RiSeG: A Logical Ring Based Secure Group Communication Protocol for Wireless Sensor Networks

Omar Cheikhrouhou* , Anis Koubâa‡§, Olfa Gaddour*, Gianluca Dini¶, and Mohamed Abid*

*CES Research Unit, National school of Engineers of Sfax, Sfax, Tunisia.
‡CISTER Research Unit, Polytechnic Institute of Porto (ISEP/IPP), Portugal.
§Al-Imam Mohamed bin Saud University, College of Computer Science and Information Systems, Riyadh, Saudi Arabia.
¶Dipartimento di Ingegneria della Informazione University of Pisa, Via Diotisalvi 2, 56100 PISA, Italy

Emails: omar.cheikhrouhou@isetsf.rnu.tn, aska@isep.ipp.pt, olfa.gaddour@enis.rnu.tn, g.dini@iet.unipi.it, mohamed.abid@enis.rnu.tn

***Abstract*—It is worth noting that securing group communication in Wireless Sensor Networks (WSNs) has recently been extensively investigated. Although many works have addressed this problem, they have considered the concept of grouping differently. In this paper, we consider a group as being a set of nodes sensing the same type of data and we alternatively propose an efficient secure group communication scheme that enables group management and secure group key distribution. The proposed scheme is based on a logical ring architecture, which permits to alleviate the task of the group controller in maintaining the group key. The proposed scheme also provides backward and forward secrecy, addresses the node compromise attack and gives solution to detect and eliminate the compromised nodes. Finally, our implementation shows that the proposed scheme is highly efficient and secure.**

I. Introduction

Wireless Sensor Networks (WSNs) are usually deployed for monitoring several types of data, and therefore, a sensor node is, generally, equipped with many sensors (temperature, humidity, light, ...). In addition, sensor nodes charged with sensing the same data type may want to form a logical group, and consequently, data circulated in one group must not be revealed by nodes alien to that group.

Eventhough several research works dealing with secure group communication exist in the literature, they have different visions to the concept of grouping and have treated that subject from different perspectives. Most of these works, for instance, have considered a group as being a set of nodes physically close to each other. However, in this paper we define a group as a set of nodes that sense the same data type and which are not necessarily close to each other. As a matter of fact, there are several potential applications such as home automation in which several nodes are responsible for controlling diverse parameters e.g. temperature, light, humidity, etc. These sensor nodes are generally spread all over the building and each one utilizes intermediate nodes to communicate with another group member. Another type of application is that of environment monitoring. For example, we can conceive a WSN deployed to sense weather temperature and pollution rate produced by factories. Thus, while the temperature information can be used to deliver a paid service for users, the pollution rate information can be used to control factories and take decision based on the sensed value (e.g. put taxes as a function of the pollution rate). The temperature information should then be delivered exclusively to the subscriber users. In fact, an attacker may try to reveal information (in order not to pay subscription fees and get information for free), but he/she has no interest in injecting false temperature values. As a result, we have to apply confidentiality to the temperature group without having to care about authentication. However, in the case of pollution-related data, information can be sent clearly as it is not a confidential information; yet we must authenticate the sensed value lest the attacker would try to decrease the real value of pollution rate. Therefore, we exclusively need to apply authentication in such a case. As a conclusion, the sensed data do not necessarily have the same critical value, nor do they require the same security service.

Contribution

In this paper, we propose a secure group communication mechanism for wireless sensor networks, whereby a group is defined as being a set of nodes collaborating to collect the same type of sensory information. The proposed mechanism allows protecting data using a group key, which is shared among group members and maintained by the group controller. This key is updated whenever the group membership changes for the sake of providing forward and backward secrecy. One of the main key novelties of this paper is the proposal of a logical ring topology that permits to alleviate the group controller task and render the rekeying process scalable.

The remainder of this paper is organized as follows. In Section II, we describe our network model and assumptions. Then, in Section III, we present our secure group communication mechanism. Then, in Section IV we expose the performance evaluation. Finally, we end up by concluding.

II. Network model, assumptions and requirements

In this section, we present the network model to which the proposed secure group communication is applied as well as the considered assumptions and requirements.

A. Network model

We consider a wireless sensor network maintained by a base station. The information within the network is routed using a routing protocol such as Ad hoc On Demand Distance Vector routing algorithm (AODV)[1] or Dynamic Source Routing (DSR)[2]. In addition to that, we have considered the following types of nodes:

- The Base Station (BS): is responsible for securing the whole network. It maintains a table containing the group controller address corresponding to each group. The BS also controls the group controller activity and maintains a black list containing the identity of compromised nodes. These nodes will not be allowed to join any group in the future and, therefore, are excluded from the network.
- The Group Controller (GC): is a node responsible for maintaining the security of its group. It also stores a table containing the list of group members ordered according to their joining time. The GC controls the group members' activity, and in case of a compromised node, it sends a notification message to the BS. The latter adds the node to the black list.
- The End Device: is a node which belongs to one or multiple groups. For each group, it maintains the next hop address in the logical ring.

B. Assumptions

In our present work, the followings assumptions are made:

- The base station is secure and can detect all compromised GC nodes. Detection of compromised GC nodes can be actually achieved by using an Intrusion Detection System (IDS) such as [3].
- The GC can detect all compromised members: as the BS controls GC nodes, these latter control, in their turn, the node members attached to their groups. GC may use the same IDS tools as the BS.
- Each node is identified by a unique address and can belong to more than one group.
- Each group has a unique identifier, which represents the sensory information corresponding to this group. These group identifiers are known to all nodes. This can be done by loading the group identifiers to nodes at the deployment phase.
- Nodes and BS are synchronized. This can be done using one of the mechanisms described in [4].
- The base station maintains a *black list* containing a list of compromised nodes with their addresses. These nodes are prevented from joining any group and therefore, excluded from the network.
- Each node periodically sends to its corresponding group controller a HELLO message that proves its presence. This permits to detect compromised nodes. Indeed, in a compromise attack, an attacker seizes a node from the sensor network, connects this node to his laptop, extracts the stored data, puts new data/behavior and takes control over that node [5], [6]. This means that a compromise attack necessitates a certain period of time to be executed and therefore, we can assume that a node is compromised if it does not prove its presence, by sending some HELLO messages, during a threshold time period. This assumption is logical, since a node is inactive for a threshold time means either that the node is compromised or that the node has failed. In both cases, the node is evicted from the group, and therefore, must be added to the black list.

C. Requirements

In what follows, we define the requirements that must be achieved by a secure group communication scheme:

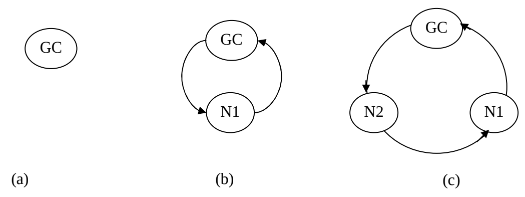
- Nodes belonging to the same group must communicate securely and their exchanged information must not be revealed to non-member nodes even if they belong to the same network.
- A node may belong to more than one group. However, it must store a per-group profile.
- Compromised nodes must be ejected from the group as soon as they are detected.
- Nodes from different groups collaborate to route data. However, data must be confidential to each group (intermediate nodes forward data without being able to reveal their value).
- Backward and forward secrecy must be achieved. Backward secrecy means that a node joining the group at instant t must not reveal information exchanged at instant $t - \Delta t$ (in the past). Forward secrecy means that a node leaving the group at instant t must not reveal information exchanged at instant $t + \Delta t$ (in the future).
- Security parameters maintenance such as the re-keying process must be lightweight and effective.

III. RiSeG: The logical ring based secure group communication scheme

In what follows, we present our proposed secure group communication scheme. It is composed of two parts: (1) the logical ring management and (2) the group membership management.

A. Logical ring management

One of the most important challenges encountered by secure group communication is scalability. In fact, the re-keying process needed in the case of membership change represents a burden task as it requires $O(n)$ messages to be sent by the GC, where n stands for the number of group members. In our work, we have solved this problem by constructing a logical ring topology. This logical ring permits to distribute



(a):the group consist only by GC. (b): joining of node N1. (c): joining of node N2.

Figure 1. Logical ring update in the case of a joining process

the task related to sending information to all members from the GC. Indeed, with this logical ring topology, information is circulated from node to node until it reaches the message source. Therefore, the GC just needs to send a unique message instead of $O(n)$ messages.

The logical ring is constructed in the following way. The ring initially starts with the GC that plays the role of the ring head. Then, each new node is added to the queue (tail) of the ring, upon request to join the group. The information is circulated as follows. Each member sends the message to the next node in the logical ring until it reaches the initial node.

The logical ring topology is maintained by the GC. Note that the GC maintains all the group members' addresses. Each node only maintains its next hop address. In the case of a joining process, the GC informs the newly joined node by its next hop, which is the latest joined node (Figure 1).

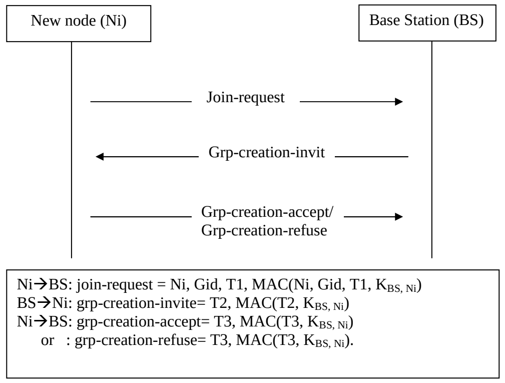
In the case of a leaving process, the leaving node must also be removed from the logical ring. This means that the GC informs the upstream node to change its next hop address to the downstream one (Figure 4).

B. Group membership management

In this section, we describe the necessary operations for maintaining the group membership such as: the group creation, the group join, the group leave, the group controller switching and the group controller leaving. Firstly, we begin by presenting the necessary parameters loaded in nodes at the pre-deployment phase.

1) Pre-deployment phase : As in [7], we propose to apply the key pre-distribution scheme proposed by Blundo et al. [8] in order to share a symmetric key between each pair of nodes. The network administrator chooses a t degree bi-variate polynomial over a finite field F_q:$f(x, y) = \sum_{i=0}^{i=t} \sum_{j=0}^{j=t} a_{i,j} x^i y^j$. The value of q is a prime number that is large enough to accommodate a cryptographic key. Then, the administrator loads in each node Ni the polynomial $f(x, Ni)$. The function f is symmetric. This means that, when two nodes Ni and Nj wish to share a pairwise key, each of them computes $K_{Ni,Nj} = f(Ni, Nj) = f(Nj, Ni)$.

2) Group creation : The group creation process is executed when a node wishes to join a non-existent group. In fact, when a node with identity Ni wishes to join the group identified by Gid, it sends a *join-request* message to the base station. The *join-request* message contains the node identity, the group identifier to which the node wishes to join, a time-stamp,



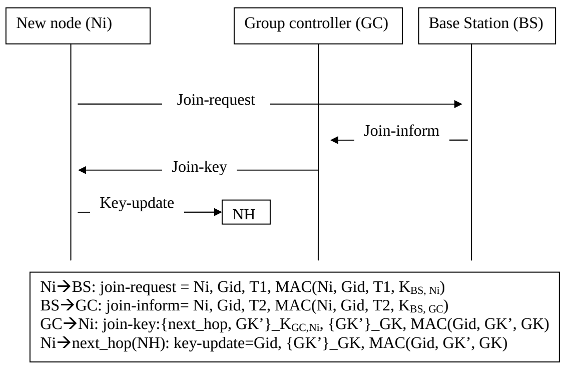
Ni→BS: join-request = Ni, Gid, T1, MAC(Ni, Gid, T1, $K_{BS, Ni}$)
BS→Ni: grp-creation-invite= T2, MAC(T2, $K_{BS, Ni}$)
Ni→BS: grp-creation-accept= T3, MAC(T3, $K_{BS, Ni}$)
or : grp-creation-refuse= T3, MAC(T3, $K_{BS, Ni}$).

Figure 2. Group creation

which represents the current time in the node Ni and a Message Authentication Code (MAC) . The time-stamp allows to avoid replay attacks and the MAC allows to avoid identity usurpation attacks.

Upon receiving this message, the base station verifies the validity of the sender node, the validity of the time-stamp and the validity of the MAC. The validity of the sender node means that the node does not belong to the blacklist and, therefore, the node is considered as not compromised. The validity of the time-stamp means that the received time-stamp is in the expected delay. The received MAC is also compared to the locally computed one, and it is considered valid if they are both equal. If any of these parameters fails the validity test, the base station will ignore the request. Otherwise, the base station replies to the node Ni by sending a *grp-creation-invite* message. This message contains a time-stamp and is also protected by a MAC. Therefore, the node is invited to be the GC of the newly formed group. If the node accepts to be a GC it replies by sending a *grp-creation-accept* message. Otherwise, it replies by sending a *grp-creation-refuse* message. Figure 2 summarizes the group creation process.

3) Group join : The group join process is executed when a node wishes to join an existent group. On receiving a *join-request* message to a group that already exists, the base station verifies the validity of the sender node. If the node is proved to be legal and its *join-request* message passes the freshness test (the time-stamp is in the expected delay) as well as the authenticity test (the MAC is valid), the base station then sends a *join-inform* message to the GC, informing the GC that a new node has joined its group. The *join-inform* message contains the identity of the node Ni and is protected by a time-stamp to avoid replay attacks and by a MAC to avoid usurpation of base station identity attack. After testing the validity of the message, the GC computes a new group key GK' and sends out to Ni this new key along with the address of the next hop of Ni and a *key-update* message that will be forwarded to all other members. The address of Ni's next hop is useful in maintaining the logical ring topology. The *key-update* message



Ni→BS: join-request = Ni, Gid, T1, MAC(Ni, Gid, T1, $K_{BS, Ni}$)
BS→GC: join-inform= Ni, Gid, T2, MAC(Ni, Gid, T2, $K_{BS, GC}$)
GC→Ni: join-key:{next_hop, GK'}_$K_{GC,Ni}$, {GK'}_GK, MAC(Gid, GK', GK)
Ni→next_hop(NH): key-update=Gid, {GK'}_GK, MAC(Gid, GK', GK)

Figure 3. Group join

is composed of the new key encrypted by the old one and a MAC that guarantees that the key is computed by the GC. The new key will be sent from node to node (members of the group) until reaching the originator (the GC) using the logical ring topology concept. Figure 3 summarizes the group join process.

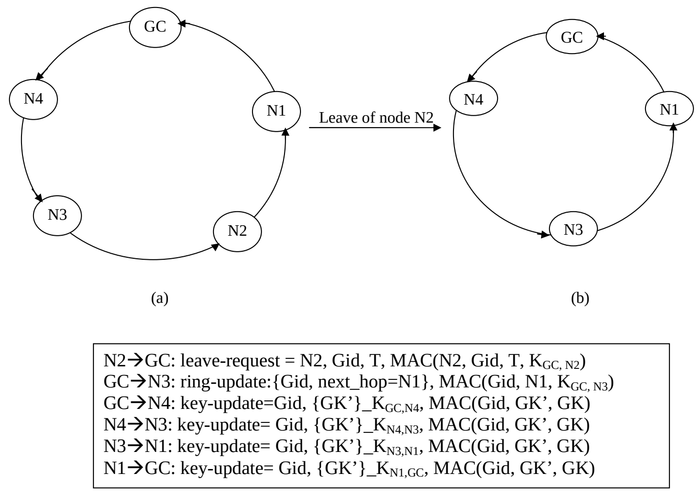
4) Group leave : The leaving process occurs when a node wishes to leave the group, breaks down or is compromised. In the first case, the GC is informed through a *leave-request* message. In the two latter cases the GC is informed through the inactivity of the leaving node. In order to achieve forward secrecy, the GC must compute a new group key and sends it to each member. The leaving process is more complicated than the joining process as we cannot send the group key encrypted by the old one. This is due to the fact that the leaving node already knows the old keys and this breaks the requirement of forward secrecy. As the key must be sent in a secure way, the only solution is to send it by unicast and encrypted by pairwise keys. As the unicast of the new key to each member is a burden task, it is wise to divide this task among all group members. This can be achieved by using the logical ring topology concept.

To illustrate the leaving process, let us consider, for instance, the leave of node $N2$ in Figure 4. When receiving a *leave-request* message, the GC checks the validity of the message. If the message is valid, the GC computes a new group key and then sends it to the following node in the logical ring. This key is protected by a MAC and is sent encrypted by the pairwise key.

IV. IMPLEMENTATION

We have implemented an early prototype of the RiSeG protocol in TinyOS operating system [9] using the nesC [10] language. For encryption we used the AES algorithm [11] with key size of 128 bits (16 bytes). For MAC computing we used MMH interface, which is provided in *tinyos-contrib/crypto* modules [12]. This interface is an implementation of the Multilinear-Modular-Hashing function [13], which provides a 32 bits MAC.

In what follows we present the performance evaluation results.



N2→GC: leave-request = N2, Gid, T, MAC(N2, Gid, T, $K_{GC, N2}$)
GC→N3: ring-update:{Gid, next_hop=N1}, MAC(Gid, N1, $K_{GC, N3}$)
GC→N4: key-update=Gid, {GK'}_$K_{GC,N4}$, MAC(Gid, GK', GK)
N4→N3: key-update= Gid, {GK'}_$K_{N4,N3}$, MAC(Gid, GK', GK)
N3→N1: key-update= Gid, {GK'}_$K_{N3,N1}$, MAC(Gid, GK', GK)
N1→GC: key-update= Gid, {GK'}_$K_{N1,GC}$, MAC(Gid, GK', GK)

Figure 4. Message exchange in a group leave process

A. Energy consumption simulation

We have simulated RiSeG in TOSSIM simulator [14]. The simulation was made using 30 nodes with a fully-meshed topology (we assume that each node has a direct connection with each other node in the network). The reason behind this assumption is to avoid using a routing protocol, which may introduce some overhead not related to security issues. The transmission power was fixed to $-50dBm$.

For energy consumption simulation we used PowerTOSSIM-Z tool [15], which simulates energy consumption using the MICAz [16] energy model. In this model, the most frequently used and energy-intensive peripheral is the radio component. In addition, in this energy model the reception mode consumes more energy than the transmission mode. The mote battery initial energy was set to 21600 J.

- Group creation: the energy consumption introduced by the group creation process is shown in Table I. These values are small as they represent about 10^{-6} of the battery initial energy indicating that the group creation procedure does not consume too much energy.

Table I
GROUP CREATION ENERGY CONSUMPTION (MJ)

Group creation	
BS	GC
233	174

- Group join: the group join process involves three entities: the base station, the group controller and the end device. The energy consumption introduced by the group join process at each entity in mJ is presented in Table II. We note that the base station and the group controller both consume only 58 mJ in a join process

Figure 5 shows the energy consumed after the join of n nodes. When a node joins a group, the BS executes a group

Table II
GROUP JOIN ENERGY CONSUMPTION (MJ)

Group join		
BS	GC	ED
58	58	

join process, the GC executes a group join and key update processes and the ED executes a key update process. We note that the energy consumption is linear with the number of joined nodes.

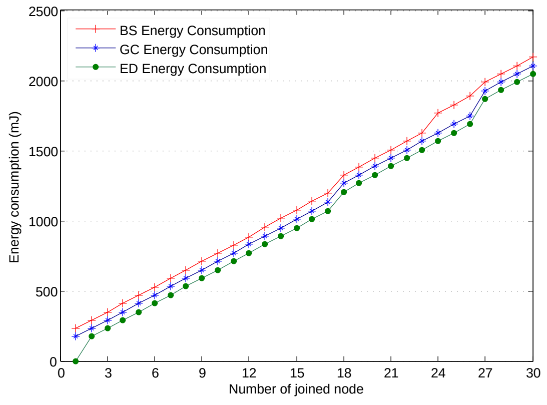


Figure 5. Energy consumption per number of joined nodes

RiSeG was also tested on real world plateform using Telosb motes [17]. Telosb mote has a 8 MHz microcontroller, 10 Kbytes of RAM memory, and 48 Kbytes of ROM memory. In what follows, we present the experimental results.

B. Memory consumption

The proposed secure group communication scheme does not require much memory overhead. In fact, due to Blundo et al.'s key distribution technique, each sensor node has to store a polynomial function which occupies $(t+1)logq$ storage space, where t stands for the degree of the polynomial and q represents the size of keys [18]. In addition, the node has to store the group key, the address of the next hop, as well as the GC address for each group it belongs to. The GC also stores the members' addresses that belong to its group. As for the base station, it has to store the GC address corresponding to each group as well as the black list containing the list of compromised nodes.

For the base station, the compiled RiSeG code consumes 24390 bytes in ROM (i.e. code size) and 5744 bytes in RAM (i.e. data). These values represent respectively 50% of ROM and 57% of RAM.

For the end device, the compiled RiSeG code consumes 35694 bytes in ROM and 6448 bytes in RAM. Note that the code supports also the code of the group controller. These values represent respectively 72% of ROM and 64% of RAM.

C. Execution Time

We have measured the execution time of the principal components of RiSeG. This is done using the $LocalTime < TMilli >$ interface provided by TinyOS. We used also the $printf$ library in order to print performance parameters through the serial port of the laptop.

Table III
EXECUTION TIME ON TELOSB MOTES

	Time (ms)
Group creation	180
Group join	700
Key update per node	~=400

The proposed secure group communication scheme is lightweight in terms of computation. In fact, the main operations of the scheme are the MAC operation and the encryption operation. The number of required operations regarding each group membership process are as follows. In the group creation process, three MAC operations are computed by both the joined node and the base station. In the group join process, the joined node needs to compute one MAC and one encryption operation, the base station computes two MAC operations, and the group controller two MAC and one encryption operations. As for the group leave process, the leaving node computes one MAC operation and the group controller three MAC and one encryption operations.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a secure group communication scheme for wireless sensor networks. We considered a group as being a set of nodes that cooperate to sense the same type of information. The proposed scheme is lightweight and effective thanks to the use of a logical ring topology. In addition, the scheme permits to fight against node compromise attacks and provides both forward and backward secrecies. The implementation of the propoed scheme shows its feasibility and that the scheme is lightweight in terms of execution time, energy consumption and memory consumption.

As a future work, we are working towards the improvement of the key management process by reducing the number of messages required to update the group key using the ring paradigm. In addition, we expect to integrate our group security approach in standard protocols such as ZigBee and 6LowPAN.

REFERENCES

[1] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1997, pp. 90–100.
[2] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Kluwer Academic Publishers, 1996, pp. 153–181.
[3] A. H. Farooqi and F. A. Khan, "Intrusion detection systems for wireless sensor networks: A survey," in *Communication and Networking*, pp. 234–241, 10.1007/978-3-642-10844-0-29.
[4] D. Fontanelli and D. Petri, "An algorithm for wsn clock synchronization: Uncertainty and convergence rate trade off," *IEEE International Workshop on Advanced Methods for Uncertainty Estimation in Measurement, 2009. AMUEM 2009.*, pp. 74 –79, july 2009.

[5] J. Abraham and K. S. Ramanatha, “Energy efficient key management protocols to securely confirm intrusion detection in wireless sensor networks,” in *Wireless Sensor and Actor Networks II*, pp. 149–160, 10.1007/978-0-387-09441-0-13.

[6] H. Song, L. Xie, S. Zhu, and G. Cao, “Sensor node compromise detection: the location perspective,” in *IWCMC ’07: Proceedings of the 2007 international conference on Wireless communications and mobile computing*. New York, NY, USA: ACM, 2007, pp. 242–247.

[7] O. Gaddour, A. Koubaa, and M. Abid, “Segcom: A secure group communication mechanism in cluster-tree wireless sensor networks,” in *First International Conference on Communications and Networking, 2009. ComNet 2009.*, Nov. 2009, pp. 1–7.

[8] C. Blundo, A. De Santis, U. Vaccaro, A. Herzberg, S. Kutten, and M. Yong, “Perfectly secure key distribution for dynamic conferences,” *Inf. Comput.*, vol. 146, no. 1, pp. 1–23, 1998.

[9] TinyOS: http://www.tinyos.net/, 2010.

[10] NesC: A Programming Language for Deeply Networked Systems. http://nescc.sourceforge.net/, 2010.

[11] “National institute of standards and technology (nist), advanced encryption standard (aes).” Federal Information Processing Standards Publications (FIPS PUBS) 197, 2001.

[12] tinyos-2.x-contrib: http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/contrib.html.

[13] S. Halevi and H. Krawczyk, “Mmh: Software message authentication in the gbit/second rates,” in *FSE ’97: Proceedings of the 4th International Workshop on Fast Software Encryption*. London, UK: Springer-Verlag, 1997, pp. 172–189.

[14] TOSSIM: http://www.cs.berkeley.edu/ pal/research/tossim.html.

[15] PowerTOSSIM-Z: http://www.scss.tcd.ie/~carbajor/powertossimz/index.html.

[16] MICAz datasheet: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf, 2010.

[17] Telosb datasheet: www.ece.osu.edu/ bibyk/ee582/telosMote.pdf.

[18] D. Liu, P. Ning, and R. Li, “Establishing pairwise keys in distributed sensor networks,” *ACM Transactions on Information and System Security (TISSEC)*, pp. 41 – 77, 2005.