Bio-inspired enhancement of reputation systems for intelligent environments

Zorana Banković , David Fraga, José Manuel Moya, Juan Carlos Vallejo, Pedro Malagón, Álvaro Araujo, Juan-Mariano de Goyeneche, Elena Romero, Javier Blesa, Daniel Villanueva, Octavio Nieto-Taladriz

A B S T R A C T

Providing security to the emerging field of ambient intelligence will be difficult if we rely only on existing techniques, given their dynamic and heterogeneous nature. Moreover, security demands of these systems are expected to grow, as many applications will require accurate context modeling. In this work we propose an enhancement to the reputation systems traditionally deployed for securing these systems. Different anomaly detectors are combined using the immunological paradigm to optimize reputation system performance in response to evolving security requirements. As an example, the experiments show how a combination of detectors based on unsupervised techniques (self-organizing maps and genetic algorithms) can help to significantly reduce the global response time of the reputation system. The proposed solution offers many benefits: scalability, fast response to adversarial activities, ability to detect unknown attacks, high adaptability, and high ability in detecting and confining attacks. For these reasons, we believe that our solution is capable of coping with the dynamism of ambient intelligence systems and the growing requirements of security demands.

1. Introduction

Ambient intelligence (AmI) is a vision of the future Information Society, where people are surrounded by an electronic environment that is sensitive to their needs, personalized to their requirements, anticipatory of their behavior, and responsive to their presence. An example is a hotel room that can adapt automatically to its customer's favorite room temperature and choice of music, and whose door opens automatically when the customer holds the doorknob. These context awareness requirements raise a number of interesting and challenging questions:

- How can we build a reliable context model based on data obtained from inherently insecure sensors [37] spread throughout the environment? Failure to recognize the customer may not only lead to annoying trivial situations like unwanted music, but it could also open the door to a dangerous intruder.
- How can we make this intelligent environment at least as secure as the previous naive environment? How can we avoid attacks in order to have a significant impact on the behavior of the intelligent environment? An attacker could inject

synthetic data to manipulate the context model in order to get a desired behavior from the environment. How can we prevent an attacker from manipulating context information?

To make AmI real, reliable context modeling is indispensable. High-quality context information must be available to any user, anytime, anywhere, and on any lightweight device. The use of reputation systems has been proposed as a feedback mechanism in order to take advantage of the spatial and temporal redundancy and thus discard faulty or manipulated data [14,35].

Since the collective opinion in a community determines an object's reputation score, reputation systems represent a form of collaborative sanctioning and praising. A low score represents the collaborative sanctioning of an object that the community perceives as having or providing low quality. Similarly, a high score represents the collaborative praising of an object that the community perceives as having or providing high quality. Reputation scores change dynamically as a function of incoming ratings. A high score can quickly be lost if rating entities start providing negative ratings, just as it is possible for an object with a low score to recover and regain a high score.

Extensive research [2,14,20,22,27,29,40] has been done on modeling and managing trust and reputation. It has been demonstrated that rating trust and reputation of individual nodes is an effective approach in distributed environments to improve security, support decision-making and promote node collaboration.

To enhance the security of reputation systems, a set of unsupervised learning algorithms [1] has been proposed to detect statistical anomalies in the environment, and to feed refined trust information back to the reputation systems [6,32]. In this way, well-behaved nodes will have more influence when inferring context information from the environment. Examples of these algorithms are self-organizing maps (SOM) [25,31] and genetic algorithms (GA) [6,36] proposed by us. On the one hand, SOMs are relatively fast and inexpensive when the dimensionality of the data is huge, they do not significantly depend on the underlying model of data and have no restrictions concerning data dimensionality. On the other hand, genetic algorithms are more robust to both parameter changes and the presence of large amounts of malicious data in the training and testing datasets.

These algorithms can be adjusted to get faster detection times, reduce the impact of an attack in the system or increase their ability to isolate attacks, while the best performing algorithm depends on the attack and the specific requirements at that moment. Therefore, it is best to have several concurrent algorithms working simultaneously and let the system evolve by taking into account only the best algorithms at any given time. The aim of this paper is to address the composition of different anomaly detectors in order to enhance reputation system performance when used to build accurate context information.

Although a potential attacker has a huge range of possibilities, due to the sensor redundancy that is common in AmI scenarios, it is reasonable to assume that the adversary will be able to manipulate only a limited number of nodes. Also due to spatial and temporal redundancy, any change in the data provided to the intelligent environment will conflict with the data provided by other sensors. For example, an attacked intelligent camera could notify the presence of a customer at the door of the room, but that information would not be coherent with the tracking information coming from many other sensors.

To summarize, we focus on those AmI applications that require accurate context information, and we consider those attacks that generate statistically identifiable differences in the data used to infer the context information. This is not a strong requirement as most identity, integrity, and denial-of-service (DoS) attacks [39] generate differences that can be used to detect and isolate them, as illustrated in Table 1.

We present a framework for intelligent environments that combines four bio-inspired techniques: reputation systems, genetic algorithms, self-organizing maps, and immunological systems. This framework can be used to ensure the reliability of the information used to build and refine the context model. Anomalies and attacks are detected and isolated, and the information sent by attacked nodes is discarded for context inferences, but other recovering actions are out of the scope of this paper.

The rest of this paper is organized as follows. Section 2 lists some previous related works. In Section 3 we provide the characterization framework for the applications being considered, which allows for the selection of the most appropriate techniques according to the dynamic application requirements. Section 5 describes the proposed distribution scheme, based

Table 1
Effect of attacks in the sensed data that can be analyzed.

Attack	Effect
Identity attacks	
Clone	Increased activity, anomalous routes
Thief, mole, sybil	Incoherency with past data and data from other nodes
Denial-of-service attacks	
Jamming, collision, flooding	Anomalous network activity
Neglect and greed	Decreased activity from some nodes
Misdirection, blackholes, wormholes	Anomalous routes, variations in local activity
Integrity attacks	
Tampering, homing	Incoherency with past data and data from other nodes

on the immune systems paradigm. Section 6.2.2 discusses some simulation results. Finally, Section 7 presents conclusions and future work.

2. Related work

The common properties of AmI (distributed system of autonomous self-aware entities, mostly loosely coupled) make them very similar to biological systems in which there is no previous information about the system. Also, they are composed of so many entities that it is not possible to control every entity in the network. It implies that the most deterministic algorithms do not work in these kinds of environments. However, bio-inspired techniques have successfully been applied in these systems. Some examples of these are self-assembly, self-replication and self-repair nodes [8], biological movement [18], cellular information exchange [19] and ants' behavior [26].

One of the bio-inspired mechanisms commonly used in WSN are reputation systems. These are based on reputation and trust concepts to ensure a minimal failure tolerance. Reputation systems are not only used in WSNs but also in P2P networks, e-commerce and multi-agent systems. In ambient intelligence they are proposed for securing routing protocols [12,13,16,28,33,37], detecting intrusions [24,38], improving decision making [32] and other more complex approaches such as CONFIDANT [11] whose main idea is to make any misbehavior unattractive, etc.

Very different algorithms for aggregating and calculating trust and reputation have been proposed: Bayesian networks [40,45], fuzzy logic [2,29] and bio-inspired [22,44].

Unsupervised algorithms have been also used to manage reputation and trust values: SOMs [6,32], GAs [6].

Both Bayesian approaches cited above rely on the beta distribution for calculating reputation values. In our previous work [32] we have demonstrated that our proposal is superior to this one when it comes to the sensor redundancy necessary for the system to work properly, as we provide similar results with lower sensor redundancy. Wang et al. [45] further use Bayesian networks in order to derive the final reputation based on multiple values covering different aspects of the behavior, rather than hand-crafted weighed sum as there is no certainty that it will reflect the real distribution. However, the Bayesian approach is supervised and needs a higher level of human interaction, which is something we are trying to avoid in this work.

The above-mentioned fuzzy systems use fuzzy logic in order to provide finer granularity of the reputation value. In other words, they do not have one threshold value for the reputation that differentiates good and bad peers, but for example nodes can be also uncertain or "possibly" bad or good. This can be helpful in the cases we want to include the notion of risk. In this way, the interaction with uncertain or "possibly bad" nodes would introduce higher risk, but might be necessary if there are no good nodes in the vicinity. This approach is not applied in the work presented in this paper, but it can be used as a possible extension.

Ant colony and swarm based approaches [44] in essence provide the optimal path towards reputation evidence and the way to update the reputation values along the path that depend on the collected evidence. This approach can be an interesting solution for integrating second-hand reputation information. However, if an adversary manages to incorporate itself as (at least) one participant in the path, he would be able to change the information forwarded to the nodes that follow, thus deteriorating the performance of the reputation system. In our work the reputation evidence is obtained in a straightforward way from the closest vicinity. Furthermore, the second-hand reputation integration is excluded in order to avoid vulnerability to attacks such as bad mouthing [15] or ballot stuffing [9].

The other cited approach based on ant colonies [35] assigns reputation values to routing paths and therefore improves routing in distributed networks. Although this approach provides robustness to routing in distributed environments, which is one of its core protocols, it does not assume that trust is multi-dimensional.

In this work, we have introduced immunological algorithms to combine the unsupervised algorithms mentioned in this section in order to enhance reputation system performance. We chose not to include supervised algorithms as the need for supervision could introduce new vulnerabilities, such as labelling training data which is error-prone.

3. Application characterization

In order to characterize an application in an intelligent environment in terms of the role of a reputation system, for the purpose of enabling or providing some of its responsibilities, the most common workflow is:

- Application definition in terms of functional and non-functional requirements, such as responsibilities, use cases, etc.
- Identification of the reputation system's specific features based on the previously defined application requirements.
- Adjustment and optimization of the reputation system's response to these requirements, modifying the reputation algorithms, the reputation system's communication protocols or even the deployment of the reputation system over the nodes of the AmI system responsible for the final application.

Obviously, this process is not optimal in terms of reducing development time, minimizing system errors and improving error tolerance. However, this issue has been analyzed in many other areas, such as classic software development, where high-level design paradigms such as patterns-based design allow developers to deal with complex systems and the development process without difficulty.

In particular, the interest of having this kind of generic high-level abstraction model for analyzing and depicting AmI applications is justified if we keep in mind the following ideas:

- Being able to identify a number of generic features or parameters which would allow us to characterize an AmI application based on the requirements imposed on a reputation system, would make it easier for us to identify "patterns" of "standard-applications". Studying AmI systems based on these patterns will allow us to find out the architectures that fit the needs of the system in a more appropriate way, as well as identify and recognize the most common difficulties in every kind of application, etc.
- Thanks to this pattern-based depiction and modeling we can create a knowledge foundation. This knowledge could allow us to face the resolution of new situations in an easier way and without the need of carrying out any intial work.
- Having a model or a pattern of a specific problem can allow us to apply solutions proposed to solve similar problems in other application fields thanks to the fact that the abstraction level is the same (and it is high enough). For example, similar SOM-based intrusion detection techniques have been applied with success to SCADA sensor networks [31] and wireless mesh networks [4]. Both applications share similar requirements in terms of the proposed criteria, and the same techniques could also be used for any AmI application sharing similar requirements.

Once we have pointed out that having a set of mechanisms to create models and identify patterns for AmI applications could be a great tool for managing our tasks as application designers or analysts, we are going to deal with the identification and definition of criteria that will allow us to generate these generic models.

3.1. Basic analysis criteria

If we focus on the main features and characteristics of reputation systems, and based on previous work [6,32], we can identify the following criteria as appropriate ones in order to depict the requirements of an arbitrary application in an intelligent environment and, with minor modification, even depict the requirements of an arbitrary application in a generic distributed system environment.

First of all, we are going to name these criteria, then, we will justify and explain how they can be useful for modeling our system.

3.1.1. Definitions

- **Response time**. It is the elapsed time since the attack starts until it is detected, i.e., since the reputation of the ill-behaved node begins decreasing.
- **Isolation capacity**. It is the portion of ill-behaved nodes that are detected as attackers.
- **System degradation**. It is the portion of well-behaved nodes detected as attackers.

3.1.2. Characterization

Now that we have defined the criteria, we will explain in more detail their importance and the way they can be useful for system characterization. The following text is summarized in Fig. 1.

- **Minimization of the response time**. It is the determining factor for all those applications where the reputation system is being used as a detection mechanism: to detect attacks, anomalous behavior, etc.

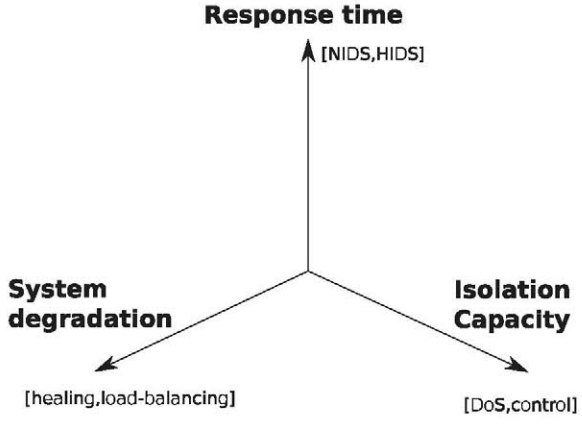


Fig. 1. Basic analysis criteria.

In other words, on many occasions the reputation system is not a sole solution for protection against the attacks or detection of anomalous behavior; it is rather a tool for fast and efficient first reaction that further delegates the response and the reaction to these events to other mechanisms.

A typical example of applications that deploy a reputation system to minimize this factor are the IDSs (Intrusion Detection Systems), both NIDS (Network IDS) and HIDS (Host IDS). For example, there are cases where SOM algorithms have been used to this end [30] providing good performance.

- **Maximization of the isolation capacity**. It is the determining factor in all those applications where few elements can cause a lot of damage to the system. The damage can be twofold: they can have a negative effect on other nodes, either in degrading their performance or even rendering them completely useless. On the other hand, these elements may provide critical information or critical functionality and their incorrect functioning can highly degrade the performance of the complete system. This is the typical case where these nodes are responsible for making decisions or they can manipulate the management information.

  Due to the last point, this characteristic is very important when we deal with "social" distributed environments. In other words, when we apply reputation systems for management of equipment or organizations, social networks, executive committees, etc., since in all these systems the act of making decisions is a fundamental factor while the influence of certain nodes on the activities of others can be very high. The collective intelligence of the users should definitely be integrated into the AmI systems. Individual user decisions are usually taken into account, but we firmly believe that a kind of "socialization" of the people-environment interaction could boost the utility of AmI infrastructures.

  A typical example of applications that use a reputation system to maximize this factor can be systems for contingency against DoS or DDoS [4] where we can see how the reputation system prevents further propagation of the damaging effects of the attack and eventually renders it ineffective.
- **Minimization of the system degradation**. It is the determining factor in cases where it is more important to have as many nodes functioning properly as possible. This added value can be of three different types: the nodes provide a higher data throughput to the system, higher processing capacity to the system or stronger validity to the information generated in the system.

  A typical example of applications that use a reputation system to maximize performance can be auto-healing or even load-balancing systems, both highly deployed in every kind of distributed systems such as WMN [4].

Therefore, relying on these three factors we are able to model and characterize any application in a distributed environment in terms of the role of the reputation system in these applications. However, it is important to point out that this analysis is not static, as the requirements of many applications evolve over time, depending on their current state.

An example that can illustrate this fact can be a reputation system that supports security-related applications in the intelligent environment.

In this context, in an initial normal state we would need the reputation system to present some optimal characteristics in terms of its "response time", in order to be able to detect anomalies (attacks in this case) in a timely fashion. However, if attacks occur, the priority of the reputation system can become to isolate the attackers in the most efficient way, so its behavior should be optimized in terms of its "isolation capacity". Finally, once having the attack isolated, the priority of the reputation system can become the restoration of the system using the tasks of healing, load balancing, etc. In this way, we would be in the situation where its behavior should be optimized in terms of minimization of the "System Degradation"/Maximization of performance.

Therefore, it is important to point out that these three factors characterize the requirements of an application, not permanently, but as a function of time. So, a fundamental requirement for a reputation system is not only to be able to optimize its functioning in terms of one of these factors, but also to be able to do it in an adaptive manner.

4. Detection of adversarial activities

We treat attacks as data outliers and deploy clustering techniques, namely SOM and unsupervised GA. Further details on the algorithm implementation can be found in [5,6]. In the following we will explain the principles of the approach. It is important to mention here that the algorithms can be trained with both clean and "unclean" data (contain traces of attacks). Furthermore, the algorithms are constantly retrained in order to decrease time lags between model training and model application. The retraining frequency depends on the dynamics of the underlying system.

There are two approaches for detecting outliers using clustering techniques [34] depending on the following two possibilities: detecting outlying clusters or detecting outlying data that belong to non-outlying clusters. For the first case, we calculate the average distance of each cluster to the rest of the clusters (or its closest neighborhood) (MD). In the latter case, we calculate quantization error (QE) of each input as the distance from its corresponding cluster center.

Hence, a cluster whose MD is greater than those of the rest of the clusters is considered to be outlying and all the inputs that belong to it are considered to be outliers, i.e. anomalies. On the other hand, even if the cluster to which the current input belongs is not outlying, e.g. outlying data is too sparse so the formation of outlying node (s) is not possible, if its QE value is much bigger than the rest of the QE values corresponding to the same cluster, it is considered to be the proof of the anomaly of the current input.

5. Collective detection: distributed organization of agents based on immune system foundation

5.1. Immune system overview

As already mentioned, in this work the basic idea of immune systems is used for establishing the distributed organization of detectors (implemented as software agents), inspired by human immune system. The immune system is the body's natural defense against invaders, such as viruses or bacteria. Through a series of steps called the immune response, the immune system attacks organisms and substances that can damage the body.

The immune system is made up of a network of cells, tissues, and organs that work together to protect the body. The cells involved are white blood cells, or leukocytes, which come in two basic types that colaborate to seek out and destroy disease-causing organisms or substances. The two basic types of leukocytes are phagocytes, cells that destroy invading organisms, and lymphocytes, cells that allow the body to remember and recognize previous invaders (B lymphocytes) and help the body destroy them (T lymphocytes). Our system is inspired by the lymphocytes and their way of functioning. Lymphocytes grow in the bone marrow and either stay there and mature into B cells, or they move onto the thymus gland, where they mature into T cells. B lymphocytes and T lymphocytes have separate functions: B lymphocytes seek out the invaders and send the appropriate defenses, while T cells destroy the identified invaders.

The immune system functions in the following way: when antigens (foreign substances that invade the body) are detected, the immune system triggers the B lymphocytes to produce antibodies, which are specialized proteins that lock onto specific antigens. Once produced, these antibodies continue to exist in the body, so that if the same antigen is presented to the immune system again, the antibodies are already there to do their job. However, the antibodies are not capable of destroying antigens by themselves. That is the function of the T cells, which are part of the system that destroys antigens tagged by antibodies or cells that have been infected or somehow changed.

The idea of lymphocytes and their maturation, validation, distributed organization with high redundancy and detection of antigens, i.e. intruders, has been used to design the distributed model of detecting agents in this work. More details are given in the next section.

5.2. Distributed organization of agents

Machine learning techniques have many parameters that should be set from the start, e.g. duration of training, size of the lattice in the case of SOM, crossover and mutation probabilities in the case of GA, etc. It is not easy to guess the optimal parameters a priori, and in our case an additional problem is the impossibility of human interaction. Moreover, in the case where an agent resides on a compromised node, it is possible for the attacker to compromise the agent as well. However, we consider that additional security measures that protect the agent from the host (and vice versa) are taken, such as those proposed in [21], so agent subversion is not a straightforward process. The detailed explanation of these techniques is out of the scope of this work.

In order to overcome these issues, we introduce agent redundancy, where more than one agent monitors the same node, and we adopt the idea of immune systems in order to establish the distributed detection model. According to this model, each physical node may contain more than one agent. In the beginning we have a group of agents that implement one of the proposed algorithms with different parameter settings. Every node is being examined by an agent that resides on another node in its vicinity and which promiscuously listens to its communication. Each of the agents is trained separately, which resembles the basic idea of immune systems [41]: all the processes of detector (agent in our case) generation and elimination of antibodies, i.e. intruders, are parallel and distributed.

Additionally, we assign a reputation to each agent. We have opted for beta reputation [20], since it has a strong background in the theory of statistics. Besides this, the research community has used techniques such as entropy [27]. The reputation is calculated according to the following formula:

$$R = E(Beta(\alpha + 1, \beta + 1)) = \frac{\alpha + 1}{\alpha + \beta + 2}, \tag{1}$$

where α stands for the number of correct decisions made by the detector, while β stands for the number of incorrect ones. The voting system decides whether a response is right or wrong based on majority voting. In this way we get a hierarchical reputation system, as depicted in Fig. 2 (S_i stands for a node that is monitored by more than one agent (A_j), four in this case, and each agent has its own reputation calculated by RA_j, which uses the information from the rest of the agents that monitor the same node to calculate the reputation of A_j). Here we assume that the majority of the agents are capable of making correct decisions. The software for calculating agent reputation can be executed in either the base station or nodes since it does not consume significant resources. The implementation in nodes has the advantage of a faster response, but its limitation consists in being capable of only seeing the response of those agents whose communication it can overhear.

It has been mentioned that the agents are considered to include security measurements that impede attackers from tampering with them. However, if we assume that the attacker is powerful and skillful enough to compromise the agent,

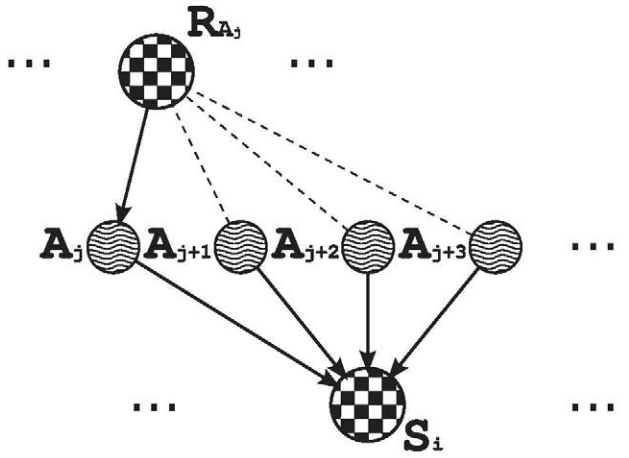


Fig. 2. Hierarchical organization of reputation systems.

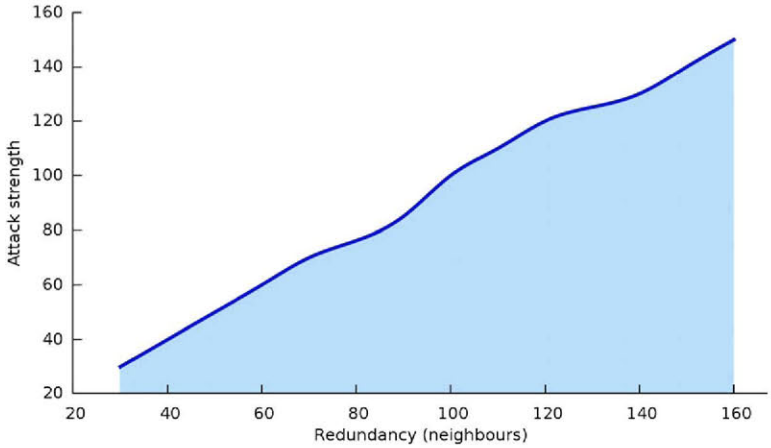


Fig. 3. Attack effort versus redundancy.

agent redundancy makes it more complicated for him to compromise the whole detection system. In the Fig. 3 the effort the attacker has to make (expressed in the number of nodes the attacker has to compromise) versus the existing redundancy is depicted. This dependency is almost linear, which means that the attacker has to compromise almost all the agents that monitor a single node in order to compromise its defense.

Each agent has to pass through a period of validation: as soon as its reputation becomes higher than an established threshold value, the agent can participate in the detection process, as this demonstrates that it was properly trained. From that point onwards it participates fully in assigning reputation values to the nodes it is monitoring. The validation process resembles the process of maturation of the lymphocytes in the immune system. Once a detector from the immune system detects an antigen, it activates the substances that eliminate the antigen. In a similar fashion, once a validated agent identifies a compromised node, it decreases the reputation of the node, which will help in isolating it from the network.

We continue to calculate agent reputations and in certain moments we check if they are above the established threshold. When the value is below the threshold, it means that either the situation in the environment has changed significantly, so the agent needs re-training, or an adversary has deliberately changed it. In either case, the agent must not be able to change the reputation values of the nodes. Additional measures should be taken to secure the learning process, such as those given in [7], but the details are out of the scope of this paper.

The threshold value can be set to the middle of the reputation value range (50 in our case) at the starting point of launching the detection agents. However, this value depends on many different factors. One of the most important factors is risk, and the threshold value is proportional to it: if the operation in the network (or in some of its parts) is critical, the threshold value should be higher, and vice versa. Thus, a process that evaluates risk should be able to update the threshold value. On the other hand, the frequency of calculating the reputation of the detectors, as well as the frequency of performing

re-training, depends on the dynamics of the underlying sensor network. In the cases where the network is observing a process that is changing slowly, the corresponding frequencies are low, and vice versa.

Concerning the characteristics of the system the proposal can be applied on, we have demonstrated that our system performs sufficiently well even in systems with low spatial redundancy, i.e. using one agent per entity [6,32]. Spatial redundancy is introduced in order to provide a higher level of accuracy. Similarly, agent redundancy provides additional robustness to attacks and also additional accuracy. Of course, this implies the cost of higher communication overload. Thus, the presented detection system can work even if the system does not provide high spatial redundancy, but it can benefit from its existence. The redundancy applied in the detector system will depend on the system characteristics, i.e. the amount of resources the system can provide for the incorporation of the proposed detector, as well as the level of security that has to be maintained.

The proposed approach offers many advantages. First of all, it provides proper training of the agents without human interaction. Furthermore, the agent redundancy makes it more robust to attacks against the reputation system itself. Finally, one of the most important advantages of this solution is its scalability. Namely, the addition of new nodes in the network requires the activation of new agents that will run on the nodes in its vicinity and examine their behavior. As a comparison, if we choose to execute all the agents on the base station, the addition of new nodes (which implies addition of new detection agents) introduces an extra load to the base station that could lead to the deterioration of its functionality. Thus, it is obvious that our solution provides much better scalability. Moreover, the sole implementation in the base station would mean the single point of failure, so in this case our solution provides higher robustness as well.

6. Experimental evaluation

6.1. Study case – Sensor networks

6.1.1. Feature extraction and formation of model

Our idea is to find temporal and/or spatial inconsistence in sensed data in order to detect manipulated data and/or compromised nodes. For this reason, we follow the idea presented in [5,6,30] based on extracted n-grams and their frequencies within different time windows. Thus, the vectors used for characterization that allow the deployment of machine learning are composed of the extracted n-grams. For the purpose of illustration, we will give a short example for a sensor that detects presence. Let a sensor give the following output during the time window of size 20: 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0. If we fix the n-gram size on 3, we extract all the sequences of size 3 each time moving one position forward. In this way we can observe the following sequences and the number of their occurrences within the time window: 111 – occurs 6 times, 110 – 2, 100 – 2, 000 – 6, 001 – 1, 011 – 1. Their corresponding frequences are the following: 111 – 0.33, 110 – 0.11, 100 – 0.11, 000 – 0.33, 001 – 0.06, 011 – 0.06. In our model, the sequences are the features and their frequencies are the corresponding feature values. Thus, the sum of the feature values is always equal to 1. In our algorithm this characterization is performed in predefined moments of time and takes the established amount of previous data, e.g. we can perform the characterization after every 20 time periods based on previous 40 values.

In a similar fashion, we form features for spatial characterization. The first step is to establish vicinities of nodes that historically have been giving consistent information. Furthermore, since an agent is supposed to reside on a node, vicinities are established using the nodes whose information can reach the agent. In this way, an n-gram for spatial characterization in a moment of time is made of the sensor outputs from that very moment. For example, if sensors S1, S2, S3 that belong to the same group each give the following output: 1 1 1 0 during four time epochs, we characterize them with the following set of n-grams (each n-gram contains at the first position the value of S1, the value of S2 at the second and the value of S3 at the third at a certain time epoch): 111 – occurs 3 times, 000 – occurs once, thus the feature value of each n-gram is: 111 – 0.75, 000 – 0.25, i.e. the frequencies within the observed period of time.

6.1.2. Reputation by GA/SOM

The system of agents is coupled with a reputation system where each node has its reputation value that basically reflects the level of confidence that others have in it based on its previous behavior. In our proposal, the output of an agent affects on the reputation system in the way that it assigns lower reputation to the nodes where it detects abnormal activities and vice versa. We further advocate avoiding any kind of interaction with the low-reputation nodes: to discard any data or request coming from these nodes or to avoid taking them as a routing hop. In this way, compromised nodes remain isolated from the network and have no role in its further performance. After this, additional actions can be performed by the base station, e.g. it can revoke the keys from the compromised nodes, reprogram them, etc.

In this work the reputation is calculated in the following way. Having in mind that the attacks will often result in creating new n-grams, it is reasonable to assume that the extracted vector of n-grams in the presence of attackers will not be a subset of any vector extracted in normal situation, thus the distance will never be lower than 1. For this reason, suspicious values of both *QE* and *MD* are those greater than 1. We further define two reputation values, *repQE* and *repMD* based on the previously defined *QE* and *MD* values and afterwards joint reputation *rep* used for updating overall reputation based on these two values:

```
if (QE < 1)
  repQE = 1;
else
  repQE = 1 − QE/2;
if (MD < 1)
  repMD = 1;
else
repMD = 1 − MD/2;
```

The value (*rep*) for updating overall reputation is calculated in the following way:

```
if (QE > 1)
  rep = repQE;
else
  rep = repMD;
```

There are two functions for updating the overall reputation of the node, depending whether the current reputation is below or above the established threshold that distinguishes normal and anomalous behavior. If the current reputation is above the threshold and the node starts behaving suspiciously, its reputation will fall quickly. On the other hand, if the reputation is lower than the established threshold, and the node starts behaving properly, it will need to behave properly for some time until it reaches the threshold in order to "redeem" itself. The first objective is provided by the function $x + log(1.2x)$. Finally, the reputation is updated in the following way:

```
if (last_rep[node] > threshold) {
  new_rep[node] = last_rep[node]
    + rep + log (1.2 * rep);
} else {
  new_rep[node] = last_rep[node]
    + c_limit * (rep + log (1.2 * rep));
}
```

The second objective is provided by the coefficient *c_limit*, which takes values lower than 1 and its purpose is to limit selective behavior of a node by decreasing the reputation growth if the reputation value is below the threshold. Very low values of this coefficient obligate nodes to behave properly most of time. For this reason, we are taking values up to 0.05 in our experiments, which obligates the nodes to behave properly during at least 95% of the time. If the final reputation value falls out from the $[0, 1]$ range, it is rounded to 0 if it is lower than 0 or to 1 in the opposite case.

However, if during the testing of temporal coherence, we get normal data different from those that the clustering algorithms saw during the training, it is possible to get high *QE* value. On the other hand, the spatial coherence should not detect any anomalies. Thus, the final reputation will fall only if both spatial and temporal algorithms detect anomalies. In the opposite case, its reputation will not change. This is implemented in the following way:

```
if (value_rep < value_threshold) {
  if (space_rep < space_threshold) {
    result = value_rep;
}
  else {
    result = old_result;
  }
} else {
  result = value_rep;
}
```

where *value_rep* is the reputation assigned by the algorithms for temporal characterization and *space_rep* is the reputation assigned by the algorithms for spatial characterization. On the other hand, as mentioned in the previous text, in the situations such as the data coming from a node exhibits large variations, temporal inconsistencies are not likely to be detected. However, spatial inconsistencies are very likely to be detected. Thus, spatial inconsistence is sufficient in order to raise an alarm.

6.2. Results and discussion

6.2.1. Simulation environment

The proposed algorithm has been tested on a simulator of sensor networks developed by our research group and designed using the C++ programming language. We have decided to design a simulator mainly because there is no available testbed for security applications in sensor networks. Attacking recorded data from a testbed does not significantly differ from the simulation. What's more, the available testbeds for wireless sensor networks contain a relatively small number of sensors (100 at most), in which case the data obtained from our simulator are more complex (simply because there are more sensors). For these reasons, we believe that until a testbed for security applications in WSNs appears, a simulator is a better choice for testing the applications. We further evaluated two well-known WSN simulators, ns-2 [42] and Castalia [10,23] over Omnet++ [43]. Yet, we eventually decided to implement our own simulator, AmiSim. This simulator focuses on the logical aspect of the problem, rather than implementing all the communication layers in detail (only their basic functionality is implemented), therefore reducing the total simulation time, which was our main reason for implementing the simulator. In the following we will describe the characteristics of the simulator that are pertinent for understanding the results of this work. The experiments, however, do not rely on any AmiSim-specific data or constraint and they can be replicated easily with other simulators.

In AmiSim the configuration of the network is defined by the block Scenario which defines the number of sensors, their position in the network area and the range of their radio signal. Block World defines the surroundings of the network. The initial value that each node "senses" is read from a file, while the values that follow are defined by the block Data Type that establishes the type of data the sensors send, its range and maximal variation between two successive outputs. There are two types of servers and one policy: NameServer, where all the active nodes are registered, ReputationServer that implements various ways of calculating reputation and RoutingPolicy that can implement various routing protocols. For the purpose of simulation, the proposed algorithms are an implementation of ReputationServer. The attacks are implemented as changes in some of the blocks mentioned above, thereby altering the normal behavior of the network. Thus, due to its fast simulation time (measured in minutes) and the easiness of simulating attacks, AmiSim is a better choice for testing logical aspects of detection algorithms over conventional simulators.

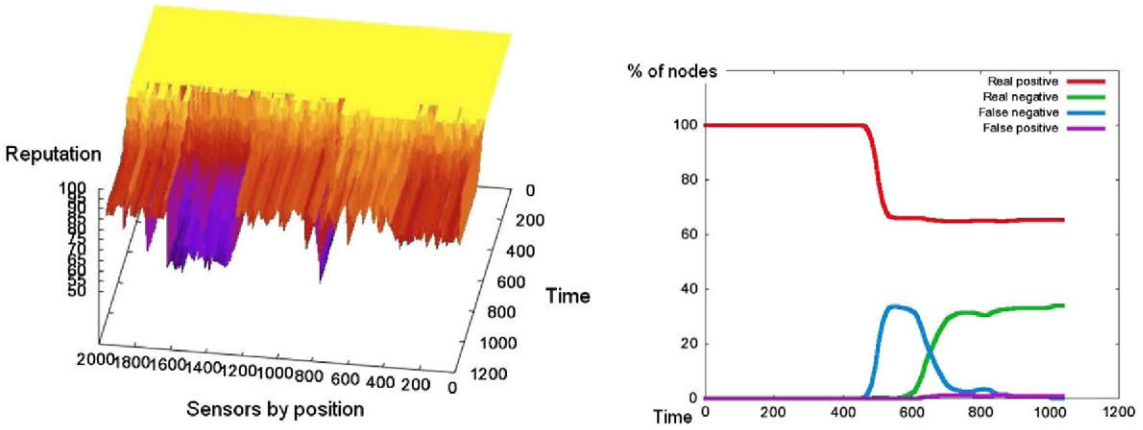
6.2.2. Results

The proposed algorithm has been tested on a simulated sensor network that contains 2000 sensor nodes attacked by a Sybil attack [17]. We have chosen Sybil due to its aggressiveness and elusiveness. Moreover, most of the internal attacks (thief, mole, clone etc.) [39] can be considered as special cases of Sybil. The network simulates a sensor network for detection of presence in the area of application. In other words, sensors give either output 1 if they detect presence of a person or an object, or 0 if they do not detect any presence.

The duration of the experiment is 1000 time ticks. One time tick in the simulator is the period of time required to perform the necessary operations in the network, and it is equivalent to a sensing interval in real sensor networks. In our previous papers [30] we have demonstrated the possibilities of single algorithms for detecting and confining the Sybil in the cases when it is both present and not present in the training data. We have also demonstrated that it is possible to find the set of parameters in order to have efficient detection and confinement of the attack, but these parameters have to be set by a human [3].

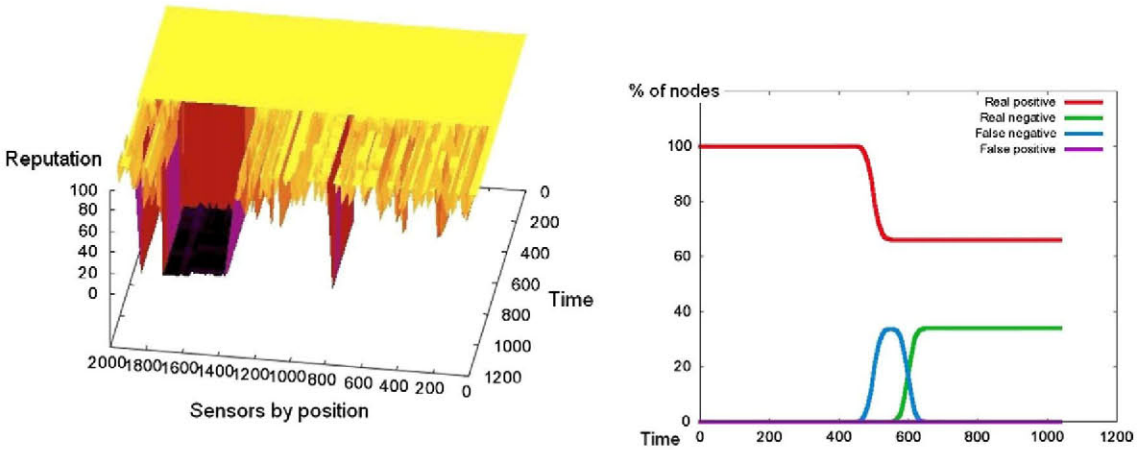
Now we will prove that relying on the immunological approach presented in this paper we can automate this process and enhance the performance of the distributed detector. What is most important, we will show that it helps the system to provide accurate context information in the presence of attacks. We start from a set of detectors that match one or more of the security requirements of our application in terms of detection time, isolation time and system degradation. Final reputation is calculated as the average value provided by all detectors. By introducing the immunological paradigm, we can obtain a global detector that performs better in certain aspects, as will be demonstrated in the following. In this particular case our goal will be to minimize the response time (detection and isolation).

In the following experiments the training ends at tick 600, Sybil starts at tick 500. New node is added at position 800, which impersonates 41existing IDs (132–171, and 180) situated at positions 1284–1652 and 1765. The node at position 800 belongs to the area that contains 40 sensor nodes. Thus, since the malicious node contains 41 IDs, it can skew the aggregation value and in this way change the context. As a reminder, the aggregation value is the value obtained by combining the data coming from different sources. Thus, if the attacker controls the majority of data, he can skew the aggregation value and change the context. In Fig. 4 we present the reputation and detection evolution assigned as the average value of a group of detectors, without introducing the immunologic paradigm. The threshold value taken for distinguishing bad and good nodes is taken to be 70 (established experimentally), as lower values cannot provide complete confinement of the attack. Fig. 4(a) represents the reputation evolution of every node, where all of them have maximal reputation in the beginning. When the Sybil attack starts, the incoherencies of the data provided by the attacked node and the data provided by its environment cause the reputation values to decrease over time. In this case, the reputation of the Sybil IDs reach very low values (the malicious node as well as the nodes whose ID has been stolen). Fig. 4(b) shows the evolution of the percentage of nodes that are correctly identified as well-behaved (real positives), the attacked nodes that are incorrectly identified as well-behaved (false positives), the attacked nodes that have been detected (real negatives), and the well-behaved nodes that are identified



(a) Reputation Evolution (b) Detection Evolution

Fig. 4. Distributed detector – Sybil starts at 500, training ends at 600.



(a) Reputation Evolution (b) Detection Evolution

Fig. 5. Distributed detector with immunologic paradigm – Sybil starts at 500, training ends at 600.

as attacked (false negatives). As can be seen, false positives are negligible, but false negatives need 400 ticks to reach a low level. Eventually, the attack is confined (using 70 as threshold).

Now we will give the results of a similar scenario (Fig. 5). Like in the previous case, a new node at position 800 impersonates 41 existing IDs (138–177,192) at positions 1397–1743 and 1866. In the same way as before, the attacker can skew the aggregation value and change the context. However, the immunological paradigm is introduced in this case, in which all the detectors with a beta reputation lower than 0.9 are discarded after the validation process. The final reputation is calculated as the average of the remaining detectors. The threshold for distinguishing bad and good nodes is 10. As can be seen in Fig. 5(b), even with much lower threshold values, the attack is completely confined much faster than in the previous case, (few ticks after starting the detection process) and the impact on the rest of the network is much lower (Fig. 5(a)).

Comparing Figs. 4 and 5 we can conclude that in both cases the attacked area, as well as the origin of the attack is distinguishable from the rest. However, in the case where the immunologic paradigm has not been used, we had to set the threshold value at 70 in order to confine the attack completely, as opposed to 10 in the case where the immunologic paradigm is present. Although the false positive rate remains low (1%), the threshold of 70 is too high and in a normal situation it is not likely to be assigned. On the other hand, in the second case, using a very low threshold we are able to confine the attack with a false positive rate of 0, which illustrates the strength and the advantages of this approach. Furthermore, having confined the attack, it will no longer affect the sensed data nor can it change the context designed on this data.

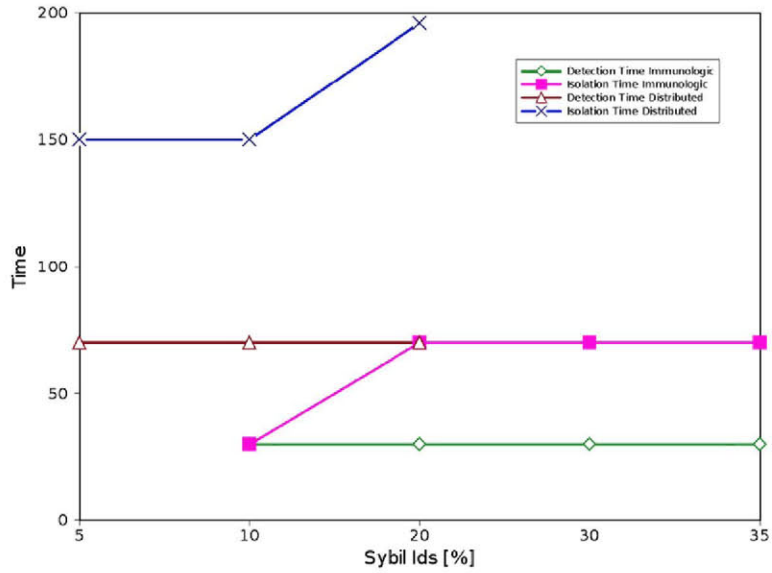


Fig. 6. Detection and isolation times.

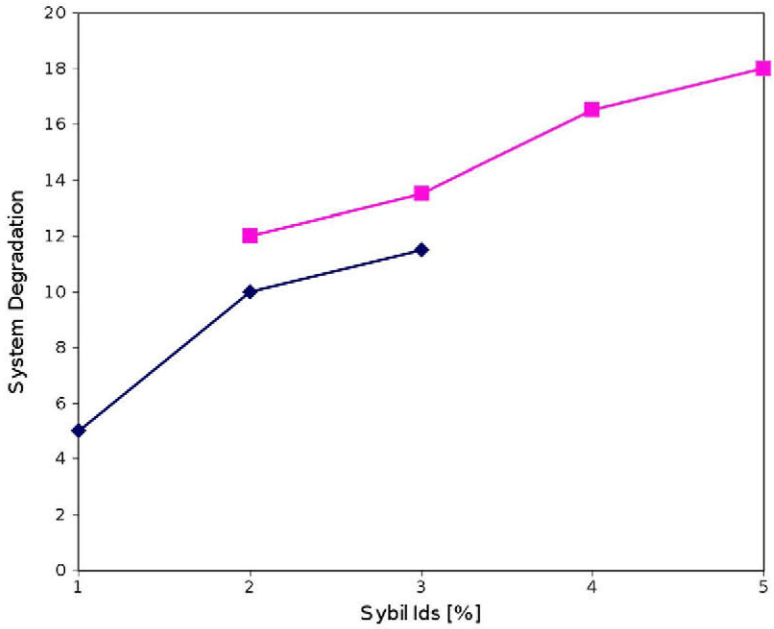


Fig. 7. System degradation.

Now we will compare both cases in terms of previously defined properties:

- Detection time: time from the launching point of the attack until the moment it has been detected.
- Isolation time: time from the launching of the attack until its complete isolation.
- System degradation: the percentage of nodes that needs to be sacrificed in order to confine the attack.

These characteristics are presented in Figs. 6 and 7, depending on the percentage of Sybil IDs, i.e. sensor IDs impersonated by Sybil. This value spans from 0 to 35% in the case where we have an immunologic algorithm and to 20% when we do not have it, since for the higher values it is not possible to detect and confine the attack completely in our test case.

We can conclude from Figs. 6 and 7 that, with the introduction of the validation process through the immunological paradigm, we are able to detect more aggressive attacks in less time and also confine them more rapidly, but at the price of higher system degradation.

This is just one example where the minimization of the response time is the key goal. However, a complete set of characterized detectors distributed over the network could be made available to the intelligent environment, and the proposed detection system could combine them in order to optimize system performance according to the evolving requirements. In case there is a high risk of attacks, the detection system will combine the detectors in order to minimize the response time. Otherwise, it will try to minimize the system degradation. But if a DoS attack is detected, it will try to maximize the isolation capacity.

7. Conclusions

In this work we have presented a bio-inspired enhancement of reputation systems for intelligent environments that is capable of coping with dynamism of ambient intelligence systems and the growing requirements of security demands. The reputation of the nodes in intelligent environments is assigned by agents that execute SOM or GA algorithms for detecting outliers in a way that well-behaving nodes are given a high reputation and vice versa. Moreover, we have proposed a distributed organization of the agents inspired by the idea of the human immune system. Our testing results demonstrate great abilities of our system in helping the system provide accurate context information even in the presence of attackers, which is indispensable for correct functioning of AmI systems.

Furthermore, we have presented the characterization of the reputation systems based on three important parameters: system degradation, response time and isolation capacity. We have proven that the reputation system can be optimized in terms of one of these parameters. Moreover, since the behavior of the reputation system occurs in a dynamic environment such as AmI applications, the same system has to be able to optimize itself according to different parameters over time. We have demonstrated that the introduction of immunological paradigm with high agent redundancy minimizes response time and isolation capacity, but at the cost of higher system degradation. Thus, varying agent redundancy we can optimize the parameters that are important in different situations.

In the future we plan to add more attacks to our attack simulator. In addition, we plan to expand our feature set with routing information and thus provide the possibility to detect the attacks that exploit the deficiencies of routing protocols.

Acknowledgements

This work was funded by the Spanish Ministry of Industry, Tourism and Trade, under Research Grant TSI-020301-2009-18 (eCID), the Spanish Ministry of Science and Innovation, under Research Grant TEC2009-14595-C02-01, and the CENIT Project Segur@.

References

[1] R.M. Aliguliyev, Performance evaluation of density-based clustering methods, Information Sciences 179 (2009) 3583–3602.
[2] F. Almenárez, A. Marin, C. Campo, C. Garcia, PTM: A pervasive trust management model for dynamic open environments, in: First Workshop on Pervasive Security, Privacy and Trust, PSPT2004 in conjuntion with Mobiquitous, volume 2004, Citeseer.
[3] Z. Banković, D. Fraga, J.M. Moya, J.C. Vallejo, Á. Araujo, P. Malagón, J.M. de Goyeneche, D. Villanueva, E. Romero, J. Blesa, Detecting and confining sybil attack in wireless sensor networks based on reputation systems coupled with self-organizing maps, in: 6th IFIP Conference on Artificial Intelligence Applications & Innovations (AIAI 2010), submitted for publication.
[4] Z. Bankovic, D. Fraga, J.M. Moya, J.C. Vallejo, P. Malagón, Á. Araujo, J.M. de Goyeneche, E. Romero, J. Blesa, D. Villanueva, O. Nieto-Taladriz, Improving security in WMNs with reputation systems and self-organizing maps, Network & Computer Applications (2010).
[5] Z. Banković, J.M. Moya, A. Araujo, D. Fraga, J.C. Vallejo, J.M. de Goyeneche, Distributed intrusion detection system for wireless sensor networks based on a reputation system coupled with kernel self-organizing maps, Integrated Computer Aided Engineering 17 (2010) 87–102.
[6] Z. Bankovic, J.M. Moya, Á. Araujo, J.M. de Goyeneche, Intrusion detection in sensor networks using clustering and immune systems, in: E. Corchado, H. Yin (Eds.), IDEAL, Lecture Notes in Computer Science, vol. 5788, Springer, 2009, pp. 408–415.
[7] M. Barreno, B. Nelson, A. Joseph, J. Tygar, The security of machine learning, Machine Learning 81 (2010) 121–148. 10.1007/s10994-010-5188-5.
[8] L. Benini, E. Farella, C. Guiducci, Wireless sensor networks: enabling technology for ambient intelligence, Microelectronics Journal 37 (2006) 1639–1649.
[9] R. Bhattacharjee, A. Goel, Avoiding ballot stuffing in ebay-like reputation systems, in: Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems, P2PECON '05, ACM, New York, NY, USA, 2005, pp. 133–137.
[10] A. Boulis, Castalia: revealing pitfalls in designing distributed algorithms in WSN, in: [23], pp. 407–408.
[11] S. Buchegger, J.L. Boudec, Performance analysis of the CONFIDANT protocol, in: Proceedings of the Third ACM International Symposium on Mobile ad hoc Networking & Computing, ACM, Lausanne, Switzerland, 2002, pp. 226–236.
[12] L. Buttyán, J. Hubaux, Enforcing service availability in mobile ad-hoc WANs, in: Proceedings of the First ACM International Symposium on Mobile ad hoc Networking & Computing, IEEE Press, Boston, MA, USA, 2000, pp. 87–96.
[13] L. Buttyán, J. Hubaux, Stimulating cooperation in self-organizing mobile ad hoc networks, Mob. Netw. Appl. 8 (2003) 579–592.
[14] J. Caverlee, L. Liu, S. Webb, The socialtrust framework for trusted social information management: architecture and algorithms, Information Sciences 180 (2010) 95–112. Special Issue on Collective Intelligence.
[15] K.T. Chen, C.C. Lou, P. Huang, L.J. Chen, Detecting bad mouthing behavior in reputation systems, in: Wikimania 2007.
[16] R. Dawkins, The Selfish Gene, second ed., Oxford University Press, New York, NY, USA, 1990.
[17] J.R. Douceur, The sybil attack, in: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, 2002, pp. 251–260.
[18] S. Doumit, D.P. Agrawal, Bio-Inspired mobility in environment aware wireless sensor networks, in: PERCOM '03: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, IEEE Computer Society, Washington, DC, USA, 2003, p. 514.

[19] F. Dressler, Network-centric actuation control in sensor/actuator networks based on bio-inspired technologies, in: 3rd IEEE International Conference on Mobile Ad Hoc and Sensor Systems (IEEE MASS 2006): Second International Workshop on Localized Communication and Topology Protocols for Ad hoc Networks (LOCAN 2006), pp. 680–684.
[20] S. Ganeriwal, L.K. Balzano, M.B. Srivastava, Reputation-based framework for high integrity sensor networks, ACM Transactions on Sensory Network 4 (2008) 1–37.
[21] M. Greenberg, J. Byington, D. Harper, Mobile agents and security, Communications Magazine, IEEE 36 (1998) 76–85.
[22] F. Gómez Mármol, G. Martínez Pérez, A.F. Gómez Skarmeta, TACS, a trust model for P2P networks, Wireless Personal Communications 51 (2009) 153–164.
[23] S. Jha (Ed.), Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, SenSys 2007, Sydney, NSW, Australia, November 6-9, 2007, ACM, 2007.
[24] I. Krontiris, T. Giannetsos, T. Dimitriou, LIDeA: a distributed lightweight intrusion detection architecture for sensor networks, in: Proceedings of the Fourth International Conference on Security and Privacy in Communication Networks, ACM, Istanbul, Turkey, 2008, pp. 1–10.
[25] P. Kumpulainen, K. Hätönen, Local anomaly detection for mobile network monitoring, Information Science 178 (2008) 3840–3859.
[26] T.H. Labella, F. Dressler, A bio-inspired architecture for division of labour in SANETs, Intelligence (SCI) 69 (2009) 211–230.
[27] H. Luo, J. Tao, Y. Sun, Entropy-based trust management for data collection in wireless sensor networks, in: WiCOM'09: Proceedings of the Fifth International Conference on Wireless Communications, Networking and Mobile Computing, IEEE Press, Piscataway, NJ, USA, 2009, pp. 3171–3174.
[28] S. Marti, T.J. Giuli, K. Lai, M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking, ACM, Boston, MA, USA, 2000, pp. 255–265.
[29] F.A. Mendoza, A.M. López, D. Diaz, J. Sanchez, Developing a model for trust management in pervasive devices, in: PerCom Workshops, IEEE Computer Society, 2006, pp. 267–271.
[30] J. Moya, Á. Araujo, Z. Banković, J. de Goyeneche, J. Vallejo, P. Malagón, D. Villanueva, D. Fraga, E. Romero, J. Blesa, Improving security for SCADA sensor networks with reputation systems and self-organizing maps, Sensors 9 (2009) 9380.
[31] J.M. Moya, Á. Araujo, Z. Banković, J.M. de Goyeneche, J.C. Vallejo, P. Malagón, D. Villanueva, D. Fraga, E. Romero, J. Blesa, Improving security for SCADA sensor networks with reputation systems and self-organizing maps, Sensors 9 (2009) 9380–9397.
[32] J.M. Moya, J.C. Vallejo, D. Fraga, A. Araujo, D. Villanueva, J. de Goyeneche, Using reputation systems and non-deterministic routing to secure wireless sensor networks, Sensors 9 (2009) 3958–3980.
[33] Y. Mun, C. Shin, Secure routing in sensor networks: security problem analysis and countermeasures, in: Computational Science and Its Applications – ICCSA 2005, Springer, 2005, pp. 459–467.
[34] A. Muñoz, J. Muruzábal, Self-organizing maps for outlier detection, Neurocomputing 18 (1998) 33–60.
[35] F. Mármol, G. Pérez, Security threats scenarios in trust and reputation models for distributed systems, Computers & Security 28 (2009) 545–556.
[36] M. Pelikan, D.E. Goldberg, S. Tsutsui, Getting the best of both worlds: discrete and continuous genetic and evolutionary algorithms in concert, Information Sciences 156 (2003) 147–171. Evolutionary Computation.
[37] A. Perrig, J. Stankovic, D. Wagner, Security in wireless sensor networks, Commun. ACM 47 (2004) 53–57.
[38] R. Roman, J. Zhou, J. Lopez, Applying intrusion detection systems to wireless sensor networks, Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE 2006, IEEE, Las Vegas, NV, USA, 2006. 640–644.
[39] T.G. Roosta, Attacks and Defenses on Ubiquitous Sensor Networks, Technical Report, University of California at Berkeley, 2008. Ph.D. Dissertation.
[40] S. Songsiri, Mtrust: A reputation-based trust model for a mobile agent system, in: [46], pp. 374–385.
[41] K. Trojanowski, S.T. Wierzchon, Immune-based algorithms for dynamic optimization, Information Sciences 179 (2009) 1495–1515. Including Special Issue on Artificial Imune Systems.
[42] K. Varadhan, The ns Manual (formerly ns Notes and Documentation), 2003.
[43] A. Varga, R. Hornig, An overview of the OMNeT++ simulation environment, in: Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), pp. 1–10.
[44] W. Wang, G. Zeng, L. Yuan, Ant-based reputation evidence distribution in P2P networks, in: Grid and Cooperative Computing, 2006. GCC 2006. Fifth International Conference, pp. 129–132.
[45] Y. Wang, V. Cahill, E. Gray, C. Harris, L. Liao, Bayesian network based trust management, in: [46], pp. 246–257.
[46] L.T. Yang, H. Jin, J. Ma, T. Ungerer (Eds.), Proceedings of the Third International conference on Autonomic and Trusted Computing (ATC 2006), Wuhan, China, September 3–6, 2006, Lecture Notes in Computer Science, vol. 4158, Springer, 2006.