Peer-to-Peer Netw. Appl. (2018) 11:1022–1037
DOI 10.1007/s12083-017-0581-5

Data aggregation with end-to-end confidentiality and integrity for large-scale wireless sensor networks

Jie Cui[1] · **Lili Shao**[1] · **Hong Zhong**[1] · **Yan Xu**[1] · **Lu Liu**[2]

Received: 20 December 2016 / Accepted: 23 May 2017 / Published online: 17 July 2017

Abstract In wireless sensor networks, data aggregation allows in-network processing, which leads to reduced packet transmissions and reduced redundancy, and thus is helpful to prolong the overall lifetime of wireless sensor networks. In current studies, Elliptic Curve ElGamal homomorphic encryption algorithm has been widely used to protect end-to-end data confidentiality. However, these works suffer from the expensive mapping function during decryption. If the aggregated results are huge, the base station has no way to gain the original data due to the hardness of the elliptic curve discrete logarithm problem. Therefore, these schemes are unsuitable for the large-scale WSNs. In this paper, we propose a secure energy-saving data aggregation scheme designed for the large-scale WSNs. We employ Okamoto-Uchiyama homomorphic encryption algorithm to protect end-to-end data confidentiality, use MAC to achieve in-network false data filtering, and utilize the homomorphic MAC algorithm to achieve end-to-end data integrity. Two popular IEEE 802.15.4-compliant wireless sensor network platforms, Tmote Sky and iMote 2 have been used to evaluate the efficiency and feasibility of our scheme. The results demonstrate that our scheme achieved better performance in reducing energy consumption. Moreover, system delay, especially decryption delay at the base station, has been reduced when compared to other state-of-art methods.

Keywords Data aggregation · Confidentiality · Integrity · Homomorphic encryption · Large-scale wireless sensor networks

This article is part of the Topical Collection: *Special Issue on Big Data Networking*
Guest Editors: Xiaofei Liao, Song Guo, Deze Zeng, and Kun Wang

✉ Hong Zhong
zhongh@ahu.edu.cn

✉ Lu Liu
l.liu@derby.ac.uk

[1] School of Computer Science and Technology, Anhui University, Hefei 230039, China

[2] Department of Electronics, Computing and Mathematics, University of Derby, Derby DE22 1GB, UK

1 Introduction

Wireless sensor networks (WSNs) have attracted a great deal of research attention due to their wide-range of potential applications, such as environmental monitoring, health care, wildlife surveillance, accident report, etc. [1, 2]. Recently, advances in microprocessor and wireless communication technologies have enabled the deployment of large-scale WSNs to obtain fine-grained, high-precision sensing data [3]. WSNs consist of large numbers of sensor nodes constrained in storage space, battery power, and computational capability. Therefore, reducing energy consumption is a critical concern for WSNs.

Data aggregation allows in-network processing, which leads to fewer packet transmissions and reduces redundancy and thus is of benefit for prolonging the overall lifetime of WSNs [4]. With such technique, data sensed by multiple member nodes are aggregated into a single one by applying some aggregation functions such as Sum, Average, MAX, etc. and finally transmitted to the base station via the wireless link. Apparently, communication overhead is lessened since only the aggregated result is transmitted to the base station. Thus, data aggregation is beneficial to increase the WSN's overall lifetime.

However, due to the hostile and unattended environments deployed, WSNs are subject to various attacks, such as replay

attacks, injection attacks, tampering attacks and so on. As the sensor nodes in the WSNs are limited in resources, this makes existing abundant security algorithms unsuitable for resource-constrained WSNs. Therefore, ensuring security for data aggregation is a challenge.

To guarantee secure data aggregation, numerous schemes are proposed successively. The authors of [5] proposed two recoverable concealed data aggregation (CDA) schemes, RCDA-HOMO for homogeneous WSNs and RCDA-HETE for heterogeneous WSNs. In [6], the authors proposed a CDAMA scheme for multi-application environments, in which ciphertexts from different applications can be aggregated into a single one and the base station can extract application-specific data from aggregated ciphertexts by a corresponding key. Unfortunately, this scheme is not suitable for the WSNs where the number of clusters or applications is large. What is more, it does not achieve data integrity protection. Shim et al. [7] proposed a data aggregation scheme with confidentiality and integrity, which provides in-network data filtering and authorized aggregation.

These above secure data aggregation schemes use Elliptic Curve ElGamal (EC-EG) homomorphic encryption algorithm to achieve end-to-end data confidentiality, which makes them suffer from an expensive mapping function during decryption and become too costly to revert. If the aggregated results are large, the base station has no way to gain the original data due to the hardness of the elliptic curve discrete logarithm problem (ECDLP). Therefore, these schemes are unsuitable for the large-scale WSNs.

Recently, Boudia et al. [4] proposed a novel secure aggregation scheme which uses a symmetric based homomorphic encryption technique to provide end-to-end data confidentiality. However, to achieve data integrity protection, the messages require to be formed as concatenations of all messages from member nodes, i.e. $m = m_1 || \ldots || m_n$. Thus, the size of the resulting message grows linearly with the number of member nodes or cluster heads. Therefore, their recoverable sensing data approach is very inefficient if the message size is large.

To solve the above problems, we design a secure data aggregation scheme that is suitable for large-scale wireless sensor networks but still reduces the energy consumption. We employ Okamoto-Uchiyama (OU) homomorphic encryption algorithm to protect end-to-end data confidentiality, use MAC to achieve in-network false data filtering, and utilize the homomorphic MAC (H-MAC) algorithm to achieve end-to-end data integrity. Our contributions can be summarized as follows:

1. We correct some errors discovered in [8]; we amend the security flaw found in [9] and strengthen security for the usage of the H-MAC algorithm.
2. We propose a secure data aggregation scheme suitable for the large-scale WSNs. In previous schemes, decryption efficiency is not high due to the hardness of ECDLP and even the base station has no way to decrypt ciphertexts if the aggregated results are large, which easily makes the system paralyzed since the base station is busy in decrypting messages. Fortunately, our scheme makes it possible for the base station to quickly decrypt ciphertexts and obtain the sensing data even though the aggregated results are very large.
3. End-to-end data confidentiality and integrity are provided using the OU homomorphic encryption algorithm and the H-MAC scheme, respectively. MAC is used to achieve in-network false data filtering and thus avoid wasting unnecessary energy by not transmitting false data. Analysis shows that our scheme has a good behavior in reducing energy consumption. Also, delay, especially decryption delay at the base station is shorter when compared to other state-of-art methods.
4. The efficiency and feasibility of our scheme have been evaluated based on its deployment on Tmote Sky and iMote 2. The performance of the proposed scheme has been compared with other related schemes in terms of computation overhead, communication overhead, energy consumption and system delay.

The rest of this paper is organized as follows. Section 2 reviews the related works. Section 3 introduces some algorithms mentioned and other related cryptographic tools. Section 4 comments on “confidentiality and integrity for data aggregation in WSN using homomorphic encryption”. Section 5 comments on “symmetric-key based homomorphic primitives for end-to-end secure data aggregation in wireless sensor networks”. Section 6 presents the system model. Section 7 describes the construction of our scheme in detail. Section 8 presents security analysis. Performance analysis is given in section 9 and section 10 concludes this paper.

2 Related works

As a vital method of data collection, data aggregation has received widespread attention. To achieve secure data aggregation, numerous data aggregation schemes have been widely proposed.

In conventional hop-by-hop aggregation schemes [10, 11], an aggregator has to decrypt each received message, then aggregate all messages according to a corresponding aggregation function and, finally, encrypt the aggregated result before forwarding to next hop. This means that aggregators are required to store keys for decryption and, thus, a compromised aggregator can reveal transmitting messages or forge aggregated results. To decrease this impact, homomorphic

encryption schemes have been applied to WSNs [12–14]. By homomorphic encryption schemes, end-to-end data confidentiality is provided. In end-to-end data aggregation schemes, aggregators directly aggregate encrypted data without decrypting them and, therefore, compromised aggregators cannot access secret information.

Girao et al. [15] introduced the method of aggregating encrypted data in WSNs. They proposed a CDA scheme based on symmetric homomorphic encryption to achieve the aggregation of encrypted data. However, all nodes in the work share a common key for encryption, which means that the system security will collapse if a node is compromised. The problem is solved by generating a temporal key for each communication in [16], the authors of which proposed a CDA scheme based on one-time pad. However, this scheme requires that identifiers of all participants are sent to ensure accuracy of the aggregated results, which increases the transmission overhead. The authors of [17] proposed an approach to avoid identifiers transmission; the algorithm requires that all nodes respond to the query from the base station and values of nodes having no sensed data are set to zero.

Parmar et al. [9] proposed an integrity and privacy preserving end-to-end secure data aggregation protocol, in which a symmetric homomorphic encryption algorithm is used to protect data confidentiality, MAC is utilized to achieve in-network false data filtering at cluster nodes and the H-MAC algorithm is employed to achieve data integrity at the base station. They claim that their protocol can resist some well-known cryptographic attacks, such as Known-Ciphertext attack, Known-Plaintext attack, the Sybil attack, Node Capture attack, and so on. However, we find that this protocol cannot achieve the security level they claimed. Also, this scheme cannot resist replay attacks.

Recently, Boudia et al. [4] proposed a novel secure aggregation scheme which uses Stateful Public Key Cryptography (StPKE), symmetric based homomorphic technique and MAC to provide end-to-end security. However, to achieve data integrity protection, the messages require to be formed as concatenations of all messages from member nodes, i.e. $m = m_1 || \ldots || m_n$. Thus, the size of the resulting messages grows linearly with the number of member nodes or cluster heads. Therefore, their recoverable sensing data approach is very inefficient if the message size is large.

Unlike these above schemes using symmetric homomorphic encryption, the authors in [18] study the suitability of a group of asymmetric based homomorphic encryption algorithms and make a detailed analysis of performance in terms of encryption, decryption and bandwidth. The authors show that EC-EG algorithm is the best candidate. However, if the aggregated result is not small enough, EC-EG requires significantly more computation power for the decryption than other schemes. It may be that, the plaintext m cannot be recovered from mP because of the hardness of the ECDLP. The authors also state that OU is the best scheme if EC-EG cannot be applied, e.g., in very large networks.

The authors of [5] proposed two recoverable CDA schemes, RCDA-HOMO for homogeneous WSNs and RCDA-HETE for heterogeneous WSNs, in which the base station not only can recover each sensing data, but also can check the integrity of each original data. However, to verify the validity of signatures, the base station needs pairing computation. It is well known that the pairing operation is very expensive. In addition, decryption of a ciphertext is equal to solution of the ECDLP, which brings much heavier computation overhead due to the hardness of the ECDLP. Although the base station is powerful, it is much too heavy, which leads to very low efficiency in verification. Also, the authors in [7] point out that RCDA-HETE cannot provide data integrity like their claim. Moreover, these two schemes do not provide authorized aggregation.

In [6], the authors proposed a CDAMA scheme for multi-application environments, in which ciphertexts from different applications can be aggregated into a single one and the base station can extract application-specific data from aggregated ciphertexts by a corresponding key. Unfortunately, this scheme is not suitable for the WSNs where the number of clusters or applications is large. What is more, it does not achieve data integrity protection.

Recently, Shim et al. [7] proposed a data aggregation scheme with end-to-end confidentiality and integrity using an EC-EG homomorphic scheme and a signature scheme, which also provides in-network data filtering and authorized aggregation, but incurs high energy consumption. Also, decryption involves solving the ECDLP, which will undoubtedly decrease the decryption efficiency.

The aforementioned asymmetric homomorphic encryption-based schemes are not suitable for the large-scale WSN. The use of the EC-EG homomorphic encryption algorithm makes it difficult that for the base station to decrypt the aggregated ciphertexts and obtain the sensing data. If the aggregated results are large, the base station has no way to gain the original data due to the hardness of ECDLP. Also, these schemes incur a considerable overhead in terms of energy consumption and delay due to the cryptographic algorithms employed.

Our contribution is motivated by the above facts that the existing secure data aggregation schemes based on symmetric or asymmetric homomorphic encryption are unsuitable for the large-scale WSNs, which also justifies the importance of this work.

3 Preliminaries

In this section, we briefly introduce some algorithms mentioned and other related cryptographic tools.

3.1 Okamoto-Uchiyama (OU) algorithm

The Okamoto-Uchiyama (OU) algorithm [19] is a public-key cryptosystem as secure as factoring and based on the ability of computing discrete logarithms in a particular subgroup [18]. Detailed descriptions are presented in Fig. 1.

3.2 Elliptic curve ElGamal (EC-EG) algorithm

EC-EG is additively homomorphic and ciphertexts are combined through addition. Its security is based on the ECDLP. This algorithm is to map plaintext m to the EC point mG, and reverse m from mG. However, the demapping of the mG back to m is impractical. Since it is very hard to be inverted for point multiplication of ECC, the only solution is a brute force computation that relies on a limited domain of the mapping [7, 18]. Detail descriptions are shown in Fig. 2.

3.3 Homomorphic MAC scheme

In 2009, a homomorphic MAC algorithm was proposed by Agrawal et al. to check the integrity of aggregated data. Previous MAC cannot achieve the additive property: $MAC(a+b) \neq MAC(a) + MAC(b)$, so it cannot be directly used for data aggregation. Fortunately, the homomorphic MAC scheme makes it possible to ensure the addition over authenticated data and, thus, can be used to verify the integrity of aggregated data. The correctness and security proof of this algorithm can be found in [20]. Fig. 3. gives detailed descriptions of the homomorphic MAC.

3.4 Hash-based Message Authentication Code (HMAC)

The HMAC is generally used to check data integrity and source. It is implemented by combining a secret key with a one-way, collision-resistant hash function, such as MD5, SHA-1 and so on. The security strength of HMAC is due to the underlying hash function. We use HMAC (k, m) to represent digest of m with a key k.

4 Comments on "Confidentiality and integrity for data aggregation in WSN using homomorphic encryption"

In secure data aggregation schemes, homomorphic encryption is usually used to protect data confidentiality. The authors of [18, 21] analyze security and performance for several common asymmetric homomorphic encryption schemes, including EC-NS, EC-OU, EC-P, EC-EG and OU. Recently, Othman et al. [8] proposed a data aggregation scheme with confidentiality and integrity in WSN using homomorphic encryption, in which the authors adopt OU to protect data confidentiality; however, EC-EG instead of OU is used as homomorphic encryption for this paper in Aggregate Phase and Verify Phase. The processes for Aggregate Phase and Verify Phase are the same as [5]. The review of Othman et al.'s data aggregation scheme is presented in Fig. 4.

From ①② in Fig. 4, we can see that the authors intend to employ OU homomorphic encryption algorithm to protect data privacy. However, we can see that (r, s) in *Aggregated Phase* and *rmap()* in *Verify Phase* do not appear in *Encrypt-Sign Phase* from ③④, because they belong to the EC-EG homomorphic encryption algorithm rather than OU.

So, we correct Othman et al.'s data aggregation scheme as follows:

Setup Phase. The base station generates parameters $\{n,g,h,p,q\}$ and then publishes parameters $\{n,g,h\}$ as its public key. These system parameters are preloaded on each sensor node. $\{p,q\}$are kept only by the base station and used as its private key.

Encrypt-Sign Phase.

1. Encoding: $m_i = d_i \mid \mid 0^{\beta}$,where $\beta = l \cdot (i-1)$, d_i is the sensing data by the node i.
2. Sign: $\sigma_i = h_i^{x_i}$,where $h_i = \mathscr{H}(d_i)$.
3. Encrypt: $c_i = g^{m_i + nr_i}$.

Fig. 1 The Okamoto-Uchiyama (OU) algorithm

Okamoto-Uchiyama (OU) algorithm

Setup. For an odd prime p, the p-Sylow subgroup is defined as $\gamma_p = \{x < p^2 \mid x = 1 (\mathrm{mod}\, p)\}$, and $|\gamma_p| = p$. A function L that maps elements from γ_p to $\mathbb{Z}_p$ is defined as $L(x) = (x-1)/p$. Function L has homomorphic properties from multiplication to addition, $L(a*b) = L(a) + L(b) (\mathrm{mod}\, p)$, and $L(a^c) = c * L(a)$, where $a, b \in \gamma_p, c \in \mathbb{Z}_p$.

KeyGen. Let p and q be random k-bit primes and set $n = p^2 q$. Next, randomly choose a $g \in_R \mathbb{Z}_n$ such that element $g_p = g^{p-1} (\mathrm{mod}\, p^2)$ has order p. Finally, set $h = g^n (\mathrm{mod}\, n)$. Then, public key is (n, g, h), and private key is (p, q).

Encrypt. Given $m \in 2^k$, pick up a random $r \in_R \mathbb{Z}_n$, compute a ciphertext $c = g^m h^r (\mathrm{mod}\, n)$.

Decrypt. $c' = c^{p-1} (\mathrm{mod}\, p^2)$, compute $m = L(c') L(g_p)^{-1} (\mathrm{mod}\, p)$.

Note that $c^{p-1} (\mathrm{mod}\, p^2) = g^{m(p-1)} g^{rn(p-1)} = g_p^m (\mathrm{mod}\, p^2)$.

Elliptic Curve ElGamal (EC-EG) algorithm

Setup. Given a security parameter κ, construct an elliptic curve E defined over a finite field $\mathbb{F}_p$ together with a prime p and generator G.

KeyGen. Select a random $x \in \mathbb{F}_p$ as private key, and compute $Y = xG$. Then, public key is (E, p, G, Y).

Encrypt. Encrypt m with public key (E, p, G, Y).

1. Select a random number $k \in \mathbb{F}_p$.
2. Compute $M = map(m)$, where $map(m) = mG$.
3. Generate ciphertext $C = (R, S)$, where $R = kG, S = M + kY$.

Decrypt. Decrypt C with private key x.

1. Compute $M = -xR + S = -xkG + M + xkG$.
2. Reverse m through $m = rmap(M)$.
3. Return the plaintext m.

Fig. 2 The Elliptic Curve ElGamal (EC-EG) algorithm

Aggregation Phase.

1. Aggregating Ciphertext:

$$\hat{c} = \prod_{i=1}^{\eta-1} g^{m_i + nr_i} = g^{\sum_{i=1}^{\eta-1} m_i + nr_i} = g^{\sum_{i=1}^{\eta-1} m_i}.$$

2. Aggregating Signature: $\hat{\sigma}_i = \sum_{i=1}^{\eta-1} \sigma_i$.

Verify Phase.

1. Decrypt:

$$c' = \hat{c}^{p-1} \bmod p^2 = g^{\sum_{i=1}^{\eta-1} m_i \cdot (p-1)} \bmod p^2 = g_p^{\sum_{i=1}^{\eta-1} m_i},$$

$$m' = \sum_{i=1}^{\eta-1} m_i = L(c') L\left(g_p\right)^{-1}.$$

2. Decoding: $d_i = m'[(i-1) \cdot l, i \cdot l - 1], i = 1, 2, \ldots, \eta - 1$.
3. Verify: $e(g_1, \hat{\sigma}) = \prod_{i=1}^{\eta-1} e(v_i, h_i)$.

Fig. 3 The Homomorphic MAC scheme

5 Comments on "Symmetric-key based homomorphic primitives for end-to-end secure data aggregation in wireless sensor networks"

Recently, Parmar et al. [9] proposed an integrity and privacy preserving end-to-end secure data aggregation protocol, in which a symmetric homomorphic encryption algorithm is used to protect data confidentiality and a homomorphic MAC algorithm is employed to achieve data integrity. They claim that their protocol can resist some well-known cryptographic attacks, such as Known-Plaintext attack, Node Capture attack, and so on. However, we found that this protocol cannot achieve the security level they claimed.

5.1 The weakness found

In this scheme, all the leaf nodes use the same symmetric key to generate a homomorphic MAC tag. What is more, the

Homomorphic MAC scheme

Setup. Given a Pseudo Random Generator $G: \mathcal{K}_G \to \mathbb{F}_q^{n+m}$ and a Pseudo Random Function $F: \mathcal{K}_F (\mathcal{I} \times [m]) \to \mathbb{F}_q$.

KeyGen. Choose $k_1 \in \mathcal{K}_G, k_2 \in \mathcal{K}_F$, then key $k = (k_1, k_2)$.

Sign$(k, id, \mathbf{v}, i)$. Given ith basis vector $\mathbf{v} \in \mathbb{F}_q^{n+m}$ and a key $k = (k_1, k_2)$. To generate a homomorphic MAC, do:

1. $\mathbf{u} \leftarrow G(k_1) \in \mathbb{F}_q^{n+m}$
2. $b \leftarrow F(k_2, (id, i)) \in \mathbb{F}_q$
3. $t \leftarrow (\mathbf{u} \cdot \mathbf{v}) + b \in \mathbb{F}_q$

Output t. t is a homomorphic MAC tag.

Aggregate. Given $(\mathbf{v}_1, t_1, \alpha_1), \ldots, (\mathbf{v}_m, t_m, \alpha_m)$, compute, $t \leftarrow \sum_{j=1}^{m} \alpha_j t_j \in \mathbb{F}_q$.

Verify$(k, id, \mathbf{y}, t)$. Given a secret key $k = (k_1, k_2)$ and $y = (y_1, \ldots, y_{n+m}) \in \mathbb{F}_q^{n+m}$, verify a tag t as follows.

1. $\mathbf{u} \leftarrow G(k_1) \in \mathbb{F}_q^{n+m}$ and $a \leftarrow (\mathbf{u} \cdot \mathbf{y}) \in \mathbb{F}_q$
2. $b \leftarrow \sum_{i=1}^{m} [y_{n+i} \cdot F(k_2, (id, i))] \in \mathbb{F}_q$
3. If $a + b = t$ output 1; otherwise output 0.

Fig. 4 The review of Othman et al.'s data aggregation scheme

Othman et al.'s secure data aggregation scheme

Setup Phase. For each sensor SN_i, the BS generates, (Sv_i, Sx_i) by KeyGen procedure, where $(Sv_i = v_i)$ and $Sx_i = x_i$. (BS_{pk}, BS_{zk}): These keys are generated by KeyGen. Procedure (see Okamoto–Uchiyama Cryptosystem)[Ⓜ] where $BS_{pk} = (n, g, k)$ and $BS_{zk} = (p, p_g)$. After that, they are loaded to SN_i for all i.

Encrypt-Sign Phase.

Encoding

After encoding, SN_i computes:

Signature: $\partial_i = x_i \times h_i$ where $h_i = x_i = \mathcal{H}(d_i)$

Ciphertext: $c_i = g^{m+n} \bmod n$.[Ⓜ]

At the end, SN_i sends the pair (c_i, ∂_i) to Aggregator.

Aggregation Phase.

Aggregated Ciphertext: $\hat{C} = (\hat{r}, \hat{s}) = (\sum_{i=1}^{n-1} r_i, \sum_{i=1}^{n-1} s_i)$,[Ⓜ]

Aggregated Signature: $\hat{\partial}_i = (\sum_{i=1}^{n-1} \partial_i)$.

Send the Aggregated Result $(\hat{C}, \hat{\partial})$ to the BS.

Verify Phase.

BS obtains $\hat{M}$ by decrypting $\hat{C}$.

BS obtains $\hat{m}$ from $\hat{M}$ through the reverse function *rmap()*: $\hat{m} = rmap(\hat{M}) = m_1 + m_2 + \ldots m_{n-1}$.[Ⓜ]

BS obtains each sensing data from $\hat{m}$ by *Decode function*.

BS verifies each ∂_i via checking whether the equation $e(g, \hat{\partial}) = \prod_{i=1}^{i} e(v_i, h_i)$ holds or not.

The BS can recover all sensed data within the entire WSN.

homomorphic MAC tag is generated over a plaintext *m*, which leads to security weakness. Once a node is compromised, messages of all the nodes are revealed and, thus, an adversary can forge the aggregated results to deceive the base station. As such, this protocol cannot provide security against Known-Plaintext attack or, Node Capture attack. The analysis is as follows.

(1) *Known-Plaintext Attack*. In such attack, an adversary tries to deduce the key or recover other plaintexts from their own ciphertexts by some plaintext-ciphertext pairs. In WSN, sensor nodes are easily compromised due to the hostile and unattended environment thus, a compromised node may produce such plaintext-ciphertext pairs. In [9], the authors state that the key for each node is only shared with the base station and other nodes cannot access it, so, even though a node is compromised and its key is revealed, an adversary has no way to gain other nodes' information through it and thus can resist Known-Plaintext Attack. However, as all the leaf nodes use the same key to encrypt the plaintext, if a node is compromised, the messages of all the nodes will be discovered. Therefore, this protocol cannot resist such attack.

(2) *Node Capture attack*. If a node is captured, sensor readings and its key information stored will be discovered. In this scheme, the authors state that they use a symmetric homomorphic encryption for privacy protection, and any captured sensor node can only reveal its own sensor readings, but cannot decrypted the ciphertexts encrypted with other sensor nodes' encryption keys. Although the adversary cannot directly decrypt ciphertexts to obtain other nodes' plaintexts, he can decrypt homomorphic MAC tags with the same k' to gain sensor readings of other nodes. Hence, the protocol cannot protect the network against node capture attacks.

Apart from the above weaknesses identified in the protocol, we also found that the protocol cannot resist replay attack. In addition, the intermediate nodes need to store key information of their child nodes, which make them especially attractive for adversaries. Therefore, to guarantee security, the intermediate nodes are best equipped with a tamper-resistant device. In addition, if the number of child nodes is large, the storage space for normal nodes is insufficient. What is more, the intermediate nodes not only encrypt messages and generate MAC and homomorphic MAC tag like their child nodes, but also verify each message received, which results in more energy consumed than their child nodes and causes an energy imbalance between the intermediate nodes and child nodes. Hence, the intermediate nodes can be considered setting as high-end nodes.

6 System model

In this section, we state two models: network model and adversary model. The network model defines the network architecture; the adversary model defines common attacks against which a secure data aggregation scheme should protect.

6.1 Network model

In our scheme, the network topology is a cluster-based aggregation structure and each cluster possesses a cluster head (CH) (see Fig.5). A WSN contains large numbers of sensor nodes and one base station. Since storing keys share with their member nodes and verify packets received, this makes them require more storage space and higher computational capability than member nodes, and thus, CHs are set to be powerful high-end sensors while member nodes are low-end nodes. We suppose that each node has a unique identifier (ID) and can be identified by their IDs. After deployment, all nodes are stationary and the base station (BS) is fixed. We assume that the BS is powerful and absolutely trusted. Finally, time is assumed to be loosely synchronized.

6.2 Adversary model

We categorize the adversary's abilities as follows:

1) An adversary can obtain secret information by passively eavesdropping data being transmitted.
2) An adversary can interfere with the communication by replaying old packets, modifying the transmitted data, injecting false data or unauthorized aggregation.
3) An adversary can physically compromise a sensor node or a CH.

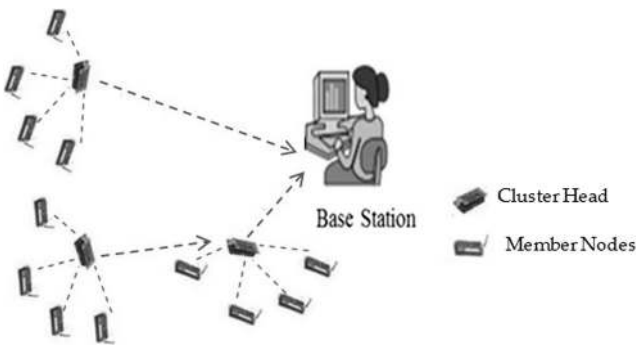


Fig. 5 The network model

Next, we further refine these three kinds of attacks into three categories based on abilities and purposes of adversaries.

In category A, an adversary is aimed at obtaining secret information.

A1: Eavesdropping attack. Eavesdropping attack concerns the passive adversary aiming to get information.

In category B, an adversary aims to send false data by modifying the contents, replaying old packets, injecting bogus data or unauthorized aggregation to deceive the base station even though he does not have the secret key.

B1: Malleability. Malleability allows an adversary to modify data without knowing the content.
B2: Replay attack. An adversary intercepts the transmitted data, and then replays it in the future to deceive the BS.
B3: Injection attack. An adversary can generate valid ciphertexts under the public key of the base station and then inject them into the network to deceive the base station and waste the transmission energy.
B4: Unauthorized aggregation. The idea of such an attack is to aggregate two or more proper ciphertexts into a forged, but authentically looking one in order to cheat the base station.

In category C, an adversary is aimed at the sensing data or secret keys, for example, keys for generating MAC and H-MAC.

C1: Compromise attack. An adversary can compromise a CH or a sensor node to try to obtain the sensing data.

7 The proposed scheme

Our secure data aggregation is composed of four phases: *Setup, Encrypt, Aggregate, Verify*. The *Setup* phase is to prepare and preload some essential secrets and system parameters for each sensor node and the BS. When a node wants to send its sensing data to its CH, it firstly carries out *Encrypt*, and then sends the result to its CH. Unlike other secure data aggregation schemes in which CH is only responsible for aggregation, in our scheme, each CH needs to verify all packets received from its member nodes, which can filter part of the bogus packets in-network and thus can save energy in transmission in *Aggregate* phase. The last phase is *Verify*. The BS individually verifies the integrity of

Table 1 Notations and their description

Notation	Description
CID_j	Cluster identifier of sensor nodes belonging to cluster j
η	Number of sensor nodes per cluster
m_{ij}	Sensing data of member node CM_{ij}
H_{ij}	The homomorphic MAC generated by CM_{ij}
n_c	Number of clusters in the whole network
CH_j	The cluster head node of the jth cluster in the network
CM_{ij}	The ith member node in the jth cluster
m_{agg}	The sum of all valid sensing data

the aggregated result for each cluster. Only the aggregated results passing integrity verification will be decrypted by the BS and have chance to participate in the final aggregation. Table 1 lists the notions used later.

[Setup]. Before the deployment of the WSN, the base station produces necessary secrets and system parameters as follows:

- The base station distributes a cluster identifier CID_j to each node in cluster j; namely, the cluster identifiers of nodes in the same cluster are identical.
- The base station chooses an identifier $ID_i \in [1, \ldots, n]$ for each node in WSN and preloads them into nodes. For simplicity, we assume that $ID_i = i$.
- The base station shares a unique symmetric-key pair $k_j = \{k_{j1}, k_{j2}\}$ with all nodes in the same cluster j, where symmetric-key pair $k_j = \{k_{j1}, k_{j2}\}$ is used to generate H-MAC to protect end-to-end data integrity.
- The cluster head CH_j shares a symmetric-key k_{i-j} with its each member nodes in cluster j and each cluster head CH_j also shares a symmetric-key k_{j-BS} with the base station, where k_{i-j} and k_{j-BS} are used to produce MAC to achieve in-network false data filtering.
- The base station generates its public key (n, g, h) and private key (p, q) according to OU algorithm, then keeps the private key and publishes its public key.

[Encrypt-Sign]. When a sensor node $CM_{ij} (i = 1, \ldots, \eta - 1)$ wants to send sensing data m_{ij} to its cluster CH_j, it needs to compute:

- *Ciphertext.* A member node CM_{ij} picks a random number $r \in_R \mathbb{Z}_n$ and computes its ciphertext $c_{ij} = g^{m_{ij}} h^{r_{ij}} \bmod n$ under the public key of the base station through OU algorithm.
- *Homomorphic MAC*. In our scheme, the homomorphic MAC H_{ij} is generated over the ciphertext instead of plaintext, which solves the problem that once a member node in one cluster is compromised, all the member nodes' data will be disclosed because the keys used to generate homomorphic MAC for each node in one cluster are identical.
- *MAC.* We use MAC to achieve in-network false data filtering, which thus avoids consuming unnecessary transmission energy to transmit false data packets. Timestamp t is employed to guarantee data freshness. Algorithm 1 presents this process.

Algorithm 1: Encrypt & Sign (CM_{ij})

Input: $m_{ij}, (n, g, h), r_{ij}, (k_{j1}, k_{j2}), k_{i-j}, t_{ij}, CID_j, i$

Output: $c_{ij}, H_{ij}, MAC_{ij}, t_{ij}$

1.Compute $c_{ij} = g^{m_{ij}} h^{r_{ij}} \bmod n$

2.Compute $H_{ij} = (\mu_{ij} \cdot c_{ij}) + b_{ij}$, where $\mu_{ij} = G(k_{j1})$, $b_{ij} = F(k_{j2}, (CID_j, i))$

3.Compute $MAC_{ij} = HMAC(k_{i-j}, c_{ij} \| H_{ij} \| t_{ij})$

[Aggregate]. When the cluster head CH_j receives (c_{ij}, H_{ij}, MAC_{ij}, t_{ij}) from its member node $CM_{ij} (i = 1, \ldots, \eta - 1)$, it will perform the following operations:

- *Check timestamps.* The CH_j checks the validity of the timestamp t_{ij}. If the timestamp t_{ij} is valid, the CH_j will verify MAC. If not, reject it.

- *Verify MAC*. The CH_j computes $HMAC(k_{i-j}, c_{ij}||H_{ij}||t_{ij})$ and compares it with MAC_{ij} received. If they are equal, then the CH_j aggregates the corresponding ciphertext and homomorphic MAC, or the packet will be rejected.
- *Aggregate*. In this step, the cluster head $CH_j (j = 1, \ldots, n_c)$ acts as a data aggregator. The CH_j aggregates $\eta - 1$ ciphertexts received from its member nodes and its own ciphertext into a single ciphertext c_j. Also, it combines $\eta - 1$ homomorphic MAC H_{ij} and its own H_{ij} into one homomorphic MAC H_j. Then, the CH_j sends the output of Algorithm 2 to the base station or the nearest CH. When a CH receives a packet from another CH, it only forwards it to the base station, and does not aggregate or verify it.

Algorithm 2: Aggregate (CH_j)

Input: $c_{ij}, H_{ij}, MAC_{ij}, t_{ij}, i$

Output: c_j, H_j, MAC_j, t_j

1.Check t_{ij}

2.Compute $HMAC(k_{i-j}, c_{ij} \| H_{ij} \| t_{ij})$ and check $HMAC(k_{i-j}, c_{ij} \| H_{ij} \| t_{ij})? = MAC_{ij}$

3.Compute $c_j = \prod_{i=1}^{\eta} g^{m_{ij}} h^{r_{ij}} = \prod_{i=1}^{\eta} g^{m_{ij}+nr_{ij}} = g^{\sum_{i=1}^{\eta}(m_{ij}+nr_{ij})} = g^{\sum_{i=1}^{\eta} m_{ij}} = g^{m_j}$

where $h = g^n \bmod n$

4.Compute $H_j = \sum_{i=1}^{\eta} H_{ij}$

5.Compute $MAC_j = HMAC(k_{j-BS}, c_j \| H_j \| t_j)$

[Verify]. When the base station receives (c_j, H_j, MAC_j, t_j), it will perform the following operations:

- *Check timestamps*. The base station checks the validity of the timestamp t_j. If the timestamp t_j is valid, the base station will verify MAC. If not, reject it.
- *Verify MAC*. The base station calculates $HMAC(k_{j-BS}, c_j||H_j||t_j)$ and compares it with MAC_j received. If they are equal, it can be sure that the ciphertext received and the corresponding homomorphic MAC are not modified by adversaries.
- *End-to-end integrity verification*. The base station verifies the integrity of each c_j through the homomorphic MAC scheme. If the verification holds, then the aggregated ciphertext c_j will be decrypted and has chance to participate in the final aggregation, otherwise it will be rejected. In our scheme, the packet of each cluster is verified individually; that is to say, after each cluster's packet is verified successfully, the base station then decrypts them and aggregates all valid plaintext m_j rather than directly verifying the final aggregated results. In this way, if the verification is failed to pass for one cluster, only the packet of this cluster is discarded. Unlike other schemes [4, 7], once the verification fails, all packets, including valid packets, will be abandoned, which means all data need to be retransmitted.
- *Decrypt*. In this step, the base station decrypts the aggregated ciphertext c_j and obtains the aggregated plaintext m_j for each cluster. For those ciphertexts failed to pass end-to-end integrity verification, it is not necessary to decrypt them, which results in saving energy consumption, because, for a modified packet, verification then decryption only involves verification overhead while decryption then verification consumes energy in both verification and decryption.
- *Get the final aggregated result* m_{agg}. Only by passing the end-to-end integrity verification, can the aggregated plaintext m_j participate in the final aggregation; namely, m_{agg} is the result of the sum for all m_j whose ciphertext passes the end-to-end integrity verification. Its advantage being that, if some received messages are not successfully verified, other valid packets can be utilized. Algorithm 3 describes the detail verification process.

Algorithm 3:End-to-end verification (Base station)

Input: All (c_j, H_j, MAC_j, t_j) , where $j \in \{1,...,n_c\}$

Output: m_{agg}

1. For each (c_j, H_j, MAC_j, t_j)

   1.1 Check t_j

   1.2 Compute $HMAC(k_{j-BS}, c_j \| H_j \| t_j)$ and check $HMAC(k_{j-BS}, c_j \| H_j \| t_j)? = MAC_j$

   1.3 Output 1 if $a + b = H_j$, otherwise output 0.

   Here, $a = c_j \cdot G(k_{j1}), b = \sum_{i=1}^{\eta} F(k_{j2}, (CID_j, i))$

   1.4 Obtain the aggregated plaintext $m_j = \sum_{i=1}^{\eta} m_{ij} = L(c')L(g_p)^{-1}$,where

$$c' = {c_j}^{p-1} \bmod p^2 = g^{\sum_{i=1}^{\eta} mij \cdot (p-1)} \bmod p^2 = {g_p}^{\sum_{i=1}^{\eta} mij} \bmod p^2 = {g_p}^{m_j} \bmod p^2$$

2. Obtain the final aggregated result m_{agg}

8 Security analysis

In this section, we analyze the security of our scheme in terms of data confidentiality and integrity.

Theorem 1 Our scheme provides end-to-end data confidentiality in the presence of the adversary of category A.

Proof Since the sensing data of each sensor node are encrypted with the public key of the base station, only the corresponding private key can decrypt the encrypted messages. However, the private key is only stored in the base station, so, even though an adversary of category A eavesdrops on the transmitted packet, he cannot decrypt the ciphertexts. In the following, we analyze how our scheme is secure against attacks launched by an adversary of category A.

Eavesdrop attack: In our scheme, the sensing data are encrypted under the public key of the base station during the transmission process. After receiving packets from its member nodes, a CH does not decrypt messages but only aggregates them. Only the base station can decrypt messages to obtain the sensing data. Even though an adversary eavesdrops on a transmitted packet, he has no way to decrypt the ciphertext without the private key of the base station. End-to-end confidentiality of our scheme can be reduced to the security of the underlying homomorphic encryption scheme, OU. Detailed security proof can be found in [19].

Theorem 2 Our scheme provides end-to-end data integrity in the presence of an adversary of category B.

Proof We utilize a homomorphic MAC scheme to solve the adversary of category B problem; if malicious behavior against data integrity occurs, the end-to-end integrity verification will not be successful. The security proof for the homomorphic MAC can be found in refer to [20]. In the following, we analyze how our scheme is secure against attacks launched by an adversary of category B.

Malleability: Malleability is a common threat for all homomorphic encryption schemes. An adversary can alter a ciphertext by injecting false data, but it will not be detected due to the homomorphic property. For example, $(m + k) \bmod n$ may be modified to $(m + x) + k \bmod n$, where x is the false data injected by an adversary. In our scheme, we use a homomorphic MAC scheme to verify the integrity of the data. If the encrypted data is tampered, the integrity verification will fail and thus the BS will refuse the received packet.

Replay attack: An adversary can impersonate any node through replaying old packets recorded from past communications; therefore, we add current timestamps to messages being signed to resist replay attacks. Thus, the receivers can ensure data freshness by checking the validity of the timestamps.

Injection attack: With public key cryptography, any adversary can generate a reasonable ciphertext and inject it into the network to deceive the base station. In our scheme, each sender (a sensor node or a CH) computes

a MAC using the symmetric key shared with the receiver (a CH or the base station), so the receiver will reject these injected packets in the "*Verify MAC*" step if an adversary injects its false data.

Unauthorized aggregation: Unauthorized aggregation is a very specific weakness of homomorphic encryption schemes [18]. The idea of such an attack is to aggregate two or more proper ciphertexts into a forged but format-valid ciphertext to deceive the BS. If the CHs only aggregate data, anyone can impersonate this to produce a false aggregated result by dropping some packets and, thus, misleading the BS. To protect a homomorphic encryption scheme from unauthorized aggregation, in our scheme, each CH not only performs aggregation operation, but also generates MAC and homomorphic MAC on the aggregated result. Therefore, the BS can check the authenticity of the CHs and the integrity of the aggregated results.

Theorem 3 Our scheme can provide security against an adversary of category C.

Proof We employ a homomorphic MAC scheme to provide end-to-end confidentiality. The CHs are only responsible for aggregation rather than decryption. Therefore, even if a CH is compromised, an adversary cannot obtain the sensing data. In addition, we compute a homomorphic MAC over ciphertext instead of plaintext and, for different clusters, the keys used to generate a homomorphic MAC are different; therefore, even if a sensor node is compromised, only its data will be disclosed, the data for other nodes will be still secure.

Compromise attack: We classify compromise attack into two cases: (1) Compromise a CH. Since the CHs store important data, it makes them likely to be targeted by adversaries. However, even if a CH is compromised, an adversary cannot obtain the sensing data, because the CHs do not store the private key of the base station, and decrypting a ciphertext is impractical. (2) Compromise a sensor node. If a sensor node is compromised, it only reveals its own sensing data and an adversary has no way to get key and data for other nodes. In [9], once an adversary gains the key used to generate the homomorphic MAC by compromising a node, all nodes' data will be disclosed, since they calculate the homomorphic MAC over the plaintext and the keys used for the homomorphic MAC are identical for all sensor nodes. However, we compute a homomorphic MAC over the ciphertext and the keys used to produce a homomorphic MAC for different clusters are different. In this way, even if a node is compromised, the result cannot have a significant impact upon the overall system security.

9 Performance analysis

In this section, we evaluate the performance of our scheme in terms of cost evaluation and execution time (or "delay"). Cost evaluation involves computation overhead, communication overhead and energy consumption. Delay includes processing delay, aggregation delay and decryption delay. We also provide a quantitative analysis of the proposed scheme compared to RCDA [5], CDAMA [6] and, Sen-SDA [7]. The reason why we choose these three works for comparison is because they provide a comparable security level (end-to-end confidentiality and integrity) to ours. Symmetric key-based data aggregation schemes are not considered, since symmetric schemes are generally more efficient, but less secure than asymmetric ones [6].

9.1 Security requirements

A 112-bits security level should be adopted to guarantee adequate security according to [22], but security requirements in WSNs are generally relaxed to satisfy efficiency constraints. For example, [23] has adopted a 64-bit security level. We employ a more conservative solution and use an 80-bit security level (RSA-1024 and ECC-160 equivalent). In addition, for the implementation of the other three works, the elliptic curve we employ is an MNT curve over with embedding degree of 6, as recommended in [24] for 80-bit security level.

9.2 Cost evaluation

9.2.1 Computation overhead

The implementation is done on Tmote Sky and iMote2 motes [24]. The iMmote2 mote is equipped with a 32-bit ARM XScale PXA27x microcontroller and the Tmote Sky mote uses a 16-bit Texas Instruments MSP430 microcontroller. On Tmote Sky, the current draw is $21.8mA$ in receiving mode and $19.5mA$ in transmitting mode according to [25]. In iMote2, according to [26], the current draw is $66mA$ in receiving/transmitting mode at 104 MHz. The iMote2 platform takes about 139 ms to execute a scalar multiplication operation when working at 104 MHz while, according to [24], Tmote Sky takes 4.1 s. We will use their experimental results to estimate the energy consumption. In our scheme, we choose iMote2 as CHs and Tmote Sky as member nodes.

To analyze the computation overhead, we denote symbols *SM* and, *E* as the cost of one scalar multiplication and, a modular exponentiation, respectively. Other cryptographic operations, such as hash operations and modular addition are not considered, since the cost of these operations is negligible compared to *SM* and *E*. In *Encrypt* phase, a member node has to compute its ciphertext, *H-MAC* and *MAC*, which requires $1E$ $(g^m h^r = g^{m+nr},$ where $h = g^n \bmod n,)$operation. In

aggregation phase, a CH verifies the packets received and then aggregates ciphertexts and *H-MAC*s received from its member nodes. The process does not involve energy-consuming operations, just hash, modular additive and concatenation operations.

In an RCDA-HOMO scheme, a member node needs to compute four *SMs*, of which one *SM* is required for signature generation and three *SM*s for ciphertext generation. A cluster head only performs point addition operation.

In a CDAMA scheme, the number of *SM*s that a member node needs to compute linearly increases with the number of clusters. If there are two clusters, then four *SM*s are required. Similarly, if the number of cluster is n, $2n$ *SM*s are needed to be calculated.

In a Sen-SDA scheme, a member node in one cluster needs to calculate a ciphertext and generate a signature corresponding to the ciphertext in the Encrypt-Sign phase, which requires four *SM*s (hash, module additive and other low-overhead operations are neglected). A cluster head needs $2N + 1$ *SM*s to verify signatures from its N member nodes and one *SM* to generate the signature of the aggregate result. The comparison of computation overhead is shown in Table 2.

From Table 2, we can find that our scheme is the best in terms of computation overhead. In the aggregation phase, the CH needs to compute $2N + 2$ SMs in order to achieve in-network false data filtering for a Sen-SDA scheme. However, only several simple operations are involved in the proposed scheme, which also provides in-network false data filtering by MAC verification.

9.2.2 Communication overhead

Firstly, to achieve a fair comparison, we unify the value of each common parameter. A point on an elliptic curve can be denoted by coordinates (x, y) in a finite field F_p, with $|p| = 163$ bits. One can gain y by computing a square root if x and one bit of y are both given. Therefore, the communication cost of sending a point is 164 bits. In addition, both sensor nodes' identities and timestamps are set to be 32 bits.

Table 2 Comparison of computation overhead on member node and CH

	Encrypt (CM_{ij})	Aggregate(CH_j)
RCDA-HOMO	4 SM	--
CDAMA ($k = 2$)	4 SM	--
CDAMA ($k = 3$)	6 SM	--
CDAMA ($k = 4$)	8 SM	--
Sen-SDA	4 SM	$2N + 2$ SM
Our scheme	1 E	--

k: the number of clusters in the whole network

N: the number of member nodes in one cluster

In our scheme, the ciphertext, *MAC, H-MAC* and timestamp need to be transmitted in the WSN. According to [18], a ciphertext generated by the OU algorithm is 1024 bits. Here, we consider a 4-byte *MAC* and *H-MAC* for calculation in accordance with [27], the authors of which have validated the security of 4-byte *MAC* for WSNs scenarios. The timestamp is also 4 bytes. Therefore, the size of one transmitted packet in our scheme is 1120 bits.

In an RCDA-HOMO scheme, the transmitted message contains ciphertext c_i and the corresponding signature σ_i. The size of the message is 482 bits, which contains two curve points (164*2 = 328 bits) and one BON [28] signature (154 bits).

In a CDAMA scheme, the size of ciphertexts is $(k+1) * 256 + 1$ bits according to [6], where k is the number of clusters. If $k = 3$, then the total length of a transmitted message is 1024 bits.

In a Sen-SDA scheme, a transmitted message includes sender's *ID*, receiver's *ID*, ciphertext C, signature σ and timestamp *tt*. The length of the ciphertext $C = < C_1 , C_2 >$ containing two points of the elliptic curve is 328 bits. The signature $\sigma_i = < R_i , T_i , z_i >$ consists of two points of the elliptic curve and one number in $\mathbb{Z}_q$, so its length is 488 bits. Two *ID*s are 64 bits and timestamp is 32 bits. Therefore, the total length of one message sent by the sensor node is 912 bits. Table 3 shows the comparison of communication overhead.

From Table 3, we can see that our scheme is not the best in communication cost. This is because the security of the proposed scheme is based on the hardness of the integer factorization problem and the curve has to be chosen from a large field, resulting in higher encryption overhead [6]. Other schemes benefit from their smaller modulus operations in both ciphertext size and computation efforts, since their security is based on the hardness of ECDLP.

However, although our scheme is relatively inefficient in terms of communication cost, the proposed scheme achieves in-network false data filtering and is very helpful to save energy. As all these above data aggregation schemes use homomorphic encryption schemes based on asymmetric cryptography to protect data confidentiality, anyone can generate valid ciphertexts. If there appears large numbers of these false packets, much energy will be wasted to transmit them.

Table 3 Comparison of communication overhead

	Communication (bits)
RCDA-HOMO	482
CDAMA (k = 2)	769
CDAMA (k = 3)	1025
CDAMA (k = 4)	1281
Sen-SDA	912
Our scheme	1120

Fortunately, our scheme can filter these bogus packets en-route and, thus, avoid consuming unnecessary energy due to transmitting them. Also, in our scheme, the packet of each cluster is verified individually. In this way, if the verification fails to pass for one cluster, only the packet of this cluster is discarded. Unlike other schemes, once the verification fails, all packets, including valid packets, will be abandoned, which means all data need to be retransmitted. To some extent, our scheme can greatly save communication overhead in the case event that one or more false packets reach to the base station. In a practical application, this case is very likely to emerge.

9.2.3 Energy consumption

Energy consumption (EC) is the core issue in WSNs. Communication and computation are two main factors that affect energy consumption.

– *ECs for computation*. We can estimate ECs of each phase utilizing the formula $W = U \times I \times t$, where U is the voltage, I is the current draw, and t is the execution time for one phase. According to [19], we find that a modular exponentiation operation takes $7k/4$ modular multiplications in the extended binary method and the encryption process of OU requires about 230 modular multiplications. Additionally, the time required to compute binary field multiplication at the 80-bit security level on Tmote Sky platform is 8706 cycles according to [29]. Since a pairing computation takes 10.4×10^6 cycles and the time consumed is 1.27 s, we can estimate the cost of a multiplication in the binary field as

$$\frac{1.27s}{10.4 \times 10^6 cycles} \times 8706 cycles = 1.06ms.$$

A multiplication in the extended field is about six times that in the binary filed. Therefore, computing a modular exponentiation operation in the extended field takes $1.06ms \times 6 \times 230 = 1.46s$. The *Encrypt* phase in our scheme requires a modular exponentiation operation (Note that we neglect the cost of other operations such as hash, modular additive and so on, because they are much smaller compared to modular exponentiation) and thus the resulting ECs is $W_c = 3V \times 1.8mA \times 1.46s = 7.88mJ$, where 3 V is the voltage and 1.8 mA is the current draw for the Tmote Sky mote.

– *ECs for communication*. The ECs for receiving and transmitting an l-bits message are $W_r = U * I_r * l/d_r$, and $W_t = U * I_t * l/d_r$, respectively, where I_r and I_t are the current draw in receiving and transmitting mode, respectively, and d_r ($d_r = 250kbps$) is a data rate. Therefore, ECs for the reception and transmission of one message on Tmote Sky are $W_r = 3 * 21.8 * 1120/250\,,000 = 0.29$ mJ and $W_t = 3 * 19.5 * 1120/250\,,000 = 0.26$ mJ, respectively. In our scheme, a member node transmits the data to its cluster head once only, so the total EC of one member node for communication is $W_t = 0.26$ mJ. The ECs for a member node are provided in Table 4. For CHs, apart from a Sen-SDA scheme, the CHs in other schemes only execute several simple operations and consume much smaller energy, which can be ignored, so we will not make a detailed description of ECs for CHs.

From the comparison results in Table 4, we can find that our scheme provides a great reduction of energy consumption compared with related woks and network lifetime can be hugely improved, which can be explained by the fact that much less computation cost is incurred in our scheme due to the use of OU homomorphic encryption algorithm.

9.3 Delay

We define that processing delay denotes the execution time to produce the ciphertexts, the corresponding *H-MAC* and *MAC* for member nodes. Aggregation delay is measured by the time spent on verifying *MAC* from member nodes, aggregating ciphertexts and *H-MAC*, and generating the *MAC* of the aggregated result. Decryption delay indicates the time spent on eventually gaining original data for the BS by verifying aggregated *H-MAC* and decrypting aggregated ciphertexts.

In Table 5, *SM, PA, AES, E,ECDLP* and *P* represent a scalar multiplication, point addition, AES encryption algorithm, modular exponentiation, the elliptic curve discrete logarithm problem and bilinear pairings, respectively.

From the results in Table 5, we can calculate that the processing delay of RCDA-HOMO, CDAMA ($k = 4$), Sen-SDA, and our scheme are 16.4 *s*, 32.8 *s*, 16.4 *s* and 1.46 *s*, respectively. Here, we neglect the cost of *PA* and *H*, since they are much smaller compared to the cost of *SM, ECDLP* and *P*. For

Table 4 Energy consumption of a member node on Tmote Sky

	EC for comp. (mJ)	EC for comm.(mJ)	Total EC (mJ)
RCDA-HOMO	88.56	0.11	88.67
CDAMA($k = 2$)	88.56	0.18	88.74
CDAMA($k = 3$)	132.84	0.24	133.08
CDAMA($k = 4$)	177.12	0.30	177.42
Sen-SDA	88.56	0.21	88.77
Our Scheme	7.88	0.26	8.14

Table 5 Comparison of delay in different phases

	Processing delay	Aggregation delay	Decryption delay
RCDA-HOMO	$4SM + 1PA + 1H$	$(2N-2)PA$	$1ECDLP + k(N+1)P$
CDAMA	$2kSM + kPA$	$(k-1)PA$	$kSM + kECDLP$
Sen-SDA	$4SM + 1PA + 1H$	$(2N+2)SM + (2N-2)PA + 1H$	$(2k+1)SM + 1ECDLP$
Our Scheme	$1E + 3H$	$(N+1)H$	$(2kN+k)H$

the aggregation delay, apart from a Sen-SDA scheme, RCDA-HOMO and CDAMA schemes' aggregation delay can be largely ignored because they do not provide in-network false data filtering and just execute several *PA* operations. Although our scheme achieves in-network false data filtering, the aggregation delay is also negligible, because only a few hash operations are required to compute.

For the decryption delay, our scheme is much smaller than all the above data aggregation schemes. To gain a more intuitive understanding, we take $N = 10$ and $k = 20$ as an example. According to [30], the execution time of *SM, P* and, *H* operations is 0.442 *ms*, 4.211 *ms* and 0.0001 *ms*, respectively. Table 6 lists the execution time of the above cryptographic operations running on an Intel I7–4770 processor with 3.40 GHz clock frequency, 4 gigabytes memory and running Windows 7 operating system. Cryptographic library MIRACL is used to measure time consumption of these three cryptographic operations. In fact, it is impractical to solve ECDLP within current computational capabilities unless the final aggregation *m* is small enough. To give an intuitive comparison, if we take m as 3 bytes, it would take about 170 *ms* to decrypt the message [7].

Quantitative comparison of decryption delay is presented in Table 7. Note that decryption delay indicates the time spent on decrypting messages of all clusters instead of one cluster.

From Table 7, we can find that RCDA-HOTO, CDAMA and, Sen-SDA schemes are much larger than our scheme in terms of decryption delay. This can be explained by the fact that these three schemes suffer from expensive mapping function during decryption, which involves the elliptic curve discrete logarithm problem and is too costly to revert. Although the BS owns considerably powerful computational capabilities, if computation burden is too heavy, the BS is busy in decrypting ciphertexts, which makes the whole network easily paralyzed. Fortunately, our scheme can quickly decrypt ciphertexts and get the sensing data, even if the size of the aggregated result is large. Therefore, our scheme is more suitable for larger WSNs.

10 Conclusion

In this paper, we correct some errors and amend the security flaws found in other data aggregation schemes, and successfully design an approach to achieve data integrity protection for the CDAMA scheme. We also propose a secure data aggregation scheme suitable for large-scale WSNs, while reducing reduce the energy consumption. We employ the OU homomorphic encryption algorithm to protect end-to-end data confidentiality, use MAC to achieve in-network false data filtering, and utilize the homomorphic MAC algorithm to achieve end-to-end data integrity. Unlike other schemes, in this proposed scheme, the base station can still quickly decrypt and obtain the original data even though the aggregation results are large, while other solutions may not be able to decrypt ciphertexts or the base station is being busy decoding and thus system may become paralyzed. In addition, each cluster's data packet reaching the base station is individually authenticated so that if data authentication of one cluster fails, only the data of the cluster will be discarded. Unlike other schemes, once the authentication fails, all data including all of the valid data will be abandoned, namely, all data need retransmission, which exceedingly wastes the energy of nodes. Besides, this scheme can greatly weaken the compromise attack: that a node is compromised will not threaten secret messages of other nodes in the same cluster. We choose two popular hardware platforms, Tmote Sky and iMote 2, to investigate the efficiency and feasibility of our scheme. The results demonstrate that our scheme has an excellent performance in

Table 6 Execution time of different cryptographic operations

	T_{SM}	T_P	T_H
Execution Time (ms)	0.442	4.211	0.0001

Table 7 Comparison of decryption delay

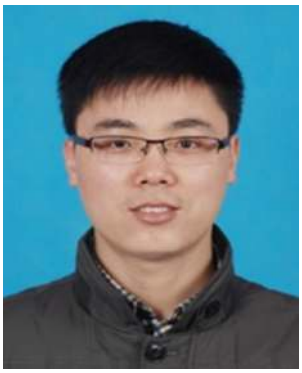
	RCDA-HOTO	CDAMA	Sen-SDA	Our scheme
Decryption Delay (ms)	1096.42	3408.84	188.122	0.042

reducing energy consumption. In addition, delay, especially decryption delay at the base station is very short unlike its counterparts, in the event where the base station is unable to decrypt ciphertexts and obtain the sensing data may arise. In the future, we aim to consider new attacks such as selective forwarding.

Acknowledgements The work was supported by the National Natural Science Foundation of China (No. 61572001, No. 61502008), the Natural Science Foundation of Anhui Province (No. 1508085QF132), and the Doctoral Research Start-up Funds Project of Anhui University. The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

References

1. He D, Zeadally S, Kumar N et al (2016) Anonymous authentication for wireless body area networks with provable security. IEEE Syst J. doi:10.1109/JSYST.2016.2544805
2. He D, Zeadally S, Wu L (2015) Certificateless public auditing scheme for cloud-assisted wireless body area networks. IEEE Syst J. doi:10.1109/JSYST.2015.2428620
3. Sardouk A, Rahim-Amoud R, Merghem-Boulahia L et al (2009) Data aggregation scheme for a multi-application WSN[C]//IFIP/IEEE International conference on Management of Multimedia Networks and Services. Springer, Berlin, pp 183–188
4. Boudia ORM, Senouci SM, Feham M (2015) A novel secure aggregation scheme for wireless sensor networks using stateful public key cryptography[J]. Ad Hoc Networks 32:98–113
5. Chen CM, Lin YH, Lin YC et al (2012) RCDA: recoverable concealed data aggregation for data integrity in wireless sensor networks[J]. IEEE Trans Parallel Distrib Syst 23(4):727–734
6. Lin YH, Chang SY, Sun HM (2013) CDAMA: concealed data aggregation scheme for multiple applications in wireless sensor networks[J]. IEEE Trans Knowl Data Eng 25(7): 1471–1483
7. Shim KA, Park CM (2015) A secure data aggregation scheme based on appropriate cryptographic primitives in heterogeneous wireless sensor networks[J]. IEEE Trans Parallel Distrib Syst 26(8):2128–2139
8. Othman SB, Bahattab AA, Trad A et al (2015) Confidentiality and integrity for data aggregation in WSN using homomorphic encryption[J]. Wirel Pers Commun 80(2):867–889
9. Parmar K, Jinwala DC (2015) Symmetric-key based homomorphic primitives for end-to-end secure data aggregation in wireless sensor networks[J]. J Inf Secur 6(1):38
10. Yang Y, Wang X, Zhu S et al (2008) SDAP: a secure hop-by-hop data aggregation protocol for sensor networks[J]. ACM Trans Inf Syst Secur (TISSEC) 11(4):18
11. Wu K, Dreef D, Sun B et al (2007) Secure data aggregation without persistent cryptographic operations in wireless sensor networks[J]. Ad hoc Netw 5(1):100–111
12. Ozdemir S, Xiao Y (2011) Integrity protecting hierarchical concealed data aggregation for wireless sensor networks[J]. Comput Netw 55(8):1735–1746
13. Zhou Q, Yang G, He L (2014) A secure-enhanced data aggregation based on ECC in wireless sensor networks[J]. Sensors 14(4):6701–6721
14. Zhu L, Yang Z, Xue J, et al (2014) An efficient confidentiality and integrity preserving aggregation protocol in wireless sensor networks[J]. Int J Distrib Sens Netw 10(2): 565480
15. Girao J, Westhoff D, Schneider M (2005) CDA: Concealed data aggregation for reverse multicast traffic in wireless sensor networks[C]//Communications, 2005. ICC 2005. 2005 I.E. international conference on. IEEE 5:3044–3049
16. Castelluccia C, Mykletun E, Tsudik G (2005) Efficient aggregation of encrypted data in wireless sensor networks[C]// mobile and ubiquitous systems: networking and services, 2005. MobiQuitous 2005. The second annual international conference on. IEEE 109-117
17. Poornima AS, Amberker BB (2010) SEEDA: secure end-to-end data aggregation in wireless sensor networks[C]//Wireless And Optical Communications Networks (WOCN), 2010 seventh international conference on. IEEE 1-5
18. Peter S, Westhoff D, Castelluccia C (2010) A survey on the encryption of convergecast traffic with in-network processing[J]. IEEE Trans Dependable Secure Comput 7(1):20–34
19. Okamoto T, Uchiyama S (1998) A new public-key cryptosystem as secure as factoring[C]//International conference on the theory and applications of cryptographic techniques. Springer, Berlin, pp 308–318
20. Agrawal S, Boneh D (2009) Homomorphic MACs: MAC-based integrity for network coding[C]//International conference on applied cryptography and network security. Springer, Berlin, pp 292–305
21. Mykletun E, Girao J, Westhoff D (2006) Public key based cryptoschemes for data concealment in wireless sensor networks[C]//2006 I.E. international conference on communications. IEEE 5:2288–2295
22. Kaliski B (2003) TWIRL and RSA key size [EB/OL]. http://www.rsa.com/rsalabs/node.asp. Accessed 6 May 2003
23. Perrig A, Szewczyk R, Tygar JD et al (2002) SPINS: Security protocols for sensor networks[J]. Wirel Netw 8(5):521–534
24. Yu S, Ren K, Lou W (2011) FDAC: Toward fine-grained distributed data access control in wireless sensor networks[J]. IEEE Trans Parallel Distrib Syst 22(4):673–686
25. Sky T (2006) Ultra low power IEEE 802.15. 4 compliant wireless sensor module [EB/OL]. http://www.cs.tau.ac.il/courses/0368-4166/docs/TelosB/tmote-sky-datasheet.pdf. Accessed 2 June 2006
26. Crossbow Technology Inc (2007) Imote2 hardware reference manual. Revision A ed, San Jose, Crossbow Technology Inc
27. Karlof C, Sastry N, Wagner D (2004) TinySec: a link layer security architecture for wireless sensor networks[C]//proceedings of the 2nd international conference on embedded networked sensor systems. ACM 162-175
28. Boneh D, Gentry C, Lynn B, Shacham H (2003) Aggregate and verifiably encrypted signatures from bilinear maps. Proc. 22nd Int'l conf. Theory and applications of cryptographic techniques (Eurocrypt), pp 416-432
29. Oliveira LB, Aranha DF, Gouvêa CPL et al (2011) TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks[J]. Comput Commun 34(3):485–493
30. He D, Zeadally S, Xu B et al (2015) An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks[J]. IEEE Trans Inf Forensics Secur 10(12): 2681–2691

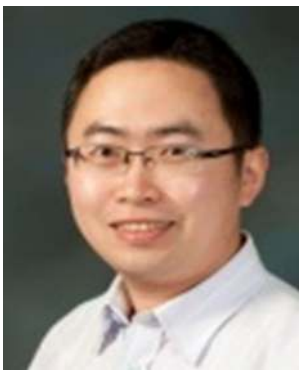


Jie Cui is now an Associate Professor in the School of Computer Science and Technology, Anhui University. He received PhD degree in University of Science and Technology of China in 2012. He has published over 20 papers. His research interests include network and information security.

Lili Shao is now a research student in the School of Computer Science and Technology, Anhui University. Her research interests include Wireless Sensor Networks, network and information security.

Hong Zhong is a Professor (from 2009) and the Dean of the School of Computer Science and Technology, Anhui University, China. She received PhD degree in University of Science and Technology of China in 2005. Her research interests cover network and information security.

Yan Xu is a Lecture in the School of Computer Science and Technology, Anhui University, China. She received PhD degree in University of Science and Technology of China in 2015. Her research interests cover network and information security.



Lu Liu is the Professor of Distributed Computing in the University of Derby, United Kingdom. Prof Liu received his PhD degree from University of Surrey, UK (funded by DIFDTC) and MSc in Data Communication Systems from Brunel University, UK. Prof Liu's research interests are in areas of cloud computing, service computing, computer networks and peer-to-peer networking. He is a Fellow of British Computer Society (BCS).