A Trusted Lightweight Synchronisation Protocol for Wireless Ad-hoc Networks

Adrien van den Bossche, Nicolas Fourty, Jean-Paul Jamont

A Trusted Lightweight Synchronisation Protocol for Wireless Ad-hoc Networks

Adrien Van den Bossche[1], Nicolas Fourty[2], Jean-Paul Jamont[2]

[1] Institut de Recherche en Informatique de Toulouse, Université Fédérale de Toulouse, CNRS, INPT, UPS, UT1, UT2J
[2] LCIS, Univ. Grenoble Alpes, Valence, France
vandenbo@univ-tlse2.fr, {nicolas.fourty, jean-paul.jamont}@lcis.grenoble-inp.fr

***Abstract*—This paper describes a trusted lightweight synchronisation protocol for wireless ad-hoc networks. The wireless ad-hoc network and wireless sensor networks is a large field of research. Their dynamic topologies make them difficult to synchronise (power consumption, slow convergence, accuracy). Moreover, these open networks are vulnerable to faulty nodes and attacks from malicious nodes. In this paper we propose to use the Simple Synchronisation Protocol (SiSP) enhanced with trust aspects making it more robust to attacks. We describe how to observe deviant behaviour and assign a confidence level to each node. Countermeasures are then adapted to the neighbourhood confidence level. The last part present simulation and results on the MASH simulator.**

Keywords—Synchronisation; Trusted protocols; Medium Access Control; Wireless Ad Hoc Networks; Wireless sensor networks.

I. INTRODUCTION

Wireless Ad Hoc Networks promise a simple and decentralised way of connecting nodes, without need of coordination or hierarchy. Because the wireless nodes are generally mobile, they usually embed a limited quantity of energy and are considered as energy constrained. Thus, Medium Access Control (MAC) protocols have to minimize activities on medium, i.e. limit not only transmission but also reception activities, because of the complexity of the receiver: on these transceivers, the energy consumption is generally more important in the receive state than in transmit state. Nodes must enter sleep mode regularly, but also being available during common periods negotiated with their 1-hop neighbours. The nodes use the principle of *rendez-vous* to get this common period.

To achieve such a negotiation, the synchronisation is an important task in the whole MAC process. Many synchronisation protocols and algorithms have been proposed [1], in order to share a common view of the time organization. To rely on the non-hierarchical characteristic of the Wireless Ad Hoc Networks, a decentralized synchronisation process is targeted, i.e. obtained a global synchronisation in a distributed fashion. Because of the collaborative nature of these protocols, their trust aspects must be studied, since they may be sensitive to attacks and malice operations.

This paper presents SiSP, a simple and collaborative-oriented synchronisation protocol and studies the trusted aspects of the protocol. A trusted version of SiSP is proposed and studied by the use of the MASH simulator.

The reminder of this paper is following: in the next section, we remind the main challenges on synchronisation and trust aspects. Then, the SiSP protocol and its weaknesses and deviant behaviour are presented. A trusted version of SiSP is then being proposed, with some results obtained with this reinforced version of the protocol.

II. MAIN CHALLENGES

Wireless Ad-hoc networks refer to a lot of specific areas of wireless networks. Since there isn't any infrastructure, all nodes must sense the environment to construct their own neighbourhood image. This characteristic has a significant impact on node lifetime, reliability or even security. In order to save energy, Wireless Ad Hoc Networks often use *rendez-vous*-based protocols or synchronisation protocols at MAC-level.

A. Synchronisation protocols

Time synchronisation over Wireless Networks is a wide research subject for a very long time [1], and is still an active field of research [2], since it impacts on several aspects such as MAC, Quality of Service, energy saving, data timestamping, etc. In the literature, the synchronisation protocols are evaluated by several metrics: synchronisation accuracy, energy consumption, overhead, convergence speed. They can use various approaches which can be classified in two categories.

A first category of synchronisation protocols is built on a hierarchical architecture, based on a hierarchical topology such as a star or a tree topology. *Timing-sync Protocol for Sensor Networks* (TPSN) [3] is an example of synchronisation protocol based on a hierarchy, where each node is assigned a hierarchical level. The root node, on zero level, is unique, and must be re-elected each time the topology changes. A time synchronisation accuracy analysis, based on a real implementation with MICA motes with TinyOS has been conducted in [3]; show an average accuracy of less than 20µs. *Flooding Time Synchronisation Protocol* (FTSP) [4] is an improvement of TPSN. With TPSN, topology changes due to node failure or link instability are heavy to report. The flooding nature of FTSP reinforces the quality of the synchronisation when the topology changes. The authors of [4] have studied the performances of FTSP and report an average synchronisation error of 10µs. *Time Synchronised Mesh Protocol* (TSMP) [5] is

the synchronisation protocol used on WirelessHART. It proposes a synchronisation service used for scheduling the medium accesses, allowing collision free schemes on a multichannel TDMA approach. TSMP synchronisation also provides a data time-stamping service used by sensors or actuators. TSMP provides a synchronisation precision of hundreds µs and supports scaling thanks to the centralized structure of the synchronisation. *Wireless Deterministic Clock Synchronisation* (WiDeCS) [6] is another example of synchronisation protocol based on a hierarchical architecture, coupled with a MB-OFDM PHY layer and a TDMA-based MAC-layer. An impressive precision of 100ns is achieved with a VHDL implementation on a simulator.

Another synchronisation protocol category is based on a shared clock obtained via a succession of consensuses [8]. This category is interesting since these protocols do not require a centralised organisation to synchronise the nodes. One of the first synchronisation protocols in this category is *Reference Broadcast Synchronisation* (RBS) [9]. RBS is based on a "clap" broadcasted by one of the node. This "clap" is considered as a reference by the other nodes of the network: just after the reception of the clap, the receiver nodes broadcast the timestamp corresponding to the instant of the clap reception, using their own clock. The timestamp exchanges enable the nodes to compute an image of their clock offset. This mode of synchronisation is often reported as "Receiver-to-receiver" synchronisation [10]. After several cycles, the nodes converge on a shared clock, based on the consensus. *Simple Synchronisation Protocol* (SiSP) [11] is also based on a consensual process: like RBS, SiSP regularly broadcasts SYNC messages containing the consensual clock, noted "Shared Clock" (SCLK). Every node receiving a SYNC message must average the received value with its local SCLK value. After several SYNC message exchanges, the nodes share a common synchronisation. As our study is based on SiSP, this protocol will be presented in the next section of this paper.

The table 1 resume the main characteristics of the protocols cited before.

TABLE 1: MAIN CHARACTERISTICS OF CITED PROTOCOLS

Protocol	*Category*	*Physical Layer*	*Accuracy*
TPSN	Hierarchical	IEEE 802.15.	17µs
FTSP	Hierarchical	38.4kbps, 433MHz (Mica2)	10µs
TSMP	Hierarchical	IEEE 802.15.4	100µs
WiDeCS	Hierarchical	MB-OFDM	100ns
RBS	Consensual	IEEE 802.11b	30µs
SiSP	Consensual	IEEE 802.15.4	50µs

B. Trust aspects

One of the most important aspects of WSN is that the network topology is very dynamic and must be self-configured and quickly reconfigured. The changes in topology, the data burst for reconfigurations, the low data rates for power saving make synchronisation in WSN a difficult challenge. SiSP protocol presented in the next chapter seems really interesting for a lightweight, accurate, energy saving, synchronisation protocol. However, in mobile ad-hoc networks (MANET), the decentralized nature of the network makes it vulnerable to deviant node behaviours. Deviant nodes can be faulty or malicious nodes.

While faulty nodes are transmitting erroneous data, malicious node tries to corrupt the synchronisation for its own interest. These interests are various (security hack, countermeasures, misused identity…) and results in a slow synchronization convergence, a traffic increase and a network life time reduced.

Security and reliability are key parameters of WSN. Our approach is to use trust aspects to improve every aspects and deal with this problem.

The trust management is social control mechanism which enable deviant behaviour detection. The general idea is to observe neighbour node behaviours in order to compare to a "normal" behaviour. In order to process this comparison they must affect to neighbour nodes a confidence level. This confidence level being is computed by the difference between observed and normal behaviour. This confidence level can then be used to adapt or prevent interactions with untrusted nodes. These last stage consequences are entities exclusion of all "social" interactions. This exclusion called social control mechanism is often associated with the reputation notion. A node reputation is built on the long term confidence level from all trusted neighbour nodes.

Node mobility and quick network reconfigurations from WSN make it difficult to model since it is an open system. A trust in the WSN state of the art can be found in [12][13]. However most of the approaches using trust aspects have been focused on routing protocols [11], data aggregation [11] or intrusion detection [11]. Very few recent works try to apply these technics into the MAC layers. For example in [17] an evaluation of MAC layer misbehaviours in the IEEE 802.11 protocol is achieved.

III. SIMPLE SYNCHRONISATION PROTOCOL

To achieve global synchronisation over the wireless network, we have proposed in [11] a lightweight protocol called SiSP, for "Simple Synchronisation Protocol".

A. SiSP description

SiSP enables the nodes to establish a Shared Clock (SCLK) among all the nodes of the wireless network. The SCLK enables the nodes to share a common time base to coordinate medium accesses, generate timestamps for application events or synchronise sleeping periods to maximise the autonomy of a battery-powered node. The SCLK is a relative clock and is subject to deviation. It can be converted into absolute time/date by a gateway node which runs a global synchronisation protocol such as Network Time Protocol (NTP). SCLK is based on a local counter which update frequency is assumed to be the same on all nodes. SiSP aims to have these counters converge via a series of successive consensus, until the instantaneous values are the same for all nodes of the network. The main advantage of SiSP over well-known synchronisation protocols such as RBS or TPSN [10] is that it does not rely on a particular topology or even a hierarchy. SiSP is totally ad-hoc

and does not require any particular MAC precaution: SiSP messages may be included in the payload of broadcasted frames such as beacons.

In the basic version, SiSP defines only one message: SYNC. SYNC contains the SCLK value of the node at the supposed instant of the SYNC message transmission. SYNC message is broadcasted as soon as possible but does not require a constant sending interval. The smaller the interval between SYNC messages, the faster is the convergence. Upon SYNC frame reception, the SiSP algorithm described below is executed.

Algorithm 1: SiSP SYNC indication subroutine

```
rclk = extracted received clock from sispPayload
if rclk ≠ sclk:
  sclk = ( sclk + rclk ) / 2
  synchroFlag ← false
else:
  synchroFlag ← true
endif
```

Let's consider a `sispSyncIndication` subroutine (Algorithm 1) called on message reception, beacon or SYNC message: The received clock (RCLK) is extracted from the incoming message payload and compared to the receiver's SCLK. If the two values are equal, the SCLK of the two nodes have converged: the two nodes are synchronised. If the values are different, the mean is calculated, rounded and applied by the receiver node as its new value of SCLK. In a few cycles, depending on the number of nodes and the topology, a consensus is reached. SCLK consensus is obtained in a decentralized way, without any hierarchical prerogative. Moreover, SiSP algorithm is simple (integer sums and division by 2) and perfectly fits on tiniest microprocessors.

B. *Protocol Weaknesses and deviant behaviours*

However the simplicity of SiSP makes it vulnerable to deviant behaviours. In order to deal with these weaknesses, we are going to make a survey of these behaviours. This section tries to list both faulty and malicious behaviours. We will take in consideration transmissions them-selves and consequences of these transmissions in order to define symptoms which can be observed at a network level.

TABLE 2 :ATTACKS DESCRIPTION

Attack Name	*Description*
Random clock (RC)	The node transmit random shared clock. This attack only tries to modify the shared clock value.
Fixed clock (FC)	The node doesn't modify its own shared clock. This attack only tries to modify the shared clock value.
Misused identity (MI)	The node transmits a faulty or misused source address. This attack tries to fake a node identity.
Quick clock (QC)	The node transmits its shared clock more often than planned. The shared clock value isn't targeted. This attack doesn't respect the normal behaviour rules.

A node using SiSP assumes that all its neighbours have a normal behaviour. It is then vulnerable to bad data or commands from these neighbours. The deviant behaviour observed can be the consequences from faulty hardware, bugs or even malicious nodes trying to break or disturb the synchronisation. Since WSN are open systems, they are particularly vulnerable to malicious node intrusion.

In this work we will focus on malicious behaviours where the node manufacturer intention is to disturb the WSN. We are going to address the malicious behaviours listed in table 2.

IV. TRUSTED SISP

In order to deal with these attacks we propose to enhance SiSP with trust aspects. However as mentioned above a confidence policy is quite complex to develop because a WSN node has constraints:

- Low computational power often not compatible with a transmission history management used to detect malicious behaviours,
- Distributed trust management because it is not possible to use a global network model,
- No authentication is used which is a crucial problem since in all approaches using trust aspects, a confidence level is associated to an identity.

The last point is really important and in this paper we propose to use trust as it is done in [5] [5] for WSN routing. However, in our case we use the confidence level to estimate neighbourhood reliability instead of an independent neighbour level. An untrusty neighbourhood is a node set where at least one malicious or faulty node is detected. As soon as a node supposes its neighbourhood untrusty, it enters a degraded mode. The degraded mode is a minimal function mode where the node only does what the network is expecting from it. This mode must prevent any neighbourhood interactions for shared clock synchronisation. The whole neighbourhood will be in quarantine when all the nodes of a zone will adopt the degraded mode.

In the next subsections we are going to address attacks listed in table 2 using trust aspects. For each attack we present observed symptoms and countermeasures envisaged.

A. *Attack symptoms and countermeasures*

For the "Random Clock" (RC) and "Fixed Clock" (FC) attacks, it is the shared clock value that is targeted. In these cases a history of neighbour last values should detect these attacks. The standard deviation of the value will tell us if the value is fixed or converging.

In the case of "Misused Identity" (MI) attacks, the lie can only be detected if the node identity is misused in its neighbourhood (confidence level will be drastically decreased). If a lie is suspected, the confidence level will be decreased proportionally to the probability of the deviance behaviour. For instance, a neighbourhood where new identities abnormally increase should be penalized.

For the "Quick Clock" (QC) attack, if a node does not lie on its identity, this behaviour is easily detected. If it is not the case the attack will be a particular case of the "Misused Identity" attack.

B. Penalization

When the attacks are detected the node confidence level should be penalized. This level will be decreased accordingly to malicious level. These levels are presented in table 3 where *n* is the identity of the node and *id* the identity of a neighbour node.

TABLE 3: ATTACKS PENALIZATION

Attacks	*Detection*	*Penalization*
MI	id = n	trust(id)=0
MI	New neighbours abnormally increase	$trust(id)_{new\ neighbours} == 0$
FC	Too small shared clock deviation estimated compared to SCLK(n)	$trust(id)_{t+1} == trust(id)_t - \alpha t$
RC	Not correlated shared clock compared to SCLK(n)	$trust(id)_{t+1} == trust(id)_t - \beta$
QC	Too many SYNC(id) messages	$trust(id)_{t+1} == trust(id)_t - \chi$

On the other hand it is necessary to restore the confidence level over time for both node and neighborhood. Indeed, even a low mobility makes malicious nodes able to quit a neighborhood for another one. Moreover the malicious nodes can be suppressed from the network by moving to a bad location or even depleted batteries. In some cases a mistrust neighborhood case could result from several exceptional events coincidence. In this case the confidence should be restored over the time by slowly increasing the confidence level. This mechanism is described by Algorithm 2, where e in [0,1] is generally called forgetfulness constant.

Algorithm 2: Confidence level restoration

```
for each id in neigboorList do:
  trust(id) = (1 - e) trust(id) + e
done
```

C. Countermeasures

Trust in identifiers is used to estimate trust in an entire neighbourhood. A neighbourhood is considered untrusty when one of its neighbours has a low confidence level.

The only countermeasure envisaged in this paper is to enter in a degraded mode. In this mode the node still listen to SYNC messages but do not send its own SYNC message until the confidence level is over the threshold. The goal of this mode is the progressive isolation of the presumed attacker node. This should resume quarantining the zone around the attacker.

V. SIMULATIONS RESULTS

A. Simulation parameters

We have used a tool called MASH (*Multi-agent Software/Hardware*) simulator [5] [5]. It enables both the simulation and the execution of embedded distributed and decentralized systems:

- using real world agent behaviours by injecting them in the simulated system (an interaction protocol can be seen as a pattern of behaviour, the behaviour is inferred from transmitted messages);
- using realistic physical models at the global level (environment models, wave propagation models, etc.) or at node level (energy consumption model, etc.). These models are computed with external tools (Matlab/Simulink, Labview).

The figure 1 presents on the left side a sensor network

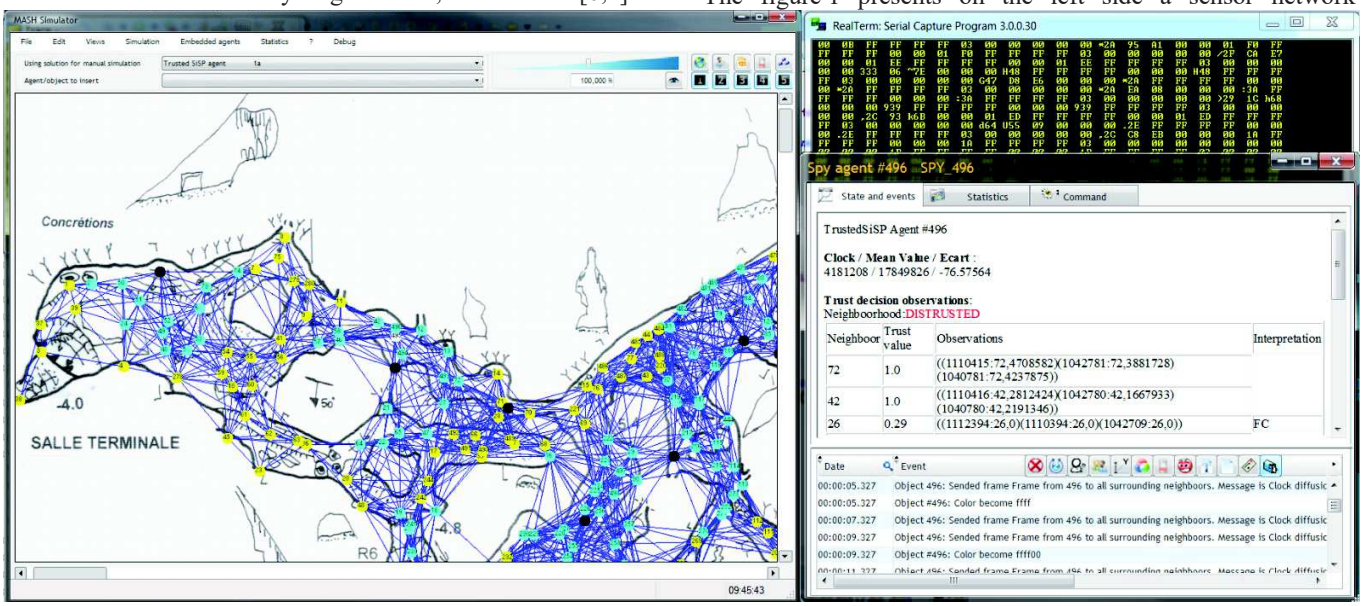

Figure 1. MASH simulator

deployed upon an underground river. On the top right hexadecimal format frames are displayed. In this figure frames are transmitted from a simulated agent to a real embedded agent. On the bottom left a window is displayed when you click on an agent in order to spy it. The figure present the agent number 496 details: its clock (4181208), the global mean clock (17849826) this clock is not known by the agent and the difference between the 2 clocks. Under these values we can notice that this node consider its neighbour, the agent number 26, which has a confidence level of 0.29, untrusty. It suspects that agent 26 is making a FC attack. The consequence is that the agent 496 enters the degraded mode not propagating its own clock.

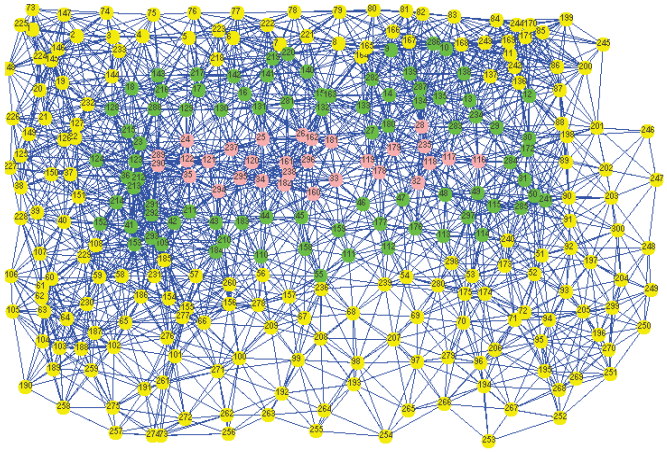

Figure 2. Test topology of trusted SiSP

For all the simulation presented hereafter, the goal is to synchronise 300 nodes. The penalization parameters are α=0.8, β=0.05, χ=0.2. These parameters have been chosen because results presented in this paper focus on FC attacks; a better value for β should be used to prevent RC attacks in future works. The threshold used for going into degraded mode is fixed to 0.5.

The test field is 750m x 500m and node communication range is fixed to 50m. The attackers are clustered in the centre of topology and all attackers perform a FC attack. All nodes start with a local clock value of 0 or 40M randomly chosen. The figure 2 presents a running simulation. The pink colour represents the attackers, the green colour represents nodes entered in degraded mode and a more or less shaded yellow colour represents the value of the local clock.

Each simulation lasts 5 minutes. For each simulation we present the convergence time between the ideal convergence clock value (without any attacks) and convergence clock value with attackers. In parameters we have represented in blue the ideal convergence clock value. Respectively in red, green and purple the convergence clock with 1%, 5% and 10 % of malicious node performing a fixed clock to 0.

B. Results

The results are presented in figure 3. In the right of this figure, we see the local clock chosen randomly. After this sequence the simulation starts and then, for the ideal case curve, the clock starts converging to the average clock value. The simulation average should be 20M but it is simulation dependant since the starting clock value is randomly chosen.

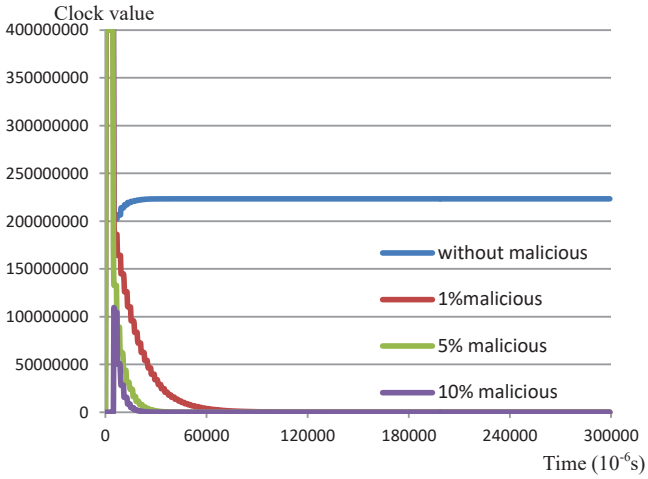

Figure 3. SiSP clock value convergence

The most interesting point is that as soon as we introduce malicious node the clock value starts decreasing towards 0. The more we introduce malicious nodes, the faster the convergence towards 0 is. This does not mean that nodes are synchronised since the clock is forced to 0. This phenomenon, describes that the attack succeeded in desynchronising the whole system. SiSP protocol is a very simple protocol not implementing safety or security countermeasures and as consequences is very sensitive to malicious nodes. These points will be treated by the trusted SiSP protocol proposed.

The figure 4 describes the trusted SiSP results. It describes the clock value difference between the ideal case with no malicious nodes and scenarios with malicious nodes introduced. The first interesting point is that the nodes are synchronised. Even if it is not the ideal clock value the nodes are synchronised. The more we introduce malicious nodes, the more the difference increases. This is due to our computation of the average clock value. Indeed this value is computed including attackers which have a fixed clock to 0.

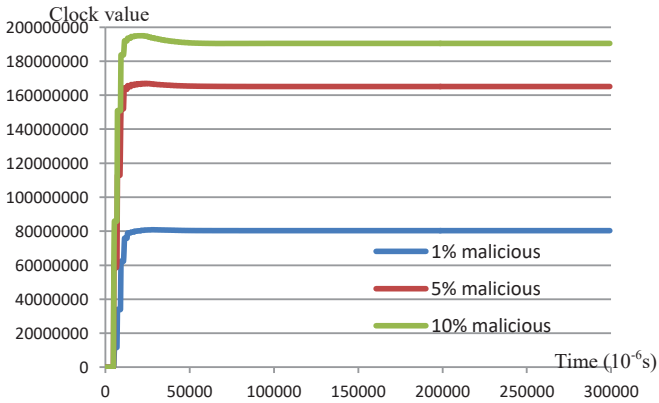

Figure 4. Trusted SiSP Clock value convergence

On this figure we can see that difference quickly increases at the simulation beginning and then stop and decreases. This is particularly visible on the 10% malicious curve. The starting of the decrease is due to attack detection and isolation of malicious nodes.

These first experiments show that FC attacks are very effective on the SiSP protocol since a malicious node can succeed to desynchronize the whole system. These attacks are detected by the trusted SiSP protocol presented in this paper. The degraded mode enables to isolate the attackers and to keep synchronisation active.

VI. Conclusions and perspectives

In this paper we have presented a trusted lightweight synchronisation protocol based on the SiSP protocol. We have proposed a very flexible solution able to detect several kinds of attacks such as Fixed Clock, Random Clock, Misused Identity… The only countermeasure presented is that when the attack is detected, the node enters a degraded mode. This countermeasure has demonstrated that it can help keeping the synchronisation while SiSP protocol cannot.

We are currently working on other kind of countermeasures by adapting the degraded mode enabling a better clock value difference. Other attacks are going to be deeply investigated in order to find the good α, β, χ, to detect and prevent all kinds of attack type. Since the MASH simulator enables hardware emulation, our next simulation should then quickly implement hardware in the loop parameters.

References

[1] Ian F. Akyildiz, Xudong Wang, Weilin Wang, Wireless mesh networks: a survey, Computer Networks, Volume 47, Issue 4, 15 March 2005, Pages 445-487, ISSN 1389-1286, 2005

[2] Y. Seongwook, A Comparison of Clock Synchronization in Wireless Sensor Networks, International Journal of Distributed Sensor Networks, vol. 2013, Article ID 532986, 10 pages, 2013.

[3] S. Ganeriwal, R. Kumar, M.B. Srivastava, Timing-sync Protocol for Sensor Networks, The First ACM Conference on Embedded Networked Sensor Systems (SenSys), p. 138-149, 2003

[4] M. Maroti, B. Kusy, G. Simon, A. Ledeczi, The Flooding Synchronization Protocol,. Second ACM Conference on Embedded Networked Sensor Systems (SenSys), 2004.

[5] K.S.J. Pister, L. Doherty, "TSMP: Time Synchronized Mesh Protocol", In Proceedings of the IASTED International Symposium on Distributed Sensor Networks (DSN08), 2008

[6] T. Beluch, D. Dragomirescu, F. Perget and R. Plana, "Cross-Layered Synchronization Protocol for Wireless Sensor Networks," Networks (ICN), 2010 Ninth International Conference on, Menuires, 2010, pp. 167-172.

[7] K. S. Yildirim and A. Kantarci, Time Synchronization Based on Slow-Flooding in Wireless Sensor Networks, IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 1, pp. 244-253, Jan. 2014.

[8] M. K. Maggs, S. G. O'Keefe and D. V. Thiel, Consensus Clock Synchronization for Wireless Sensor Networks, IEEE Sensors Journal, vol. 12, no. 6, pp. 2269-2277, June 2012.

[9] J. Elson, D. Estrin, Fine-Grained Network Time Synchronization using Reference Broadcast, The Fifth Symposium on Operating Systems Design and Implementation (OSDI), p. 147-163, 2002

[10] M. Roche, Time Synchronization in Wireless Networks, 2006, http://www.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/ Accessed on 13th March 2016

[11] A. Van Den Bossche, T. Val, R. Dalcé, SiSP: A lightweight synchronization protocol for Wireless Sensor Networks, 16th IEEE conference on Emerging Technologies & Factory Automation (ETFA), 2011

[12] L. Vercouter, J.-P. Jamont, Lightweight trusted routing for wireless sensor networks. Progress in Artificial Intelligence, 193-202, 2012.

[13] G. Han, J. Jiang, L. Shu, J. Niu, H.-C. Chao,Management and applications of trust in Wireless Sensor Networks: A survey. J. Comput. Syst. Sci. 80(3): 602-617 (2014)

[14] F. Bao, I. R. Chen ; M. Chang ; J. H. Cho, Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection, IEEE Transactions on Network and Service Management, pp169 – 183, vol 9, issue 2, 2012.

[15] M. Rezvani, A. Ignjatovic, E. Bertino, S. Jha,Secure Data Aggregation Technique for Wireless Sensor Networks in the Presence of Collusion Attacks. IEEE Trans. Dependable Sec. Comput. 12(1): 98-110 (2015)

[16] M. Singh , R. Das, M. Kanti Sarkar, K. Majumder, S. Kumar Sarkar, KT3F: A Key-Based Two-Tier Trust Management Filtering Scheme for Intrusion Detection in Wireless Sensor Network, Proceedings of the Second International Conference on Computer and Communication Technologies Volume 379,pp 679-690, Springer, 2015

[17] H. Diwanji and J. Shah, "Effect of MAC layer protocol in building trust and reputation scheme in mobile ad hoc network," in Engineering (NUiCONE), 2013 Nirma University International Conference on, Nov 2013, pp. 1–3

[18] L. Vercouter, J.-P. Jamont: Lightweight Trusted Routing for Wireless Sensor Networks, Advances on Practical Applications of Agents and Multiagent Systems - 9th International Conference on Practical Applications of Agents and Multiagent Systems, pp87—96, Springer, 2011.

[19] Jean-Paul Jamont, Michel Occello, Eduardo Mendes, Decentralized Intelligent Real World Embedded Systems: A Tool to Tune Design and Deployment, Advances on Practical Applications of Agents and Multi-Agent Systems, pp133-144, Springer, 2013.

[20] Jean-Paul Jamont, Michel Occello: Meeting the challenges of decentralised embedded applications using multi-agent systems. International Journal of Agent-Oriented Software Engineering, 5(1): 22-68 (2015)