Wireless Netw
DOI 10.1007/s11276-009-0177-y

Secure encrypted-data aggregation for wireless sensor networks

Shih-I Huang · Shiuhpyng Shieh · J. D. Tygar

Abstract This paper proposes a secure encrypted-data aggregation scheme for wireless sensor networks. Our design for data aggregation eliminates redundant sensor readings without using encryption and maintains data secrecy and privacy during transmission. Conventional aggregation functions operate when readings are received in plaintext. If readings are encrypted, aggregation requires decryption creating extra overhead and key management issues. In contrast to conventional schemes, our proposed scheme provides security and privacy, and duplicate instances of original readings will be aggregated into a single packet. Our scheme is resilient to known-plaintext attacks, chosen-plaintext attacks, ciphertext-only attacks and man-in-the-middle attacks. Our experiments show that our proposed aggregation method significantly reduces communication overhead and can be practically implemented in on-the-shelf sensor platforms.

This work was supported in part by National Science Foundation, ITRI, Chung Shan Institute of Science and Technology, the International Collaboration for Advancing Security Technology (iCAST) and Taiwan Information Security Center (TWISC), under National Science Council grants NSC96-3114-P-001-002-Y and NSC96-2219-E-009-013, respectively.

S.-I. Huang · S. Shieh
Department of Computer Science and Information Engineering, National Chiao Tung Univerisity, Hsinchu, Taiwan
e-mail: ssp@csie.nctu.edu.tw

S.-I. Huang (✉)
Industrial Technology Research Institute, Hsinchu, Taiwan
e-mail: sihuang@csie.nctu.edu.tw

J. D. Tygar
University of California Berkeley, Berkeley, USA
e-mail: doug.tygar@gmail.com

Published online: 07 May 2009

Keywords Data aggregation · Wireless · Sensor networks · Secrecy · Privacy

List of symbols

S_i	Sensor mote i
g	A one-way function having the following property: $g(x \oplus y) = g(x) \oplus g(y)$
K_i^{EK}	An *encryption key* randomly generated by sensor mote i
K_i^{VK}	A *verification key* used to verify data from sensor mote i

1 Introduction

Wireless Sensor Networks (*WSN*) have emerged as an important new area in wireless technology. A wireless sensor network [1] is a distributed system interacting with physical environment. It consists of *motes* equipped with task-specific sensors to measure the surrounding environment, e.g., temperature, movement, etc. It provides solutions to many challenging problems such as wildlife, battlefield, wildfire, or building safety monitoring. A key component in a WSN is the *sensor mote*, which contains (a) a simple microprocessor, (b) application-specific sensors, and (c) a wireless transceiver. Each sensor mote is typically powered by batteries, making energy consumption an issue.

A major application for a wireless mote is to measure environmental values using embedded sensors, and transmit sensed data to a remote repository or a remote server. Because of limited transmission capabilities, this often requires multi-hop forwarding of messages, and is power consuming.

One specific power-saving mechanism used in wireless sensor networks is *data aggregation* [2–8]. Our paper

proposes a novel method for eliminating duplicate encrypted data during aggregation without decryption. Data aggregation [9–16] has been put forward as an essential paradigm in sensor networks. The aggregator uses specific functions, such as addition, subtraction or exclusive-or, to aggregate incoming readings, and only aggregated result are forwarded [17–23]. Therefore, communication overhead can be reduced by decreasing the number of transmitted packets [24–30]. Without encryption, adversaries can monitor and inject false data into the network. Encryption can solve this problem, but how can we aggregate over encrypted data [31]?

Adversaries can use the following attacks:

- Adversaries can deploy sensors near existing sensors to determine their likely value.
- Adversaries can use common key encryption systems (which always encrypt common sensor data in the same way) to see when two readings are identical. By using nearby sensors under the adversaries' control, adversaries can conduct a *known-plaintext attack*.
- Adversaries can tamper with sensors to force them to predetermined values (such as heating a temperature sensor) and thus conduct a *chosen-plaintext attack*.
- Adversaries can inject false readings or resend logged readings from legitimate sensor motes to manipulate the data aggregation process, conducting a *man-in-the-middle attack*.

Table 1 presents encryption policies, possible attacks, and vulnerabilities in data aggregation schemes.

This paper proposes a new method for determining and eliminating duplicate data while protecting privacy (using encryption) without excessive key-management or power management issues. Our scheme has the following contributions. First, we provide a lightweight data aggregation mechanism which protects data when data are processed in aggregators. Aggregators can help to eliminate redundant data without decrypting data. Thus, aggregators do not need to spend extra power in data decryption, and more network lifetime can be guaranteed. Second, our proposed scheme is resilient to known-plaintext attacks, chosen-plaintext attacks, ciphertext-only attacks, and man-in-the-middle attacks. The rest of the paper is organized as follows: Sect. 2 provides background on related work. In Sect. 3, we describe our system architecture and proposed aggregation protocol.Security analysis and performance evaluation are given in Sects. 5 and 6 offers conclusions and future directions.

2 Related work

Previous work in data aggregation assumes that every mote is honest and only transmits their correct readings. Intanagonwiwat, Govindan, Estrin, and Heidemann [17] proposed a data-centric diffusion method to aggregate data. Their method enables diffusion to achieve energy savings by selecting empirically good paths and by caching and processing data in-network. Though their method can achieve significant energy savings, security is not put into consideration in their design.

Hu and Evans [13] further examined the problem that a single compromised sensor mote can render the networks useless, or worse, mislead the operator into trusting a false reading. They proposed an aggregation protocol that is resilient to both intruder devices and single device key compromises, but their scheme suffers a problem that the aggregated data will be expanded every time when it was aggregated and forwarded by any intermediate sensor mote.

Przydatek et al. [30] proposed a secure information aggregation protocol to answer queries over the data acquired by the sensors. In particular, their proposed protocols are designed especially for secure computation of the median and the average of the measurements, for the estimation of the network size and for finding the minimum and maximum sensor reading. Even though their scheme provided data authentication to provide secrecy, the data is still delivered in plaintext format which provides no privacy during transmission.

Table 1 Encryption plicies, attacks and vulnerabilities in data aggregation schemes

Encryption policy	Possible attacks	Secrecy	Privacy	Data aggregation
Sensors transmit readings without encryption	Man-in-the-middle	No	No	Generating wrong aggregated results
Sensors transmit encrypted readings with permanent keys	Known-plaintext attack Chosen-plaintext attack Man-in-the-middle	Yes	No	Data aggregation cannot be achieved when data are encrypted unless the aggregator has encryption keys
Sensors transmit encrypted readings with dynamic keys	None of above	Yes	Yes	Data aggregation cannot be achieved when data are encrypted unless the aggregator has encryption keys

Wagner [14] presented a paper studying related attacks on data aggregation in sensor networks. He thoroughly examined current aggregation functions and proved that these aggregation functions are vulnerable and insecure under several attacks. He also proposed a theoretical framework for evaluating data aggregation resiliently in sensor networks and in its security against these attacks. Still privacy is not guaranteed in his scheme.

Acharya and Girao [2] proposed an end-to-end encryption algorithm supporting operations over ciphertexts for wireless sensor networks. Their scheme uses a special class of encryption functions, *Privacy Homomorphisms (PH)* [32–35, 16], that allow end-to-end encryption and provide aggregation functions that are applied to ciphertexts. PH is an encryption transformation that allows direct computations on encrypted data. Two functions E and D are additively homomorphic encryption and decryption if the following property is satisfied: for plaintext operands x and y and key k, $x + y = D_k(E_k(x) + E_k(y))$. However, privacy homomorphisms have exponential bound in computation. It is too computationally expensive to implement in wireless sensor networks. Moreover, it has been proved that privacy homomorphism is insecure even against ciphertext only attacks which are commonly encountered in wireless sensor networks.

Cam et al. [6] proposed a secure energy-efficient data aggregation (*ESPDA*) to prevent redundant data transmission in data aggregation. Unlike conventional techniques, their scheme prevents the redundant transmission from sensor motes to the aggregator. Before transmitting sensed data, each sensor transmits a secure pattern to the aggregator. The secure pattern is generated by associating original data with a random number. Instead of transmitting "real" data, the sensor mote transmits the secure patter to the cluster-head before transmitting it. The cluster-head then uses these secure patterns to check which sensors have same readings. Then, the cluster-head notifies certain sensor motes to transmit their data. Only sensors with different data are allowed to transmit their data to the cluster-head. However, since each sensor at least needs to transmit a packet containing a pattern once, power cannot be significantly saved. In addition, each sensor mote uses a fixed encryption key to encrypt data; data privacy cannot be maintained in their scheme.

Perrig and Tygar [36] proposed several secure broadcast schemes suitable for wireless sensor networks. The computation overhead for their schemes is affordable for tiny sensor motes. They proposed a hashed key-chain scheme to sequentially generate encryption/decryption keys for sensor motes without notifying others. Przydatek, Song and Perrig [30] further extended these schemes and proposed a secure data aggregation scheme for sensor network. Their scheme provided an efficient random sampling mechanisms and interactive proofs to enable the querier to verify that the answer given by the aggregator is a good approximation of the true value, even when the aggregator and some sensor motes were compromised.

3 Problem statement and proposed data aggregation

Data aggregation uses primitive functions, such as mean, average, addition, subtraction, and exclusive or to eliminate identical readings, and only unique results are be forwarded, reducing the cost of data transmission.

Figure 1 depicts an overview of data aggregation flow.

3.1 Proposed data aggregation method

3.1.1 Architecture

There are two commonly used network topologies in sensor networks. One is the *self-organized sensor network* (Fig. 2). A self-organized network is a multi-hop, temporary autonomous system composed of sensor motes with wireless transmission capability. It is easy to form such networks but every mote in such networks consumes significant amounts of power in data transmission as each node must spend power to transmit/forward data to other sensor nodes because of the dynamic network topology. The other network topology is the *clustered sensor network* (Fig. 3). In this architecture, the entire network is partitioned into non-overlapping clusters. Each cluster has an *aggregator* (or *cluster head)* to receive readings from other sensor motes and to forward these readings to the remote server. To extend operation lifetime, we choose the clustered topology as our network architecture [29]. In a clustered sensor network, each mote temporarily belongs to a cluster, and sensors in this cluster will receive and forward data for sensors in the same cluster. Since a mote only transmits data for several motes instead of all motes, it can obviously reduce its power consumption for data transmission.

In a clustered WSN, we assume the network is divided into clusters. Each cluster owns an aggregator having a more powerful wireless transceiver that can transmit data

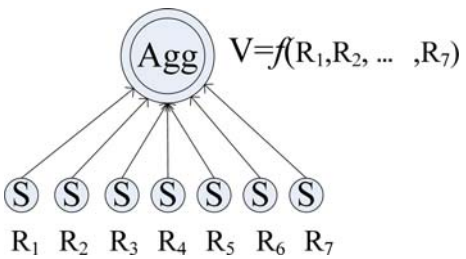

Fig. 1 Conventional data aggregation process

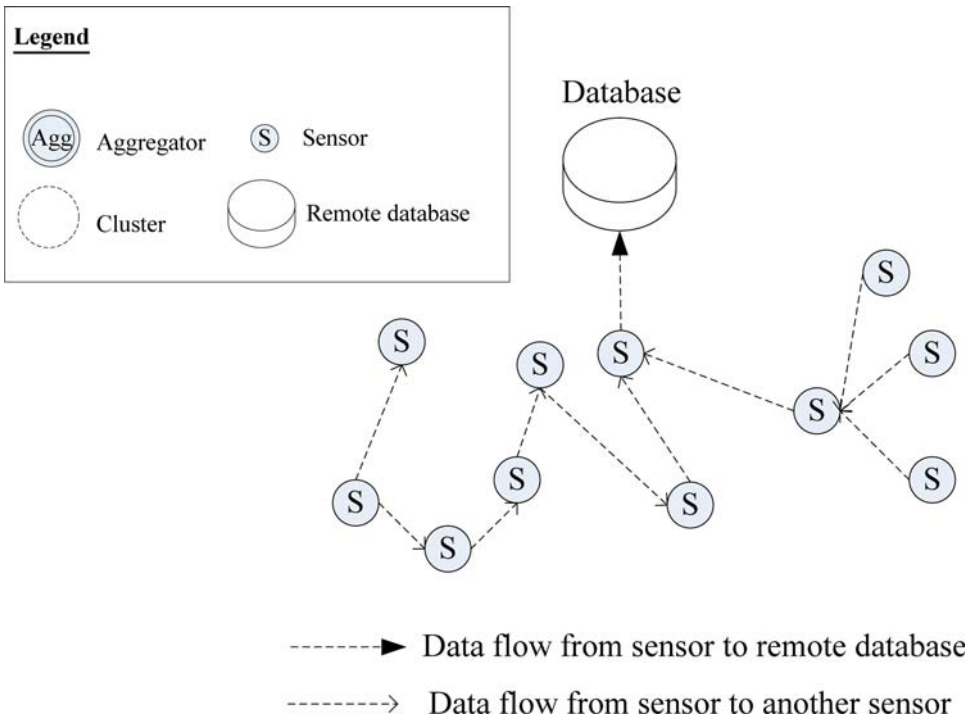

Fig. 2 Self-organized WSN architecture and its data aggregation flow

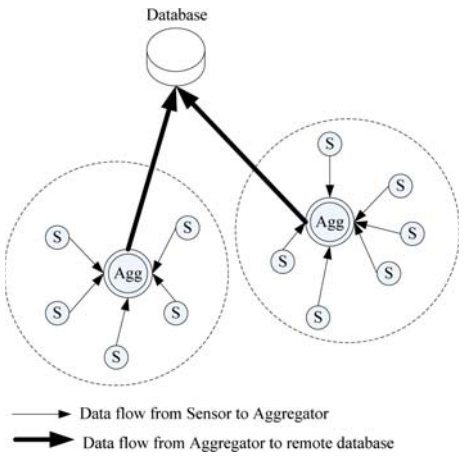

Fig. 3 A clustered sensor network topology

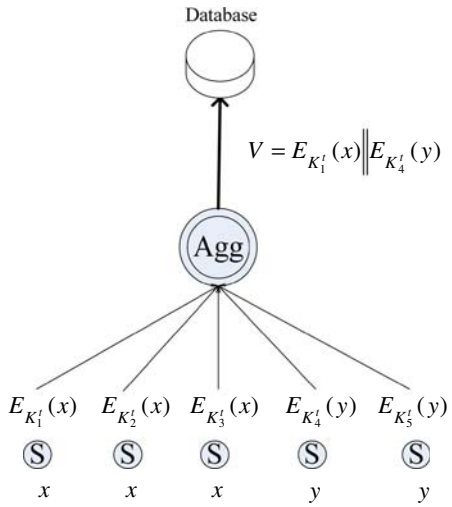

Fig. 4 Encrypted-data aggregation

directly to the backend server. In our framework, we also assume each sensor transmits data only to the aggregator; hence, each sensor mote can reduce overhead in forwarding data packets. We also assume sensor motes have no mobility, i.e., they are fixed in a position and will not be moved forever. The question of how to best deploy sensor motes and how to cluster these sensor motes is interesting to consider but is beyond the scope of this paper.

Using a clustered network to reduce power consumption, we propose a data aggregation method which maintains both secrecy and privacy. In terms of secrecy, each sensor mote encrypts its reading and transmits the encrypted data to the aggregator (Fig. 4). Adversaries will not be able to recognize what reading it is during data transmission. In terms of privacy, our design aims to eliminate redundant reading for data aggregation but this reading remains secret to the aggregator, i.e., the aggregator cannot know anything about these readings. Besides, our design can also prevent known-plaintext attacks, chosen-plaintext attacks and ciphertext-only attacks.

3.1.2 System setup

Before deploying a wireless sensor network, we have to set up three roles: the sensor mote, the aggregator, and the remote database.

1. The sensor mote: each sensor mote i is assigned an one-way function g, and a *verification key* K_i^{VK}.
2. The aggregator: the aggregator is given the one-way function g, and all $K_i^{VK} \oplus K_{i+1}^{VK}$ $\forall i$. Hereafter, these keys are referred as *aggregation verification keys*.
3. The remote database: The remote database needs to decrypt aggregated data, and thus we need to store the one-way hash function f, the one-way function g, and all verification key K_i^{VK} for all i.

Necessary keys, identities, and functions are pre-distributed in the sensor mote, the aggregator, and the remote database before they are physically deployed and used. Table 2 lists all pre-installed elements in individual roles.

Key pre-distribution is a scheme where keys are distributed among all sensor motes prior to deployment. Our proposed key pre-distribution scheme does not rely on prior deployment knowledge. Sensor motes are installed with random keys for encryption. These encryption keys have no mandatory relations between each other, and this makes system setup more flexible.

Random keys can be generated by using random source of data, such as values based on CPU clock, radioactive decay, or atmospheric noise. The question of how to generating random numbers is interesting to consider but is beyond the scope of this paper.

3.1.3 Proposed scheme

There are two phases in our proposed scheme: *data encryption phase* and *data aggregation phase*. The encryption phase provides a lightweight encryption algorithm that supports data aggregation property, and provides secrecy and privacy for data transmission. The data aggregation phase provides a method to eliminate redundant readings from sensor motes without decrypting them. Since the aggregator cannot decrypt incoming packets, the aggregator cannot know anything about the plaintext, and therefore more power can be saved.

Data encryption phase Our encryption design aims to provide lightweight encryption overhead and secrecy while providing data aggregation property.

When a sensor mote i has a reading m_i and wishes to transmit this reading to the aggregator, it first randomly generates a new key K_i^{EK}, which will be used as the next-round encryption key. By using g, K_i^{EK}, and K_i^{VK}, the corresponding ciphertext $E_i(m_i)$ is defined in Eq. 1.

$$E_i(m_i) = m_i \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK} \| K_i^{EK} \oplus K_i^{VK}, \tag{1}$$

where $\|$ indicates data concatenation.

Our proposed scheme is very close to the one-time pad method [37] as each mote changes to a different key for encrypting data but provides more capabilities. It is obvious that the length of data is required to be at least as long as the length of encryption key in our proposed scheme. When the length of data is shorter than the length of the key, extra padding must be appended to the data so that the appended data can be encrypted. As the message m_i is xored with $g\left(K_i^{EK}\right) \oplus K_i^{EK}$, it does not matter if we pad random values or fixed values (e.g., all 0's or 1's). It does not reduce any security strength in our scheme.

Next, we will introduce how to find out redundant readings among these ciphertexts without decrypting them in our data aggregation phase.

Data aggregation phase Our data aggregation method provides a pair-wise method to identify if two readings are identical. Although the goal of our data aggregation scheme is to find redundant readings among n incoming encrypted packets in the aggregator, our aggregation scheme can be further extended by pairing off these n incoming encrypted packets. By iteratively performing pair-wise comparisons we can eliminate all redundant readings among them. If n same readings are encrypted and transmitted to the aggregator, the aggregator needs to check $n - 1$ times to verify these inputs and save $n - 1$ packet transmission. It needs computation overhead for data aggregation but saves more energy from fewer data transmissions.

In the following section, first we will introduce our approach to find redundant readings in two packets; then, we will introduce how to extend our approach to find redundant readings among n packets.

Assume sensor mote i and j sends two encrypted readings to the aggregator, and these encrypted readings can be expressed by the following equations:

$$E_i(m_i) = m_i \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK} \| K_i^{EK} \oplus K_i^{VK}, \tag{2}$$

$$E_j(m_j) = m_j \oplus g\left(K_j^{EK}\right) \oplus K_j^{EK} \| K_j^{EK} \oplus K_j^{VK}. \tag{3}$$

First, the aggregator XOR the first parts of these two ciphertexts, and it can be expressed by the following equation:

$$m_i \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK} \oplus m_j \oplus g\left(K_j^{EK}\right) \oplus K_j^{EK} \tag{4}$$

Next, since the aggregator is pre-installed with $K_i^{VK} \oplus K_{i+1}^{VK} \forall i, K_i^{VK} \oplus K_j^{VK}$ can be obtained by $\left(K_i^{VK} \oplus K_{i+1}^{VK}\right) \oplus \left(K_{i+1}^{VK} \oplus K_{i+2}^{VK}\right) \oplus \cdots \oplus \left(K_{j-1}^{VK} \oplus K_j^{VK}\right)$, the aggregator can XOR the last two parts of Eqs. 2 and 3 to obtain:

Table 2 Pre-installed elements in three roles

Role	Pre-installed elements
The sensor mote	SID_i, g, and K_i^{VK}
The aggregator	g, and $K_i^{VK} \oplus K_{i+1}^{VK}$ $\forall i$.
The remote server	g, and K_i^{VK} $\forall i$

$$K_i^{EK} \oplus K_i^{VK} \oplus K_j^{EK} \oplus K_j^{VK} \oplus K_i^{VK} \oplus K_j^{VK} = K_i^{EK} \oplus K_j^{EK} \quad (5)$$

It can be found that the aggregator can use $E_i(m_i)$ and $E_j(m_j)$ to retrieve $K_i^{EK} \oplus K_j^{EK}$, but cannot retrieve K_i^{EK} or K_j^{EK} separately; therefore, the aggregator cannot decrypt $E_i(m_i)$ and $E_j(m_j)$.

Next, we define a *check value* V, and V is calculated by XOR Eqs. 4 and 5 and $g\left(K_i^{EK} \oplus K_j^{EK}\right)$. The check value is used to distinguish if two encrypted readings are the redundant in their plaintext format. As a result, the check value V can be expressed by the following equation:

$$V_{(i,j)} = m_i \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK} \oplus m_j \oplus g\left(K_j^{EK}\right) \oplus K_j^{EK} \oplus K_i^{EK} \oplus K_j^{EK} \oplus g\left(K_i^{EK} \oplus K_j^{EK}\right) \quad (6)$$

By using the properties of function g, Eq. 6 can be further reduced to:

$$\begin{aligned} V_{(i,j)} &= m_i \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK} \oplus m_j \oplus g\left(K_j^{EK}\right) \\ &\quad \oplus K_j^{EK} \oplus K_i^{EK} \oplus K_j^{EK} \oplus g\left(K_i^{EK} \oplus K_j^{EK}\right) \\ &= m_i \oplus m_j \end{aligned} \quad (7)$$

It is easier observed that if m_i is equal m_j, $V_{(i,j)} = m_i \oplus m_j = 0$, and vice versa. We can formally describe $V_{(i,j)}$ by the following equations:

$$if \begin{cases} V_{(i,j)} = 0, then \quad m_i = m_j \\ V_{(i,j)} \neq 0, otherwise \end{cases} \quad (8)$$

Figure 5 depicts the data aggregation phase.

If these two readings are the same, the aggregator just needs to send either $E_i(m_i)$ or $E_j(m_j)$ to the remote server. If these two readings are different, the aggregator then sends

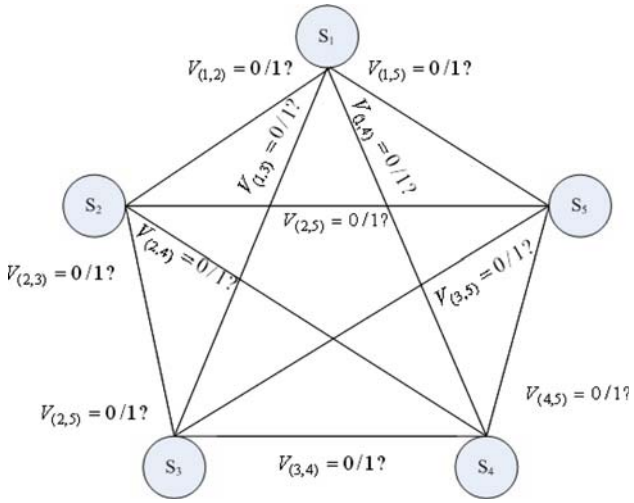

Fig. 6 Data aggregation verification steps for $n = 5$

$E_i(m_i)\|E_j(m_j)$ to the remote server. Since remote server is pre-installed with the verification key K_i^{VK}, the remote server therefore can use K_i^{VK} to obtain $K_i^{EK}$by: $K_i^{EK} = (K_i^{EK} \oplus K_i^{VK}) \oplus K_i^{VK}$. Then, the original data m_i can be recovered by: $m_i = \left(m_i \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK}\right) \oplus g\left(K_i^{EK}\right) \oplus K_i^{EK}$. In above case, the aggregator only needs to examine two incoming ciphertexts, but in general cases, the aggregator usually receives more than two incoming ciphertexts. When the aggregator receive n $(n > 2)$ incoming ciphertexts $(E_1, E_2, \ldots, E_n)$, our proposed scheme can be easily extended. First, we group these ciphertexts into pairs, i.e., $\left(E_i, E_j\right)\forall i$. Then, we can repeat above steps to generate their check value V. Next, we can use V to check if E_i has the same reading with E_j. Finally, if $V_{(1,2)} = V_{(2,3)} = \cdots = V_{(n-1,n)}$, then we can conclude that $E_1, E_2, \ldots, E_n$ has the same reading. Figure 6 depicts necessary comparisons for data aggregation when $n = 5$. It can be observed that

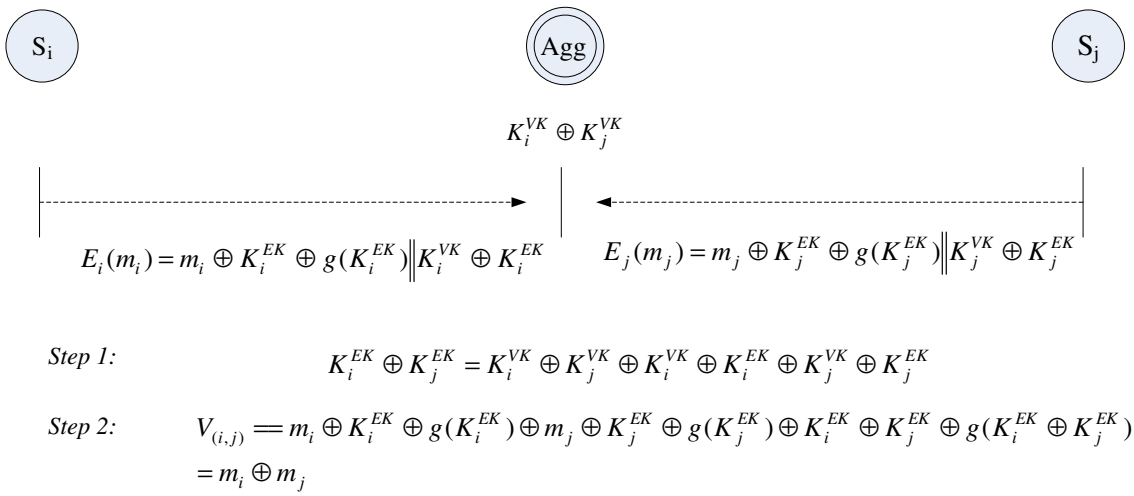

Fig. 5 Data aggregation phase

these comparisons can be viewed as all edges in a complete graph, and we will discuss this property in next section.

Preliminaries: When the aggregator receives n encrypted readings, the minimum number of comparisons is $n-1$ under the condition that all these readings (when unencrypted) are the same. The maximum number of comparisons is $\frac{n(n-1)}{2}$ when all these readings (when unencrypted) are totally different from each other.

4 Threat models

The goals of the adversaries are to read, insert, and even modify sensor readings. We consider several possible threats, classified according to the capabilities of the adversaries.

4.1 Known-plaintext attacks

To implement known-plaintext attacks, no capabilities are need except the ability to deploy malicious sensors close to legitimate sensors. In this scenario, an adversary can

- Collect all readings from all sensors, calculated aggregated values, know their routing paths, and inject wrong readings or aggregated values to the network.
- Collect abundant encrypted readings to enhance the compromise of encryption keys.

In practice, known-plaintext attacks can be easily achieved by deploying same sensor very close to legitimate sensors. The goal of these attacks is merely to read readings and to record corresponding responses of a sensor mote.

4.2 Chosen-plaintext attacks

- Adjust the sensors by changing physical conditions, such as temperature or moisture.
- Log all plaintext-ciphertext mappings without knowing what the encryption keys are.

In practice, adversaries can take some physical methods to adjust the sensing environment in order to make sensor motes generate false readings the adversaries desired. For example, adversaries can use heaters to raise the temperature to a certain degree, and temperature sensors will send the false temperature readings making the aggregators generate incorrect results.

4.3 Man-in-the-middle attacks

- Read, insert, or modify messages between sensor motes.
- Inject false readings or resend logged readings on behalf of legitimate sensor motes to malfunction data aggregation.

Significantly, we assume that an adversary cannot retrieve encryption keys from a sensor mote by physically compromising it. Otherwise, there will be no security at all.

5 Security analysis and performance evaluation

In this section, we evaluate our proposed scheme according to two aspects: theoretical and practical. In theoretical aspect, we use random oracle model to justify our protocol is secure in terms of provable security. We firstly built an ideal random oracle model and show that our proposed encryption algorithm is an implementation of the ideal random oracle. Then, we use the random oracle model to justify that it can resist know-ciphertext attacks. In practical aspect, we estimate necessary time for compromising our proposed scheme using different key lengths. The result shows that using encryption keys longer than 80 bits would be considerable secure enough even if the adversary uses 1,000,000 4 GHz PCs running simultaneously to compromise our scheme. Then, we show that our proposed scheme can resist known-plaintext attacks, chose-plaintext attacks, and know-ciphertext attacks.

Before we proceed to theoretical proof, we first describe the security requirement specifying the adversary's abilities and when the latter is considered successful. The abilities and disabilities of the adversary include:

- The adversary has an arbitrary polynomial-time computation power.
- The adversary can eavesdrop on messages in the air.
- The adversary can know the original readings of any sensor.
- The adversary cannot access the encryption keys.

An attack is considered to be successful if the adversary can compromise the encryption keys. In terms of system security, we adopt the idea in [38]. A system is considered secure if any adversary with the given abilities has only a negligible probability of success.

A random oracle is a theoretical black box that replies to queries with random response chosen uniformly in its output domain. A methodology for designing a cryptographic protocol can be divided into two steps. In first step, one designs an ideal system in which all participants as well as adversaries have oracle access to a truly random function, and proves the security of the ideal system. In second step, we replace the random oracle by a "good cryptographic hashing function". We can therefore obtain an implementation of the ideal system in a real-word where

random oracles do not exist. This methodology is referred to as the random oracle methodology.

Before we build our ideal system, we first describe the notion.

$\{0, 1\}^*$	the space of finite binary strings
$\{0, 1\}^\infty$	the space of infinite binary strings
$G : \{0,1\}^* \rightarrow \{0,1\}^\infty$	a random generator
f	a trapdoor permutation with inverse f^{-1}
k	the security parameter
$H : \{0,1\}^* \rightarrow \{0,1\}^k$	a random has function
$G(r) \oplus x$	the bitwise XOR of x with the first $\|x\|$ bits of the output of $G(r)$

5.1 Preliminaries

Definition A function $\varepsilon(k)$ is negligible if for every c there exists a k_c satisfying $\varepsilon(k) \leq k^{-c}$ for every $k \geq k^{-c}$.

Definition If A_P is a probabilistic algorithm, then for any inputs m_1, m_2,..., $A_P(m_1, m_2,\ldots)$ is the probability space which to the sting σ assigns the probability that A_P outputs σ. For probabilistic spaces S, T,..., $P_r[x \leftarrow S; y \leftarrow T;\ldots:p(x, y,\ldots)]$ denotes the probability that the predicate $p(x, y,\ldots)$ is true after the execution of the algorithms $x \leftarrow S, y \leftarrow T$, etc.

Definition A random oracle R is a map from $\{0, 1\}^*$ to $\{0, 1\}^\infty$ chosen by selecting each bit of $R(x)$ uniformly for every x.

Without lost of generosity, our proposed scheme can be formulated as the following oracle:

$$E_r^G(m) = m \oplus G(r)\|f(r)\ldots \tag{9}$$

5.2 Known-plaintext security

For known-plaintext attacks, the adversary knows some m, and $P_r[\text{The attacker successfully guesses } G(r_1)]$ can be described as:

$P_r[r_1 \leftarrow G(r)] = \frac{1}{2^{|r_1|}} \approx 0$, when r_1 is large enough.

We suggest that $|r_1| \geq 88$ is adequate and mathematical induction will be given later.

5.3 Chosen-plaintext security

We adapt the notion of CP-adversary (chosen-plaintext adversary) in [39] to the random oracle model. A CP-adversary A is a pair of non-uniform polynomial algorithms (F, A_1), each with access to an oracle. For an encryption algorithm ϑ to be secure, it requires that

$$\begin{aligned} &P[\text{Chosen} - \text{Plaintext Fails}] = P_r R \leftarrow 2^\infty; (E,D) \leftarrow \vartheta\left(1^k\right); \\ &(m_0,m_1) \leftarrow F^R(E); b \leftarrow \{0,1\}; \sigma \leftarrow E^R(m_b): \\ &A_1^R(E,m_0,m_1,\alpha) = b \leq 0.5 + k^{-w(1)}. \end{aligned} \tag{10}$$

Proof The proof is by contradiction. Let $A = (F, A_1)$ be an adversary that defeats our protocol. Often, the adversary gains advantage $\lambda(k)$ for some inverse polynomial λ. We construct an algorithm $M(f, d, y)$ that, when $(f, f^{-1}, d) \leftarrow \vartheta(1^k); r \leftarrow d(1^k); y \leftarrow f(r)$, manages to compute $f^{-1}(y)$. It simulates the oracle G and samples $(m_0, m_1) \leftarrow F^G(E)$. If G is asked an r such that $f(r) = y$, then Moutputs r and halts; otherwise, the $F(E)$ terminates and M chooses $\alpha \leftarrow y||s$ for $s \leftarrow \{0,1\}^{|m_0|}$. Then M simulates $A_1^G(E, m_0, m_1, \alpha)$, watching the oracle queries that A_1 makes to see if there is any oracle query r for which $f(r) = y$. Let A_k be the event that A_1 does not ask for the image of G at r. It satisfies that

$1/2 + \lambda(k) = P_r[\text{A succeeds } |A_K] \cdot P_r[A_k] + P_r$
$[\text{A succeeds } |\overline{A_k}] \cdot P_r[\overline{A_k}]$.

Thus, Eq. 10 is satisfied.

5.4 Chosen-ciphertext security

The chosen-ciphertext attack is defined as: the adversary can adaptively choose ciphertexts and access to the decryption algorithm to get the corresponding plaintexts. Though it is usually occurred in asymmetric cryptographic systems, it can also be happened in our scheme as the adversary can know both ciphertexts and plaintexts (by using same sensors) in the same time. We adapt the definition of [39, 40] to the random oracle [38] setting. An RS-adversary ("Rackoff-Simon adversary") A is a pair of non-uniform algorithms $A = (F, A_1)$, each with access to an oracle R and a black box implementation of D^R. The algorithm F is used to generate two messages m_0 and m_1 such that if A_1 is given the encryption α, A_1 won't be able to guess well whether α comes m_0 or m_1. Formally, an encryption scheme ϑ is secure against RS-attack if the following equation is satisfied:

$$\begin{aligned} &P[\text{Chosen} - \text{Ciphertext Fails}] \\ &= P_r[R \leftarrow 2^\infty; (E,D) \leftarrow \vartheta(1^k); (m_0,m_1) \\ &\quad \leftarrow F^{R,D^R}(E); b \leftarrow \{0,1\}; \alpha \leftarrow E^R(m_b) : \\ &\quad A_1^{R,D^R}(E,m_0,m_1,\alpha) = b] \leq 0.5 + k^{-w(1)} \end{aligned} \tag{11}$$

Proof To see our scheme is secure against chosen ciphertext attacks, we prove the above equation is satisfied. Let A_k denotes the event that $a\|b \leftarrow F(E)$, for some a and b. Let $A = (F, A_1)$ be an RS-adversary that succeeds with probability $\frac{1}{2} + \lambda(k)$ for some non-negligible function $\lambda(k)$. The adversary A can make some oracle call of $G(r_1)$ or $H(G(r_1))$. Let L_k denotes the event that A_1 asked $D^{G,H}$ some queries where $a = m \oplus f^{-1}(r_1) \oplus H(f^{-1}(r_1))$, but A_1 never asked its H-oracle for $H(f^{-1}(r_1))$. Let $n(k)$ denotes the total number of oracle queries made. It is easy to see that $Pr[L_k] \leq n(k)2^{-k}$ and $Pr[\text{A succeeds } \bar{L}_K \cap \bar{A}_K] = 0.5$ according to [39].

Thus $P_r[\text{A succeeds}] = P_r[\text{Choosen} - \text{Cipher Attack succeeds}] = \frac{1}{2} + \lambda(k)$ is bounded above by

$$P_r[\text{A succeeds } L_K]P_r[L_k] + P_r[\text{A succeeds } |\bar{L}_k \cap A_k]P_r[\bar{L}_k \cap A_k] + P_r[\text{A succeeds } |\bar{L}_k \cap \bar{A}_k]P_r[\bar{L}_k \cap \bar{A}_k] \leq n(k)2^{-k} + P_r[A_k] + \frac{1}{2}.$$

Therefore, our proposed scheme satisfies Eq. 11, and is chosen-ciphertext-attack resistant.

In practice aspect, we evaluate the difficulties to brute force our proposed scheme. To brute force our proposed scheme, first the adversary need to spend time generating all possible keys and test the result with every possible key. We assume that the adversary can generate an encryption key and test the result in one duty cycle, our proposed scheme uses σ-bit keys to encrypt data, and the adversary uses a η G-Hz PC to brute force our propose scheme. To completely test all possibilities by exhaustive search, the adversary would need to spend

$$\frac{2^{\sigma}(cycles)}{\eta * 10,000,000 * 86400 * 365}$$

years to compromise our scheme.

Assume the adversary uses a 4G-Hz PC to brute force our scheme which uses a 64-bit encryption key, the adversary needs to generate all 2^{64} keys and uses these keys to test the result. If we assume that the adversary can test our encryption scheme within one duty cycle, the total computation time to test all 2^{64} keys is:

$$2^{64}/4G/86400/365 \approx 14,624 \text{ years.}$$

However, the adversary can use more PCs simultaneously to compromise our algorithm. If the adversary uses 1,000,000 PCs running simultaneously to compromise our scheme, the total computation time to test all conditions is

$$2^{64}/4G/86400/365/1M \approx 0.01 \text{ years.}$$

In this case, it takes about 3–4 days to compromise our scheme which is unacceptably insecure. Table 3 lists estimated time to brute force our proposed scheme with different key lengths. To maintain acceptable security while using minimal key length, we suggest use 80-bit keys to encrypt data as the adversaries need about 958 years to compromise our scheme even if they use 1,000,000 PCs to attack our scheme in parallel.

Moreover, using longer encryption keys can dramatically increase difficulties to compromise our scheme as it exponentially expend the key space which makes adversaries spend more time to brute force the proposed scheme. Figure 7 illustrate the growth rate of key size (2^{σ}) and the growth rate (η) of PCs.

Assume the adversary know sensor reading m and corresponding ciphertext $E_i(m_i) = m_i \oplus g(K_i^{EK}) \oplus K_i^{EK} \| K_i^{EK} \oplus K_i^{VK}$, the adversary can therefore know $g(K_i^{EK}) \oplus K_i^{EK}$ and $K_i^{EK} \oplus X_i^{VK}$. Without knowing K_i^{VK} in advance, the adversary cannot compromiseK_i^{EK}. Furthermore, since the encryption keys will be arbitrarily changed, our scheme can hence resist known-plaintext attacks. Even adversary can generate designate data m to confuse sensor motes, still the adversary cannot learn anything about the encryption keys. Therefore, our scheme can resist know-ciphertext attacks and chosen-ciphertext attacks.

One workload we have to pay is the number of comparisons it takes to verify encrypted-data from n motes. Our proposed scheme can reduce the number of comparisons as it has transitive property. The transitive property is described as: Given $E_h(m_h)$, $E_i(m_i)$, and $E_j(m_j)$, if $V_{(h,i)} = 0$ and $V_{(i,j)} = 0$, then $V_{(h,j)} = 0$. This is pretty simple to prove. If $V_{(h,i)} = 0$ and $V_{(i,j)} = 0$, then $m_h = m_i$ and $m_i = m_j$ It can therefore be easily seen that $m_h = m_i = m_j$.

With this transitive property, if all readings are the same, the minimum number comparisons for verifying data from n sensor motes is $n - 1$. And, according to Fig. 6, the maximum number comparisons for verifying data from n

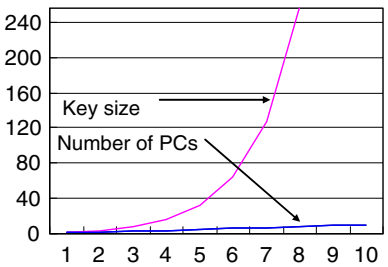

Fig. 7 The growth rate of key size (2^{σ}) and the growth rate (η) of PCs

Table 3 Estimated time (years) to brute force our proposed scheme with different key lengths

Key length	One 4 GHz PC	10,000 4 GHz PCs	100,000 4 GHz PCs	1 M 4 GHz PCs
64 bits	14624	1.5	0.15	0.01
72 bits	374363	374.4	37.4	3.74
80 bits	958369660	95837	9583.7	958.37
88 bits	2.45E+11	24534263	2453426.3	245342.6

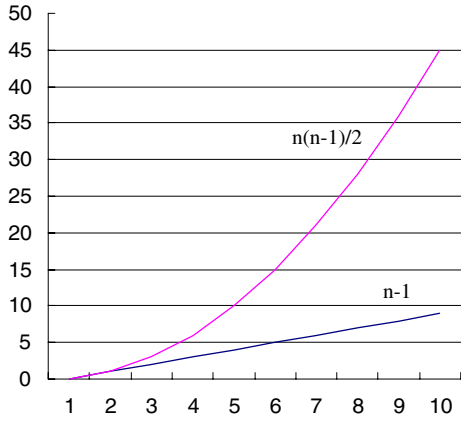

Fig. 8 The number of comparisons for verifying n encrypted-data

sensor motes is equal to the number of edges in a n-complete graph which is $\frac{n(n-1)}{2}$. It is shown in Fig. 8 that our computation bound is limited between $O(n)$ and $O(n^2)$, and this can be affordable for off-the-shelf sensor platforms.

In comparison with other schemes, our encryption algorithm uses XOR and a hash function. Our encryption algorithm is more lightweight. Our proposed encryption algorithm changes its encryption key whenever there's a reading that needs to be transmitted. This makes our scheme more feasible for wireless sensor networks. Table 4 lists the differences between our scheme and other schemes.

Key Compromise One major issue in our scheme is the key compromise problem. As the aggregator stored $K_i^{VK} \oplus K_{i+1}^{VK}\ \forall i$, once an encryption key K_i^{VK} is been compromised, all other encryption key $K_j^{VK}\ \forall j \neq i$ will be compromised. Therefore, the aggregator must have stronger security protection than sensor motes. One way to enhance the hardware security strength in the aggregator is to install a TPM (Trust Platform Module) chip inside the sensor mote, and all paired encryption keys $K_i^{VK} \oplus K_{i+1}^{VK}\ \forall i$ are stored inside TPM. It can significantly reduce the possibility that adversaries compromise the aggregators.

Data Size Variation Here we discuss the storage requirement when the length of data is increased. When the length of data is increased, the encryption key must be increased correspondingly. Assume the length of data is increased by l', the length of key as well will increase l' bits. As each sensor mote stores K_i^{VK} only, it requires more l' bits to store the encryption key. For the aggregator, as the aggregator stores all paired encryption keys $K_i^{VK} \oplus K_{i+1}^{VK}\ \forall i$, it requires more $\theta \cdot l'$ bits where θ is the number of sensor motes in the cluster. It can be seen that when the length of data is increased linearly, the storage requirement for storing keys is also increased linearly.

Efficiency Here we discuss the efficiency caused from our proposed scheme. Our proposed saves power by eliminating redundant packets. Thus, the more packets are eliminated, the more power can be saved. As we mentioned earlier, the minimum number of comparisons is n-1 under the condition that all these readings (when unencrypted) are the same, amd the maximum number of comparisons is $\frac{n(n-1)}{2}$. For the best case, it reduces (n-1) packet transmissions. For the worst case, it does not reduce any packet transmission overhead. For average case, assume that there totally n packets and m of them are the same, the number of comparisons is $(m-1)+(n-m)(n-m-1)/2$. It reduces $n-m$ packets in average case. Table 5 lists the efficiency comparisons for the best, average, and the worst cases.

6 Conclusion and future directions

In this paper, we proposed a secure encrypted-data aggregation scheme for wireless sensor networks. Our scheme has the following enhancements: (1) the aggregator does

Table 4 Performance evaluations compared with other schemes

	Our proposed scheme	Flooding-base scheme	Privacy homomorphism-based scheme
Encryption	Lightweight	Heavyweight	Heavyweight
Encryption key	Easy to change, and always changes	Only one encryption key, and is hard to change	Only one encryption key, and is hard to change
Decryption (in aggregator)	No	Yes	Yes
Aggregated result	Only one data	Many redundant data	Only one data

Table 5 Efficiency comparisons for the best, average, and the worst case

	Best case	Average case	Worst case
Number of comparison	$n-1$	$(m-1)+(n-m)(n-m-1)/2$	$n(n-1)/2$
Packet eliminated	$n-1$	$n-m$	0

not need to decrypt its received encrypted-data to verify if these data are the same; no extra power are wasted in data decryption, (2) the aggregator does not have decryption keys and therefore cannot know anything about the data, and (3) our proposed scheme uses random keys to encrypt data; this property makes our scheme resilient to known-plaintext attacks, chosen-plaintext attacks, ciphertext-only attacks, and man-in-the-middle attacks. Moreover, compared with conventional PH-based data aggregation schemes, received data can be recovered and decrypted to be further analyzed. Our proposed scheme provides secrecy and privacy in the sense that each sensor mote randomly generates a new encryption key each time providing semantic security for data encryption phase proposed data aggregation, and the intermediate aggregators cannot decrypt these encrypted-data. Aiming at secrecy and privacy, our proposed scheme is resilient to several attacks in sensor networks, and makes data aggregation more practical in these environments.

Our proposed scheme extends one-time pad to provide a secure encrypted-data aggregation paradigm for wireless sensor. Though it supports secrecy and privacy, our scheme provides only equality check. More general mathematical operations, such as addition, subtraction, and so on, should be further investigated under the same condition: the encryption keys are always changing and the aggregator cannot decrypt data through it. Except these mathematical operands, operands for strings, such as finding substring, should also be provided.

Currently, our scheme is workable in a one-level clustered network environment, i.e., the aggregator can one-hop to the base station. However, in real deployment, it is usually not the case. Our future work toward this problem is to extend our scheme to multi-level cluster environment. Another problem in our scheme is that our experimental sensor motes must be fixed to a cluster and can no longer be moved to another cluster. We will also address this issue in our future work.

For key management, our proposed scheme pre-installs keys for verification and data aggregation in the aggregator before deployment. This limits the flexibility of system deployment and aggregation. In future work, we expect to modify our key management method so that these keys will not be stored in aggregators in advance but will be exchanged and retrieved when necessary. We also look forward to extending privacy homomorphism functions to support dynamic key management to bring more flexibility in data aggregation.

Our protocol uses only XOR operations and an irreversible hash function to encrypt data. The security strength is not as strong as block cipher encryption algorithms, such as AES, DES, etc. We also expect to extend our scheme to adopt block-cipher encryption algorithms to provide higher security strength for aggregation.

References

1. Akyildiz, I., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications Magazine, 40*, 102–114. doi:10.1109/MCOM.2002.1024422.
2. Acharya, M., & Girao, J. (2005). Secure comparison of encrypted data in wireless sensor networks. *In 3rd international symposium on modeling and optimization in mobile, ad hoc, and wireless networks* (pp. 47–53).
3. Al-Karaki, J., Ul-Mustafa, R., & Kamal, A. (2004). Data aggregation in wireless sensor networks—exact and approximate algorithms. *In proceedings of the workshop on high performance switching and routing* (pp. 241–245).
4. Atonishi, T., & Matsuda, T. (2006). Impact of aggregation efficiency on GIT routing for wireless sensor networks. *In proceedings of IEEE international conference oon parallel processing workshops*.
5. Buttyan, L., Shaffer, P., & Vajda, I. N. (2006). Resilient aggregation with attack detection in sensor networks. *In proceedings of the fourth annual IEEE international conference on pervasive computing and communications workshops* (p. 332).
6. Cam, H., Ozdemir, S., Nair, P., Muthuavinashinappan, D., & Ozgur Sanli, H. (2006). ESPDA: Energy-efficient secure pattern based data aggregation for wireless sensor networks. *Computer Communication, 29*, 446–455.
7. Chen, Y., Liestman, A., & Liu, J. (2006). A hierachical energy-efficient framework for data aggregation in wireless sensor networks. *IEEE Transactions on Vehicular Technology, 55*, 789–796. doi:10.1109/TVT.2006.873841.
8. Choi, J., Lee, J., Lee, K., Choi, S., Kwon, W., & Park, H. (2006). Aggregation time control algorithm for time constrained data delivery in wireless sensor networks. *In proceedings of vehiculare technology* (pp. 563–567).
9. Considine, J., Li, F., Kollios, G., & Byers, J. (2004). Approximate aggregation techniques for sensor databases. *In proceedings of IEEE conference on data engineering* (p. 449).
10. Gatani, L., Lo Re, G., & Ortolani, M. (2006). Robust and efficient data gathering for wireless sensor networks. *In proceedings of the 39th Hawaii international conference on system sciences* (p. 235).
11. Girao, J., Westhoff, D., & Schneider, M. (2005). CDA: Concealed data aggregation for reverse multicast traffic in wireless sensor networks. *In proceedings of 40th international conference on communications* (pp. 3044–3049).
12. Girao, J., Westhoff, D., & Scheneider, M. (2004).Concealed data aggregation in wireless sensor networks. *In proceedings of ACM WiSe conference*.
13. Hu, L., & Evans, D. (2003). Secure aggregation for wireless networks. *In proceedings of applications and internet workshops* (pp. 27–31).
14. Wagner, D. (2004). Resilient aggregation in sensor networks. *In proceedings of the 2nd ACM workshop on security of ad hoc and sensor networks* (pp. 78–87).
15. Westhoff, D., Girao, J., & Acharya, M. (2006).Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution and routing adaptation. *In proceedings of IEEE transactions on mobile computing* (pp. 1417–1431).
16. Wu, K., Dreef, D., Sun, B., & Xiao, Y. (2006). Secure data aggregation without persistent cryptographic operations in

wireless sensor networks. *In proceedings of performance, computing, and communications conference* (p. 6).
17. Intanagonwiwat, C., Govindan, R., Estrin, D., & Heidemann, J. (2003). Directed diffusion for wireless sensor networking. *In IEEE/ACM transactions on networking* (pp. 2–16).
18. Jiang, H., & Jin, S. (2006). Scalable and robust aggregation techniques for extracting statistical information in sensor networks. *In proceedings of the 26th IEEE international conference on distributed computing systems* (p. 69).
19. Krishnamachari, L., Estrin, D., & Wicker, S. (2002). The impact of data aggregation in wireless sensor networks. *In proceedings of distributed computing systems workshops*.
20. Li, H., Yu, H., & Liu, A. (2006). A tree based data collection scheme for wireless sensor network. *In proceedings of the IEEE international conference of networking* (p. 119).
21. Raina, M., Ghosh, S., Patro, R., Viswanath, G., & Chadrashekhar, T. (2006). Secure data aggregation using commitment schemes and quasi commutative functions. *In proceedings of 1st international symposium on wireless pervasive computing* (pp. 16–18).
22. Shin, S., Lee, J., Baek, J., & Seo, D. (2006). Reliable data aggregation protocol for ad-hoc sensor network environments. *In proceedings of the 8th international conference on advanced technology*.
23. Shrivastava, N., Buragohain, C., Agrawal, D., & Suri, S. (2004). Medians and beyond: New aggregation techniques for sensor networks. *In proceedings of the 2nd international conferece on embedded networked sensor systems* (pp. 239–249).
24. Li, Z., Li, K., Wen, C., & Soh, Y.(2003). A new chaotic secure communication system. *In proceedings of IEEE transactions on communications* (pp. 1306–1312).
25. Li, T., Wu, Y., & Zhu, H. (2006). An efficient scheme for encrypted data aggregation on sensor networks. *In proceedings of vehicular technology conference* (pp. 831–835).
26. Madden, S., Franklin, M. J., Hellerstein, J. M., & Hong, W. (2002). TAG: A tiny aggregation service for ad-hoc sensor networks. *In proceedings of 5th symposium on operating systems design and implementation.*
27. Mahimkar, A., & Rappaport, T. (2004). SecureDAV: A secure data aggregation and verfication protocol for sensor networks. *In proceedings of global communication*.
28. Misra, R., & Mandal, C. (2006). Ant-aggregation: Ant colony algorithm for optimal data aggregation in wireless sensor networks. *In proceedings of international conference on wireless and optical communication networks* (p. 5).
29. Moussaoui, O., Ksentini, A., Naimi, M., & Gueroui, M. (2006). Efficient energy saving in wireless sensor networks through hierarchical-based clustering. *In proceedings of the seventh IEEE international symposium on computer networks*.
30. Przydatek, B., Song, D., & Perrig, A. (2003). SIA: Secure information aggregation in sensor networks. *In proceedings of ACM SenSys conference* (pp. 255–265).
31. Chandramouli, R., Bapatla, S., & Subbalakshmi, K. P. (2006). Battery power-aware encryption. *In proceedings of ACM transactions on information and system security* (pp. 162–180).
32. Bao, F. (2003). Cryptoanalysis of a provable secure additive and Multiplicative Privacy Homomorphism. *In proceedings of the international workshop on coding and cryptography* (pp. 43–50).
33. Benaloh, J. (1986). Secret sharing homomorphisms: Keeping shares of a secret sharing. In *Advances in Cryptology—CRYPTO* (pp. 251–260).
34. Cramer, R., Damgard, I., & Nielsen, J. B. (2001). Multiparty computation from threshold homomorphic encryption. *In advances in cryptology—EUROCRYPT* (pp. 280–299).
35. Domingo-Ferrer, J. (2002). A provably secure additive and multiplicative privacy homomorphism. *In proceedings of information security conference* (pp. 471–483).
36. Perrig, A., & Tygar, J. D. (2002). *Secure boradcast communication in wired and wireless networkss*. Dordrecht: Kluwer Academic Publisher.
37. Schneider, M., & Felten, E. (2000). Efficient commerce protocols based on one-time pads. *In proceedings of 16th annual computer security applications conference* (p. 317).
38. Canetti, R., Goldreich, O., & Halevi, S. (1998). The random oracle methodology, revisited. *In proceedings of the 30th annual ACM symposium on the theory of computing* (pp. 209–218).
39. Bellare, M., & Rogaway, P. (1993). Random oracles are practical: a paradigm for designing efficient protocols. *In proceedings of 1st conferendce on computer and communications security* (pp. 62–73).
40. Rackoff, C., & Simon, D. (1991). Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. *In proceedings of advances in cryptology*.

Author Biographies

Shih-I Huang received B.S. and M.S. degrees in Applied Mathematics from National Sun-Yat Sen University, and he's working toward his Ph.D. in EECS in National Chiao Tung University. He's also currently a R&D engineer and project leader in Industrial Technology Research Institute in Taiwan. His research interests include network security, information security, wireless sensor network, data protection, and data privacy.

Shiuhpyng Shieh received the M.S. and Ph.D. degrees in Electrical and Computer Engineering from the University of Maryland, College Park, respectively. He is a professor of the Department of Computer Science, National Chiao Tung University (NCTU), and the Director of Taiwan Information Security Center at NCTU. He served in the past as the Computer Science Department Chair of NCTU, Director of GSN-CERT/CC, Advisor to National Information and Communication Security Task Force, and Advisor to National Security Bureau. Dr. Shieh currently serves as the Chair of IEEE Reliability Society Taipei and Tainan Chapter, and a steering committee member of ACM SIGSAC. He is also an associate editor of IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Reliability, ACM Transactions on Information and System Security, Journal of Computer Security, former editor of Journal of Information Science and Engineering, and guest editor of IEEE Internet Computing, respectively. He was the former President of Chinese Cryptology and Information Security Association (CCISA), the largest non-profit academic organization for security research. He was on the organizing committees of numerous conferences, such as Steering Committee Chair of ACM Symposium on Information, Computer and Communications Security. Dr. Shieh has published over a hundred academic articles, including papers, patents, and books.

Recently he received ACM Award for his contribution to ACM, and Distinguished Information Technology Award for his contribution to computer security research. His research interest includes network and system security, wireless security, and cryptography.

J. D. Tygar is Professor of Computer Science at UC Berkeley and also a Professor of Information Management at UC Berkeley. He works in the areas of computer security, privacy, and electronic commerce. His current research includes privacy, security issues in sensor webs, digital rights management, and usable computer security. His awards include a National Science Foundation Presidential Young Investigator Award, an Okawa Foundation Fellowship, a teaching award from Carnegie Mellon, and invited keynote addresses at PODC, PODS, VLDB, and many other conferences. Doug Tygar has written three books; his book *Secure Broadcast Communication in Wired and Wireless Networks* (with Adrian Perrig) is a standard reference and has been translated to Japanese. He designed cryptographic postage standards for the US Postal Service and has helped build a number of security and electronic commerce systems including: Strongbox, Dyad, Netbill, and Micro-Tesla. He served as chair of the Defense Department's ISAT Study Group on Security with Privacy, and was a founding board member of ACM's Special Interest Group on Electronic Commerce. He helped create and remains an active member of TRUST (Team for Research in Ubiquitous Security Technologies). TRUST is a new National Science Foundation Science and Technology Center with headquarters at UC Berkeley and involving faculty from Berkeley, Carnegie Mellon, Cornell, Stanford, and Vanderbilt. Before coming to UC Berkeley, Dr. Tygar was tenured faculty at Carnegie Mellon's Computer Science Department, where he continues to hold an Adjunct Professor position. He received his doctorate from Harvard and his undergraduate degree from Berkeley.