Lightweight trusted routing for Wireless Sensor Networks

Laurent Vercouter and Jean-Paul Jamont

Abstract Communication in *ad hoc* network, such as Wireless Sensor Networks (WSN), needs the use of decentralised routing algorithms requiring that several sensors behave in an expected way. This introduces a vulnerability 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.

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 doesn't 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.

Laurent Vercouter
École Nationale Supérieure des Mines de Saint-Étienne, ISCOD/LSTI group, 158 cours Fauriel, 42023 Saint-Étienne cedex 02, France e-mail: laurent.vercouter@emse.fr

Jean-Paul Jamont
University of Grenoble, LCIS Labs, 51 rue Barthlmy de Laffemas, 26000 Valence, France e-mail: jean-paul.jamont@iut-valence.fr

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. Our proposal has been integrated to the MWAC model [3] 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 MWAC simulator and evaluate experimentally the benefits of using trust for routing.

2 The MWAC model

MWAC[1] has been proposed in previous works [3] to handle communication in WSN. 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 multi-hop routing mechanism that decreases the energy expense compared to flooding techniques.

Figure 1 shows an example of the use of MWAC.

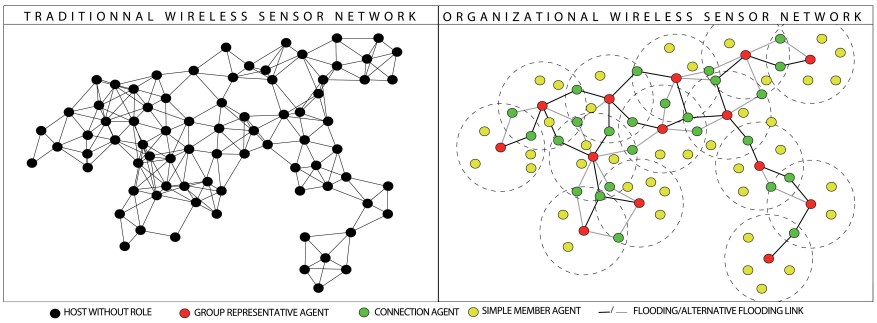

Fig. 1 MWAC organizational structure

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

[1] *Multi Wireless Agent Communication model*

- 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-organisazing mechanisms performed in MWAC. The detailled mechanisms are described in the MWAC reference paper [3]. 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 equation 1.

$$((a, r),^* [(r, c), (c, r)], (r, b)) \tag{1}$$

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 doesn't 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-organisation process

The efficiency of the routing protocols depends on the allocation of roles to agents and to the maintenance of a consistent organisation. A self-organisation process is used to allocate dynamically the agent roles and to build an efficient organisation. 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, 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 wireless sensor networks 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 functionning 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-organisation. 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 in which an 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 adressed 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 adressed 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 occurence 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 wireless sensor networks, 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 wireless sensor networks. The second subsection presents the local trust assessment processes that we propose for MWAC. An adaptation of the role allocation algorithm using trust is then described.

3.1 Decentralized trust management for wireless sensor networks

Trust management aims at protecting a system against bad behaviors of some of its entities. The idea is to observe other agents' behaviors, to compute and assign them trust values and to avoid agents that are not trustworthy. Decentralized trust management mechanisms have also been proposed to increase the reliability of routing in large scale networks such as *ad hoc* [1] or peer-to-peer [5] networks. In a more general perspective, a global overview of existing trust systems for multi-agent systems can be found in this survey [4].

However, all these existing systems 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 wireless sensor networks, 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.

The lack of authentication makes it impossible to use classical 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. Decentralized trust management in WSN raises then an original problem that requires to adopt a new 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 progressivly 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-organisation 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.

3.2 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 authentified with an id) in order to represent the way an id has been used in the past.

3.2.1 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.

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 sur 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 occured 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

Lie on	**Detection by node n**	**Trust decrease**
id	if $usedid = id$ of n	$Trust(id) = 0$
	if $usedid = id$ of a new worsktation	$Trust(id) = Trust(id) - \eta$ and call *check_old_route_procedure*
group	if n is a representative and his group is not included in the set of groups of the message	$Trust(id) = 0$
	if a new group is presented and if it is the only agent that connects it *see below the* new group checking *process*	*see below*
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)	*see below*

Table 1 Lie in the introduction message

of its neighbors hide the group it represents. Trust is here set to the minimal value.

The second case corresponds to a new neighboor which declares to be a workstation. Workstations are used in WSN to be the final recipient of messages and are 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 detailled 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.

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	$Trust(id) = Trust(id) - \beta$

Table 2 Result of the new group checking process

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

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.

3.2.2 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.

3.2.3 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).

The threshold θ_1 represents the limit of the trust value under which an id is distruted. It takes a value in the range: $0 < \theta_1 < 1$.

Algorithm 2 Neighborhood trust decision algorithm

```
for all id in neighborhood.getUsedIds() do
  if Trust(id) < θ1 then
    return distrusted
  end if
end for
return trusted
```

3.3 Adaptation of the MWAC model

The original MWAC model has been modified to integrate trust in a node's neighborhood. If a node distruts its neighborhood, it should place itself in quarantine and run in a backup mode in which he does not participate anymore to routing. In MWAC, the backup mode consists in adopting systematically a role of *Simple Member*.

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() = distrusted then
  myRole ← SIMPLEMEMBER
else if myRole = REPRESENTATIVE and neighborhood.nbOfRepresentative() >
0 then
  conflictResolutionProcedure()
else if neighborhood.nbOfRepresentative() = 0 then
  myRole ← REPRESENTATIVE
else if neighborhood.nbOfRepresentative() = 1 then
  myRole ← SIMPLEMEMBER
else {neighborhood.nbOfRepresentative() > 1}
  myRole ← CONNECTION
end if
```

4 Experimental evaluation

The robustness improvement provided by trust integration in MWAC has been experimentally evaluated in the MWAC simulator [2]. 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 wireless sensor network involve two type of entities are evolved: 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.

A popular attack of this type of WSN consists in usurping the collect workstation id. This is a way to spy the data collected or to prevent the real collect workstation to obtain them. We introduced in the MWAC simulator some sensors with this malicious behavior.

A scenario in which sensors send a measure every 5 seconds has been implemented. The network contains 600 sensor agents and 1 to 10 malicious agents. The network topology is partially visible on the background of the figure 2. Malicious agents are uniformly distributed on this topology.

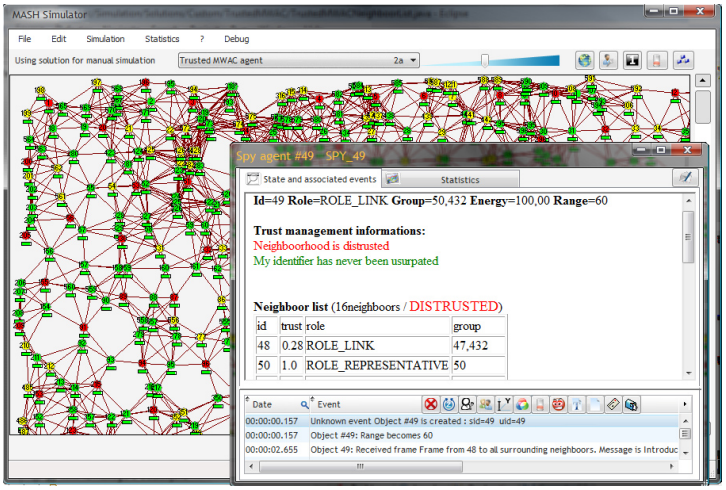

Fig. 2 The MASH simulator (simulated WSN in background, internal state of an agent in foreground)

4.2 Results

Figure 3 gives an insight of the performances of the trusted version of MWAC in comparison its original version. The charts represent the number of missing measure, *i.e.* sensor data sent that are not received by the workstation. $m1$, $m5$ and $m10$ are the results obtained with the original version of MWAC deploying respectively 1, 5 and 10 malicious agents. $t1$, $t2$ and $t10$ are the

results obtained with the trusted version deploying respectively 1, 5 and 10 malicious agents.

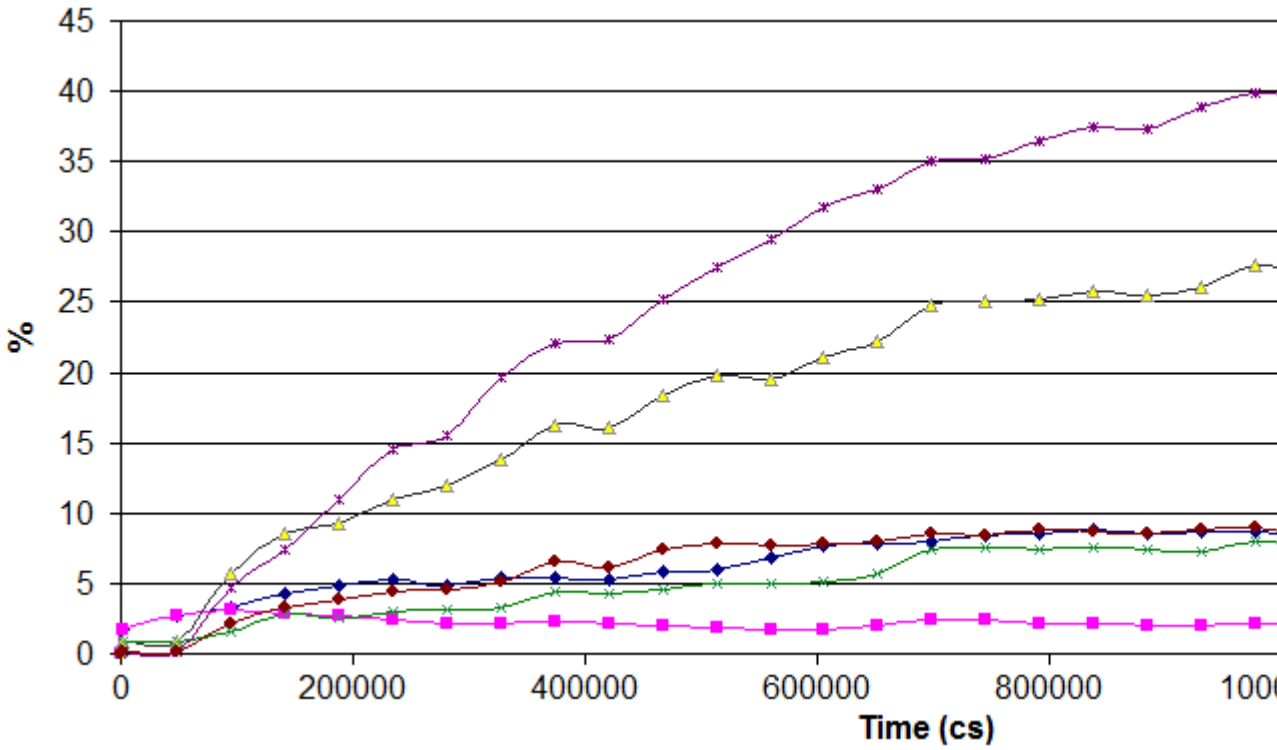

Fig. 3 Percentage of missing measures

Before $t = 0cs$ the WSN has stabilized its self-organizion and sensor agents have exchanged measures. Some representative agents have learn some path to route messages to the workstation.

At $t = 0cs$, some sensors agent are replaced by malicious agents. The experiments show the high impact that a few malicious behaviors have in the original version of MWAC. The percentage of missing measures is close to respectively 45% and 30% with only 10 and 5 malicious agents.

With the trusted version of MWAC, the percentage of missing measures is significantly lower. It reaches 9%, 7% and 2% with respectively 10, 5 and 1 malicious agents. It may be surprising that there are still missing measures whereas all the malicious agents are in quarantine. In fact, the undelivered messages are not intercepted by malicious nodes but corresponds to the measures directly emitted from quarantine zones.

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 in quarantine untrusted parts of the networks.

This is an ongoing work that has for the moment be applied to the self-organisation in MWAC. Our proposal has been experimentally evaluated against attacks consisting in stealing the identity of a workstation. 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 other processes used in self-organisation and to message routing.

References

1. Griffiths, N., Jhumka, A., Dawson, A., Myers, R.: A simple trust model for on-demand routing in mobile ad-hoc networks. In: C. Badica, G. Mangioni, V. Carchiolo, D.D. Burdescu (eds.) Proceedings of the 2nd International Symposium on Intelligent Distributed Computing - IDC 2008, *Studies in Computational Intelligence*, vol. 162, pp. 105–114. Springer, Catania, Italy (2008)
2. Jamont, J.P., Occello, M.: A multiagent tool to simulate hybrid real/virtual embedded agent societies. In: IAT, pp. 501–504 (2009)
3. Jamont, J.P., Occello, M., Lagrèze, A.: A multiagent approach to manage communication in wireless instrumentation systems. Measurement **43**(4), 489 – 503 (2010)
4. Sabater, J., Sierra, C.: Review on computational trust and reputation models. Artif. Intell. Rev. **24**(1), 33–60 (2005)
5. Vercouter, L., Muller, G.: L.I.A.R.: achieving social control in open and decentralized multiagent systems. Appl. Artif. Intell. **24**, 723–768 (2010)