Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **6**, Number **1**, February **2021**

An Efficient Aggregate Signature Scheme for Healthcare Wireless Sensor Networks

Jian-Neng Chen

Minnan Normal University
Zhangzhou,363000, China
cjn610@163.com

Yu-Ping Zhou

Key Laboratory of Data Science and Intelligence Application
Zhangzhou,363000, China
yp_ zhou@mnnu.edu.cn

Zhen-Jie Huang

Lab of Granular Computing
Zhangzhou,363000, China
zhj_ huang@163.com

Tsu-Yang Wu

College of Computer Science and Engineering
Shandong University of Science and Technology
Qingdao 266590, China
wutsuyang@gmail.com

Fu-Min Zou

Fujian Key Laboratory of Automotive Electronics and Electric Drive
Fujian Provincial Key Laboratory of Big Data Mining and Applications
Fujian University of Technology
National Demonstration Center for Experimental Electronic Information and Electrical Technology Education
Fujian University of Technology, 350008, Fuzhou, China
fmzou@fjut.edu.cn

Raylin Tso

Department of Computer Science
National Chengchi University,
Taipei, Taiwan
raylin@cs.nccu.edu.tw

Corresponding Author: Tsu-Yang Wu
(wutsuyang@gmail.com)

Received August 2020; revised October 2020

Abstract. *In healthcare wireless sensor networks, there are a large number of sensors, which need to transmit a lot of information in real time. The aggregate signature scheme combines a great deal of signatures signed by different signers on different messages into one short signature, which greatly improves the efficiency of data transmission. In this paper, we present a scheme of certificate based aggregate signature, and prove its security in the random oracle model. The performance analysis shows that the scheme is efficient in sign and verify phase, and suitable for healthcare wireless sensor networks.*
Keywords: Healthcare Wireless Sensor Networks, Certificate-based, Aggregate signature

1. **Introduction.** Wireless sensor network (WSN) has turned into an emerging technology, which is possible to acquire dynamic data from the environment [1, 2]. In healthcare sensor networks, dynamic data obtained from the sensors provides great convenience for doctors and patients. Sensor based technology has invented many medical micro devices. The devices can be divided into two types: implanted devices and wearable devices [3]. The implanted devices is injected in human body. As long as patients move along with them, they can send the basic physiological data to the medical server in real time. Wearable devices are attached to the surface of human body. It's convenience for professional medical personnel to perceive the patient's physical condition. However, modified data may be the cause of serious medical accidents of patients. Data privacy has become an important issue. Digital signature is a technique of public key cryptography that is widely accepted to provide integrity, authenticity and unforgeability of messages [4–7].

In the year 2003, The notion of aggregate signatures was introduced by Boneh et al. [8] on the European cryptographic international conference. A sequential aggregate signature scheme was proposed by Lysyanskaya et al. [9] in the same year. In the year 2009, the concept of certificate-based aggregate signature was present by Liu et al. [10] who constructed the first certificate-based sequential aggregate signature scheme. Unfortunately, the efficiency of the scheme needs to be improved. After pioneer work, many aggregate signature schemes [11–13] have been proposed by the researchers. In an aggregate signature, any user could map n different signatures signed by different n signers on n different messages to a single signature [14]. Verifier just needs to verifies the final aggregate signature. The verification of n different signatures can be completed at one time. The storage space and communication overhead the are greatly reduced. This process reduces the bandwidth and calculation cost. Sensors are kind of micro devices which have less bandwidth and limited storage power, therefore aggregate signature is suitable for wireless sensor networks. Many data aggregation technology for WSNs have been present by the different researchers based on bilinear pairing [14–22]. In 2015, a certificateless aggregate signature for vehicular sensor networks was proposed by Horng et al. [16], which can realize conditional privacy protection. The efficiency was not considered carefully as the roadside units aggregated signatures in the scheme. Recently, some researches focus on applying signature technology to the field of medical, health care, 5G mobile communication networks, internet of things and so on [23–33].

Although aggregate signature is a good choice for HWSNs, the time of generating a signature often increases with the number of signatures, and the number of bilinear pairings. In this paper, we proposed a certificate-based aggregate signature scheme with constant bilinear pairings for HWSNs. In the HWSNs, there are a large number of sensors which sign the signatures but have limited resource. The scheme we proposed needs less calculation time in the phase of sign and verify, and is suitable for healthcare wireless sensor networks.

The rest of this paper is organized as follows. The security model and notions of certificate-based aggregate signature scheme is introduced in section 2. section 3 describes the proposed CBAS scheme for HWSNs. section 4 presents security proof of our scheme. The performance analysis in terms of calculation cost are given in section 5, and section 6 draws the conclusions.

2. Model and security notions for CBAS.

2.1. Definition of CBAS.

We define a certificate-based aggregate signature scheme by seven algorithms.

Setup: This algorithm is executed by the certificate authority (CA). It inputs a security parameter 1^k and returns the system parameters *param*, CA's public key and private key.

KeyGen: In this algorithm, users take *param* as inputs. It returns a key pair(SK_{IDi}, PK_{IDi}) as users' private and public key.

CertGen: In this algorithm, certificate authority takes *param*, CA's private key, the user's identity ID_i and public key PK_{IDi} as inputs. It then outputs a certificate $Cert_{IDi}$ for the user.

Sign: In this algorithm, signers take their private key SK_{IDi}, *param*, $Cert_{IDi}$, and message m_i as inputs. Then outputs a signature.

Verify: In this algorithm, verifier takes user's public key,*param*, CA's public key and a signature as inputs. If the verification process succeeds, it outputs 1, otherwise,0.

Aggregate: In this algorithm, aggregator takes *param*, n signatures$(\sigma_1, \sigma_2, ..., \sigma_n)$, user's identity ID_i and public keys as inputs. Then outputs an aggregate signature.

AggVerify: In this algorithm, verifier takes *param*, messages m_i and an aggregate signature as inputs. If the verification process succeeds, it outputs 1, otherwise,0.

2.2. Security model for single signature.

Now, security model for single signature is defined according to document [34]. In our scheme, there exists two types of adversaries called A_I and A_{II}.

In the Game 1, the security against public key substitution attack is defined between adversary A_I and challenger C, which is described as follow.

Initialization.C runs setup algorithm for *param* and CA's public key and private key. After that, C maintains five lists L_K,L_C, L_{H1}, L_{H2} and L_S, and returns *param* and CA's public key PK_{CA} to the adversary A_I.

Queries. A_I makes queries to C adaptively as follow:

(a) **KeyGen query:** When A_I makes a query with identity ID_i, C returns PK_{IDi} if the list L_K contains ID_i. Otherwise, C runs the **KeyGen** algorithm for (PK_{IDi},SK_{IDi}). Then, C returns PK_{IDi} and adds (ID_i, SK_{IDi}, PK_{IDi}) to L_K.

(b) **CertGen query:** When A_I makes a query with identity ID_i, C returns $Cert_{IDi}$, if L_C contains ID_i. Otherwise, C runs the **CertGen** algorithm for ID_i's certificate $Cert_{IDi}$. Finally, C returns $Cert_{IDi}$ and adds (ID_i, PK_{IDi}, $Cert_{IDi}$) to L_C.

(c) **Hash query:** When A_I makes this query, C picks a random value and returns to A_I.

(d) **Corrupt query:** When A_I makes this query on ID_i, C returns SK_{IDi}, if L_K contains ID_i. Otherwise, C runs **KeyGen** algorithm for (SK_{IDi}, PK_{IDi}). Then, C returns SK_{IDi}, and adds (ID_i, SK_{IDi},PK_{IDi}) to L_K.

(e) **Sign query:** When A_I makes this query on (m_i,ID_i,PK_{IDi}), C returns (m_i, σ_i), if L_S contains (m_i, ID_i,PK_{IDi}). Otherwise, C runs **Sign** algorithm for a signature σ_i. After that, C returns σ_i and adds (m_i, PK_{IDi}, ID_i, σ_i) to L_S.

(f) **Replacing public key request:** A_I picks a value and replace user's public key PK_{IDi}.

Forgery: A_I outputs a signature tuple (ID_i^*,m_i^*,σ_i^*,PK_{IDi}^*). A_I wins Game 1 if

(i) **Verify**(m_i^*,σ_i^*,PK_{IDi}^*, PK_{CA})=1.
(ii) ID_i^* has not been queried to **CertGen query**.
(iii) (ID_i^*,m_i^*,PK_{IDi}^*) has never been issued to **Sign query**.

Definition 2.1. *A single certificate-based signature scheme is secure against a public key substitution attack, if the success probability of* A_I *wins Game 1* $\mathrm{Adv}^{\mathrm{A}_I}_{\mathrm{Game}_1}(\mathrm{t})$ *is negligible.*

In Game 2, the security of single signature scheme against the certifier is described between adversary A_{II} and challenger C.

Initialization. C runs **setup** algorithm for *param* and CA's private key and public key. After that, C maintains five lists L_K,L_C,L_{H1},L_{H2} and L_S, and returns (*param*, PK_{CA}, SK_{CA}) to the adversary A_{II}.

Queries. A_{II} makes queries to C adaptively as follow:

(a) **KeyGen query:** When A_{II} makes a query with identity ID_i, C returns PK_{IDi} if the list L_K contains ID_i. Otherwise, C runs **KeyGen** algorithm for (PK_{IDi}, SK_{IDi}). Then, C returns PK_{IDi} and adds (ID_i,SK_{IDi},PK_{IDi}) to L_K.
(b) **Hash query:** When A_{II} makes this query, C picks a random value and returns to A_{II}.
(c) **Corrupt query:** When A_{II} makes this query on ID_i, C returns SK_{IDi}, if L_K contains ID_i. Otherwise, C runs **KeyGen** algorithm for (SK_{IDi},PK_{IDi}). Then, C returns SK_{IDi} and adds (ID_i, SK_{IDi}, PK_{IDi}) to L_K.
(d) **Sign query:** When A_{II} makes this query on (m_i,ID_i,PK_{IDi}), C returns (m_i, σ_i), if L_S contains (m_i,ID_i,PK_{IDi}). Otherwise, C runs **Sign** algorithm for a signature σ_i. After that, C returns σ_i and adds (m_i,PK_{IDi},ID_i,σ_i) to L_S.

Forgery: A_{II} outputs a tuple (ID_i^*,m_i^*,σ_i^*,PK_{IDi}^*). A_{II} wins Game 2 if:

(i) **Verify**(ID_i^*,m_i^*,σ_i^*,PK_{IDi}^*)=1.
(ii) ID_i^* has not been queried to **Key query** and the **Corrupt query**.
(iii) (PK_{IDi}^*,m_i^*,ID_i^*) has never been issued to **Sign query**.

Definition 2.2. *A single certificate-based signature scheme is secure against the certifier, if the success probability of* A_{II} *wins Game 2* $\mathrm{Adv}^{\mathrm{A}_{II}}_{\mathrm{Game}_2}(t)$ *is negligible.*

2.3. **Security model for aggregate signature.** In Game 3, the security against public key substitution attack is described between adversary A_I and challenger C, in which A_I tries to forge an aggregate signature under PK_1, PK_2, ..., PK_n without the corresponding certificate $Cert^*$.

Initialization. C runs **setup** algorithm for *param* and CA's public key and private key. After that, C maintains six lists L_K,L_C,L_{H1},L_{H2},L_S and L_{AS}. Then, A_I is provided *param* and PK_1. We assume that PK_1 is the target public key without loss generality.

Queries. A_I makes queries to C adaptively. When received **Key query**, **Corrupt query** and **Certificate query**, C responds the same as in game 1.

(a) **Hash query:** When A_I makes this query, C picks a random value and returns to A_I.
(b) **Sign query:** When A_I makes this query on (m_i,ID_i,PK_{IDi}), C returns (m_i,σ_i), if L_S contains (m_i,ID_i,PK_{IDi},σ_i). Otherwise, C runs **Sign** algorithm for a signature σ_i. Then, C returns σ_i and adds (m_i,ID_i,PK_{IDi},σ_i) to L_S.
(c) **AggSign query:** When A_I makes a query on (M,ID,PK) = [(m_1,ID_1,PK_1),...,(m_i,ID_i, PK_i),...,(m_n,ID_n,PK_n)], for $1 \leq i \leq n$, C searches the list L_{AS} first. If (M,ID,PK, σ_{Ai}) has existed on L_{AS}, C returns σ_{Ai}. Otherwise,
   (i) If PK_{IDi}=PK_1, for $1 \leq i \leq n$, C aborts.

(ii) If $PK_{IDi} \neq PK_1$, for $1 \leq i \leq n$, C searches the list L_S for $(m_i,ID_i,PK_{IDi},\sigma_i)$. If it doesn't exits on the list, C runs the **Sign** algorithm for a signatures σ_i and adds $(m_i,ID_i,PK_{IDi},\sigma_i)$ to L_S. Then, C runs **Aggregate** algorithm to generate an aggregate signature σ_{Ai}. Finally, C returns σ_{Ai} to A_I and adds (M,ID,PK,σ_{Ai}) to L_{AS}.

(d) **Replacing public key request.** A_I picks a value and replace user's public key PK_{IDi}.

Forgery: Finally, A_I outputs k distinct messages $(m_1,m_2,...,m_k)$, *k-1* public keys $(PK_2,...,PK_k)$, and an aggregate signature σ^* under $(PK_1,PK_2,...,PK_k)$, where $k \leq n$. A_I wins Game 3 if

(i) **AggVerify**$(m_i,ID_i,PK^*_{IDi},\sigma^*)$=1.
(ii) $(PK_1,PK_2,...,PK_k)$ must be included in the set of PK_{IDi}.
(iii) (m_i,ID_1,PK_1) has never be asked for **Sign query**.

Definition 2.3. *A certificate-based aggregate signature scheme is secure against the existential forgery, if the success probability of* A_I *wins Game 3* $\mathrm{Adv}^{\mathrm{A}_I}_{\mathrm{Game}_3}(t)$ *is negligible.*

In Game 4, the security of aggregate signature against the certifier is defined between adversary A_{II} and challenger C, which is described as follow.

Initialization. C runs **setup** algorithm for *param* and CA's public key PK_C and private key sk_C. After that, C maintains six lists $L_K,L_C,L_{H1},L_{H2},L_S$ and L_{AS}. Then, A_{II} is provided *param* and PK_1. We assume that PK_1 is the target public key without loss generality.

Queries. A_{II} makes queries to C adaptively. When received **Corrupt query**, **Key query**, **Hash query** and **Sign query**, C responds the same as in game 2.

AggSign Queries: When A_{II} makes this query on $(M,ID,PK) = [(m_1,ID_1,PK_1),...,(m_i, ID_i,PK_i),...,(m_n,ID_n,PK_n)]$, for $1 \leq i \leq n$, C searches L_{AS} first. If (M,ID,PK,σ_{Ai}) has existed on L_{AS}, C returns σ_{Ai}. Otherwise,

(i) If $PK_{IDi}=PK^*_{IDi}$, C aborts.
(ii) If $PK_{IDi} \neq PK^*_{IDi}$, C searches the list L_S for $(m_i,ID_i,PK_{IDi},\sigma_i)$. If it isn't on the list, C runs the **Sign** algorithm for a signatures σ_i and adds $(m_i,ID_i,PK_{IDi},\sigma_i)$ to L_S. Then, C runs **Aggregate** algorithm for an aggregate signature σ_{Ai}. Finally, C returns σ_{Ai} and adds (M,ID,PK,σ_{Ai}) to L_{AS}.

Forgery: Finally, A_{II} outputs an aggregate signature σ^* under public keys$(PK_1,PK_2,...,PK_{IDi},...,PK_n)$, A_{II} wins Game 4 if

(i) **AggVerify**$(m_i,ID_i,PK_{IDi},\sigma^*)$=1.
(ii) L_K contains PK_{IDi}, where $1 \leq i \leq n$.
(iii) (m_i,ID_1,PK_1) has never be asked for **Sign query**.

Definition 2.4. *A certificate-based aggregate signature scheme is secure against the certifier, if the success probability of* A_{II} *wins Game 4* $\mathrm{Adv}^{\mathrm{A}_{II}}_{\mathrm{Game}_4}(t)$ *is negligible.*

3. The CBAS scheme for HWSNs.

3.1. **system model.** Our system model for HWSNs(figure 1) has four components: sensors, aggregators, medical server, and healthcare professionals. Sensors are implanted in or worn on patients to get the data signals they need. The patient's health message is then signed and transferred to the aggregator by sensor modules. Then aggregators submits the signature to the medical server. Healthcare professionals can access the server at any time to obtain the required information. Healthcare professionals then provide patients with treatment advice, medication, or specific medical treatment.

Sensors are tiny devices with limited computing, storage, range and communication capabilities. There are many kinds of sensors, such as temperature sensor, pressure sensor, heart rate sensor, motion sensor, tactile sensor and so on. They can collect basic vital signs of patients, such as body temperature, heart rate, blood pressure, etc. Each sensor has a unique ID, public key and private key. The patient's vital signs are signed by the sensors with the private key and transmitted wirelessly to the aggregator.

Aggregator has certain computing power and communication power. Each care district can contain many sensors, but only one aggregator. The aggregator collects a single signature within the care district and verifies its validity. Valid single signatures are then aggregated into a signature and transferred to the medical server.

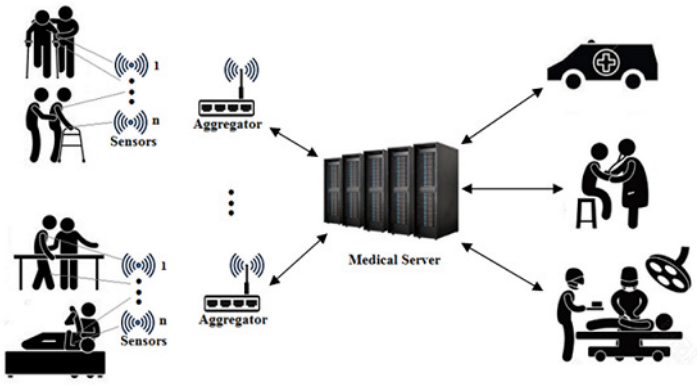

FIGURE 1. System Model

Medical server has large storage space and strong computing power. Therefore, most of the calculation and storage is done on the server. For example, when the system is initialized, the server will get the public key, ID, etc. of all the sensors, and generate certificates. During the signing phase, it collects the aggregate signature and rejects the invalid signature and corresponding message. Vital signs, medical reports, medical advice, prescriptions, medications used in treatment, treatment results, etc. are all stored on the medical server.

Healthcare professionals analyze the patient information provided by the server and then make the appropriate treatment. To ensure the safety of treatment, healthcare professionals must be registered and authorized.

3.2. **Prilimilary.** G_1 and G_2 are two additive cyclic groups of order q. G_1 is a subgroup of abelian group $E(FP)$ and G_2 is a subgroup of finite field FP. P and Q belong to G_1. The properties of bilinear pair e: $G_1 \times G_1 \to G_2$ is described below:

(i) Bilinearity: $e(aP, bP)=e(P,Q)^{ab}$. for $a,b \in Z_q^*$.
(ii) Non-degeneracy: If identity $1_{G1} \in G_1$, then $(1_{G1},1_{G1})$ is also an identity of G_2.
(iii) Computablility: There are efficient algorithms to compute $e(P, Q)$.

3.3. **The proposed CBAS scheme. Setup:** Medical server runs **setup** algorithm for parameters and key pairs.

(a) Picking two different generator P and Q in G_1.
(b) Selecting medical server's private key $s \in_R Z_q^*$, and computing its public key $PK_{MS}= s \cdot P$.
(c) Two hash functions are chosen, $H_1 : \{0,1\}^* \to Z_q^*$, $H_2 : \{0,1\}^* \to G_1$.

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

KeyGen: Each sensor's private/public key pair is $(x_i,\ PK_{IDi})$. Public key $PK_{IDi}= x_iP$, where x_i is a random number $x_i \in_R Z_q^*$, and P is a generator in G_1.

CertGen: Each sensor submits its identity ID_i and public key PK_{IDi} to medical server. Then medical server computes $Q_i=H_1(PK_{IDi},ID_i)$ and returns a certificate $Cert_{IDi}=sQ_i$.

Sign: To send a message m_i, sensors choose a random number $a_i \in_R Z_q^*$,computes $U_i=a_iP$, $h_i=H_2(m_i, PK_{IDi}, U_i)$, and $T_i= x_iQ+(a_iQ+Cert_{IDi})h_i$. Finally, $\sigma_i=(U_i, T_i)$ is the signature of m_i.

Verify: To verify a signature $\sigma_i=(U_i, T_i)$, aggregator computes $Q_i=H_1(PK_{IDi},ID_i)$, $h_i=H_2(m_i, PK_{IDi}, U_i)$, and outputs 1 if the following equation holds.

$$e(T_i, P) = e(PK_{IDi} + h_iU_i, Q)e(h_iQ_i, PK_{MS}) \tag{1}$$

Aggregate: When receives n signature tuples $(m_i, PK_{IDi}, ID_i, \sigma_i)$ from sensors, aggregator computes $T = \sum_{i=1}^{n} T_i$. Then, aggregate signature $\sigma_i=(T, U_i)$ is submit to medical server.

AggVerify: The aggregator verifies an aggregate signature σ_I by computing $h_i=H_2(m_i, ID_i, U_i), V = \sum_{i=1}^{n} h_iQ_i$ and $H = \sum_{i=1}^{n}(PK_{IDi} + h_iU_i)$. If the following equation holds, he outputs 1, otherwise, 0.

$$e(T, P) = e(H, Q)e(V, PK_{MS}) \tag{2}$$

3.4. **correctness of aggregate verification.** The correctness analysis of aggregate verification in our scheme is as follows.

$$\begin{aligned}
e(T, P) &= \prod_{i=1}^{n} e(T_i, P) && (3)\\
&= \prod_{i=1}^{n} e(x_iQ + (a_iQ + Cert_{IDi})h_i, P) && (4)\\
&= \prod_{i=1}^{n} e(x_iP, Q)e(h_ia_iP, Q)e(h_iCert_{IDi}, P) && (5)\\
&= \prod_{i=1}^{n} e(PK_{IDi}, Q)e(h_iU_i, Q)e(h_iPK_{MS}, Q_i) && (6)\\
&= \prod_{i=1}^{n} e(PK_{IDi} + h_iU_i, Q)e(h_iQ_i, PK_{MS}) && (7)\\
&= e[\sum_{i=1}^{n}(PK_{IDi} + h_iU_i), Q]e(\sum_{i=1}^{n} h_iQ_i, PK_{SM}) && (8)\\
&= e(H, Q)e(V, PK_{MS}) && (9)
\end{aligned}$$

4. Scheme Analysis.

4.1. **Computational Diffie-Hellman Problem(CDHP).** Input P, aP, bP and unknown a, $b \in Z_q^*$, to output abP. In this paper, we assume that there is no algorithm can solve the Computational Diffie-Hellman Problem with an not negligible success probability in polynomial time. For details about computational Diffie-Hellman assumption, readers can refer to [35, 36] for a full description.

4.2. **Unforgeability of single signature.** The security of single signature scheme based on the hardness of CDHP is shown as follow.

Theorem 4.1. *If there exists an adversary A_I with an non-negligible success probability $Adv^{A_I}_{Game_1}$(t) in forging a valid single signature, the Computational Diffie-Hellman Problem would be solved with a success probability which is not negligible.*

Proof: C interacts with A_I as follow:

Initialization. C executes **Setup** algorithm for *param* =(G_1,G_2,e,q,H_1,H_2,P,Q) and sets medical server's public key PK_{MS}=aP. Meanwhile, C initializes lists L_K, L_C, L_{H1}, L_{H2} and L_S as empty and sends (*param*, PK_{MS}) to A_I.

Queries. A_I make follow queries to C:

(a) **Key query:** When A_I makes this query with sensor identity ID_i, C returns PK_{IDi} if the list L_K contains ID_i. Otherwise, C runs **KeyGen** algorithm for (SK_{IDi}, PK_{IDi})=(x_i, x_iP). Then, C returns PK_{IDi} and adds (ID_i, SK_{IDi}, PK_{IDi}) to L_K.

(b) **Certificate query:** When A_I makes a query with sensor identity ID_i, C returns $Cert_{IDi}$,if L_C contains ID_i, otherwise, he searches L_K with sensor identity ID_i. C aborts if the corresponding public key has been replaced. Otherwise, C searches(ID_i, PK_{IDi}) in L_{H1}. C adds (ID_i,PK_{IDi}, $coin_i$, t_i, Q_i) to the list L_{H1} as in **H1 querys**, if L_{H1} does not contains (ID_i, PK_{IDi}). After that, C picks a random number $coin_i \in \{0,1\}$, for δ=1/(q_c+q_s) and Pr[$coin_i$=1]=δ, where q_c is the total number A_I asks for **Certificate query**, and q_s is the total number A_I asks for **sign query**. Then C performs as fallows.
   (i) If $coin_i$=1, C aborts.
   (ii) If $coin_i$=0, C calculates $Cert_{IDi}$=t_iPK_{CA}, then returns $Cert_{IDi}$ to A_I and adds (ID_i,PK_{IDi},$Cert_{IDi}$) to L_C.

(c) **H1 query:** When A_I makes this query with (PK_{IDi},ID_i), C returns Q_i, if (PK_{IDi},ID_i,Q_i) exists on L_{H1}, otherwise,
   (i) If $coin_i$=1, C sets Q_i=bP and adds (ID_i,PK_{IDi},$coin_i$, Q_i) to L_{H1}. Then C returns Q_i to A_I.
   (ii) If $coin_i$=0, C picks a number $q_i \in_R Z_q^*$, and calculates Q_i=q_iP. Then C adds (ID_i,PK_{IDi},$coin_i$, q_i, Q_i) to the list L_{H1} and returns Q_i to A_I.

(d) **H2 query:** When A_I makes this query with (m_i,PK_{IDi},U_i), C returns h_i, if (m_i,PK_{IDi},U_i,h_i) exists on L_{H2}, otherwise, C returns A_I a random value $h_i \in_R Z_q^*$, and adds (m_i,PK_{IDi},U_i,h_i) to L_{H2}.

(e) **Corrupt query:** When A_I makes this query with sensor identity ID_i, C returns sensor's private key SK_{IDi}, if L_K contains PK_{IDi}. Otherwise, C runs the **KeyGen** algorithm for the sensor's private key SK_{IDi} and public key PK_{IDi}. Then, C returns SK_{IDi} to A_I and adds (PK_{IDi},SK_{IDi},ID_i) to L_K. Note that C returns $\perp$, if the user's public key has been replaced.

(f) **Sign query:** When A_I makes this query on (m_i,ID_i,PK_{IDi}), C returns σ_i, if L_S contains (m_i,ID_i,PK_{IDi},σ_i). Otherwise, C searches L_K for the sensor's private key SK_{IDi}=x_i. Then, C finds the tuple (ID_i,PK_{IDi},$coin_i$,q_i,Q_i) on the list L_{H1}.
   (i) If $coin_i$=1, C aborts.
   (ii) If $coin_i$=0, C calculates $Cert_{IDi}$=q_iPK_{MS}, and then searches list L_{H2} to obtain (m_i,PK_{IDi},U_i,h_i). If it does not exist, C will make a **H2 query**. After that, C runs the **Sign** algorithm with (m_i,ID_i,PK_{IDi},$Cert_{IDi}$) to generate a signature U_i=a_iP, T_i=x_iQ+(a_iQ+$Cert_{IDi}$)h_i. Finally, C sends σ_i =(U_i, T_i) to A_I and adds (m_i,ID_i,PK_{IDi},σ_i) to L_S.

(g) **Replacing public key request** When A_I makes this query with (ID_i,PK'_{IDi}), C replaces (ID_i,SK_{IDi},PK_{IDi}) with (ID_i, $\perp$, PK'_{IDi}) on L_K.

Forgery: Finally, A_I outputs a signature tuple $(m_i^*, PK_{IDi}^*, ID_i^*, U_i^*, T_i^*)$

Assume that an adversary A_I outputs a valid signature σ_i^*. Then, with a non-negligible success probability, applying Forking Lemma [37], another signature $\sigma_i^{'}$=$(U_i^{'}, T_i^{'})$ can be forged under different oracle and in the same random tape. Then, we have

$T_i^* = x_i Q + (a_i Q + Cert_{IDi}) h_i$ and $T_i^{'} = x_i Q + (a_i Q + Cert_{IDi}) h_i^{'}$.It implies that $(T_i^*\text{-}T_i^{'}) = (a_i Q + Cert_{IDi})(h_i - h_i^{'})$. Note that $PK_{MS} = aP$ and $Q_i = bP$, $Cert_{IDi} = abP = (T_i^*\text{-}T_i^{'})/(h_i\text{-}h_i^{'})\text{-}a_i Q$.

Therefore, if A_I forged a valid single signature with success probability $Adv_{Game_1}^{A_I}(t)$, C solves the CDH problem with success probability $\varepsilon^{'} \geq (1-\delta)^{q_c}\delta(1-\delta)^{q_s} Adv_{Game_1}^{A_I}(t) \geq \frac{1}{e(q_c+q_s)} Adv_{Game_1}^{A_I}(t)$, for q_s is the total number of **sign query**, q_c is the total number of **Certificate query** and e is the base of natural logarithm.

Theorem 4.2. *If there exists an adversary* A_{II} *with a non-negligible success probability* $\mathrm{Adv}_{\mathrm{Game}_2}^{\mathrm{A}_{II}}(\mathrm{t})$ *in forging a valid single signature, the Computational Diffie-Hellman Problem would be solved with a non-negligible success probability.*

Proof: A_{II} interacts with C as follow:

Initialization. C runs **Setup** algorithm for system parameters $param$=$(G_1, G_2, H_1, H_2, e, q, P, Q)$ and sets Q=bP, PK_{MS}=sP, where $s \in_R Z_q^*$. After that, C initializes five empty lists L_K,L_C,L_{H1},L_{H2} and L_S. Then, C sends $param$ and medical server's private key s to A_{II}.

Queries A_{II} makes queries to C as follow:

(a) **Key query:** When A_{II} makes a query with a sensor identity ID_i,
  (i) If $ID_i \neq ID_i^*$, C searches the list L_K. If ID_i has existed on L_K, C returns (SK_{IDi}, PK_{IDi}). Otherwise, C runs **KeyGen** algorithm for ID_i's private and public keys (SK_{IDi}, PK_{IDi})=$(x_i, x_i P)$. Then, C sends (SK_{IDi}, PK_{IDi}) to A_{II} and adds $(ID_i, SK_{IDi}, PK_{IDi})$ onto L_K.
  (ii) If $ID_i = ID_i^*$, C returns $PK_{IDi}^* + h_i^* U_i^* = aP$ and adds $(ID_i, \perp, aP)$ onto L_K.

(b) **H1 query:** When A_{II} makes this query with(ID_i, PK_{IDi}), C returns Q_i if (ID_i, PK_{IDi}) appears in a tuple (ID_i, PK_{IDi}, Q_i) on L_{H1}. Otherwise, C picks a number $Q_i \in_R Z_q^*$ and adds (ID_i, PK_{IDi}, Q_i) on L_{H1}. Finally C returns Q_i to A_{II}.

(c) **H2 query:** When A_{II} makes a query with (m_i, PK_{IDi}, U_i), If L_{H2} contains h_i, C outputs h_i to A_{II}, otherwise, C selects a value $coin_i \in \{0, 1\}$, for $\Pr[coin_i{=}1]{=}\delta$, where δ=$1/(q_c{+}q_s)$, q_c is the total number A_{II} makes to **Corrupt query**, and q_s is the total number A_{II} makes to **sign query**, then C returns h_i as follows.
  (i) If $coin_i$=1, C chooses a random value $h_i \in_R Z_q^*$ to A_{II}, and adds $(m_i^*, PK_{IDi}^*, U_{IDi}^*, h_{IDi}^*)$ to L_{H2}.
  (ii) If $coin_i$=0, C picks a random value $h_i \in_R Z_q^*$ to A_{II}, and adds $(m_i, PK_{IDi}, U_i, h_i)$ to L_{H2}.

(d) **Corrupt query:** When A_{II} makes this query with PK_{IDi}, C returns $SK_{IDi} = x_i$, if L_K contains(PK_{IDi}, SK_{IDi}). Otherwise, C runs **KeyGen** algorithm for it's private key SK_{IDi} and public key PK_{IDi}. If $ID_i \neq ID_i^*$, C returns SK_{IDi} to A_{II} and adds $(PK_{IDi}, SK_{IDi}, PK_{IDi})$ into L_K. Otherwise, C outputs $\perp$.

(e) **Sign query:** When A_{II} makes this query with(m_i, PK_{IDi}, ID_i), C returns σ_i, if L_S contains $(m_i, PK_{IDi}, ID_i, \sigma_i)$. Otherwise, C searches list L_K for the private key x_i.
  (i) If $PK_{IDi} = PK_{IDi}^*$, C aborts.
  (ii) If $PK_{IDi} \neq PK_{IDi}^*$, C searches list L_{H1} for Q_i, and calculates $Cert_{IDi}$=sQ_i, then searches list L_{H2} for $(m_i, PK_{IDi}, U_i, h_i)$. If it does not exist, C will make a **H2 queries**. After that, C runs **Sign** algorithm for a signature σ_i

$=x_i Q+(a_i Q+Cert_{IDi})h_i$. Finally, C outputs σ_i and adds $(m_i,ID_i,PK_{IDi},\sigma_i)$ to L_S.

Forgery: After all, A_{II} outputs a signature tuple $(m_i^*,ID_i^*,PK_{IDi}^*, \sigma_i^*)$. Assume that A_{II} succeeds in forging a signature tuple $(m_i^*,ID_i^*,PK_{IDi}^*,\sigma_i^*)$. Where σ_i^* can be expressed as $\sigma_i^* = x_i^* Q+(a_i Q+Cert_{IDi})h_i^*$. Then, A_{II} can compute $abP= (x_i^*+a_i\ h_i)Q=\sigma_i^*\text{-}Cert_{IDi})h_i^*$, for $PK_{IDi}^*+h_i^* U_i^* = aP$, $Q_i = bP$. Therefore, if A_{II} forged a single signature with a non-negligible success probability $Adv_{Game_2}^{A_{II}}(t)$, then the CDHP would be solved with an advantage $\varepsilon' \geq (\frac{1}{n})^{q_k}\delta(1-\frac{1}{n})^{q_s+q_c} Adv_{Game_2}^{A_{II}}(t) \geq \frac{1}{n_{qk}(q_c+q_s)e} Adv_{Game_2}^{A_{II}}(t)$, where q_k is the total number A_{II} issues **Key queries**, q_c is the total number A_{II} issues **Corrupt queries** and q_s is the total number A_{II} issues **sign queries**, n is the total number of sensors and e is the base of natural logarithm.

4.3. **Unforgeability of aggregate signature.**

Theorem 4.3. *If there exists an adversary* A_I *with an non-negligible success probability* $\mathrm{Adv}_{\mathrm{Game_3}}^{\mathrm{A}_I}(\mathrm{t})$ *in forging a valid aggregate signature, the Computational Diffie-Hellman Problem would be solved by a challenger C with a success probability which is not negligible.*

Proof: C interacts with A_I as follow:

Initialization. C executes **Setup** algorithm for $param=(G_1,G_2,e,q,P,Q,H_1,H_2)$ and sets medical server's public key $PK_{MS}=aP$. Meanwhile, C initializes five lists L_K, L_C, L_{H1}, L_{H2} and L_S as empty and sends $(param,PK_{MS},PK_1)$ to A_I, where PK_1 is target public key.

Queries. A_I make queries to C adaptively. When received **Key query**,**Corrupt query**, **Certificate query** and **H2 query**, C responds the same as in game 1.

(a) **H1 query:** When A_I makes this query with (PK_{IDi}, ID_i), C returns Q_i, if (PK_{IDi}, ID_i, Q_i) exists on L_{H1}, otherwise, C picks a number $coin_i \in \{0,1\}$, for $\Pr[coin_i=1]=\delta$, where $\delta=\frac{1}{q_c+q_s}$, q_c is the total number A_I makes to **Certificate query** and q_s is the total number A_I makes to **sign query**.
  (i) If $coin_i=1$, C sets $Q_i=bP$ and adds $(coin_i, \perp, Q_i)$ on the list L_{H1}. Then C returns Q_i to A_I.
  (ii) If $coin_i=0$, C picks a number $q_i \in_R Z_q^*$, and calculates $Q_i=q_iP$. Then C adds $(coin_i,q_i,Q_i)$ to the list L_{H1} and returns Q_i to A_I.

(b) **Sign query:** When A_I makes this query with (m_i,PK_{IDi},ID_i), C returns σ_i, if L_S contains $(m_i,PK_{IDi},ID_i,\sigma_i)$. Otherwise,
  (i) If $coin_i=1$, C aborts.
  (ii) If $coin_i=0$, C searches list L_{H1} for (q_i, Q_i), and calculates $Cert_{IDi}=q_iPK_{MS}$, and then searches list L_{H2} for $(m_i, PK_{IDi}, U_i, h_i)$. If it does not exist, C will make a **H2 queries**. After that, C runs the **Sign** algorithm for a signature $U_i= a_iP$, $T_i = x_iQ+(a_iQ+Cert_{IDi})h_i$. Finally, C returns $\sigma_i =(U_i,T_i)$ to A_I and adds $(m_i,PK_{IDi},ID_i,\sigma_i)$ to L_S.

(c) **AggSign query:** When A_I makes this query with $(M,ID,PK)=[(m_1, ID_1, PK_1),...,(m_i, ID_i, PK_{IDi}),...,(m_n, ID_n, PK_{IDn})]$, for $1\leq i \leq n$, if L_{AS} contains (M,ID,PK,σ_{Ai}), C returns σ_{Ai}. Otherwise,
  (i) If $PK_{IDi}=PK_1$, for $1\leq i \leq n$, C aborts.
  (ii) If $PK_{IDi} \neq PK_1$, for $1\leq i \leq n$, C searches the list L_S for $(m_i,ID_i,PK_{IDi},\sigma_i)$. If it doesn't exit on the list, C runs **Sign** algorithm for a signature σ_i and adds $(m_i, ID_i, PK_{IDi}, \sigma_i)$ to L_S. Then, C runs **Aggregate** algorithm for a

aggregate signature $\sigma_{Ai} = (T = \sum_{i=1}^{n} T_i,\ U_i)$. Finally, C sends σ_{Ai} to A_I and adds (M,ID,PK,σ_{Ai}) to L_{AS}.

(d) **Replacing public key request:** When A_I makes this query with (ID_i,$PK^{'}_{IDi}$), (ID_i,SK_{IDi},PK_{IDi}) would be replaced with (ID_i, $\perp$, $PK^{'}_{IDi}$) on L_K.

Forgery: A_I outputs k messages (m_1,m_2,...,m_k), k-1 public keys (PK_2,...,PK_k), a corresponding aggregate signature σ^* under PK_1,PK_2,...,PK_k, where $k \leq n$. If A_I forged a valid aggregate signature $\sigma^* = \sum_{i=1}^{k} \sigma_i = x_1 Q + (a_1 Q + Cert_1)h_1 + \sum_{i=2}^{k} \sigma_i$ with a non-negligible success probability, then, under different oracle and in the same random tape, another aggregate signature $\sigma^{'} = \sum_{i=1}^{k} \sigma^{'}_i = x_1 Q + (a_1 Q + Cert_1)h^{'}_1 + \sum_{i=2}^{k} \sigma_i$ would be forged with a non-negligible success probability. Then, we have (σ_1- $\sigma^{'}_1$) = ($a_i Q$+$Cert_{IDi}$)(h_i-$h^{'}_i$). Note that $PK_{MS} = aP$ and $Q_i = bP$, $Cert_{IDi} = abP$ = (σ_1- $\sigma^{'}_1$)/(h_1-$h^{'}_1$)- $a_i Q$, a contradiction.

Therefore, if A_I succeed with a non-negligible success probability $Adv^{A_I}_{Game_3}(t)$, the CDHP would be solved with an advantage $\varepsilon^{'} \geq (1-\delta)^{q_c}\delta(1-\delta)^{q_s}(1-1/n)^{q_A} Adv^{A_I}_{Game_3}(t) \geq \frac{1}{e^2(q_c+q_s)} Adv^{A_I}_{Game_3}(t)$, for q_c is the total number A_I asks for **Certificate query**, q_s is the total number A_I asks for **Sign query**, q_A is the total number A_I asks for **AggSign query**, and e is the base of natural logarithm.

Theorem 4.4. *If there exists an adversary* A_{II} *with an non-negligible success probability* $\mathrm{Adv}^{\mathrm{A}_{II}}_{\mathrm{Game}_4}(\mathrm{t})$ *in forging a valid aggregate signature, the Computational Diffie-Hellman Problem would be solved by a challenger C with a success probability which is not negligible.*

Initialization. C executes **Setup** algorithm for *param*=(G_1,G_2,H_1,H_2,P,Q,e,q) and sets $Q = bP$, medical server's public key $PK_{MS} = sP$. Meanwhile, C initializes six lists L_C, L_K, L_{H1}, L_{H2}, L_S and L_{AS} as empty and returns ($param$, SK_{MS}, PK^*_{IDi}) to A_{II}, where (SK_{MS}, PK^*_{IDi}) = (s, aP).

Queries. A_{II} make queries to C adaptively. When received **Corrupt query**, **H1 query** and **H2 query**, C responds the same as in game 2.

(a) **Key query:** when A_{II} makes a query with a sensor identity ID_i, C returns PK_{IDi}, if (ID_i, PK_{IDi}) has existed on L_K, otherwise,
  (i) If $ID_i \neq ID^*_i$, C runs **KeyGen** algorithm for ID_i's private key SK_{IDi}=x_i, and public keys PK_{IDi}=$x_i P$. Then, C returns PK_{IDi} to A_{II} and adds (ID_i, SK_{IDi}, PK_{IDi}) to L_K.
  (ii) If ID_i=ID^*_i, C returns PK^*_{IDi}+$h^*_i U^*_i = aP$ and adds (ID_i, $\perp$, aP) to L_K.

(b) **Sign query:** When A_{II} makes a query with (m_i,PK_{IDi},ID_i), C returns σ_i, if L_S contains (m_i,PK_{IDi},ID_i,σ_i). Otherwise, C searches L_K for the private key x_i and selects a value $coin_i \in \{0,1\}$, for Pr[$coin_i$=1]=δ, where δ=1/(q_c +q_s), q_s is the total number A_I asks for **Sign query** and q_c is the total number A_I asks for **Certificate query**.
  (i) If $coin_i$=1, C aborts.
  (ii) If $coin_i$=0, C searches L_{H1} for Q_i, and calculates $Cert_{IDi} = sQ_i$, then searches L_{H2} for (m_i, PK_{IDi},U_i, h_i). If it does not exist, C will make a **H2 querys**. After that, C runs **Sign** algorithm for an aggregate signature σ_i =$x_i Q$+($a_i Q$+$Cert_{IDi}$)h_i. Finally, C outputs σ_i and adds (m_i,ID_i,PK_{IDi},σ_i) to L_S.

(c) **AggSign query:** When A_{II} makes this query with (M, ID, PK) = [(m_1,ID_1, PK_1),..., (m_i,ID_i,PK_{IDi}),...,(m_n,ID_n,PK_{IDn})], for 1$\leq$ i $\leq$ n, C searches the list L_{AS} first. If (M,ID,PK,σ_{Ai}) has existed on L_{AS}, C returns σ_{Ai}. Otherwise,

(i) If $coin_i$ =1, C aborts.
(ii) If $coin_i$ =0, C searches L_S for (m_i,ID_i,PK_{IDi},σ_i). If it isn't on the list, C runs **Sign** algorithm for signatures σ_i and adds (m_i, ID_i, PK_{IDi}, σ_i) to L_S. Then, C runs **Aggregate** algorithm for a aggregate signature $\sigma_{Ai} = (T = \sum_{i=1}^{n} T_i, U_i)$. Finally, C sends σ_{Ai} to A_I and adds (M, ID, PK,σ_{Ai}) onto L_{AS}.

Forgery: A_{II} outputs k public keys (PK_1,...,PK^*_{IDi},...,PK_k), k messages (m_1, m_2,...,m_k), a corresponding aggregate signature σ^* under PK_1,...,PK^*_{IDi},...,PK_k, where $k \leq n$.

If A_{II} forged a valid aggregate signature tuple (M, ID, PK,σ^*). Then,$\sigma^*=\sum_{i=1}^{n} \sigma_i$ =(σ_1+... $+\sigma_j$+...+σ_n), A_{II} can compute $\sigma_j=\sum_{i=1}^{n} \sigma_i$- $\sum_{i=1,i\neq j}^{n} \sigma_i$, note that $\sigma_j = (x^*_i Q+(a_i Q+Cert_{IDi})h^*_i$ is the signature on tuple (m^*_i,ID^*_i,PK^*_{IDi}). Finally, A_{II} can compute $abP = (x^*_i+a_i h_i)Q=$ σ_j-$Cert_{IDi}h^*_i$. for given $PK^*_{IDi}+h^*_i U^*_i = aP, Q_i = bP$.

Therefore, if A_{II} succeed with a non-negligible probability $Adv^{A_{II}}_{Game_4}(t)$, the CDHP would be solved with a success probability $\varepsilon' \geq (\frac{1}{n})^{q_k}\delta(1-\delta)^{q_s+q_{As}}(1-\frac{1}{n})^{q_c} Adv^{A_{II}}_{Game_4}(t) \geq \frac{1}{e^2 q_a \times n^{qk}} Adv^{A_{II}}_{Game_4}(t)$, for q_k is the total number A_{II} asks for **Key query**, q_c is the total number A_{II} asks for **Corrupt query**, q_{AS} is the total number A_{II} asks for **AggSign query**, q_s is the total number A_{II} asks for **Sign query**, n is total number of sensors and e is base of natural logarithm.

5. **Performance analysis.** Now, we analyse the efficiency of our scheme. In terms of calculation cost, table 1 displays the comparison of some aggregate signature schemes.

TABLE 1. Efficiency comparison of some aggregate signature schemes

Schemes	Sign	Verify	Aggregate	AggVerify
FZZD [24]	$3T_h$ +$2T_{Exp}$+$3T_G$	$3T_{BP}$+$4T_h$ +T_{Exp}+T_G	(n-1)T_G	$3T_{BP}$+2nT_{Exp} +(3n+1)T_h + nT_G
YMCWW[17]	$2T_h$ +$4T_{Exp}$	$3T_{BP}$+$3T_h$+ $3T_{Exp}$+$2T_G$	nT_{BP}+T_h	2nT_{BP}+3nT_{Exp}+ (3n+1)T_h+2nT_G
WZSGS[38]	$2T_h$ +$4T_{Exp}$ +$3T_G$	$3T_{BP}$+$3T_h$ +T_{Exp}+$2T_G$	nT_{BP}+T_h	(2n+3)T_{BP}+nT_h+ 2nT_G
Ours	T_h +$4T_{Exp}$+$2T_G$	$3T_{BP}$+$2T_h$ +$2T_{Exp}$+T_G	(n-1)T_G	$3T_{BP}$+2nT_{Exp}+ 2nT_h+nT_G

Some notations about execution time in Table 1 which indicate the efficiency of the scheme we proposed are defined as follows:

T_G: calculation time for an addition operation in an additive group,
T_{Exp}: calculation time for a multiplication operation in an additive group,
T_{BP}: calculation time of a bilinear pairing operation,
T_h: calculation time of a hash function operation.
n: the total number of sensors.

To evaluate performance objectively, We employ a bilinear pairing e: $G_1 \times G_1 \rightarrow G_2$, G_1 and G_2 are additive groups, P is generated point on an elliptic curve E: $y^2=x^3+x$ mod p, q is their order. To achieve 80 bits security level, the size of the order q is 160 bits and

the size of prime p in the elliptic curve $E(Fp)$ is 512 bits. The comparison is performed on Intel i3 2.4GHz laptop with 8GB RAM under windows 7 operation system.

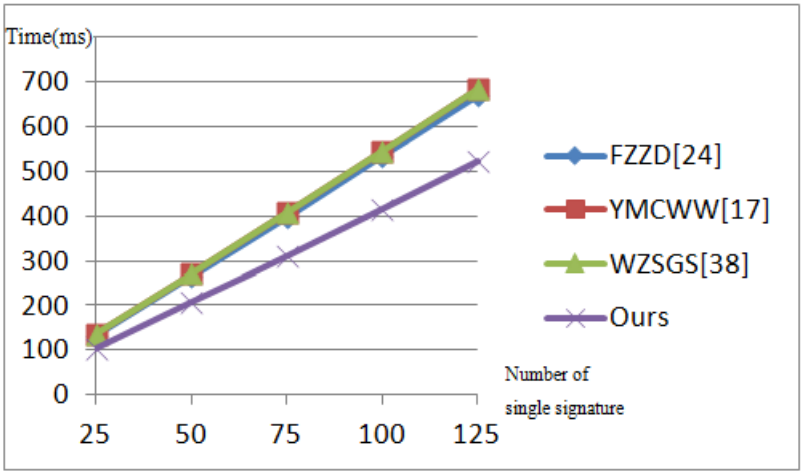

FIGURE 2. Time cost of sign algorithm

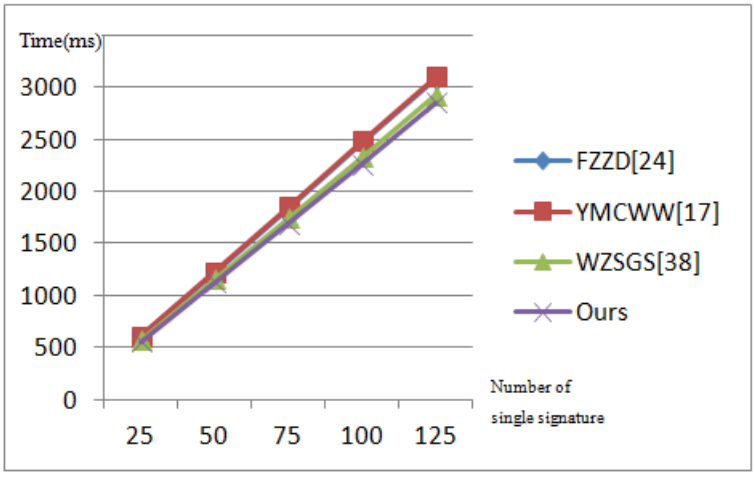

FIGURE 3. Time cost of verify algorithm

From figure 2, it's noticeable that the work presented by Fang et al.[24],Yang et al.[17] and Wu et al.[38] have similar efficiency in the sign phase. The calculation time of our signature scheme is more efficient than that of theirs. From figure 3, represents the calculation time in the verify phase. The calculation power of sensors is very limited in healthcare wireless sensor networks. It's easy to see that the aggregate signature scheme we proposed needs less calculation time in the process of sign and verify, and is suitable for data transmission in HWSNs.

6. **Conclusions.** In this paper, for efficient and secure communication in HWSNs, we proposed a certificate-based aggregate signature scheme with constant bilinear pairing operation. The aggregate scheme protects the on-line data from the dishonest certifier and unauthorized entities in HWSNs. Our scheme is proved secure by the computational Diffie-Hellman problem under the random oracle model. It is apparent that our scheme is efficient and suitable for HWSNs, through the comparison of calculation cost.

Acknowledgment. The work was supported by the research project of Fujian provincial education department of China under grant No.JAT190360, the natural science foundation of Fujian Province of China under grant No.2019I001,2019J01752,2019J01750,2020J01814. natural science foundation of Fuzhou city (No.2019-G-40).

REFERENCES

[1] J. S. Pan, T. T. Nguyen, S. C. Chu, T. K. Dao and T. G. Ngo, Diversity enhanced ion motion optimization for localization in wireless sensor network, *Journal of Information Hiding and Multimedia Signal Processing*, vol.10, no.1, pp.221–229, 2019.

[2] J. Zhang, H. R. Nian, X. C. Ye, X. R. Ji and Y. He, A spatial correlation based partial coverage scheduling scheme in wireless sensor networks, *Journal of Network Intelligence*, vol.5, no.2, pp.34–43, 2020.

[3] P. K. Kumar, S. Kumari,V. Sharma, A. K. Sangaiah, J. H. Wei, and X. Li, A certificateless aggregate signature scheme for healthcare wireless sensor network. *Sustainable Computing:Informatics and Systems*, vol.18, pp.80–89, 2018.

[4] T. Y. Wu, C. M. Chen, K. H. Wang, C. Meng and E. K. Wang, A provably secure certificateless public key encryption with keyword search,*Journal of the Chinese Institute of Engineers*, vol.42, no.1, pp.20–28, 2019.

[5] T. Y. Wu,C. M. Chen, K. H. Wang, and M. T. Wu, Security analysis and enhancement of a certificateless searchable public key encryption scheme for IIoT environments,*IEEE Access*,vol.7, pp.49232–49239, 2019.

[6] Z. J. Huang, K. F. Chen, and Y. M. Wang, Efficient identity-based signatures and blind signatures,*4th International Conference on Cryptology and Network Security, Lecture Notes in Computer Science*, vol.3810, pp.120–133, Berlin, Germany, 2005.

[7] T. Y. Wu, T. T. Tsai, and Y. M. Tseng, Revocable ID-based signature scheme with batch verifications,*Proc. Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp.49–54, Piraeus-Athens, Greece, 2012.

[8] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, Aggregate and verifiably encrypted signatures from bilinear maps, *Proceedings of Eurocrypt 2003*, Vol.2656, pp. 416–432,Berlin, Germany ,2003.

[9] Lysyanskaya, S. Micali, L. Reyzin, H. Shacham, Sequential aggregate signatures from trapdoor permutations, *Proceedings of Eurocrypt 2004*, Vol.3027, pp.74–90, Berlin Germany, 2004.

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

[11] J. N. Chen, Q. S. Chen, and F. M. 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.

[12] P. V. Reddy, P.V.S.S.N. Gopal, Identity-based key-insulated aggregate signature scheme,*Computer and Information Sciences*, vol.29, pp.303–310, 2017.

[13] K. Lee, D.H. Lee, and M. Yung , Sequential aggregate signatures with short public keys:design, analysis and implementation studies,*PublicKey Cryptography PKC 2013. Lecture Notes in Computer Science* ,vol.7778, pp.423-442, 2013.

[14] J. N. Chen, F. M. Zou,T. Y. Wu, and Y. P. Zhou, A new certificate-based aggregate signature scheme for wireless sensor networks,*Journal of Information Hiding and Multimedia Signal Processing*, vol.9, no.5, pp.1264–1280, 2018.

[15] J. N. Chen, Z. J. Huang, Y. P. Zhou, F. M. Zou, C. M. Chen, J. M. T. Wu, and T. Y. Wu, An efficient certificate-based aggregate signature scheme for vehicular Ad Hoc networks,*IET Networks*, DOI:10.1049/iet-net.2020.0019.

[16] S. J. Horng, S. A. F. Tzeng, P. H. Huang, X. Wang, T. R. Li, and M. K. Khan, An efficient certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks,*Information Sciences*, vol.317, pp.48–66, 2015.

[17] X. D. Yang, T. C. Ma, C. L. Chen, P. L. Wang and C. F. Wang, Security analysis and improvement of certificateless aggregate signature scheme for vehicular Ad Hoc networks, *Journal of Electronics and Information Technology*, vol.41, no.5, pp.1265–1270, 2019.

[18] C. M. Chen, B. Xiang, T. Y. Wu, and K. H. Wang, An anonymous mutual authenticated key agreement scheme for wearable sensors in wireless body area networks,*Applied Sciences*, vol.8, pp.1074–1088, 2018.

[19] C. M. Chen, X. Zheng, T. Y. Wu, A Complete Hierarchical Key Management Scheme for Heterogeneous Wireless Sensor Networks,*The Scientific World Journal*, Vol.2014, Article ID: 816549.

[20] C. M. Chen, Y. H. Lin, Y. C. Lin, H. M. Sun, Recoverable concealed data aggregation for data integrity in wireless sensor networks.*IEEE Transactions on parallel and distributed systems*, vol.23, no.4, pp.727–734 ,2012.

[21] C. M. Chen, Y. H. Lin, Y. H. Chen, H. M. Sun, Secure Aggregation via Successively Hierarchical Inspecting of Message Integrity on WSN.*Journal of Information Hiding and Multimedia Signal Processing*, vol.4, no.1, pp.57–72,2013.

[22] H. M. Sun, Y. C. Hsiao, Y. H. Lin, C. M. Chen, An efficient and verifiable concealed data aggregation scheme in wireless sensor networks.*2008 International Conference on Embedded Software and Systems*, pp.19–26,2008.

[23] C. T. Li, T. Y. Wu, C. L. Chen, C. C. Lee, and C. M. Chen, An efficient user authentication and user anonymity scheme with provably security for IoT-based medical care system,*Sensors*, vol.17, pp.1482–1499, 2017.

[24] B. G. Fang, B. C. Zhong,J. L. Zhang and J. R. Ding, An improved certificateless aggregation signature scheme in wireless medical sensor networks.*Intelligent Computer and Applications*,vol.10, no.1, pp.128–132, 2019.

[25] X. T. Wang and H. F. Zhu, A novel two-party key agreement protocol with the environment of wearable device using chaotic maps,*Data Science and Pattern Recognition*, vol.3, no.2, pp.12–23, 2019.

[26] T. Y. Wu, Z. Y. Lee, M. S. Obaidat, S. Kumari, S. Kumar and C. M. Chen, An authenticated key exchange protocol for multi-server architecture in 5G networks,*IEEE Access*, vol.8, pp.28096–28108, 2020.

[27] C. M. Chen, B. Xiang, K.H. Wang, Y. Zhang, and T.Y. Wu, An efficient and secure smart card based authentication scheme,*Journal of Internet Technology*, vol.20, no.4, pp.1113–1123, 2020.

[28] C. M. Chen, K. H. Wang, W. C. Fang, T. Y. Wu and E. K. Wang, Reconsidering a lightweight anonymous authentication protocol,*Journal of the Chinese Institute of Engineers*, vol.42, no.1, pp.9–14, 2019.

[29] L. Wu, Z. Xu, D. He, and X. Wang, New certificateless aggregate signature scheme for healthcare multimedia social network on cloud environment,*Security and Communication Networks*, 2595273:1–2595273:13, 2018.

[30] J. Kim, H. Oh, FAS: Forward secure sequential aggregate signatures for secure logging,*Information Science*. vol.471, pp.115–131, 2019.

[31] A. P. G. Lopes, P. R. L. Gondim, Group authentication protocol based on aggregated signatures for D2D communication,*Computer Networks*, vol.178, 107192, 2020.

[32] K. H. Wang, C. M. Chen, W. C. Fang, T. Y. Wu, On the security of a new ultra-lightweight authentication protocol in IoT environment for RFID tags,*The Journal of Supercomputing*, Vol.74, Issue.1, pp.65–70, 2018.

[33] K. H. Wang, C. M. Chen, W. C. Fang and T. Y. Wu, A secure authentication scheme for Internet of Things,*Pervasive and Mobile Computing*, vol.42, pp.15–26, 2017.

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

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

[36] L. Chen, Z. Cheng and N. P. Smart, Identity-based key agreement protocols from pairings,*International Journal Information Security*, vol.6, no.4, pp.213–241,2007.

[37] D. Pointcheval and J. Stern, Security arguments for digital signatures and blind signatures,*Journal of Cryptology*, vol.13, no.3, pp.361–396, 2000.

[38] G. Wu, F. T. Zhang, L. M. Shen, F. C. Guo, and W. Susilo, Certificateless aggregate signature scheme secure against fully chosen-key attacks.*Information Science*. vol.514, pp.288–301, 2020.