Perfectly Anonymous Data Aggregation in Wireless Sensor Networks

Levente Buttyán and Tamás Holczer
Laboratory of Cryptography and Systems Security (CrySyS)
Budapest University of Technology and Economics
Email: {buttyan, holczer}@crysys.hu

Abstract

Clustering and data aggregation in wireless sensor networks improves scalability, and helps the efficient use of scarce resources. Yet, these mechanisms also introduce some security issues; in particular, aggregator nodes become attractive targets of physical destruction and jamming attacks. In order to mitigate this problem, we propose a new private aggregator node election protocol that hides the identity of the elected aggregator nodes both from external eavesdroppers and from compromised nodes participating in the protocol. We also propose a private data aggregation protocol and a corresponding private query protocol which allows the aggregators to collect sensor readings and respond to queries of the base station, respectively, without revealing any useful information about their identity to external eavesdroppers and to compromised nodes.

1. Introduction

The nodes in wireless sensor networks are often assumed to be organized into clusters. Clustering introduces a hierarchy in the network and it helps to improve its scalability properties. A typical scenario is that sensor readings are first collected in each cluster by a designated node that aggregates them and sends only the result of the aggregation to the base station. In another scenario, the base station may not be present permanently in the network, and the aggregated data must be stored by the designated node in each cluster temporarily until the base station can eventually fetch the data. In both scenarios, the combination of clustering and data aggregation improves the efficiency of resource utilization (energy and memory, respectively).

While clustering and data aggregation in wireless sensor networks are clearly advantageous with respect to scalability and efficiency, they introduce a security issue: the designated nodes that store aggregated sensor readings and communicate with the base station become attractive targets of physical destruction and jamming attacks. Indeed, it is a good strategy for an attacker to locate these designated nodes and disable them, because he can prevent the reception of data from the entire cluster by just attacking a single node. Even if the aggregator role is changed regularly by some election process, some security issues remain; in particular, the attacker can locate and attack the node that was aggregator in a specific time epoch.

In order to mitigate this problem, we introduced the concept of private aggregator node election, and we proposed the first private aggregator node election protocol in our earlier work [1]. Briefly, our earlier protocol ensures that the identity of the elected aggregator remains hidden from an attacker who observes the execution of the election process. However, our earlier protocol ensures only protection against an external eavesdropper that cannot compromise sensor nodes, and it does not address the problem of identifying the aggregators by means of traffic pattern analysis in the aggregation phase.

In this paper, we address all the shortcomings of our earlier scheme: We propose a new private aggregator node election protocol that is resistant even to internal attacks originating from compromised nodes, and we also propose a new private data aggregation protocol and a new private query protocol which preserves the anonymity of the aggregator nodes during the data aggregation process and when they provide responses to queries of the base station. In our new private aggregator node election protocol, each node decides locally in a probabilistic manner to become an aggregator or not, and then the nodes execute an anonymous veto protocol to verify if at least one node became aggregator. The anonymous veto protocol ensures that non-aggregator nodes learn only that there exists at least one aggregator in the cluster, but they do not learn any information on its identity. Hence, even if such a non-aggregator node is compromised, the attacker learns no useful information regarding the identity of the aggregator.

Our new private data aggregation protocol is based on a special broadcast communication scheme, where the nodes of the cluster organize themselves into a ring and each data packet to be included in the aggregated result is sent around that ring. Note that a broadcast communication scheme is necessary, because in our scheme the nodes do not know the identity of the aggregators, therefore, they can only send data to the aggregators by broadcasting. We chose the ring based broadcast scheme, because our private query protocol exploits its properties. Our new private query protocol allows the aggregator nodes to respond to the queries of the base station without leaking any information about their identity. For this, a query token is passed around the ring, and each non-aggregator node adds some noise to the token, while the aggregator adds noise and the aggregated result. When the token is returned to the base station, it extracts the aggregated result by removing the noise.

The remainder of the paper is organized as follows: In Section 2, we introduce our system and attacker models. In Section 3, we present our protocol that consist of some initialization, a private aggregator node election sub-protocol,

a private data aggregation sub-protocol, and private query sub-protocol. In Section 4, we give an overview of some related work, and in Section 5, we conclude the paper and sketch some future research directions.

2. System and attacker models

A sensor network consists of sensor nodes that can communicate with each other via wireless channels. Every node can collect some environmental information, and store it or forward it to another node. Each node can directly communicate with the nodes within its radio range. In order to communicate with distant nodes (outside the radio range), the nodes use multi-hop communication. The sensor network has an operator as well, who can communicate with some of the nodes through a gateway node, or can communicate directly with the nodes if the operator moves close to the network.

Throughout the paper, a data driven sensor network is envisioned, where every sensor node sends its measurement to the data aggregator regularly.

It is assumed that every node i shares a secret k_i key with the operator. These keys are unique for every node, and the operator can store them in a lookup table, or can be generated from a master key and the node's identifier on demand.

The whole WSN can be partitioned into smaller parts, called clusters. In the following, it is assumed, that the partitioning of the network is done, and every node is aware of the cluster it belongs to. The leader of each cluster is called cluster aggregator, or simply aggregator. The protocols defined in Section 3 are defined for only one cluster, and independent protocol instances are run independently in each cluster.

As mentioned in Section 1 an attacker can gain much more information by attacking an aggregator node, then attacking a normal node. To attack a data aggregator node either physically or logically, first the attacker must identify that node. In the following of the paper the attackers goal is to identify the cluster aggregator (which means that simply preventing, jamming or confusing the aggregation is not goal of the attacker).

An attacker who wants to reveal the identity of the aggregators can eavesdrop the communication between any nodes, can actively participate in the communication (by deleting modifying and inserting messages) and can physically compromise some of the nodes. A compromised node is under the full control of the attacker, the attacker can fully review the inner state of that node, and can control the messages sent by that node.

The goal and limitations of the proposed protocol are the followings:

- The identity of the cluster aggregator remains secret even in the presence of passive and active attackers or compromised nodes.
- An attacker can force a compromised node to be aggregator, but does not know anything about the existence or identity of the other aggregators.
- The attacker cannot achieve that no aggregator is elected in the cluster, however with small probability, all the elected aggregator(s) can be compromised.

In the following, we will assume, that the time is slotted, and one measurement is sent to the data aggregator in each time slot.

3. Protocol

The private cluster based data aggregation protocol consists of four main parts. The first part is the initialization, which provides the required communication channel. The second part is needed for the data aggregator election. This subprotocol must ensure that the cluster does not remain without a cluster aggregator. This must be done without revealing the identity of the elected aggregator. The third part is needed for the data aggregation. This subprotocol must be able to forward the measured data to the aggregator without knowing the identifier of that node. The last part must support the queries, where a possibly mobile operator queries some stored data.

3.1. Initialization

The initialization phase is responsible for providing the media for authenticated broadcast communication. In the following, we shortly review the approaches of broadcast authentication in wireless sensor networks, and give some efficient methods for broadcast communication.

The initialization relies on some data stored on each node before deployment. Each node has some unique cryptographic credentials to enable the authentication, and aware of the cluster identifier it belongs to. In the following, without any further notion, we will assume, that each message contains the cluster identifier. Every message addressed to a different cluster a node belongs to is discarded by the node. First we briefly review the state of the art in broadcast authentication, then we propose a ring based broadcast communication method, which fits well to the following aggregation and query phases.

3.1.1. Broadcast authentication. Broadcast authentication enables a sender to broadcast some authenticated messages efficiently to a big number of potential receivers. In the literature, this problem is solved with either digital signatures or hash chains. In this section, we introduce some solutions from both approaches.

Digital signatures are asymmetric cryptographic primitives, where only the owner of a private key can compute a digital signature over a message, but any other node can verify that signature. Computing a digital signature is a time consuming task for a typical sensor node, but there exist some efficient elliptic curve based approaches in the literature. Up to now, to the best of our knowledge, the fastest implementations are the TinyPBC [2] proposed by Oliveira *et al.*, and the TinyPairing proposed by Xiong *et al.* in [3].

Another approach is proposed for broadcast authentication in wireless sensor networks by Perrig *et al.* in [4]. The μTESLA scheme is based on delayed reveal of hash chain values used in MAC computations. The scheme needs secure loose time synchronization between the nodes, which in practice is a digital signature based solution itself. The μTESLA scheme is efficient if it is used for authenticating many messages, but inefficient if the messages are sparse.

In the following we will assume, that an efficient broadcast authentication scheme is used without any indication.

3.1.2. Broadcast communication. Broadcast communication is a method that enables sending information from one source to every other participant of the network. In wireless networks it can be implemented in many ways, like flooding the network or with a sequence of unicast messages.

In this section we focus on the later, more precisely generating a Hamilton cycle for the network. A Hamilton cycle in graph theory is a permutation of all possible vertices, where all consecutive vertices are connected. A Hamilton cycle can be envisioned in WSNs as a ring of the nodes. This topology is assumed by the later protocol parts.

Finding a Hamilton cycle in a graph is a NP-complete problem, but there are many efficient heuristics which can find a Hamilton cycle in a graph with high probability. In the following we review one complex solution, which provides a Hamilton cycle with high probability, and another solution, which provides a ring-like overlay for the network.

The problem of finding Hamiltonian cycles in faulty random geometric networks is analyzed in [5]. The faulty random geometric graph model is very close to WSNs as a random geometric graph is a graph whose vertices correspond to points uniformly and independently distributed in the unit square, and whose edges connect any pair of vertices if their distance is below some specified bound. A faulty random geometric network is a random geometric network whose vertices or edges fail at random. In [5] a centralized algorithms is presented to solve the problem. The main idea of the algorithm, is to partition the network into small boxes. In each box, it is easy to find a Hamilton cycle, and the small cycles can be connected to a system wide Hamilton cycle with high probability.

The distributed version of the previous algorithm is presented in [6]. That algorithm finds a Hamiltonian cycle almost sure, but assumes that each node knows its position. The position information can be gathered from a GPS receiver or can be stored at installation time.

The construction of a ring-like overlay on WSNs is discussed in [7]. A ring-like overlay is an overlay path where every node is visited at least once. The advantage of this solution is that the generation of a ring-like topology is far easier than a Hamilton cycle. The disadvantage is the possibly longer ring (some nodes are possibly visited more than once), which results in higher communication load.

Note here that our solution is not restricted to the previous solutions, we only require that each node knows its next hop on a ring, and this next hop is reachable from the node.

3.2. Data aggregator election

The main goal of the aggregator node election protocol is to elect a node, that can store the measurements of the whole cluster, but hide the identity of that node. The election is successful if at least one node is elected. The protocol is unsuccessful if no node is elected, thus no one stores the data. In some cases, electing more than one node can be advantageous, because the multiplied storage can withstand the failure of some nodes. In the following, we propose an election protocol, where the expected number of aggregators can be determined by the system operator, and the protocol ensures that at least one aggregator is always elected.

The election process consists of two main steps: (i) Every node decides, whether it wants to be an aggregator, based on some random values. This step does not need any communication, the nodes compute the results themselves. (ii) In the second step, an anonymous veto protocol is run, which reveals only the information, that at least one node elected itself to be aggregator node. If no aggregator is elected, it will be clear for every participant, and every participant can run the protocol once again.

Step (i) can be implemented easily. Every node elects itself as a aggregator with a given probability p. Let us denote the random variable representing the number of elected aggregators with C. Obviously the distribution of C is binomial (N is the total number of nodes in one cluster):

$$\Pr(C = c) = \left(\begin{array}{c} N \\ c \end{array} \right) p^c \, (1-p)^{N-c}$$

The expected number of aggregators after the first step is: $c_E = Np$.

To avoid the anarchical situation that no node is elected, the nodes must run step (ii) which proves that at least one node is elected as aggregator node, but the identity of the aggregator remains secret. This problem can be solved by an anonymous veto protocol. Such a protocol is suggested by Hao and Zieliński in [8].

Unfortunately the description of the protocol must be left out because of space limitations. Here we only summarize the properties of the protocol: The protocol consist of two consecutive communication rounds, which can be realized with an authenticated broadcast channel described in Section 3.1. After the second round, every participant can compute a product, which equals to 1 only if no aggregator is elected. The anonymity of the protocol is based on the decisional Diffie-Hellmann assumption.

If we consider the effect of the second step (new election is run if no aggregator is elected), the expected number of aggregators is slightly higher then in binomial distributions. The expected number of aggregators are:

$$c_E = \frac{Np}{1-(1-p)^N}$$

The anonymity of the election subprotocol depends on the parts of the protocol. Obviously the random number generation does not leak any information about the identity of the

aggregator nodes, if the random number generator is secure. A cryptographically secure random number generator, called TinyRNG, is proposed in [9] for wireless sensor networks. Using a secure random number generator, it is unpredictable, who elects itself to be aggregator node.

The anonymity analysis of the anonym veto protocol can be found in [8]. The anonymity is based on the decisional Diffie-Hellman assumption, which is considered as a hard problem.

The message complexity of the election is $O(N^2)$, which is acceptable as the election is run infrequently (N is the number of nodes in the cluster).

As a summary, after the election subprotocol every node is equiprobably aggregator node, an outsider attacker does not know the identity of the aggregators or even the actual number of the elected nodes. An attacker, who compromised one or more nodes, can decide whether the compromised nodes are aggregators, but cannot be certain about the other nodes.

The election subprotocol ensures for the following steps, that at least one aggregator is elected and this node(s) is aware of its status.

3.3. Data aggregation

The main goal of the WSN is to measure some data from the environment, and store the data for later use. This section describes how the data is forwarded to the aggregator(s) without the explicit knowledge of the identifier(s) of the aggregator(s).

The aggregation directly uses the ring topology created by the initialization phase, and can use broadcast authentication if required.

In each timeslot, one randomly chosen node starts the aggregation. If more than one node starts the aggregation simultaneously, then the initiator with the highest ID will be the actual initiator, and the other initiator's packets can be discarded (like in the IEEE 802.5 token ring protocol).

The choice of the initiator can be based on random timers. The first node whose timer expires starts the aggregation by sending the measured data as a token to its right neighbor on the ring. If a node receives an aggregation token, it aggregates its measurement to the token, and passes it to its right neighbor. When the initiator of the token gets its token back, it sends it around again, so the aggregators can store the final aggregated data of all nodes in the second round.

The aggregation function can be simply inserting the measurement to the list of previous measurements, which means that the token's size increases with every hop. If some statistics are enough, then the token's size can remain the same. Some easily implementable and widely used statistic are the minimum, maximum, sum or average. On Figure 1, the aggregation protocol is visualized with five nodes and two aggregators using the average as an aggregation function.

At the second round, every aggregator can store the measurements in its internal memory. The stored data includes the timeslot in which the measurement was recorded, and the environmental variables if more than one variable (e.g. temperature and humidity) is recorded.

The anonymity analysis of the aggregation subprotocol is quite simple. The initiator of the aggregation is independent of the identity of the aggregators. After the aggregation, every node poses the same information as an external attacker can get. This information is the measured data itself, without knowing anything about the identity of the aggregators. If the operator wants to hide the measured data, it can use some techniques discussed in Section 4.

The message complexity of the aggregation is $O(N)$, where N is the number of nodes in the cluster. This is the best complexity achievable, because to store all the measurements by a single aggregator, all nodes must send the measurements towards the aggregator, which leads to $O(N)$ complexity.

3.4. Query

The long term goal of the sensor network is to forward the measured data to the operator on request. This mode is called query driven operation. The aggregation subprotocol ensures, that the measured data is stored by the aggregators. The goal of the query subprotocol is the following: provide the requested data to the operator, while the aggregators remain anonym.

One solution would be that the operator visits all the nodes, and connects to them by wire. However, this solution would leak no information about the identities of the aggregators to any eavesdropping attacker, the execution is very time consuming. Hence, a solution is proposed, where it is enough for the operator to get in wireless communication range of any of the nodes.

First, the operator authenticates itself to node A (agent) using the k_A key. After the authentication, node A starts the query protocol. First, the node sends the Q query data and an R_1 special pseudo random number to its right neighbor. Q describes what information the operator is interested in: At what time, which environmental variable (e.g. Q = "01. 01. 1970. 00:01, temperature"). Every node adds an R_i pseudo random number to the sum of the previous random numbers, and forwards the query (Q, R) to the next neighbor. Node A receives $R = \sum R_i$ at the end, and sends it to the operator. From R, the operator can calculate the stored data, if R_i is defined as follows.

R_i is the hash value of the Q query data and the k_i key of the node for non aggregators. For aggregators, the same hash value is sent plus the measurement in which the operator is interested in:

$$R_i = \begin{cases} h(Q|k_i), \text{ for non aggregators} \\ h(Q|k_i) + M, \text{ for aggregators} \end{cases}$$

The operator can simply regenerate the hash values, because it stores the same k_i keys and it sent the Q query data. The operator can subtract the hash values from R getting cM, where c is the actual number of aggregators,

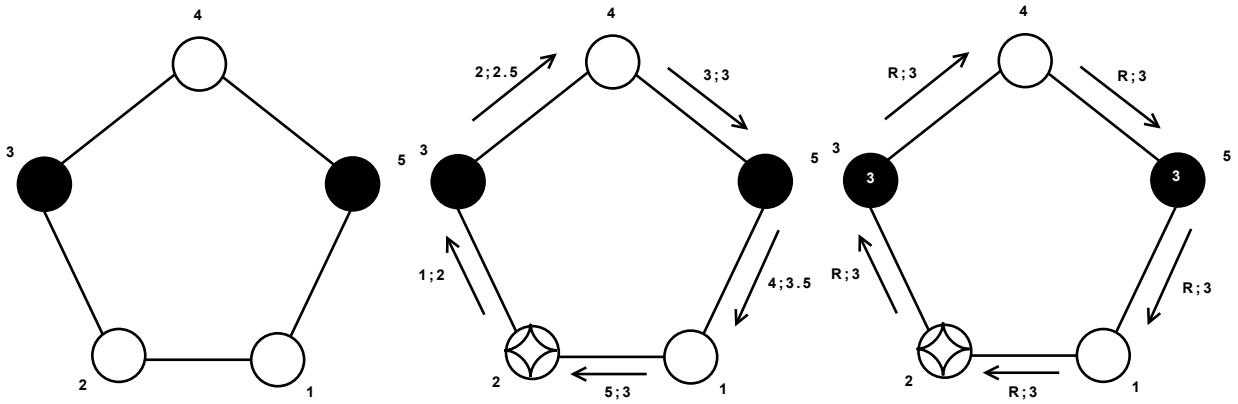

Figure 1. Aggregation protocol. The subfigures from left to right represents the three part of the aggregation: Ready to aggregate, calculate the aggregated value, store the final value. The black nodes are the aggregators, the node with the sign is the initiator. The aggregation function is the average, the measured values are shown next to the nodes (1-5). The passed values are pairs of visited nodes (or R for ready to store) and actual averages.

and M is the stored data. c is unknown to the operator, as no one knows the identity or even the number of the aggregators. The task of the operator is the following: Find a suitable $\hat{c}$ number with what it can divide cM to get M. If M is selected from a special interval, only one $\hat{c}$ exists which after the division leads to the specific interval. In this case $c = \hat{c}$. In the following, we define such a specific interval, where always only one $\hat{c}$ exists.

The $[A, B]$ integer interval should be defined to meet three criteria: (i) only one valid $\hat{c}$ should exist independently from M and c, this ensures that $c = \hat{c}$, (ii) even the highest value (BN, N is the number of nodes in the cluster, B is the maximal value in the interval) must be representable, and (iii) at least D different values must be distinguishable.

The first (i) criteria is met, if every possible cM values belong to only one of the N possible intervals $[iA, iB]$, $i = 1, \ldots, N$ ($i = 1$ represents the case of only 1 aggregator, while $i = N$ represents the unlikely case that every node in the cluster is an aggregator). With other words, the N possible intervals must be not overlapping. This can be formulated as the lower end of an interval is bigger than the higher end of the previous interval:

$$0 < iA - (i-1)B = i(A-B) + B, \ i = 1, \ldots, N$$

, if this holds for $i = N$, then it holds for every i, because $A - B$ is negative constant and B is a positive constant. So it is enough to consider only the $i = N$ case:

$$\begin{array}{c} 0 < N(A-B) + B \\ B < \frac{N}{N-1} A \end{array} \quad (1)$$

The second (ii) criteria can be formalized as follows (L is the highest value representable on the given data type, e.g. 2^{16} or 2^{32}):

$$\begin{array}{c} BN < L \\ B < \frac{L}{N} \end{array} \quad (2)$$

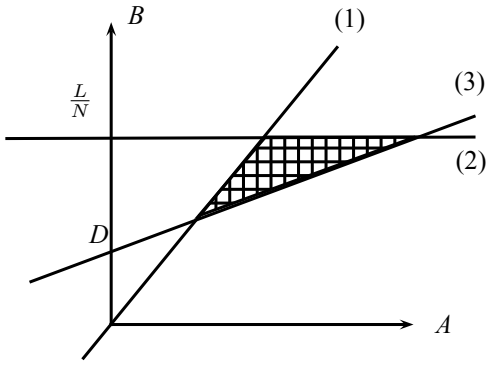

Figure 2. Graphical representation of the suitable intervals

The third (iii) criteria can be formalized as follows (D different values must be represented):

$$\begin{array}{c} D < B - A \\ B > A + D \end{array} \quad (3)$$

A typical case of the three criteria is represented on Figure 2. The crossed area represents the possible (A, B) pairs.

A solution exists only if the B coordinate of intersection of inequality (1) and (3) meets criteria (2):

$$NM < \frac{L}{N}$$

Any measurement interval can be transformed to the (A, B) interval by a simple shifting. For example a positive x value whose maximum is C ($x \in (0, C)$) can be shifted to the (A, B) interval by adding A to the measurement: $x' = x + A$.

The protocol has many advantageous properties. The message length is fixed and independent from the number

of nodes or aggregators. The operator does not need to know the identity of the aggregators, thus cannot leak that knowledge accidentally.

The protocol does not leak any information about the identity of the aggregators. An attacker can eavesdrop the Q query information, and the R_i pseudo random numbers, but cannot deduce the identity of the aggregators.

The message complexity of the query is $O(N)$, where N is the number of nodes in the cluster. This is the best complexity achievable, because the operator does not know the identity of the aggregator(s).

4. Related work

Due to space constraints, we can only give a short overview of the privacy protection techniques of WSNs. In [10] the techniques can be classified into two main groups: data-oriented and context oriented protection.

In data-oriented protection, the confidentiality of the measured data must be preserved. It is also a research direction, how can the operator verify if the received data is correct. The main focus is on the confidentiality in [11], while the verification of the received data is also ensured in [12].

According to [10] context oriented protection covers the location privacy of the source and the sink. The source location privacy is mainly a problem in event driven networks, where the existence and location of the event is the information, which must be hidden. The location privacy of the sink is discussed in [13]. The main difference between hiding the sink and the in network aggregators is that a WSN contains only one sink which is a predefined node, while at the same time there are more in network aggregators used in one network, and the nodes used as aggregators are periodically changed.

The most related work is [1], where a private cluster head election protocol is suggested. That solution solves the election problem, but sensitive to node compromisation, and does not solve either the aggregation or the query problem.

5. Conclusion

In this paper, we proposed a new private aggregator node election protocol for wireless sensor networks that hides the identity of the elected aggregator nodes. We also proposed a private data aggregation protocol and a corresponding private query protocol which allow the aggregators to collect sensor readings and respond to queries of the operator, respectively, without revealing any useful information about their identity. Our protocols are resistant to both external eavesdroppers and compromised nodes participating in the protocol.

Our current and future work is concerned with the replacement of the ring based broadcast communication scheme with spanning trees. Trees would provide a better solution, because their existence is guaranteed in connected graphs, unlike the existence of Hamiltonian cycles. Trees can also be constructed much more efficiently. However, switching to trees require some modifications in our query protocol. We are also planning to develop a prototype implementation of our protocols and to analyze their performance.

Acknowledgement. The work described in this paper is based on results of the WSAN4CIP project (`http://www.wsan4cip.eu`), which receives research funding from the European Community's 7th Framework Programme. Apart from this, the European Commission has no responsibility for the content of this paper. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

References

[1] L. Buttyán and T. Holczer, "Private cluster head election inwireless sensor networks," in *Proceedings of the Fifth IEEE International Workshop on Wireless and Sensor Networks Security (WSNS'09)*, IEEE. IEEE, 2009.

[2] L. B. Oliveira, M. Scott, J. Lopez, , and R. Dahab, "TinyPBC: Pairings for Authenticated Identity-Based Non-Interactive Key Distribution in Sensor Networks," in *Proceedings of the 5th International Conference on Networked Sensing Systems (INSS'08)*, IEEE. Kanazawa/Japan: IEEE, June 2008.

[3] X. Xiong, D. S. Wong, and X. Deng, "TinyPairing: A Fast and Lightweight Pairing-based Cryptographic Library for Wireless Sensor Networks," in *Proceedings of the IEEE Wireless Communications & Networking Conference*. IEEE, 2010.

[4] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The TESLA Broadcast Authentication Protocol," *RSA CryptoBytes*, vol. 5, no. Summer, 2002.

[5] J. Petit, "Hamiltonian cycles in faulty random geometric networks," in *Proceedings of the 2nd International Workshop on Approximation and Randomization Algorithms in Communication Networks (ARACNE 2001)*, vol. 10.

[6] E. Levy, "Distributed algorithms for finding hamilton cycles in faulty random geometric graphs," Master's thesis, Université Libre de Bruxelles, 2002.

[7] A. Banerjee and C.-T. King, "Building Ring-Like Overlays on Wireless Ad Hoc and Sensor Networks," *IEEE Transactions on Parallel and Distributed Systemsy*, vol. 20, no. 11, 2009.

[8] F. Hao and P. Zielinski, "A 2-round anonymous veto protocol," in *Proceedings of the 14th International Workshop on Security Protocols, Cambridge, UK*, 2006.

[9] A. Francillon and C. Castelluccia, "TinyRNG: A cryptographic random number generator for wireless sensors network nodes," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, April 2007.

[10] N. Li, N. Zhang, S. Das, and B. Thuraisingham, "Privacy preservation in wireless sensor networks: A state-of-the-art survey," *Ad Hoc Networks*, 2009.

[11] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: Privacy-preserving data aggregation in wireless sensor networks," in *Proceedings of Infocom*. IEEE, 2007.

[12] B. Sheng and Q. Li, "Verifiable privacy-preserving range query in two-tiered sensor networks," in *Proceedings of Infocom*. IEEE, 2008.

[13] J. Deng, R. Han, and S. Mishra, "Decorrelating wireless sensor network traffic to inhibit traffic analysis attacks," *Pervasive and Mobile Computing*, vol. 2, no. 2, 2006.