Lightweight trusted routing for wireless sensor networks

Laurent Vercouter · Jean-Paul Jamont

Received: 25 July 2011 / Accepted: 7 February 2012 / Published online: 24 May 2012
© Springer-Verlag 2012

Abstract Communication in ad hoc network, such as wireless sensor networks (WSN), needs the use of decentralized routing algorithms requiring that several sensors behave in an expected way. This introduces vulnerabilities as the global issue of decentralized tasks depends on local behaviors and is compromised in case of failures or malicious intrusions. We propose here an adaptation of a routing protocol for WSN, the MWAC model, that introduces trust decisions to detect and avoid sensors that exhibit an incorrect behavior. The proposed trusted routing algorithms takes into account the low energy, communication and memory capacity of sensors to provide a realistic improvement of the routing robustness.

Keywords Trust management · Wireless sensor networks · Trusted routing · Self-organization

1 Introduction

Communication in wireless sensor networks (WSN) is usually supported by the creation of an ad hoc network connecting each sensor to the ones that are within its communication range. The decentralized nature of such networks implies the use of a multi-hop communication protocol in which several nodes are involved in the routing tasks. A drawback of relying on a collective activity for such a global task

is that it increases the system vulnerability faced to a failure or a malicious intrusion. If a sensor does not behave as expected, it will influence the issue of the global task. Moreover, another specificity of WSN is that the sensors have low energy, communication and memory capacities. Lightweight mechanisms are required and that represents an obstacle to the use of classical trust management techniques to protect the system against incorrect local behaviors.

We propose in this paper a lightweight trust model to improve the robustness of routing in WSN. The main originality of this trust model is that it is designed to work in a network where authentication cannot be ensured. As a node cannot authenticate its neighbors, our approach is based on trust estimation of the nodes' overall neighborhood. If it is distrusted, a node switches to a backup mode so that it contributes to move its neighborhood into quarantine. Quarantine is a way to disconnect a set of nodes of the network, that includes at least one sensor behaving wrongly. This disconnection ensures that the misbehaving sensor can not harm the overall functioning anymore. Our proposal has been integrated to the multi-wireless agent communication (MWAC) model [6] that allows a low cost routing in WSN.

The second section of the paper describes the MWAC model. Then, the trust model we propose is described as well as its integration in MWAC algorithms. Section 4 shows a practical implementation of our proposal in the MASH simulator and evaluates experimentally the benefits of using trust for routing.

2 The MWAC model

Multi-wireless agent communication model has been proposed in previous works [6] to handle communication in WSN. This model has been used in a real-world application to

L. Vercouter (✉)
INSA de Rouen, LITIS Laboratory, MIU group,
Avenue de l'Université, BP 8, 76801 Saint-Étienne
du Rouvray Cedex, France
e-mail: laurent.vercouter@insa-rouen.fr

J.-P. Jamont
LCIS Laboratory, University of Grenoble, 51 rue Barthélemyde
Laffemas, 26000 Valence, France
e-mail: jean-paul.jamont@iut-valence.fr

Fig. 1 MWAC organizational structure

monitor an underground river system in France (EnvSys system [6]) or in a flood decision support system in China [10].

The specificity of WSN is that each sensor has a limited communication range and low energy, memory and computing capacity. The MWAC model proposes a self-organization process to organize sensors in groups. This process is associated to a multi-hop routing mechanism that decreases the energy expense compared to flooding techniques.

The left part of Fig. 1 shows a network physical topology and the right one an organised view using MWAC model. It relies on an organizational structure based on agent roles and groups. Each group is composed by:

- one and only one group representative agent (r) managing the communication in its own group;
- some connection agents (c) belonging to more than one group and that are connected to the representative agent of each of their groups;
- some simple members (s) that do not have any routing task to ensure (unless they are the final sender or receiver of a message).

This section gives a general view of the routing and self-organizing mechanisms performed in MWAC. The detailed mechanisms are described in the MWAC reference paper [6]. The last subsection presents the problem of vulnerability against failures and malicious intrusions that appears in most of the decentralized routing mechanisms and that we tackle in the next section by proposing a robust variant of the MWAC model.

2.1 Message routing in MWAC

A message in MWAC follows a path between a source a and a receiver b corresponding to the definition of formula 1.

$$((a, r), * [(r, c), (c, r)], (r, b)) \quad (1)$$

Representative agents use local routing tables to choose some connection agents to which the message should be sent. Then, these connection agents propagate the message to the representatives of the groups they belong to. Each of these representative agents will continue this propagation with other connection agents unless the final receiver is in its own group. The local routing tables are updated by the way of eavesdropping. If a routing table does not allow a representative to build a message path, it uses a flooding protocol.

The energy saving comes thanks to the fact that the propagation is only directed to the representative agent of the groups and to some connection agents, instead of all the neighbors of a node as in flooding algorithms.

2.2 MWAC self-organization process

The efficiency of the routing protocols depends on the allocation of roles to agents and to the maintenance of a consistent organization. A self-organization process is used to allocate dynamically the agent roles and to build an efficient organization. The general idea of the algorithm is that an agent checks the role of its neighbors. If there is no representative in its neighborhood, it creates a group and adopts the representative role. If there is more than one representative in its neighborhood, it becomes a connection agent. In the other cases, it is a simple member. In order to communicate with their neighbors, agents follow an introduction protocol in which they send periodically a message to describe themselves. This message contains the id, the role and the groups of its sender.

Conflicts occur when two or more representative agents communicate. In this case, the representatives exchange another message containing a score calculated from the sender's amount of energy and its number of neighbors. The representative with the highest score keeps this role while

those with lower scores drop it to become simple members or connection agents.

2.3 Vulnerability against intrusions and failures

MWAC is especially suited to deal with high-scale WSN composed of nodes having very limited communication and computing capacities. However, as it is often the case when one considers decentralized algorithms, it is assumed that the working network nodes interact as specified in the local algorithms. If a sensor does not behave as expected (because of a failure or a malicious intrusion) when interacting with other sensors, it will impact the decisions taken by other nodes and it may corrupt the overall functioning of the system. In this paper, we will use the term “lie” to refer to incorrect interactions even if they are not intentional and due to failures.

Lies can disturb both the routing and the self-organising processes. The work described in this paper is only focused on the impact of lies on the issue of self-organization. Increasing the robustness of MWAC against lies during the routing process will be considered in our future works. Here, we consider essentially lies that occur in the introduction message which allows a sensor agent to present itself to its neighbors. In this message, the agent declares its id, its role and its groups.

There are various consequences of a lie in an introduction message. If it is a lie on the agent’s id, it may modify the real route followed by messages and receive those addressed to another node (either as a final destination or as an intermediary step in a route). A lie on a group or on a role, pretending that a node is a representative or a connection agent, would attract messages addressed to a given group.

We propose in this paper a variant of the MWAC protocol that includes a trust model in the decision process. Trust is calculated from the detection of the occurrence of lies and is used to adapt the role allocation process.

3 A robust variant of the MWAC model

Weaknesses against failures and malicious intrusions are commonly encountered when dealing with networks that rely on decentralized algorithms. As these algorithms are executed by several agents, if one of them does not execute correctly its portion of code, it can make the global algorithm fail. Trust has been proposed in existing works to tackle this problem. However, some specificities of WSN, especially the limitation of energy and communication costs makes it difficult to apply existing trust models.

The first subsection shows why existing approaches of decentralized trust management cannot be used for WSN. The second subsection gives an overview of the agent communication architecture using the trusted version of MWAC.

The third subsection presents the local trust assessment processes that we propose for MWAC. An adaptation of the role allocation algorithm using trust is then described in Sect. 3.4 and a variant using propagation in Sect. 3.5.

3.1 Decentralized trust management for wireless sensor networks

Trust management aims at protecting a system against bad behaviors of some of its entities. In the context of WSN, the idea is to observe other sensors’ behaviors, to compute and assign them trust values and to avoid sensors that are not trustworthy.

3.1.1 WSN constraints on trust management

Decentralization Trust management systems have been proposed using different approaches. A clear distinction of approaches concern the architecture that may be centralized or decentralized. Recent works in multi-agent systems have emphasized the advantages of decentralized mechanisms. The main reasons are that it facilitates the scalability of the system and that it prevents relying on a critical central bottleneck. A large number of decentralized trust models have then been proposed for multi-agent systems (see [14] for a global survey). Trust is assessed locally by agents based on their local perceptions and experiences with others. Communicated experiences may also take place in order to allow agents to help others in their trust calculation. In that case, specific mechanisms should be deployed to estimate the trustworthiness of a gossip and to avoid false experience reports.

In the context of WSN, the need of decentralization is enhanced by their physical nature (multi-hop communication under energy constraints and low memory resources). Recently, some decentralized models have been proposed (like [1–4, 12, 13]).

No authentication Most decentralized trust management works share the mutual assumption that there exists an authentication system. It is indeed essential that an id is assigned to each agent without any doubt, so that agents can attach trust estimation to identities. Service infrastructures, such as Public Key Infrastructure, are frequently used to provide authentication. Yet, in WSN sensors have very low capacities for communication or data storage, especially in large scale networks where the cost of each sensor should be as low as possible. It is therefore not realistic to consider that each sensor stores a public key for every other sensor, nor that each one can communicate with a central repository storing all these keys. Some works [8, 9, 11] have thus been interesting in making the authentication mechanisms lighter.

The lack of authentication makes it impossible to use classical decentralized trust models. As a matter of fact, when an agent receives a message, it may have been sent by any

other agent inside its communication range. Believing the id claimed by the sender is not a solution as a malicious one can claim any id.

Resource limitations Trust algorithms often use a history of observations as well as communications to share observations between agents. This implies some memory and communication costs that should be considered when they are deployed in a system with high energy and hardware constraints. It is the case of the kind of WSN addressed in this paper. The sensors used here have microcontroller clock often less than 1 MHz, 32 kb of memories (for code and data) and reduced periods of time during which the sending of messages is allowed in order to save energy.

Because of these important constraints brought by the context of WSN, decentralized trust management raises here an original problem that requires to adopt a new approach.

3.1.2 Our approach

The work proposed in this paper follows a new approach to decentralized trust management that can be used when authentication is not possible. We suggest to use trust to assess the reliability of the neighborhood as a whole rather than a separate assessment for each neighbor. An untrustworthy neighborhood would then mean that there should be at least one malicious or defective agent in it.

When a node believes that its neighborhood is not trustworthy, it changes its behavior to work in a backup mode, performing only the minimal required tasks. This backup mode should involve as less as possible the node's neighbors. The global aim of this approach is that all the neighbors of a malicious or failing sensor will progressively detect a wrong behavior in their neighborhood and switch to the backup mode. The part of the network composed of the malicious agent and its neighbors will then be in quarantine and will no longer have the possibility to influence self-organization with lies. The messages used to assess trust in the neighborhood can be all the messages exchanged in this neighborhood if eavesdropping can be used.

The contribution described in this paper is an extension of the MWAC model to improve the robustness of its self-organization process. Protecting the construction of clusters is essential as they determine the network logical topology on which relies the routing protocols. There exists in the literature several works based on other cluster-based routing protocols that have emphasized the need of trust management for cluster formation. In [7], authors propose a group leader election procedure which takes into account several parameters including a trust value. Another proposal is the one of [15] that consists in determining trust about a cluster head by monitoring nodes. Besides the use of a routing protocol different from these existing works, our approach,

based on trust in the neighborhood and backup modes, can deal with the lack of authentication.

3.2 Communication architecture

Traditionally, communications in embedded systems are decomposed into a classical¹ three-layer architecture because of frame length, code/memory sizes and time constraints. These three layers are:

1. the *physical layer* which defines electrical and physical specifications for communication devices,
2. the *data link layer* which focuses on the data transfer between directly connected communication systems,
3. the *applicative layer* which includes traditional functionalities of the five data link upon OSI layers and not only networked services as in OSI model.

Figure 2 gives an insight of agent communication architecture using the trusted MWAC model. The self-organization process and the routing process are functionalities offered by the applicative layer. Applicative code of agents using the trusted version of MWAC cannot address directly the MWAC communication primitives. These primitives are only accessible through the MWAC trust management components.

This paper focuses on the trust management of the MWAC self-organization process.

3.3 Trust assessment

Trust is estimated locally by each agent by supervising the messages sent in its neighborhood. Even if authentication is not possible, nodes have to use an id (real or fake) when sending a message. We propose to use trust in an id (rather than trust in an agent authenticated with an id) in order to represent the way an id has been used in the past.

3.3.1 Trust initialisation and updates

A new trust value is created each time a new id value is used in the sender's neighborhood. A trust value takes a value in the range [0; 1]. The initial value of an id *id* is the highest ($Trust(id) = 1$).

In the proposed trust model, trust values are decreased when a lie is detected or suspected but it is not increased when correct messages are perceived. As an id is first fully trusted, there is no need to reward good behaviors in using this id, whereas a lie should imply a drastic decrease.

¹ Wireless sensor networks implements often these three minimal layers but the number of layers can be higher like in the IPv6 sensors world.

Fig. 2 Architecture of the trusted version of MWAC

Table 1 Lie in the introduction message

Lie on	Detection by node n	Trust decrease
id	if $usedid = id$ of n if $usedid = id$ of a new workstation	$Trust(id) = 0$ $Trust(id) = Trust(id) - \eta$ and call <i>check_old_route_procedure</i>
$group$	if n is a representative and his group is not included in the set of groups of the message if a new group is presented and if it is the only agent that connects it <i>see below the new group checking process</i>	$Trust(id) = 0$ call <i>new_group_checking</i>
$role$	if the node claims he has a connection role, it is the same situation than the presentation of a new group (see previous case)	call <i>new_group_checking</i>

Depending on its nature, a lie can be either detected without any doubt or only suspected. In the first case, trust in the corresponding id is decreased at the lowest value (0) as it is sure that a liar is in the neighborhood and has used this id. In the second case, the occurrence of a lie is only suspected as an unusual event occurred but it may not be caused by a lie. For instance, a node may send a false information if it has temporarily a false belief (e.g. a node informs that it is in a given group while it has just left the communication range of the group representative without having detected it yet). If a lie is suspected, trust should be decreased with an importance proportional to the likelihood that it is a lie.

Table 1 summarizes the lies that can occur during the introduction protocol.

The first and the third cases refer respectively to the case where a node perceives a message using its own id and where a representative see that one of its neighbors hide the group it represents. Trust is here set to the minimal value.

The second case corresponds to a new neighbor which declares to be a workstation. There is usually a single workstation in WSN, used to be the final recipient of messages. The workstation is generally not mobile. It is therefore unlikely that a new one appears in a neighborhood. If this

happens the sensor should already have some routes to reach the workstation as it is a final recipient of messages. These old routes are used to check that it is really the claimed workstation.

The fourth and fifth cases correspond to a node claiming to be a connection node creating a link with a new group. The appearance of a new group is rare and it is unlikely that a node is the only connection to another group. However, this may happen and a lie can only be suspected. The node suspecting such a lie should then follow a *new group checking* process consisting trying to reach the representative of the new group with a query asking the ids of its group members. Due to a lack of space, the detailed mechanism to send the query is not detailed here. An important characteristic is that this specific query should be sent by flooding in paths trying to avoid the suspected id. Several replies may then be received. Depending on these replies, trust in the suspected id is updated as shown in Table 2.

If no reply is received or if all the received replies state that the node is inside the representative's group, there is probably no lie and trust should not be decreased. If all the replies state that the suspected node is not in the representative's group, it may be a lie or a false belief of the suspected

Table 2 Result of the new group checking process

Replies received	Trust decrease
No reply	No decrease
All replies stating that the node is in the group	No decrease
All replies stating that the node is not in the group	$Trust(id) = Trust(id) - \alpha$
Inconsistent replies received	$Trust(id) = Trust(id) - \beta$

node. Trust should be decreased by a value α but not at the minimum value as there are still some doubts.

The last case occurs when some different replies from the representative arrive, some stating that the node is in the group and some stating that it is not. The most likely here is that fake replies are sent by a liar. But it can also happen without any wrong behavior, for instance if the neighborhood changes between the sending of two replies. As these situations will be quite rare, the sanction β on trust should be stronger. We propose then to set these sanctions in the interval $0 < \beta < \alpha < 1$. Trust values are set to 0 if they are decreased to negative values.

3.3.2 Trust recovery

It is important that trust in the neighborhood can be recovered, especially if the neighborhood of an agent may change (and maybe the malicious node that caused trust decreases has left). Trust recovery is done by forgiveness with a slow trust increase as time goes by. Algorithm 1 shows this recovery.

Algorithm 1 Trust recovery algorithm

```

for all  $id$  in  $neighborhood.getUsedIds()$  do
   $Trust(id) = (1 - \lambda) * Trust(id) + \lambda$ 
end for

```

Two parameters are used to configure the speed of trust recovery: (i) the frequency ν of invocation of the trust recovery function; (ii) the evaporation rate λ , with $0 \leq \lambda < 1$ setting the amount of trust recovered. The value of these parameters should be set according to the expected mobility of nodes. If the nodes are very mobile, the neighborhoods will frequently change and it may be interesting to have high evaporation and frequency. Otherwise, if the network is very static, these values should be quite low.

3.3.3 Trust decision algorithm

Trust in the ids is used to estimate the trustworthiness of the whole neighborhood of a node. The neighborhood of a node

is considered untrustworthy if there is at least one id that is distrusted (Algorithm 2).

Algorithm 2 Neighborhood trust decision algorithm

```

for all  $id$  in  $neighborhood.getUsedIds()$  do
  if  $Trust(id) < \theta$  then
    return distrusted
  end if
end for
return trusted

```

The threshold θ_1 represents the limit of the trust value under which an id is distrusted. It takes a value in the range: $0 < \theta < 1$.

3.4 Adaptation of the MWAC model

The original MWAC model has been modified to integrate trust in a node's neighborhood. If a node distrusts its neighborhood, it should place itself in quarantine and run in a backup mode in which he does not participate anymore to routing. The backup mode consists in a new role *BACKUP*.

The proposed adaptation of the MWAC role attribution process is shown in Algorithm 3.

Algorithm 3 Adaptation of the role attribution algorithm

```

if  $neighborhood.isEmpty()$  then
  // No possible organizational structure
else if  $neighborhood.trust() = \text{distrusted}$  then
   $myRole \leftarrow \text{BACKUP}$ 
else if  $myRole = \text{REPRESENTATIVE}$  and  $neighborhood.nbrOfRepresentative() > 0$  then
   $conflictResolutionProcedure()$ 
else if  $neighborhood.nbrOfRepresentative() = 0$  then
   $myRole \leftarrow \text{REPRESENTATIVE}$ 
else if  $neighborhood.nbrOfRepresentative() = 1$  then
   $myRole \leftarrow \text{SIMPLEMEMBER}$ 
else  $\{neighborhood.nbrOfRepresentative() > 1\}$ 
   $myRole \leftarrow \text{CONNECTION}$ 
end if

```

The new role, *BACKUP*, that we introduce here has a similar behavior than the *SIMPLEMEMBER* role. The only difference is that an agent adopting a *BACKUP* role has to inform its neighbors of this change to trigger a local re-organization. During this re-organization, agents having a *BACKUP* role are not able to move to *CONNECTION* or *REPRESENTATIVE* roles. Thus, preventing an agent having a distrusted neighborhood to participate to routing tasks contributes to the social exclusion of the malicious nodes. If all the neighborhoods in which the malicious node is included are distrusted, all its neighbors will be in *BACKUP* roles and it will not be able to impact on routing anymore.

3.5 Propagating backup modes

A classical approach to share trust information is to build reputations or to communicate its own social evaluations of others. However, the lack of authentication in our case prevents using these traditional approaches as it is also essential to identify the target of a reputation or a trust value.

In our case, an interesting information for a node is to know if some of its neighbors have a *BACKUP* role and for which reason. This means that a malicious node may be close and that it may be in the neighborhood of the node receiving such an information.

We propose here a second variant of MWAC in which agents take into account the *BACKUP* role of their neighbors. As defined in the previous section, a node adopting a *BACKUP* role notifies it to its neighbors to trigger a re-organization process. In this variant, the message should also contain the distrusted ids. The motivation for sharing this information is that the malicious node may be in the neighborhood of other nodes that have not detected its lies.

In order to take into account the role of the neighbors, we propose in the Algorithm 4 an adaptation of Algorithm 2 for trust decisions.

Using the distrusted ids is essential to avoid propagating the backup role to the whole network. This propagation will thus stop when it reaches nodes that do not have the distrusted ids in their neighborhood.

This variant introduces a new possibility of attack for malicious nodes if they adopt the *BACKUP* and pretend that some ids are distrusted. To counter this attack, a node that hears that its own id is distrusted while it believes that it is not true, should distrust the sender of this message (by setting the trust value of its id to 0). This protection will not prevent the correct node that is attacked to be isolated (but with trust recovery, it may be recovered in the future), but it will also isolate the malicious node.

Algorithm 4 Trust decision using neighbors' role

```

for all id in neighborhood.getUsedIds() do
  if Trust(id) <  $\theta$  then
    return distrusted
  end if
  if (role(id) = BACKUP(IDS)) and (IDS  $\cap$ 
    neighborhood.getUsedIds()  $\neq \emptyset$ ) then
    return distrusted
  end if
end for
return trusted

```

4 Experimental evaluation

The robustness improvement provided by trust integration in MWAC has been experimentally evaluated in the MASH

simulator [5]. In this section, we describe this experimental protocol and then give an insight to the results of this evaluation.

4.1 Experimental protocol

Instrumentation applications of WSN involve two types of entities: sensors and a collect workstation. Sensors are spatially distributed measurement nodes which monitor environments. The collect workstation is the final destination node that should acquire the data.

An attack against the network functioning consists in trying to prevent data to reach the workstation. A malicious node may then adopt different strategies to do that. The most efficient and direct one is to usurp the workstation id.

The experiments shown in this section have been made in a simulation with 100 sensors, during 100 time steps. At each time step, each sensor takes a measurement and sends it in a message for the workstation. During a first initialisation period, all the nodes behave as expected. This corresponds to the initial deployment of the sensor network during which we can reasonably imagine that there has not been yet any malicious intrusion. At the time step 20, we change the behavior of a few nodes to simulate an attack.

The simulations have been done with an adapted version of the MWAC protocol in the MASH simulator [5]. Figure 3 is a screen copy of the simulator showing the network topology on the back and a dialog box with a node's details in the front. An example of local trust assessments is here displayed about the ids 48 and 50. The low trust in id 48 implies distrust in the neighborhood.

For the experiments, the model parameters have been set to : $\theta = 0.5$; $\alpha = 0.4$; $\beta = 0.8$ et $\lambda = 0$.

4.2 Scenario 1: Usurping the workstation id

A first experimental scenario simulates the introduction of sensors usurping the workstation id. The malicious nodes declare that their id is the one of the workstation in order to attract measurements. Figure 4 presents the amount of lost messages caused by this usurpation.

A first test has been done without usurpation to check that 100 % of the messages are well received. Then, with the original version of MWAC, the experiments show an important impact of usurping the workstation id. When one malicious node is introduced, around 30 % of the measurements are lost after 100 time steps. Another simulation performed with two usurpers show that the loss is around 42 %. Indeed, representative nodes update their local routing tables if they find shorter path to the workstation and may then route the measurements to malicious nodes instead of the real workstation. The percentage of lost messages should only been interpreted as a general indication as the position of the

Fig. 3 The MASH simulator (simulated WSN in background, internal state of an agent in foreground)

Fig. 4 Ratio of lost messages with nodes usurping the workstation id

malicious node in the network has a great influence on the exact impact on the attack.

The two charts entitled TrustedMWAC show the contribution of using the trust algorithms presented in this paper. The simulations have been performed with the same settings, topology and attacks than in the previous case. The amount of lost measurements is much lower, around 3 and 6 % for respectively one and two usurpers. The neighbors of an usurper will check that the node claiming to be the workstation is really the workstation. As they will receive contradictory answer, they will move to a *BACKUP* role and isolate the usurper.

The measurements that are still lost come from the malicious sensors and from a few nodes that may be disconnected by the quarantine zone.

4.3 Scenario 2: Usurping a sensor id

In the second simulated scenario, the attack consists in usurping the id of a another sensor. The malicious node tries to usurp the id of a representative in its neighborhood. For the sensors that are both in the neighborhood of the real representative and of the usurper, there is no change as they do not have the possibility to distinguish the real sender of the

Fig. 5 Ratio of lost messages with nodes usurping other nodes id

message. For other nodes, the appearance of a new neighbor may be interpreted as a consequence of the sensors mobility. The only agent that is able to detect the attack is the representative agent which id has been usurped. Figure 5 gives some experimental results on this scenario.

The impact of this attack is lower than in the previous scenario. Between 6 and 7 % of the measurements are lost. This loss consists in measurements that have not been received by the real representative (otherwise it would have propagate them by a valid route) but by the fake one. Using the trusted extension of MWAC allows the representative to detect that its id has been usurped and to move in *BACKUP* mode. But if we use only this reaction, it increases the damages of the attack with around 11% of the measurements lost. Indeed, the real representative does not fulfill his role anymore and leaves all the measurements that it should receive to the malicious node.

Using the propagation of backups as proposed in Sect. 3.5 gives better results. Around 2 % of the measurements are lost. These measurements only concern the sensors isolated by the quarantine. Two sensors are here isolated: the malicious node and the real representative. As there is no possibility for nodes to distinguish between the two, this approach protects the network by isolating both.

5 Conclusion

We describe here an extension of the MWAC model to increase its robustness against failures and malicious intrusion. MWAC is used to enable message routing in wireless sensor networks and this kind of application raises an original problem for trust modelling which is to take into account the absence of authentication. We propose here to use a trust model in an agent's neighborhood, rather than in its neighbors, to move into quarantine distrusted parts of the networks.

This is an ongoing work that has for the moment be applied to the self-organization in MWAC. Our proposal has been

experimentally evaluated against attacks consisting in stealing the identity of a workstation or of another sensor. Experiments show an important decrease of the messages lost when trust is used. Our current work is to extend the usage of trust to the routing protocol used on top of a secured organization.

References

1. Boukerche, A., Xu, L., El-Khatib, K.: Trust-based security for wireless ad hoc and sensor networks. *Comput. Commun.* **30**(11–12), 2413–2427 (2007)
2. Chen, H., Wu, H., Hu, J., Gao, C.: Agent-based trust management model for wireless sensor networks. In: *International Conference on Multimedia and Ubiquitous Engineering*, pp. 150–154. IEEE Computer Society (2008)
3. Dai, H., Jia, Z., Qin, Z.: Trust evaluation and dynamic routing decision based on fuzzy theory for manets. *J. Sofw.* **4**(10), 1091–1101 (2009)
4. Han, G., Choi, D., Lim, W.: A novel sensor node selection method based on trust for wireless sensor networks. In: *International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 2397–2400. IEEE Computer Society (2007)
5. Jamont, J.P., Occello, M.: A multiagent tool to simulate hybrid real/virtual embedded agent societies. In: *Proceedings of the IEEE/WIC/ACM Conference on Intelligent Agent Technology*, pp. 501–504. IEEE computer society (2009)
6. Jamont, J.P., Occello, M., Lagrèze, A.: A multiagent approach to manage communication in wireless instrumentation systems. *Measurement* **43**(4), 489–503 (2010)
7. Kifayat, K., Merabti, M., Shi, Q., Llewellyn-Jones, D.: An efficient algorithm to detect faulty reading in wireless sensor network using the concept of reputation. In: *International Conference on Network and Service Security*, pp. 1–5. IEEE Computer Society (2009)
8. Lewis, N., Foukia, N., Govan, D.G.: Using trust for key distribution and route selection in wireless sensor networks. In: *IEEE/IFIP Network Operations and Management Symposium: Pervasive Management for Ubiquitous Networks and Services*, pp. 787–790. IEEE (2008)
9. Liu, T., Zhou, M.Z.: Implementing a trust-aware routing protocol in wireless sensor nodes. In: *IEEE International Conference on Progress in Informatics and Computing (PIC)*, pp. 556–559. IEEE Computer Society (2010)

10. Luo, J., Xu, L., Jamont, J.P., L., Z., Shi, Z.: Flood decision support system on agent grid: method and implementation. *Enterp. Inf. Syst.* **1**(1), 49–68 (2007)
11. Pei, Q., Wang, L., Yin, H., Pang, L., Tang, H.: Layer key management scheme on wireless sensor networks. In: *Proceedings of the Fifth International Conference on Information Assurance and Security*, pp. 427–431. IEEE Computer Society (2009)
12. Peng, X., Wei, Z.: The trustworthiness based on hash chain in wireless sensor network. In: *IEEE/IPIP International Conference on Embedded and Ubiquitous Computing*, pp. 101–106. IEEE Computer Society (2008)
13. Pietro, R.D., Michiardi, P., Molva, R.: Confidentiality and integrity for data aggregation in wsn using peer monitoring. *Secur. Commun. Netw.* **2**(2), 181–194 (2009)
14. Sabater-Mir, J., Vercouter, L.: *Multi-Agent Systems*, chap. In: *Trust and Reputation in Multiagent Systems*, vol. 9, 2nd edn. 9. MIT Press, Cambridge (2012)
15. Zhou, Y., Huang, T., Wang, W.: A trust establishment scheme for cluster-based sensor networks. In: *5th International Conference Wireless Communications, Networking and Mobile Computing*, pp. 1–4. IEEE (2009)