RCDA: Recoverable Concealed Data Aggregation for Data Integrity in Wireless Sensor Networks

Chien-Ming Chen, Yue-Hsun Lin, Ya-Ching Lin, and Hung-Min Sun

Abstract—Recently, several data aggregation schemes based on privacy homomorphism encryption have been proposed and investigated on wireless sensor networks. These data aggregation schemes provide better security compared with traditional aggregation since cluster heads (aggregator) can directly aggregate the ciphertexts without decryption; consequently, transmission overhead is reduced. However, the base station only retrieves the aggregated result, not individual data, which causes two problems. First, the usage of aggregation functions is constrained. For example, the base station cannot retrieve the maximum value of all sensing data if the aggregated result is the summation of sensing data. Second, the base station cannot confirm data integrity and authenticity via attaching message digests or signatures to each sensing sample. In this paper, we attempt to overcome the above two drawbacks. In our design, the base station can recover all sensing data even these data has been aggregated. This property is called "recoverable." Experiment results demonstrate that the transmission overhead is still reduced even if our approach is recoverable on sensing data. Furthermore, the design has been generalized and adopted on both homogeneous and heterogeneous wireless sensor networks.

Index Terms—Concealed data aggregation, wireless sensor networks, privacy homomorphism encryption.

1 Introduction

Wireless sensor networks (WSN) have been widely deployed in many applications, e.g., military field surveillance, health care, environment monitor, accident report, etc. A WSN is composed of a large number of sensors which collaborates with each other. Each sensor detects a target within its radio range, performs simple computations, and communicates with other sensors.

Generally, sensors are constrained in battery power, communication, and computation capability; therefore, reducing the power consumption is a critical concern for a WSN. Recently, a practical solution called *data aggregation* [1], [2], [3] was introduced. The original concept is to aggregate multiple sensing data by performing algebraic or statistical operations such as addition, multiplication, median, minimum, maximum, and mean of a data set, etc. Normally, data aggregation is performed by *cluster heads* if the whole network is divided into several groups known as clusters. For example, in military fields, sensors are deployed to measure radiation or chemical pollution. The base station (sink) may require the maximum value of all sensing data to trigger the immediate response; thus, each cluster head selects the maximum value of multiple sensing data of its cluster members and sends the result to the base station. Obviously, communication cost is reduced since only aggregated results reach the base station.

Unfortunately, an adversary has the ability to capture cluster heads. It would cause the compromise of the whole cluster; consequently, several schemes, such as ESPDA [4] and SRDA [5], have been proposed. However, these schemes restrict the data type of aggregation or cause extra transmission overhead. Besides, an adversary can still obtain the sensing data of its cluster members after capturing a cluster head.

To solve above problems completely, two ideas are used in recent research [6], [7], [8]. First, data are encrypted during transmission. Second, cluster heads directly aggregate encrypted data without decryption. A well-known approach named *Concealed Data Aggregation* (CDA) [6] has been proposed based on these two ideas. CDA provides both end-to-end encryption and in-networking processing in WSN. Since CDA applies privacy homomorphism (PH) encryption with additive homomorphism, cluster heads are capable of executing addition operations on encrypted numeric data. Later, several PH-based data aggregation schemes [7], [8] have been proposed to achieve higher security levels.

In the above PH-based schemes [6], [7], [8], the base station receives only the aggregated results. However, it brings two problems. First, the usage of aggregation functions is constrained. For example, these schemes only allow cluster heads to perform additive operations on ciphertexts sent by sensors; therefore, they are ineffective if the base station desires to query the maximum value of all sensing data. Second, the base station cannot verify the integrity and authenticity of each sensing data. These problems seem to be solved if the base station can receive all sensing data rather

• *The authors are with the Department of Computer Science, National Tsing Hua University, No. 101, Section 2, Kuang-Fu Road, Hsinchu, Taiwan 30013, R.O.C.*
E-mail: {kkyy, tenma, 9962602}@is.cs.nthu.edu.tw, hmsun@cs.nthu.edu.tw.

Manuscript received 19 Apr. 2010; revised 26 Feb. 2011; accepted 7 Apr. 2011; published online 10 Aug. 2011.
Recommended for acceptance by X.-Y. Li.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2010-04-0229.
Digital Object Identifier no. 10.1109/TPDS.2011.219.

than aggregated results, but this method is in direct contradiction to the concept of data aggregation—that the base station obtains only aggregated results. Thus, we attempt to design an approach that allows the base station to receive all sensing data but still reduce the transmission overhead.

Contributions. In this paper, we introduce a concept named *Recoverable Concealed Data Aggregation* (RCDA). In RCDA, a base station can recover each sensing data generated by all sensors even if these data have been aggregated by cluster heads (aggregators). With these individual data, two functionalities are provided. First, the base station can verify the integrity and authenticity of all sensing data. Second, the base station can perform any aggregation functions on them. Then, we propose two RCDA schemes named RCDA-HOMO and RCDA-HETE for homogeneous and heterogeneous WSN respectively. In the security analysis, we demonstrate that the proposed schemes are secure under our attack model. Through experiments, we show that the performance of our design is reasonable and affordable. We also provide detailed comparisons with other schemes.

2 Related Works

Numerous secure data aggregation schemes have been proposed. These schemes are designed for different security requirements.

A number of schemes [9], [10] have been proposed based on the *commit-and-attest* principle. In these schemes, the base station broadcasts aggregation results to all sensors. Then, every sensor verifies that its sensing data were indeed counted. Another work [11] can actually count and sum even if a few compromised sensors inject false values. Yu [12] introduces a random sampling technique that enables aggregation queries to not only detect malicious sensors, but also to tolerate them.

On the other hand, several studies [6], [7], [8] attempt to provide confidentiality. That is, an aggregator can directly execute addition operations on encrypted numeric data. CDA [6] places more emphasis on passive attacks. More specifically, it considers if adversaries can eavesdrop the communications on the air. After CDA, succeeding research [7], [8] have been proposed to achieve higher security levels. They consider the following scenario. If sensors within the same cluster encrypt their sensing data with a common secret key, an adversary may decrypt or fake the aggregated ciphertext by compromising only one sensor. Castelluccia et al. [7] proposed a new PH-based aggregation scheme to overcome this security problem by generating a temporal key for each transmission. Although the influence of compromising a sensor is actually reduced, two practical issues must be considered. First, rekeying operations for each sensor cause this scheme to be impractical. Second, a synchronization mechanism should be provided. Later, Mykletun et al. [8] proposed a data aggregation scheme based on addition homomorphic public-key encryption. It seems more secure since every sensor stores only public key. The adversary cannot launch the same attack through compromising only one sensor. Nevertheless, the adversary can still impersonate other legal sensors to send the forged cipertexts to the cluster head with the same public key. Authenticity of data is not supported.

In our work, we desire to design a scheme which provides both integrity and confidentiality.

3 Preliminaries

In this section, we first describe the network models and define the attack model. Then, Mykletun et al.'s [8] and Boneh et al.'s schemes [13] are reviewed since they are the foundation of the proposed schemes.

3.1 Network Model

A WSN is controlled by a base station (BS). A BS has large bandwidth, strong computing capability, sufficient memory, and stable power to support the cryptographic and routing requirements of the whole WSN. Besides the BS, sensors (SNs) are also deployed to sense and gather responsible results for the BS. Typical SNs are small and low cost; hence, SNs are limited on computation, storage, and communication capability.

Generally, all SNs in a WSN may be divided into several clusters after being deployed. Several research [14], [15], [16] have shown that a cluster-based WSN has several advantages such as efficient energy management, better scalability of MAC (medium access control) or routing, etc. Each cluster has a cluster head (CH) responsible for collecting and aggregating sensing data from SNs within the same cluster. A CH then sends the aggregation results to the BS. In a homogeneous WSN, cluster heads act as normal SNs. On the other hand, cluster heads act as by powerful high-end sensors (H-Sensors), in a heterogeneous WSN which incorporates different types of SNs with different capabilities.

3.2 Attack Model

The attack model is defined based on the ability of adversaries. Here, we consider the following three cases:

1. Without compromising any SN or CH. An adversary can only eavesdrop on packets in the air, so he can modify or inject the forged messages with this public information.
2. Compromising SNs. After compromising a SN, an adversary can obtain secrets such as encryption/decryption keys. Then, an adversary can obtain sensing data and packets passed through the captured SN or impersonate this compromised sensor to forge malicious data.
3. Compromising CHs. After compromising a CH, an adversary can obtain the secrets and perform the following attacks. First, an adversary can decrypt the ciphertext of sensing data sent by its cluster members. Second, an adversary can generate forged aggregation results.

3.3 Mykletun et al.'s Encryption Scheme

Mykletun et al. [8] proposed a concealed data aggregation scheme based on the elliptic curve ElGamal (EC-EG) cryptosystem. It consists of four procedures: key generation (KeyGen), encryption (Enc), aggregation (Agg), and decryption (Dec). Details are illustrated in Fig. 1. In Fig. 1, symbol $+$ and $\times$ denote addition and scalar multiplication on elliptic curve points, respectively.

Mykletun *et al.*'s encryption scheme
KeyGen(τ): τ is security parameter.
1. Construct an elliptic curve $\mathcal{E}$ over a finite field $\mathcal{F}_p$ satisfying: p is a prime number, and the order of $\mathcal{E}$, $\#\mathcal{E}(\mathcal{F}_p)$, has a large prime factor.
2. Select private key ζ randomly from $\mathcal{F}_p$.
3. Generate public key $\eta = (Y, \mathcal{E}, p, G, n)$ where $n = \#\mathcal{E}(\mathcal{F}_p)$, $Y = \zeta \times G$, and G is a generator on $\mathcal{E}$, i.e., $n \times G = \infty$.
4. Return key pair (η, ζ).

Enc$_\eta(m)$: Encrypt message m with public key η.
1. Randomly select $k \in [0, n-1]$ where $n \in \eta$.
2. Compute $M = map(m)$, where $map(s)$[1] $= s \times G$.
3. Output cipher $C = (R, S) = (k \times G, M + k \times Y)$.

Agg(C_1, C_2): Aggregate two ciphers C_1 and C_2 to C' where $C_1 = (R_1, S_1) = (k_1 \times G, M_1 + k_1 \times Y)$ and $C_2 = (R_2, S_2) = (k_2 \times G, M_2 + k_2 \times Y)$.
1. Computed C' as: $C' = (R', S') = C_1 + C_2 = (R_1 + R_2, S_1 + S_2)$ $= (k_1 \times G + k_2 \times G,\ M_1 + k_1 \times Y + M_2 + k_2 \times Y)$ $= ((k_1 + k_2) \times G,\ M_1 + M_2 + (k_1 + k_2) \times Y)$

Dec$_\zeta(C)$: Decrypt cipher C with private key ζ where $C = (R, S)$.
1. Compute $M = -\zeta \times R + S = M + k \times Y - k \times Y$.
2. Reverse m through $m = rmap$[2](M).
3. Return the plaintext m.

Boneh *et al.*'s scheme for entity i
KeyGen(τ): τ is security parameter.
1. Generate private key x_i randomly selected from $\mathcal{Z}_p$.
2. Generate public key $v_i \in G_2$ where $v_i = x_i \times g_2$.
3. Output key pair (x_i, v_i) for entity i.

Sign$_{x_i}(m)$: Sign message m with the private key x_i.
1. Compute $h = H(m)$ where $h \in G_1$.
2. Generate signature $\sigma = x_i \times h$ and return (m, σ).

Verify$_{v_i}(m, \sigma)$: Verify message m with signature σ by public key v_i.
1. Compute $h = H(m)$.
2. Compute $e_n(\sigma, g_2)$, $e_n(h, v_i) \in G_T$.
3. If $e_n(\sigma, g_2) = e_n(h, v_i)$, accept; Otherwise, reject.

Agg$_k(\delta, M)$: Aggregate k signatures where message set $M = \{m_1, \ldots, m_k\}$; m_i from entity i and $\delta = \{\sigma_1, \ldots, \sigma_k\}$; σ_i is signature of m_i.
1. Produce aggregated signature $\hat{\sigma}$=$\sigma_1 + \cdots + \sigma_k$=$\sum_{i=1}^{k} \sigma_i$, where $\hat{\sigma}, \sigma_1, \cdots, \sigma_k \in G_1$.

Agg-Verify$_\vartheta(\hat{\sigma}, M)$: Verify the aggregated signature. $\hat{\sigma}$ is the aggregated signature of message set M where $M = \{m_1, \cdots, m_k\}$; m_i from entity i, and public key set $\vartheta = \{v_1, \cdots, v_k\}$; $v_i \in U_i$.
1. Compute $h_i = H(m_i)$, for $1 \le i \le k$.
2. Accept if $e(\hat{\sigma}, g_2) = \prod_{i=1}^{k} e(h_i, v_i)$, where $e(\hat{\sigma}, g_2)$, $e(h_i, v_i) \in G_T$; Otherwise, reject.

Fig. 1. Mykletun et al.'s and Boneh et al.'s schemes.

3.4 Boneh et al.'s Signature Scheme

Boneh et al. [13] proposed an aggregate signature scheme which merges a set of distinct signatures into one aggregated signature. This scheme consists of five procedures: key generation (KeyGen), signing (Sign), verifying (Verify), aggregation (Agg), and verifying aggregated signature (Agg-Verify). Details are given in Fig. 1.

Boneh et al.'s scheme is based on bilinear map e_n which is defined as $e_n = G_1 \times G_2 \rightarrow G_T$, where groups G_1, G_2, and G_T are cyclic groups of prime order n. G_1 and G_2 are n-torsion point groups on an elliptic curve $\mathcal{E}$ under a finite field $\mathcal{F}_p$, i.e., $n \times P = n \times Q = \infty$, where $\forall P \in G_1$ and $\forall Q \in G_2$. G_T is the group of nth root of unity in an extension field $\mathcal{F}_{p^k}$, i.e., $G_T = \{x \in \mathcal{F}_{p^k} | x^n = 1_T\}$. The group operation in G_1 and G_2 is point addition and one in G_T is multiplication over a finite field.

1. $map()$ maps a scalar value m to a curve point M. $map()$ satisfies the additive homomorphic property, i.e., $map(m_1 + \cdots + m_n) = (m_1 + \cdots + m_n) \times G = m_1 \times G + \cdots + m_n \times G = map(m_1) + \cdots + map(m_n)$.
2. The reverse function $rmap()$ maps a given point M to the scalar value $m \in \mathcal{F}_p$. $rmap()$ can be achieved by Pollard-λ method on elliptic curve cryptosystems [8], [17].

TABLE 1
Notations Used in RCDA-HOMO

Symbol	Description	Symbol	Description
BS	Base station	CH	Cluster head
SN_i	Sensor node i	$\mathcal{H}$	hash function
(P_i, R_i)	public-private key of entity i	$m[i,k]$	substring of m from index i to k, e.g., $m = 1101_2, m[0,1] = 01_2$
d_i	sensing data of SN_i	m_i	encoded result of d_i
0^β	β serial 0 bits	l	bit-length of payload
c_i	ciphertext of m_i	σ_i	signature of d_i
$\hat{c}$	aggregated ciphertext	$\hat{\sigma}$	aggregated signature
$\|$	concatenation	$\cdot$	integer multiplication
$+$	addition on elliptic curves	$\times$	scalar multiplication on elliptic curves

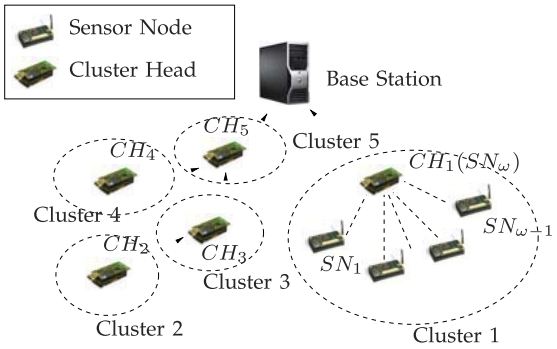

Fig. 2. Example of homogeneous WSN environment.

4 A RCDA Scheme for Homogeneous WSN (RCDA-HOMO)

In this section, we propose a recoverable concealed data aggregation scheme named *RCDA-HOMO* for homogeneous WSN. Table 1 lists the notations that we will use later.

4.1 Construction of RCDA-HOMO

RCDA-HOMO is composed of four procedures: *Setup*, *Encrypt-Sign*, *Aggregate*, and *Verify*. The *Setup* procedure is to prepare and install necessary secrets for the BS and each sensor. When a sensor decides to send sensing data to its CH, it performs *Encrypt-Sign* and sends the result to the CH. Once the CH receives all results from its members, it activates *Aggregate* to aggregate what it received, and then sends the final results (aggregated ciphertext and signature) to the BS. The last procedure is *Verify*. The BS first extracts individual sensing data by decrypting the aggregated ciphertext. Afterward, the BS verifies the authenticity and integrity of the decrypted data based on the corresponding aggregated signature.

To present RCDA-HOMO in a simple way, we choose Cluster 1 (see Fig. 2) as an example. SN_ω is selected as CH of Cluster 1 which contains the remaindering sensors, $\{SN_1, \ldots, SN_{\omega-1}\}$. The detailed procedures are listed as follows:

Setup: BS generates the following key pairs:

1. (P_{SN_i}, R_{SN_i}): For each sensor SN_i, the BS generates (P_{SN_i}, R_{SN_i}) by KeyGen procedure (see Boneh et al.'s scheme in Fig. 1) where $P_{SN_i} = v_i$ and $R_{SN_i} = x_i$.
2. (P_{BS}, R_{BS}): These keys are generated by KeyGen procedure (see Mykletun et al.'s scheme in Fig. 1) where $P_{BS} = \{Y, E, p, G, n\}$ and $R_{BS} = \zeta$.

After that, R_{SN_i}, P_{BS}, and $\mathcal{H}$ are loaded to SN_i for all i. Finally, the BS keeps all public keys P_{SN_i} and its own R_{BS} in privacy.

Encrypt-Sign: This procedure is triggered while a sensor decides to send its sensing data to the cluster head (CH_1 in Fig. 2). Detailed steps are listed as follows:

1. Encoding d_i: $m_i = d_i \| 0^{\beta}$, where $\beta = l \cdot (i-1)$.
2. After encoding, SN_i computes:
   a. Signature: $\sigma_i = x_i \times h_i$, where $h_i = \mathcal{H}(d_i)$.
   b. Ciphertext: $c_i = (r_i, s_i) = (k_i \times G, M_i + k_i \times Y)$, where k_i is randomly selected from $\{0, \ldots, n-1\}$, $M_i = map(m_i) = m_i \times G$, and n, G, $Y \in P_{BS}$.
3. At the end, SN_i sends the pair (c_i, σ_i) to CH_1.

Aggregate: The *Aggregate* procedure is launched after the CH has gathered all cipertext-signaturte pairs, i.e., CH_1 gathered $\omega - 1$ pairs $((c_1, \sigma_1), \ldots, (c_{\omega-1}, \sigma_{\omega-1}))$ over a period of time. Aggregation operations are given as follows:

1. Aggregated ciphertext:
$$\hat{c} = (\hat{r}, \hat{s}) = \sum_{i=1}^{\eta-1} c_i = \left(\sum_{i=1}^{\eta-1} r_i, \sum_{i=1}^{\eta-1} s_i \right).$$
2. Aggregated signature: $\hat{\sigma} = \sum_{i=1}^{\eta-1} \sigma_i$.
3. Send the aggregated result $(\hat{c}, \hat{\sigma})$ to the BS.

Verify: While receiving $(\hat{c}, \hat{\sigma})$ from CH_1, BS can recover and verify each sensing data via the following steps:

1. BS obtains M' by decrypting $\hat{c}$ with R_{BS} $M' = -t \times \hat{r} + \hat{s} = M_1 + \cdots + M_{\omega-1}$.
2. BS obtains m' from M' through the reserve function $rmap()$: $m' = rmap(M') = m_1 + \cdots + m_{\omega-1}$.
3. BS obtains each sensing data from m' by $Decode$ function: $Decode(m', \omega - 1, l)$: $d_i = m'[(i-1) \cdot l, i \cdot l - 1]$, where $i = 1, \ldots, \omega - 1$.
4. BS verifies each d_i via checking whether the equation $e_n(\hat{\sigma_1}, g_2) = \prod_{i=1}^{\eta-1} e_n(h_i, P_{SN_i})$ holds or not. Each element h_i is derived from hashing d_i, i.e., $h_i = \mathcal{H}(d_i)$. Note that e_n is the bilinear map (see Section 3.4). For all d_i, if the equation holds, BS accepts; otherwise, BS rejects.

Similarly, the BS may receive other ciphertext and signature pairs form other clusters. The BS can recover all sensing data within the whole WSN. After confirming the integrity of all data, the BS can perform any operations if it wants since all individual data are reverted.

4.2 A Concrete Example

Now we give an example to demonstrate how RCDA-HOMO works. Assume that a WSN consists of five sensors denoted as $\{SN_1, \ldots, SN_5\}$ and SN_5 is selected as CH. Assume that sensing data of each sensor are $d_1 = 5$, $d_2 = 3$, $d_3 = 7$, and $d_4 = 8$. Length l is set as 4 since 4 bits is sufficient to represent all sensing data in this example. SN_3 performs the *Encrypt-Sign* procedure as follows:

1. Encode d_3: $m_3 = d_3 \| 0^{\beta} = (011100000000)_2$, where $\beta = l \cdot (i-1) = 4 \cdot 2 = 8$.
2. Compute (c_3, σ_3) and send it to CH.

Similarly, SN_1, SN_2, and SN_4 send their own (c_i, σ_i) pair to the CH. Note that $m_1 = (0101)_2 =, m_2 = (00110000)_2$, $m_4(1000000000000000)_2$. After gathering four pairs of (c_i, σ_i) where $i = 1, \ldots, 4$, the CH aggregates ciphertexts and signatures through *Aggregate* procedure. The CH then sends the aggregated result $(\hat{c}, \hat{\sigma})$ to the BS. After that, the BS executes *Verify* procedure.

1. BS decrypts the ciphertext $\hat{c}$ through R_{BS}. It obtains $M' = M_1 + M_2 + M_3 + M_4$ and further retrieves $m' = (1000011100110101)_2$.
2. BS applies $Decode(m', 4, 4)$ to obtain individual sensing data, i.e., $d_i = m'[4 \cdot (i-1), 4 \cdot i - 1], \forall i$; $d_1 = (0101)_2 = 5$, $d_2 = (0011)_2 = 3$, $d_3 = (0111)_2 = 7$, and $d_4 = (1000)_2 = 8$.
3. Finally, BS verifies the aggregated signature $\hat{\sigma}$ by checking whether the below equation holds or not: $e_n(\hat{\sigma}, g_2) = \prod_{i=1}^{4} e_n(h_i, P_i)$, where $h_i = \mathcal{H}(d_i)$.

5 A RCDA Scheme for Heterogeneous WSN

Here, we consider another environment, heterogeneous WSN. A concealed data aggregation scheme for heterogeneous WSN has been proposed [18]; however, their scheme does not provide data integrity and recovery. We first propose naïve RCDA-HETE scheme. Later, we will propose another scheme named RCDA-HETE if H-Sensors are designed to be tamper-resistant.

5.1 Naïve RCDA-HETE Scheme

Actually, RCDA-HOMO can be applied to heterogeneous WSN without modification. We call this approach naïve RCDA-HETE. Since H-Sensors are capable of stronger computation ability and stable power supply, they can perform more complex tasks than L-Sensors. Thus, H-sensors can act as cluster heads. Obviously, naïve RCDA-HETE also achieve the *Recovery* property.

5.2 RCDA-HETE Scheme

Here, we attempt to fully exploit H-Sensors which have stronger computing capability. Operations on L-Sensors could be switched to H-Sensors. In addition, H-Sensors can be designed to be tamper-resistant, so we may allow H-Sensors to store the partial secret information if required. With these considerations, we redesign an RCDA scheme named RCDA-HETE.

While the use of tamper-resistant devices may rise the hardware cost; however, in a heterogeneous WSN, majority of sensors are low-end sensors (L-Sensors). In our design, computation cost on L-Sensors is switched to H-Sensors, so L-Sensors can be very cheap and simple. In fact, the overall hardware cost is reduced.

RCDA-HETE is composed of five procedures: *Setup*, *Intracluster Encrypt*, *Intercluster Encrypt*, *Aggregate*, and *Verify*. In the *Setup* procedure, necessary secrets are loaded to each H-Sensor and L-Sensor. *Intracluster Encrypt* procedure involves when L-Sensors desire to send their sensing data to the corresponding H-Sensor. In the *Intercluster Encrypt* procedure, each H-Sensor aggregates the received data and then encrypts and signs the aggregated result. In addition, if an H-Sensor receives ciphertexts and signatures from other H-Sensors on its routing path, it activates the

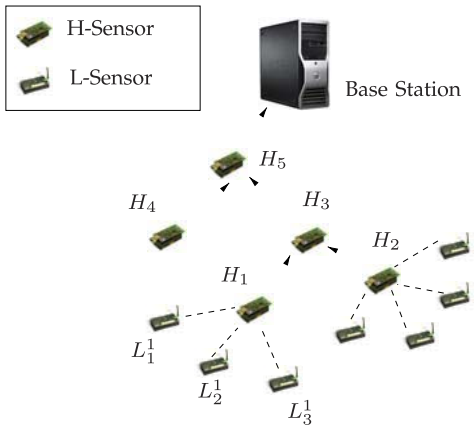

Fig. 3. An example of heterogeneous WSN.

Aggregate procedure. Finally, the *Verify* procedure ensures the authenticity and integrity of each aggregated result. To explain RCDA-HETE clearly, a heterogeneous WSN is given in Fig. 3. The notations are listed in Table 2 and the detailed procedures are described below.

Setup: In the beginning, the BS generates the following keys:

1. (R_{H_i}, P_{H_i}): the BS generates this key pair for each H-Sensor according to KeyGen of Boneh et al.'s scheme (see Fig. 1), i.e., $R_{H_i} = x_i$ and $P_{H_i} = v_i$.
2. (R_{BS}, P_{BS}): This key pair is generated by KeyGen of Mykletun et al.'s scheme (see Fig. 1), i.e., $P_{BS} = \eta = \{Y, E, p, G, n\}$ and $R_{BS} = \zeta$.

Then, the BS loads P_{BS} to all L-Sensors. On the other hand, each H-Sensor is loaded its own key pair (P_{H_i}, R_{H_i}), P_{BS} and several necessary aggregation functions.

In our design, each L-Sensor is requiresd to share a pairwise key with its cluster head. For example, L-Sensor L_i^j would share a key K_i^j with the corresponding cluster head H_j. If the BS knows the cluster information before deployment, the pairwise keys can be preloaded to all L-Sensors and H-Sensors. However, in most WSN environment, sensors are randomly deployed. Thus, we propose a simple key exchange scheme. The detailed steps are described in the Section 1 of *Supplemental Material*, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.219.

Intracluster Encrypt: This procedure ensures the establishment of a secure channel between L-Sensors and their H-Sensor. Take Fig. 3 as an example, L_i^1 encrypts d_i^1 with K_i^1 and sends $E_{K_i^1}(d_i^1)$ to H_1. After receiving $E_{K_i^1}(d_i^1)$, H_1 decrypts the ciphertexts to obtain the plaintext d_i^1.

Intercluster Encrypt: After collecting all sensing data from all cluster members, an H-Sensor performs the preferred aggregation function on these data as its result. For example, in Fig. 3, H_1 select d_i^1 as the aggregated result δ_1 by predefined property, such as maximum or minimum. Then, H_1 performs the following steps:

1. Encoding δ_1 as m: $m = \delta_1 \| 0^\beta$, where $\beta = l \cdot (i-1)$.
2. After encoding, H_1 computes:
   a. Signature: $\sigma_1 = x_1 \times h_1$, where x_1 is R_{H_1} and $h_1 = \mathcal{H}(\delta_i)$.
   b. Ciphertext: $c_1 = (r_1, s_1) = (k_1 \times G, M_1 + k_1 \times Y)$, where k is randomly selected from $\{1, \ldots, n\}$, $M_1 = map(m_1) = m_1 \times G$, and $n, G, Y \in P_{BS}$.
3. H_1 sends the pair (c_1, σ_1) to H_3. Similarly, each H_j also calculates (c_j, σ_j) from δ_j in other clusters.

TABLE 2
Notations Used in RCDA-HETE[*3]

Symbol	Description	Symbol	Description
H_i	H-Sensor i	L_i^j	L-Sensor i belongs to H_j
$E_k(m)$	encrypt m with key k	d_i^j	sensing data of L_i^j
δ_j	selected data by H_j	m_j	encoded result of δ_j
K_i^j	Pairwise key shared by L-Sensor i with H-Sensor j		

Aggregate: As an example shown in Fig. 3, if H_3 receives (c_1, σ_1) from H_1 and (c_2, σ_2) from H_2, H_3 will execute this procedure to aggregate (c_1, σ_1), (c_2, σ_2) and its own (c_3, σ_3) as follows:

1. Aggregated ciphertexts: $\hat{c_3} = (\sum_{i=1}^{3} r_i, \sum_{i=1}^{3} s_i)$.
2. Aggregated signature: $\hat{\sigma_3} = \sum_{i=1}^{3} \sigma_i$.

Finally, H_3 sends $(\hat{c_3}, \hat{\sigma_3})$ to H_5. Similarly, H_5 can also aggregate (c_4, σ_4), (c_5, σ_5), and $(\hat{c_3}, \hat{\sigma_3})$ and get a new aggregated result $(\hat{c_5}, \hat{\sigma_5})$ to the BS.

Verify: After receiving the end result $(\hat{c_5}, \hat{\sigma_5})$, BS will perform the following steps:

1. Obtain M' by decrypting $\hat{c_5}$: $M' = -t \times r + s = M_1 + M_2 + \cdots + M_5$.
2. Obtain m' from M' through the reserve function $rmap()$: $m' = rmap(M') = m_1 + m_2 + \cdots + m_5$.
3. Obtain δ_i from m' using the $Decode$ function: $Decode(m', 5, l) : \delta_i = m'[(i-1) \cdot l, i \cdot l - 1]$, where $i = 1, \ldots, 5$.
4. Check whether $e_n(\hat{\sigma_5}, g_2) = \prod_{i=1}^{5} e_n(h_i, P_i)$ holds or not. Element h_i is derived by hashing δ_i, i.e., $h_i = \mathcal{H}(\delta_i)$. e_n is the bilinear map. If the equation holds, accept all δ_i; otherwise, reject.

After checking the integrity of each δ_i, the BS can further perform the aggregation function on all δ_i.

5.3 Recovery Property

The *Recovery* property attempts to provide two functionalities. First, BS can verify the integrity and authenticity of all sensing data. Second, BS can perform arbitrary aggregation operations on these data. However, in RCDA-HETE, the BS only recovers individual aggregated result generated by each cluster rather than all sensing data. Now we will show that RCDA-HETE also provides these functionalities.

1. RCDA-HETE can verify each sensing data through the aid of H-Sensors. More precisely, *Intracluster Encrypt* procedure allows L-Sensor L_i^j to send not only $E_{K_i^j}(d_i^j)$, but also the MAC (message authentication code) of $E_{K_i^j}(d_i^j)$ to its cluster head H_j; therefore, H_j can verify the integrity of the data sent from its cluster members.
2. Every H-Sensor is loaded several necessary aggregation functions before deployment, so the BS can command every H-Sensor to perform the designated

3. Notations defined in Table 1 are not repeated here.

aggregation function. For example, if BS decides to obtain the summation of all data, it assigns H-Sensors to perform the addition operation. Then, the BS can perform the last addition when it recovers every result from every H-Sensor. Similarly, if BS then decides to perform maximum-selection operation, the BS notifies every H-Sensor to select the maximum value among the sensing data in the *Intercluster Encrypt* procedure.

6 Security and Scalability Analysis

In this section, we demonstrate the proposed schemes are secure under the attack model defined in Section 3.2. More detailed security analysis and scalability analysis are described in Section 2 of the Supplemental Material available online.

We first assume that an adversary does not compromise sensors. The proposed schemes are secure because sensing messages are encrypted. In RCDA-HOMO, each sensor encrypts their messages with P_{BS} before transmitting. In RCDA-HETE, intracluster traffic is encrypted with pairwise keys. Besides, our design generates the corresponding signature for each sensing data. Consequently, an adversary cannot modify messages and inject forged messages since he cannot sign forged messages without private keys.

If an adversary has the ability to compromise sensors, we consider the following situations. An adversary can compromise a sensor and perform it as a legal one. Detecting compromised sensors that still act normally is infeasible in all existing detection mechanisms in WSN. Also, if the value of a forged message is in a reasonable range, detecting it is still infeasible. An adversary can also try to manipulate the aggregated result. He may generate false data, modify legal messages, or impersonate other sensors. The proposed schemes are still secure against above attacks because of the signature required for each generated message. On the other hand, we discuss the situation when an adversary compromises a cluster head in RCDA-HOMO. First, he cannot decrypt the aggregated ciphertext or each individual ciphertext because no decryption private key is stored in a cluster. Second, the compromised cluster head may selectively drop some ciphertexts and signatures in the *Aggregate* procedure. This kind of attack which is called selective forwarding attack was first described in [19]. Fortunately, previous research [20], [21] proposed mechanisms to defend against this attack.

7 Implementation and Evaluation

In this section, the implementation of the proposed schemes is given first. Then, the evaluated results on the proposed schemes are given.

7.1 Implementation

The proposed schemes were all implemented on physical sensors. For homogeneous WSNs, MICAz is selected as our platform. For heterogeneous WSNs, MICAz acts as L-Sensor, and SCAN-ZB32 [22] produced by ITRI is selected as H-Sensor. Software libraries and programs are implemented functions from Mykletun et al.'s (called MYK for short) and Boneh et al.'s schemes (called BON for short). Functions in MYK all involve elliptic curve cryptography; hence, we utilize the TinyECC (v1.0) library to implement MYK. Since BON requires bilinear map construction, we adopt TinyPBC [23] to meets this requirement. For detailed description, please refer to Section 3 of the Supplemental Material available online.

TABLE 3
Performance and Cost Evaluation of the Proposed Schemes (L Denotes L-Sensor, H Denotes H-Sensor)

	RCDA-HOMO	naïve RCDA-HETE	RCDA-HETE
Processing Delay(ms)	3702.09	3702.09	2.970(L) 132.22(H)
Processing Energy(μJ)	12079.9	12079.9	49.69(L) 431.43(H)
Aggregation Delay(ms)	73.71[4]	3.371	3.371
Aggregation Energy(μJ)	204.52	57.81	57.81
Payload Size(bit)	476	476	256(L) 476(H)
Comm. Cost(μJ)	338.4(Send) 377.88(Receive)	338.4(L) 502.7(H)	153.6(L) 351.97(H)

7.2 Performance and Cost Evaluation

To evaluate the performance of the proposed schemes, execution time (or "delay") is the main measurement of performance evaluation. Without loss of generality, we define processing delay and aggregation delay for deployed sensors. Processing delay indicates the execution time for sensors to produce ciphertexts and corresponding signatures before transmission. Aggregation delay is also evaluated by measuring time spent on processing time on aggregating ciphertexts and signatures in the proposed schemes. The last delay, decryption delay, is not considered since the base station is considerably powerful as a workstation. Therefore, this delay is negligible and can be ignored.

Another criterion is cost evaluation. Cost evaluation involves communication and computation aspects. According to Wander et al.'s result, computation cost can be easily calculated based on the wasted clock cycles [24]. They showed that executing 2,090 clock cycles equals approximately $7.4\ \mu$J. For communication, we choose Meulenaer et al.'s result [25]. They found that a MICAz node consumes $0.6\ \mu$J to send per bit and $0.67\ \mu$J to receive per bit averagely. After evaluation the proposed schemes on physical sensors, results are given in Table 3.

On average, the processing delay on MICAz takes about 3.7 second. For cluster heads, delay is required since each CH must process and aggregate ciphertexts and signatures. A CH needs 73.71 ms to aggregate two data from its children. In other words, If a CH has 10 child nodes, it spends $9{*}73.71 = 663.39$ ms. For communication cost, length of message is 476 bits, which contains MYK's ciphertext $(161{*}2 = 322$ bits) and BON's signature (finite field 3^{97} occupies 154 bits). Plus 802.16 header (11 bytes), the packet length is $476 + 88 = 564$ bits. Via Meulenaer et al.'s result, the cost for sending a packet in RCDA-HOMO is $564 * 0.6 = 338.4\ \mu$J, and receiving a packet at a CH is $564 * 0.67 = 377.88\ \mu$J.

4. If a CH has k children, the aggregation spends $((k-1){*}73.71)$ ms.

TABLE 4
Comparison Results of Selected Literatures

	Processing Delay(ms)	Aggregation Delay(ms)	Payload Size(bit)	Comm. Cost(μJ)
RCDA-HOMO	3702.09	73.71	476	338.4
MYK	2781.71	49.14	322	246
CDA	3.097	3.15	128	129.6

In naïve RCDA-HETE, MICAz nodes act as L-Sensors which gather data and perform *Encrypt-Sign* the same way as RCDA-HOMO. Hence, the processing delay equals to the delay in RCDA-HOMO. For aggregation delay, cluster heads are ZB32 nodes, which are more powerful. Therefore, delay is reduced to 3.371 ms. Compared with RCDA-HOMO, aggregation performance is approximately 21.9 times faster than in RCDA-HOMO.

The last scheme, RCDA-HETE, has been revised from naïve RCDA-HETE to enhance the performance of L-Sensors (see Table 3). Processing delay on L-Sensors decreases (2.97 ms) since *Intraencrypt* leverages symmetric cryptography. Most of computation cost has been switch to H-Sensors instead. Although *Interencrypt* is similar to *Encrypt-Sign*, H-Sensors performs better and saves more energy than L-Sensors. Another improvement is the decreased communication costs. Compared with RCDA-HOMO, a MICAz spends 338.4 and 377.88 μJ while sending (as SN) and receiving a packet (as CH). In RCDA-HETE, payload is reduced to 256 bits (AES-256). Meanwhile, the energy for sending a packet on MICAz (L-Sensor) is also reduced, $153.6\ \mu J$. Since the length of packet decreases, energy consumed on receiving is also reduced, only 351.97 μJ on H-Sensors.

To summarize the results from the proposed schemes, RCDA-HETE utilizes the benefits and advantages of H-Sensors. The naïve RCDA-HETE reduces the corresponding delays during aggregations compared with RCDA-HOMO. However, H-Sensors require more energy on communication in naïve RCDA-HETE. In Section 3 of the Supplemental Material available online, we further simulate a WSN while applying RCDA-HOMO, RCDA-HETE, and *Nonaggregate* model.

8 Comparisons

We choose two related literatures, CDA and Mykletun et al.'s scheme, to use as comparator for the proposed schemes. Since their schemes are adopted on homogeneous WSN, RCDA-HOMO is selected as the candidate for comparison. An overall comparison is showed in Table 4.

Delays on processing and aggregation. For node processing delay, CDA is the most efficient one since it adopts symmetric cryptography. RCDA-HOMO and Mykletun et al.'s scheme takes longer because of asymmetric cryptography. Delay in RCDA-HOMO is approximately 1.5 times of delay in Mykletun et al.'s scheme because RCDA-HOMO requires an additional operation, signature generation. Considering aggregation delay, CDA is still the most effective. Similarly, delay of RCDA-HOMO is 1.5 times of delay in Mykletun et al.'s scheme.

Communication cost (comm. cost). Communication cost increases linearly when the size of ciphertext increases. Among them, CDA has the shortest length since CDA adopted RC5 as its cipher, only 128 bits [6]. The cost is 144.72 μJ per transmission on a MICAz sensor. On the other hand, the size of ciphertexts in RCDA-HOMO and Mykletun et al.'s scheme require 476 bits and 322 bits, respectively. Therefore, the energy consumed on communication increases to 377.88 and 274.7 μJ, respectively.

According to the above comparisons, RCDA-HOMO seems to be the worst in performance evaluation. This is because RCDA-HOMO provides better security. Fortunately, the overall cost in RCDA-HOMO is still affordable for WSN. On the other hand, CDA and Mykletun et al.'s scheme could combine other secure mechanisms to achieve the same security level with RCDA-HOMO. However, the cost of involved mechanisms is raised and unpredictable.

9 Conclusion

In this paper, we have proposed recoverable concealed data aggregation schemes for homogeneous/heterogeneous WSNs. A special feature is that the base station can securely recover all sensing data rather than aggregated results, but the transmission overhead is still acceptable. Moreover, we integrate the aggregate signature scheme to ensure data authenticity and integrity in the design. Even though signatures bring additional costs, the proposed schemes are still affordable for WSNs after evaluation. Considering a large WSN (over 100 nodes), we also performed simulations on the proposed schemes. The results are available in the online *Supplemental Material*.

Acknowledgments

The authors would like to thank anonymous reviewers for their valuable comments and suggestions, which certainly led to improvements of this paper. This work was supported in part by the National Science Council, Taiwan, under Contracts NSC 99-2218-E-007-012 and NSC 100-2218-E-007-006. The corresponding author is Professor Hung-Min Sun.

References

[1] R. Rajagopalan and P. Varshney, "Data-Aggregation Techniques in Sensor Networks: A Survey," *IEEE Comm. Surveys Tutorials*, vol. 8, no. 4, pp. 48-63, Oct.-Nov. 2006.

[2] S. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," *Proc. Fifth Symp. Operating Systems Design and Implementation*, 2002.

[3] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust Computation of Aggregates in Wireless Sensor Networks: Distributed Randomized Algorithms and Analysis," *IEEE Trans. Parallel Distributed Systems*, vol. 17, no. 9, pp. 987-1000, Sept. 2006.

[4] H. Çam, S. Özdemir, P. Nair, D. Muthuavinashiappan, and H. Ozgur Sanli, "Energy-Efficient Secure Pattern Based Data Aggregation for Wireless Sensor Networks," *J. Computer Comm.*, vol. 29, pp. 446-455, 2006.

[5] H. Sanli, S. Ozdemir, and H. Cam, "SRDA: Secure Reference-Based Data Aggregation Protocol for Wireless Sensor Networks," *Proc. IEEE 60th Int'l Conf. Vehicular Technology (VTC '04-Fall)*, vol. 7, pp. 4650-4654, Sept. 2004.

[6] D. Westhoff, J. Girao, and M. Acharya, "Concealed Data Aggregation for Reverse Multicast Traffic in Sensor Networks: Encryption, Key Distribution, and Routing Adaptation," *IEEE Trans. Mobile Computing*, vol. 5, no. 10, pp. 1417-1431, Oct. 2006.

[7] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient Aggregation of Encrypted Data in Wireless Sensor Networks," *Proc. Second Ann. Int'l Conf. Mobile and Ubiquitous Systems,* pp. 109-117, July 2005.

[8] E. Mykletun, J. Girao, and D. Westhoff, "Public Key Based Cryptoschemes for Data Concealment in Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Comm.,* vol. 5, pp. 2288-2295, June 2006.

[9] H. Chan, A. Perrig, and D. Song, "Secure Hierarchical In-Network Aggregation in Sensor Networks," *Proc. ACM 13th Conf. Computer and Comm. Security,* pp. 278-287, 2006.

[10] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: A Secure Hop-by-Hop Data Aggregation Protocol for Sensor Networks," *ACM Trans. Information and System Security (TISSEC),* vol. 11, no. 4, pp. 1-43, 2008.

[11] S. Roy, S. Setia, and S. Jajodia, "Attack-Resilient Hierarchical Data Aggregation in Sensor Networks," *Proc. ACM Fourth Workshop Security of Ad Hoc and Sensor Networks,* pp. 71-82, 2006.

[12] H. Yu, "Secure and Highly-Available Aggregation Queries in Large-Scale Sensor Networks via Set Sampling," *Proc. IEEE Int'l Conf. Information Processing in Sensor Networks,* pp. 1-12, 2009.

[13] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," *Proc. 22nd Int'l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt),* pp. 416-432, 2003.

[14] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Trans. Wireless Comm.,* vol. 1, no. 4, pp. 660-670, Oct. 2002.

[15] M. Demirbas, A. Arora, V. Mittal, and V. Kulathumani, "A Fault-Local Self-Stabilizing Clustering Service for Wireless Ad Hoc Networks," *IEEE Trans. Parallel Distributed Systems,* vol. 17, no. 9, pp. 912-922, Sept. 2006.

[16] S. Basagni, M. Mastrogiovanni, A. Panconesi, and C. Petrioli, "Localized Protocols for Ad Hoc Clustering and Backbone Formation: A Performance Comparison," *IEEE Trans. Parallel Distributed Systems,* vol. 17, no. 4, pp. 292-306, Apr. 2006.

[17] J. Pollard, "Monte Carlo Methods for Index Computation (mod p)," *Math. of Computation,* vol. 32, pp. 918-924, 1978.

[18] S. Ozdemir, "Concealed Data Aggregation in Heterogeneous Sensor Networks Using Privacy Homomorphism," *Proc. IEEE Int'l Conf. Pervasive Services,* pp. 165-168, July 2007.

[19] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," *Proc. IEEE First Int'l Workshop Sensor Network Protocols and Applications,* pp. 113-127, May 2003.

[20] B. Yu and B. Xiao, "Detecting Selective Forwarding Attacks in Wireless Sensor Networks," *Proc. IEEE 20th Int'l Symp. Parallel and Distributed Processing (IPDPS' 06),* Apr. 2006.

[21] T.H. Hai and E.-N. Huh, "Detecting Selective Forwarding Attacks in Wireless Sensor Networks Using Two-Hops Neighbor Knowledge," *Proc. IEEE Seventh Int'l Symp. Network Computing and Applications,* pp. 325-331, July 2008.

[22] W. Li, C. Chou, and Z. Lin, "Design and Implementation of a Zigbee-Based Communication Substrate for Wireless Sensor Networks," *Proc. Nat'l Computer Symp. Conf.,* 2006.

[23] L. Oliveira et al., "TinyPBC: Pairings for Authenticated Identity-Based Non-Interactive Key Distribution in Sensor Networks," *Computer Comm.,* vol. 34, pp. 485-493, vol. 34, 2010.

[24] A. Wander, N. Gura, H. Eberle, V. Gupta, and S. Shantz, "Energy Analysis of Public-Key Cryptography for Wireless Sensor Networks," *Proc. IEEE Third Int'l Conf. Pervasive Computing and Comm. (PERCOM '06),* 2005.

[25] G. De Meulenaer, F. Gosset, F.X. Standaert, and L. Vandendorpe, "On the Energy Cost of Communication and Cryptography in Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Wireless and Mobile Computing, Networking and Comm.,* pp. 580-585, 2008.

Chien-Ming Chen received the BS and MS degrees in computer science and information engineering from Fu-Jen Catholic University in 2001 and 2003, respectively, and the PhD degree in computer science from National Thing Hua University in 2010. He is a postdoctoral researcher at National Tsing Hwa University. His research interests include wireless sensor network, multimedia security, and applied cryptography.

Yue-Hsun Lin received the MS and PhD degrees in computer science from National Tsing Hua University in 2005 and 2010, respectively. Currently, he is a postdoctoral researcher at Intel-NTU Connected Context Computing Center at National Taiwan University. His research interests include wireless sensor network, network security, and applied cryptography.

Ya-Ching Lin received the BS degree in information management from the National Central University in 2004. Currently, she is working toward the graduate degree in the Institute of Information System and Application at National Tsing Hua University. Her current research interests include multimedia security, applied cryptography, and broadcast encryption.

Hung-Min Sun received the BS and MS degrees in applied mathematics from National Chung-Hsing University in 1988 and 1990, respectively, and the PhD degree in computer science and information engineering from National Chiao-Tung University in 1995. He was an associate professor with the Department of Information Management, Chaoyang University of Technology from 1995 to 1999, and the Department of Computer Science and Information Engineering, National Cheng-Kung University from 2000 to 2002, and the Department of Computer Science, National Cheng-Kung University from 2002 to 2008. Currently, he is working as a full professor with the Department of Computer Science, National Tsing Hua University. He has published more than 150 international journal and conference papers. He was the program cochair of 2001 National Information Security Conference, and the program committee members of many international conferences. He was the honor chairs of 2009 International Conference on Computer and Automation Engineering, 2009 International Conference on Computer Research and Development, and 2009 International Conference on Telecom Technology and Applications. He severs as the editor-in-chief in *International Journal of Digital Content Technology and its Applications,* and the editor members of many international journals including *ISRN Communications and Networking,* and *International Journal of Security, Advances in Information Sciences and Service Sciences*: an *International Journal of Research and Innovation,International Journal of Intelligent Information Processing*, and *Journal of Next Generation Information Technology*. He won many best paper awards in academic journal and conferences, including the annual best paper award in J*ournal of Information Science and Engineering* in 2003, the best paper award in MobiSys09, NSC05, NISC06, NISC07, CISC09, and ICS'2010. He won Y. Z. Hsu Scientific Paper Award, Far Eastern Y. Z. Hsu Science and Technology Memorial Foundation, 2010. His research interests include network security, cryptography, and wireless networks.