Improving Wireless Sensor Network Resilience with the INTERSECTION Framework

Gareth Tyson, Adam T. Lindsay, Steven Simpson, and David Hutchison

Computing Department, Lancaster University
Lancaster, LA1 4WA, UK
{g.tyson,atl,ss,dh}@comp.lancs.ac.uk

Abstract. This paper investigates the INTERSECTION Framework's ability to build resilience into Wireless Sensor Networks. The framework's general details are examined along with general approaches to protection against misbehaving WSN nodes. Three different approaches to remediation were implemented and tested on a TinyOS network for their different operating characteristics.

Key words: WSN, remediation, sensor networks, distributed systems

1 Introduction

Many aspects of computing involve the need to intelligently and dynamically select between many possibilities to achieve optimisation. One example of such a field is *network resilience*. Modern networks are exposed to a wide-range of possible security attacks that can be remediated through a variety of mechansisms. The suitability of these mechanisms, however, can dynamically vary over time and therefore the optimality of a decision can often only be resolved at attack-time. Design-time 'best practice' approaches to defending against such attacks therefore become ineffective.

The INTERSECTION framework [1] was created to address these concerns. It offers an extensible, modular and dynamic remediation service for dealing with security and resilience issues in networked infrastructure. The framework executes a control loop that takes distributed network observations that fuel the deployment of optimisations for handling security threats. Importantly, decisions are taken dynamically (and autonomously) to reflect current operating conditions as well as higher level policy concerns. This framework has successfully been deployed in a range of environments, including Internet service providers and satellite networks. However, one field that provides a more significant challenge is the deployment of these principles within *wireless sensor networks*.

Wireless sensor networks have distinct requirements with respect to resilience due to their high vulnerability and limited defence capabilities. This stems from their often physically open deployment alongside their limited processing and battery capacity. These design concerns result in the frequent inability to utilise sufficiently sophisticated techniques to defend against security threats. Further, even 'best practice' approaches can vary in their effectiveness, making the possible overheads they introduce unnecessary.

This paper details the application of the INTERSECTION framework in the realm of wireless sensor networking. To achieve this, a case-study has been chosen which we term *the misbehaving node*. This anomaly consists of a mis-function where a node is in an operating state that negatively affects other nodes in the network. This could be through a malicious reconfiguration or a hardware-related malfunction. Due to resource restrictions, many WSN communications protocols are insecure and susceptible to such anomalies. Further, the nature of the anomalous behaviour can vary greatly between different networks. As such, we use this to highlight the advantages of utilising the INTERSECTION framework. We present three possible remediations to the attack before investigating them using both quantitative and qualitative means.

2 The INTERSECTION Framework

This section details the INTERSECTION security framework and its use in securing wireless sensor networks.

2.1 Overview

The INTERSECTION framework is a security framework developed as part of the EU FP-7 INTERSECTION project. It offers a holistic approach to network security management and strives towards the autonomic defence of network infrastructure. It works, in essence, on a control-loop through which it consumes statistics, detects attacks and then deploys remediations. This chain of operation is fuelled by the ability to configure each instantiation with rules defined by its operating conditions and higher level policies. It was informed in no small part by the ResiliNets initiative [2].

The framework has been tested and deployed within a range of network technologies ranging from traditional IP infrastructure to satellite networks. This paper, however, focusses on the framework's use in the field of wireless sensor network (WSN) security. As such, a subset of its relevant functionality is now detailed; readers are directed to [1] for a full architectural overview.

2.2 Probes

The first component used in the INTERSECTION framework's WSN support is the *probe*. A probe is a entity that resides in the network with the task of collecting statistics. There are no predefined set of probes that must be inserted into the network. Instead, they can be dynamically added and removed.

Within a WSN deployment the probe is most likely to reside at the sink. This allows the probe to collect important statistics such as packet rate, packet size packet contents etc. Alongside this, probes can also exist within the network on one or more sensor nodes. These nodes, however, will usually be required to forward their data through the sink and therefore can be considered as constituents of the probe at the sink. Each node collects statistics – reception rates of packets from each of its neighbours, implemented as an additional sensor value – and forwards them to the sink. The sink then collects this information and passes it to the rest of the framework using IPFIX [3] messages.

2.3 Detection Engine

The second important component is the *Detection Engine*. This is responsible for processing statistics gathered by the probes. It is a pluggable component that resides (by default) at a remote location. It receives IPFIX messages from probes and uses detection techniques to detect any attacks [4].

If an attack is detected, the Detection Engine generates a message containing the identifier of the attack alongside any important parameters (e.g. the source). Messages are formatted using IDMEF [5] which is a standard XML message format for describing network intrusions. This IDMEF message is then forwarded to the Remediation Engine using a pub/sub middleware (e.g. JMS [6]).

2.4 Remediation Engine (PITS)

The third important component is the *Remediation Engine*; the current implementation of this is called PITS (Pluggable Intrusion Tolerance System) [7]. PITS is a constituent element of the INTERSECTION framework that is responsible for deploying, revoking and managing remediation mechanisms within the network. Remediation contrasts with traditional 'best practices' as it is a *temporary* measure to resolve a particular runtime problem. As such, it incurs a specific cost that must be weighed against its benefits.

PITS listens for alerts from the Detection Engine regarding any attacks occurring under its jurisdiction. This is identified by the parameters provided within the IDMEF message.

Once the alert is received, the Remediation Engine is responsible for selecting the best remediation for the event. This is based on extensible policy rules that are associated with each Remediation Engine that reflect a cross between policy, capabilities, and run-time information on the anomaly. Each remediation is implemented as a pluggable component that can be dynamically instantiated at a remote location to counteract a given attack. The Remediation Engine therefore deploys these properly configured components to these remote locations (termed Remediation Points) on demand. PITS can reside at any accessible location in the network, but in this scenario, PITS is deployed at the WSN access point.

2.5 Remediation Points

The fourth important component is the *Remediation Point* (RP). These are programmable entities within the INTERSECTION framework that reside at strategic locations in the network. For instance, in an IP network various border routers are likely to be Remediation Points. RPs listen for reconfiguration requests from the Remediation Engine, informing them to perform specific actions. Concretely, they receive requests to instantiate pluggable remediation components that can counteract some observed attack. Requests are structured as XML documents that stipulate which components to instantiate, how to connect them and any required parameters.

When the reconfiguration request is received, the Remediation Point begins to perform the remediation. Alongside this, it also collects statistics that can be fed back to the Detection and Remediation Engines so that they may signal the withdrawal of the remediation. Current WSN operating systems and software do not typically support dynamic reconfiguration to this extent and therefore any remediation functionality must be statically deployed on each node. This functionality is then invoked using authenticated protocol messages sent by the sink (connected to a more powerful machine). We are currently also investigating the use of emerging dynamic WSN operating systems such as Lorien [8] which would offer the necessary capabilities.

3 The misbehaving node

This section details the anomaly of interest with the intent of highlighting the potential of the INTERSECTION framework.

3.1 Scenario

The anomaly investigated in this paper is termed the *misbehaving node* scenario. The scenario is an by-product of lightweight – and insecure – configuration protocols utilised by simple wireless sensor network deployments, such as the Collection Tree Protocol [9].

Many sensor networks use remote configuration protocols that allow nodes to be configured and reconfigured dynamically post-deployment. This has many benefits, specifically allowing a large number of nodes to have their behaviour modified without the need for staff to manually re-program all nodes. We term this method of dynamic re-configuration a *control protocol.*

Control protocols are generally application specific and designed by the developer to match their exact needs. These needs can vary between different deployments; common examples are sensor sampling rate, remote update rate and which sensors to probe.

There are a countless ways for a wireless sensor node to misbehave in the face of hardware damage or malicious or negligent reconfiguration. To provide focus, however, we address one particular type of anomaly – one that adversely affects the well-being of the sensor network as a whole – termed *packet injection.* This occurs when the misbehaving node has an increased data transmission rate. This is dangerous as all packets received by the root appear valid, as if they had been generated by an existing member of the network. This has three effects:

- Denial of Service: Nodes within the network will have an increased routing load. This can result in greater packet loss and delay.
- Battery Drain: The increased routing load will result in high battery consumption from all nodes 'downstream' from the anomaly.
- Untrustworthy data: The increased transmission rate may cause spurious or untrustworthy data to be entered into the system.

3.2 Intuitive Solutions

A number of intuitive approaches exist to remediate against this anomaly. The first would be *encryption*; this would involve using encryption to authenticate control messages. This is feasible in some situations, however, such authentication algorithms may take up to minutes to process. Depending on the frequency of control messages this can be considered too long for many applications. More recent advances can reduce that to seconds [10], though at greater expense in cost and energy. Further, the complexity of using such algorithms could be considered to be inappropriate by many developers that do not see their sensor networks as susceptible enough to such anomalies to justify the increased ongoing cost.

The second solution would be to remove the usage of the control protocol, therefore eliminating the vulnerability. Once again, this is a possibility, however, this cannot be performed if the control protocol is integral to the network's correct operation. Alternatively, only a subset of the protocol could be disabled; this, however, only mitigates the problem – it does not solve it.

The final intuitive solution that we consider is placing hard coded limitations on the protocol's parameters. This would prevent malicious reconfigurations that emit values that lie outside a certain threshold (e.g. X must be > 10 and < 20). This, once again, would mitigate the problem but not solve it. Also, in many situations the defence could be worked around by making small variations on many nodes rather than one large variations on a few nodes.

4 WSN Deployment

This section details three pluggable remediation mechanisms that are utilised by the INTERSECTION framework to handle the security problems outlined in Section 3. Each can be dynamically selected based on the environment in which the WSN and attack(s) operate in.

4.1 INTERSECTION Configuration

As discussed in Section 2, the framework consists of four primary elements: probes, reaction engine(s), remediation engine(s) and remediation point(s). Within the current WSN setup, these are deployed as follows:

- Probes: Each sensor node acts as a probe. It counts the number of packets it receives from each of its neighbours then periodically forwards this information

to the root. The root also acts as a probe by counting the number of received packets. Both sets of information are then forwarded to the Reaction Engine.
– Reaction Engine: One Reaction Engine exists in the setup. This is located at a remote host that has connectivity with the root.
– Remediation Engine: One Remediation Engine operates at the root. This is configured with policy rules based on the particular WSN.
– Remediation Point: The Java implementation of the Remediation Point is co-located with the Remediation Engine. It interacts with WSN using a serial connection to the root through which is can forward messages through. Each sensor node also has remediation functionality that can be invoked by the Remediation Point using a authenticated protocol.

4.2 Remediation Mechanisms

To resolve the case-study attack detailed in Section 3, three remediations are proposed. These are embodied in pluggable components that can be selected and instantiated at the Remediation Point located at the WSN root. Once instantiated, they disseminate an authenticated protocol messages into the WSN informing a particular node to instantiate a remediation strategy. These possible strategies are now detailed.

Node Re-Initialisation This is the simplest remediation available as it involves no network reconfiguration. Instead, the remediation mechanism simply sends a reset message to the compromised node, informing it to return to its default configuration. Subsequently, it should return to its normal rate.

Route Around The second approach remediates against the possibility that a node cannot be reinitialised. This occurs when an attacker repeatedly misconfigures the node by periodically sending malicious control messages. Therefore, whenever the node is reinitialised, the attacker re-sends the command telling the node to increase its sending rate.

The INTERSECTION remediation for this attack is to broadcast a reconfiguration message to all nodes. This message informs the recipients to stop forwarding messages from the compromised node, effectively shunning the anomaly. Subsequently, the node's messages are limited to one hop in the collection tree.

Node Shutdown The third approach remediates against the situation in which the compromised node is sending at such a high rate, that its immediate neighbours (within radio range) are having their batteries depleted by processing the messages. Therefore, the compromised node is still having a significant impact on a subset of the nodes. To remedy this, all nodes within radio range of the compromised node are shutdown so that it remains quarantined without the necessity to filter its messages. These node subsequently operate with resume timeouts so that the nodes turn resume operation after a given period. If the attack is still taking place and the circumstances haven't changed, these nodes can then be shutdown again.

5 Evaluation

This section evaluates the INTERSECTION framework's ability to provide on-demand remediation in a wireless sensor environment. The key aim of the evaluation is to examine the framework's ability to select and deploy appropriate remediations based on its operating context. To this end, the performance and overhead of each of the remediations are examined. Following this, each strategy is investigated to find its suitability in different situations.

5.1 Methodology

The INTERSECTION framework has been implemented in Java alongside the three remediation mechanisms by modifying the CTP [9] implementation in TinyOS [11]. Using this modified CTP, a simple sensor sampling application is setup in the TOSSIM simulator. A 64 node grid topology is constructed with each node spaced 2 metres apart with a noise floor of -105 dBM. Every minute, each node sends a sensor reading to the sink. The attack begins after 60 minutes; this attack involves a single node's sending rate increasing from 1/min to 1200/min. For the purposes of this evaluation, an alert simulating detection of the anomaly is generated after 4 minutes, and the chosen remediation is deployed.

5.2 Evaluation of Remediation Mechanisms

As shown in Section 4, there often exist multiple approaches to correcting an anomaly. Subsequently, to make an informed decision about remediation strategies it is necessary to understand the behaviour and characteristics of the available remediations. This section now explores the performance and overhead issues of each of the three possible remediations detailed in this paper.

Performance To evaluate the performance of the remediation mechanisms we use the packet reception rate. This can simply be defined as the number of packets received by the CTP root over a given period of time. Any deployed sensor network will have upper and lower bounds on this reception rate; subsequently, extended periods of operation outside of these limits indicates an anomaly. Figure 1 shows the reception rate at the root when operating with the three remediation strategies. It can be observed that after 60 minutes, the attack begins with a dramatic increase in the reception rate. The detection mechanism alerts the INTERSECTION framework 4 minutes after the attack starts and subsequently a remediation is deployed.

The *re-initialisation* remediation is the most effective with an immediate drop in the reception rate. Further, the network is returned entirely to its previous state without any drops below the original reception rate.

The *route around* mechanism also has similar results; a notable difference, however, is that this remediation involves severing the compromised node from the network. Subsequently, its data is lost and the reception rate drops to reflect this loss of data.

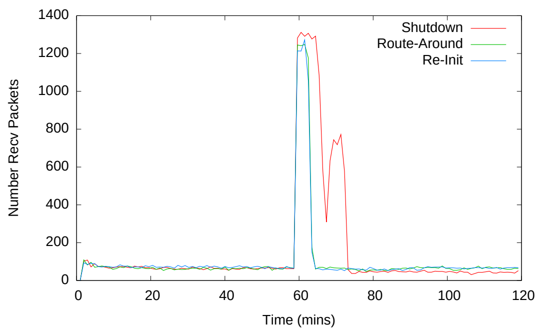

Fig. 1. Packet Reception Rate at Root with Different Remediations

Remediation	Before	After	Diff
Re-Init	68	68	0
Route Around	61	58	3
Shutdown	65	43	22

Table 1. Reception Rates Before and After Remediations (per min)

The *shutdown* remediation has the most significant effect on the reception rate. Once this remediation is executed, the nodes that neighbour the compromised node begin to shutdown. This happens progressively as the dissemination protocol contacts each node. As it is necessary to contact a greater number of nodes (18) than previously, the reaction of this remediation is slower. Once the remediation begins to take effect there is a sharp fall in the reception rate. However, as not all neighbouring nodes have been shutdown yet, it becomes possible for the compromised node to recover its route to the sink by using alternative hops. This results in an increase, yet again, in the reception rate. This, however, is short-lived as the rest of the neighbours soon receive the remediation command, thereby fully quarantining the compromised node from the rest of the network. Once the remediation has taken effect, the reception rate can be seen to be well below the previous level due to the number of network members that cease to send their sensor data.

A summary of the results is shown in Table 5.2. This shows the average reception rate (per minute) both before and after the remediations have been put in place. It can be seen that the re-initialisation remediation has little effect on the reception rate. In contrast, the route around and shutdown remediations decrease the sending rates. This results in fewer sensor samples being received.

Overhead A vital consideration when deploying a remediation is its overhead. A remediation that is not associated with an overhead should be considered a best practice that should be under constant deployment. Therefore, generally the overhead of a remediation forms the foundation for the selection criteria. This section now explores the overhead of the three remediations investigated.

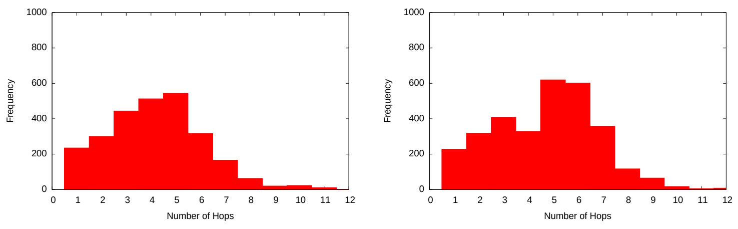

Fig. 2. Histogram of Routing Hops (i) before and (ii) after Route-Around remediation

Two forms of overhead are identified; the first is an increased number of hops between the clients and the root. The second is an increased delay between the clients and the root.

The *re-initialisation* remediation is, unsurprisingly, the optimal mechanism in terms of overhead. Once the attack is detected it is simply a matter of sending a single message to the compromised node. Following this, the sending rate returns to normal and no further overhead is suffered.

In contrast, the *route-around* remediation has a larger overhead as it involves the modification of routing behaviour. Figure 2 provides a histogram of the number of hops both before and after the remediation is put in place. After executing the remediation within the experiment, 28 nodes suffer from an increased number of hops. On average, these nodes witness an increase of 1.2 hops. This subsequently means that the network will suffer from an increased level of battery consumption as the routing process is spread over an increased number of nodes. Despite this, there is little discernible effect in terms of the delay observed within the network; this can be attributed to the ease at which the nodes can route around the single failure.

The *shutdown* remediation has the greatest overhead because it creates the greatest disruption in the network. Within the experiment, 18 nodes (28%) are within range of the compromised node; subsequently, these members of the network must be shutdown to quarantine the attack and preserve their batteries. Figure 3 shows the effect that the remediation has on the number of hops. It can be observed that this has a significant impact on the number of hops a node requires to contact the root. Before the remediation, the mode average of hops is 5, however, this increases to 7 after the remediation is put in place. This increase is driven by the 28 nodes (44%) in the network which find their route(s) compromised by the remediation. These nodes witness a mean increase of 2.3 hops. Globally, this results in the mean average increasing from 4.9 hops to 6.2. Unlike, the route-around remediation, however, this increase does result in a noticeable change in delay. Before the remediation, the nodes have an average delay of 733 ms, however, after the remediation this increases by 137 ms to 870 ms.

5.3 Analysis of Remediation Characteristics

The previous quantitative investigation has exposed the differences between the behaviour of each remediation strategy. Despite their collective ability to resolve

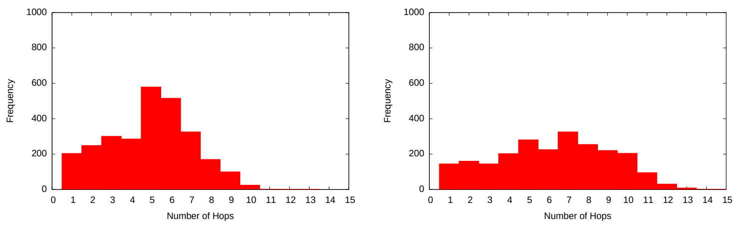

Fig. 3. Histogram of Routing Hops (i) before and (ii) after Shutdown remediation

the attack, it is evident that there are different advantages and costs related with each. This section now explores these with a mind to ascertaining their suitability in different circumstances.

Re-initialisation Remediation The first remediation works on the basis that the anomaly is a 'one off,' so that the misbehaving node can be returned to normal with a software reset, and there is no malicious party that will reconfigure the node again. If these are not the case, the remediation is ineffective because every time the re-initialisation takes place, the anomaly will reoccur. Despite this, it is by far the most effective as it capable of returning the network to completely normal operating conditions. The performance and overheads of the re-initialisation remediation are therefore the most attractive.

This remediation should always be the first to be attempted as it offers the greatest utility at the lowest cost. However, its deployment must also be performed alongside continued monitoring. If it is unsuccessful in its task then the remediation must be withdrawn and replaced with an alternative.

Route-Around Remediation The second remediation is a network level solution that prevents the misbehaving node from polluting the network with an increased packet rate. Unlike the previous remediation, this remediation is resilient against the continuing presence of the attacker. However, this also results in the loss of data from the compromised sensors. Subsequently, if the data is highly important and the network can handle the increased traffic, it could be considered a superior option to allow the node to continue sending.

This remediation should be the default choice after the failure of the re-initialisation remediation as it will ensure that the compromised node is quarantined. It is highly feasible in most circumstances and provides the best performance when operating with a continually present attacker. However, because the compromised node remains in contact with the other nodes, there can be a notable level of battery consumption due to their new function as filters. This largely depends on the sending rate and range of the compromised node.

Shutdown Remediation The third remediation is an extreme solution that can be taken if the compromised node's sending rate is adversely affecting those around it. It works by shutting down all peers that are within the transmission range of the offending node. This therefore maintains quarantine whilst also ensuring that the surrounding nodes do not have to expend energy filtering the

Factor	Re-init	Route Around	Shutdown
Effectiveness	Low	High	High
Increased Delay	None	Limited	High
Increased Hops	None	Limited	High
Loss of data	None	Limited	High
Increased Battery	None	None	Reduced

Table 2. Overview of Remediation Strategies

Algorithm 1 Default Algorithm for Remediation Selection

```
1: success = deployReInit();
2: if success == true then
3:    return SUCCESS;
4: else if (sensor data not critical || high replication in network) && battery life
   important then
5:    deployShutdown();
6: else
7:    deployRouteAround();
8: end if
```

messages they receive. This thereby lowers the battery consumption of the neighbouring nodes so that they can be restarted at a later date once the offending node has been removed.

Shutdown has the side-effect of removing the relevant nodes' sensor data from the system. This can be considered a necessary sacrifice (as defined by policy) if the human interception time is extended or if the data is not vital (e.g., it can be covered by other sensors). The selection of the shutdown remediation is therefore largely based on environmental and application issues that would indicate it is acceptable to shutdown the nodes in favour of maintaining their batteries.

Summary It is evident that the three remediation mechanism are suitable for deployment when operating in different conditions. Table 1 provides an overview of their differences. To formalise these results, Algorithm 1 also details the selection policy. In essence, the re-initialise remediation should *always* be tried first. If this fails, one of the other two remediations should be deployed. If the sensor data is not critical or, alternatively, there are many sensors that can replace the lost ones, the next approach is to perform the shutdown remediation (assuming battery preservation is important). However, if these pre-requisites are not fulfilled then the policy default is to use the route around remediation.

6 Conclusion

This paper has outlined the application of the INTERSECTION framework in the domain of wireless sensor networking. The *misbehaving node* wireless sensor anomaly has been detailed alongside a set of three possible remediations.

This has then been placed in the context of the INTERSECTION framework to investigate the advantages of allowing such a security system to dynamically select which remediation to deploy based on environmental conditions and higher level policy concerns. Through this, it has been shown that the three remediations – re-initialisation, route-around, and shutdown – possess greatly different characteristics and, as such, are not uniformly suitable for all situations.

There are a number of possible areas of future work. It is necessary to investigate the usage of the INTERSECTION approach with a range of different security threats. Different WSN routing protocols should be investigated for their applicability to the lightweight and flexible remediation strategies we examined in the context of the CTP protocol. Finally, the research would be greatly strengthened with a more ecological investigation into varied range of actual environmental deployments.

Acknowledgements. This research is funded by the EU INTERSECTION Project under grant agreement no. 216585. The authors thank Utz Roedig for his feedback on early drafts.

References

1. D'Antonio, S., Romano, S.P., Simpson, S., Smith, P., Hutchison, D.: A Semi-Autonomic Framework for Intrusion Tolerance in Heterogeneous Networks. In: IWSOS. (2008)
2. Sterbenz, J.P., Hutchison, D.: Resilinets wiki. http://wiki.ittc.ku.edu/resilinets_wiki/
3. Claise, B.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information. Number 5101 in RFC. IETF (January 2008)
4. Coppolino, L., D'Antonio, S., Esposito, M., Romano, L.: Exploiting diversity and correlation to improve the performance of intrusion detection systems. In: International Conference on Network and Service Security, Paris, France (June 2009)
5. Debar, e.a.: Ietf intrusion detection exchange format working group. Internet Draft. IETF (2004)
6. Sun: Java Message Service (JMS). http://java.sun.com/products/jms/
7. Simpson, S.: Pluggable Intrusion Tolerance System (PITS). http://www.activenet.lancs.ac.uk/pits/
8. Porter, B., Coulson, G.: Lorien: a pure dynamic component-based operating system for wireless sensor networks. In: 4th International Workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks (MidSens'09), Urbana Champaign, Illinois, USA, ACM (2009) 7–12
9. TinyOS Network Working Group: The Collection Tree Protocol (CTP). http://www.tinyos.net/tinyos-2.x/doc/html/tep123.html
10. Szczechowiak, P., Oliveira, L., Scott, M., Collier, M.and Dahab, R.: NanoECC: Testing the Limits of Elliptic Curve Cryptography in Sensor Networks. In: European Conference on Wireless Sensor Networks (EWSN'05), Bologna, Italy (2005)
11. TinyOS Alliance: TinyOS. http://www.tinyos.net/