Cryptanalysis and Improvement of a Pairing-Free Certificateless Aggregate Signature in Healthcare Wireless Medical Sensor Networks

Yu Zhan, Baocang Wang, and Rongxing Lu, *Senior Member, IEEE*

***Abstract*—The healthcare wireless medical sensor network is gradually changing the traditional mode of medical treatments with the rapid development of Internet of Things. Specifically, patients' healthcare data can be continuously collected by medical sensor nodes and transmitted to the medical specialists for disease monitoring, diagnosis and treatments. Recently, due to its advantages of low computational and communication overheads in a multiuser environment, the certificateless aggregate signature (CLAS) scheme has been adopted to prevent the sensitive healthcare data from being tampered and damaged, thereby ensuring the integrity and authenticity of data. In order to further improve the efficiency of CLAS schemes for the sensor nodes with limited resources, several CLAS schemes without bilinear pairing have been proposed. However, security issues prevent them from being fully applied in the practical scenarios. In this article, we analyze the security of a pairing-free CLAS scheme proposed by Liu *et al.* [IEEE Internet of Things Journal, vol. 7, no. 6, pp. 5256–5266, 2020] by pointing out that their scheme is insecure against adversaries. After that, we introduce an improved scheme to solve the security vulnerability. The security proofs show that our improved scheme is existentially unforgeable against chosen message attacks under the random oracle model. In addition, the length of the aggregate signature in our proposal does not increase with the growth of the number of users, which greatly reduces the communication cost. Finally, the efficiency of our scheme is illustrated through both performance analyses and comparisons of related work.**

***Index Terms*—Certificateless aggregate signature (CLAS), cryptanalysis, elliptic curve cryptosystem, healthcare wireless medical sensor networks (HWMSNs), without pairing.**

Manuscript received June 24, 2020; revised September 4, 2020 and October 7, 2020; accepted October 20, 2020. Date of publication October 23, 2020; date of current version March 24, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802000; in part by the National Natural Science Foundation of China under Grant U19B2021 and Grant U1736111; in part by the National Cryptography Development Fund under Grant MMJJ20180111; in part by the Fundamental Research Funds for the Central Universities; and in part by the Innovation Fund of Xidian University under Grant 502210221150004. *(Corresponding author: Baocang Wang.)*

Yu Zhan and Baocang Wang are with the State Key Laboratory of Integrated Service Networks and the Crptographic Research Center, Xidian University, Xi'an 710071, China (e-mail: yzhan1993@163.com; bcwang79@aliyun.com).

Rongxing Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).

Digital Object Identifier 10.1109/JIOT.2020.3033337

I. Introduction

INTERNET of Things (IoT) [1] is an extension and development of the traditional Internet, which interconnects of all things through the data interaction network composed of kinds of sensor devices and the information media. As an information technology that can change the way of human life, IoT maximizes the value of data and resources through the continuously comprehensive perception of the transaction, the real-time data sharing and transmission, as well as the efficient information extraction and intelligent analysis. Therefore, IoT has been deployed in various fields, such as industries, economies, medical treatments, education, and public services.

Healthcare wireless medical sensor network (HWMSN) [2] is a significant application of IoT in the medical field. A typical HWMSN system consists of various medical sensor nodes (MSNs), a central control agency and a medical center. Several medical sensors are placed on the body surface of patients or implanted into the body to monitor their medical information and vital signs in real time, including respiration, heartbeat, temperature, blood pressure, blood glucose, blood oxygen saturation, etc. Patients' medical data is transmitted from the sensors to the central control for packaging and integration, then sent to the medical center. Healthcare professionals make diagnoses and put forward the views of the treatments for patients according to these medical data. Obviously, with HWMSN systems, medical resources are integrated and efficiently distributed, and also patients can get the timely and accurate medical feedback, improving the comfort of treatments. Although HWMSN is an incipient technology, its development prospects are certainly remarkable with the development of IoT.

Nevertheless, there are worrisome privacy and security issues in the HWMSN system, as the data collected and transmitted by sensors are the healthcare data of patients, which are very sensitive. In the view of privacy, unauthorized individuals should be prevented from intercepting patients' data. While for security, the data of patients should not be forged, tampered or injected, since it will lead to a wrong diagnosis made by healthcare professionals, which may endanger the life and health of patients. Signature schemes have the ability to ensure the integrity of data while supporting the unforgeability and public verifiability of signatures [3]. However, the

computational and communication costs of ordinary signature schemes are enormous when the number of users and signatures is large. As a result, it is inadvisable to apply the ordinary signature schemes to HWMSN directly.

Certificateless aggregate signature (CLAS) [4] enjoys the advantages of the certificateless cryptosystem while providing the functionalities of aggregate signature schemes. In particular, CLAS supports aggregating n signatures signed by n different users into a single signature. In this way, a verifier only needs to verify once to determine whether all signatures are valid, which greatly reduces the computational and communication costs in the verification procedure. Meanwhile, due to the characteristic of certificateless, CLAS does not suffer from the certificate management and key escrow problems in the public key infrastructure-based and identity-based public cryptosystems. CLAS is a suitable and powerful solution to the security issue of HWMSN.

Recently, Gayathri *et al*. [5] constructed an efficient and secure certificateless aggregate scheme without pairing for HWMSN. Their scheme greatly improves the efficiency of signing and verification, and reduces the communication overhead of transmitting signatures while claiming to be secure. However, their solution has a fatal security hole. Liu *et al*. [6] put forward effective attack methods to prove that Gayathri *et al*.'s CLAS scheme is insecure against two kinds of attacks. Furthermore, Liu *et al*. gave an improved CLAS to solve the security problems. Unfortunately, we found that Liu *et al*.'s scheme can not achieve the expected goal.

In this article, we first point out that the security of Liu *et al*.'s improved scheme cannot be guaranteed. An attacker is capable of forging a valid signature easily, which damages the interests and even health of patients. After that, we fix the security breach in their scheme to prevent both external and internal attackers from forging signatures. Specifically, our main contributions are summarized as follows.

1) First, an attack algorithm is presented to show that Liu *et al*.'s scheme cannot resist the forgery attack launched by the attackers who only can obtain public information.
2) Second, we improve Liu *et al*.'s scheme to achieve a secure signature aggregation. The improved CLAS scheme is proven to be existentially unforgeable against chosen message attack (CMA) under the random oracle model. Concretely, the improved scheme is secure against the public key replacement attack launched by external attackers, as well as the attack launched by a malicious medical server (MS).
3) Third, the aggregate signature length of our proposal is constant, which costs a few communication resources.
4) Finally, we analyze the execution efficiency and communication complexity of the proposed scheme. The comparison results show that our scheme is more efficient than related works [7]–[10].

The remainder of this article is organized as follows. We review the related work in Section II and introduce some background knowledge in Section III. The system model and security model of the proposed CLAS scheme for HWMSN are given in Section IV. The descriptions and security analysis of Liu *et al*.'s scheme are given in Section V. We present our improved CLAS scheme and corresponding security proofs in Section VI. The computational and communication analyses and the comparison of related works are shown in Section VII. Finally, we conclude in Section VIII.

II. Related Work

In 2003, Boneh *et al*. [11] and Al-Riyami and Paterson [12] first introduced the concepts of aggregate signature and certificateless public key cryptosystem, respectively. In a multiuser environment, aggregate signature could enhance the efficiency of signature verification and reduce the bandwidth cost of signature transmission. For the certificateless public key cryptosystem, it solves the problem of certificate management in the traditional public key infrastructure-based cryptosystem, and overcomes the inherent key escrow issue in the identity-based cryptosystem. It is a matter of course to combine such two powerful cryptographic tools to maximize their advantages.

In 2007, Castro and Dahab [4] presented the first CLAS scheme with bilinear pairing. In the same year, Gong *et al*. [13] proposed two CLAS schemes from bilinear maps. However, the computational complexity of these schemes is too high, since the number of involved bilinear pairing operations is linear. To overcome the deficiency, Xiong *et al*. designed a CLAS scheme with constant pairing operations. Unfortunately, a security flaw was discovered in their scheme by Tu *et al*. [14], Cheng *et al*. [15], and He *et al*. [16], respectively. In addition, Li *et al*. [7] pointed out that He *et al*.'s improved scheme [16] is still hard to resist a malicious key generation center (KGC). Successively, some existing schemes have also been pointed out as potential safety hazards. Zhang *et al*. [17] indicated that Chen *et al*.'s [18] scheme cannot resist the public key replacement attack and a malicious KGC. At the same time, Liu *et al*.'s [19] scheme was found to be insecure in [17].

On the other hand, researchers have also focused on the scenarios where the CLAS scheme is suitable for application, such as vehicular ad-hoc networks [20]–[24], HWMSN [5], [6], [8], [9], [25], [26], and so on. For a novel CLAS scheme constructed by Malhi and Batra [20], Kumar and Sharma [21] successfully designed an attack algorithm to forge a signature, and proposed an improved scheme. Yang *et al*. suggested that Kumar *et al*.'s scheme did not achieve the expected security goal. In 2017, Kumar *et al*. [25] came up with a CLAS scheme specifically for the scene of HWMSN. However, their scheme was broken by Wu *et al*. [27] and Zhan and Wang [28], respectively.

Due to the bilinear pairing operation costs a lot of computational resources, the efficiency of the CLAS scheme needs to be further improved to better fit the resource-limited sensor nodes. Xie *et al*. [9], Cui *et al*. [24], and Qu and Mu [29] successively put forward CLAS schemes without pairing. They adopted the additive elliptic curve group to avoid using bilinear pairing. However, the schemes [24], [29] were proved to be insecure by Du *et al*. [8].

In addition to HWMSN, there is also a healthcare system based on IoT named Healthcare IoT Network [30]. The infrastructures of the two systems are similar. The difference is that

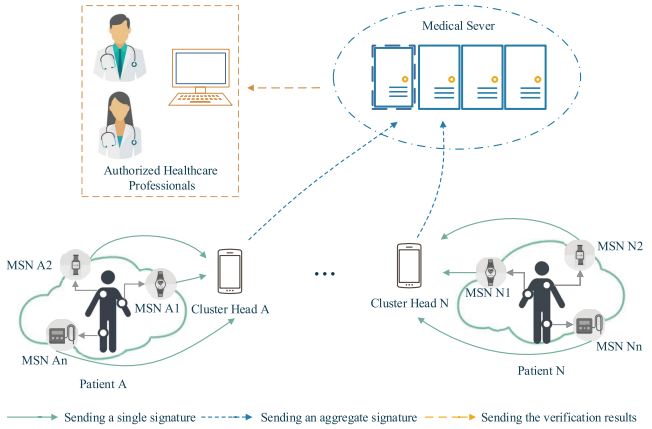


Fig. 1. Framework of CLAS for HWSN.

the cloud server in Healthcare IoT Network is a semi-trusted third party, while the MS in HWMSN is much more trusted than the cloud. Healthcare IoT Network aims at the scene that the user's healthcare data is stored and shared through the cloud server. The scenario targeted by HWMSN is that MS is in charge of collecting user data and transmitting it to the authorized healthcare professional (AHP) for analysis.

III. Preliminaries

In this section, we briefly discuss some necessary background knowledge.

For a prime order finite field $\mathbb{F}_q(q > 3)$, an elliptic curve E over $\mathbb{F}_q$ is defined as the set of points $P = (x, y)$ [31], [32], where x and y are the solutions of the equation $E : y^2 = x^3 + ax + b(\text{ mod } q)$. In particular, a and b are constants in $\mathbb{F}_q$ and satisfy $4a^3 + 27b^2 \neq 0(\text{ mod } q)$. There is an infinity point O that makes the set of points be a group G, i.e., $G = |P = (x, y)|x, y$ satify the above equation$| \cup |O|$. Based on the elliptic curve, there is the following assumption of intractability problem..

Elliptic Curve Discrete Logarithm Problem (ECDLP): Given a point pair $(P, Q) \in G \times G$, find z from $\mathbb{Z}_q^*$ such that $Q = zP$.

IV. System Model and Security Model of the CLAS Scheme for HWMSN

A. System Model of the CLAS Scheme for HWMSN

As shown in Fig. 1, there are four parties involved in a CLAS scheme for HWMSN, including MSNs, cluster head (CH), MS, and AHPs [5], [6]. The descriptions of these parties are given as follows.

1) *MSNs:* MSNs are resources-limited devices installed on the surface or inside of a patient's body to collect healthcare data. While transmitting the sensitive messages to the corresponding CH, each MSN signs the message with its own secret key and sends the signature to the CH.
2) *CH:* MSNs on the same patient correspond to a CH which is responsible for data pretreatments. After receiving the messages and signatures from MSNs, the CH aggregates all signatures into one aggregate signature, and integrates all messages. Finally, the CH sends the aggregate signature and messages to MS.
3) *MS:* MS is in charge of receiving and verifying the validity of messages. Specifically, MS uses the public keys of MSNs to verify the aggregate signature. If the aggregate signature is valid, which implicates that all signatures are legal, MS sends the patient's healthcare data to AHPs.
4) *AHP:* AHPs with professional medical knowledge make diagnosis and treatment plan based on the received patient's data.

B. Related Algorithm

As introduced in [5], [6], [15], and [33], a CLAS scheme is composed of seven algorithms, which are described as follows.

MasterKeyGen: Take a security parameter k as input, this algorithm outputs the master secret key msk and system parameters $params$.

PartialKeyGen: Input the system parameters $params$, the master secret key msk, and the real identity RID_i of a sensor node MSN_i. This algorithm outputs a partial private key D_i and a pseudo identity ID_i for MSN_i. The pseudo identity is used to prevent the disclosure of real identities, which further protects users' privacy.

UserKeyGen: Enter the identity ID_i of a sensor node MSN_i. This algorithm outputs a public/secret key pair (pk_i, sk_i) for MSN_i.

Sign: Input the identity ID_i, the secret key sk_i, the partial private key D_i of a sensor node MSN_i, and a message m_i. This algorithm outputs a single signature σ_i on m_i.

Verify: With the inputs of a signature σ_i, a message m_i, and the public key pk_i under ID_i of a sensor node MSN_i, this algorithm outputs *Ture* if the signature σ_i is valid, or a symbol $\perp$ otherwise.

Aggregate: Input n signatures $\{\sigma_i, i = 1, \ldots, n\}$, and n messages $\{m_i, i = 1, \ldots, n\}$. This algorithm outputs an aggregate signature σ on $\{m_i, i = 1, \ldots, n\}$.

AggregateVerify: Input an aggregate signature σ, n messages $\{m_i, i = 1, \ldots, n\}$, n public keys $\{pk_i, i = 1, \ldots, n\}$ under $\{\text{ID}_i, i = 1, \ldots, n\}$. This algorithm outputs *Ture* if the aggregate signature σ is valid, or a symbol $\perp$ otherwise.

C. Security Models for CLAS Scheme

A CLAS scheme is existentially unforgeable against CMAs if it can resist two types adversaries, i.e., Type I and Type II adversaries.

1) *Type I Adversary:* An "external" adversary who has the ability to launch the public key replacement attack. Specifically, the Type I adversary can compromise the secret key of a sensor node or replace a node's public key with the value chosen by him. However, the Type I adversary cannot obtain the master secret key or the partial private keys of sensor nodes.
2) *Type II Adversary:* An "internal" adversary, i.e., malicious MS, who owns the master secret key. The Type II adversary cannot compromise the secret keys or replace the public keys of sensor nodes.

The CMA security model for CLAS schemes consists of four games. Before delving into the details of games, we introduce the following oracles provided by the challenger that adversaries can query.

1) *Create User Oracle $\mathcal{O}_{CU}$ (ID_i):* When adversaries query this oracle, the challenger runs $UserKeyGen(\mathrm{ID}_i) \rightarrow (pk_i, sk_i)$ and $PartialKeyGen(msk, \mathrm{ID}_i) \rightarrow D_i$. Then, the challenger records $(pk_i, sk_i, D_i, \mathrm{ID}_i)$ in a list $\mathcal{L}$ and returns the public key pk_i.
2) *Secret Key Oracle $\mathcal{O}_{SK}$(ID_i):* When adversaries query this oracle, the challenger finds the tuple $(pk_i, sk_i, D_i, \mathrm{ID}_i)$ from the list $\mathcal{L}$, then returns the secret sk_i as the query result.
3) *Partial Private Key Oracle $\mathcal{O}_{\mathrm{PPK}}$($ID_i$):* When adversaries query this oracle, the challenger searches the list $\mathcal{L}$ to find $(pk_i, sk_i, D_i, \mathrm{ID}_i)$. Then, the challenger returns the partial private key D_i as the query result.
4) *Replace Key Oracle $\mathcal{O}_{RK}$(ID_i,pk'_i, sk'_i):* When adversaries query this oracle, the challenger finds $(pk_i, sk_i, D_i, \mathrm{ID}_i)$ from the list $\mathcal{L}$ and replaces this record with $(pk'_i, sk'_i, D_i, \mathrm{ID}_i)$.
5) *Sign Oracle $\mathcal{O}_S$(m_i,ID_i):* When adversaries query this oracle, the challenger executes as follows.
   a) If there is no record about ID_i in the list $\mathcal{L}$, return a symbol $\perp$ as the result.
   b) Otherwise, find the current public/secret key pair from the list $\mathcal{L}$, and return the result of running $Sign(\mathrm{ID}_i, sk_i, D_i, m_i)$.

Game I and Game II are aimed at the security of the single signature in the CLAS scheme.

Game I: In this game, $\mathcal{A}_1$ is a probability polynomial time (PPT) Type I adversary.

Setup: In this phase, the challenger $\mathcal{C}_1$ executes *MasterKeyGen* with a security parameter k to produce the master secret key *msk* and system parameters *params*. Then, $\mathcal{C}_1$ keeps *msk* secretly and sends *params* to $\mathcal{A}_1$.

Query: In the query phase, the adversary $\mathcal{A}_1$ makes queries on the oracles $\mathcal{O}_{CU}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{\mathrm{PPK}}$, $\mathcal{O}_{RK}$, and $\mathcal{O}_S$.

Forgery: In the final phase, $\mathcal{A}_1$ chooses a target sensor node MSN_i^* with the identity ID_i^* and the public key pk_i^*, then outputs σ^* as a forged signature on m_i^*. $\mathcal{A}_1$ wins the game if the result of $Verify(\sigma^*, m_i^*, pk_i^*, \mathrm{ID}_i^*)$ is *True* and

1) $\mathcal{O}_S(m_i^*, \mathrm{ID}_i^*)$ has never been queried;
2) $\mathcal{O}_{\mathrm{PPK}}(\mathrm{ID}_i^*)$ has never been queried.

Game II: This game is executed between a PPT Type II adversary $\mathcal{A}_2$ and the challenger $\mathcal{C}_2$.

Setup: The differences from the *Setup* phase in the Game I is that the algorithm *MasterKeyGen* is performed by $\mathcal{A}_2$. Then, $\mathcal{A}_2$ transmits the master secret key *msk* and system parameters *params* to $\mathcal{C}_2$.

Query: In the query phase, the adversary $\mathcal{A}_2$ makes queries on the oracles $\mathcal{O}_{CU}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, and $\mathcal{O}_S$.

Forgery: After selecting a target sensor node with ID_i^* and pk_i^*, $\mathcal{A}_2$ outputs σ^* as a forged signature on m_i^*. $\mathcal{A}_2$ wins the game if the result of $Verify(\sigma^*, m_i^*, pk_i^*, \mathrm{ID}_i^*)$ is *True* and

1) $\mathcal{O}_S(m_i^*, \mathrm{ID}_i^*)$ has never been queried;
2) $\mathcal{O}_{SK}(\mathrm{ID}_i^*)$ has never been queried;
3) $\mathcal{O}_{RK}(\mathrm{ID}_i^*, pk_i', sk_i')$ has never been queried.

Game III and Game IV focus on the security of the aggregate signature in the CLAS scheme.

Game III: A PPT Type I adversary $\mathcal{A}_1$ and the challenger $\mathcal{C}_3$ play the following game.

The *Setup* and *Query* phases are the same as in the Game I.

Forgery: In this phase, $\mathcal{A}_1$ chooses a target set of sensor nodes $\mathcal{U}^* = \{\mathrm{MSN}_1^*, \mathrm{MSN}_2^*, \ldots, \mathrm{MSN}_n^*\}$. The corresponding sets of identities and public keys are $\mathcal{ID}^* = \{\mathrm{ID}_i^*, \mathrm{ID}_2^*, \ldots, \mathrm{ID}_n^*\}$ and $\mathcal{PK}^* = \{pk_1^*, pk_2^*, \ldots, pk_n^*\}$, respectively. After that, $\mathcal{A}_1$ outputs an aggregate signature σ^* on $\mathcal{M}^* = \{m_1^*, m_2^*, \ldots, m_n^*\}$ and wins the game if the following conditions are satisfied.

1) $AggregateVerify(\sigma^*, \mathcal{M}^*, \mathcal{PK}^*, \mathcal{ID}^*) \rightarrow True$.
2) At least one identity ID_j^* that has not been queried for $\mathcal{O}_S(m_j^*, \mathrm{ID}_j^*)$ and $\mathcal{O}_{\mathrm{PPK}}(\mathrm{ID}_j^*)$.

Game IV: The game played by a PPT Type II adversary $\mathcal{A}_2$ and the challenger $\mathcal{C}_4$ is described as follows.

The *Setup* and *Query* phases are the same as in the Game II.

Forgery: In this phase, $\mathcal{A}_2$ outputs an aggregate signature σ^* on $\mathcal{M}^* = \{m_1^*, m_2^*, \ldots, m_n^*\}$, and states the target set of sensor nodes $\mathcal{U}^* = \{\mathrm{MSN}_1^*, \mathrm{MSN}_2^*, \ldots, \mathrm{MSN}_n^*\}$. The identities of the target sensor nodes and the corresponding public keys are $\mathcal{ID}^* = \{\mathrm{ID}_i^*, \mathrm{ID}_2^*, \ldots, \mathrm{ID}_n^*\}$ and $\mathcal{PK}^* = \{pk_1^*, pk_2^*, \ldots, pk_n^*\}$, respectively. $\mathcal{A}_2$ wins the game if the following conditions hold.

1) $AggregateVerify(\sigma^*, \mathcal{M}^*, \mathcal{PK}^*, \mathcal{ID}^*) \rightarrow True$.
2) At least one identity ID_j^* has not been queried for $\mathcal{O}_S(m_j^*, \mathrm{ID}_j^*)$, $\mathcal{O}_{SK}(\mathrm{ID}_j^*)$ and $\mathcal{O}_{RK}(\mathrm{ID}_j^*, pk_j', sk_j')$.

If the probabilities that any PPT Type I adversary wins the Game I and Game III are negligible, and the probabilities that any PPT Type II adversary wins the Game II and Game IV are negligible, the CLAS scheme is believed to be CMA secure.

V. Description and Security Analyses of the CLAS Scheme Proposed by Liu *et al.*

A. Liu et al.'s CLAS Scheme

In this section, we briefly review the improved CLAS scheme proposed by Liu *et al*. [6]. The details of their scheme are described as follows.

1) *MasterKeyGen:* The system is bootstrapped by MS. First, MS selects a group G of order q and a generator P according to the security parameter k, where q is a prime. After that, MS randomly selects $s \in \mathbb{Z}_q^*$ as the master secret key *msk*, and calculates the corresponding public key as $P_{\mathrm{pub}} = sP$. Then, MS chooses five secure hash functions H, H_1, H_2, H_3, H_4, where $H : G \times G \rightarrow \mathbb{Z}_q^*$, $H_1 : \{0,1\}^* \times G \times G \rightarrow \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times G \rightarrow \mathbb{Z}_q^*$, and $H_3, H_4 : \{0,1\}^* \times \{0,1\}^* \times G \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. Finally, MS makes the parameters $params = \{G, q, P, P_{\mathrm{pub}}, H, H_1, H_2, H_3, H_4\}$ public, and holds the master secret key s on its own.
2) *PartialKeyGen:* Given the public parameters *params*, the master secret key s and the real identity RID_i of a MSN_i, MS randomly selects $r_i \in \mathbb{Z}_q^*$ and computes $R_i = r_iP$, $\mathrm{ID}_i = \mathrm{RID}_i \oplus H(r_iP_{\mathrm{pub}}, T_i)$, $h_{1i} = H_1(\mathrm{ID}_i, R_i, P_{\mathrm{pub}})$ and $d_i = r_i + sh_{1i} \mod q$ in turn, where T_i denotes

the pseudo identity ID_i validity time period. Then, MS sets the partial private key as $D_i = (d_i, R_i)$ and sends $(\mathrm{ID}_i, T_i, D_i)$ to the MSN_i secretly. The MSN_i verifies the valid of the partial private key by checking whether $d_iP = R_i + h_{1i}P_{\mathrm{pub}}$ holds.

3) *UserKeyGen:* The MSN_i with ID_i randomly selects $x_i \in \mathbb{Z}_q^*$. Then, the secret key of the MSN_i is set as $sk_i = (x_i, d_i)$, and the corresponding public key is set as $pk_i = (X_i, R_i) = (x_iP, r_iP)$.
4) *Sign:* The operations that the MSN_i signs a message m_i at time t_i are given as follows.
   a) Randomly select $y_{1i} \in \mathbb{Z}_q^*$ and compute $Y_{1i} = y_{1i}P$.
   b) Compute $u_i = H_2(m_i, \mathrm{ID}_i, Y_{1i})$, $h_{3i} = H_3(m_i, \mathrm{ID}_i, pk_i, t_i)$ and $h_{4i} = H_4(m_i, \mathrm{ID}_i, pk_i, t_i)$.
   c) Compute $W_i = [u_i(y_{1i} + h_{3i}x_i) + h_{4i}d_i]P$.
   d) Output $\sigma_i = (Y_{1i}, W_i)$ as the signature on $m_i||t_i$.
5) *Verify:* The CH uses the public key pk_i on ID_i to verify a signature σ_i on $m_i||t_i$ through following steps.
   a) Compute $u_i = H_2(m_i, \mathrm{ID}_i, Y_{1i})$, $h_{3i} = H_3(m_i, \mathrm{ID}_i, pk_i, t_i)$, and $h_{4i} = H_4(m_i, \mathrm{ID}_i, pk_i, t_i)$.
   b) The signature is valid if the following equation holds:

$$W_i - u_i(Y_{1i} + h_{3i}X_i) = h_{4i}(R_i + h_{1i}P_{\mathrm{pub}}).$$

6) *Aggregate:* The CH computes an aggregate signature with n signatures $\{\sigma_i, i = 1, \ldots, n\}$ on n messages $\{m_i||t_i, i = 1, \ldots, n\}$ from n MSNs as follows.
   a) Compute

$$Y = \sum_{i=1}^{n} u_iY_{1i} = \sum_{i=1}^{n} H_2(m_i, \mathrm{ID}_i, Y_{1i})Y_{1i}.$$

   b) Compute $W = \sum_{i=1}^{n} W_i$.
   c) Set the aggregate signature as $\sigma = (Y, W)$.
7) *AggregateVerify:* To verify an aggregate signature σ signed by n MSNs on $m_i||t_i$, $i = 1, \ldots, n$, MS performs the following operations.
   a) Compute $h_{1i} = H_1(\mathrm{ID}_i, R_i, P_{\mathrm{pub}})$, $h_{3i} = H_3(m_i, \mathrm{ID}_i, pk_i, t_i)$ and $h_{4i} = H_4(m_i, \mathrm{ID}_i, pk_i, t_i)$, $i = 1, \ldots, n$.
   b) Compute $U = \sum_{i=1}^{n} u_ih_{3i}X_i$.
   c) Accept the signature if

$$W - Y - U = \sum_{i=1}^{n} h_{4i}(R_i + h_{1i}P_{\mathrm{pub}})$$

   holds.

B. Cryptanalysis of Liu et al.'s CLAS Scheme

Liu *et al.* set $u_i = H_2(m_i, \mathrm{ID}_i, Y_{1i})$ to enhance the connection between u_i and the other part of a signature. Through this way, u_i becomes verifiable and hard to be tampered with. However, they compute W_i as a point over the elliptic curve E which damages this connection and the effect of the secret keys (x_i, d_i). Hence, an adversary $\mathcal{A}$ could forge a valid signature on an arbitrary message $m_i||t_i$ even without obtaining an existing signature.

1) Randomly select $y_{1i} \in \mathbb{Z}_q^*$ and compute $Y_{1i} = y_{1i}P$.
2) Compute $u_i = H_2(m_i, \mathrm{ID}_i, Y_{1i})$.
3) Compute $h_{1i} = H_1(\mathrm{ID}_i, R_i, P_{\mathrm{pub}})$, $h_{3i} = H_3(m_i, \mathrm{ID}_i, pk_i, t_i)$ and $h_{4i} = H_4(m_i, \mathrm{ID}_i, pk_i, t_i)$.
4) Compute

$$W_i = u_i(Y_{1i} + h_{3i}X_i) + h_{4i}(R_i + h_{1i}P_{\mathrm{pub}}).$$

5) Output a forged signature as $\sigma_i = (Y_{1i}, W_i)$.

Hence, a valid signature can be forged through only using public information. Obviously, the reason for this is that W_i is not a member in $\mathbb{Z}_q^*$ but a point over the elliptic curve E. Thereafter, the most intuitive way to modify the scheme is to replace W_i with $w_i = (u_i(y_{1i} + h_{3i}x_i) + h_{4i}d_i) \bmod q$. In this way, the signature on $m_i||t_i$ is set as $\sigma_i = (Y_{1i}, w_i)$, and the corresponding verification algorithm is changed to the following form.

1) Compute $u_i = H_2(m_i, \mathrm{ID}_i, Y_{1i})$, $h_{1i} = H_1(\mathrm{ID}_i, R_i, P_{\mathrm{pub}})$, $h_{3i} = H_3(m_i, \mathrm{ID}_i, pk_i, t_i)$ and $h_{4i} = H_4(m_i, \mathrm{ID}_i, pk_i, t_i)$.
2) The signature is valid if the following equation holds:

$$w_iP - u_i(Y_{1i} + h_{3i}X_i) = h_{4i}(R_i + h_{1i}P_{\mathrm{pub}}).$$

With this modification, Liu *et al.*'s CLAS scheme can prevent such adversaries from forging signatures. However, the modified scheme is still insecure against the Type II adversary.

Suppose $\mathcal{A}$ is a Type II adversary. Given the master key s, the partial private key $D_i = (d_i, R_i)$, a signature $\sigma_i = (Y_{1i}, u_i, w_i)$ on a message $m_i||t_i$, $\mathcal{A}$ generates a signature on $m_i'||t_i'$ as follows.

1) Compute $h_{3i} = H_3(m_i, \mathrm{ID}_i, pk_i, t_i)$ and $h_{4i} = H_4(m_i, \mathrm{ID}_i, pk_i, t_i)$.
2) Compute $h_{3i}' = H_3(m_i', \mathrm{ID}_i, pk_i, t_i')$ and $h_{4i}' = H_4(m_i', \mathrm{ID}_i, pk_i, t_i')$.
3) Compute $u_i = H_2(m_i, \mathrm{ID}_i, Y_{1i})$.
4) Set $Y_{i1}' = h_{3i}'h_{3i}^{-1}Y_{1i} = h_{3i}'h_{3i}^{-1}y_{1i}P$.
5) Compute

$$\begin{aligned}\delta &= (h_{3i}'h_{3i}^{-1})\left[u_i^{-1}(w_i - h_{4i}d_i)\right] \bmod q \\ &= (h_{3i}'h_{3i}^{-1})\left[u_i^{-1}((u_i(y_{1i}+h_{3i}x_i)+h_{4i}d_i)-h_{4i}d_i)\right] \bmod q \\ &= (h_{3i}'h_{3i}^{-1})y_{1i} + h_{3i}'x_i \bmod q.\end{aligned}$$

6) Compute $u_i' = H_2(m_i', \mathrm{ID}_i, Y_{1i}')$.
7) Compute

$$\begin{aligned}w_i' &= u_i'\delta + h_{4i}'d_i \bmod q \\ &= u_i'\left[(h_{3i}'h_{3i}^{-1}y_{1i}) + h_{3i}'x_i\right] + h_{4i}'d_i \bmod q.\end{aligned}$$

8) Set $\sigma' = (Y_{1i}', w_i')$ as a forged signature on $m_i'||t_i'$.

The correctness of the valid signature σ' is easy to verify. The reason why the scheme is insecure against the Type II adversary is that u_i can be easily removed from $u_i(y_{1i}+h_{3i}x_i)$.

VI. Our Proposal

In this section, we proposed an improved CLAS scheme based on the ECDLP assumption to solve the security issue of Liu *et al.*'s scheme, and the security analysis will be presented later.

A. Improved CLAS Scheme

In our improved scheme, we modify the way of signing and the definitions of hash functions to enhance the security.

1) *MasterKeyGen:* Given a security parameter k, MS selects a group G of prime order q and a generator P. Then, MS randomly selects $s \in \mathbb{Z}_q^*$ as the master secret key, and sets $P_{\text{pub}} = sP$, chooses four secure hash functions H, H_1, H_2, H_3, where $H : G \times G \to \mathbb{Z}_q^*$, $H_1 : \{0,1\}^* \times G \times G \to \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times G \times \{0,1\}^* \times G \to \mathbb{Z}_q^*$, and $H_3 : \{0,1\}^* \times \{0,1\}^* \times G \times \{0,1\}^* \to \mathbb{Z}_q^*$. Finally, the system parameters are $params = \{G, q, P, P_{\text{pub}}, H, H_1, H_2, H_3\}$ and the master secret key is s.
2) The algorithms *PartialKeyGen* and *UserKeyGen* are the same with the corresponding parts in Liu *et al.*'s schemes.
3) *Sign:* The MSN_i signs a message m_i at time t_i as follows.
   a) Choose a random value $y_i \in \mathbb{Z}_q^*$ and compute $Y_i = y_i P$.
   b) Compute $u_i = H_2(m_i, \text{ID}_i, pk_i, t_i, Y_i)$ and $h_{3i} = H_3(m_i, \text{ID}_i, pk_i, t_i)$.
   c) Compute $w_i = [u_i y_i + h_{3i}(x_i + d_i)] \mod q$.
   d) Output $\sigma_i = (Y_i, w_i)$ as the signature on $m_i||t_i$.
4) *Verify:* The CH verifies a signature σ_i on $m_i||t_i$ with a public key pk_i on ID_i as follows.
   a) Compute $h_{1i} = H_1(\text{ID}_i, R_i, P_{\text{pub}})$, $u_i = H_2(m_i, \text{ID}_i, pk_i, t_i, Y_i)$ and $h_{3i} = H_3(m_i, \text{ID}_i, pk_i, t_i)$.
   b) Accept the signature if

$$w_i P - u_i Y_i = h_{3i}(X_i + R_i + h_{1i} P_{\text{pub}})$$

   holds.
5) *Aggregate:* Given n signature $\{\sigma_i, i = 1, \ldots, n\}$ on n messages $\{m_i||t_i, i = 1, \ldots, n\}$ form n MSNs, the CH generates an aggregate signature as follows.
   a) Compute $u_i = H_2(m_i, \text{ID}_i, pk_i, t_i, Y_i)$, $i = 1, \ldots, n$.
   b) Compute $U = \sum_{i=1}^{n} u_i Y_i$.
   c) Compute $w = \sum_{i=1}^{n} w_i$.
   d) Output the aggregate signature $\sigma = (U, w)$.
6) *AggregateVerify:* Given an aggregate signature σ on $\{m_i||t_i, i = 1, \ldots, n\}$, and n public keys $\{pk_i, i = 1, \ldots, n\}$ on identities $\{\text{ID}_i, i = 1, \ldots, n\}$, MS performs the following operations.
   a) Compute $h_{1i} = H_1(\text{ID}_i, R_i, P_{\text{pub}})$, and $h_{3i} = H_3(m_i, \text{ID}_i, pk_i, t_i)$, $i = 1, \ldots, n$.
   b) Accept the signature if

$$wP - U = \sum_{i=1}^{n} h_{3i}(X_i + R_i + h_{1i} P_{\text{pub}})$$

   holds.

Correctness:

$$\begin{aligned} w_i P - u_i Y_{1i} &= \left[u_i y_{1i} + h_{3i}(x_i + d_i)\right] P - u_i Y_{1i} \\ &= u_i y_{1i} P + h_{3i}(x_i P + d_i P) - u_i Y_{1i} \\ &= h_{3i}(X_i + R_i + h_{1i} P_{\text{pub}}). \end{aligned}$$

B. Security Analysis

As mentioned in the security model for CLAS schemes, it needs to prove that there is no PPT adversaries could win the Games I–IV with nonnegligible probabilities, respectively.

Theorem 1: In the random oracle model, the single signature in our improved CLAS scheme is CMA secure under the assumption that the ECDLP in G is intractable.

Proof: This theorem is derived from the Lemmas 1 and 2. ∎

Lemma 1: If there exists a PPT Type I adversary $\mathcal{A}_1$ could forge a valid signature after querying $\mathcal{O}_{CU}$ q_{CU} times, querying $\mathcal{O}_{SK}$ q_{SK} times, querying $\mathcal{O}_{\text{PPK}}$ q_{PPK} times, querying $\mathcal{O}_{RK}$ q_{RK} times, querying $\mathcal{O}_S$ q_S times, querying the random oracles H, H_1, H_2, and H_3, q_H, q_{H_1}, q_{H_2}, q_{H_3}times, respectively, then the ECDLP in G can be solved in polynomial time.

Proof: Given an instance of ECDLP $(P, Q = zP) \in G \times G$, where z is randomly selected from $\mathbb{Z}_q^*$. Suppose the probability that $\mathcal{A}_1$ forges a valid signature successfully is ε, then an algorithm $\mathcal{C}_1$ that can obtain z from (P, Q) in polynomial time is constructed as follows.

Setup: In this phase, $\mathcal{C}_1$ randomly selects $s \in \mathbb{Z}_q^*$ as the master secret key msk, and calculates $P_{\text{pub}} = sP$. Then, $\mathcal{C}_1$ sends $params = \{G, q, P, P_{\text{pub}}\}$ to $\mathcal{A}_1$ and keeps msk secretly.

Query: In the query phase, the adversary $\mathcal{A}_1$ is allowed to make queries on the oracles $\mathcal{O}_{CU}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{\text{PPK}}$, $\mathcal{O}_{RK}$, $\mathcal{O}_S$, H, H_1, H_2, and H_3. $\mathcal{C}_1$ responses the queries as follows.

1) *$H(r_i P_{\text{pub}}, T_i)$:* $\mathcal{C}_1$ maintains an initially empty list $\mathcal{L}_0$. On receiving a query $H(r_i P_{\text{pub}}, T_i)$, $\mathcal{C}_1$ directly returns h_{0i} to $\mathcal{A}_1$ if there exists a tuple $(r_i P_{\text{pub}}, T_i, h_{0i})$ in $\mathcal{L}_0$. Otherwise, $\mathcal{C}_1$ randomly selects $h_{0i} \in \mathbb{Z}_q^*$ and records $(r_i P_{\text{pub}}, T_i, h_{0i})$ in $\mathcal{L}_0$. Then, $\mathcal{C}_1$ returns h_{0i} to $\mathcal{A}_1$.
2) *$H_1(\text{ID}_i, R_i, P_{\text{pub}})$:* After receiving a query $H_1(\text{ID}_i, R_i, P_{\text{pub}})$, $\mathcal{C}_1$ finds the item $(\text{ID}_i, R_i, P_{\text{pub}}, h_{1i})$ from the initially empty list $\mathcal{L}_1$ and returns h_{1i} to $\mathcal{A}_1$. If there is no such item, $\mathcal{C}_1$ randomly selects $h_{1i} \in \mathbb{Z}_q^*$ and records $(\text{ID}_i, R_i, P_{\text{pub}}, h_{1i})$ in $\mathcal{L}_1$. Then, $\mathcal{C}_1$ returns h_{1i} to $\mathcal{A}_1$.
3) *$H_2(m_i, \text{ID}_i, pk_i, t_i, Y_i)$:* When $\mathcal{A}_1$ submits a query $H_2(m_i, \text{ID}_i, pk_i, t_i, Y_i)$, $\mathcal{C}_1$ recovers the item $(m_i, \text{ID}_i, pk_i, t_i, Y_i, h_{2i})$ from a maintained list $\mathcal{L}_2$ and returns h_{2i} as the result. If $\mathcal{C}_1$ fails to do that, $\mathcal{C}_1$ selects $h_{2i} \in \mathbb{Z}_p^*$ randomly and adds it to $\mathcal{L}_2$. Finally, h_{2i} is sent to $\mathcal{A}_1$.
4) *$H_3(m_i, \text{ID}_i, pk_i, t_i)$:* When $\mathcal{A}_1$ submits a query $H_3(m_i, \text{ID}_i, pk_i, t_i)$, if there is an item $(m_i, \text{ID}_i, pk_i, t_i, h_{3i})$ in the list $\mathcal{L}_3$, $\mathcal{C}_1$ returns h_{3i} as the query result. Otherwise, $\mathcal{C}_1$ randomly chooses $h_{3i} \in \mathbb{Z}_p^*$ and adds it to $\mathcal{L}_3$ before sending it to $\mathcal{A}_1$.
5) *$\mathcal{O}_{CU}(\text{ID}_i)$:* When $\mathcal{A}_1$ queries this oracle, $\mathcal{C}_1$ finds the record about ID_i in the list $\mathcal{L}_{CU}$, then returns $(\text{ID}_i, pk_i) = (\text{ID}_i, R_i, X_i)$ to $\mathcal{A}_1$. If there is no such record, $\mathcal{C}_1$ uses the Coron's skill [34] to select a bit $\xi_i \in \{0, 1\}$, where $\Pr[\xi_i = 1] = \theta$ and $\Pr[\xi_i = 0] = 1 - \theta$. If $\xi_i = 1$, $\mathcal{C}_1$ executes as follows.

a) Randomly select $x_i, r_i \in \mathbb{Z}_q^*$ and compute $X_i = x_i P$, $R_i = r_i P$.
b) Query $H(r_i P_{\text{pub}}, T_i)$ to get h_{0i}.
c) Set $\text{RID}_i = \text{ID}_i$ and compute $\text{ID}_i = \text{RID}_i \oplus h_{0i}$.
d) Query $H_1(\text{ID}_i, R_i, P_{\text{pub}})$ to obtain h_{1i}.
e) Compute $d_i = r_i + sh_{1i} \bmod q$.
f) Record $(\xi_i, \text{ID}_i, \text{RID}_i, r_i, R_i, d_i, x_i, X_i)$ in the list $\mathcal{L}_{CU}$.

If $\xi_i = 0$, $\mathcal{C}_1$ preforms the following operations.
a) Randomly select $x_i \in \mathbb{Z}_q^*$ and compute $X_i = x_i P$.
b) Set $R_i = Q$.
c) Obtain h_{0i} by querying $H(sQ, T_i)$.
d) Set $\text{RID}_i = \text{ID}_i$ and compute $\text{ID}_i = \text{RID}_i \oplus h_{0i}$.
e) Record $(\xi_i, \text{ID}_i, \text{RID}_i, -, R_i, -, x_i, X_i)$ in the list $\mathcal{L}_{CU}$.

Finally, $\mathcal{C}_1$ returns (ID_i, R_i, X_i) to $\mathcal{A}_1$.

6) $\mathcal{O}_{SK}(\text{ID}_i)$*:* When $\mathcal{A}_1$ queries this oracle, if there is no record about ID_i in $\mathcal{L}_{CU}$, $\mathcal{C}_1$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{C}_1$ returns x_i to $\mathcal{A}_1$.
7) $\mathcal{O}_{\text{PPK}}(\text{ID}_i)$*:* When $\mathcal{A}_1$ queries this oracle, if there is no record about ID_i in $\mathcal{L}_{CU}$ or the corresponding $\xi_i = 0$, $\mathcal{C}_1$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{C}_1$ returns d_i to $\mathcal{A}_1$.
8) $\mathcal{O}_{RK}(\text{ID}_i, pk_i', sk_i')$*:* When $\mathcal{A}_1$ queries this oracle, if there is no record about ID_i in $\mathcal{L}_{CU}$, $\mathcal{C}_1$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{C}_1$ replaces (x_i, X_i) in the record with (sk_i', pk_i').
9) $\mathcal{O}_S(m_i, \text{ID}_i)$*:* After receiving the query, if any records about ID_i cannot be found in $\mathcal{L}_{CU}$, $\mathcal{C}_1$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, if $\xi_i = 1$, $\mathcal{C}_1$ returns the result of running $Sign(\text{ID}_i, x_i, d_i, m_i)$ to $\mathcal{A}_1$. In the case $\xi_i = 0$, $\mathcal{C}_1$ executes as follows.
a) Randomly select $w_i, u_i \in \mathbb{Z}_q^*$.
b) Query $H_1(\text{ID}_i, R_i, P_{\text{pub}})$ and $H_3(m_i, \text{ID}_i, pk_i, t_i)$ to obtain h_{1i} and h_{3i}, respectively.
c) Compute $Y_i = u_i^{-1}[w_i P - h_{3i}(X_i + Q + h_{1i} P_{\text{pub}})]$.
d) If the item $(m_i, \text{ID}_i, pk_i, t_i, Y_i, u_i)$ already exists in the list $\mathcal{L}_2$, $\mathcal{C}_1$ reselects a different $u_i \in \mathbb{Z}_q^*$ and performs step *c*). Otherwise, $\mathcal{C}_1$ adds $(m_i, \text{ID}_i, pk_i, t_i, Y_i, u_i)$ to $\mathcal{L}_2$, and returns (Y_i, w_i) to $\mathcal{A}_1$.

Forgery: In this phase, $\mathcal{A}_1$ chooses a target user with identity ID_i^* and a message $m_i^* || t_i^*$. We recall that $\mathcal{A}_1$ is forbidden to query $\mathcal{O}_S(m_i^*, \text{ID}_i^*)$. If there is no record about ID_i^* in the list $\mathcal{L}_{CU}$ or the corresponding $\xi_i^* = 1$, $\mathcal{C}_1$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{A}_1$ outputs a valid signature $\sigma_i^* = (Y_i^*, w_i^*)$ on $m_i^* || t_i^*$ such that $w_i^* P - u_i^* Y_i^* = h_{3i}^*(X_i^* + Q + h_{1i}^* P_{\text{pub}})$. Then, with the Forking Lemma [35], $\mathcal{A}_1$ outputs another valid signature $\sigma_i' = (Y_i^*, w_i')$ on $m_i^* || t_i^*$ with the same random tapes y_i^* and different hash value h_{3i}', i.e.,

$$\begin{aligned} w_i^* &= u_i^* y_i^* + h_{3i}^*(x_i^* + d_i^*) \mod q \\ &= u_i^* y_i^* + h_{3i}^*(x_i^* + z' + sh_{1i}^*) \mod q \\ &= u_i^* y_i^* + h_{3i}^* x_i^* + h_{3i}^* z' + h_{3i}^* sh_{1i}^* \mod q \end{aligned}$$

and

$$\begin{aligned} w_i' &= u_i^* y_i^* + h_{3i}'(x_i^* + d_i^*) \mod q \\ &= u_i^* y_i^* + h_{3i}'(x_i^* + z' + sh_{1i}^*) \mod q \\ &= u_i^* y_i^* + h_{3i}' x_i^* + h_{3i}' z' + h_{3i}' sh_{1i}^* \mod q. \end{aligned}$$

$\mathcal{C}_1$ computes

$$\begin{aligned} W &= (w_i^* - h_{3i}^* x_i^* - h_{3i}^* sh_{1i}^*) - (w_i' - h_{3i}' x_i^* - h_{3i}' sh_{1i}^*) \bmod q \\ &= h_{3i}^* z' - h_{3i}' z' \mod q \end{aligned}$$

and

$$z' = W(h_{3i}^* - h_{3i}')^{-1} \mod q.$$

The probability that $\mathcal{C}_1$ successfully obtains z form (P, zP) is analyzed here. First, we define the following events.
1) *P1:* There is no interruption during the q_{PPK} queries launched by the adversary $\mathcal{A}_1$ to the oracle $\mathcal{O}_{\text{PPK}}$.
2) *P2:* In the *Forgery* phase, $\mathcal{C}_1$ does not abort.
3) *P3:* The challenger $\mathcal{C}_1$ does not abort in the complete game.

In the game procedure, $\mathcal{C}_1$ simulates a real environment to $\mathcal{A}_1$ if there is no interruption. Hence, σ_i^* is a valid signature on $m_i^* || t_i^*$ when the event *P3* occurs. Based on the condition that $\Pr[\xi_i = 1] = \theta$ and $\Pr[\xi_i = 0] = 1 - \theta$, the probability of each event is as follows.
1) $\Pr[\text{P1}] \geq \theta^{q_{\text{PPK}}}$.
2) $\Pr[\text{P2}] \geq 1 - \theta$.
3) Since P1, P2 are independent of each other

$$\begin{aligned} \Pr[\text{P3}] &= \Pr[\text{P1} \wedge \text{P2}] = \Pr[\text{P1}] \Pr[\text{P2}] \\ &\geq \theta^{q_{\text{PPK}}}(1 - \theta) \\ &\geq \frac{1}{e(1 + q_{\text{PPK}})}. \end{aligned}$$

It is worth mentioning that the function $f(x) = x^{q_{\text{PPK}}}(1 - x)$ gets the maximum value $[1/e(1 + q_{\text{PPK}})]$ when $x = [1/(1 + q_{\text{PPK}})]$, where e is the base of natural logarithm. As a result, the probability that $\mathcal{C}_1$ successfully obtains z is

$$\begin{aligned} \Pr[z = z'] &= \Pr[z = z' | \text{P3}] \Pr[\text{P3}] + \Pr[z = z' | \overline{\text{P3}}] \Pr[\overline{\text{P3}}] \\ &= \varepsilon \Pr[\text{P3}] + \frac{1}{q} \Pr[\overline{\text{P3}}] \\ &\geq \frac{\varepsilon}{e(1 + q_{\text{PPK}})}. \end{aligned}$$

Hence, $\mathcal{C}_1$ can solve the ECDLP in G with a nonnegligible probability in polynomial time. ■

Lemma 2: If there exists a PPT Type II adversary $\mathcal{A}_2$ could forge a valid signature after querying $\mathcal{O}_{CU}$ q_{CU} times, querying $\mathcal{O}_{SK}$ q_{SK} times, querying $\mathcal{O}_{RK}$ q_{RK} times, querying $\mathcal{O}_S$ q_S times, querying the random oracles H, H_1, H_2, and H_3, q_H, q_{H_1}, q_{H_2}, q_{H_3}times, respectively, then the ECDLP in G can be solved in polynomial time.

Proof: Given an instance of ECDLP $(P, Q = zP) \in G \times G$, where z is randomly selected from $\mathbb{Z}_q^*$. Suppose the probability that $\mathcal{A}_2$ forges a valid signature successfully is ε, then an algorithm $\mathcal{C}_2$ that can obtain z from (P, Q) in polynomial time is constructed as follows.

Setup: In this phase, $\mathcal{A}_2$ randomly selects $s \in \mathbb{Z}_q^*$ as the master secret key msk, and calculates $P_{\text{pub}} = sP$. Then, $\mathcal{A}_2$ sends $params = \{G, q, P, P_{\text{pub}}\}$ to $\mathcal{C}_2$.

Query: In the query phase, the adversary $\mathcal{A}_2$ is allowed to make queries on the oracles $\mathcal{O}_{CU}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_S$, H, H_1, H_2 and H_3. $\mathcal{C}_2$ responses the queries as follows.

1) The responses of querying the random oracles $H(r_iP_{\text{pub}}, T_i)$, $H_1(\text{ID}_i, R_i, P_{\text{pub}})$, $H_2(m_i, \text{ID}_i, pk_i, t_i, Y_i)$, and $H_3(m_i, \text{ID}_i, pk_i, t_i)$ are the same as in *Lemma* 1.
2) $\mathcal{O}_{CU}(\text{ID}_i)$*:* When $\mathcal{A}_2$ queries this oracle, $\mathcal{C}_2$ finds the record about ID_i in the list $\mathcal{L}_{CU}$, then returns $(\text{ID}_i, pk_i) = (\text{ID}_i, R_i, X_i)$ to $\mathcal{A}_2$. Similar as in *Lemma* 1, if there is no such record, $\mathcal{C}_2$ uses the Coron's skill [34] to pick a number $\xi_i \in \{0, 1\}$. In the case $\xi_i = 1$, $\mathcal{C}_2$ randomly selects $x_i \in \mathbb{Z}_q^*$ and computes $X_i = x_iP$. In another case $\xi_i = 0$, $\mathcal{C}_2$ sets $X_i = Q$. Then, $\mathcal{C}_2$ executes as follows.
   a) Randomly select $r_i \in \mathbb{Z}_q^*$ and compute $R_i = r_iP$.
   b) Query $H(r_iP_{\text{pub}}, T_i)$ to get h_{0i}.
   c) Set $\text{RID}_i = \text{ID}_i$ and compute $\text{ID}_i = \text{RID}_i \oplus h_{0i}$.
   d) Query $H_1(\text{ID}_i, R_i, P_{\text{pub}})$ to obtain h_{1i}.
   e) Compute $d_i = r_i + sh_{1i} \bmod q$.
   f) Record $(\xi_i, \text{ID}_i, \text{RID}_i, r_i, R_i, d_i, x_i, X_i)$ in the list $\mathcal{L}_{CU}$ if $\xi_i = 1$. Otherwise, record $(\xi_i, \text{ID}_i, \text{RID}_i, r_i, R_i, d_i, -, X_i)$ in the list $\mathcal{L}_{CU}$.
   g) Return (ID_i, R_i, X_i) to $\mathcal{A}_2$.
3) $\mathcal{O}_{SK}(\text{ID}_i)$*:* When $\mathcal{A}_2$ queries this oracle, if there is no record about ID_i in $\mathcal{L}_{CU}$ or the corresponding $\xi_i = 0$, $\mathcal{C}_1$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{C}_2$ returns x_i to $\mathcal{A}_2$.
4) $\mathcal{O}_{RK}(\text{ID}_i, pk_i', sk_i')$*:* When $\mathcal{A}_2$ queries this oracle, if there is no record about ID_i in $\mathcal{L}_{CU}$ or the corresponding $\xi_i = 0$, $\mathcal{C}_2$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{C}_2$ replaces (x_i, X_i) in the record with (sk_i', pk_i').
5) $\mathcal{O}_S(m_i, \text{ID}_i)$*:* After receiving the query, if any records about ID_i cannot be found in $\mathcal{L}_{CU}$, $\mathcal{C}_2$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, if $\xi_i = 1$, $\mathcal{C}_1$ returns the result of running $Sign(\text{ID}_i, x_i, d_i, m_i)$ to $\mathcal{A}_2$. In the case $\xi_i = 0$, $\mathcal{C}_2$ executes as follows.
   a) Randomly select $w_i, u_i \in \mathbb{Z}_q^*$.
   b) Query $H_1(\text{ID}_i, R_i, P_{\text{pub}})$ and $H_3(m_i, \text{ID}_i, pk_i, t_i)$ to obtain h_{1i} and h_{3i}, respectively.
   c) Compute $Y_i = u_i^{-1}[w_iP - h_{3i}(Q + R_i + h_{1i}P_{\text{pub}})]$.
   d) If the item $(m_i, \text{ID}_i, pk_i, t_i, Y_i, u_i)$ already exists in the list $\mathcal{L}_2$, $\mathcal{C}_1$ reselects a different $u_i \in \mathbb{Z}_q^*$ and performs step *c*). Otherwise, $\mathcal{C}_2$ adds $(m_i, \text{ID}_i, pk_i, t_i, Y_i, u_i)$ to $\mathcal{L}_2$, and returns (Y_i, w_i) to $\mathcal{A}_2$.

Forgery: In this phase, $\mathcal{A}_2$ chooses a target user with identity ID_i^* and a message $m_i^*||t_i^*$. It is worth mentioning that $\mathcal{A}_2$ is forbidden to query $\mathcal{O}_S(m_i^*, \text{ID}_i^*)$. If there is no record about ID_i^* in the list $\mathcal{L}_{CU}$ or the corresponding $\xi_i^* = 1$, $\mathcal{C}_2$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, $\mathcal{A}_2$ outputs a valid signature $\sigma_i^* = (Y_i^*, w_i^*)$ on $m_i^*||t_i^*$ such that $w_i^*P - u_i^*Y_i^* = h_{3i}^*(Q + R_i^* + h_{1i}^*P_{\text{pub}})$. Then, with the Forking Lemma [35], $\mathcal{A}_2$ outputs another valid signature $\sigma_i' = (Y_i^*, w_i')$ on $m_i^*||t_i^*$ with the same random tape y_i^* and different hash value h_{3i}', i.e.,

$$\begin{aligned} w_i^* &= u_i^*y_i^* + h_{3i}^*(z' + d_i^*) \mod q \\ &= u_i^*y_i^* + h_{3i}^*(z' + r_i^* + sh_{1i}^*) \mod q \\ &= u_i^*y_i^* + h_{3i}^*z' + h_{3i}^*r_i^* + h_{3i}^*sh_{1i}^* \mod q \end{aligned}$$

and

$$\begin{aligned} w_i' &= u_i^*y_i^* + h_{3i}'(z' + d_i^*) \mod q \\ &= u_i^*y_i^* + h_{3i}'(z' + r_i^* + sh_{1i}^*) \mod q \\ &= u_i^*y_i^* + h_{3i}'z' + h_{3i}'r_i^* + h_{3i}'sh_{1i}^* \mod q. \end{aligned}$$

$\mathcal{C}_2$ computes

$$\begin{aligned} W &= (w_i^* - h_{3i}^*r_i^* - h_{3i}^*sh_{1i}^*) - (w_i' - h_{3i}'r_i^* - h_{3i}'sh_{1i}^*) \bmod q \\ &= h_{3i}^*z' - h_{3i}'z' \mod q \end{aligned}$$

and

$$z' = W(h_{3i}^* - h_{3i}')^{-1} \mod q.$$

Similarly, $\mathcal{C}_2$ simulates a real environment to $\mathcal{A}_2$ if there is no interruption during whole game. Furthermore, there are several events defined as follows.

1) *P1:* There is no interruption during the q_{SK} queries launched by the adversary $\mathcal{A}_2$ to the oracle $\mathcal{O}_{SK}$.
2) *P2:* There is no interruption during the q_{RK} queries launched by the adversary $\mathcal{A}_2$ to the oracle $\mathcal{O}_{RK}$.
3) *P3:* The challenger $\mathcal{C}_2$ does not abort during the *Forgery* phase.
4) *P4:* The challenger $\mathcal{C}_2$ does not abort in the complete game.

And the corresponding probabilities are as follows.

1) $\Pr[\text{P1}] \geq \theta^{q_{SK}}$.
2) $\Pr[\text{P2}] \geq \theta^{q_{RK}}$.
3) $\Pr[\text{P3}] \geq 1 - \theta$.
4) The probability of the event P4 occurring is

$$\begin{aligned} \Pr[\text{P4}] &= \Pr[\text{P1} \wedge \text{P2} \wedge \text{P3}] \\ &= \Pr[\text{P1}]\Pr[\text{P2}]\Pr[\text{P3}] \\ &\geq \theta^{q_{SK}+q_{RK}}(1-\theta) \\ &\geq \frac{1}{e(1 + q_{SK} + q_{RK})}. \end{aligned}$$

Hence, the probability that $\mathcal{C}_2$ solves the ECDLP in G is

$$\begin{aligned} \Pr[z = z'] &= \Pr[z = z'|\text{P4}]\Pr[\text{P4}] + \Pr[z = z'|\overline{\text{P4}}]\Pr[\overline{\text{P4}}] \\ &= \varepsilon \Pr[\text{P4}] + \frac{1}{q}\Pr[\overline{\text{P4}}] \\ &\geq \frac{\varepsilon}{e(1 + q_{SK} + q_{RK})} \end{aligned}$$

which is nonnegligible. ■

Theorem 2: In the random oracle model, the aggregate signature in our improved CLAS scheme is CMA secure under the assumption that the ECDLP in G is intractable.

Proof: This theorem is derived from the *Lemmas* 3 and 4. ■

Lemma 3: If there exists a PPT Type I adversary $\mathcal{A}_3$ could forge a valid aggregate signature, the ECDLP in G can be solved in polynomial time. Suppose that $\mathcal{A}_3$ queries $\mathcal{O}_{CU}$ q_{CU} times, queries $\mathcal{O}_{SK}$ q_{SK} times, queries $\mathcal{O}_{\text{PPK}}$ q_{PPK} times,

queries $\mathcal{O}_{RK}$ q_{RK} times, queries $\mathcal{O}_S$ q_S times, queries the random oracles H, H_1, H_2, and H_3, q_H, q_{H_1}, q_{H_2}, q_{H_3} times, respectively,

Proof: Given an instance of ECDLP $(P, Q = zP) \in G \times G$, where z is randomly selected from $\mathbb{Z}_q^*$. Suppose the probability that $\mathcal{A}_3$ forges a valid aggregate signature successfully is ε, then an algorithm $\mathcal{C}_3$ that can obtain z from (P, Q) in polynomial time is constructed as follows.

The *Setup* and *Query* phases are the same as described in *Lemma* 1.

Forgery: $\mathcal{A}_3$ chooses a target set of users with identities $\{\mathrm{ID}_1^*, \mathrm{ID}_2^*, \ldots, \mathrm{ID}_n^*\}$ and messages $\{m_1^*||t_1^*, m_2^*||t_2^*, \ldots, m_n^*||t_n^*\}$. Each ID_i^* should be found in the list $\mathcal{L}_{CU}$. If all $\xi_i^* = 1$, $\mathcal{C}_3$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, by using the Forking Lemma [35], $\mathcal{A}_3$ outputs two valid aggregate signatures $\sigma^* = (U^*, w^*)$ and $\sigma' = (U^*, w')$. Without loss of generality, it is assumed that ξ_1^* corresponding to ID_1^* is 0. Then, it has

$$w^* = \sum_{i=2}^{n} w_i^* + w_1^* = \sum_{i=2}^{n} w_i^* + u_1^* y_1^* + h_{31}^* x_1^* + h_{31}^* z' + h_{31}^* s h_{11}^* \mod q$$

and

$$w' = \sum_{i=2}^{n} w_i^* + w_1' = \sum_{i=2}^{n} w_i^* + u_1^* y_1^* + h_{31}' x_1^* + h_{31}' z' + h_{31}' s h_{11}^* \mod q.$$

Hence, $\mathcal{C}_3$ obtain z' by calculating

$$W = (w^* - h_{31}^* x_1^* - h_{31}^* s h_{11}^*) - (w' - h_{31}' x_1^* - h_{31}' s h_{11}^*) \mod q$$

and

$$z' = W(h_{31}^* - h_{31}')^{-1} \mod q.$$

The probability that there is no interruption in the *Forgery* phase is $(1 - \theta^n)$. Hence, as analyzed in *Lemma* 1, the probability that $\mathcal{C}_3$ successfully solves the ECDLP in G is larger than $\varepsilon\theta^{q_{\mathrm{PPK}}}(1 - \theta^n)$. ■

Lemma 4: If there exists a PPT Type II adversary $\mathcal{A}_4$ could forge a valid aggregate signature after querying $\mathcal{O}_{CU}$ q_{CU} times, querying $\mathcal{O}_{SK}$ q_{SK} times, querying $\mathcal{O}_{RK}$ q_{RK} times, querying $\mathcal{O}_S$ q_S times, querying the random oracles H, H_1, H_2 and H_3, q_H, q_{H_1}, q_{H_2}, q_{H_3}times, respectively, then the ECDLP in G can be solved in polynomial time.

Proof: Similarly, Given an instance of ECDLP $(P, Q = zP) \in G \times G$, where z is randomly selected from $\mathbb{Z}_q^*$. Suppose the probability that $\mathcal{A}_4$ forges a valid signature successfully is ε, then an algorithm $\mathcal{C}_4$ that can obtain z from (P, Q) in polynomial time is constructed as follows.

The *Setup* and *Query* phases are the same as described in *Lemma* 2.

Forgery: $\mathcal{A}_4$ chooses a target set of users with identities $\{\mathrm{ID}_1^*, \mathrm{ID}_2^*, \ldots, \mathrm{ID}_n^*\}$ and messages $\{m_1^*||t_1^*, m_2^*||t_2^*, \ldots, m_n^*||t_n^*\}$. All identities $\{\mathrm{ID}_i^*, i = 1, \ldots, n\}$ should be found in the list $\mathcal{L}_{CU}$. If all $\xi_i^* = 1$, $\mathcal{C}_4$ returns a random number $z' \in \mathbb{Z}_q^*$ and aborts. Otherwise, without loss of generality, it is assumed that ξ_1^* corresponding to ID_1^* is 0. Then, $\mathcal{A}_4$ outputs two valid aggregate signatures $\sigma^* = (U^*, w^*)$ and $\sigma' = (U^*, w')$ with the Forking Lemma [35]

$$w^* = \sum_{i=2}^{n} w_i^* + w_1^* = \sum_{i=2}^{n} w_i^* + u_1^* y_1^* + h_{31}^* z' + h_{31}^* r_1^* + h_{31}^* s h_{11}^* \mod q$$

and

$$w' = \sum_{i=2}^{n} w_i^* + w_1' = \sum_{i=2}^{n} w_i^* + u_1^* y_1^* + h_{31}' z' + h_{31}' r_1^* + h_{31}' s h_{11}^* \mod q.$$

After receiving σ^* and σ', $\mathcal{C}_4$ computes

$$W = (w_i^* - h_{3i}^* r_i^* - h_{3i}^* s h_{1i}^*) - (w_i' - h_{3i}' r_i^* - h_{3i}' s h_{1i}^*) \mod q$$

and

$$z' = W(h_{3i}^* - h_{3i}')^{-1} \mod q.$$

Hence, $\mathcal{C}_4$ could solve the ECDLP with a probability of more than $\varepsilon\theta^{q_{SK}+q_{RK}}(1 - \theta^n)$. ■

According to *Theorems* 1 and 2, our improved CLAS scheme is CMA secure under the ECDLP assumption in the random oracle model. Hence, the security requirements, including *MessageAuthentication*, *MessageIntegrity*, and *Nonrepudiation* can be achieved directly. In addition, the improved CLAS scheme also satisfies *Anoymity* and *Traceability*.

For *Anoymity* and *Traceability*, a pseudo identity ID_i is adopted to prevent the information of the real identity RID_i from being obtained by other sensor nodes or external attackers. Furthermore, the pseudo identity is calculated as $R_i = r_i P$, $\mathrm{ID}_i = \mathrm{RID}_i \oplus H(r_i P_{\mathrm{pub}}, T_i)$. Hence, $H(r_i P_{\mathrm{pub}}, T_i)$ cannot be obtained without r_i and s. However, r_i is abandoned after calculation and s is held in secret by MS. Other sensor nodes and external attackers cannot know the real identity of the sensor node, which guarantees *Anoymity*. MS could trace the real identity with the master secret key if it is necessary, which guarantees *Traceability*.

VII. Performance Analysis

In this section, we analyze the computational and communication costs of our improved CLAS scheme. The results of comparison with related works are given to show the efficiency of our proposal.

A. Computational Cost

In this section, we use the results of the single operation execution time measured by Liu *et al*. to analyze the computational costs of the algorithms *Sign*, *Verify*, *Aggregate*, and *AggregateVerify* in our scheme, respectively. The results are given in Table I and Fig. 2.

TABLE I
COMPARISON WITH RELATED WORK IN TERMS OF COMPUTATIONAL COST

Scheme	Single Sign Cost	Single Verify Cost	Aggregate and AggregateVerify Cost
Zhang *et al.*'s [10]	$2T_{mg}+3T_{bpsm}+2T_{bppa}$ =14.910142ms	$4T_{bp}+T_{mg}$=17.906854ms	$(n+1)T_{mg}+(n+3)T_{bp}+3(n-1)T_{bppa}$
Li *et al.*'s [7]	$2T_{mz}+T_{mg}+5T_{bpsm}+3T_{bppa}$ =23.078915ms	$2T_{mz}+3T_{bp}+2T_{bpsm}+$ $2T_{bppa}$=23.078915ms	$2nT_{mz}+3T_{bp}+2nT_{bpsm}+(5n-3)T_{bppa}$
Xie *et al.*'s [9]	$2T_{mz}+T_{ecsm}$=0.168785ms	$3T_{mz}+4T_{ecsm}+3T_{ecpa}$ =0.670432ms	$(3n+1)T_{mz}+(5n+5)T_{ecsm}+(6n-3)T_{ecpa}$
Du *et al.*'s [8]	$2T_{mz}+T_{ecsm}$=0.168785ms	$3T_{mz}+4T_{ecsm}+3T_{ecpa}$ =0.670432ms	$3nT_{mz}+(3n+1)T_{ecsm}+(4n-1)T_{ecpa}$
Our Scheme	$2T_{mz}+T_{ecsm}$=0.168785ms	$3T_{mz}+4T_{ecsm}+3T_{ecpa}$ =0.670432ms	$3nT_{mz}+(3n+1)T_{ecsm}+(4n-1)T_{ecpa}$

TABLE II
EXECUTION TIME OF SINGLE OPERATION

Notaion	Description	Running time(ms)
T_{bp}	Bilinear pairing	4.441043
T_{bpsm}	Bilinear pairing scalar multiplication	4.87256
T_{bppa}	Bilinear pairing point addition	0.003549
T_{ecsm}	ECC scalar multiplication	0.165217
T_{ecpa}	ECC point addition	0.001404
T_{mg}	map to G hash	0.142682
T_{mz}	map to $\mathbb{Z}_q^*$ hash	0.001784

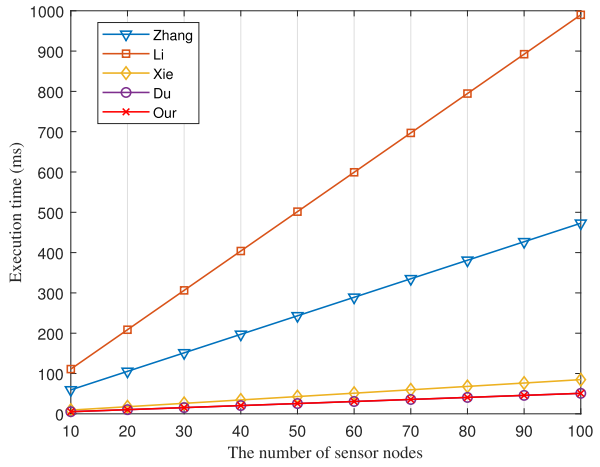


Fig. 2. Comparison with related work in terms of verification efficiency.

Liu *et al.*'s measurement results [6] are given in Table II. The situation they considered in bilinear pairing-based schemes is a super singular elliptic curve. Specifically, the elliptic curve $E_1 : y^2 = x^3 + x \bmod p_1$ is over the finite field $\mathbb{F}_{q_1}$. The bit length of q_1 is 512 b. The bit length of the element in the elliptic curve group G_1 based on E_1 is 1024 b. For pairing-free schemes, they set a group G based on the Koblitz elliptic curve $E : y^2 = x^3 + ax + b \bmod p$ over a finite field $\mathbb{Z}_q$. The bit length of q and the element in G are set to 160 b and 320 b, respectively, to achieve the 1024 b RSA security level.

We compare the computational costs of our proposal with the related works that are still secure, including Zhang and Zhang scheme [10], Li *et al.*'s scheme [7], Xie *et al.*'s scheme [9], and Du *et al.*'s scheme [8]. The comparison results are shown in Table I. The parameter n denotes the number of MSNs. To generate a single signature on a message, our scheme needs one ECC scalar multiplication operation and two map-to-$\mathbb{Z}_q^*$ hash operations. The addition and multiplication over $\mathbb{Z}_q^*$ are too fast to ignore their time. Hence, the execution time of signing a message is $2T_{mz} + T_{ecsm} = 2 \times 0.001784 + 0.165217 = 0.168785$ *ms*. In the same way, the running time of *Verify* is $3T_{mz}+4T_{ecsm}+3T_{ecpa} = 0.670432$ *ms*. We mainly consider the computational costs of resource-limited sensor nodes. For CH and MS with strong computing power, we combine the time spent in *Aggregate* and *AggregateVerify* to compare with others, and the results are given in Fig. 2 for a more intuitive display. Hence, as Table I and Fig. 2 show, our scheme supports the rapid calculation of individual signatures and verification of aggregate signatures.

TABLE III
COMPARISON WITH RELATED WORK IN TERMS OF COMMUNICATION COST

Scheme	System Type	Single Signature	Aggregate Signature
Zhang[10]	Bilinear Pairing	$2\|G_1\|=2048$ *bits*	$(n+1)\|G_1\|$ $=(n+1)2048$ *bits*
Li[7]	Bilinear Pairing	$2\|G_1\|=2048$ *bits*	$(n+1)\|G_1\|$ $=(n+1)2048$ *bits*
Xie[9]	ECC	$\|G\|+\|\mathbb{Z}_q^*\|=480$ *bits*	$n\|G\|+\|\mathbb{Z}_q^*\|$ $=320n+160$ *bits*
Du[8]	ECC	$\|G\|+\|\mathbb{Z}_q^*\|=480$ *bits*	$n\|G\|+\|\mathbb{Z}_q^*\|$ $=320n+160$ *bits*
Our Scheme	ECC	$\|G\|+\|\mathbb{Z}_q^*\|=480$ *bits*	$\|G\|+\|\mathbb{Z}_q^*\|=480$ *bits*

B. Communication Cost

The elliptic curve group G we considered is over Koblitz elliptic curve on $\mathbb{Z}_q^*$. In particular, the bit length of an element of G is 320 b, and the bit length of q is 160 b. The single signature of our scheme is in the form of $(Y_i, w_i) \in G \times \mathbb{Z}_q^*$. Hence, each sensor node only sends a 480 b message to the CH. Compared with the schemes based on the bilinear pairing whose single signature is 2048 b long, our scheme saves a lot of communication costs for resource-limited sensor nodes. In addition, the aggregate signature in our scheme is in the form of $(U, w) \in G \times \mathbb{Z}_q^*$ whose bit length is a constant value, 480 b. Table III shows the comparison results of communication costs among our scheme with related works. The communication complexity of our scheme is $O(1)$, and the communication complexity of the schemes [7]–[9] and [10] are $O(n)$. It is clear that the communication overheads of the proposed scheme is much lower than related schemes.

Comprehensive the computational and communication costs, our scheme is more efficient than [7]–[9] and [10].

VIII. Conclusion

CLAS can provide an efficient message authentication function for HWMSN based on IoT. In this article, we have analyzed the security of Liu *et al.*'s pair-free CLAS scheme, and given the specific attack algorithm. To fix security holes in their scheme, we have modified the sign algorithm. The security analyses indicate that our improved CLAS scheme is secure against both Type I and Type II adversaries. The improved scheme is still based on ECC cryptosystem which makes the signing faster and the signature length shorter. In particular, the length of the aggregate signature in the proposed scheme is fixed, which greatly reduce the communication resources costs. The results of comparison with related works show that our scheme is more effective in practice.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] M. R. Yuce, S. W. Ng, N. L. Myo, J. Y. Khan, and W. Liu, "Wireless body sensor network using medical implant band," *J. Med. Syst.*, vol. 31, no. 6, pp. 467–474, 2007.

[3] W. Mao, *Modern Cryptography: Theory and Practice*. Upper Saddle River, NJ. USA: Prentice Hall PTR, 2003.

[4] R. Castro and R. Dahab, "Efficient certificateless signatures suitable for aggregation," IACR Cryptol. ePrint Archive, Lyon, France, Rep. 2007/454, 2007.

[5] N. B. Gayathri, G. Thumbur, P. R. Kumar, M. Z. U. Rahman, P. V. Reddy, and A. Layekuakille, "Efficient and secure pairing-free certificateless aggregate signature scheme for healthcare wireless medical sensor networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 9064–9075, Oct. 2019.

[6] J. Liu, L. Wang, and Y. Yu, "Improved security of a pairing-free certificateless aggregate signature in healthcare wireless medical sensor networks," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5256–5266, Jun. 2020.

[7] J. Li, H. Yuan, and Y. Zhang, "Cryptanalysis and improvement for certificateless aggregate signature," *Fundamenta Informaticae*, vol. 157, no. 2, pp. 111–123, 2018.

[8] H. Du, Q. Wen, and S. Zhang, "An efficient certificateless aggregate signature scheme without pairings for healthcare wireless sensor network," *IEEE Access*, vol. 7, pp. 42683–42693, 2019.

[9] Y. Xie, X. Li, S. Zhang, and Y. Li, "*iCLAS:* An improved certificateless aggregate signature scheme for healthcare wireless sensor networks," *IEEE Access*, vol. 7, pp. 15170–15182, 2019.

[10] L. Zhang and F. Zhang, "A new certificateless aggregate signature scheme," *Comput. Commun.*, vol. 32, no. 6, pp. 1079–1085, 2009.

[11] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. 22nd Int. Conf. Theory Appl. Cryptograph. Techn.*, 2003, pp. 416–432.

[12] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Security*, 2003, pp. 452–473.

[13] Z. Gong, Y. Long, X. Hong, and K. Chen, "Two certificateless aggregate signatures from bilinear maps," in *Proc. IEEE 8th ACIS Int. Conf. Softw. Eng. Artif. Intell. Netw. Parallel/Distrib. Comput. (SNPD)*, vol. 3.Qingdao, China, 2007, pp. 188–193.

[14] H. Tu, D. He, and B. Huang, "Reattack of a certificateless aggregate signature scheme with constant pairing computations," *Sci. World J.*, vol. 2014, Mar. 2014, Art. no. 343715.

[15] L. Cheng, Q. Wen, Z. Jin, H. Zhang, and L. Zhou, "Cryptanalysis and improvement of a certificateless aggregate signature scheme," *Inf. Sci.*, vol. 295, pp. 337–346, Feb. 2015.

[16] D. He, M. Tian, and J. Chen, "Insecurity of an efficient certificateless aggregate signature with constant pairing computations," *Inf. Sci.*, vol. 268, pp. 458–462, Jun. 2014.

[17] J. Zhang, X. Zhao, and J. Mao, "Attack on Chen *et al.*'s certificateless aggregate signature scheme," *Security Commun. Netw.*, vol. 9, no. 1, pp. 54–59, 2016.

[18] Y. Chen, R. Tso, M. Mambo, K. Huang, and G. Horng, "Certificateless aggregate signature with efficient verification," *Security Commun. Netw.*, vol. 8, no. 13, pp. 2232–2243, 2015.

[19] H. Liu, M. Liang, and H. Sun, "A secure and efficient certificateless aggregate signature scheme," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. 97, no. 4, pp. 991–995, 2014.

[20] A. K. Malhi and S. Batra, "An efficient certificateless aggregate signature scheme for vehicular ad-hoc networks," *Discrete Math. Theor. Comput. Sci.*, vol. 17, no. 1, pp. 317–338, 2015.

[21] P. Kumar and V. Sharma, "On the security of certificateless aggregate signature scheme in vehicular ad hoc networks," *Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing*, vol. 583. Singapore: Springer, 2018, pp. 715–722.

[22] X. Yang, C. Chen, T. Ma, Y. Li, and C. Wang, "An improved certificateless aggregate signature scheme for vehicular ad-hoc networks," in *Proc. IEEE 3rd Adv. Inf. Technol. Electron. Autom. Control Conf. (IAEAC)*, Chongqing, China, 2018, pp. 2334–2338.

[23] I. A. Kamil and S. O. Ogundoyin, "An improved certificateless aggregate signature scheme without bilinear pairings for vehicular ad hoc networks," *J. Inf. Security Appl.*, vol. 44, pp. 184–200, Feb. 2019.

[24] J. Cui, J. Zhang, H. Zhong, R. Shi, and Y. Xu, "An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks," *Inf. Sci.*, vols. 451–452, pp. 1–15, Jul. 2018.

[25] P. Kumar, S. Kumari, V. Sharma, A. K. Sangaiah, J. Wei, and X. Li, "A certificateless aggregate signature scheme for healthcare wireless sensor network," *Sustain. Comput. Informat. Syst.*, vol. 18, pp. 80–89, Jun. 2018.

[26] L. Shen, J. Ma, X. Liu, and M. Miao, "A provably secure aggregate signature scheme for healthcare wireless sensor networks," *J. Med. Syst.*, vol. 40, no. 11, pp. 1–10, 2016.

[27] L. Wu, Z. Xu, D. He, and X. Wang, "New certificateless aggregate signature scheme for healthcare multimedia social network on cloud environment," *Security Commun. Netw.*, vol. 2018, Jun. 2018, Art. no. 2595273.

[28] Y. Zhan and B. Wang, "Cryptanalysis of a certificateless aggregate signature scheme for healthcare wireless sensor network," *Security Commun. Netw.*, vol. 2019, Jun. 2019, Art. no. 6059834.

[29] Y. Qu and Q. Mu, "An efficient certificateless aggregate signature without pairing," *Int. J. Electron. Security Digit. Forensics*, vol. 10, no. 2, pp. 188–203, 2018.

[30] S. Xu, Y. Li, R. H. Deng, Y. Zhang, X. Luo, and X. Liu, "Lightweight and expressive fine-grained access control for healthcare Internet-of-Things," *IEEE Trans. Cloud Comput.*, early access, Aug. 20, 2019, doi: 10.1109/TCC.2019.2936481.

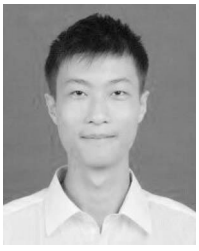
[31] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comput.*, vol. 48, no. 177, pp. 203–209, 1987.

[32] V. S. Miller, "Use of elliptic curves in cryptography," in *Proc. Conf. Theory Appl. Cryptograph. Techn.*, 1985, pp. 417–426.

[33] H. Xiong, Z. Guan, Z. Chen, and F. Li, "An efficient certificateless aggregate signature with constant pairing computations," *Inf. Sci.*, vol. 219, pp. 225–235, Jan. 2013.

[34] J. S. Coron, "On the exact security of full domain hash," in *Proc. 20th Annu. Adv. Cryptol. (Crypto)*, vol. 1880, 2000, pp. 229–235.

[35] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *J. Cryptol.*, vol. 13, no. 3, pp. 361–396, 2000.



Yu Zhan received the B.Sc. degree from Chang'an University, Xi'an, China, in 2015. He is currently pursuing the Ph.D. degree in cryptography with the School of Telecommunications Engineering, Xidian University, Xi'an.

His main research interests include public key cryptography and cryptanalysis.

Baocang Wang received the B.S. and M.S. degrees in mathematics and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2001, 2004, and 2006, respectively.

He is a Professor with the School of Telecommunications Engineering, Xidian University. His main research interests include public key cryptography, wireless network security, and data mining.

Rongxing Lu (Senior Member, IEEE) received the Ph.D. degree (awarded the Canada Governor General's Gold Medal) from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He also worked as an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from April 2013 to August 2016. He is an Associate Professor with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Fredericton, NB, Canada. He worked as a Postdoctoral Fellow with the University of Waterloo, from May 2012 to April 2013. His research interests include applied cryptography, privacy enhancing technologies, and IoT-big data security and privacy. He has published extensively in his areas of expertise (with citation over 17 600 and H-index 66 from Google Scholar in November 2019).

Dr. Lu won the 8th IEEE Communications Society Asia-Pacific Outstanding Young Researcher Award in 2013. He was a recipient of nine best (student) paper awards from some reputable journals and conferences. He currently serves as the Vice-Chair (Publication) of IEEE Communications and Information Security Technical Committee. He is the Winner of Excellence in Teaching Award, FCS, UNB from 2016 to 2017. He is currently a Senior Member of IEEE Communications Society.