega(R_{k,S_j})),\ \forall S_j \in SP(\Omega) \quad (1)$$

where $max\{Rep_\Omega(S_{R_k})\}$ is the highest reputation value amongst all sensors assigned to domain Ω that monitor the requirement R_k; ψ is a measure of dispersion amongst the reputation of sensors in Ω monitoring R_k; $Rep_\Omega(S_{j,R_k})$ represents the reputation of the j-th sensor assigned to Ω and configured to monitor R_k; and $\mu_\Omega(R_{k,S_j}) \in [0,1]$ defines the risk incurred if requirement R_k is not satisfied, this being monitored by S_j.

Note that, in the above formula, dispersion is weighted by the maximum reputation amongst sensors. This guarantees that diversity is greater as the

reputation values of the sensors are higher. For example, assume two monitoring configurations where two sensors monitor the same requirement. The reputation values are 0.1 and 0.4 in the first case, and 0.6 and 0.9 in the latter. If we choose the range as measure of dispersion, the value is the same in both cases ($\psi = 0.3$). However, the trust diversity of the first configuration would be worse.

Finally, $\mu_\Omega(R_{k,S_j})$ can be defined according to the impact associated with a compromise of R_k and the number of properties configured in the sensors to monitor R_k. Thus, the risk associated with R_k can be computed as:

$$\mu_\Omega(R_{k,S_j}) = \frac{Imp_\Omega(R_k) \cdot |P_\Omega(S_{j,R_k})|}{|P_\Omega(R_k)|} \tag{2}$$

where $Imp_\Omega(R_k)$ corresponds to the impact for Ω when R_k is compromised; $|P_\Omega(S_{j,R_k})|$ is the number of properties configured in S_j to monitor R_k; and $|P_\Omega(R_k)|$ is the total number of properties required to monitor R_k in Ω. For example, for the monitoring configuration shown in Figure 1, the trust diversity of requirement R_1 demanded by D_3 would be $TD_{D_3}(R_1) = 0.1622$, using the range as measure of dispersion. The complete computation would be $TD_{D_3}(R_1) = 0.91 \cdot ((0.91 \cdot 0.81) - (0.69 \cdot 0.81))$, knowing that R_1 is being monitored in D_3 by a single property of both S_4 and S_5 (the other sensor deployed in such domain, S_2, is not configured to monitor R_1). Furthemore, $Rep_{D_3}(S_4) = 0.91$ –the sensor in D_3 with the highest reputation–, $Rep_{D_3}(S_5) = 0.69$ and 0.81 represents the impact for D_3 if R_1 –$Imp_{D_3}(R_1)$– is compromised. Following the same example, the remaining equirements demanded by D_3 –R_2 and R_3– would have $TD_{D_3}(R_2) = 0.2002$ and $TD_{D_3}(R_3) = 0.574$.

4.1.2. Trust diversity at the domain level

After computing the trust diversity for each requirement, the trust diversity for the same generic domain Ω, $TD_\Omega \in [0, 1]$, can be obtained through an aggregation operation as the one defined in (3).

$$TD_\Omega = \bigoplus_{k=1}^{\Phi_\Omega(R)} TD_\Omega(R_k) \tag{3}$$

where $\oplus$ is an aggregation operation; $\Phi_\Omega(R)$ is the total number of requirements demanded by Ω; and $TD_\Omega(R_k)$ represents the trust diversity of the k-th requirement demanded by Ω, computed in (1).

Various aggregation operations can be used here, notably average values such as the arithmetic or harmonic mean. For example, the trust diversity of D_3 in the configuration shown in Figure 1 would be $TD_{D_3} = 0.3121$, if the arithmetic mean is used in (3) as aggregation operation; or $TD_{D_3} = 0.2325$ if the harmonic mean is chosen. (These figures come from aggregating the trust values of each individual requirement as detailed at the end of the preceding section: $TD_{D_3}(R_1) = 0.1622$, $TD_{D_3}(R_2) = 0.2002$ and $TD_{D_3}(R_3) = 0.574$.)

4.1.3. Trust diversity at the information system level

The trust diversity on the entire system is computed similarly by aggregating each domain's diversity:

$$TD_{IS} = \bigoplus_{i=1}^{\Phi_{IS}(D)} TD_{D_i} \tag{4}$$

where $\oplus$ is again an aggregation operation; $\Phi_{IS}(D)$ is the total number of domains that comprise the information system; and TD_{D_i} represents the trust diversity of each domain, $\forall D_i \in IS$, computed in (3).

Finally, Table 2 presents a summary of the notation used above, including where each symbol is defined and used, together with a brief description.

Function	**Eq.**	**Used**	**Description**
$TD_\Omega(R_k)$	(1)	(3,5)	Trust diversity of requirement R_k that is demanded by a generic domain Ω
$Rep_\Omega(S_{j,R_k})$		(1,6)	Reputation for Ω about a sensor S_j assigned to it that monitors R_k
ψ		(1)	Measure of dispersion chosen by the administrator
$\mu_\Omega(R_{k,S_j})$	(2)	(1,7)	Risk for Ω if R_k, monitored by S_j, is not satisfied
$Imp_\Omega(R_k)$		(2)	Impact for domain Ω if R_k is compromised
$P_\Omega(S_{j,R_k})$		(2)	Set of properties configured in sensor S_j, deployed in Ω, to monitor R_k
$P_\Omega(R_k)$		(2)	Set of properties demanded by Ω to monitor R_k
TD_Ω	(3)	(4)	Trust diversity of a generic domain Ω
$\oplus$		(3,4,5)	Aggregation operation chosen by the administrator
$\Phi_\Omega(R)$		(3)	Total number of requirements demanded by Ω
TD_{IS}	(4)		Trust diversity at the information system level
$\Phi_{IS}(D)$		(4)	Total number of domains of the information system

Table 2: Functions and variables for computing trust diversity

4.2. Behavioural sensor modelling

When the monitoring system receives a new event from one or more sensors, it first assesses the trust in that event being true and, subsequently, updates the reputation of the sensors associated with it. Note that the set of sensors associated with an event includes not only those that reported it, but also those that should have reported it and did not issue any alert.

We denote events by E_{R_k} to emphasise that they refer to a particular requirement R_k. To compute the trust in a new event, the monitoring system takes into consideration the (dis)agreement level reached by all the sensors configured to detect such an event, the number of domains where the event has been produced and the trust diversity in all these domains. Thus, the trust on a new event, denoted as $T(E_{R_k}) \in [0, 1]$, is given by:

$$T(E_{R_k}) = \bigoplus_{i=1}^{\Phi_D(E_{R_k})} |\delta_{D_i}(E_{R_k})| \cdot TD_{D_i}(R_k) \tag{5}$$

where $\oplus$ is an aggregation operation; $\Phi_D(E_{R_k})$ represents the number of domains where the event E reporting on the compromise of R_k has been generated; $\delta_{D_i}(E_{R_k})$ is the absolute agreement level reached by the sensors deployed in the i-th domain D_i, related to the generation of E_{R_k}; and $TD_{D_i}(R_k)$ is the trust diversity of each D_i where E_{R_k} has been generated, as given in (1). Note that the aggregation operations chosen for (3), (4) and (5) can be different.

Trust in an event, as defined in (5), expresses the confidence that the monitoring system can place in such an event when it is reported by sensors, either as a consequence of an honest behaviour, producing a true positive, or exhibiting a malicious behaviour, producing an MFP. $T(E_{R_k})$ only computes the trust on E_{R_k} indicating whether it is truthful or not. The discrimination between both types of events is achieved by the agreement level amongst sensors.

The agreement level reached by sensors follows a voting-based scheme that takes into account the reputation of the sensors that have sent the event E_{R_k} and those that have not done it but however are configured to detect it. Hence, the agreement amongst the sensors of a domain Ω involved in reporting E_{R_k}, denoted as $\delta_\Omega(E_{R_k}) \in [-1, 1]$, can be computed by using (6).

$$\delta_\Omega(E_{R_k}) = \frac{\sum_{j=1}^{\Phi_{S_\Omega}(E_{R_k})} Rep_\Omega(S_{j,R_k})}{\Phi_{S_\Omega}(E_{R_k})} - \frac{\sum_{j=1}^{\Phi_{S_\Omega}(\neg E_{R_k})} Rep_\Omega(S_{j,R_k})}{\Phi_{S_\Omega}(\neg E_{R_k})} \tag{6}$$

where $\Phi_{S_\Omega}(E_{R_k})$ is the number of sensors deployed in Ω that have sent the event E_{R_k}, while $\Phi_{S_\Omega}(\neg E_{R_k})$ is the number of sensors in Ω that did not report it; and $Rep(S_{j,R_k})$ represents the reputation of the j-th sensor deployed in Ω that is configured to monitor R_k. A neutral agreement, i.e., $\delta_\Omega(E_{R_k}) = 0$, indicates total uncertainty on whether E_{R_k} is actually a true or false positive. Similarly, $\delta_\Omega(E_{R_k}) = 1$ indicates full confidence on the fact that the event is a true positive reported by sensors we trust, while $\delta_\Omega(E_{R_k}) = -1$ indicates full confidence in E_{R_k} being an MFP coming from dishonest sensors.

As sensors can report their detection responses about the same event in different time instants, it is required to establish a mechanism that allows the monitoring system to gather all instances of an event before knowing the proper agreement level reached by sensors (i.e., before executing the previous voting-based scheme). In this sense, many research works in IDS-based solutions make use of the concept of *time window*, where events are gathered in a given period of time of, for example, two seconds [8].

Note that the agreement function in (6) is a straightforward majority-based voting scheme. More complex procedures can be used, for example, by clustering sensors' reputation according to their trust diversity. For the purposes of this work, the choice of which scheme to use is quite irrelevant, as long as they provide a good (dis)agreement level when assessing an event.

Once $T(E_{R_k})$ is obtained, the system can decide if the event is reliable enough to be considered as truthful. The simplest way to do this is by checking whether $T(E_{R_k}) \geq T_\sigma$, where T_σ is some minimum confidence threshold set by the administrator. Richer decision models are possible too. For example, events can be grouped into different classes according to their trust, so priority will be given to those more trustworthy. How such information is used is an element external to our system.

4.3. Updating sensors' reputation

The overall trust value derived for each event can be used to update the reputation of all the sensors associated with it. Associated sensors will be then rewarded or punished, increasing or decreasing their reputation depending on the global assessment of the event and their behaviour in reporting it.

The reputation update process takes into account three factors: 1) the global assessment of the event, explained in the previous section; 2) the behaviour observed for each sensor, i.e., if it reported the event or not, and if it should have done so; and 3) a mechanism to adjust reputation over time, so that recent behaviours weigh more than older ones. Thus, the reputation of sensor S_j at time t, denoted as $Rep(S_j)^{(t)}$, is updated as follows:

$$Rep(S_j)^{(t)} = \omega \cdot Rep(S_j)^{(t-1)} + (1-\omega) \cdot \frac{\sum_{k=1}^{\Phi_{S_j}(E)} Sat(S_j, E_k) \cdot \mu(R_{E_k,S_j}) \cdot \xi(E_k)}{\Phi_{S_j}(E)} \quad (7)$$

where $Rep(S_j)^{(t-1)}$ is the latest reputation stored for S_j; $\Phi_{S_j}(E)$ represents the total number of events in which S_j has been involved, either because it has sent the event or because it failed to report when it is configured to detect it; $Sat(S_j, E_k) \in [0, 1]$ is the satisfaction with the behaviour shown by S_j regarding the k-th event; $\mu(R_{E_k,S_j})$ is the risk associated with the requirement affected by E_k, computed in (2); and $\xi(E_k) \in [0, 1]$ is a forgetting factor for the time passed since E_k was reported up to the current time t. The weight of each term in (7) is controlled by $\omega \in [0, 1]$, which accounts for the importance of past behaviour.

The satisfaction model for the behaviour of a sensor S_j involved in a particular event E_k, denoted $Sat(S_j, E_k)$, depends on the trust placed in such an event during its assessment –given by (5)– and the subsequent action taken by S_j with respect to that event: reporting or not. This satisfaction is computed as:

$$Sat(S_j, E_k) = \begin{cases} |\delta(E_k)| & if\ (T(E_k) \geq T_\sigma \wedge S_j \subseteq G_S(E_k)) \\ & \vee\ (T(E_k) < T_\sigma \wedge S_j \nsubseteq G_S(E_k)) \\ -|\delta(E_k)| & otherwise \end{cases} \quad (8)$$

where $|\delta(E_k)|$ is the agreement level reached by sensors deployed in the same domain as S_j, computed in (6); $T(E_k)$ is the trust on E_k, computed in (5); T_σ is the threshold set by the administrator to decide whether E_k is trustworthy or not; and $G_S(E_k)$ is the set of sensors that have generated E_k.

This satisfaction with events contributes to the reward or punishment of sensors in reputation terms. If the event is considered genuine and it has been sent by the sensor, or if it is not genuine and it has not been sent by the sensor –it is not an MFP–, the sensor will be rewarded by increasing its reputation. Otherwise, if the event is genuine and the sensor did not report it –it is an MFN–, or if it actually did not happen and the sensor reported it –it is an MFP–, the sensor will see its reputation decreased.

Finally, it is also crucial to incorporate a suitable time-dependent function to model the weight of sensor behaviours exhibited at different time instants. For example, it seems reasonable that recent behaviours should have more importance than those observed time ago. Such a function is generally known in the literature as a *forgetting factor* [15], and it is often modelled as a linear function where the relative importance between two events maintains a constant proportion over time. Other alternatives exist. For example, humans tend to give more importance to the difference between events if they are recent [16]. In this work, we follow this approach and model the forgetting factor as:

$$\xi(E_k) = e^{-\Delta t(E_k)^{\lambda}} \tag{9}$$

where $\Delta t(E_k)$ is the difference between the current time and the time when the event E_k was generated; and λ defines the variation of the forgetting factor by giving higher or lower values to the events generated more recently. The effect of this weight is graphically illustrated in Figure 4.

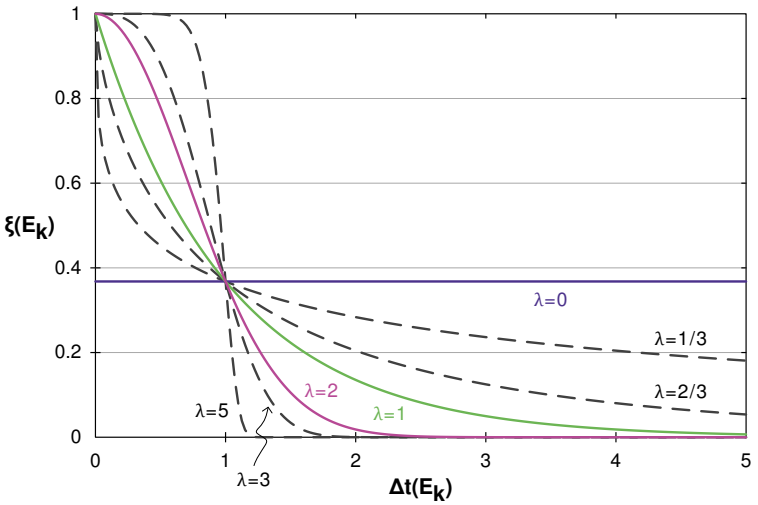

Figure 4: Importance of events over time (forgetting factor)

The importance of an event may vary considerably over time depending on λ. For example, events are quickly forgotten when $\lambda = 1/3$, and the difference between events has a similar importance from a certain point on. On the

contrary, for $\lambda = 5$ recent events maintain a high importance but are rapidly forgotten.

4.4. Evolving towards a new state

There are several reasons to switch from a monitoring configuration to a new one. One natural goal is to improve the monitoring quality in terms of the requirements demanded by the information system. Another could be to rapidly identify dishonest or incapable sensors, even if in doing so we leave some parts of the system with a deficient protection.

In this work, we choose the overall trust diversity as an indicator of the monitoring quality. Note that, because of the way that we have defined it, it can be easily understood as a measure of the *quality of the placement*. Thus, when it is too low, we have no means to detect dishonest behaviours exhibited by sensors and, therefore, the situational picture is no longer useful.

As discussed in Section 4.1, trust diversity is defined at three different levels. Consequently the reconfiguration decision can be made at any of them. However, moving sensors amongst domains can potentially affect various diversity values, which might force us to recompute trust diversity at upper levels. This fact does not imply that further computations are required. For example, all sensors will keep the same reputation score they had before enforcing the new monitoring configuration, regardless of whether they are relocated or not. Reputation would only change when their behaviour is subsequently assessed by using (7), provided that the trust diversities have changed due to the new neighbouring sensors' reputation.

As for *when* a reconfiguration should take place, one could check if the trust diversity for each requirement R_k –computed in (1)– is below some tolerable threshold. Thus, if $TD_{\Omega}(R_k) < TD_{\Omega,\sigma}(R_k)$, the monitoring system should move to a new state that provides such a protection. Alternatively, reconfiguration may be triggered by thresholds associated with the trust diversity at the information system or at the domain level. Note that it is perfectly possible to define different thresholds for different requirements and different domains.

Decisions about reconfiguring sensor placements can be driven by more complex rules. For example, sensors may be dynamically deployed when needed. In [8], it is suggested to count the number of events received in a predefined time window and pay more attention, e.g., by assigning more sensors, to those domains where more events are happening. We suggest keeping an eye on the converse too, i.e., to sensors that report too little, as this might be evidence of MFNs.

Finally, and as we did previously, Table 3 summarises functions, variables and other parameters introduced above to model sensor behaviour.

5. Experimental results

In this section, we discuss some experimental results obtained with a prototype implementation of the ideas presented in this paper. The fundamental

Function	Eq.	Used	Description
$T(E_{R_k})$	(5)	(8)	Trust on an event E that affects requirement R_k
$\Phi_D(E_{R_k})$		(5)	Total number of domains where E affecting R_k (E_{R_k}) was reported
$\delta_\Omega(E_{R_k})$	(6)	(5,8)	Agreement level reached by the sensors deployed in domain Ω involved in reporting E_{R_k}
$\Phi_{S_\Omega}(E_{R_k})$		(6)	Number of sensors deployed in Ω reporting E_{R_k}
$\Phi_{S_\Omega}(\neg E_{R_k})$		(6)	Number of sensors deployed in Ω not reporting E_{R_k}
$Rep(S_j)^{(t)}$	(7)		Reputation of sensor S_j at time t
$Rep(S_j)^{(t-1)}$		(7)	Latest reputation of S_j
ω		(7)	Weight balancing the importance between past and current behaviours
$\Phi_{S_j}(E)$		(7)	Number of events E in which S_j was involved
$Sat(S_j, E_k)$	(8)	(7)	Satisfaction on S_j's behaviour in reporting E_k
$\xi(E_k)$	(9)	(7)	Forgetting factor for E_k
T_σ		(8)	Threshold set by the administrator to decide if an event is trustworthy or not
$G_S(E_k)$		(8)	Set of sensors in S that reported E_k
$\Delta t(E_k)$		(9)	Time difference since E_k was reported
λ		(9)	Weight putting higher or lower importance on events reported more recently

Table 3: Functions and variables involved in modelling the behaviour of a sensor

aim is to demonstrate how the joint operation of reputation, trust diversity and reconfiguration strengthens the quality of network monitoring, both increasing the quality of the assessments made about the information received from sensors and adapting the sensor system accordingly.

5.1. Experimental setting

We have built a simulator that allows us to specify information systems (in terms of domains and requirements) and monitoring capabilities (number of sensors, their features and their location), as discussed above. A simulation takes as input these two specifications and the sequence of events that will take place: where and when attack attempts will happen and which sensors are progressively compromised. At each time instant, a number of such events takes place and the affected sensors respond in accordance with their reputation by reporting or not. Once all alerts have been received, the system assesses the trust in the reported event, updates sensor reputations and decides on reconfiguration, if needed.

In the experiments described below we have used an information system composed of 20 domains, 10 requirements and 500 sensors. For simplicity, only

one property per requirement is considered. Each sensor's initial reputation is assigned a random number in $[0, 1]$.

In order to demonstrate the benefits of reconfiguration, we used a simple optimisation algorithm to search for new placements. Over the last decades, problems with a very similar structure have been approached by means of heuristic optimisation procedures, in some cases with remarkable results such as the ones presented in [17, 18]. In our case, we used a variant of a simulated annealing algorithm [19], which can be viewed as a basic hill-climbing algorithm coupled with the probabilistic acceptance of non-improving solutions [20].

The search starts at some initial solution $S_0 \in \mathbb{S}$, where $\mathbb{S}$ denotes the entire solution space. We represent placements as unique assignments of each sensor to one domain. The algorithm employs a control parameter $T \in \mathbb{R}^+$ known as the *temperature*. This starts at some positive value T_0 and is gradually lowered at each iteration, typically by geometric cooling: $T_{i+1} = \alpha T_i$, $\alpha \in (0, 1)$. At each temperature, a number MIL (Moves in Inner Loop) of neighbour states are considered. A candidate state C in the neighbourhood $N(S_i)$ of S_i is obtained by applying some move function to S_i. In our case, this consists of randomly deciding whether to swap the location of two sensors, or randomly picking one sensor and moving it to a different domain. In this process, it is guaranteed that the detection requirements are satisfied.

The new solution is accepted if it is better than S_i (as measured by a fitness function $F : \mathbb{S} \to \mathbb{R}$). To escape from local optima, the technique may also accept a candidate that is slightly worse than S_i, meaning that its fitness is no more than $|T \ln U|$ lower, with U a uniform random variable in $(0, 1)$. As T becomes smaller, this term gets closer to 0, so as the temperature is gradually lowered it becomes harder to accept worse moves. In our case, function F is simply the trust diversity of the information system.

The algorithm terminates when some stopping criterion is met, usually after a fixed number MaxIL of inner loops have been executed, or when some maximum number MUL of consecutive inner loops without improvements has been reached. The basic algorithm is shown in Algorithm 1.

Algorithm 1 Basic simulated annealing for maximisation problems

```
1:  S ← S_0
2:  T ← T_0
3:  repeat
4:     for MIL times do
5:        Pick C ∈ N(S) with uniform probability
6:        Pick U ∈ (0, 1) with uniform probability
7:        if F(C) > F(S) + T ln U then
8:           S ← C
9:        end if
10:    end for
11:    T ← αT
12: until criterion is met
```

Simulated annealing offers a good balance between simplicity of the optimisation procedure and quality of the found solutions. This was the main reason for choosing it for rapid prototyping and experimentation. As shown later, such a simple search mechanism suffices to find significantly improved placements. Note, however, that optimality is not guaranteed, and more sophisticated optimisation algorithms will very likely provide better solutions. In practice, the choice of one optimisation algorithm or another will depend on a number of factors, including the time betwen reconfigurations, the place where the search will take place (centralized vs. distributed), etc. In any case, we emphasise that the specific optimisation procedure is an external element to our proposal and different schemes can be used as long as they improve trust diversity.

In our simulations the search for a better placement is triggered when trust diversity falls below a given threshold. (This depends on the choice of measure of dispersion and aggregation operations.) The search parameters certainly need some tuning depending on how much time is given to searching for a better placement.

5.2. Experiment 1: Improving trust assessment

In this first experiment we show how trust assessment on the events –$T(E_k)$, computed in (5)– varies depending on the trust diversity amongst sensors in a given sensor placement. Thus, the agreement level reached by sensors in assessing such events will be a key factor in deciding if they are truthful. If not, they will be discarded to achieve a better detection coverage.

Let us consider the monitoring configuration shown in Figure 5a. Each row in the map represents a domain and each column represents a requirement. A cell (i, j) is given a colour indicating the trust diversity of the j-th requirement in the i-th domain. A gray scale has been used, with darker colours indicating low trust diversity and vice versa. IQR has been chosen as measure of dispersion, as defined in (1). Cells represented with a $\times$ indicate that the domain does not require protection for such a requirement. In the figure, the trust diversity values for each domain and for the entire placement are shown too.

The placement shown in Figure 5a corresponds to the one existing at a given simulation time, whereas Figure 5b shows the placement configuration suggested by the *Optimisation* module. We can observe that the trust diversity increases considerably in all the domains, resulting in an overall increase of 92.24% (it jumps from 0.4277 to 0.8222).

In order to compare both monitoring configurations, we simulated 2000 events (1000 true positives and 1000 MFPs) and evaluated the overall trust assessment obtained depending on sensors' reports, as defined in (5). The reputation of each sensor remains the same in order to maintain the same experimental conditions for both configurations. Table 4 shows the average results obtained. Four different measures of dispersion have been chosen for (1) and four aggregation operations are used for (5).

As shown in Table 4, the average trust that the system derives for the same events varies considerably from the initial sensor placement –Figure 5a– to the

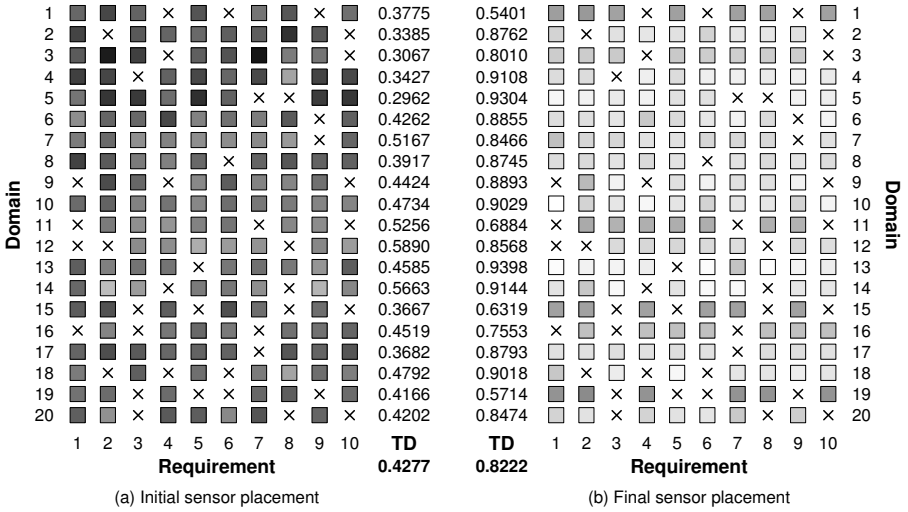

Figure 5: Getting a more reliable new state

	Aggregation operation	Measure of dispersion: **Absolute difference**	**Standard deviation**	**Mean difference**	**IQR**
Initial SP	**Minimum value**	0.0663	0.0233	0.0189	0.0273
	Maximum value	**0.4698**	0.1541	0.1682	0.2892
	Arithmetic mean	0.2587	0.0800	0.0862	0.1377
	Harmonic mean	0.1927	0.0525	0.0636	0.0898
Final SP	**Minimum value**	0.1129	0.1086	0.0751	0.0973
	Maximum value	**0.4837**	**0.4748**	**0.4308**	**0.9062**
	Arithmetic mean	0.2885	**0.3658**	0.3269	**0.6013**
	Harmonic mean	0.2351	0.2653	0.2277	0.3315

Table 4: Average trust on events for different measures, operations and sensor placements

new monitoring configuration with higher trust diversity amongst its sensors –Figure 5b. Results also vary depending on the particular choice of functions used in (1) and (5). In our experiments, we found that the best performing aggregation operation is the arithmetic mean, while the standard deviation and IQR tend to offer the best results as measures of dispersion.

It is worth mentioning that the trust threshold used to accept or discard an event is a critical factor. T_σ should be defined in a different way depending on the choice of functions. For example, if $T_\sigma = 1/3$ (marked in bold in Table 4) and the arithmetic mean and the standard deviation or IQR are chosen, then all events will be discarded in the initial sensor placement (SP) as they are not reliable enough. Better results could be obtained using the absolute difference and the maximum, although they are functions that do not take into consideration the data distribution; only two sensors are required, one with the highest reputation and another with the lowest one, to reach the maximum trust diversity. Instead, the standard deviation, mean difference and IQR generally provide better results as they consider not only the data dispersion but also their distribution. Experiment 3, presented later in Section 5.4, analyses in more depth the influence of the trust threshold in accepting and discarding events for a given measure of dispersion and aggregation operation.

The increase in the trust assessment over these events translates into a better discrimination between honest and bogus events (i.e., between true positives and MFPs, respectively). This is directly related to the level of agreement reached amongst the sensors affected by an event. The average agreement levels amongst sensors in the initial sensor placement, computed in (6), are 0.3139 and -0.3102 in assessing honest and bogus events, respectively. Although the average trust on such events is quite low, as shown in Table 4, sensors do not reach a satisfactory agreement level, demonstrating the uncertainty in assessing what is really happening. Instead, agreement levels reach 0.7253 (honest events) and -0.7098 (bogus events) in the final sensor placement, thus providing sufficient agreement to accept or discard them.

The overall consequences of such an improvement translate into a remarkable improvement in detection quality, as shown in Table 5. We have set a minimum threshold of 0.5 to consider an agreement level amongst sensors as satisfactory; that is, $\delta_\Omega(E_{R_k}) \geq 0.5$ in (6).

	Honest events		**Bogus events**	
	Accepted	Discarded	Accepted	Discarded
Initial SP	61	939	942	58
Final SP	843	157	159	841

true positive	**MFN**	**MFP**	true negative

Table 5: Events accepted and discarded depending on agreement levels reached by sensors

5.3. Experiment 2: Resilience to misbehaviour

Apart from the benefits in terms of detection quality, analysed and discussed in the previous experiment, sensor placements with higher trust diversity show higher resilience to the presence of malicious or unreliable sensors. We have explored this in a second set of experiments. Starting with a given monitoring configuration, we simulate how the system is affected by an increasing number of compromised sensors that report falsely. At each step, i.e., with a given percentage of compromised sensors, we inject 2000 events as before (half honest, half bogus) and compute the overall trust derived by the system for each one of them. Figure 6 shows the distribution (box plots) and average values (lines) of trust values for two monitoring configurations with low (blue) and high (red) trust diversity.

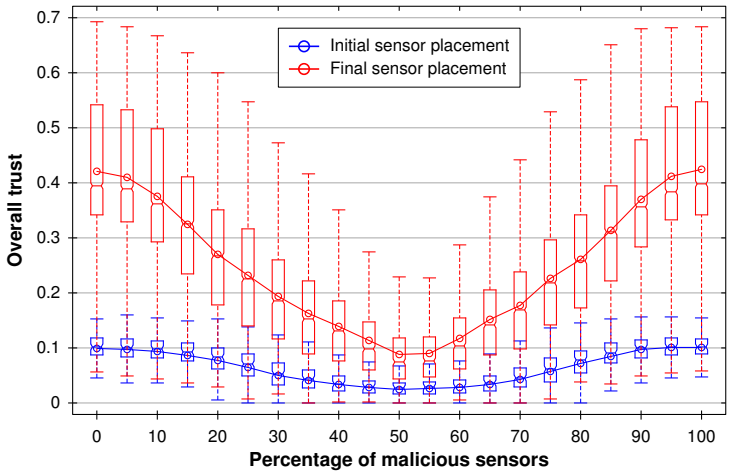

Figure 6: Distribution of trust values over events for varying number of compromised sensors

There is a common behaviour in both cases: trust decreases as the number of malicious sensors increases, reaching a minimum when the disparity in their reputations is maximal (i.e., around 50% of each type). From that point on, malicious behaviour prevails and honest sensors are incorrectly accused of misbehaving by the weight of the majority. This will inevitably affect their reputation and, indirectly, those of dishonest sensors too, which will see their reputation increased. As a consequence, trust will increase again, although now the system is completely deceived in its perception of what is really happening. Note that this is unavoidable in any trust-driven system.

However, trust diversity makes a difference here. Firstly, as discussed in the previous experiment, trust in events increases dramatically with an optimised configuration. This is clearly noticeable here. Besides, the overall trust comparing the two monitoring configurations as the number of malicious sensors increases, i.e., the decreases in both curves, is larger in the final sensor placement than in the initial one. This can play a key role in detecting which

parts of the detection network are being compromised (e.g., by observing that trust across all registered events gets lower and lower). Indirectly, the same phenomenon could be observed in the dynamics of reputation values for the population of sensors.

If the pace at which sensors are compromised is not very high, the administrator will notice this through a progressive decrease in the trust of the alerts. Contrarily, in placements with low trust diversity, this fact is more difficult to detect: an adversary can compromise sensors one by one, at a slow pace, and no significant changes in the overall trust might be observed.

5.4. Experiment 3: Assessing agreement levels amongst sensors

Having demonstrated the resilience to misbehaviour in trust terms our last experiment shows how trust diversity affects the agreement levels reached by sensors when the fraction of malicious sensors varies over time. We have run a new set of experiments using the same monitoring configurations described in the first experiment and shown in Figure 5. As before, the reputation of the sensors remains the same to maintain the same experimental conditions.

In this experiment, we run several simulations increasing the number of malicious sensors at each step, injecting 1000 honest events classified as true positives and, finally, computing the number of such events that are accepted by the system, indicating a minimum satisfaction in the agreement level reached by sensors according to several trust thresholds T_σ. Figure 8 shows the results for both monitoring configurations, where each curve reflects the number of events accepted for a given percentage of malicious sensors and for a fixed trust threshold; that is, we increase T_σ progressively to force greater agreement levels amongst sensors, computed as in (6), in order to accept an event as truthful. Note that no bogus events are injected here, as the curves will follow patterns complementary to those shown here.

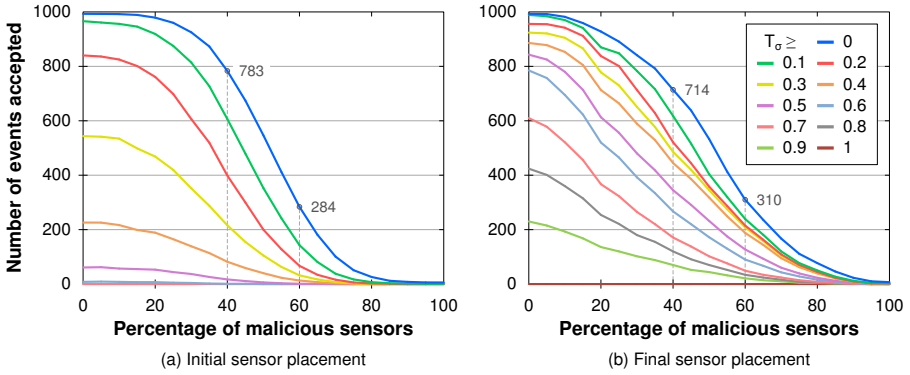

Figure 7: Distribution of trust values over events for varying number of compromised sensors

Figure 8: Agreement levels reached by sensors as they are progressively compromised

Similarly to the previous experiment, agreement levels in both monitoring configurations decrease as the number of malicious sensors increases, achieving better results with the final sensor placement as it properly classifies a greater number of events for a given trust threshold. For example, the initial sensor placement with $T_\sigma = 0.4$ presents similar results –number of accepted honest events– than the final sensor placement with $T_\sigma = 0.9$, a much more ambitious trust threshold for concluding that an event is true.

Another interesting conclusion can be drawn from the rate at which curves decrease as the number of malicious sensors increases. The drop for the initial sensor placement is more pronounced than for the final one. For example, for dotted lines in Figure 8, the range between 40% and 60% of malicious sensors delimits two zones with the greatest fall in agreement level terms for both configurations. For the initial sensor placement, the curve falls to 36.96% on average for a trust threshold range between 0 and 0.3, i.e., $0 \leq T_\sigma \leq 0.3$, whereas there is a fall of 34.18% for the final sensor placement. This difference becomes greater when $T_\sigma \leq 0.2$ (from 43.16% to 36.3%). This fact translates into better agreement levels –higher resilience– to the presence of malicious sensors when using a monitoring configuration with higher trust diversity.

It is also worth noting that, although the initial sensor placement provides a better result when there is a 40% level of malicious sensors with $T_\sigma = 0$ (783 versus 714 accepted events), the remaining figures, more realistic from a trust threshold perspective, are worst when compared to the final sensor placement.

6. Related work

Collaborative Intrusion Detection Networks (CIDN) were developed with the aim of detecting attacks where pieces of evidence need to be gathered at different network locations and subsequently correlated. Besides, they can also reduce the costs involved in threat mitigation by sharing intrusion detection resources amongst networks [21].

A CIDN consists of multiple distributed detection units (sensors) logically organised in a network topology. In centralised systems, such as DIDS [22], DShield [23] and NSTAT [24], each sensor shares alerts with a central correlation unit. Hierarchical approaches, e.g., GrIDS [25], EMERALD [26] and DSOC [27], attempt to address the scalability issues of centralised approaches by organising detection units into a tree-like topology. Finally, fully distributed approaches operate with nodes participating in a periodic exchange of information, such as DOMINO [28] or the one proposed in [29] for Peer-to-Peer (P2P) networks. We refer the reader to [1] for a more comprehensive account of existing CIDN technologies.

CIDN involving different partners are rare nowadays, as organisations are particularly reluctant to share useful information with almost any other actor. Trust plays an important role in CIDN. In most cases, the overall detection accuracy depends on all parties exhibiting honest behaviour, particularly in terms of the trustworthiness of reported alerts. These issues are ignored or inadequately

addressed in existing CIDNs, in part because most of them were not conceived for collaborative detection by multiple and heterogeneous organisations.

Trust diversity has been already identified as a key concept in the literature related to optimising cooperation between the members of a group. Essentially, all works in this area are related to socio-economic disciplines and mainly focus on dealing with human subjects that act as sources of information. In general, each source is modelled by a number of features, such as age, gender, salary, work position or role within the organisation. For example, in [30, 31], the authors presented two models to measure the impact of diversity when establishing a group of people for the purpose of developing information system projects. Naturally, factors other than diversity are also considered when establishing a work group, like potential conflicts of interests, difficulties in personal relationships, communication between members or expected contributions by each member.

In the context of intrusion detection networks, a number of works (such as the ones presented in [32, 33]) use the concept of diversity in detection capabilities to increase robustness when detecting security threats (e.g., by using different detection techniques and configurations to monitor the same set of requirements). Nevertheless, as far as we know none of them applies diversity to trust with the aim of increasing the quality and resilience of collaborative attack detection.

To the best of our knowledge, our proposal is the first to suggest using trust diversity within a computational environment and, particularly, amongst monitoring sensors by using their reputation as input to compute it. Trust and reputation are two main concepts in this work used to assess and improve the quality of information sources –sensors– in a sharing environment. In other areas, such as heterogeneous systems, trust and reputation management is a widely studied issue [34, 39], although none of these works focus on assessing information sources related to detection capabilities.

In this context, [3, 40] are amongst the first initiatives that evaluate malicious attitudes of IDSs. In [3], a trust-based framework to avoid sharing bogus alerts within collaborative intrusion detection networks was presented. However, this proposal is restricted to the management of Host-based IDSs (HIDS). Network-based IDSs (NIDS) were excluded from this framework, thus ignoring the detection of critical attacks that can only be observed by inspecting network packets (e.g., very common probe attacks like IPsweep, launched in the first stages of a multi-step attack to discover available services in the target network). Furthermore, this approach does not consider the reoutational differences amongst HIDSs. On the other hand, the work presented in [40] proposed a complete trust and reputation system for HIDSs and NIDSs, including their management in intra- and inter-domain environments. However, no distinction is made amongst different sensors, e.g., using their reputation, to maximise information gain through the trust diversity of the domain where they are deployed.

7. Conclusion and future work

Distributed and collaborative attack detection environments present many challenges. Dealing with the potentially massive numbers of events generated by sensors distributed over network locations is difficult. The situation is further complicated by the fact that reports may come from different organisations that choose to share information but that do not fully trust each other. In this work we have presented a novel approach to increase the quality of the assessments made about such alerts, providing, at the same time, the basis for decision-making processes resilient to malicious reporting, either by dishonesty or compromise.

We have introduced the notion of trust diversity and argued that it helps in deriving optimal monitoring configurations. In particular, we have proposed a way of quantifying such a measure in order to use it as an evaluation function when choosing amongst configurations. The trust assessment system proposed in this work allows us to determine if an alert should be believed or not. In turn, this can be leveraged to re-assess our perception of sensors' honesty (through reputation) so as to make better decisions in the future and/or choose better monitoring configurations. We have reported our experiences through simulation, showing that this scheme can play an important role in improving the quality of collaborative attack detection.

Our work can be extended in a number of ways. Reputation is but a proxy for trust in behaviour; the way we use it here might be enhanced by incorporating other factors when assessing each sensor's reputation (e.g., known failures or some particularities of its operation environment). Furthermore, we have not elaborated on how trust assessments about events can be combined to, for example, perform alert correlation or match them against an attack graph. We have intentionally avoided discussion of potential restrictions on what sorts of reconfiguration are possible. For example, not all domains might accept a given sensor. We believe that such restrictions can be easily modelled and incorporated into the optimisation process.

We also plan to test our proposal in a real-world scenario. This will allow us to doscover further complexities and issues arising from our model. Similarly, the integration of our reputation model and reconfiguration procedures into a real CIDN will prove valuable. In principle, such integration should not incur substantial costs, but first-hand experience will be essential to refine and/or re-design some system components.

Acknowledgments

This work has been partially funded with support from the Spanish MICINN (project RECLAMO, *Virtual and Collaborative Honeynets based on Trust Management and Autonomous Systems applied to Intrusion Management*, with code TIN2011-28287-C02-02) and the European Commission (FEDER/ERDF). Thanks also to the Funding Program for Research Groups of Excellence granted by the Séneca Foundation with code 04552/GERM/06.

References

[1] C.V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," Computers & Security, vol. 29, no. 1, pp. 124–140, February 2010, doi:10.1016/j.cose.2009.06.008.

[2] S. Salah, G. Maciá-Fernndez, and J.E. Díaz-Verdejo, "A model-based survey of alert correlation techniques," Computer Networks, In Press, January 2013, doi:10.1016/j.comnet.2012.10.022.

[3] C. Fung, J. Zhang, I. Aib, and R. Boutaba, "Trust management and admission control for host-based collaborative intrusion detection," Journal of Network and Systems Management, vol. 19, no. 2, pp. 257–277, June 2011, doi:10.1007/s10922-010-9176-7.

[4] M. Gil Pérez, F. Gómez Mármol, G. Martínez Pérez, and A.F. Gómez Skarmeta, "Mobility in collaborative alert systems: Building trust through reputation," NETWORKING 2011 Workshops, Workshop on Wireless Cooperative Network Security, vol. 6827 of Lecture Notes in Computer Science, pp. 251–262, May 2011, doi:10.1007/978-3-642-23041-7_24.

[5] J. Sabater and C. Sierra, "Review on computational trust and reputation models," Artificial Intelligence Review, vol. 24, no. 1, pp. 33–60, September 2005, doi:10.1007/s10462-004-0041-5.

[6] T. Dohi and T. Uemura, "An adaptive mode control algorithm of a scalable intrusion tolerant architecture," Journal of Computer and System Sciences, vol. 78, no. 6, pp. 1751–1774, November 2012, doi:10.1016/j.jcss.2011.10.022.

[7] P. Mukherjee and S. Sen, "Comparing reputation schemes for detecting malicious nodes in sensor networks," The Computer Journal, vol. 54, no. 3, pp. 482–489, March 2011, doi:10.1093/comjnl/bxq035.

[8] S.X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," Applied Soft Computing, vol. 10, no. 1, pp. 1–35, January 2010, doi:10.1016/j.asoc.2009.06.019.

[9] Sourcefire, Inc., "Snort: An open source network intrusion prevention and detection system," http://www.snort.org.

[10] G.P. Spathoulas and S.K. Katsikas, "Reducing false positives in intrusion detection systems," Computers & Security, vol. 29, no. 1, pp. 35–44, February 2010, doi:10.1016/j.cose.2009.07.008.

[11] J.-T. Oh, S.-K. Park, J.-S. Jang, and Y.-H. Jeon, "Detection of DDoS and IDS evasion attacks in a high-speed networks environment," International Journal of Computer Science and Network Security, vol. 7, no. 6, pp. 124–131, June 2007.

[12] M. Gen, R. Cheng, and L. Lin, "Network models and optimization: Multiobjective genetic algorithm approach," Springer Publishing Company, Incorporated, 2008, doi:10.1007/978-1-84800-181-7.

[13] J.E. Tapiador and J.A. Clark, "Learning autonomic security reconfiguration policies," Proceedings of the 10th IEEE International Conference on Computer and Information Technology, pp. 902–909, June 2010, doi:10.1109/CIT.2010.168.

[14] D. De Catanzaro and J.C. Taylor, "The scaling of dispersion and correlation: A comparison of least-squares and absolute-deviation statistics," British Journal of Mathematical and Statistical Psychology, vol. 49, no. 1, pp. 171–188, May 1996, doi:10.1111/j.2044-8317.1996.tb01081.x.

[15] Y. Sun, Z. Han, and K.J.R. Liu, "Defense of trust management vulnerabilities in distributed networks," IEEE Communications Magazine, vol. 46, no. 2, pp. 112–119, February 2008, doi:10.1109/MCOM.2008.4473092.

[16] T.D. Huynh, "Trust and reputation in open multi-agent systems," Ph.D. thesis, University of Southampton, Electronics and Computer Science, June 2006.

[17] J.A. Clark and J.L. Jacob, "Two-stage optimisation in the design of boolean functions," Proceedings of the 5th Australasian Conference on Information Security and Privacy, vol. 1841 of Lecture Notes in Computer Science, pp. 242–254, July 2000, doi:10.1007/10718964_20.

[18] J.E. Tapiador, J.A. Clark, and J.C. Hernández-Castro, "Non-linear cryptanalysis revisited: Heuristic search for approximations to S-boxes," Proceedings of the 11th IMA International Conference on Cryptography and Coding, vol. 4887 of Lecture Notes in Computer Science, pp. 99–117, December 2007, doi:10.1007/978-3-540-77272-9_7.

[19] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671–680, May 1983, doi:10.1126/science.220.4598.671.

[20] J.E. Tapiador, J.C. Hernández Castro, J.A. Clark, and S. Stepney, "Highly entangled multi-qubit states with simple algebraic structure," Journal of Physics A: Mathematical and Theoretical, vol. 42, no. 41, pp. 415301, October 2009, doi:10.1088/1751-8113/42/41/415301.

[21] T. Gamer, "Collaborative anomaly-based detection of large-scale internet attacks," Computer Networks, vol. 56, no. 1, pp. 169–185, January 2012, doi:10.1016/j.comnet.2011.08.015.

[22] S.R. Snapp, J. Brentano, G.V. Dias, T.L. Goan, L.T. Heberlein, C.-L. Ho, K.N. Levitt, B. Mukherjee, S.E. Smaha, T. Grance, D.M. Teal, and D. Mansur, "DIDS (Distributed Intrusion Detection System) - Motivation

architecture, and an early prototype," Proceedings of the 14th National Computer Security Conference, pp. 167–176, October 1991.

[23] SANS Internet Storm Center, "DShield," http://www.dshield.org.

[24] R.A. Kemmerer, "NSTAT: A model-based real-time network intrusion detection system," Technical Report 1997-18, University of California, Santa Barbara, November 1998.

[25] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle, "GrIDS-A graph-based intrusion detection system for large networks," Proceedings of the 19th National Information Systems Security Conference, pp. 361–370, October 1996.

[26] P.A. Porras and P.G. Neumann, "EMERALD: Event monitoring enabling responses to anomalous live disturbances," Proceedings of the 20th National Information Systems Security Conference, pp. 353–365, October 1997.

[27] A.K. Ganame, J. Bourgeois, R. Bidou, and F. Spies, "A global security architecture for intrusion detection on computer networks," Computers & Security, vol. 27, no. 1-2, pp. 30–47, March 2008, doi:10.1016/j.cose.2008.03.004.

[28] V. Yegneswaran, P. Barford, and S. Jha, "Global intrusion detection in the DOMINO overlay system," Proceedings of Network and Distributed Security Symposium, February 2004.

[29] M.E. Locasto, J.J. Parekh, A.D. Keromytis, and S.J. Stolfo, "Towards collaborative security and P2P intrusion detection," Proceedings of the 6th Annual IEEE SMC on Information Assurance Workshop, pp. 333–339, June 2005, doi:10.1109/IAW.2005.1495971.

[30] G. Garrison, R.L. Wakefield, X. Xu, and S.H. Kim, "Globally distributed teams: The effect of diversity on trust, cohesion and individual performance," ACM SIGMIS Database, vol. 41, no. 3, pp. 27–48, August 2010, doi:10.1145/1851175.1851178.

[31] T.-P. Liang, J.C.-H. Wu, J.J. Jiang, and G. Klein, "The impact of value diversity on information system development projects," International Journal of Project Management, vol. 30, no. 6, pp. 731–739, August 2012, doi:10.1016/j.ijproman.2011.11.006.

[32] E. Totel, F. Majorczyk, and L. Mé, "COTS diversity based intrusion detection and application to web servers," Recent Advances in Intrusion Detection, vol. 3858 of Lecture Notes in Computer Science, pp. 43–62, September 2005, doi:10.1007/11663812_3.

[33] K. Bartos and M. Rehak, "Trust-based solution for robust self-configuration of distributed intrusion detection systems," Proceedings of the 20th European Conference on Artificial Intelligence, pp. 121–126, August 2012, doi:10.3233/978-1-61499-098-7-121.

[34] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," Decision Support Systems, vol. 43, no. 2, pp. 618–644, March 2007, doi:10.1016/j.dss.2005.05.019.

[35] H. Chen, Y.B. Al-Nashif, Q. Guangzhi, and S. Hariri, "Self-configuration of network security," Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference, pp. 97–110, October 2007, doi:10.1109/EDOC.2007.45.

[36] V. Stankovic and L. Strigini, "A survey on online monitoring approaches of computer-based systems," Technical Report, Centre for Software Reliability, City University London, London, UK, June 2009.

[37] Distributed Management Task Force, Inc., "Web-Based Enterprise Management (WBEM)," http://www.dmtf.org/standards/wbem.

[38] I. Brahmi, S.B. Yahia, and P. Poncelet, "A Snort-based mobile agent for a distributed intrusion detection system," Proceedings of the 2011 International Conference on Security and Cryptography, pp. 198–207, July 2011.

[39] F. Gómez Mármol and G. Martínez Pérez, "Towards pre-standardization of trust and reputation models for distributed and heterogeneous systems," Computer Standards & Interfaces, vol. 32, no. 4, pp. 185–196, June 2010, doi:10.1016/j.csi.2010.01.003.

[40] M. Gil Pérez, F. Gómez Mármol, G. Martínez Pérez, and A.F. Skarmeta Gómez, "RepCIDN: A reputation-based collaborative intrusion detection network to lessen the impact of malicious alarms," Journal of Network and Systems Management, vol. 21, no. 1, pp. 128–167, March 2013, doi:10.1007/s10922-012-9230-8.