Journal of Information Hiding and Multimedia Signal Processing ©2018 ISSN 2073-4212
Ubiquitous International Volume **9**, Number **5**, September **2018**

A New Certificate-Based Aggregate Signature Scheme for Wireless Sensor Networks

Jian-Neng Chen[1,2], Fu-Min Zou[2,3,4], Tsu-Yang Wu[5,3,4,*] and Yu-ping Zhou[1,6]

[1]College of computer, Minnan Normal University
Zhangzhou, Fujian, China, 363000
jn_chen@mnnu.edu.cn

[2]Beidou Navigation and Smart Traffic Innovation Center of Fujian Province
Fuzhou, Fujian, China, 350008
fmzou@fjut.edu.cn

[3]Fujian Provincial Key Laboratory of Big Data Mining and Applications,
Fujian University of Technology

[4]National Demonstration Center for Experimental Electronic Information and Electrical Technology Education, Fujian University of Technology
Fuzhou, Fujian, China, 350008
fmzou@fjut.edu.cn

[5]College of Computer Science and Engineering,
Shandong University of Science and Technology, Qingdao, Shandong, China, 266590

[6]Key Laboratory of Applied Mathematics(Putian University), Fujian Province University
Putian,Fujian,China, 351100
yp_zhou@mnnu.edu.cn

*Corresponding author: wutsuyang@gmail.com

Received October, 2017; revised May, 2018

Abstract. *Data transmission is one of the important basic technologies in wireless sensor networks (WSN). Due to the limited energy, storage capacity, computing power and communication ability of sensor nodes, the efficiency of data transmission is greatly limited. Certificate-based aggregate signature scheme provides an efficient way to combine numerous signatures into one short signature. In this paper, we propose a certificate-based aggregate signature scheme in WSN. In the random oracle model and under the computational Diffie-Hellman (CDH) Problem and bilinear Diffie-Hellman (BDH), we demonstrate that our scheme is provably secure against forgery attack. The performance analysis demonstrates that our scheme provides an efficient way for data transmission and is suitable in WSN.*

Keywords: Wireless sensor networks; Certificate-based signature; Aggregate signature.

1. **Introduction.** A sensor network is composed of a large number of sensor nodes, which are densely deployed either inside the phenomenon or very close to it [1, 2, 3]. Wireless sensor network (WSN) is a kind of distributed sensor network. Many types of sensors in WSNs are capable detecting data, including seismic, electromagnetic, temperature, humidity, noise, light intensity, pressure, soil composition, size of moving objects, speed and direction. In other words, WSNs use a large number of cheap, small and highly restricted sensor nodes to sense the physical world [4]. So that WSN has a very wide range of applications, including environmental monitoring, event detection, target tracking and

surveillance, biomedical health monitoring, and critical facility tracking. It can also be used in certain hazardous environments, such as nuclear power plants.

As work in remote and open environment, sensor nodes are prone to attack, and security problems such as data confidentiality are very serious [5]. On the other hand, sensor node resources are usually very limited. For example, limited energy determines the short distance of data transmission. Low capacity for storage and processing determines poor computing power. Limited battery storage determines that communication between sensor nodes can not be too frequent. There are a large number of sensor nodes in a specific wireless sensor network. Each sensor node periodically sends data to the coordinator. The coordinator then sends data to the data center. Suppose a sensor network has one thousand nodes, the coordinator must send one thousand data to the data center in a time period. The efficiency and security of data transmission in WSN are getting more and more attention. Therefore, it is very important to design a safe and effective data aggregation method for WSNs.

Aggregate signature scheme is a cryptographic primitive. It provides an efficient way to transmit and verify signatures. The concept of aggregate signature was first proposed by Boneh et al [6]. in Eurocrypto 2003. In the aggregate signature scheme, n different messages are signed by n different signers, and then the n signature results are integrated into one aggregate signature. The aggregator only needs to transmit the aggregate signature instead of all the single signatures. Verifier just verifies the final aggregate signature. Later, Lysyanskaya et al. [7]construct a sequential aggregate signature scheme. The generation process of the aggregate signature requires the signer to sign one by one. After that, many aggregate signature schemes have been presented. [8, 9, 10, 11, 12, 13, 14] Until 2009, Liu et al. [15]first gave the concept of certificate-based aggregate signature and constructed the first certificate-based sequential aggregation signature scheme, but the signature scheme was inefficient. After that, a number of secure and effective aggregation signature schemes have been proposed [16, 17, 18]. However, most of these schemes require a relatively large number of calculations in the signature and aggregation process. Xiong et al. [19]introduced an efficient certificate-less aggregate signature scheme in 2013. The verification process requires only a small constant pairing count. Unfortunately, both [20]and [21]suggests that the certificate-less aggregate signature is unsafe. Some recent studies try to construct aggregate signatures with special properties [22, 23].

It is obvious that the aggregate signature scheme is very attractive for data transmission in WSNs because it saves a lot of bandwidth, storage space, and computation time. In 2015, Kim et al. [24]proposed a mediated aggregate signature by extending Mediated RSA to achieve sensor authentication and data integrity in WSNs. Its bandwidth overhead does not increase with the number of nodes, and is reduced to a constant. In the same year, Horng et al. [25]proposed a certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks. The signatures were aggregated by the roadside units, so the efficiency of the aggregator was not considered carefully in the scheme. In 2016, Shen et al. [26]introduced an identity (ID)-based aggregate signature scheme in

WSNs. Recently, Shen et al. [27]proposed an aggregate signature scheme in healthcare WSNs.

In this paper, we propose a certificate based aggregation signature (CBAS) scheme in WSNs. Our CBAS provides the advantages of both aggregate signatures and certificate-based cryptography and is suitable for WSNs. The contributions of our schemes are summarized as follows:

First, we define a framework of CBAS which is consisted of seven algorithms: Setup, KeyGen, CertGen, Sign, Verify, Aggregate, Aggverify. Then, a concrete scheme is proposed. In our CBAS scheme, users denote a large number of sensor nodes, the coordinator node denotes an aggregator, and data center denotes the designated verifier, respectively.

Second, we define the security model and adversarial model in order to demonstrate the security of our proposed CBAS scheme. In the random oracle model [28, 29, 30], we provide formal security proofs to show our CBAS scheme is secure against forgery attacks for single and aggregate signatures under the computational Diffie-Hellman and bilinear Diffie-Hellman assumptions [31, 32, 33, 34, 35].

Finally, we make the performance comparisons and demonstrate that our CBAS scheme is efficient in communication and storage overhead as well as is suitable for WSNs.

The rest of the paper is organized as follows. Section 2 defines the system model, Framework and security model of certificate-based aggregate signature schemes for WSN. In Section 3, we describe the proposed CBAS scheme. In Section 4, we present a detailed security proof of our scheme based on the computational Diffie-Hellman assumption and bilinear Diffie-Hellman. In Section 5, we analyze the performance of our CBAS scheme in terms of communication and computation cost, and the conclusions are draw in Section 6.

2. System Model.

Without of loss generality, we assume there exists one data center (DC) and n sensor nodes formed a wireless sensor network (WSN).

2.1. System model for WSN.

The security requirements of wireless sensor networks are mainly data integrity and authenticity. In the data aggregation scheme, it is important that the data is not tampered with during the transmission. Therefore, we mainly focus on the protection of data integrity. The main consideration of our system model is to protect the integrity of the data, while reducing the bandwidth and storage cost of wireless sensor networks. Fig. 2 illustrates a wireless sensor networks system consists of four parts: data Center, aggregator, router and sensor nodes.

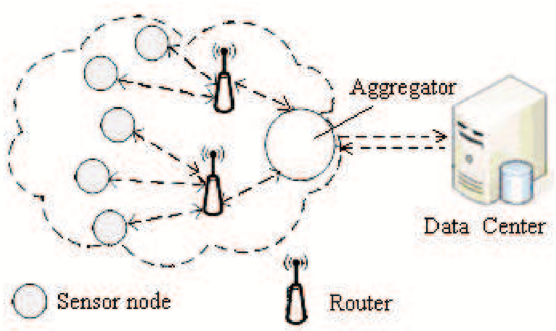

FIGURE 1. System model

- Data center has a strong computing power and storage space. Therefore, it can handle all the raw data collected by the sensor nodes. When the system initialization, data center will get the public key and private key pair (PK_d, SK_d), and release its public key PK_d. In our system, data center plays as the designated verifier.

- Aggregator is a special kind of sensor node with a certain ability of computing and communication range. It can sign messages collected from the physical world,can get the data center's public key from public channel. The aggregator can sign the signature from the sensor nodes into a signature, and can send the aggregated signature to the data center.
- Router transmits the data collected by the Sensor node to the aggregator.
- Sensor node's computing power, storage capacity, battery capacity and communication capacity are limited. In order to seek balance between the communication effect and energy loss, the communication ability is limited, so the messages should be forwarded to the data center through the aggregation. Identity information of each sensor node is represented by $info_i$. Certificate Authority (CA) issues a certificate $cert_i$ for each sensor node. Sensor node can use its private key and certificate to sign the message. In our system, each sensor node belongs to a cluster, and sends the signature to the aggregator.

2.2. **Framework of certificate based aggregate signature in WSN.**

2.2.1. *Definition of Certificate-Based Aggregate Signature in WSN.*

A certificate-based aggregate signature scheme consists of following seven algorithms.

Setup: This algorithm takes a security parameter 1^k as input. It returns the certificate authority (CA)'s master key and the CA's public key. Meanwhile, it returns public parameters *param* used to setup the system.

KeyGen: This algorithm takes public parameters *param* as inputs. It returns data center and sensor node i's private/public key pair (PK_i, SK_i).

CertGen: This algorithm takes public parameters *param*, the CA's public key, the sensor node i's public key PK_i, the sensor node i's identity ID_d, and the CA's master key as inputs. It returns a certificate $cert_i$ of sensor node i.

Sign: This algorithm takes a message m, public parameters *param*, sensor node i's certificate $cert_i$, and the sensor node i's private key SK_i as inputs. It returns a signature of m.

Verify: This algorithm takes a signature with message m, public parameters *param*, sensor node i's public key PK_i, and the CA's public key as inputs. It returns 1, if the verification is true. Otherwise, it returns reject.

Aggregate: This algorithm takes n divisional signatures, DC's public key, public parameters *param* as inputs. It returns an aggregate signature.

AggVerify: This algorithm takes an aggregate signature, DC's private key, and public parameters *param* as inputs. It returns 1, if the verification is true. Otherwise, it returns reject.

2.3. **Security Model of Certificate-Based Aggregate Signature in WSN.**

Here, we follow [36] to define our adversarial model and security model for single signature. There two types of adversaries called A_I and A_{II} in our scheme. We first define the adversarial model of A_I.

The ability of adversary A_I. We assume AI can replace the target $ID^{*'}$s public key. However, it cannot obtain the $ID^{*'}$s certificate and private key.

The ability of adversary A_{II}. We assume A_{II} can obtain the system master key. However, it cannot replace any ID's public key.

The security of our certificate based aggregate signature scheme in WSN for single signature against a public key replace attack is defined by the following Game 1 between A_I and a challenger C.

Setup. The challenger C runs the **Setup** algorithm to generate CA's master/public key pair and public parameters *param*. Meanwhile, C initializes lists L_K, L_C, L_{H1}, L_{H2} and L_S which are initially empty. Then, C returns CA's public key PK_{CA} and *param* to A_I.

Queries. A_I can adaptively make following queries to the challenger C.

(a) **KeyGen query:**Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns PK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i). Then, C sends PK_i to A_I and adds (ID_i, SK_i, PK_i) into L_K.

(b) **CertGen query:** Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_C. If ID_i has existed in L_C, C returns $Cert_i$ from L_C. Otherwise, C searches L_K with ID_i and runs the **CertGen** algorithm to generate node i's certificate $Cert_i$. Then, C sends $Cert_i$ to A_I and adds (ID_i, PK_i, $Cert_i$) into L_C.

(c) **Hash query:** Upon receiving this query with message m, C returns a random value as hash value to A_I.

(d) **Corrupt query:** Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns SK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i). Then, C returns SK_i to A_I and adds (ID_i, SK_i, PK_i) into L_K.

(e) **Sign query:** Upon receiving this query with (m_i, ID_i), C first checks the list L_S. If (m_i, ID_i) has existed in L_S, C returns (R_i, σ_i). Otherwise, C runs the **Sign** algorithm for (m_i, ID_i) to generate a signature $\rho_i = (R_i,\sigma_i)$. Then, C sends ρ_i to A_I and adds (m_i, ID_i, R_i,σ_i) into L_S.

Replacing public key request.A_I can request the replacement of ID_i's public key PK_i with a value selected by A_I in L_K. Note that A_I can make this request, repeatedly.

Forgery. Finally, A_I outputs a signature tuple (m_i^*, ID_i^*,ρ_i^*, PK_i^*)

We say that A_I wins Game 1, if the following conditions satisfied:

(1).The responds of the **Verify** algorithm on (m_i^*, ID_i^*,ρ_i^*, PK_i^*) is true.

(2).ID_i^* cannot be queried by **KeyGen**, **CertGen**, and the **corrupt queries**.

(3).(m_i^*, ID_i^*) cannot be queried by the **Sign query**.

The advantage of A_I wins Game 1 is defined by $Adv_{Game_1}^{A_I}(t)$.

Definition 1. We say that a certificate based aggregate signature scheme in WSN is secure against a public key replace attack for single signature, if for any adversary A_I the advantage $Adv_{Game_1}^{A_I}(t)$is negligible.

The security of our certificate based aggregate signature scheme in WSN for single signature against the certifier is defined by the following Game 2 between A_{II} and a challenger C.

Setup. The challenger C runs the **Setup** algorithm to generate CA's master/public key pair and public parameters *param*. Meanwhile, C initializes lists L_K, L_C, L_{H1}, L_{H2} and L_S which are initially empty. Then, C returns (*mpk*,*msk*,*param*) to A_{II}.

Queries. A_{II} can adaptively make following queries to the challenger C

(a) **KeyGen query:** Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns PK_i. Otherwise, C runs

the**KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i). Then, C sends PK_i to A_{II} and adds (ID_i, SK_i, PK_i) into L_K.
(b) **Hash query:** Upon receiving this query with message m, C returns a random value as hash value to A_{II}.
(c) **Corrupt query:** Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns SK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i). Then, C returns SK_i to A_{II} and adds (ID_i, SK_i, PK_i) into L_K.
(d) **Sign query:** Upon receiving this query with (m_i, ID_i), C first checks the list L_S. If (m_i, ID_i) has existed in L_S, C returns (R_i, σ_i). Otherwise, C runs the **Sign** algorithm for (m_i, ID_i) to generate a signature $\rho_i = (R_i,\sigma_i)$. Then, C sends ρ_i to A_{II} and adds (m_i, ID_i, R_i,σ_i) into L_S.
Forgery. Finally, A_{II} outputs a signature tuple (m_i^*, ID_i^*,ρ_i^*, PK_i^*)

We say that A_{II} wins Game 2, if the following conditions satisfied:

(1).The responds of the **Verify** algorithm on (m_i^*, ID_i^*,ρ_i^*, PK_i^*) is true.

(2).ID_i^* cannot be queried by **KeyGen** and the **corrupt** queries.

(3).(m_i^*, ID_i^*) cannot be queried by the **Sign query**.

The advantage of A_{II} wins Game 2 is defined by $Adv_{Game_2}^{A_{II}}(t)$.

Definition 2. We say that a certificate based aggregate signature scheme in WSN is secure against the certifier for single signature, if for any adversary A_{II} the advantage $Adv_{Game_2}^{A_{II}}(t)$is negligible.

The goal of adversary A_I is to forge a valid signature under a public key PK_i^* without the corresponding certificate $certi_i^*$.The goal of adversary A_{II} who has the system master key is to forge a valid signature under a public key PK_i^*.

Here, we define our security model for aggregate signature.

The security of our certificate based aggregate signature scheme in WSN for aggregate signature against a public key replace attack is defined by the following Game 3 between A_I and a challenger C. Note that the goal of adversary A_I is to forge a valid aggregate signature under public keys PK_1, PK_2,...,PK_n without the corresponding certificate $certi_i^*$, where $1 \leq i \leq n$.

Setup. The challenger C runs the **Setup** algorithm to generate CA's master/public key pair and public parameters *param*. Meanwhile, C initializes lists L_{H1}, L_{H2} and L_S which are initially empty. Then, A_I is provided *param* and PK_1, without loss generality. Note that PK_1 is the target user's public key.
Queries. A_I can adaptively make queries to the challenger C.
(a) **Hash query:**Upon receiving this query with message m_i, C returns a random value as hash value to A_I.
(b) **Sign query:** Upon receiving this query with (m_i, ID_i), C first checks the list L_S. If (m_i, ID_i) has existed in L_S, C returns (R_i, σ_i). Otherwise, C runs the **Sign** algorithm for (m_i, ID_i) to generate a signature $\rho_i = (R_i,\sigma_i)$. Then, C sends ρ_i to A_I and adds (m_i, ID_i, R_i,σ_i) into L_S.

Forgery. Finally, A_I outputs a value emphk(where $k \leq n$), *k-1* additional public keys PK_2,..., PK_k,k distinct messages m_1,m_2,...,m_k, and a corresponding aggregate signature σ_i^* under PK_1,PK_2,..., PK_k.

We say that A_I wins Game 3, if the following conditions satisfied:

(1).The responds of the Aggregate **Verify** algorithm on (m_i, ID_i,σ^*, PK_i) is true.where $1 \leq i \leq n$.

(2).PK_1 must be included in the set of PK_i.

(3).(m_1, ID_1) must not be queried by **Sign query**.

The advantage of A_I wins Game 3 is defined by $Adv_{Game_3}^{A_I}(t)$.

Definition 3. We say that a certificate based aggregate signature scheme in WSN is secure against existential forgery for aggregate signature, if for any adversary A_I the advantage $Adv_{Game_3}^{A_I}(t)$ is negligible.

The security of our certificate based aggregate signature scheme in WSN for aggregate signature against the certifier is defined by the following Game 4 between A_{II} and a challenger C. Note that the goal of adversary A_{II} who has the system master key is to forge a valid aggregate signature under public key PK_1, PK_2,..., PK_n.

Setup. The challenger C runs the **Setup** algorithm to generate CA's master/public key pair and public parameters *param*. Meanwhile, C initializes lists L_K,L_{H1},L_{H2} and L_S which are initially empty. Then, C returns (*msk*,*param*,PK_1) to A_{II}, where PK_1 is the target user's public key.

Queries. A_{II} can adaptively make queries to the challenger C

(a) **KeyGen query:** Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns PK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i). Then, C sends PK_i to A_{II} and adds (ID_i, SK_i, PK_i) into L_K.

(b) **Hash query:** Upon receiving this query with message m_i, C returns a random value as hash value to A_{II}.

(c) **Sign query:** Upon receiving this query with (m_i, ID_i), C first checks the list L_S. If (m_i, ID_i) has existed in L_S, C returns (R_i, σ_i). Otherwise, C runs the **Sign** algorithm for (m_i, ID_i) to generate a signature ρ_i=(R_i,σ_i). Then, C sends ρ_i to A_{II} and adds (m_i, ID_i, R_i,σ_i) into L_S.

Forgery. Finally, A_{II} outputs an aggregate signature σ^* on message m_1,m_2,...,m_n, under public keys PK_1,PK_2,...,PK_n.

We say that A_{II} wins Game 4, if the following conditions satisfied:

(1).The responds of the Aggregate **Verify** algorithm on (m_i,ID_i,σ^*,PK_i) is true.

(2).PK_i must be included in the L_K.where $1 \leq i \leq n$.

(3).(m_1, ID_1) cannot be queried by the **Sign query**.

The advantage of A_{II} wins Game 4 is defined by $Adv_{Game_4}^{A_{II}}(t)$.

Definition 4. We say that a certificate based aggregate signature scheme in WSN is secure against the certifier for aggregate signature, if for any adversary A_{II} the advantage $Adv_{Game_4}^{A_{II}}(t)$ is negligible.

3. Proposed Certificate-Based Aggregate Signature in WSN.

3.1. Bilinear pairings.

Let G_1 and G_2 be two additive cyclic groups with large prime order q, where G_1 is a subgroup of abelian group $E(\mathrm{F}_P)$ and G_2 is a subgroup of finite field F_P. A bilinear pairing e is a map defined by e:$G_1 \times G_1 \rightarrow G_2$ satisfying the following three properties:

(1). Bilinear: For $P, Q \in G_1$ and $a, b \in Z_q^*$, $e(aP, bQ) = e(P, Q)^{ab}$.

(2). Non-degenerate: For an identity $1_{G1} \in G_1$, $e(1_{G1}, 1_{G1})$ is also an identity of G_2.

(3). Computable. There exist several efficient algorithms to compute $e(P,\ Q)$ for $P, Q \in G_1$.

For details about bilinear pairings, the readers can refer to [31, 32, 33, 34, 35, 37, 38, 39] for a full descriptions.

3.2. **Proposed scheme.** In our scheme, five roles are involved which are certificate authority (CA), data center, sensor node, router and aggregator. Note that data center plays as verifier, sensor node plays as signer, and aggregator is a special node belongs to a cluster. We assume there are n sensor nodes in our system.Our scheme describes as follows.

Setup:CA executes the **Setup** algorithm to generate keys and parameters.
(a) A bilinear pairing e is chosen mentioned above.
(b) Selecting CA's master key $s \in_R Z_q^*$, two different generators P and Q in G_1 and computing the corresponding public key $PK_{CA} = sP$.
(c) Four cryptographic hash functions are chosen, $H_1 : \{0,1\}^* \rightarrow G_1$, and $H_2 : \{0,1\}^* \rightarrow Z_q^*$, $H_3 : G_2 \rightarrow Z_q^*$, respectively.

Finally, the public parameters $param$ is defined as $\{G_1, G_2, e,\ q, P, Q, H_1, H_2, H_3\}$.

KeyGen:CA generates data center (DC) and each sensor node i's private/public key pair as follows.
(a) Selecting $x \in_R Z_q^*$, as DC's private key and computing xP as DC's public key.
(b) For each sensor node i for $i = 1,\ 2,...,n$, selecting $x_i \in_R Z_q^*$ as i's private key and computing $PK_i = x_i P$ as i's public key.

CertGen: Each sensor node i submits its public key PK_i and identity ID_i to CA over an authentic channel. Upon receiving the request, CA computes $Q_i = H_1(PK_i \| ID_i)$ and returns the corresponding certificate $Cert_i = s \cdot Q_i$, where ID_i denotes the identity of sensor node i.

Sign: To sign a message m_i, sensor node i selects $r_i \in_R Z_q^*$ and computes $R_i = r_i \cdot Q$, $h_i = H_2(m_i \| R_i)$, and $\sigma_i = Cert_i \cdot h_i + (x_i + r_i)Q$. Finally, the signature of m_i is defined by (R_i, σ_i).

Verify: To verify a signature tuple (m_i, R_i, σ_i) on messages m_i, any verifier can verify it by the equation:

$$e(\sigma_i, P) = e(PK_{CA}, h_i \cdot Q_i)e(PK_i, Q)e(R_i, P) \tag{1}$$

Aggregate: When an aggregator receives n signature tuples (m_i, R_i, σ_i) from sensor node i, for $i = 1,\ 2,...,\ n$, it computes $\sigma = H_3(e(\sigma', PK_{DC}))$, where $\sigma' = \sum_{i=1}^{n} \sigma_i$. Here, $(\sigma, R_1, R_2, ..., R_n)$ is an aggregate signature with identities $(ID_1, ID_2, ..., ID_n)$ on messages $(m_1, m_2, ..., m_n)$.

AggVerify: To verify an aggregate signature$(\sigma, R_1, R_2, ..., R_n)$ with identities $(ID_1, ID_2, ..., ID_n)$ on messages $(m_1, m_2, ..., m_n)$, the data center computes $Q' = \sum_{i=1}^{n} h_i Q_i$. and verifies

$$\sigma = H_3(e(Q', x \cdot PK_C)\prod_{i=1}^{n} e(Q, x \cdot PK_i)e(R_i, PK_{DC})) \tag{2}$$

where $Q_i = H_1(PK_i \| ID_i)$ and $h_i = H_2(m_i \| R_i)$.

Here, we provide the correctness of our scheme

$$\begin{aligned}
\sigma &= H_3(e(\sigma_1, PK_{DC}), ..., e(\sigma_n, PK_{DC})) && (3)\\
&= H_3(e(cert_1 h_1 + (x_1 + r_1)Q, PK_{DC}), ..., e(cert_n h_n + (x_n + r_n)Q, PK_{DC}) && (4)\\
&= H_3(e(sQ_1 h_1 + (x_1 + r_1)Q, xP), ..., e(sQ_n h_n + (x_n + r_n)Q, xP) && (5)\\
&= H_3(e(h_1 Q_1, x \cdot sP)e((x_1 Q + r_1 Q), xP), ..., e(h_n Q_n, x \cdot sP)e(x_n Q + r_n Q, xP) && (6)\\
&= H_3(e(h_1 Q_1, x \cdot PK_c)e(Q, x \cdot PK_1)e(R_1 PK_{DC}), .. && (7)\\
&\quad , e(h_n Q_n, x \cdot PK_c)e(Q, x \cdot PK_n)e(R_n PK_{DC})) && (8)\\
&= H_3(e(Q', x \cdot PK_C)\prod_{i=1}^{n} e(Q, x \cdot PK_i)e(R_i, PK_{DC})) && (9)
\end{aligned}$$

4. **Security Analysis.** The security of our scheme is based on the Computational Diffie-Hellman Problem (CDHP) and Bilinear Diffie-Hellman Problem (BDHP). Here, we provide definition and assumptions. **Definition (CDHP).**Given P, aP, $bP \in G_1$ for $a,b \in Z_q^*$ unknown, the CDHP is to compute $abP \in G_1$.

Definition (CDHP assumption). No probabilistic polynomial time algorithm can solve this problem.

Definition (BDHP). Given P, aP, bP, $cP \in G_1$ for $a,b,c \in Z_q^*$ unknown, the BDHP is to compute $e(P,P)^{abc} \in G_2$.

Definition (BDHP assumption). No probabilistic polynomial time algorithm can solve this problem.

4.1. **Unforgeability for single signature. Theorem 1.**In the random oracle model and under the computational Diffie-Hellman (CDH) assumption, Assume that there exists an adversary A_I can forge a valid single signature tuple of our scheme with a non-negligible advantage$Adv_{Game_1}^{A_I}(t)$. Then, there exists an algorithm C can solve the CDH problem with a non-negligible advantage.

Proof. We show that if there is an adversary A_I can forge the above signature scheme with non-negligible advantage. Then a challenger C will solves Computational Diffie-Hellman Problem. Challenger C will interact with A_I as described below:

Setup. The challenger C runs the **Setup** algorithm to generate public parameters $param = (G_1, G_2, e, q, P, Q,$
$H_1, H_2, H_3)$ and sets CA's public key $PK_{CA} = aP$.

Meanwhile, C initializes lists L_K, L_C, L_{H1}, L_{H2} and L_S which are initially empty. Then, C returns CA's public key PK_{CA} and $param$ to A_I.

Queries. A_I can adaptively make following queries to the challenger C:

(a) **KeyGen query:**Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns PK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i)=(x_i, x_iP). Then, C sends PK_i to A_I and adds (ID_i, SK_i,PK_i) into L_K.

(b) **CertGen query:** Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_C. If ID_i has existed in L_C, C returns $Cert_i$ from L_C. Otherwise, C searches L_K with ID_i If ID_i's public key has been replaced, C aborts. Otherwise, C searches L_{H1} with (ID_i, PK_i).If it does not appear in the list, C can adds (ID_i,PK_i,$coin_i$, q_i,Q_i) on the listL_{H1} as the same way it responds to H1 queries. Otherwise, C chooses a random number $coin_i \in \{0,1\}$,such that $Pr[coin_i = 1] = \delta$,where $\delta = 1/(\ q_c + \ q_s)$ where q_c is the maximum number A_I makes to the **CertGen query**, q_s is the maximum number A_I makes to the **sign query**,then C generates node i's certificate as fallows.

(1).If $coin_i$=1,C outputsfailure and aborts.
(2).If $coin_i$=0, C computes $Cert_i$= $q_i PK_{CA}$=$q_i aP$, and returns it to A_I.
(c) **H1 query:** Upon receiving this query with ($PK_i \| ID_i$),C first checks the list L_{H1}. If($PK_i \| ID_i$) appears in a tuple (ID_i,PK_i,$coin_i$,q_i, Q_i) on the list, C returns Q_i as hash value to A_I,Otherwise,
(1).If $coin_i$=1, C sets Q_i=bP and adds (ID_i,PK_i,$coin_i$,q_i, Q_i) on the list L_{H1}.
(2).If $coin_i$=0, C chooses a random number$q_i \in Z_q^*$, and calculates Q_i=$q_i P$.It then adds (ID_i,PK_i,$coin_i$,q_i, Q_i) on the list L_{H1}.

In either case, C returns Q_i to the adversary A_I.

(d) **H2 query:**Upon receiving this query with ($m_i \| R_i$), C first checks the list L_{H2}. If ($m_i \| R_i$) appears in the list, C outputs $h_i = H_2(m_i \| R_i)$ as answer to A_I,otherwise, C returns a random value $h_i \in Z_q^*$ as hash value to A_I, and adds (m_i, R_i, h_i) on the list L_{H2}.
(e) **Corrupt query:**Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns SK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i). Then, C returns SK_i to A_I and adds (ID_i,SK_i, PK_i) into L_K. If ID_i's public key has been replaced, C returns $\perp$.
(f) **Sign query:** Upon receiving this query with (m_i, ID_i), C first checks the list L_S. If (m_i, ID_i) has existed in L_S, C returns (R_i, σ_i). Otherwise,C checks list L_K to find the ID_i's original secret key x_i. If ID_i's public key has been replaced, C returns $\perp$.Otherwise, it finds the tuple (ID_i,PK_i,$coin_i$, q_i, Q_i) in the list L_{H1}.

(1).If $coin_i$=1, C aborts.

(2).If $coin_i$=0, C checks list L_{H1} to obtain (q_i, Q_i),and calculates $Cert_i = q_i PK_{CA}$, and then checks list L_{H2} to obtain (m_i,R_i,h_i).

If (m_i,R_i) does not appears in the list, C will adds (m_i,R_i,h_i) in list L_{H2} as responds to H2 queries. Finally, C runs the **Sign** algorithm for (m_i,ID_i) to generate a signature $\sigma_i = Cert_i \cdot h_i + (x_i + r_i)Q$.

Then, C sends σ_i to A_I and adds (m_i, ID_i,R_i, σ_i) into L_S.

Replacing public key request. Upon receiving this query with (ID_i, $PK_i^{'}$).C replaces (ID_i,SK_i,PK_i) with (ID_i, $\perp$, $PK_i^{'}$)into L_K.

Forgery. Finally, A_I outputs a signature tuple (m_i^*,ID_i^*,σ_i^*,PK_i^*)

Assume that an A_I adversary A can forge a valid signature σ_i^* with a non-negligible advantage.

Then, applying Forking Lemma, A with a non-negligible advantage can forge another signature $\sigma_i^{'}$ in the same random tape and under different oracle. Then, we can obtain

$$\sigma_i^*{=}Cert_i \cdot h_i{+}(x_i{+}r_i)Q \text{ and } \sigma_i^{'}{=}Cert_i \cdot h_i^{'}{+}(x_i{+}r_i)Q$$

It implies that σ_i^*-$\sigma_i^{'}$=$Cert_i \cdot (h_i$-$h_i^{'})$.Set PK_{CA}=aP and Q_i=bP.$Cert_i$=abP=$(\sigma_i^*$-$\sigma_i^{'})/(h_i$-$h_i^{'})$, a contradiction.

Therefore, if A_I can forge a valid single signature tuple of our scheme with a non-negligible advantage $Adv_{Game_1}^{A_I}(t)$,C can solve the CDH problem with a non-negligible advantage $\varepsilon^{'} \geq (1-\delta)^{q_c}\delta(1-\delta)^{q_s} Adv_{Game_1}^{A_I}(t) \geq \dfrac{1}{e(q_c+q_s)} Adv_{Game_1}^{A_I}(t)$,where q_c is the maximum number A_I makes to the **CertGen query**, q_s is the maximum number A_I makes to the **sign query**, and e is the base of natural logarithm.

Theorem 2. In the random oracle model and under the Bilinear Diffie-Hellman (BDH) assumption, assume that there exists an adversary A_{II} can forge a valid single signature tuple of our scheme with a non-negligible advantage. Then, there exists an algorithm C can solve the BDH problem with a non-negligible advantage.

Proof.

We show that if there is an adversary A_{II} can forge the above signature scheme with non-negligible advantage. Then a challenger C will solves Bilinear Diffie-Hellman Problem. Challenger C will interact with A_{II} as described below:

Setup. The challenger C runs the **Setup** algorithm to generate public parameters $param$=(G_1, G_2, e,q,P,
H_1, H_2, H_3) and sets Q=bP,$PK_{CA} = sP$, where $s \in Z_q^*$. Meanwhile, C initializes lists L_K, L_C, L_{H1}, L_{H2} and L_S which are initially empty. Then, C sends CA's secret key SK_{CA} and $param$=(G_1, G_2, e,q,P, H_1, H_2, H_3) to A_{II}.

Queries.A_{II} can adaptively make following queries to the challenger C.

(a)**KeyGen query:**Upon receiving this query for sensor node i with identity ID_i,
(1).If $ID_i \neq ID_i^*$,C checks the list L_K. If ID_i has existed in L_K, C returns PK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i, PK_i)=(x_i,x_iP). Then, C sends PK_i to A_{II} and adds (ID_i, SK_i, PK_i) into L_K.
(2).If $ID_i = ID_i^*$,C returns PK_i=aP and adds (ID_i,$\perp$, PK_i) into L_K.
(b)**H1 query:**Upon receiving this query with ($PK_i \parallel ID_i$), C first checks the list L_{H1}. If ($PK_i \parallel ID_i$) appears in a tuple (ID_i, PK_i, $coin_i$,q_i,Q_i) on the list, C returns Q_i as hash value to A_{II},Otherwise, C chooses a random number $coin_i \in \{0, 1\}$,such that Pr[$coin_i$=1]=δ, where $\delta = 1/q_a$ and q_a is the number of all queries made by A_{II}.
(1).If $coin_i = 1$,C sets Q_i=cP and adds (ID_i, PK_i, $coin_i$, q_i, Q_i) on the list L_{H1}.
(2).If $coin_i = 0$,C chooses a random number $q_i \in Z_q^*$, and calculates $Q_i = q_iP$. It then adds (ID_i, PK_i, $coin_i$,
q_i, Q_i) on the list L_{H1}.

In eigher case, C returns Q_i to the adversary A_{II}.

(c)**H2 query:**Upon receiving this query with ($m_i \parallel R_i$), C first checks the list L_{H2}. If ($m_i \parallel R_i$) appears in the list, C outputs $h_i = H_2(m_i \parallel R_i)$ as answer to A_{II},otherwise, C generates the hash value as follows:
(1).If $(m_i \parallel R_i) \neq (m_i^* \parallel R_i^*)$,C chooses a random number $h_i \in Z_q^*$ as hash value to A_{II}, and adds (m_i, R_i, h_i) on the list L_{H2}.
(2).If $(m_i \parallel R_i) = (m_i^* \parallel R_i^*)$,C sets $h_i = s^{-1}$, and returns h_i to A_{II}. Then (m_i^*, R_i^*, h_i) was add into L_{H2}.
(d)**Corrupt query:**Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns SK_i. Otherwise, C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i,PK_i). If $ID_i \neq ID_i^*$, C returns SK_i to A_{II} and adds (ID_i,SK_i,PK_i) into L_K.Otherwise, C returns $\perp$.
(e)**Sign query:**Upon receiving this query with (m_i,ID_i), C first checks the list L_S. If(m_i,ID_i) has existed in L_S,C returns (R_i,σ_i). Otherwise, C checks list L_K to find the ID_i's original secret key x_i.
(1).If$ID_i = ID_i^*$, C aborts.
(2).If$ID_i \neq ID_i^*$,C checks list L_{H1} to obtain (q_i, Q_i),and calculates $Cert_i = sQ_i$, and then checks list L_{H2} to obtain (m_i,R_i,h_i). If (m_i,R_i) does not appears in the list, C will adds (m_i,R_i,h_i) in list L_{H2} as responds to **H2 queries**. Finally, C runs the **Sign** algorithm for (m_i, ID_i) to generate a signature σ_i=$Cert_i \cdot h_i$+$(x_i + r_i)Q$. Then, C sends σ_i to A_{II} and adds (m_i,ID_i,R_i, σ_i) into L_S.

Forgery. Finally, A_{II} outputs a signature tuple (m_i^*,ID_i^*,σ_i^*,PK_i^*).

Assume that A_{II} can forge a valid signature tuple (m_i^*,R_i^*,σ_i^*). Then,σ_i^* can be expressed as $\sigma_i^* = sH_2(m_i^* \parallel R_i^*) \cdot Q_i^* + x_i^*Q + R_i^*$. Then, A_{II} can compute $e(\sigma_i^* - s \cdot H_2(m_i^* \parallel$

$R_i^*) \cdot Q_i^* - R_i^*, Q_i^*)$=$e(abP,cP) = e(P,P)^{abc}$ for given $PK_i^* = aP, Q = bP, Q_i^* = cP$ and $h_i = s^{-1}$.

Therefore, if A_{II} can forge a valid single signature tuple of our scheme with a non-negligible advantage $Adv_{Game_2}^{A_{II}}(t)$, C can solve the CDH problem with a non-negligible advantage $\varepsilon' \geq (\frac{1}{n})^{q_k}\delta(1-\frac{1}{n})^{q_s} Adv_{Game_2}^{A_{II}}(t) \geq \frac{1}{eq_a n^{q_a}} Adv_{Game_1}^{A_I}(t)$,where q_k is the maximum number A_{II} makes to the **KeyGen** query, q_s is the maximum number A_{II} makes to the **Sign query**, q_a is the number of all queries made by A_{II}, and n is the number of total users.

4.2. **Unforgeability for aggregate signature.** **Theorem 3.**In the random oracle model and under the computational Diffie-Hellman (CDH) assumption, assume that there exists an adversary A_I can forge a valid aggregate signature tuple of our scheme with a non-negligible advantage $Adv_{Game_3}^{A_I}(t)$. Then, there exists an algorithm C can solve the CDH problem with a non-negligible advantage.

Proof. We show that if there is an adversary A_I can forge a valid aggregate signature scheme with non-negligible advantage.

Then a challenger C will solves the CDH Problem. Challenger C will interact with A_I as described below:

Setup. The challenger C runs the **Setup** algorithm to generate public parameters $param$=(G_1, G_2, e, q,
P, H_1, H_2, H_3) and sets CA's public key $PK_{CA} = aP$, $Q = tP$ for some $t \in Z_q^*$. Meanwhile, C initializes lists L_{H1}, L_{H2} and L_S which are initially empty. Then, C returns CA's public key PK_{CA}, PK_1,Q and $param$ to A_I. Where PK_1 is target user i's publick key.

Queries. A_I can adaptively make following queries to the challenger C:

(a)**H1 query:** Upon receiving this query with $(PK_i \| ID_i)$,C first chooses a random number $coin_i \in \{0,1\}$,such that $\Pr[coin_i = 1]$=δ,where the value of δ=$1/q_s$, and q_s is the maximum number A_I makes to the **Sign query**.

(1).If $coin_i$=1, C sets Q_i=bP and adds (ID_i,PK_i,$coin_i$,$\perp$, Q_i) on the list L_{H1}.

(2).If $coin_i$=0, C chooses a random number$q \in Z_q^*$, and computes Q_i=qP.

It then adds (ID_i,PK_i,$coin_i$,q, Q_i) on the list L_{H1}.

In either case, C returns Q_i to the adversary A_I.

(b) **H2 query:**Upon receiving this query with $(m_i \| R_i)$, C first checks the list L_{H2}. If $(m_i \| R_i)$ appears in the list, C outputs $h_i = H_2(m_i \| R_i)$ as answer to A_I,otherwise, C returns a random value $h_i \in Z_q^*$ as hash value to A_I, and adds (m_i, R_i, h_i) on the list L_{H2}.

(c) **Sign query:** Upon receiving this query with (m_i, ID_i), C first checks the list L_S. If (m_i, ID_i) has existed in L_S, C returns (R_i, σ_i). Otherwise,

(1).If $coin_i$=1, C aborts.

(2).If $coin_i$=0, C checks list L_{H1} to obtain q,and calculates $Cert_i = qPK_{CA}$, and then checks list L_{H2} to obtain (m_i,R_i,h_i). If (m_i,R_i) does not appears in the list, C will adds (m_i,R_i,h_i) in list L_{H2} as responds to H2 queries. Finally, C runs the **Sign** algorithm for (m_i,ID_i) to generate a signature $\sigma_i = Cert_i \cdot h_i + t \cdot PK_i + R_i$. Then, C sends σ_i to A_I and adds (m_i,R_i, σ_i) into L_S.

Forgery. A_I outputs a vallue K (where $K \leq n$), K-1 additional public keys PK_2,... PK_k, k distinct messagesm_1, m_2,..., m_k,a corresponding aggregate signature σ^* under PK_1,PK_2,...,PK_k.

Assume that A_I can forge a valid aggregate signature $\sigma^* = \sum_{i=1}^{k} \sigma_i = cert_1 \cdot h_1 + (x_1 + r_1) \cdot Q + \sum_{i=2}^{k} \sigma_i$ with a non-negligible advantage. Then, applying Forking Lemma [40], A

with a non-negligible advantage can forge another signature $\sigma^{'} = \sum_{i=1}^{k} \sigma_i^{'} = cert_1 \cdot h_1^{'} + (x_1 + r_1) \cdot Q + \sum_{i=2}^{k} \sigma_i$ in the same random tape and under different oracle. Then, we can obtain:

$$\sigma_1 = Cert_1 \cdot h_1 + (x_1 + r_1) \cdot Q \tag{10}$$

$$\sigma_1^{'} = Cert_1 \cdot h_1^{'} + (x_1 + r_1) \cdot Q \tag{11}$$

It implies that σ_1-$\sigma_1^{'}$=$Cert_1(h_1 - h_1^{'})$.Set PK_{CA}=aP and Q_1=bP.$Cert_1$=abP=(σ_1-$\sigma_1^{'}$)/(h_1-$h_1^{'}$), a contradiction.

Therefore, if A_I can forge a valid single signature tuple of our scheme with a non-negligible advantage $Adv_{Game_3}^{A_I}(t)$,C can solve the CDH problem with a non-negligible advantage $\varepsilon^{'} \geq \delta(1-\delta)^{q_s} Adv_{Game_3}^{A_I}(t) \geq \frac{1}{eq_s} Adv_{Game_3}^{A_I}(t)$,where q_s is the maximum number A_I makes to the **sign query**, and e is the base of natural logarithm.

Theorem 4. In the random oracle model and under the Bilinear Diffie-Hellman (BDH) assumption, assume that there exists an adversary A_{II} can forge a valid single signature tuple of our scheme with a non-negligible advantage $Adv_{Game_4}^{A_{II}}(t)$. Then, there exists an algorithm C can solve the BDH problem with a non-negligible advantage.

Proof. We show that if there is an adversary A_{II} can forge the above signature scheme with non-negligible advantage. Then a challenger C will solves the BDH Problem. The challenger C will interact with A_{II} as described below:

Setup. The challenger C runs the **Setup** algorithm to generate public parameters $param$=(G_1, G_2, e,q,P,
H_1, H_2, H_3) and sets CA's public key $PK_{CA} = sP$, Q=bP. Meanwhile, C initializes lists L_K, L_{H1}, L_{H2} and L_S which are initially empty. Then, C returns CA's public key PK_{CA} target user's public key $PK_1 = ap$, s and $param$ to A_{II}.

Queries.A_{II} can adaptively make following queries to the challenger C.

(a)**KeyGen query:**Upon receiving this query for sensor node i with identity ID_i, C first checks the list L_K. If ID_i has existed in L_K, C returns PK_i.Otherwise,

(1).If $ID_i = ID_1$,C sends PK_1 to A_{II} and adds (ID_1, $\perp$, PK_1) into L_K.

(2).If $ID_i \neq ID_1$,C runs the **KeyGen** algorithm for ID_i to generate node i's private/public key pair (SK_i,PK_i). Then, C sends PK_i to A_{II} and adds (ID_i,SK_i,PK_i) into L_K.

(b)**H1 query:**Upon receiving this query with ($PK_i \parallel ID_i$), C first chooses a random number $coin_i \in \{0, 1\}$,such that Pr[$coin_i$=1]=δ, where δ=1/qs and qs is the maximum number A_{II} makes to the **Sign query**.

(1).If $coin_i = 1$,C sets Q_i=cP and adds (ID_i, PK_i, $coin_i$, $\perp$, Q_i) on the list L_{H1}.

(2).If $coin_i = 0$,C chooses a random number $q_i \in Z_q^*$, and calculates $Q_i = q_i P$.Then, C adds (ID_i, PK_i, $coin_i$,
q_i, Q_i) on the list L_{H1}.

In eigher case, C returns Q_i to the adversary A_{II}.

(c)**H2 query:**Upon receiving this query with ($m_i \parallel R_i$), C first checks the list L_{H2}. If ($m_i \parallel R_i$) appears in the list, C outputs $h_i = H_2(m_i \parallel R_i)$ as answer to A_{II},otherwise, C generates the hash value as follows:

(1).If $coin_i = 0$,C chooses a random number $h_i \in Z_q^*$ as hash value to A_{II}, and then adds (m_i, R_i, h_i) on the list L_{H2}.

(2).If $coin_i = 1$,C sets $h_i = s^{-1}$, and returns h_i to A_{II}. Then,C adds(m_i^*, R_i^*, h_i) into L_{H2}.

(d)**Sign query:**Upon receiving this query with (m_i,ID_i), C first checks the list L_S. If(m_i,ID_i) has existed in L_S,C returns (R_i,σ_i). Otherwise,

(1). If$coin_i = 1$, C aborts.

(2). If$coin_i = 0$,C first checks list L_K to obtain x_i,then C checks list L_{H1} to obtain q_i, and computes $Cert_i = q_i PK_{CA}$. Meanwhile, C checks list L_{H2} to obtain (m_i, R_i,h_i). If (m_i, R_i) does not appears in the list, C will adds (m_i,ID_i,R_i,h_i) in list L_{H2} as responds to **H2 queries**. Finally, C runs the **Sign** algorithm for (m_i,ID_i) to generate a signature $\sigma_i = Cert_i \cdot h_i + x_i + R_i$. Then, C sends σ_i to A_{II} and adds (m_i,ID_i,R_i, σ_i) into L_s.

Forgery: A_{II} outputs an aggregate signature σ^*, on message m_1,m_2,...,m_n under PK_1,PK_2,...,PK_n.

Assume that A_{II} can forge a valid aggregate signature tuple (R^*,σ^*). Then, σ^* can be expressed as $\sigma^* = \sum_{i=1}^{n} \sigma_i = cert_1 h_1 + (x_1 + r_1) Q + \sum_{i=2}^{n} \sigma_i$.Then, A_{II} can compute $\sigma_1 = \sum_{i=1}^{n} \sigma_i - \sum_{i=2}^{n} \sigma_i = cert_1 \cdot h_1 + (x_1 + r_1) \cdot Q$ and $e(\sigma_1 - s \cdot H_2(m_1^* \parallel R_1^*) \cdot Q_1^* - R_1^*, Q_1^*) = e(abP, cP) = e(P,P)^{abc}$ for give $PK_1 = aP, Q = bP, Q_1^* = cP$ and $h_i = s^{-1}$.

Therefore, if A_{II} can forge a valid single signature tuple of our scheme with a non-negligible advantage $Adv_{Game_4}^{A_{II}}(t)$, C can solve the CDH problem with a non-negligible advantage $\varepsilon' \geq \delta^2(1-\delta)^{q_s} Adv_{Game_4}^{A_{II}}(t) \geq \frac{1}{e q_s^2} Adv_{Game_4}^{A_{II}}(t)$,where q_s is the maximum number A_{II} makes to the **sign query**, and e is the base of natural logarithm.

After all, the proposed certificate based aggregate scheme is unforgeable under the hardness assumption of Computational Diffie-Hellman Problem and Bilinear Diffie-Hellman Problem.

5. **Performance analysis.** In this section, we analyze the efficiency of the proposed signature scheme. Table 1 gives the communication cost comparison of two versions: un-aggregate scheme and aggregate scheme. To the best of our knowledge, the proposed scheme is the first certificate aggregate scheme proposed for wireless sensor networks. Therefore, we compare our scheme with Other's scheme.

The comparison is performed in terms of computation cost.

TABLE 1. Performance comparison of communication cost

	Un-aggregate	aggregate
Sensors→ Aggregator	$n \mid G_1 \mid + \mid m \mid$	$n \mid G_1 \mid + \mid m \mid$
Aggregator→ Data Center	$n \mid G_1 \mid + \mid m \mid$	$\mid G_1 \mid + \mid m \mid$

Definition of notations in table 1 is as follows:

Aggregate: aggregate scheme;

un-aggregate: un-aggregate scheme;

$\mid m \mid$: the overall length of $\{m_1, m_2, ..., m_n\}$;

$\mid G_1 \mid$: the overall length of value in G_1.

The results indicate the efficiency of the proposed scheme. We summarize the results in Table 2 where the following notations are used:

T_G: computation time for a multiplication in a multiplicative group or an addition in an additive group.

T_{Exp}: computation time for an exponentiation in a multiplicative group or an multiplication in an additive group.

T_{BP}: computation time of one bilinear pairing operation.

TABLE 2. Efficiency Comparison of Some Aggregate Signature Schemes

Schemes	Sign	Aggregate	Verify
KYLH[24]	$3T_{Exp}+T_G+2T_h$	$(n\text{-}1)T_G$	$nT_G+T_{Exp}+3T_h$
HTHW[25]	$2T_{Exp}+T_G+T_h$	$3nT_{BP}+(2n\text{-}1)T_G+nT_{Exp}+2nT_h$	$3T_{BP}+(3n\text{-}1)T_G+nT_{Exp}+(2n\text{-}1)T_h$
SMLW[26]	$2T_{Exp}+T_G+T_h$	$nT_{BP}+(n\text{-}1)T_G+T_{Exp}+T_h$	$(3n+1)T_{BP}+3nT_{Exp}+(2n+1)T_h$
SMLM[27]	$T_{Exp}+T_h$	$nT_{BP}+(n\text{-}1)T_G+T_{Exp}+T_h$	$(2n+1)T_{BP}+2nT_{Exp}+(n+1)T_h$
ours	$3T_{Exp}+T_G+T_h$	$T_{BP}+(n\text{-}1)T_G+T_h$	$(2n+1)T_{BP}+(n\text{-}1)T_G+(2n+1)T_{Exp}+(2n+1)T_h$

T_h: computation time of one hash operation.
n: the number of signers.

In wireless sensor networks, the computational power of nodes is very limited. The proposed aggregate signature scheme needs less computation in the process of aggregation and is suitable for data transmission in wireless sensor networks.

6. **Conclusions.** Certificate-based aggregate signature enables any user to combine n signatures signed by different n signers on different n messages into a short signature. Combining the characteristics of wireless sensor networks with the concept of certificate-based aggregate signature, in this paper, we present a certificate-based aggregate signature scheme for wireless sensor networks which can significantly improve the data transmission efficiency of wireless sensor networks. And then we proved the scheme's security under the Computational Diffie-Hellman Problem assumption.

Acknowledgment. The work of Tsu-Yang Wu was supported by the Natural Science Foundation of Fujian Province under Grant no. 2018J01636. This work was supported by the National Natural Science Foundation of China (No.61304199), Fujian Science and Technology Department (No.2013HZ0002-1, No.2014H0008); the Natural Science Foundation of Fujian Provincial Department of Education under Grant No. JAT160306. Key Laboratory of Applied Mathematics of Fujian Province University (Putian University) (NO. SX201705), Natural Science Foundation of Zhang zhou city (No. ZZ2017J29).

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci.Wireless sensor networks: a survey, *Computer Networks*, vol. 38, pp.393-422 ,2002.

[2] F. C. Chang and Hsiang-Cheh Huang, A Survey on Intelligent Sensor Network and Its Applications, *Journal of Network Intelligence*, vol. 1, no. 1, pp. 1-15, Feb 2016.

[3] Y. -J. Chen, Gwo-Jiun Horng, and Sheng-Tzong Cheng, A Distributed Cross-Layer Compromise Detection Mechanism for Wireless Sensor Networks, *Journal of Network Intelligence*, vol. 2, no. 1, pp. 147-161, Feb 2017.

[4] M. M. E. A. Mahmoud and X. Shen, A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks, *IEEE Trans. Parallel Distrib. Syst.*,vol. 23, no. 10, pp. 1805-1818, 2012.

[5] Kun Xie, Xueping Ning, Xin Wang, et al. An efficient privacy-preserving compressive data gathering scheme in WSNs. *Information Sciences*, vol. 390, pp. 82-94, 2017.

[6] D. Boneh, C. Gentry, B. Lynn, H. Shacham, Aggregate and verifiably encrypted signatures from bilinear maps, *in Proceedings of Eurocrypt 2003*, ed. by E. Biham. LNCS, vol. 2656, pp. 416-432,2003.

[7] Lysyanskaya, S. Micali, L. Reyzin, H. Shacham, Sequential aggregate signatures from trapdoor permutations, *in Proceedings of Eurocrypt 2004*, ed. by C. Cachin, J. Camenisch. vol. 3027, pp.74-90, 2004.

[8] K. Lee, D. H. Lee, M. Yung, Sequential Aggregate Signatures with Short Public Keys: Design, Analysis and Implementation Studies, *Public Key Cryptography -PKC 2013. Lecture Notes in Computer Science*, vol. 7778, pp. 423-442, 2013.

[9] Z. Shao, Enhanced aggregate signatures from pairings,*in Proce. CISC 2005, LNCS 3822, Springer Veralg*, pp.140-149, 2005.

[10] J. Xu, Z. Zhang and D. Feng, ID-Based Aggregate Signatures from Bilinear Pairings,*in Proc.4th International Conference, CANS 2005*, LNCSvol. 3810, Springer-Veralg, pp. 110-119, 2005.

[11] G. Gentry and Z. Ramzan, Identity-based aggregate signatures, *in Proc. Public Key Cryptography*, LNCSvol. 3958, pp. 257-273, 2006.

[12] J. Herranz, Deterministic identity-based signatures for partial aggregation, *The Computer Journal*, vol. 49, no. 3, pp. 322-330, 2006.

[13] J. Li, K. Kim, F. Zhang and X. Chen, Aggregate proxy signature and verifiably encrypted proxy signature, *in Proc. the International Conference on Provable Security*, LNCS 4784, Springer-Veralg, pp. 208-217, 2007.

[14] Z. Gong, Y. Long, X. Hong and K. Chen, Two certificateless aggregate signatures from bilinear maps. *in Proc. SNPD 2007, IEEE Press*, Qingdao, China, pp. 188-193, 2007.

[15] J. K. Liu , Baek J, Zhou J. Certificate-based sequential aggregate signature. *ACM Conference on Wireless Network Security, WISEC 2009*, Zurich, Switzerland, March. DBLP, pp.21-28,2009.

[16] Y. Wen, J. Ma and H. Huang, An Aggregate Signature Scheme with Specified Verifier, *Chinese Journal of Electronics*, vol. 20, no. 2, pp.333-336, 2011.

[17] S. S. D. Selvi, S. S. Vivek, J. Shriram et al., Identity based partial aggregate signature scheme without pairing,*in Proc. 35th IEEE. Sarnoff Symposium (SARNOFF)*, pp.1-6, 2012.

[18] L. Zhang, B. Qin, Q. Wu, F. Zhang, Efficient many-to-one authentication with certificateless aggregate signatures, *Comput. Netw*, vol. 54, no. 14 ,pp. 2482-2491, 2010.

[19] H. Xiong, Z. Guan, Z. Chen and F. Li, An efficient certificateless aggregate signature with constant pairing computations, *Information Sciences*,vol. 219, no. 10, pp. 225-235, 2013.

[20] F. Zhang, L. Shen and G. Wu, Notes on the security of certificateless aggregate signature schemes, *Information Sciences*,vol. 287, pp. 32-37, 2014.

[21] D. He, M. Tian and J. Chen, Insecurity of an efficient certificateless aggregate signature with constant pairing computations, *Information Sciences*,vol. 268, pp. 458-462, 2014.

[22] Jian-Neng Chen, Qun-Shan Chen and Fu-Min Zou. Certificate-Based Aggregate Signature Scheme without Bilinear Pairings. *Journal of Information Hiding and Multimedia Signal Processing*,vol. 7, no. 6, pp. 1330-1336, 2016.

[23] P. Vasudeva Reddy, P. V. S. S. N. Gopal, Identity-based key-insulated aggregate signature scheme, *Computer and Information Sciences*, no. 29, pp.303-310, 2017.

[24] J. Kim, W. Yang, S. Lee, et al., Mediated aggregate signature schemes in wireless sensor networks. *IEEE International Conference on Sensing, Communication, and NETWORKING - Workshops.* IEEE, pp.1-6,2015.

[25] S. J. Horng, S. F. Tzeng, P. H. Huang, et al., An efficient certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks. *Information Sciences*,vol. 317, pp.48-66, 2015.

[26] L. Shen, J. Ma, X. Liu, et al., A Secure and Efficient ID-Based Aggregate Signature Scheme for Wireless Sensor Networks. *IEEE Internet of Things Journal*, pp.1-9, 2016.

[27] L. Shen, J. Ma, et al., A Provably Secure Aggregate Signature Scheme for Healthcare Wireless Sensor Networks. *Journal of Medical Systems*,vol. 40, no. 11, pp.244-253, 2016.

[28] M. Bellare and P. Rogaway, Random oracles are practical: A paradigm for designing efficient protocols, *in Proc. 1st ACM Conf. Comput. Commun. Secur.*, pp.62-73, 1993.

[29] R. Canetti, O. Goldreich, and S. Halevi, The random oracle methodology, revisited,*J. ACM*,vol. 51, no. 4, pp.557-594, 2004.

[30] T. Y. Wu, Y. M. Tseng, S. S. Huang, Y. C. Lai, Non-repudiable Provable Data Possession Scheme with Designated Verifier in Cloud Storage Systems, *IEEE Access*,vol. 5, pp. 19333-19341, Oct 2017.

[31] D. Boneh, Franklin M, Identity-based encryption from the Weil pairing. *Advances in cryptology. CRYPTO 2001*, LNCS,vol. 2139. Springer,Heidelberg, pp. 213-229, 2001.

[32] L. Chen, Cheng Z, Smart NP. Identity-based key agreement protocols from pairings. *International Journal Information Security*,vol. 6, no. 4:213-241,2007.

[33] T. Y. Wu, Y. M. Tseng, An ID-based mutual authentication and key exchange protocol for low-power mobile devices. *Computer Journal*,53(7):1062-1070,2010.

[34] T. Y. Wu, T. T. Tsai, Y. M. Tseng, Efficient searchable ID-based encryption with a designated server, *Annals of telecommunications*,vol. 69(7-8), pp.391-402, 2014.

[35] T. Y. Wu, Y. M. Tseng, T. T. Tsai, A revocable ID-based authenticated group key exchange protocol with resistant to malicious participants, *Computer Networks*, vol. 56, no. 12, pp.2994-3006, 2012.

[36] J. Li, X. Huang, Y. Zhang, L. Xu, An efficient short certificate-based signature scheme, *The Journal of Systems and Software*,vol. 85,pp.314-322, 2012.

[37] C. T. Li, T. Y. Wu, C. L. Chen, C. C. Lee, C. M. Chen, An Efficient User Authentication and User Anonymity Scheme with Provably Security for IoT-based Medical Care System, *Sensors*, 2017, 17, 1482; 18 pages, doi:10.3390/s17071482, June 2017.

[38] T.Y. Wu, Y.M. Tseng, Publicly verifiable multi-secret sharing scheme from bilinear pairings, *IET Information Security*,vol. 7, no. 3, pp.239-246, 2013.

[39] T.Y. Wu, C.M. Chen, K.H. Wang, J.S. Pan, W. Zheng, S.C. Chu, J. F. Roddick, Security Analysis of Rhee et al.'s Public Encryption with Keyword Search Schemes: A Review, *Journal of Network Intelligence*,vol. 3, no. 1, pp.16-25, Feb 2018.

[40] D. Point cheval and J. Stern, Security arguments for digital signatures and blind signatures, *Journal of Cryptology*,vol. 13, no. 3, pp.361-396, 2000.