SPDA: Secure and Privacy-preserving Data Aggregation in Wireless Sensor Networks

Wenbo He
Department of Computer Science
University of Illinois at Urbana-Champaign

Hoang Nguyen
Department of Computer Science
University of Illinois at Urbana-Champaign

Xue Liu
School of Computer Science
McGill University

Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign

Tarek Abdelzaher
Department of Computer Science
University of Illinois at Urbana-Champaign

ABSTRACT

Data aggregation is an efficient mechanism to collect statistics of data in wireless sensor networks. However, the shared-medium nature of communication facilitates eavesdropping and packet tampering/injection by adversaries. Hence, wireless sensor networks pose both privacy and security concerns on data aggregation. In this paper, we present *SPDA*—a secure and privacy-preserving data aggregation technique. We utilize disjoint aggregation paths/trees to collect data of interest, thus the *base station* can verify the integrity of the aggregation result by comparing the result from different paths/trees. To address data privacy, we cloak the privacy-sensitive data by sliding and reassembling technique. We evaluate the *SPDA* scheme in terms of capacity of privacy-preservation, communication overhead, and data aggregation accuracy. The evaluation is conducted through both theoretical analysis and simulation.

1. INTRODUCTION

Wireless sensor networks are attractive to many potential applications from military surveillance to civilian usage. However, the protocol design for real-world wireless sensor networks is very challenging. First of all, sensor nodes usually have limited computation and communication resources. Though low-power wireless devices may work collaboratively to fulfill certain tasks, each individual sensors cannot afford complex computation or large communication load. Second, wireless sensors are are deployed in open and/or hostile environments, where wireless sensors may be incapable to provide reliable functions or compromised by malicious adversaries. Hence, it is important to preserve the integrity of the collected data. Third, as sensor network applications expand to more and more civilian usage, such as wireless meter services to get sensitive measurements of everyday life, preserving data privacy becomes an increasingly important concern.

A sensor network is usually used to get particular attributes of the investigated area and send the readings or statistical results to the *base station*. Therefore, an important feature of sensor networks is the ability to answer queries about the data acquired by the sensors. When handling queries, data aggregation can achieve an order of magnitude reduction in communication compared to centralized approaches which return all raw data to the *base station* [1] . However, the information compression nature of data aggregation poses additional challenges to secure and privacy-preserving data aggregation in wireless sensor networks. In data aggregation, an intermediate node may maliciously manipulate (modify, forge, or discard) the aggregation values, so a single compromised node is able to significantly alter the final aggregation value. It is hard for a *base station* to verify the correctness of the aggregation result without knowing individual sensor readings. Additionally, privacy-preserving mechanisms may barricade the security measures, since sensor nodes cannot enforce public surveillance with the privacy concern.

Note that end-to-end encryption and/or decryption is not a good candidate to achieve secure and privacy-preserving data aggregation in sensor networks. If end-to-end communications are encrypted, the intermediate nodes could not easily perform in-network processing to get aggregated results. Castelluccia, Mykletun and Tsudik propose to use additively homomorphic encryption in [2], which allows efficient aggregation of encrypted data without decryption involved in the intermediate nodes. However, the proposed encryption scheme is malleable, so that the transformations on the cipher-text will produce meaningful changes in the plain-text. The malleability is an undesired property for an encryption scheme, since it allows an attacker to modify the contents of a message easily.

In despite of these challenges, we are in great demand of protocols to reinforce both privacy and security (data integrity) of data aggregation in large sensor networks. As an example, the advanced metering systems for data collection and control on electronic power grid [3] demonstrate such demand. Moreover, the advanced meters could be used for several other purposes for economic viability. It is crucial to preserve the integrity and confidentiality of data within

each application, while still permitting controlled cooperation between different applications.

Przydatek, Song and Perrig present *SIA* protocol in [4]. *SIA* addresses the integrity of the aggregation by constructing efficient random sampling mechanisms and interactive proofs. Due the the random sampling mechanisms, the final aggregation results accepted by the *base station* may not be very accurate. Moreover, when the sample size is large, the additional communication overhead may cancel out the benefit from data aggregation in bandwidth consumption. Yang, Wang, Zhu and Cao propose *SDAP* protocol [5] for secure data aggregation in sensor networks using "divide-and-conquer" and "commit-and-attest" principles. Similar to *SIA*, due to the statistical detection, *SDAP* may not able to detect the attacks which change the intermediate aggregation result mildly.

In privacy-preservation domain, Huang, Wang and Borisov address the problem in a peer-to-peer network application in [6]. Privacy-preserving data aggregation schemes in wireless sensor network environment have been studies in [7]. However, the work in privacy preservation domain does not assume data manipulation attacks.

In this paper, we present a novel method *SPDA* (Secure and Privacy-preserving Data Aggregation), to address the challenges for additive aggregation functions, such as *sum*, *count*, *average*, *variance* and other *moment* of the measured data.

In the *SPDA* scheme, we build node-disjoint aggregation trees interweaving with each other in a sensor network. Since a node can only be in a single aggregation tree, malicious node can only pollute the aggregation result on a single tree. In this case, the *base station* can easily verify the integrity of the aggregation results. To preserve the data privacy, a sensor hides its reading by slicing it into pieces and sends encrypted data slices to different aggregators within its vicinity. Upon receiving the pieces from sensor nodes, aggregators calculate the intermediate aggregate values and further aggregate them to the *base station* along the disjoint trees. As long as the inputs to the disjoint trees are the same from each sensor node, the final aggregation results should be the same from two disjoint aggregation trees, if without node failure or packet loss. Therefore, both privacy and data integrity can be preserved by *SPDA* while aggregation is carrying out.

To our best knowledge, this work is the first to address both security and privacy preservation of data aggregation in wireless sensor networks. The proposed *SPDA* scheme is light-weighted in terms of computation and communication. Moreover, our scheme achieves almost 100% accuracy of the aggregation results in a reasonable dense network.

The rest of the paper is organized as follows. Section 2 describes the background and requirements of secure and privacy-preserving data aggregation in wireless sensor networks. Section 3 provides our detailed scheme of *SPDA*. Section 4 evaluates the proposed scheme by analysis and simulation. After that, Section 5 summarizes the related work. We conclude our findings and lay out future research directions in Section 6.

2. MODEL AND BACKGROUND

A wireless sensor network is deployed in a certain area to detect the common phenomena, or answer queries about (or obtain statistics of) the investigated objects. There are two observations that promote the data aggregation techniques in wireless sensor networks. First, there may be high redundancy in the raw data. Hence, it is unnecessary to report the raw data to users. Second, in-network processing can help to prune the unnecessary information, thus save tremendous bandwidth. Therefore, many applications in wireless sensor network are based on data aggregation techniques, where the sensor network provides a condensed view of the physical environment to users.

2.1 Network Model

In this paper, a sensor network is modelled as a connected graph $G(\mathbb{V}, \mathbb{E})$. A vertex v, $v \in \mathbb{V}$ in the graph represents a sensor node. An edge e, $e \in \mathbb{E}$ represents a wireless link. As long as two sensors are able to communicate directly, there exists an edge connecting them in the graph.

Generally, there are three types of nodes in a data aggregation protocol: *base station*, *aggregator* and *leaf node*. The *base station* is the node where users initiate a query and the aggregation result is destined. In this paper, we only consider a single *base station* case. It is readily extensible to multiple *base station* cases. In data aggregation protocols [1] [8] [5], a spanning tree (usually refer to *aggregation tree*) root at the *base station* is constructed. The non-leaf nodes, except the root, in the *aggregation tree* serve as intermediate *aggregators*, which are responsible for forwarding queries and combining answers from its children and forwarding a single message of the intermediate aggregation result to its parent. Note that any sensor node may serve as an aggregator. In the data aggregation, a node decides whether of not to be an aggregator when it hears the query either from the *base station* or forwarded by an aggregator. Note that a node follows different rules to elect itself as an aggregator in different protocols. If a node decides to be an aggregator, then it continues forwarding the query. Upon receiving intermediate aggregation results from its children, the aggregator combines these results and sends the local intermediate result to its parent. It is desired that all the sensor nodes forward the query when the network is not dense enough. In this case, all the sensors serve as aggregators.

2.2 Data Aggregation Function

We consider a network of N sensors. A generic aggregation function is defined as $y(t) \triangleq f(r_1(t), r_2(t), \cdots, r_N(t))$, where $r_i(t)$ is the individual sensor reading by node i at time t. Typical functions of f include *sum*, *average*, *min*, *max* and *count*. In this paper, we focus on additive aggregation functions. It is worth noting that using additive aggregation functions is not too restrictive, because it is the base of many statistics functions, such as *average*, *count*, *variance*, *standard deviation*, and so on. For example, to get the variance of all the sensing data $r_i(t), i \in \mathbb{V}$, $f(t) = E(r_i^2(t)) - E(r_i(t))^2$ for $i \in \mathbb{V}$, each sensor needs to contribute three inputs as the original data in the data aggregation, they are 1 (*count*), $r_i(t)$, and $r_i^2(t)$. Without loss of generality, we assume $y(t) \triangleq f(d_1(t), d_2(t), \cdots, d_N(t))$, where $d_i(t)$ is a function of $r_i(t)$ and $d_i(t)$ is viewed as the individual sensor input (or virtual reading) in the rest of the paper.

2.3 Key Management Model

To prevent the eavesdropping and other malicious attacks by outsiders, messages in a wireless sensor network may be encrypted and authenticated. Substantial work [9, 10, 11, 12, 13, 14, 15] has been conducted to set up secret keys between two sensor nodes. We do not want to limit ourselves to any single key management scheme. Instead, we assume each pair of sensor nodes share a secret key. In *SPDA*, we only encrypt messages when individual sensors report their privacy-sensitive data at the first mile to protect the data privacy (in Section 3).

2.4 Attack Model

A malicious attacker can perform a wide variety of attacks to break privacy and integrity of results. Different attackers are likely to have different objectives. In this paper, we focus on defence of

the attacks in the following categories.

- *Eavesdropping:* In an eavesdropping attack, an attacker attempts to obtain private information by either overhearing the transmissions over its neighboring wireless links or gossiping (colluding) with other sensor nodes to uncover the secret of a certain node. Eavesdropping threatens the data privacy. With privacy concern, each sensor node should only keep its readings to itself.

- *Data Pollution:* In a data pollution attack, an attacker tampers with the intermediate aggregation result. The purpose of the attack is to make the *base station* accept the wrong aggregation results, and thus make the improper decisions. In this paper, we do not consider the attack where a sensor node reports a false value instead of the real sensor reading. As indicated in [16][5], the impact of such attack is usually limited. Therefore, a more serious concern is the case where a non-leaf node close to the root of aggregation tree is compromised.

- *Node Failure:* In a node failure, an aggregator may not respond queries or not forward the intermediate aggregation results. In a reasonable dense network, not responding queries is not very harmful. In this paper, we consider the misbehavior of an aggregator which stops forwarding aggregation results. Malicious attacks may cause node failures. Or even without malicious attacks, it is likely that some sensor nodes fail to forward the aggregated results due selfishness or resource constraint.

2.5 Design Goal

The overarching design goal of this paper is to provide a secure and privacy-preserving data aggregation scheme, which is robust against *eavesdropping*, and capable to detect *data pollution* and *node failures*. Therefore, the desired characteristics of a data aggregation scheme should satisfy the following criteria:

1. **Privacy-preservation:** Privacy is one of the major obstacles to apply the wireless sensor networks to civilian applications, where curious individuals may attempt to determine more detailed information about their neighbors by eavesdropping on the communications of wireless meters. It is increasingly important to develop privacy-preserving data aggregation schemes to ensure data privacy against eavesdropping and gossiping by curious neighbors.

2. **Data Integrity:** Since data aggregation results may be used to make critical decisions, a *base station* needs to attest the integrity of the aggregated result before accept it. Therefore, it is desired that data aggregation scheme has the ability for integrity check.

3. **Efficiency:** Data aggregation achieves bandwidth efficiency by using in-network processing. In integrity- and privacy-preserving data aggregation schemes, additional communication overhead is unavoidable to achieve additional features. However, we will keep the additional overhead as small as possible.

4. **Accuracy:** An accurate aggregation result of sensor readings is usually desired. Therefore we take accuracy as a criterion to estimate the performance of integrity- and privacy-preserving data aggregation schemes. To ensure the accuracy of aggregation results, randomization techniques [17, 18, 19, 20] are not applicable.

3. SECURE AND PRIVACY-PRESERVING DATA AGGREGATION PROTOCOL

3.1 Protocol Overview

Data aggregation is initiated by a *base station*, which broadcasts a query to the whole network. Upon receiving the query, leaf nodes report their readings to their aggregators (parents along the spanning tree rooted at the *base station*), and then aggregators perform in-network processing and route the aggregated results back to the *base station*. However, in most conventional data aggregation protocols, data integrity and privacy are not preserved at the same time.

To address privacy, we adopt *Slice-Mix-AggRegaTe (SMART)* technique [7]. In *SMART*, each participating sensor node (either a leaf node or an aggregator) hides its individual data by slicing the data and sending encrypted data slices to different neighboring aggregators[1], then the aggregators collect and route aggregated results back to the *base station*. Due to the associative property of addition, *SMART* is able to conceal the original sensor readings as well as keep the aggregation efficient and accurate.

To achieve the integrity, we resort to redundancy check by constructing two disjoint aggregation trees. Each sensor node needs to send its reading to both aggregation trees, and makes the inputs to both trees equal. The disjoint aggregation trees perform data aggregation individually. Therefore, data pollution attacks can be detected at the *base station* by comparing aggregation results along the disjoint aggregation trees. If the aggregation results agree with each other, then the *base station* will accept the result. Otherwise, the *base station* knows that there exist either data pollution attacks or node failures, or both.

In this section, we present the details of the *SPDA* protocol. There are three phases: *tree construction*, *privacy-preserving data report*, *secure data aggregation* in *SPDA* as follows.

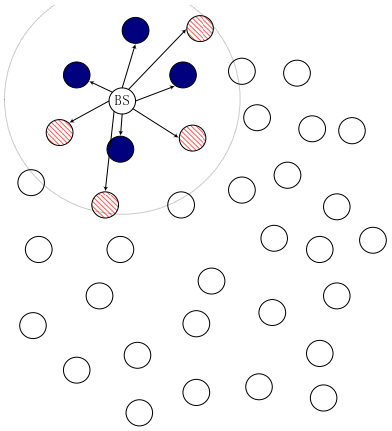
3.2 Disjoint Aggregation Tree Construction

In order to utilize redundancy to verify integrity of aggregation results, we construct node-disjoint aggregation trees in the first phase of *SPDA*. In this paper, we build two disjoint aggregation trees. Assuming m is the number of disjoint aggregation trees, $m = 2$. We call the two aggregation trees, *red aggregation tree* and *blue aggregation tree* respectively. The disjoint aggregation tree construction phase can be easily generalized to build multiple aggregation trees ($m > 2$). However, to achieve good coverage of disjoint trees when $m > 2$, the network must be very dense. In this phase, each node, except the *base station*, takes one of the three roles: *red aggregator*, *blue aggregator* or *leaf node*. The *base station* is the root of both *red aggregation tree* and *blue aggregation tree*, so it is both a *red aggregator* and a *blue aggregator*.

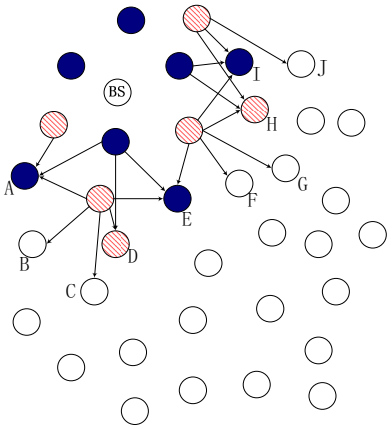
The disjoint tree construction follows the procedure illustrated in Figure 1, where the dark colored solid nodes represent blue aggregators and light colored solid nodes represent red aggregators. First, the *base station* BS initiates a query by issuing a *HELLO* message. Upon receiving the *HELLO* messages from both red and blue aggregators, a node makes the decision on its role. A node becomes a *red aggregator* with probability $p_r (0 < p_r < 1)$, becomes a *blue aggregator* with probability $p_b (0 < p_b < 1)$ and $0 < p_r + p_b \leq 1$), and becomes a *leaf node* with probability $1 - p_r - p_b$.

Note that if a node is unable to reach either red aggregators or

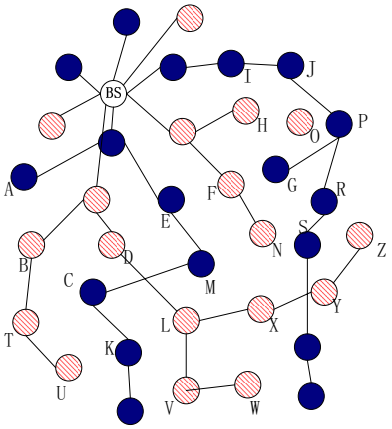
[1]Though a node only has one parent node (aggregator) in an aggregation tree, it is very likely that the node is able to reach other aggregators within its transmission range in a reasonable dense network



(a) BS triggers the aggregation by a HELLO message, then nodes receive such a message select their roles.



(b) Aggregators forward the HELLO messages. Node A, D, E, H, I receives HELLO messages from both blue aggregators and red aggregators, so they randomly choose to be red or blue aggregators. Node B, C, F, G, J only receives HELLO message from red aggregators, so these nodes should wait until they receive HELLO messages from both blue and red aggregators to join either blue tree or red tree.



(c) As the disjoint tree construction procedure continues, we can form two disjoint aggregation tree.

Figure 1: Illustration of disjoint tree construction for $k = 4$ (hence $p = 1$ that all nodes serve as aggregators.)

blue aggregators within one hop, the node cannot send its data values directly to both colored aggregators. In order to achieve the separation of data aggregation along the disjoint trees, red aggregators are not allowed to forward the data for blue aggregators, and vice versa. Therefore, If a node never receives *HELLO* message from either red or blue tree, the node does not participate in data aggregation.

To make more nodes receive *HELLO* messages from both red and blue aggregators, it is desired to balance the red aggregators and blue aggregators in a given neighborhood. Hence, a node is likely to choose red color, if there are more blue aggregators than red aggregators in its neighborhood. A node can estimate the number of red/blue aggregators in its neighborhood from the received *HELLO* messages. In this case, upon receiving *HELLO* massage from at least one *blue aggregator* and at least one *red aggregator*, a node waits for a certain period of time to get enough *HELLO* messages before it makes the decision on its color. Therefore, the node can have a good estimation of colors of its neighbors, and selects its color to maximize the chance that other nodes will receive *HELLO* messages both red and blue aggregators. We will show that only a very small portion of nodes do not participate the data aggregation in our scheme when the network is dense enough (in Section 4).

If a node becomes a *red/blue aggregator*, it will join the corresponding red/blue aggregation tree and forward the *HELLO* message to its neighbors; otherwise, the node is a *leaf node*. As this procedure goes on, disjoint aggregation trees, red tree and blue tree, are constructed. In *SPDA*, the following properties are desired:

- The disjoint aggregation trees are interweaved with each other. Therefore almost every node can find a *blue aggregator* and a *red aggregator* in its neighborhood. Since if a node does not have a red aggregator or blue aggregators in its neighborhood, the node cannot participate the data aggregation. In order to have more nodes participate the data aggregation, thus the aggregation result is more accurate, both aggregation trees should cover network as much as possible. In this case, we should have enough number of aggregators.

- On the other hand, in a very dense network, we desire that only a portion of nodes serve as blue or red aggregators. Since leaf nodes do not need to forward *HELLO* message and intermediate results to its parents, we can reduce the bandwidth consumption by reducing the number of aggregators.

To ensure these two contradictory properties, we adopt adaptive strategy to determine p_r and p_b for each individual node according to the number of *HELLO* messages the node received from *red aggregators* and *blue aggregators*. The value $p_r + p_b$ should be larger, if a node gets a smaller number of *HELLO* messages. Therefore, we can get better coverage of the aggregation tree. Also, if a node hears more *HELLO* messages from *red aggregators* than from *blue aggregators*, the node will take larger chance to be a *blue aggregator* to balance the blue and red aggregation trees. Therefore, we can determine p_r and p_b accordingly,

$$p_r = p\frac{N_{blue}}{N_{blue} + N_{red}},$$
$$p_b = p\frac{N_{red}}{N_{blue} + N_{red}}. \quad (1)$$

where N_{blue} is the number of *HELLO* messages from the blue aggregators, and N_{red} is the number of *HELLO* messages from the red aggregators. p is the probability that a node becomes an aggregator (either red or blue), hence $p = p_r + p_b$. We can determine

value p as follows

$$p = \begin{cases} \frac{k}{N_{blue}+N_{red}}, & \text{if } (N_{blue}+N_{red}) > k \\ 1 & \text{, otherwise.} \end{cases}$$

In the above equation, $k(k \geq 2)$ is predetermined parameter. Value k balances the coverage of the aggregators and communication overhead. If k is large, then all nodes are aggregators. If k is small, some nodes in the network may not be covered by aggregation trees. The compelling features of using a fixed k value are its simplicity and its inherent adaptability to network density. That is, in a dense network, a portion of nodes are aggregators; in a non-dense network[2], all nodes are aggregators. In a reasonable dense network, we can reduce Equation (1) to Equation (2) below for simplicity.

$$p_r = p_b = 0.5 \quad (p = 1). \tag{2}$$

To ensure the security of data aggregation results, the disjoint tree construction protocol should guarantee that a node cannot be in both blue tree and red tree, so that the constructed aggregation trees are node-disjoint. Though it is possible that an adversary may intent to send two *HELLO* messages with different colors. Such behavior can be easily detected by its neighbors due to the shared-medium nature of wireless links. Therefore, the adversary can be excluded from both aggregation trees.

3.3 Privacy-preserving Data Report

To preserve the privacy in data aggregation, sensors need to hide the their original readings in the first hop data reporting. In *SPDA*, each sensor hides its reading by slicing it into pieces and randomly sends encrypted data slices to its neighboring aggregators. Then aggregators assemble the received data and treat the assembled data as their own readings. Then aggregators follows aggregation procedure described in Section 3.4 to route the aggregated result to the *base station*. Privacy-preserving Data Report phase includes two steps: *data slicing* and *data assembling*.

3.3.1 Slicing

First, a node needs to randomly select l red aggregators and l blue aggregators from its neighboring nodes (including itself). If a node itself is a red aggregator, then it always selects itself and $l-1$ other red aggregators. Then the node randomly slices the data into l pieces and send a piece to each of the selected neighboring red aggregators including itself. The node also slices the original reading into l pieces independently to the previous l slices, and then sends a piece to each of the selected blue aggregators in the neighborhood. Totally, each node takes $2l-1$ transmissions in the slicing step. Note that when nodes send the sliced data pieces to their neighbors, link level encryption is needed. Without encrypting sliced pieces, an adversary is able to eavesdrop all the transmissions by a given sensor node due to the shared-medium nature of wireless links. Hence, the adversary can easily recover the original data of that node[3].

Figure 2 depicts the slicing step at node i, assuming node i is a red aggregator. We denote $d(i)$ as the private data at node i, and d_{ij} as a slice of data sent from node i to node j. Note d_{ii} is kept locally at node i, no transmission is needed for d_{ii}. For nodes to which node i does not send any slice, $d_{ij} = 0$. The final aggregation result is expressed as

[2]Note that in a sparse network, even if all the nodes are aggregators, the coverage is not good. So *SPDA* requires adequate network density.

[3]In *TAG*, even if the link level encryption is used, neighbors of leaf nodes can easily know the original data held by the leaf nodes.

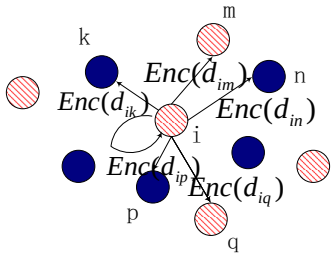


Figure 2: Slicing step by node i ($l = 3$)

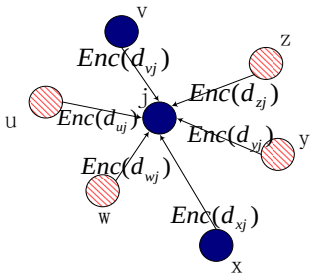


Figure 3: Assembling at node j

$$f = \sum_{i=1}^{N} d(i) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}}{2}. \tag{3}$$

Let $\mathcal{B}$ stands for the blue aggregator set, and $\mathcal{R}$ stands for the red aggregator set. Then

$$f = \sum_{i=1}^{N} d(i) = \sum_{i=1}^{N}\sum_{j\in\mathcal{B}} d_{ij} = \sum_{i=1}^{N}\sum_{j\in\mathcal{R}} d_{ij}. \tag{4}$$

3.3.2 Assembling

When a node j receives an encrypted slice, it decrypts the data using its shared key with the sender. Upon receiving the first slice, the node waits for a certain time, which guarantees that all slices of this round of aggregation are received. Then, it sums up all the received slices $r(j) = \sum_i^N d_{ij}$, where $d_{ij} = 0$, if node i does not send a sliced data to node j. Figure 3 describes the assembling step, where $r(j) = d_{vj} + d_{uj} + d_{wj} + d_{xj} + d_{yj} + d_{zj} + d_{jj}$. After the assembling, node j treats $r(j)$ as its data reading in the secure data aggregation phase (in the next section).

3.4 Secure Data Aggregation

Now, two disjoint aggregation trees have been constructed, and aggregators hold an assembled data for data aggregation. The secure data aggregation just needs to follow the standard aggregation protocol along two disjoint trees: When a node gets all data slices, it forwards a message of the sum addressed to its parent (the aggregator of the same color with the node). The parent then forwards the message along the tree. Eventually the aggregation reaches the root (*base station*). When there is no attack, it is easy to derive that

$$\sum_{j\in\mathcal{B}} r(j) = \sum_{j\in\mathcal{B}} \left(\sum_{i=1}^{N} d_{ij}\right) = \sum_{i=1}^{N}\sum_{j\in\mathcal{B}} d_{ij} = \sum_{i=1}^{N} d(i) = f. \tag{5}$$

Similarly,

$$\sum_{j\in\mathcal{R}} r(j) = \sum_{j\in\mathcal{R}} \left(\sum_{i=1}^{N} d_{ij}\right) = \sum_{i=1}^{N}\sum_{j\in\mathcal{R}} d_{ij} = \sum_{i=1}^{N} d(i) = f. \quad (6)$$

Since not any single node is on two aggregation trees, if a node inserts or alters the intermediate aggregation value, the aggregation results from different trees will be different. Therefore, at the *base station* if the aggregation results from different trees agree with each other: $\sum_{j\in\mathcal{B}} r(j) = \sum_{j\in\mathcal{R}} r(j)$, then the *base station* will accept the aggregation result; otherwise, reject it.

4. EVALUATION

4.1 Theoretical Analysis

4.1.1 Coverage of Aggregation Trees

In *SPDA*, a sensor node reports its reading to the *base station* by aggregation only when the sensor node is able to reach both red and blue aggregation trees within one hop. Otherwise, the sensor node does not contribute its reading. In the case that a node cannot reach both aggregation trees, the node is disconnected from the *base station*. If a node is connected to the *base station*, the node is covered by aggregation trees. We define $\Phi(G)$ as the probability that all the nodes in graph G are covered by both aggregation trees. The coverage of aggregation trees implies the accuracy of aggregation results, since if the coverage is poor, then a large number nodes cannot contribute their readings to the aggregation result.

Consider a random graph $G(N, r)$, where N is number of nodes and r is the transmission range of a node. As shown in [21], as N is large, $G(N, r)$ is connected if and only if there is no isolated nodes (nodes with degree zero). Therefore, if we randomly assign red or blue color to nodes in the graph $G(N, r)$, assuming X as the number of nodes which are isolated from either blue nodes or red nodes, then

$$\Phi(G) = P(X = 0). \quad (7)$$

Define X_i as the indicator variable of whether node i has both blue and red neighbors within one hop distance, so

$$X_i = \begin{cases} 0, \text{ i has both blue and red neighbors;} \\ 1, \text{ otherwise.} \end{cases} \quad (8)$$

As network size is large enough, $\{X_i\}$ can be approximated to identical independent distribution (IID). Therefore, the total number of nodes which are isolated by either of the aggregation tree is $X = \sum_{i=1}^{N} X_i$. Denote d_i as the number of physical neighbors of node i. The probability that i is isolated by red aggregation tree is given as $p_b^{d_i}$. Similarly, i is isolated by blue aggregation tree with probability $p_r^{d_i}$. Therefore, the probability that a node is surrounded by all blue neighbors or red neighbors is:

$$p_i = 1 - (1 - p_b^{d_i})(1 - p_r^{d_i}). \quad (9)$$

p_i is the probability that a node i is isolated by either blue nodes or red nodes, thus $p_i = P(X_i = 1)$. Since $X = \sum_{i=1}^{N} X_i$. Applying Markov Inequality $P(X \geq 1) \leq E[X] = \sum_{i=1}^{N} p_i$, we can obtain the lower bound of $\Phi(G)$.

$$\Phi(G) \geq 1 - \sum_{i=1}^{N} p_i. \quad (10)$$

This bound is tight for small p_i values, which holds for dense networks, where d_i is large. As an example, consider a d-regular graph, assuming $p_b = p_r = 0.5$, we have $\Phi(G) \geq 1 - N(1 - \frac{1}{2^{2d}})$ according to Equation (9). Therefore, $\Phi(G) \geq 0.999$ for $N = 1000$ and $d = 10$. We can conclude from Equation (10) that the coverage of aggregation trees are very good for dense networks.

4.1.2 Communication Overhead

Figure 4 summarizes the communication messages sent and received by each node in data aggregation under *TAG* and *SPDA* respectively. In *TAG*, each node sends two messages to answer a query: a *HELLO* message and a message for intermediate result. In *SPDA*, additional $2l - 1$ messages are introduced by slicing the original privacy-sensitive data. Hence, totally $2l + 1$ messages are sent by each node. We can conclude that the communication overhead ratio of *SPDA* to *TAG* is $\frac{2l+1}{2}$, where l the number of pieces that a node slices its private data.

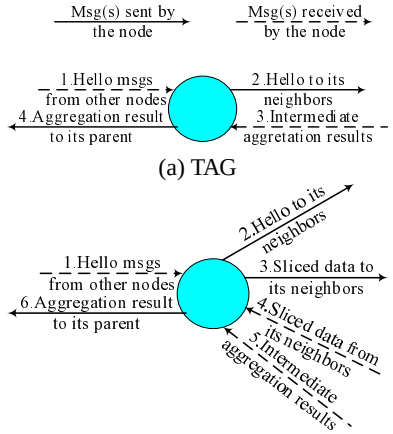


Figure 4: Communication messages

4.1.3 Capacity of Privacy-preservation

As illustrated in Section 3.3, *SPDA* achieves privacy-preservation by slicing the original data. In *SPDA* we use link level encryption to prevent the slices data pieces from being overheard by an adversary. If an eavesdropper gets enough data slices, it can reassemble the slices and obtain the original data. According to different assumptions and design goals, sensor networks may use different types of key management and encryption schemes. One of the merits of *SPDA* protocol is that *SPDA* does not assume any particular key management scheme. In spite of the encryption, there are two reasons that cause the privacy violation:

- Under some key distribution schemes (e.g. random key predistribution [9] [10]), two neighboring nodes share a common key for communication. However, a third node may also hold the key and is able to decrypt messages between the two nodes.
- An attacker compromises multiple neighbors of a node and gets the shared keys with the node. In this case, the attacker may decrypt enough slices of data sent by a node, hence obtain the original data.

In general, we assume p_x as the probability that an attacker breaks the security of a given link. Then we concern the capacity of privacy-preservation at a certain node i. The capacity is represented by the probability $P^i_{disclose}(p_x)$, which is the probability that node i discloses its reading to someone else under a given p_x.

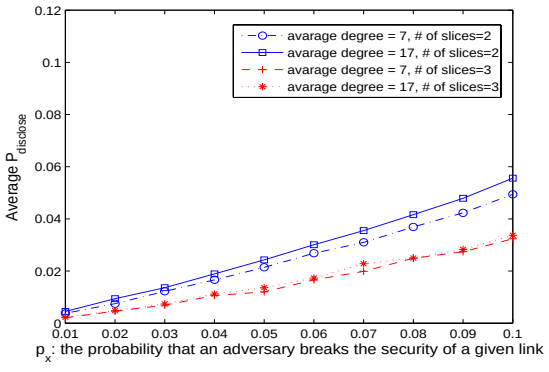


Figure 5: Capacity of privacy-preservation in SPDA

Assuming a node slices the original reading into l pieces, to reveal the privacy-sensitive data held by a node i, an attacker needs either to break l outgoing links, when node i sends l slices to aggregators of different colors; or to break $l-1$ outgoing links and all of the incoming links as well, when node i sends $l-1$ slices to aggregators of the same color (in this case one of the slices is kept locally at node i). Denote $E[n_l(i)]$ as the expected number of incoming links of node i. Then $E[n_l(i)] = \sum_{j \in Neighbor(i)} \frac{(2l-1)}{d_j}$, where $Neighbor(i)$ is the set of node i's one hop neighbors, and d_j is the physical degree of node j. We conclude that

$$P^i_{disclose}(p_x) = 1 - (1-p_x^l)(1-p_x^{l-1+E[n_l(i)]}). \quad (11)$$

As an example, let us consider a d-regular network ($d >> l$), where $E[n_l(i)] = 2l-1$. For $l=3$, $d=10$ and $p_x = 0.1$, the probability that for privacy violation is $P^i_{disclose}(0.1) = 0.001$. For a random network topology, the average of $P^i_{disclose}(p_x)$ is defined as $\overline{P_{disclose}(p_x)} = \frac{1}{N}\sum_{i=1}^{N} P^i_{disclose}(p_x)$, which is much larger than that in regular graph.

Figure 5 plots $\overline{P_{disclose}(p_x)}$ over p_x, when 1000 sensor nodes are distributed in a square area, and the average degree of a node is 7 and 17 respectively. We can see that the privacy preservation capacity of *SPDA* is insensitive to network density. We also notice that although $\overline{P_{disclose}(p_x)}$ is smaller for $l=3$ than that for $l=2$, the difference is not tremendous. Since the privacy preservation performance for $l=2$ is good enough, and additionally, a larger l yields larger overhead, we recommand $l=2$ in *SPDA*.

4.1.4 *Capable to Detect Data Pollution and Node Failures*

In *SPDA*, encryption is a necessity for privacy-preservation. However, no encryption or decryption is needed to achieve the integrity when there exists data pollution or node failures. *SPDA* is able to detect multiple attackers as long as they do not collude with each other. Here we utilize redundancy to verify the integrity by constructing disjoint aggregation trees. Any individual attackers may manipulate the intermediate aggregation results along one aggregation tree, but the attackers cannot pollute the data on the other tree. When the *base station* gets the aggregation results from both aggregation trees, say S_b and S_r, it accepts the results if $|S_b - S_r| \leq Th$. Th is usually not zero, because data loss may occur due to collisions. The denser the network is, the larger Th should be, since more contention and collision occur in a denser network.

4.2 Simulation Results

SPDA employs redundancy for integrity and data shredding for privacy. When comparing with standard data aggregation protocols, say *TAG*, *SPDA* achieves two important features, integrity and privacy, at the cost of communication overhead. We provide the analytical results regarding the aggregation performance in 4.1. Next, we will assess the performance of *SPDA* by simulation. In our experiments, sensor nodes are randomly deployed over a 400meters $\times$ 400meters area. The transmission range of a sensor node is 50 meters and data rate is 1 *Mbps*.

4.2.1 *Communication Overhead*

We implement *TAG* and *SPDA* in ns-2 simulator. Figure 6 shows the communication overhead of *TAG*, *SPDA* without slicing ($l=1$), and *SPDA* with $l=2$. The simulation verifies our theoretical conclusion that when we slice the data into l pieces, the total bandwidth consumption is around $\frac{2l+1}{2}$ times of that in standard *TAG* protocol. When we deploy less than 300 sensors in the 400m $\times$ 400m square, the average degree is less than 14. Such a network density is relatively low. In this case some sensor nodes may not receive the *HELLO* message, some nodes may not have enough red and blue aggregators in their one hop neighborhood to send the sliced data. Therefore, they cannot participate the data aggregation according to *SPDA* protocol. So the total bandwidth consumption is low when $N < 300$. This also explains why the accuracy under *SPDA* is poor as shown in Section 4.2.2, when network density is low ($N < 300$). To show the effect of network density on communication overhead and accuracy metrics, Table 1 summarizes the average node degree according to a given number of nodes on a 400meter $\times$ 400meter square.

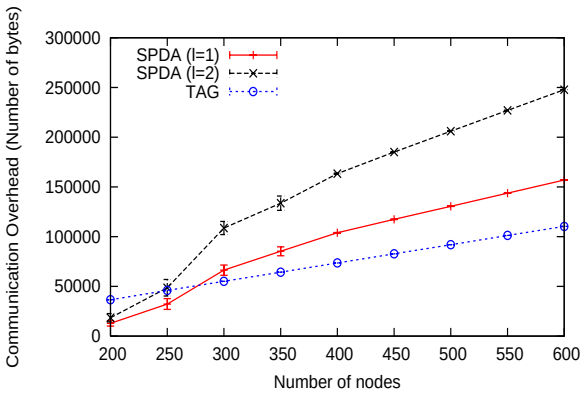


Figure 6: Bandwidth consumption of *SPDA* v.s. *TAG*

Table 1: Network size v.s. network density

Number of nodes	200	300	400	500	600
Average degree	8.8	13.7	18.6	23.5	28.4

4.2.2 *Coverage and Accuracy*

When there is no data loss in the data aggregation, *SPDA* yields 100% accurate aggregation results. However, in a real sensor network data loss is inevitable due to the following reasons:

1. In disjoint tree construction stage, if the network density is low, then some nodes may be unreachable by both red and blue aggregation trees. In this case, those nodes do not participant in the data aggregation. Thus, some data is missing in the final aggregation result.

2. In the data slicing stage, assuming each reading is sliced into l pieces, if a red node cannot find $l-1$ red neighbors and l blue neighbors within one hop, the node does not participate the data aggregation. Hence, the data held by such a node get lost.

3. In disjoint tree construction, slicing, and data aggregation stages, the data loss may be caused by collision in wireless channels.

Figure 7 shows the percentage of nodes which participate in the data aggregation. Figure 8 demonstrates the accuracy metric of *SPDA* comparing with *TAG*. We define the accuracy metric as the ratio of the collected sum by a given data aggregation protocol to the real sum of all individual sensors. A higher accuracy value means the collected sum using the specific aggregation scheme is more accurate. We use value 1.0 representing the ideal situation, where there is no data loss. Factor 1) and factor 2) resulting in data loss are embodied in Figure 7. All three factors are reflected in Figure 8. Due to the similarity of Figure 7 and Figure 8, we conclude that factor 1) and 2) are dominating factors to cause data loss in sparse network. However, when average degree of a network is large enough, factor 3) is the major reason for data loss, which is very small though (usually less 5%). From Figure 7, we can also conclude that in order to achieve excellent accuracy under *SPDA* with the recommended parameter $l = 2$, the average network density should be larger than 18.

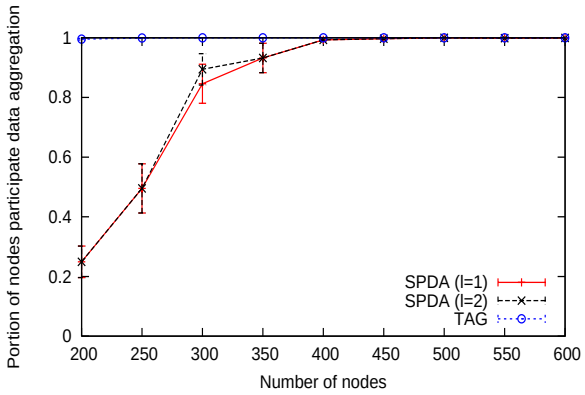


Figure 7: Percentage of nodes which participate in data aggregation

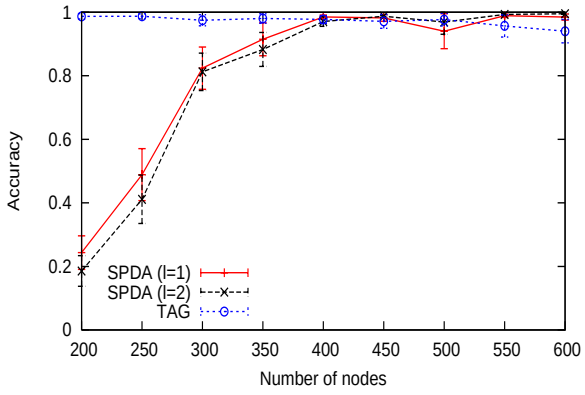


Figure 8: Accuracy of data aggregation under *SPDA* v.s. *TAG*

5. RELATED WORK

LeMay, Gross, Gunter and Garg summarize the functional characteristics of wireless metering sensors and categorizes attackers in [3]. Both privacy and security are concerns in the given scenarios. To prune redundant messages to save bandwidth and energy, previous work [1, 22, 23, 24, 25, 26, 27, 28] address data aggregation with the assumption that all sensors are working in trusted and friendly environment. However, in reality, sensor networks are likely to be deployed in an untrusted environment, where links, for example, can be eavesdropped. An adversary may compromise cryptographic keys and manipulate the data.

Previous work [4, 5, 29] investigate secure data aggregation against the adversaries who try to tamper the intermediate aggregation result. To reinforce security in sensor network, communication should be encrypted and authenticated. Many papers study how to set up secret keys between sensor nodes to bootstrap secure communication, such as [9, 10, 11, 12, 13, 14, 15, 8]. For encrypted data, an aggregator has to decrypt each received message, then aggregate the messages according to the corresponding aggregation function, and finally encrypt the aggregation result before forwarding it. It is fairly expensive and complicated to perform "decryption-aggregation-encryption" procedure for data aggregation. As a result, [30] and [2] propose homomorphic stream ciphers, that allow efficient aggregation of encrypted data without decryption. However, due to the malleability of additive homomorphic encryption, an attacker may produce meaningful changes in the original data.

Huang, Wang and Borisov propose a privacy-preserving peer-to-peer troubleshooting scheme using PeerPressure in [6]. They constructed a friends peer-to-peer overlay to gather PC configuration samples using history-less random walk, during which search is carried out simultaneously with secure parameter aggregation for troubleshooting. Privacy-preservation has also been studied in the data mining domain [17, 18, 19, 20]. Two major classes of schemes are used. The first class is based on data perturbation (randomization) techniques. In a data perturbation scheme, a random number drawn from a certain distribution is added to the private data. Given the distribution of the random perturbation, recovering the aggregated result is possible. At the same time, by using the randomized data to mask the private values, privacy is achieved. However, data perturbation techniques have the drawback that they do not yield accurate aggregation results. Furthermore, as shown by Kargupta et al. in [19] and by Huang et al. in [20], certain types of data perturbation might not preserve privacy well. Another class of privacy-preserving data mining schemes [31, 32, 33] is based on Secure Multi-party Computation (SMC) techniques [34, 35, 36]. SMC deals with the problem of a joint computation of a function with multi-party private inputs. SMC usually leverages public-key cryptography. Hence SMC-based privacy-preserving data mining schemes are usually computationally expensive, which is not applicable to resource-constrained wireless sensor networks.

6. CONCLUSIONS

Data aggregation is an important technique to save communication bandwidth for data collection in wireless sensor networks. With more and more applications of wireless sensor networks in various domains, the integrity and privacy of the collected data are becoming crucial. However, it is a challenging to design secure and privacy-preserving data aggregation protocols in large sensor networks when some nodes may be malicious. We propose the *SPDA* scheme, using disjoint trees for data aggregation, hence, the *base station* can tell whether or not the collected data is polluted by intermediate aggregators. To protect the privacy of individual sensor

readings, we utilize slicing technique to hide the original privacy-sensitive data. To balance the communication overhead and the capability of privacy preservation, we suggest $l = 2$ in *SPDA* protocol. A notable property of *SPDA* is that it is not a sampling or approximation based protocol, so we can get accurate aggregation result for a reasonable dense network. Simulations show that when the average degree of a network is larger than 18, the aggregation accuracy is very high (around 99% accuracy). The proposed *SPDA* protocol is light-weighted in terms of computational complexity and communication overhead. When sensors aggregate their intermediate results along disjoint aggregation trees, no authentication is needed. To our best knowledge, this work is the first to address both security and privacy preservation of data aggregation in wireless sensor networks.

In the future, we will investigate secure and private-preserving data aggregation schemes under collusive malicious attacks. We will also study privacy-preserving schemes for more general aggregation functions.

7. REFERENCES

[1] S. Madden, M. J. Franklin, and J. M. Hellerstein, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," *OSDI*, 2002.

[2] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient Aggregation of Encrypted Data in Wireless Sensor Networks," *Mobiquitous*, 2005.

[3] M. LeMay, G. Gross, C. A. Gunter, and S. Garg, "Unified architecture for large-scale attested metering," in *proceedings of HICSS-40*, January 2007.

[4] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure Information Aggregation in Sensor Networks," *In Proc. of ACM SenSys*, 2003.

[5] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: A Secure Hop-by-Hop Data Aggregation Protocol for Sensor Networks," *ACM MobiHoc*, 2006.

[6] Q. Huang, H. J. Wang, and N. Borisov, "Privacy-preserving friends troubleshooting network," in *Symposium on Network and Distributed Systems Security (NDSS)*, San Diego, CA, Feburary 2005.

[7] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "PDA: Privacy-preserving Data Aggregation in Wireless Sensor Networks," *UIUC Department of Computer Science Techniqual Report UIUCDCS-R-2006-2773*, September 2006.

[8] A. Mahimkar and T. Rappaport, "SecureDAV: a secure data aggregation and verification protocol for sensor networks," *GLOBECOM*, 2004.

[9] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, November 2002, pp. 41–47.

[10] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *IEEE Symposium on Research in Security and Privacy*, 2003, pp. 197–213.

[11] S. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," in *Proceedings of 9th European Symposium On Research in Computer Security (ESORICS 04)*, 2004.

[12] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, October 2003, pp. 42–51.

[13] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS03)*, October 2003, pp. 52–61.

[14] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient security mechanisms for large-scale distributed sensor networks," in *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS03)*, October 2003, pp. 62–72.

[15] M. J. Miller and N. H. Vaidya, "Leveraging Channel Diversity for Key Establishment in Wireless Sensor Networks," in *IEEE INFOCOM*, 2006.

[16] L. Hu and D. Evans, "Secure Aggregation for Wireless Networks," *In Workshop on Security and Assurance in Ad hoc Networks*, January 2003.

[17] R. Agrawal and R. Srikant, "Privacy preserving data mining," in *ACM SIGMOD Conf. Management of Data*, 2000, pp. 439–450.

[18] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy Preserving Mining of Association Rules," in *Proceedings of The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2002.

[19] H. Kargupta, Q. W. S. Datta, and K. Sivakumar, "On The Privacy Preserving Properties of Random Data Perturbation Techniques," in *the IEEE International Conference on Data Mining*, November 2003.

[20] Z. Huang, W. Du, and B. Chen, "Deriving Private Information from Randomized Data," in *Proceedings of the ACM SIGMOD Conference*, June 2005.

[21] M. D. Penrose, "On k-connectivity for a geometric random graph," *Source Random Structures & Algorithms archive, John Wiley & Sons, Inc. New York, NY, USA*, vol. 15(2), pp. 145–164, September 1999.

[22] C. Itanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *MobiCom*, 2002.

[23] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of Network Density on Data Aggregation in Wireless Sensor Networks," *In Proceedings of the 22nd International Conference on Distributed Computing Systems*, 2002.

[24] A. Deshpande, S. Nath, P. B. Gibbons, and S. Seshan, "Cache-and-query for wide area sensor databases," *SIGMOD*, 2003.

[25] I. Solis and K. Obraczka, "The impact of timing in data aggregation for sensor networks," *ICC*, 2004.

[26] T. Abdelzaher, T. He, and J. Stankovic, "Feedback Control of Data Aggregation in Sensor Networks," *43rd IEEE Conference on Decision and Control*, December 2004.

[27] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust Computation of Aggregates in Wireless Sensor Networks: Distributed Randomized Algorithms and Analysis," *IPSN*, 2005.

[28] X. Tang and J. Xu, "Extending network lifetime for precision-constrained data aggregation in wireless sensor networks," *INFOCOM*, 2006.

[29] H. Chan, A. Perrig, and D. Song, "Secure Hierarchical In-Network Aggregation in Sensor Networks," *In Proc. of ACM CCS*, November 2006.

[30] J. Girao, D. Westhoff, and M. Schneider, "CDA: Concealed Data Aggregation for Reverse Multicast Traffic in Wireless Sensor Networks," in *40th International Conference on Communications, IEEE ICC*, May 2005.

[31] B. Pinkas, "Cryptographic techniques for privacy preserving data mining," *SIGKDD Explorations*, vol. 4, no. 2, pp. 12–19, 2002.

[32] W. Du and M. J. Atallah, "Secure multi-party computation problems and their applications: A review and open problems," in *Proceedings of the 2001 Workshop on New Security Paradigms*. Cloudcroft, NM: ACM Press, September 2001, pp. 13–22.

[33] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1026–1037, 2004.

[34] A. C. Yao, "Protocols for secure computations," in *23rd IEEE Symposium on the Foundations of Computer Science (FOCS)*, 1982, pp. 160–164.

[35] I. D. Ronald Cramer and S. Dziembowski, "On the Complexity of Verifiable Secret Sharing and Multiparty Computation," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, 2000, pp. 325–334.

[36] J. Halpern and V. Teague, "Rational Secret Sharing and Multiparty Computation," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, 2004, pp. 623–632.