RPIDA: Recoverable Privacy-preserving Integrity-assured Data Aggregation Scheme for Wireless Sensor Networks

Lijun Yang[1], Chao Ding[2] and Meng Wu[3]
[1] School of Internet of Things, [1] School of Computer Science, Nanjing University of Posts and Telecommunications
Nanjing, Jiangsu 210046- China
[e-mail: yanglijun1119@163.com, dingchao_129@163.com]
[13] Key Lab of "Broadband Wireless Communication and Sensor Network Technology" of Ministry of Education,
Nanjing, Jiangsu 210046- China
[e-mail: wum@njupt.edu.cn]
*Corresponding author: Lijun Yang, Meng Wu

Received April 19, 2015; revised August 2, 2015; accepted September 28, 2015; published December 31, 2015

Abstract

To address the contradiction between data aggregation and data security in wireless sensor networks, a Recoverable Privacy-preserving Integrity-assured Data Aggregation (RPIDA) scheme is proposed based on privacy homomorphism and aggregate message authentication code. The proposed scheme provides both end-to-end privacy and data integrity for data aggregation in WSNs. In our scheme, the base station can recover each sensing data collected by all sensors even if these data have been aggregated by aggregators, thus can verify the integrity of all sensing data. Besides, with these individual sensing data, base station is able to perform any further operations on them, which means RPIDA is not limited in types of aggregation functions. The security analysis indicates that our proposal is resilient against typical security attacks; besides, it can detect and locate the malicious nodes in a certain range. The performance analysis shows that the proposed scheme has remarkable advantage over other asymmetric schemes in terms of computation and communication overhead. In order to evaluate the performance and the feasibility of our proposal, the prototype implementation is presented based on the TinyOS platform. The experiment results demonstrate that RPIDA is feasible and efficient for resource-constrained sensor nodes.

Keywords: wireless sensor networks, data aggregation, data privacy, data integrity, prototype implementation

A preliminary version of this paper appeared in the 2nd International Conference on Sensors, Mechatronics and Automation, Dec. 28-29, 2014, Shenzhen, China. This version includes a concrete analysis and supporting implementation results on MICAz sensor nodes. This research was supported by National Basic Research Program of China (973 Program) under Grants 2011CB302903, the National Natural Science Foundation of China (Grant No. 61100213), the Key Program of Natural Science for Universities of Jiangsu Province (Grant No.10KJA510035), and the Science and Technology Program of Nanjing (201103003).

1. Introduction

Wireless sensor networks (WSNs) are comprised of a large number of small sensor nodes which are spatially distributed across the field of interest. WSNs have been widely deployed in many areas including military, healthcare and environment, etc. Sensor nodes are usually resource-limited and power-constrained. In order to reducing communication bandwidth and energy consumption, data aggregation technology was introduced. The concept of data aggregation [1] is to aggregate multiple data by performing algebraic or statistical operations such as addition, multiplication, median, minimum, maximum, and mean of a data set, etc. Generally, aggregation is performed by intermediate node on route, such as cluster head. Finally, only the aggregated results reach the base station (BS), so the transmission cost is significantly reduced.

Unfortunately, as WSNs are usually deployed in remote and hostile environments to transmit sensitive information, sensor nodes are prone to node compromise attacks. In terms of security, data aggregation is risky. A sensor node that is compromised by an adversary can either illegally disclose the data it collects from other nodes or report arbitrary values as its aggregation results. Therefore, an adversary can compromise both the confidentiality and the integrity of the data of a large portion of the WSN by capturing some data aggregators that are close to the BS.

Actually, data aggregation techniques are in contradiction with the security goals. On one hand, data privacy prefers data to be encrypted at the source node and decrypted only by the BS in order to achieve end-to-end confidentiality. However, data aggregation usually requires each intermediate aggregator to perform aggregation on plain data so that energy efficiency is maximized. On the other hand, data integrity requires data not be altered during transmission. Whereas, performing data aggregation will inevitably change the original sensing data. Therefore, it is challenging to provide both data privacy and integrity during data aggregation.

Both data aggregation and security are critical for WSNs, so achieving secure data aggregation has been an attractive goal for researchers. Recently, a popular idea to study secure data aggregation is using privacy homomorphism (PH). In PH-based secure data aggregation schemes such as CMT [2], CDA [3], CDAMA [4], and Mykletun *et al*'s scheme [5], the aggregators directly aggregate ciphertext without decryption. However, these PH-based schemes usually limit the type of aggregation functions, and could not provide data integrity. On the other hand, some secure data aggregation schemes, such as delay aggregation [6], SIA [7], SDAP [8], and EIPDAP [9] have been proposed to solve the problem of data authentication. Although verifying the integrity hop-by-hop are relatively easy to achieve, these schemes usually require the participation of the plaintext data, and most of them cause negative effect on other performance metrics, such as communication overhead, delay and data confidentiality. Generally, providing end-to-end aggregate authentication in WSNs is difficult since messages lose entropy through aggregation, making it hard to verify the aggregate result.

End-to-end privacy and aggregate authenticity are the two major security goals for secure data aggregation of WSNs. In this paper, we aim to address the above issues all at once. We propose a Recoverable Privacy-preserving Integrity-assured Data Aggregation (RPIDA) scheme which is based on the privacy homomorphism and the aggregate message authentication code (MAC) techniques [10]. In aggregate MAC, the MAC tags on different messages are aggregated and all individual messages must be available for the verification. However, in data aggregation the messages themselves are aggregated and hence the original individual messages are not available for verification. In our proposal, the BS can recover each sensing data collected by all sensors even if these data have been aggregated by aggregators, thus can verify the integrity of all sensing data. Besides, with these individual sensing data, the BS can perform any aggregation functions on them, which means RPIDA is not limited in types of aggregation functions. The major contributions of this work can be summarized as follows:

(1) We ingeniously integrate PH and aggregate MAC techniques along with data aggregation to provide both end-to-end data privacy and data integrity for data aggregation in WSNs. To the best of our knowledge, our proposal is the first one to do so.

(2) We analysis the security properties of our proposal. The analysis results show that our proposal is resilient against general security attacks; besides, it can detect and locate the malicious nodes in a certain range.

(3) We compare our proposal with other asymmetric PH-based approaches in terms of computation and communication overhead. The comparisons results demonstrate that RPIDA is more efficient than other asymmetric PH-based schemes.

(4) We presented a prototype implementation of our proposal on the TinyOS platform, and evaluate the feasibility and performance of RPIDA deployed on the specific sensor nodes (MICAz motes). The performance evaluation data indicate that RPIDA is feasible and efficient for resource-constrained sensor nodes.

The remainder of the paper is organized as follows. The related work is presented in Section 2. In Section 3, we describe the system model, attacks model and preliminaries for understanding the proposed scheme. In Section 4, we describe the construction of RPIDA in detail. Section 5 presents the security analysis of RPIDA. In section 6, we show the performance analysis and comparisons with other schemes. In section 7, prototype implementation and performance evaluation of RPIDA is given. Finally, we conclude RPIDA in section 8.

2. Related Work

To support both data privacy and integrity during data aggregation in WSNs, a number of secure data aggregation schemes have been proposed. Ozdemir and Cam [11] propose a protocol called DAA to integrate false data detection with data aggregation and confidentiality. In DAA protocol, the monitoring nodes of every aggregator also perform data aggregation and compute the MACs for data verification; the sensors between two consecutive aggregators verify the integrity of the encrypted data rather than the plain data. However, its topology is constrained, and it could not resist node compromise attack.

He *et al*. present two data aggregation schemes, named iPDA [12] and iCPDA [13], which piggyback on SMART [14] and CPDA [14] scheme respectively. iPDA achieves privacy protection through data slicing and assembling techniques as SMART and achieves integrity through redundancy by constructing disjoint aggregation trees. In iCPDA protocol, cluster members can detect data pollution attacks through monitoring the cluster leaders, so iCPDA spends a little more message overhead to achieve data integrity. However, both schemes need much more communication and computation overheads.

Albath *et al*. [15] proposed an algorithm using homomorphic encryption and additive digital signature, which is based on Elliptic Curve Digital Signature Algorithm (ECDSA), to achieve confidentiality, integrity and availability for aggregation in WSNs. Sun *et al*. [16] proposed a recoverable concealed data aggregation scheme, named RCDA, which combines Mykletun *et al*.'s concealed data aggregation scheme [9] and Boneh *et al*.'s aggregate signature scheme [17]. RCDA could protect both privacy and integrity of aggregated data. However, these signature-based schemes have heavy computation and communication overhead. Ozdemir *et al*. [18] proposed an integrity protecting hierarchical concealed data aggregation (IPHCDA), which integrates homomorphic encryption and MAC to offer data integrity and confidentiality together. Zhou *et al*. [19] proposed a secure-enhanced data aggregation based on Elliptic Curve Cryptography (SEDA-ECC), which is based on the principles of homomorphic encryption and divide-and-conquer. However, the above two schemes are not practical for large scale network due to its high costs. Papadopoulos *et al*. [20] proposed an exact aggregation scheme with integrity and confidentiality, named SIES, which combines the symmetric homomorphic encryption and secret sharing. Although it can cover numerous aggregates and only introduces a small amount of bandwidth consumption, the data transmission efficiency is low due to the oversize space of secret keys.

3. System Model and Preliminaries

3.1 System Model

We consider a cluster-based WSN, which is comprised of a BS and a number of sensor nodes (*SN*). Generally, BS which connects the system to the networks and users has large bandwidth, strong computing capability, and sufficient memory and stable power to support the cryptographic and routing requirements of the whole WSN. Typically, *SN*s deployed to sense and gather related data are tiny and low-cost devices, hence *SN*s are limited on computation, storage and communication capability. After deployment, all *SN*s are divided in several clusters. *SN*s in the same cluster select one of them as the cluster head (*CH*), which is responsible for collecting and aggregating sensor data from *SN*s within the same cluster and finally sends aggregated results to BS. **Fig. 1** shows a typical cluster-based WSN.

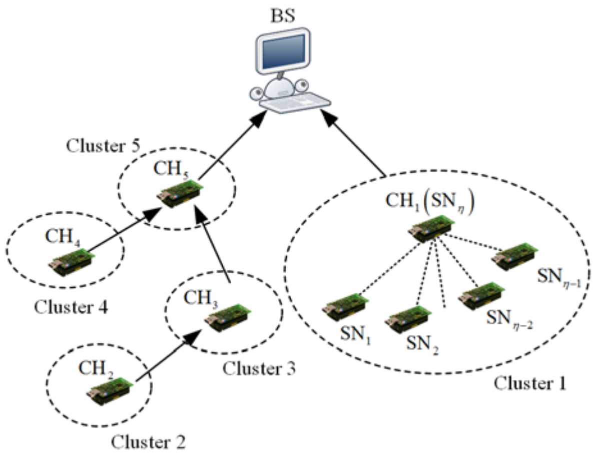

Fig. 1. Typical cluster-based WSN

3.2 Attack Model

The attack model is defined according to the ability of adversaries. In our attack model, the adversaries are able to eavesdrop on transmitted packets, forge or inject the false data and compromise secrets in sensor nodes or cluster heads (aggregators). Then, according to the adversaries' purposes and abilities we present the targeted attacks, which are the most possible attacks against any secure data aggregation scheme summarized in [21].

(1) Ciphertext analysis

Actually this is the most basic attack. In ciphertext analysis, the adversary tries to obtain information or deduce the key material only by analyzing the ciphertext. A secure cryptographic system must ensure that it is impossible to obtain any inappropriate information (statistical information, plaintext, key), and an attacker cannot determine whether a ciphertext corresponds to a particular plaintext or not.

(2) Known plaintext attacks

Given some known plaintext and corresponding ciphertext, the adversary attempts to determine some secret information. In this kind of attack, the adversary aims at the deduction of the secret key or at gathering some additional information for further attacks with known plaintext and corresponding ciphertext.

(3) Malleability

The idea of malleability attack is to change the content of a valid ciphertext without leaving marks. A simple variation of this attack is randomly generating false ciphertext that are syntactically correct to harm the system. Due to the algebraic properties of PH, the adversary may alter a valid ciphertext without knowing the content and drawing attention. Thus, PH-based schemes are very vulnerable to malleability attack.

(4) Forgery

In forgery attack, the adversary is able to generate an appropriate ciphertext, which contains a specific content. The adversary could simply replace the actual packets with the forged one, instead of modifying the existing one.

(5) Unauthorized aggregation

This kind of attack is defined under a strong assumption, which assumes that no node is captured. The attacker is able to perform unauthorized aggregation without any additional information, so as to inject the false aggregation result into the network for the detriment of the system.

(6) Node compromise attacks

The adversary is able to compromise sensor nodes or aggregator nodes. If the adversary captures sensor nodes and achieves their secrets, he may launch attacks such as modify or fake the sensing data, or impersonate other sensors. If the adversary compromises an aggregator node, he may easily eavesdrop, modify the aggregated data, and launch malleability attacks.

3.3 Privacy Homomorphism

A privacy homomorphism is an encryption transformation that allows direct computation on encrypted data. Assume $E_K(\cdot)$ is an encryption function and $D_K(\cdot)$ is the corresponding decryption function, where K is the key space. Let x and y be two plaintext. If under some operation $\circ$ there exists an efficient algorithm Alg that satisfies the follow equation

$$\mathrm{Alg}\left(E_K(x), E_K(y)\right) = E_K(x \circ y) \tag{1}$$

the encryption transformation $E_K(\cdot)$ is a privacy homomorphism (or homomorphic encryption) under the operation $\circ$.

Privacy homomorphism encryption can be achieved using symmetric or asymmetric cryptography. However, symmetric cryptography-based privacy homomorphism has been shown to be insecure in chosen plaintext attacks for some specific parameter settings [22]. Therefore, for mission critical networks, asymmetric cryptography-based privacy homomorphism should be used instead of symmetric cryptography-based privacy homomorphism.

Considering that asymmetric cryptography-based privacy homomorphism incurs high computational overhead, we employ the elliptic curve ElGamal (EC-EG) cryptosystem, which is shown to be one of the most promising homomorphic encryption for WSNs by the experiment results in [9]. The EC-EG scheme provides additive homomorphism, and is consist of three polynomial algorithms, i.e. *KeyGen*(τ), *Enc*(m, Y), *Dec*(C, x). Let + and * respectively denotes addition and scalar point multiplication operation over a cycle group of points on a given elliptic curve.

KeyGen(τ): τ is a given a security parameter.

(1) construct an elliptic curve E over a finite field F_p, where p is a large prime, and the order of E, $\#E\left(\mathsf{F}_p\right)$, has a large prime factor; choose an arbitrary generator P of $E\left(\mathsf{F}_p\right)$ with the prime order n;

(2) choose randomly $x \in_R \left[1, n-1\right]$ as private key, and compute $Y = xP$ as public key

(3) publish the 4-tuple $D = (p, E, P, n)$ as the system parameter, and return key pair (x, Y).

Enc(m, Y): given a plaintext message $m \in \left[0, p-1\right]$, public key Y and parameter D, it encrypts m with public key Y.

(1) map the message m into a point on the elliptic curve, $M = map(m)$;

(2) randomly choose $k \in_R \left[1, n-1\right]$;

(3) compute $R = k{*}P$, $S = M + k{*}Y$, and return cipher $C = (R, S)$.

Dec(C, x): given parameter D, private key x, and cipher C, it decrypts cipher C with private key x where $C = (R, S)$.

(1) compute $M = - x{*}R + S = - x{*}k{*}P + M + x{*}k{*}P$;

(2) reverse m through $m = rmap(M)$;

(3) return the plaintext m.

the function *map*() maps the plaintext m into a curve point M, and reverse function *rmap*(M) maps a given point M to the original plaintext m. we employ the function $map: m \rightarrow m * P$ to satisfy the desired additive homomorphic property, since

$$map\left(m_1 + \cdots + m_n\right) = \left(m_1 + \cdots + m_n\right) * P = m_1 * P + \cdots + m_n * P = map\left(m_1\right) + \cdots map\left(m_n\right) \quad (2)$$

3.4 Aggregate Message Authentication Codes

An aggregate message authentication code (MAC) scheme is a MAC that has the property: multiple MAC tags computed by (possibly) different senders on multiple (possibly different) messages, can be aggregated into a single tag that can still be verified by a recipient who shares a distinct key with each sender. Technically, aggregate MAC can be constructed from any standard message authentication code. In this paper, we employ a simple aggregate MAC scheme proposed in [10], which consists of a tuple of probabilistic polynomial-time algorithms (*Mac*, *Agg*, *Vrfy*).

(1) Authentication algorithm *Mac*: upon input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^*$, algorithm *Mac* outputs a tag on message m. We denote this procedure by $\text{tag} \leftarrow Mac_k(m)$.

(2) Aggregation algorithm *Agg*: given tags $\text{tag}_1, \cdots, \text{tag}_l$, associated with message/key pairs (m_i, k_i), where $i = 1, \cdots, l$, algorithm *Agg* aggregates these tags by computing the XOR of all the tags, and outputs a new tag, i.e. $\text{tag} = \text{tag}_1 \oplus \text{tag}_2 \oplus \cdots \oplus \text{tag}_l$.

(3) Verification algorithm *Vrfy*: upon receiving a set of message/key pairs (m_i, k_i), where $i = 1, \cdots, l$, and the corresponding aggregate MAC tag, algorithm *Vrfy* computes $\text{tag}' = \bigoplus_{i=1}^{l} \text{Mac}_{k_i}(m_i)$, and outputs 1 denoting acceptance if and only if $\text{tag}' = \text{tag}$.

The above aggregate MAC algorithm is existentially unforgeable under an adaptive chosen message attack. For the detailed formal security proof, please refer to the literature [10].

4. RPIDA: Recoverable Privacy-preserving Integrity-assured Data Aggregation Scheme for WSNs

4.1 The Proposed Scheme

In this section, we proposed a recoverable privacy-preserving Integrity-assured data aggregation scheme named RPIDA based on the privacy homomorphism and aggregate MAC techniques. The proposed scheme is composed of four polynomial algorithms: *Setup*, *Encrypt-MAC*, *Aggregate* and *Decrypt-Verify*. Before network deployment, the management system runs *Setup* algorithm offline to generate necessary parameters and key materials and preload them into the BS and each sensor node. After deployment, all sensor nodes are divided into several clusters. While a sensor node needs to transmit sensing data to its *CH*, it executes *Encrypt-MAC* and sends the resulting ciphertext and MAC tag to its *CH*. After receiving all the packets from member nodes, *CH* performs *Aggregate* to aggregate all the ciphertext and MAC tags, and then sends the aggregated data to BS. Once BS receives all the data during a sampling period, it performs *Decrypt-Verify* algorithm. BS decrypts the aggregated ciphertext to retrieve each individual sensing data, and then verifies the integrity of each sensing data with the aggregated MAC tag. In order to depict the proposed scheme obviously, the notations we used are listed in **Table 1**.

Table 1. Notations

Symbol	Description	Symbol	Description
BS	Base station	CH	Cluster head
SN_i	Sensor node i	$data_i$	Sensing data of node i
c_i	Ciphertext of sensor node i	C	Aggregated ciphertext
l	Bit-length of $data_i$	m_i	Encoded result of $data_i$
tag_i	MAC of $data_i$	Tag	Aggregated MAC tag
0^β	β serial 0 bits, i.e., 0^3=000	$m[u, v]$	Substring of m from index u to v, e.g., $m=1101_2$, $m[0,1]=01_2$
$\|\|$	Concatenation	(x, Y)	Key pair of BS

To present RPIDA in a simple way, we consider the case that there is only one BS in the sensor network. We choose Cluster 1(see **Fig. 1**) as an example. Without loss of generality, we assume that there are η sensor nodes i.e. $SN_1, SN_2, \cdots, SN_\eta$, and SN_η is selected as *CH* of this cluster. The detailed procedures are listed as follows.

Setup: takes a security parameter τ as input, and returns the master key and system parameters. Before network deployment, the management system does the following.

(1) chooses an appropriate elliptic curve, and runs the *KeyGen*(τ) algorithm in EC-EG scheme to generate elliptic curve parameter $D = (p, E, P, n)$ and the key pair (x, Y) for BS;

(2) generates secret key sk_i for each sensor SN_i, which is shared with BS;

(3) preloads system parameters $\langle D, Y, sk_i \rangle$ into each sensor SN_i, and $\langle D, x, Y, sk_i \rangle$ into BS.

Encrypt-MAC: takes sensing data as input, and returns ciphertext and MAC tag. Before sending sensing data to it *CH*, the sensor node does the following steps.

(1) generates MAC tag $tag_i = MAC_{sk_i}(data_i)$;

(2) encodes the sensing data, encode$(data_i)$: $m_i = data_i \,||\, 0^\beta$, where $\beta = l \cdot (i-1)$;

(3) maps the encoded message m_i to a point on the curve: $M_i = map(m_i)$;

(4) computes the ciphertext, i.e.

$$c_i = (R_i, S_i) = (k_i * P, M_i + k_i * Y) \tag{3}$$

where $k_i \in_R [1, n-1]$;

(5) sends (c_i, tag_i) to the *CH*.

Aggregate: takes receiving messages as input, and returns aggregated results. *CH* runs this procedure after it has gathered all ciphertext-tag pairs of member nodes.

(1) computes the aggregated ciphertext for the η-1 receiving ciphertext $c_1, \cdots, c_{\eta-1}$, i.e.

$$C = \sum_{i=1}^{\eta-1} c_i = \left(\sum_{i=1}^{\eta-1} R_i, \sum_{i=1}^{\eta-1} S_i\right) \tag{4}$$

(2) computes the aggregated MAC tags, i.e.

$$Tag = \oplus_{i=1}^{i=\eta-1} tag_i = \oplus_{i=1}^{i=\eta-1} MAC_{sk_i}(data_i) \tag{5}$$

(3) sends the aggregated results (C, Tag) to the BS via multi-hop fashion.

Decrypt-Verify: takes the aggregated results as input, and returns individual sensing data. On receiving (C, Tag) from *CH*, BS performs this procedure to recover each sensing data and verify the data integrity.

(1) BS obtains the aggregated plaintext *M* by decrypting with its private key *x*, i.e.

$$M = -x * R + S = M_1 + \cdots + M_{\eta-1} \tag{6}$$

(2) BS computes the message $m = rmap(M) = m_1 + \cdots + m_{\eta-1}$;

(3) BS obtains each sensing data from *m* by performing the decode function,

$$Decode(m, \eta-1, l) = \{data_i = m[(i-1) \cdot l, i \cdot l - 1]\} \tag{7}$$

where $i = 1, \cdots, \eta-1$, and the function *m*[*u*, *v*] means getting substring of *m* from index *u* to

v, e.g., $m=1101_2$, $m[0,1]=01_2$;

(4) BS computes MAC for each sensing data with the secret key sk_i shared with each node, and validates whether the equation $Tag = \oplus_{i=1}^{i=\eta-1} MAC_{sk_i}\left(data_i\right)$ holds or not. If the equation holds BS accepts the data; otherwise, rejects.

At the same time, the BS also receives aggregated ciphertext and tag pairs from other clusters. According to the final aggregated results, BS could retrieve individual sensing data collected by sensor nodes, and verify the data integrity. Thus, BS can perform any operations on these original data on demand without the limitation of types of aggregation functions.

4.2 An Illustrative Example

In this section, an example is given to demonstrate how the RPIDA scheme works. For simplicity, we assume that cluster 1 comprises 5 sensor nodes denoted as $\left\{SN_1,\cdots,SN_5\right\}$, and SN_5 is chosen as *CH*. Assume that the sensing data of each sensor node is $data_1 = 7, data_2 = 5, data_3 = 8, data_4 = 10$. Considering that 4 bits are sufficient to represent values in the example, length l is set as 4. Each member node performs the *Encrypt-MAC* function. Take SN_3 for instance, it takes the following steps.

(1) computes the MAC tag $tag_3 = MAC_{sk_3}\left(data_3\right)$;

(2) encodes the sensing data, encode $\left(data_3\right): m_3 = data_3 \,||\, 0^{\beta} = \left(100000000000\right)_2$, where $\beta = l \cdot \left(i-1\right) = 8$;

(3) maps m_3 to a point on the curve: $M_3 = map\left(m_3\right) = m_3 {*} P$;

(4) computes the ciphertext $c_3 = \left(R_3, S_3\right)$ with the public key of BS;

(5) sends $\left(c_3, tag_3\right)$ to the *CH*.

Similarly, other sensor nodes also execute *Encrypt-MAC* function and send their $\left(c_i, tag_i\right)$ pairs to *CH*. Once receiving data packets from all member nodes, *CH* performs *Aggregate* function as follows.

(1) computes the aggregated ciphertext $C = \sum_{i=1}^{4} c_i = c_1 + c_2 + c_3 + c_4$;

(2) computes the aggregated MAC tags, $Tag = \oplus_{i=1}^{i=4} tag_i = tag_1 \oplus tag_2 \oplus tag_3 \oplus tag_4$;

(3) sends the aggregated results $\left(C, Tag\right)$ to BS.

Because $m_1 = \left(0111\right)_2$, $m_2 = \left(01010000\right)_2$ and $m_4 = \left(1010000000000000\right)_2$, the aggregated plaintext value corresponding to the aggregated ciphertext C should be $m_1 + m_2 + m_3 + m_4 = \left(1010100001010111\right)_2$. Once receiving the aggregated results $\left(C, Tag\right)$ from *CH*, BS performs *Decrypt-Verify* function to retrieve each sensing data and verify the data integrity.

(1) BS achieves the aggregated plaintext M through decrypting C, and computes the corresponding plaintext message $m = \left(1010100001010111\right)_2$;

(2) BS performs the decode function $Decode(m,4,4)=\{data_i = m[4\cdot(i-1), 4\cdot i-1]\}$, where $i=1,\cdots,4$, to extract individual sensing data, i.e. $data_1=(0111)_2=7$, $data_2=(0101)_2=5$, $data_3=(1000)_2=8$ and $data_4=(1010)_2=10$;

BS computes MAC for each $data_i$ with the secret key sk_i shared with each node, and validates whether the equation $Tag=\oplus_{i=1}^{i=4} MAC_{sk_i}(data_i)$ holds or not.

5. Security Analysis

In this section, we show our proposal could provide both end-to-end data privacy and data integrity while performing data aggregation for WSNs. On one hand, all sensing data are encrypted with public key of BS before transmitting, thus are concealed from intermediate nodes. The proposed scheme employs EC-EG encryption algorithm, which is based on elliptic curve discrete logarithm problem (ECDLP). Compared with factoring problem based cryptosystems, such as RSA, it provides the same security strength with shorter key and ciphertext. On the other hand, our proposal generates the corresponding MAC for each sensing data. Thus, the attacker is not able to modify data or inject forged data since he cannot compute MAC for forged data without secret keys. Next, we show our proposal is resistant to attacks that are described in [21], where authors summarize all possible attacks against any concealed data aggregation scheme. In addition, RPIDA is able to detect and locate malicious nodes in a certain range.

5.1 Ciphertext Analysis Attack

Ciphertext analysis attack is the most basic attack in which the adversary tries to obtain key material or plaintexts information only by interpreting cipher texts. RPIDA is robust to ciphertext analysis since the homomorphic encryption algorithm employed depends on the ECDLP.

5.2 Known Plaintext Attack

In this kind of attack, the adversary tries to deduce the secret key or gathering some additional information for further attacks with known plaintexts and corresponding cipher texts. The RPIDA scheme is resistant to known plaintext attacks, because its encryption process relies on random numbers, and the resulting ciphertext is probabilistic.

5.3 Malleability & Forgery Attacks

Both malleability and forgery attacks take the data integrity as the target. RPIDA employs the aggregate MAC scheme, which is existentially unforgeable under an adaptive chosen message attack [10], to resist attacks targeting data integrity. In our proposal, each sensor node generates corresponding MAC tag for its sensing data, and the cluster head aggregates all MAC tags into an aggregated MAC tag. Considering that RPIDA employs an unforgeable MAC algorithm (such as HMAC), the adversary is not able to successfully alert or forge the content of a ciphertext since he cannot generate a valid MAC tag for the false ciphertext.

5.4 Unauthorized Aggregation Attack

In this kind of attack, an adversary cannot aggregate the ciphertext or modify aggregate results if he does not compromise any sensor nodes or cluster heads. In our proposal, the encryption scheme is based on elliptic curve cryptography, and aggregation operation requires the point addition function. However, the adversary cannot perform unauthorized aggregation without realizing how the curve is constructed. Thus, the RPIDA scheme is resistant against unauthorized aggregation attack.

5.5 Node Compromise Attacks

If an adversary has the ability to compromise sensor nodes, he may launch attacks such as modify or fake the sensing data, or impersonate other sensors. We consider the following three cases. First, the adversary compromises a sensor node and acts it as a legal one. However, if the captured sensor nodes still behave normally, it is unfeasible to detect the attack in all existing detection schemes in WSNs. Second, the captured sensor node acts maliciously, e.g. sending false data. If the value of forged message is in a reasonable range, detection is still infeasible in previous secure aggregation schemes [7; 9]. Fortunately, if the value of the false data is out of reasonable range, it can be detected by our scheme (the details are mentioned in section 5.6). Third, the adversary tries to impersonate other sensor nodes. In our scheme, the malicious sensor node cannot impersonate any other legal nodes without the corresponding secret keys.

On the other hand, if an attacker captures a cluster head, he may eavesdrop and modify the aggregated data. However, he cannot decrypt the aggregated ciphertext or each individual ciphertext since the private key of BS is not stored in any sensor node or cluster head. Further more, the attacker cannot forge the aggregated MAC tag for the modified aggregated ciphertext without the secret keys of sensor nodes. Besides, the compromised cluster head may selectively drop some cipher texts and MAC tags when performing the *Aggregate* algorithm. Fortunately, there has been some research [23] aiming at defending against this attack.

5.6 Malicious Node Detection

Our proposal could provide an additional security service, i.e. detecting and then locating the malicious sensor node, which sent the forged message out of reasonable range. We take an example to illustrate this functionality. We assume length l is set as 4 and the node SN_2 in cluster 1 is compromised. If $data_2 = 5 = (0101)_2$, and the corresponding plaintext is modified as $m_2 = (01011000)_2$, we can find m_2 is illegal because the last four bits are not all zeros. While the BS receives the final aggregated result (C, Tag), it performs *Decrypt-Verify* procedure. Undoubtedly, verification would be failed. In order to locate malicious sensor nodes, the cluster head CH_1 is required to pass each (c_i, tag_i) pair to BS without aggregation. Then, BS extracts individual m_i and finds the format of m_2 is incorrect, thus identifies SN_2 as a malicious node. However, if the original m_2 is modified as $m_2 = (11010000)_2$, BS cannot detect the malicious node since the value in forged message is in the reasonable range. In conclusion, abnormal messages will be identified by the BS since each data should be verified in the *Decrypt-Verify* procedure. The value of fake data is constrained in a relatively small range.

6. Performance Analysis and Comparisons

In this section, we analyze the performance of our proposal in terms of computation and communication overhead respectively, and compare it with other schemes. Generally, symmetric key-based homomorphic schemes are more efficient than asymmetric ones; however, the security of symmetric schemes is weaker than that of asymmetric ones. For the sake of fairness, we choose three asymmetric privacy homomorphism based secure data aggregation schemes for comparison, i.e. EC-EG [9], which only provides data confidentiality; RCDA [16] and IPHCDA [18], which support data privacy and integrity protection.

6.1 Computation Overhead

We employ the EC-EG homomorphism encryption algorithm to provide data end-to-end privacy, the operation parameters are selected from the elliptic curve defined on the finite field F_p. The decryption and integrity validation operations are run on the BS, which is a resource-rich node. Thus, we are mainly concerned with the cost on the sensor node and cluster head (aggregator). The sensor node runs the *Encrypt-MAC* function, which needs two $|p|$-bit scalar point multiplication, one $|p|$-bit point addition, and one hash operations. The cluster head runs *Aggregate* function, which needs 2(η-2) $|p|$-bit point addition, and (η-2) XOR operations when the number of cluster member is (η-1).

In order to achieve the 1024-bit RSA equivalent security, parameter p is selected as a 160-bit large prime. Assuming the same system model, the parameters of the three schemes for comparison, i.e. EC-EG, RCDA and IPHCDA, are also selected based on the same security level. The comparison of computation overhead of the three secure data aggregation schemes with our proposal is shown in **Table 2**, where the computation overhead of *CH* is the overhead of aggregating data from two cluster members; H is hash function; M_{160}, M_{271} and M_n respectively denote 160-bit, 271-bit and n-bit scalar point multiplication; A_{160}, A_{271}, and A_n respectively denote 160-bit, 271-bit, and n-bit elliptic curve point addition; n is $(k+1)|q|$, and q is a 341-bit prime.

Table 2. Comparison of computation overhead.

Scheme	Sensor Node (SN)	Cluster Head (CH)
EC-EG	$2M_{160}+1A_{160}$	$2A_{160}$
RCDA	$2M_{160}+1A_{160}+1M_{271}+1H$	$2A_{160}+1A_{271}$
IPHCDA	$2M_n+1A_n+1H$	$2A_n$
Our proposal	$2M_{160}+1A_{160}+1H$	$2A_{160}+\mathrm{XOR}$

Since we are comparing schemes built upon different parameters, it becomes important to have a base unit of measurement common to all schemes in order to fairly compare them. We choose that unit to be 1024-bit modular multiplications and follow the same methodology for comparison as in [24]. We convert the elliptic curve scalar point multiplication and point addition of different parameters to 1024-bit modular multiplications.

The comparison results of computation cost for the sensor node and the cluster node are shown in **Fig. 2** and **Fig. 3** respectively. For the sensor node, our proposal and the EC-EG scheme cost the least, since compared to EC-EG our proposal only needs one more hash function operation, which is negligible compared to the public key operations such as scalar point multiplication and point addition. The computational overhead of our proposal is only 29.2% of the RCDA scheme, and much lower than the IPHCDA scheme. For the cluster head, our proposal and the EC-EG scheme are comparable and cost the least. The aggregation overhead of our proposal is only 40.6% of the RCDA, and respectively 2.4%, 1.4% of the IPHCDA when $k = 2$ and $k = 3$. Therefore, our proposal has a significant advantage over the similar type of schemes in terms of computation overhead.

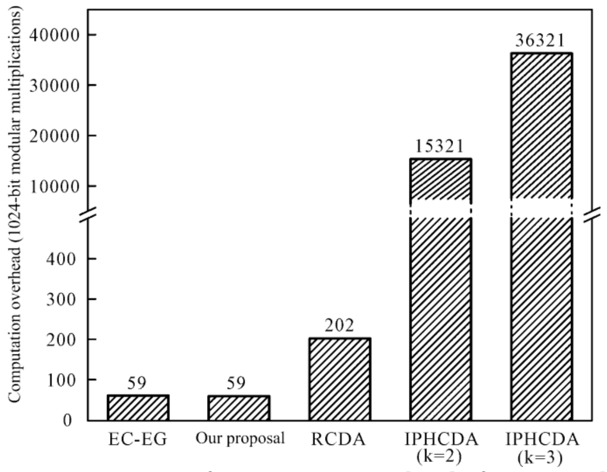

Fig. 2. Comparison of computation overhead of sensor node

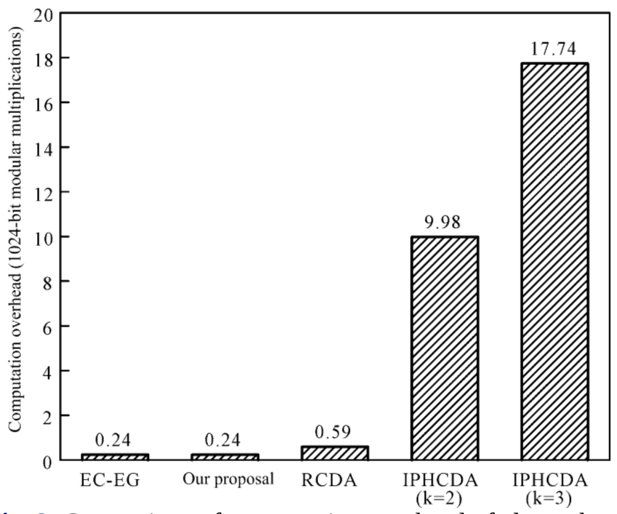

Fig. 3. Comparison of computation overhead of cluster head

6.2 Communication Overhead

Communication Overhead. To evaluate the communication cost, we first compute the ciphertext size of each scheme. For our proposal, we choose the elliptic curve $E(\mathbb{F}_{160})$, where one point (x, y) occupied 2×160 bit. With the help of node compression techniques, the point can be compressed to 161bits (21 byte). In our scheme, the message sent by each sensor node is in the form of $(c_i \| \text{tag}_i)$, which is comprised of a ciphertext and a MAC tag. The ciphertext $c_i = (R_i, S_i)$ occupies 42 bytes. The MAC employs HMAC algorithm, thus tag_i is 128 bits (16 byte). Therefore, the message sent by each node has a length of 58 bytes. Here we do not take the extra overhead arose from TinyOS packet encapsulation into consideration. As for the cluster head, it performs *Aggregate* function and sends the aggregated result to BS via a multi-hop fashion. The aggregated message (C, Tag) is also a constant, which is in the same form with $(c_i \| \text{tag}_i)$. Therefore the communication cost of each cluster head and each sensor node are the same. At the same security level, we select the elliptic curve defined on the finite field $\mathsf{F}_{2^{271}}$ for the signature algorithm employed in the RCDA scheme. The signature occupies 34 bytes, thus the message sent by each node in the RCDA scheme is 76 bytes. In the IPHCDA scheme, the ciphertext has a length of $(k+1) \times |q|$ bits, where q is a 341-bit prime, and k denotes the number of node deployment areas. The comparison of ciphertext size of the three secure data aggregation schemes with our proposal is shown in **Fig. 4**.

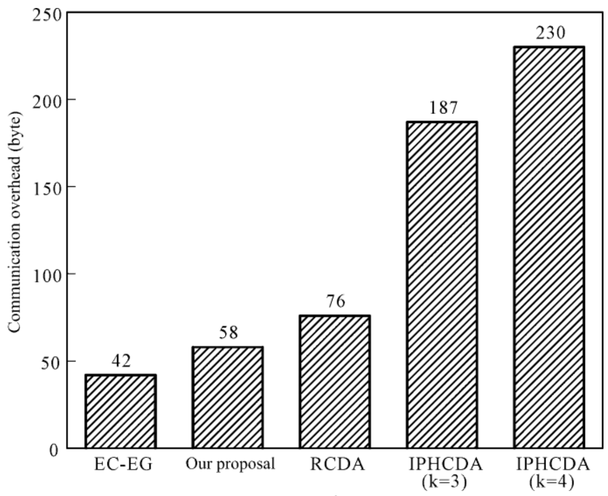

Fig. 4. Comparison of ciphertext size

According to the results in **Fig. 4**, our proposal outperforms the RCDA and the IPHCDA scheme. This is due to the employment of more efficient privacy homomorphism and authentication algorithms. Compared with the EC-EG scheme, our scheme not only provides the integrity protection, but also supports the recovery of sensing data from aggregated results, adding only 16 bytes communication cost.

Simulation Results. To verify the theoretical analysis results in the above section, we establish a simulating network model as depicted in **Fig.1** using TOSSIM. In the simulation network, 120 sensor nodes are randomly distributed over a 200 m × 200 m rectangle region where the BS is placed at the central point. The entire network is divided into 3 clusters where each cluster has a cluster head, performing the function of aggregator. Due to the harsh communication environment, a retransmission mechanism is employed to mitigate collision. In this mechanism, each sensor node waits a random time to send data, and retransmits the lost packets for up to 5 times when the collision occurs.

To compare the communication cost, the total data transmission amount in the network is measured for our proposal, EC-EG, RCDA and IPHCDA schemes. The simulation of each scheme runs for 600 seconds, where each aggregation round lasts for 2.5 seconds. Each simulation repeats 10 times for an average result. All the results are shown in **Fig. 5**. From **Fig. 5** we notice that the total data transmission of IPHCDA is less than that of our proposal when the number of deployment areas is 3 (*k*=3). That is because in case of IPHCDA, cluster heads are allowed to aggregate the received data from other aggregators, thereby achieving hierarchical data aggregation, whereas our proposal doesn't support multilevel aggregation. However, our proposal takes on an obvious advantage over IPHCDA when k>3. That means when k>3 our proposal should be used instead of IPHCDA.

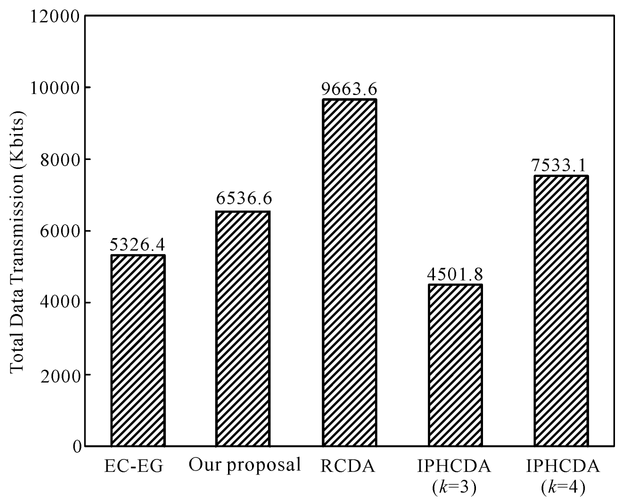

Fig. 5. Comparison of total amount of data transmission in the network

7. Implementation and Performance Evaluation

7.1 Prototype Implementation

In order to evaluate the feasibility and performance of our proposal deployed on the specific sensor node, we have implemented a prototype of RPIDA on the TinyOS platform. Current implementation involves the offline operations performed on the management system (a PC with TinyOS installed), the online operations performed on the BS node, the cluster head and the sensor nodes. The management system performs the *Setup* step and preloads the system parameters and key materials into each sensor node. The sensor node, the cluster head and the BS runs the *Encrypt-MAC* function, the *Aggregate* function and the *Decrypt-Verify* function, respectively.

All the codes running online have been developed in NesC, a C-like language for developing applications in TinyOS. The prototype implementation of RPIDA is based on the RELIC cryptographic toolkit [25], which provides the necessary tools to perform operations on elliptic curves. RELIC is a publicly available and open source library which is specifically designed for resource-constrained devices.

As recommended by NIST, we adopt an 80-bit security level (RSA-1024 equivalent). Thus we select the standardized SECG Secp160r1 curve defined on the prime finite field $\mathbb{F}_{160}$ for the prototype implementation. We use the left-to-right NAF algorithm [26] for general scalar point multiplication, and use the single-table COMB pre-computation method [27] to accelerate the fixed point multiplication.

As shown in **Fig. 6**, the core functions of RPIDA are modeled as several TinyOS components, as shown in the gray region of the diagram. The *SecPrimM* component encapsulates the necessary security primitives from RELIC library, and provides the calling interface for other components. By calling the fundamental cryptographic operations from the *SecPrimM* interface, the *EcEgM* and *HMacM* components provide the EC-EG encryption/decryption and MAC generation/verification services, respectively. The *CodecM* component takes the responsibility for coding/decoding the sensor data. The *EncMacM* component, which is deployed on the sensor nodes, provides coding, encryption and MAC generation for sensor data by integrating the functions of *CodecM*, *HMacM* and *EcEgM*. Similarly, the *DecVerM* component, which is deployed on the BS, provides decoding, decryption and MAC verification of aggregated data for the aggregated data. Besides, the *AggM* component, which is deployed on the cluster head, provides aggregation of cipher texts and MAC tags received from the member nodes. The above 7 components provides calling interfaces of their corresponding functions, and *RpidaC* component makes use of these interfaces to provide the integrated interface *Rpida*.

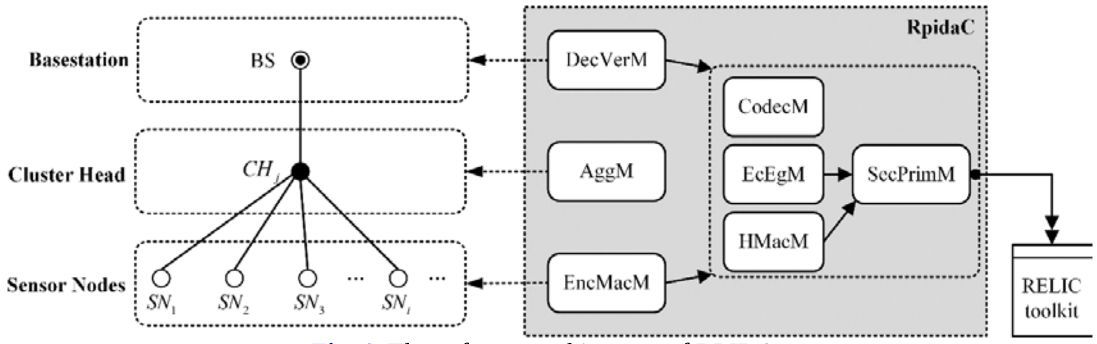

Fig. 6. The software architecture of RPIDA

7.2 Performance Evaluation

We evaluate the feasibility and performance of RPIDA on the Crossbow MICAz sensor motes, a popular choice among the research community. The MICAz is based on the low-power 8-bit microcontroller ATmegal128L, which operates at 7.3828MHz and offers 4KB of RAM memory and 128KB of program space.

We suppose that the BS is a more powerful device, and is equipped with continuous power supply. We are mainly concerned with the computation cost and the memory occupation of RPIDA running on the sensor node and the cluster head (aggregator). Thus, here we only discuss the online performance of our protocol. We use Avrora [28], an instruction-level AVR-microcontroller-oriented analysis tool, to evaluate the time and energy performance on the MICAz motes. We measured the memory occupation by examining the binary image of the programs of RPIDA in detail.

Computation Overhead. The experimental computation performance of RPIDA running on the MICAz mote is shown in **Table 3**. *Initialization* comprises the parameter configuration of RELIC and the load of system parameters. This step is only executed once at the bootstrap of the nodes.

Table 3. The computation overhead of RPIDA.

	Initialization	**Encrypt-MAC (Sensor Node)**	**Aggregate (Cluster Head)**
CPU Cycles	8,595,851	5,882,565	536,263
Energy (mJ)	26.46	15.48	1.65
Time (s)	1.166	0.798	0.073

As shown in **Table 3**, in our scheme the sensor node takes 0.798 seconds and consumes 15.48mJ to encrypt one sensor data and compute a MAC tag. In general, the data sample period of sensor networks is about 15~60 seconds, and the RPIDA protocol is able to fully meet the requirement. The cluster head takes 0.073 seconds and consumes 1.65mJ to aggregate data from two cluster members. That means the cluster head takes $0.073\times(\eta-2)$ seconds and consumes $1.65\times(\eta-2)$ mJ when the number of cluster member is (η-1). Besides, each node takes 1.166 seconds and 26.46mJ to initialize the hardware and software.

Memory Overhead. In our scheme, each node needs to store the RELIC codes, the TinyOS codes, the secret key sk_i shared with BS, the public key of BS, and the sensor data to be handled. Since the sensor nodes only handle a fixed quantity of data in each aggregating round, the memory overhead of the sensor node is fixed and can be considered a constant. On the other hand, the cluster head needs to store the ciphertext and MAC tags from cluster members, thus its memory overhead is linear with the number of cluster member.

Table 4. Memory overhead of RPIDA (Byte).

Role	**RAM**		**ROM**
	.data	.bss	.text
Sensor Node	130	1075	53,922
Cluster Head	80	1060	51,432

Through statistical analysis of the object file of RPIDA program, we obtain the memory occupation of our protocol running on the MICAz mote, as shown in **Table 4**, where the *.data* segment consumes the static memory of RAM, *.bss* segment consumes the dynamic memory of RAM, and the *.text* segment consumes the memory of ROM. As the data shown in **Table 4**, the sensor node consumes 1205 bytes RAM and 53,922 byte ROM, and the cluster head consumes about $1060+(42+16)\times(\eta-3)$ bytes RAM and 51,432 bytes ROM when the number of cluster member is (η-1). Note that the RAM occupation in cluster head may seem a little high for MICAz motes, but most RAM occupation comes from the stack and is only reserved for the duration of aggregation, which means once the aggregation operation is finished, those memory is available for other operations. Furthermore, compared with hop-by-hop encryption and integrity verification protocols, the cluster head in our scheme do not need to store the pairwise key with every member in its cluster, which dramatically reduces the memory occupation in RAM. Hence, we believe that the memory consumption of the proposed RPIDA is acceptable and reasonable for resource-constrained sensor networks.

8. Conclusion

In order to achieve privacy-preserving and aggregate integrity simultaneously for the data aggregate pattern in Wireless Sensor Networks, we propose a recoverable privacy-preserving integrity-assured data aggregation scheme based on privacy homomorphism and aggregate MAC. In our scheme, a BS is able to retrieve each original sensing data even if these data have been aggregated by the intermediate aggregators, which make it possible to verify the integrity of all sening data and the aggregated data. Besides, with these individual sensing data, BS is capable to perform any further operations over them on demand, which means RPIDA is not limited in types of aggregation functions. The security analysis shows that our proposal is resilient against typical security attacks; besides, it can detect and locate the malicious nodes in a certain range. The performance analysis shows that the proposed scheme has remarkable advantage over other asymmetric schemes in terms of computation and communication overhead. In order to evaluate the performance and the feasibility of our proposal, the prototype implementation is presented based on the TinyOS platform. The experiment results demonstrate that RPIDA is feasible and efficient for resource-constrained sensor nodes.

References

[1] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 14, no. 2, pp. 70-87, 2007. Article (CrossRef Link).

[2] C. Castelluccia, A. C.-F. Chan, E. Mykletun, and G. Tsudik, "Efficient and provably secure aggregation of encrypted data in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 5, no. 3, pp. 1-36, 2009. Article (CrossRef Link).

[3] D. Westhoff, J. Girao, and M. Acharya, "Concealed Data Aggregation for Reverse Multicast Traffic in Sensor Networks: Encryption, Key Distribution, and Routing Adaptation," *IEEE Transactions on Mobile Computing*, vol. 5, no. 10, pp. 1417-1431, 2006. Article (CrossRef Link).

[4] L. Yue-Hsun, C. Shih-Ying, and S. Hung-Min, "CDAMA: Concealed Data Aggregation Scheme for Multiple Applications in Wireless Sensor Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 7, pp. 1471-1483, 2013. Article (CrossRef Link).

[5] E. Mykletun, J. Girao, and D. Westhoff, "Public Key Based Cryptoschemes for Data Concealment in Wireless Sensor Networks," in *Proc. of 6th International Conference on Communication (ICC'06)*, pp. 2288-2295, June 11-15, 2006. Article (CrossRef Link).

[6] H. Lingxuan and D. Evans, "Secure aggregation for wireless networks," in *Proc. of 2003 symposium on Applications and the Internet Workshops (SAINT'03)*, pp. 384-391, Jan. 27-31, 2003. Article (CrossRef Link).

[7] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure information aggregation in sensor networks," in *Proc. of the 1st International Conference on Embedded Networked Sensor Systems (Sensys'03)*, pp. 255-265, Nov. 5-7, 2003. Article (CrossRef Link).

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

[9] L. Zhu, Z. Yang, M. Li, and D. Liu, "An Efficient Data Aggregation Protocol Concentrated on Data Integrity in Wireless Sensor Networks," *International Journal of Distributed Sensor Networks*, vol. 2013, no, 7, pp. 1-9, 2013. Article (CrossRef Link).

[10] J. Katz and A. Lindell, "Aggregate Message Authentication Codes," in *Proc. of the Cryptographers' Track at the RSA Conference*, pp. 155-169, April 8-11, 2008. Article (CrossRef Link).
[11] S. OzeDemir and H. Cam, "Integration of false data detection with data aggregation and confidential transmission in wireless sensor networks," *IEEE/ACM Transaction on Networking*, vol. 18, no. 3, pp. 736-749, 2010. Article (CrossRef Link).
[12] H. Wenbo, N. Hoang, L. Xue, K. Nahrstedt, and T. Abdelzaher, “iPDA: An integrity-protecting private data aggregation scheme for wireless sensor networks,” in *Proc. of IEEE Military Communications Conference (MILCOM 2008)*, pp. 1-7, Nov. 16-19, 2008. Article (CrossRef Link).
[13] H. Wenbo, L. Xue, N. Hoang, and K. Nahrstedt, "A Cluster-Based Protocol to Enforce Integrity and Preserve Privacy in Data Aggregation," in *Proc. of 29th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS Workshops '09)*, pp. 14-19, June 22-26, 2009. Article (CrossRef Link).
[14] H. Wenbo, L. Xue, N. Hoang, K. Nahrstedt, and T. T. Abdelzaher, "PDA: Privacy-Preserving Data Aggregation in Wireless Sensor Networks," in *Proc. of 26th IEEE International Conference on Computer Communications (IEEE INFOCOM 2007)*, pp. 2045-2053, May 6-12, 2007. Article (CrossRef Link).
[15] J. Albath and S. Madria, "Secure hierarchical data aggregation in wireless sensor networks," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC '09)*, pp. 1-6, April 5-8, 2009. Article (CrossRef Link).
[16] H. Sun, C. Chen, and Y. Lin, "RCDA: Recoverable concealed data aggregation for data integrity in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 4, pp. 727-734, 2011. Article (CrossRef Link).
[17] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," in *Proc. of International Conference on the Theory and Applications of Cryptographic Techniqueseedings*, pp. 416-432, May 4-8, 2003. Article (CrossRef Link).
[18] S. Ozdemir and Y. Xiao, "Integrity protecting hierarchical concealed data aggregation for wireless sensor networks," *Computer Networks*, vol. 55, no. 8, pp. 1735-1746, 2011. Article (CrossRef Link).
[19] Q. Zhou, G. Yang, and L. He, “A Secure-Enhanced Data Aggregation Based on ECC in Wireless Sensor Networks,” *Sensors (Basel, Switzerland)*, vol. 14, no. 4, pp. 6701-6721, 2014. Article (CrossRef Link).
[20] S. Papadopoulos, A. Kiayias, and D. Papadias, "Exact In-Network Aggregation with Integrity and Confidentiality," *IEEE Transactions on Knowledge & Data Engineering*, vol. 24, no. 10, pp. 1760-1773, 2012. Article (CrossRef Link).
[21] S. Peter, D. Westhoff, and C. Castelluccia, "A Survey on the Encryption of Convergecast Traffic with In-Network Processing," *IEEE Transactions on Dependable and Secure Computing*, vol. 7, no. 1, pp. 20-34, 2010. Article (CrossRef Link).
[22] D. Wagner, "Cryptanalysis of an algebraic privacy homomorphism," in *Proc. of 6th International Conference on Information Security (ISC'03)*, pp. 234-239, Oct. 1-3, 2003. Article (CrossRef Link).
[23] T. H. Hai and E.-N. Huh, "Detecting Selective Forwarding Attacks in Wireless Sensor Networks Using Two-hops Neighbor Knowledge," in *Proc. of 7th IEEE International Symposium on Network Computing and Applications (NCA)*, pp. 325-331, July 10-12, 2008. Article (CrossRef Link).
[24] N. Gura, A. Patel, A. Wander, H. Eberle, and S. Shantz, "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs," *Cryptographic Hardware and Embedded Systems - CHES 2004*. vol. 3156, pp. 119-132, 2004. Article (CrossRef Link).
[25] Diego F. Aranha. RELIC Cryptographic Toolkit [Online]. Article (CrossRef Link).
[26] B. King, "wNAF *, an Efficient Left-to-Right Signed Digit Recoding Algorithm," *Applied Cryptography and Network Security*, vol. 5037, pp. 429-445, June, 2008. Article (CrossRef Link).

[27] J. Taverne, A. Faz-Hernández, D. Aranha, F. Rodríguez-Henríquez, D. Hankerson, and J. López, "Speeding scalar multiplication over binary elliptic curves using the new carry-less multiplication instruction," *Journal of Cryptographic Engineering*, vol. 1, no. 3, pp. 187-199, 2011. Article (CrossRef Link).

[28] Titzer, B.L.; Lee, D.K.; Palsberg, J. "Avrora: Scalable sensor network simulation with precise timing," in *Proc. of Fourth International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 477-482, April 25-27, 2005. Article (CrossRef Link).

Lijun Yang received the B.S. degree in Radio and Television Engineer and the Ph.D. degree in Information and Network Security from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2007 and 2014 respectively. Currently, she is a lecture of the School of Internet of Things, Nanjing University of Posts and Telecommunications. Her main research interests include wireless sensor networks, information security, privacy preservation, and public key cryptography.

Chao Ding received the B.S. degree in 2006 from the School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, China, where he is currently pursuing his Ph.D. degree in Information and Network Security. His current research interests include applied cryptography, anomaly detection in sensor networks and implementation of real sensor platforms.

Meng Wu received his B.S., M.S. and Ph.D. degrees in Communication Engineering and Computer Science from Zhejiang University, Shanghai Jiao Tong University, Southeast University, in 1985, 1990 and 1993, respectively. Currently, he is a professor and the leading person of Information Security doctoral discipline of Nanjing University of Posts and Telecommunications. His main research areas are wireless communication and information security.