New Efficient Certificateless Aggregate Signature Scheme

Zhiyan Xu[1,2], Libing Wu[1], Yongjun Ren[3], Debiao He[1]

[1] Computer School, Wuhan University, China
[2] College of Computer, Hubei University of Education, China
[3] Nanjing University of Information Science and Technology, China
csxzy@whu.edu.cn, wu@whu.edu.cn, renyj100@126.com, hedebiao@163.com

Abstract

Certificateless aggregate signature (CLAS) scheme has been widely adopted in resource-constrained wireless mobile networks, because it not only solves the problems of the certificate management and private key escrow, but also achieves batch validation. In this paper, we first analyze the security for Kang et al.'s CLAS scheme which was claimed to be secure against various types of attacks. Unfortunately, we find that their scheme can not resist malicious KGC's attack and present a concrete attack method. To enhance security, we propose a new certificateless aggregate signature scheme. Furthermore, we formally prove the security of the scheme in the random oracle model. Finally, we evaluate the performance of our proposed scheme. Compared with the original scheme, our new scheme fixes the security flaw, while reducing the total computational costs, so that is more practical and adoptable in real life.

Keywords: Cryptanalysis, Signature forgery attack, Provable security, Certificateless aggregate signature

1 Introduction

Digital signature is of vital importance in terms of ensuring the integrity of the network messages and the legitimacy of clients' identities to prevent messages repudiation or illegal service requests.

For example, in vehicular networks, each node (ie., the vehicles, RSUs, OBUs) needs to authenticate the identities of all other nodes before sharing its information [1]. And in wireless roaming authentication, to prevent illegal requests of roaming service, the foreign agent server (FAS) needs to verify the legitimacy of the messages provided by the mobile nodes [2-3]. In such scenarios, the number of signatures increases as the number of nodes grows, so that the message verifier has to take a lot of computation overhead if it verifies the single signature one by one.

To address the aforementioned problems, Boneh et al. [4] first introduced the concept for aggregate signature (AS), the aggregator can aggregate n different signatures with respect to n messages from n users into a single short signature. The validity of each single signature involved in the aggregation is guaranteed by verifying the validity of aggregate signature. Because the aggregate signature can reduce the length of the signature, realize batch validation and reduce the validation overhead, it is more suitable for batch identification problem in resource-constrained wireless networks.

Because traditional PKI-based public key cryptography (PKI) and identity-based cryptography (IBC) suffer from inherent certificate management and key escrow problems. Certificateless public key cryptography (CL-PKC) was introduced to solve the aforementioned two problems, since the user's private key is generated by key generation center (KGC) and himself in CL-PKC.

Most recently, an efficient certificateless aggregate signature scheme was proposed by Kang et al. [5]. And they claimed their signature scheme was secure against the two typical adversaries in certificateless aggregate signature under the assumption of computational Diffie-Hellman problem (CDH). However, in this paper, we reveal that their scheme is vulnerable to the signature forgery attack. We also present a new efficient certificateless aggregate signature scheme and formally prove the security of the proposed scheme under the CDH assumption in the random oracle model. We then evaluate the performance of our proposed scheme, and demonstrate that our proposal is more practical in real-world applications in comparison to Kang et al.'s scheme.

1.1 Our Research Contributions

In this paper, we put forward a secure and efficient certificateless aggregate signature scheme which yields better efficiency and security for solving batch identification problem in resource-constrained wireless networks. The contributions of this paper can be summarized as follows:

Firstly, we identify a security weakness and present a

*Corresponding Author: Debiao He; E-mail: hedebiao@163.com
DOI: 10.3966/160792642018121907006

concrete method of the signature forgery attack against Kang et al.'s certificateless aggregate signature (CLAS) scheme.

Secondly, we propose a novel CLAS scheme to fix this security flaw. In other words, our scheme can satisfy the security requirements.

Finally, we prove the security of our proposed CLAS scheme and show it can improve the security with lower computation overhead compared with Kang et al.'s CLAS scheme.

1.2 Organization of the Paper

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 gives some preliminaries related to our paper, and then the brief review of Kang et al.'s CLAS scheme is presented in Section 4. In Section 5, we demonstrate an attack against Kang et al.'s CLAS scheme. Furthermore, we present details of the proposed CLAS scheme in Section 6. In Section 7 and Section 8, the security proof and the performance analysis are followed. Finally, we give some concluding remarks in the last Section.

2 Related Work

In traditional PKI-based public key cryptography, each user first needs to choose a key pair (sk, pk) and then sends the public key pk to a trusted certificate authority (CA) for a valid certificate [6-7]. However, with the increasing of the number of users, PKC faces all kinds of problems for certificate management, such as distribution, storage, revocation and high computational overhead and so on. Although the identity-based cryptography can solve the certificate management problem in PKI [8], it is still not secure enough for communications or authentications. Because the task of private key generation is delegated to the key generation center according to users' identities (ie., emails, telephone numbers), the KGC is able to access all users' private keys. Therefore, the IBC suffers from inherent key escrow problem.

To address the above problems, a new paradigm called certificateless public key cryptography (CL-PKC) [9] was put forward by Al -Riyami and Paterson. In CL-PKC, the user does not need to apply for a certificate, and the user's private key is composed of his partial private key and a secret value, whereas the former is generated by KGC and the latter is chosen by the user himself. Thus the certificate management problem in PKI and the key escrow problem in IBC can be solved through this way. Since CL-PKC was first introduced, it has attracted more and more research attention.

Huang et al. [10] showed that the certificateless signature scheme proposed in [9] suffered from the public key replacement attack, and proposed an improved certificateless signature scheme to solve this weakness. Similarly, Li et al. [11] also proposed a new certificateless signature scheme to improve the security of [12], which is subject to the public key replacement attack as well.

As for the malicious KGC attack existing in some certificateless signature schemes, Au et al. [13] proposed an enhanced security model, where a malicious KGC is allowed to produce the key pair in any way. However, some certificateless encryption and signature schemes proposed in [14-16] have been found to be insecure against malicious-but-passive KGC attack.

Combining certificateless public key cryptography with aggregate signature, Castro et al. [17] first produced the concept for certificates aggregate signature (CLAS). Since then, CLAS schemes [18-25] have attracted much research attention. Gong et al. [18] proposed the first CLAS scheme, but they did not give a formal security proof to the scheme. Zhang and Zhang [19] redefined the concept and security model for CLAS. Furthermore, they put forward a new CLAS scheme, but their scheme has been proved that it cannot resist the malicious KGC attack. Xiong et al. [20] presented a CLAS scheme, but He et al. [21] showed that their scheme was forgeable, and proposed an improved CLAS scheme. The CLAS scheme proposed in [22] has been shown to be insecure against the malicious-but-passive KGC attack by the researchers in [23-25].

3 Preliminaries

In this section, we give a brief description of the bilinear map and the complexity assumption.

3.1 Bilinear Map

Let G_1 and G_2 are two cyclic groups with the same order q, where q is a prime. $e: G_1 \times G_1 \to G_2$ is a billinear map. For all $P, Q \in G_1$, $a, b \in Z_q^*$, e should satisfy the properties as follows.

(1) Bilinear: $e(aP, bQ) = e(P, Q)^{ab}$.

(2) Non-degeneracy: There exists $P, Q \in G_1$ such that $e(P, Q) \neq 1_{G_2}$.

(3) Computable: There exists efficient algorithm to compute $e(P, Q)$.

3.2 Complexity Assumption

Computational Diffie-Hellman (CDH) Problem: Given a generator P of an additive cyclic group G_1 with the order q, and a tuple (aP, bP) to compute abP, where $a, b \in Z_q^*$ are both unknown.

4 Review of Kang et al.' s CLAS Scheme

In this section, we review Kang *et al.*'s CLAS scheme, which is composed of six algorithms as follows.

Setup. KGC performs the following steps to complete the system initialization.

(1) Given a security parameter λ, choose two cyclic groups, G_1 and G_2 with the order q, where q is a prime. A billinear map $e: G_1 \times G_1 \to G_2$. P is a generator of G_1.

(2) Choose a random value $s \in Z_q^*$ as the master key and compute $P_{pub} = s \cdot P$ as the system public key.

(3) Define four hash functions: $H_1, H_3, H_4 : \{0,1\}^* \to G_1$ and $H_2 : \{0,1\}^* \times \{0,1\}^* \times G_1 \times G_1 \to Z_q^*$.

(4) Let $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ as the system parameter.

Partial-Private-Key-Extract. KGC performs the following steps to generate the partial private key.

(1) Input ID_i as the identity of user U_i.

(2) Compute $Q_i = H_1(ID_i)$ and $D_i = s \cdot Q_i$.

(3) Output D_i as the partial private key of user U_i.

User-Pubic-Key-Gen. The user U_i performs the following steps to generate the user public key.

(1) Input ID_i as the identity of user U_i.

(2) Choose a random value $x_i \in Z_q^*$ as the secret value.

(3) Compute $P_i = x_i \cdot P$ as the public key of U_i.

Single-Sign. The signer performs the following steps to generate the signature of the message m_i from the user U_i.

(1) Input ID_i as the identity of user U_i, public key P_i, message m_i and a state information w.

(2) Choose a random value $r_i \in Z_q^*$.

(3) Compute $R_i = r_i \cdot P$, $h_i = H_2(m_i, ID_i, P_i, R_i)$, $Z = H_3(w)$ and $F = H_4(w)$.

(4) Compute $T_i = h_i D_i + x_i Z + r_i F$.

(5) Output $\sigma_i = (R_i, T_i)$ as the signature of the message m_i from the user U_i.

Aggregate-Sign. The aggregator performs the following steps to generate the aggregate signature with respect to n tuples $(U_i, m_i, P_i, \sigma_i)_{1 \le i \le n}$.

(1) Input n tuples $(U_i, m_i, P_i, \sigma_i)$, where $1 \le i \le n$.

(2) Compute $T = \sum_{i=1}^{n} T_i$.

(3) Output $\sigma = (R, T)$ as the aggregate signature, where $R = \{R_1, R_2, \cdots R_n\}$.

Aggregate Verify. The verifier performs the following steps to verify the aggregate signature $\sigma = (R, T)$.

(1) Input n tuples $(U_i, m_i, P_i, \sigma_i)$, and aggregate signature $\sigma = (R, T)$, where $1 \le i \le n$.

(2) For $1 \le i \le n$, compute $Q_i = H_1(ID_i)$ and $h_i = H_2(m_i, ID_i, P_i, R_i)$.

(3) Compute $Z = H_3(w)$ and $F = H_4(w)$.

(4) Verify

$$e(T, P) = e(P_{pub}, \sum_{i=1}^{n} h_i Q_i) e(Z, \sum_{i=1}^{n} P_i) e(F, \sum_{i=1}^{n} R_i) \quad \textbf{(1)}$$

(5) If *Equ*(1) holds, output true; Otherwise, output false.

5 Vulnerability of Kang et al.' s CLAS Scheme

We present a concrete attack method to demonstrate that the signature Kang et al.' S CLAS scheme is forgeable. We mainly consider the type II adversary, namely $A = A_2$, then A needs to satisfy two conditions of the game defined in section 7.2. The attack details are described as follows.

Setup. The challenger C runs *Setup* algorithm to generate system parameters *params* and master key s. Then C returns *params* and s to the adversary A.

Queries. The adversary A could get the signature σ_i on m_i signed by U_i with the identity ID_i via *signature queries*, and get the value h_i, Z, F via *hash queries,* where

$$\sigma_i = \begin{cases} R_i = r_i P \\ T_i = h_i D_i + x_i Z + r_i F \end{cases} \quad \textbf{(2)}$$

Forgery. In order to forge the signature σ_j on m_j signed by U_i with the identity ID_i, the adversary A implements its attack as follows.

(1) The adversary A first set $R_j^* = r_j^* P = R_i = r_i P$. Owing to the openness of message m_j, the signer U_i, the user's public key P_i, system public key P_{pub}, then the adversary A can calculate $Q_i = H_1(ID_i)$, $D_i = sQ_i$ and $h_j^* = H_2(m_j, ID_i, P_i, R_j^*)$.

(2) The adversary A can get $x_i Z + r_i F = T_i - h_i D_i$ from the equations (2).

(3) The adversary A can calculate

$$\begin{aligned} T_j^* &= h_j^* D_i + x_i Z + r_j^* F \\ &= h_j^* D_i + x_i Z + r_i F \\ &= h_j^* D_i + T_i - h_i D_i \end{aligned} \quad \textbf{(3)}$$

Namely, the forged signature is σ_j^*, where

$$\sigma_j^* = \begin{cases} R_j^* = r_i P \\ T_j^* = h_j^* D_i + x_i Z + r_i F \end{cases} \quad (4)$$

Verify. It is easy to verify the validity of the forged signature. For $1 \le j \le n$, the verifier calculates $Q_i = H_1(ID_i)$, $h_j^* = H_2(m_j, ID_i, P_i, R_j^*)$. Furthermore, the verifier calculates $Z = H_3(w)$, $F = H_4(w)$. Then we use signature σ_j^* to verify equation (5), the concrete process is as follows.

$$\begin{aligned} & e(T^*, P) = e(\sum_{j=1}^{n} T_j^*, P) \\ & = e(\sum_{j=1}^{n} (h_j^* D_i + x_i Z + r_i F), P) \\ & = e(\sum_{j=1}^{n} h_j^* D_i, P) \cdot e(\sum_{j=1}^{n} x_i Z, P) \cdot e(\sum_{j=1}^{n} r_i F, P) \\ & = e(\sum_{j=1}^{n} h_j^* Q_i, P_{pub}) \cdot e(\sum_{j=1}^{n} P_i, Z) \cdot e(\sum_{j=1}^{n} R_i, F) \\ & = e(P_{pub}, \sum_{j=1}^{n} h_j^* Q_i) \cdot e(Z, \sum_{j=1}^{n} P_i) \cdot e(F, \sum_{j=1}^{n} R_i) \end{aligned} \quad (5)$$

We can find the equation (5) holds. The forged signature pass verification, Kang *et al.*'s CLAS scheme is insecure.

6 The Proposed CLAS Scheme

To overcome the security flaw of the original scheme, we propose a new certificateless aggregate signature scheme. New scheme is composed of six algorithms as follows.

Setup. KGC performs the following steps to complete the system initialization.

(1) Given a security parameter k , choose two cyclic group, G_1 and G_2 , with the order q , a billinear map $e: G_1 \times G_1 \to G_2$. P is a generator of G_1.

(2) Choose a random value $s \in Z_q^*$ as the master key and compute $P_{pub} = s \cdot P$ as the system public key.

(3) Define four hash functions: $H_1 - H_2 : \{0,1\}^*$, $\to G_1, h_1, h_2 : \{0,1\}^* \to Z_q^*$.

(4) Let $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, h_1, h_2\}$ as the system parameter.

Partial-Private-Key-Extract. KGC performs the following steps to generate the partial private key.

(1) Input ID_i as the identity of user U_i .

(2) Compute $Q_i = H_1(ID_i)$ and $D_i = s \cdot Q_i$.

(3) Output D_i as the partial private key of user U_i .

User-pubic-Key-Gen. The user U_i performs the following steps to generate the user public key.

(1) Input ID_i as the identity of user U_i .

(2) Choose a random value $x_i \in Z_q^*$.

(3) Compute $P_i = x_i \cdot P$ as the public key of U_i .

Single-Sign. The signer performs the following steps to generate the signature of the message m_i from the user U_i .

(1) Input ID_i as the identity of user U_i , public key P_i, and message m_i .

(2) Choose a random value $w_i \in Z_q^*$.

(3) Compute $W_i = w_i \cdot P$, $\alpha_i = h_1(m_i, ID_i, P_i, R_i)$, $\beta_i = h_2(m_i, ID_i)$ and $Z = H_2(P_{pub})$.

(4) Compute $T_i = \alpha_i D_i + (\beta_i x_i + w_i) Z$.

(5) Output $\sigma_i = (W_i, T_i)$ as the signature of the message m_i from the user U_i .

Aggregate-Sign. The aggregator performs the following steps to generate the aggregate signature from n user-message- signature pairs $(U_i, m_i, \sigma_i)_{1 \le i \le n}$.

(1) Input n tuples $(U_i, m_i, P_i, \sigma_i)$, where $1 \le i \le n$.

(2) Compute $T = \sum_{i=1}^{n} T_i$.

(3) Output $\sigma = (W, T)$ as the aggregate signature, where $W = \{W_1, W_2, \cdots, W_n\}$.

Aggregate Verify. The verifier performs the following steps to verify the validity of the aggregate signature $\sigma = (W, T)$.

(1) Input n tuples $(U_i, m_i, P_i, \sigma_i)$, and the aggregate signature $\sigma = (W, T)$, where $1 \le i \le n$.

(2) For $1 \le i \le n$, compute $Q_i = H_1(ID_i)$, $\alpha_i = h_1(m_i, ID_i, P_i, R_i)$ and $\beta_i = h_2(m_i, ID_i)$.

(3) Compute $Z = H_2(P_{pub})$.

(4) Verify

$$e(T, P) = e(P_{pub}, \sum_{i=1}^{n} \alpha_i Q_i) e(Z, \sum_{i=1}^{n} (\beta_i x_i + w_i) P) \quad (6)$$

(5) If $Equ(6)$ holds, output true. Otherwise, output false.

7 Security Analysis

A certificateless aggregate signature scheme should satisfy the following requirements: correctness and unforgeability.

7.1 Correctness

Theorem 1. The proposed certificateless aggregate scheme is correct, if and only if the single signature and aggregate signature generated by our scheme make $Equ(6)$ hold. The correctness of the protocol is elaborated as follows:

$$
\begin{aligned}
e(T,P) &= e(\sum_{i=1}^{n} T_i, P) \\
&= e(\sum_{i=1}^{n} (\alpha_i D_i + (\beta_i x_i + w_i)Z), P) \\
&= e(\sum_{i=1}^{n} \alpha_i Q_i, sP) e(\sum_{i=1}^{n} (\beta_i x_i + w_i)P, Z) \\
&= e(P_{pub}, \sum_{i=1}^{n} \alpha_i Q_i) e(Z, \sum_{i=1}^{n} (\beta_i x_i + w_i)P)
\end{aligned} \tag{7}
$$

7.2 Unforgeability

In this subsection, we first give the security model of CLAS scheme, and then give the Security proof to show that the proposed CLAS scheme is secure under the security model.

7.2.1 Security Model

There exists two types of adversaries in CL-PKC: A_1, A_2. A_1 simulates an outside attacker, who can not get the master key, but can replace any user's public key with a value of his choice. While A_2 simulates an honest-but-curious KGC, who can not replace any user's public key, but can get the master key.

The security of a CLAS scheme is defined by a game (denoted by *Game*) played between an adversary $A \in \{A_1, A_2\}$ and a challenger C, and more details are defined as follows.

Game (Including the adversary A_1 and A_2)

Setup. C performs the *Setup algorithm to generate* the master key s and the system parameter list *params*. If $A = A_1$, C sends *params* to the adversary A while keeps the master key secret; Otherwise, if $A = A_2$, C sends *params* and the master key s to the adversary A.

Attack. In the game, A can send queries to C and C answers them as follows.

- *Hash queries*: A can access any hash oracle in the scheme.
- *Partial-Private-Key-Extract queries:* When A executes the query with the user U_i's identity ID_i, in respond, C generates the private key of U_i and stores it in the list D^{list}.
- *Public-Key-Gen queries*: When A executes the query with the user U_i's identity ID_i, in respond, C generates the public key of U_i and stores it in the list P^{list}.
- *Secret-Value queries*: When A executes the query with the user U_i's identity ID_i, in respond, C generates the secret value of U_i and stores it in the list X^{list} (if the user U_i's public key has been replaced, it output $\perp$).
- *Public-Key-Replacement queries*: When A executes the query with the user U_i's identity ID_i, in respond, C replaces the real public key of U_i with A's choice in the list P^{list} (if $A = A_2$, it does not perform the query).
- *Single-Sign queries*: When A executes the query with the user U_i's identity ID_i, public key P_i, and message m_i, C generates $\sigma_i = (R_i, T_i)$ as the signature of the message m_i from the user U_i.
- *Forgery*: A outputs an aggregate signature σ^* with respect to n user-message-public key-signature pairs $(U_i^*, m_i^*, P_i^*, \sigma_i^*)_{1 \le i \le n}$. We say that A wins the *Game*, iff

1. σ^* is a valid aggregate signature with respect to n user-message-public key-signature pairs $(U_i^*, m_i^*, P_i^*, \sigma_i^*)_{1 \le i \le n}$.

2. if $A = A_1$, U_i^* has not been submitted during the *Partial-Private-Key-Extract queries*; Otherwise, if $A = A_2$, U_i^* has not been submitted during the *Secret-Value queries*.

3. (U_i^*, m_i^*) has not been submitted during the *Single-Sign queries*.

7.2.2 Provable Security

In this subsection, we prove that the proposed CLAS scheme is secure under the security model described in section 7.2.1. Our security proof includes two parts:(1) the CLAS scheme is unforgeable against type I adversary A_1. (2) the CLAS scheme is unforgeable against type II adversary A_2.

Theorem 2. The proposed certificateless aggregate scheme is existentially unforgeable against type I adversary A_1, if the assumption that the CDH problem is intractable in G_1.

Proof. We can prove the unforgeable against type I of our proposed CLAS scheme with a game that involves an adversary A_1 and an algorithm called simulator C.

Setup. Given an instance of the CDH problem $(P, Q_1 = a \cdot P, Q_2 = b \cdot P)$. Set $P_{pub} = Q_1 = a \cdot P$ and return system parameter $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, h_1, h_2\}$ to A_1. Randomly select ID_{gt} as the identity of user challenged. A_1 performs the inquiries as follows:

H_1 queries:

C maintains a list denoted H_1^{list}, and the structure of H_1^{list} is $(ID_i, P_i, \vartheta_i, Q_i, \omega_i)$, H_1^{list} is initialized to null. When A_1 executes the query with the identity ID_i, C

checks whether a tuple $(ID_i, \vartheta_i, Q_i, \omega_i)$ exists in H_1^{list}, if exists, C returns Q_i to A_1; Otherwise, C randomly selects $\vartheta_i \in Z_q^*$ and $\omega_i \in \{0,1\}$, where the probability of $\omega_i = 0$ is $\zeta = \frac{1}{q_{H_1}}$ and the probability of $\omega_i = 1$ is $1-\zeta$. If $\omega_i = 0$, set $Q_i = \vartheta_i P$; Otherwise $\omega_i = 1$, set $Q_i = \vartheta_i bP$. Return Q_i to A_1 and store $(ID_i, P_i, \vartheta_i, Q_i, \omega_i)$ in H_1^{list}.

H_2 queries:

C maintains a list denoted H_2^{list}, and the structure of H_2^{list} is (P_{pub}, υ, Z), H_2^{list} is initialized to null. When A_1 executes the query with P_{pub}, C checks if a tuple (P_{pub}, υ, Z) exists in H_2^{list}, if exists, C returns Z to A_1; Otherwise, C randomly selects $\upsilon \in Z_q^*$ and computes $Z = \upsilon P$. Return Z to A_1 and store (P_{pub}, υ, Z) in H_2^{list}.

h_1 queries:

C maintains a list denoted h_1^{list}, and the structure of h_1^{list} is $(ID_i, m_i, P_i, W_i, \alpha_i)$, h_1^{list} is initialized to null. When A_1 executes the query, C checks if a tuple $(ID_i, m_i, P_i, W_i, \alpha_i)$ exists in h_1^{list}, if exists, C returns α_i to A_1; Otherwise, C randomly selects $\alpha_i \in Z_q^*$. Return α_i to A_1 and store $(ID_i, m_i, P_i, W_i, \alpha_i)$ in h_1^{list}.

h_2 queries:

C maintains a list denoted h_2^{list}, and the structure of h_2^{list} is (ID_i, m_i, β_i), h_2^{list} is initialized to null. When A_1 executes the query, C checks if a tuple (ID_i, m_i, β_i) exists in h_2^{list}, if exists, C returns β_i to A_1; Otherwise, C randomly selects $\beta_i \in Z_q^*$. Return β_i to A_1 and store (ID_i, m_i, β_i) in h_2^{list}.

Partial-Private-Key-Extract queries. C maintains a list denoted D^{list}, and the structure of D^{list} is (ID_i, D_i), D^{list} is initialized to null. When A_1 executes the query with the identity ID_i, C first checks whether $ID_i = ID_{gt}$, if it holds, outputs $\perp$; Otherwise, C checks whether a tuple (ID_i, D_i) exists in D^{list}, if exists, C returns D_i to A_1; Otherwise, C checks whether $\omega_i = 1$, if it holds, aborts; Otherwise, C sets $D_i = \vartheta_i aP$. Return D_i to A_1 and store (ID_i, D_i) in D^{list}.

Secret-Value queries. C maintains a list denoted x^{list}, and the structure of x^{list} is (ID_i, x_i), x^{list} is initialized to null. When A_1 executes the query with the identity ID_i, C first checks whether $ID_i = ID_{gt}$, if it holds, outputs $\perp$; Otherwise, C checks whether a tuple (ID_i, x_i) exists in x^{list}, if exists, C returns x_i to A_1; Otherwise, C randomly selects $x_i \in Z_q^*$. Return x_i to A_1 and store (ID_i, x_i) in x^{list}.

User-Public-Key queries. C maintains a list denoted P^{list}, and the structure of P^{list} is (ID_i, P_i), P^{list} is initialized to null. When A_1 executes the query with the identity ID_i, C checks whether a tuple (ID_i, P_i) exists in P^{list}, if exists, C returns P_i to A_1; Otherwise, C accesses x^{list} to get x_i and computes $P_i = x_i.P$. Return P_i to A_1 and store (ID_i, P_i) in P^{list}.

Public-Key-Replacement queries. When A_1 executes the query, in respond, C replaces the real public key of U_i with P_i' chosen by A_1 in the list P^{list}.

Single-Sign queries. When A_1 executes the query with the user U_i's identity ID_i and public key P_i, message m_i, C accesses H_1^{list}, h_1^{list}, h_2^{list} and H_2^{list} to get ω_i, Q_i, α_i, β_i and Z respectively. Furthermore, C randomly selects $W_i \in G_1$, if $\omega_i = 0$, C computes $T_i = \alpha_i \vartheta_i P_{pub} + \beta_i \upsilon P_i + \upsilon W_i$; Otherwise, if $\omega_i = 1$, C computes $T_i = \alpha_i \vartheta_i bP_{pub} + \beta_i \upsilon P_i + \upsilon W_i$. Return $\sigma_i = (W_i, T_i)$ to A_1 as the signature of the message m_i from the user U_i.

Forgery. Finally, A_1 outputs a forged aggregate signature $\sigma^* = (W^*, T^*)$ from n message-identity-public key pairs $(m_i^*, ID_i^*, P_i^*)_{1 \le i \le n}$. For each i, if $\omega_i^* = 0$, A_1 aborts; Otherwise, let $ID_{gt} = ID_1$, that is, $\omega_1^* = 1$, $\omega_i^* = 0$ $(2 \le i \le n)$, then the forged aggregate signature should satisfy:

$$e(T^*, P) = e(P_{pub}, \sum_{i=1}^{n} \alpha_i^* Q_i^*) e(Z, \sum_{i=1}^{n} (\beta_i^* x_i^* + w_i^*) P) \quad \textbf{(8)}$$

where, $Q_i^* = \vartheta_i^* P$ $(2 \le i \le n)$; $Q_1^* = \vartheta_1^* bP$, $Z = \upsilon P$, $T^* = \sum_{i=1}^{n} T_i^*$, $W^* = \{W_1^*, W_2^*, ..., W_n^*\}$. Furthermore, the derivation process is shown as follows.

$$e(T^*, P)$$

$$= e(P_{pub}, \sum_{i=2}^{n} \alpha_i^* Q_i^*) e(P_{pub}, \alpha_1^* Q_1^*) e(Z, \sum_{i=1}^{n} (\beta_i^* P_i^* + W_i^*))$$

$$\Rightarrow e(P_{pub}, \alpha_1^* Q_1^*)$$

$$= e(T^*, P)[e(P_{pub}, \sum_{i=2}^{n} \alpha_i^* Q_i^*) e(Z, \sum_{i=1}^{n} (\beta_i^* P_i^* + W_i^*))]^{-1}$$

$$\Rightarrow e(\alpha_1^* \vartheta_1^* abP, P)$$

$$= e(T^*, P)[e(P_{pub}\sum_{i=2}^{n}\alpha_i^*\vartheta_i^*, P)e(\upsilon\sum_{i=1}^{n}(\beta_i^*P_i^* + W_i^*), P)]^{-1}$$

$$\Rightarrow \alpha_1^*\vartheta_1^* abP = T^* - P_{pub}\sum_{i=2}^{n}\alpha_i^*\vartheta_i^* - \upsilon\sum_{i=1}^{n}(\beta_i^*P_i^* + W_i^*)$$

$$\Rightarrow abP$$

$$= (\alpha_1^*\vartheta_1^*)^{-1}(T^* - P_{pub}\sum_{i=2}^{n}\alpha_i^*\vartheta_i^* - \upsilon\sum_{i=1}^{n}(\beta_i^*P_i^* + W_i^*))$$

Theorem 3. The proposed certificateless aggregate scheme is existentially unforgeable against type II adversary A_2 if the assumption that the CDH problem is intractable in G_1.

Proof. We can prove the unforgeable against type II of our proposed CLAS scheme with a game that involves an adversary A_2 and an algorithm called simulator C.

Setup. Given an instance from the CDH problem $(P, Q_1 = a.P, Q_2 = b.P)$, C sets $P_{pub} = Q_1 = \lambda.P$ and returns master key λ and system parameter $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, h_1, h_2\}$ to A_2. Randomly select ID_{gt} as the identity of user challenged. A_2 performs the inquiries as follows:

h_1, h_2, H_1 and Secret-Value queries are the same as the corresponding queries in Theorem 2. Since A_2 can access the master key, there is no need to the Partial-Private-Key-Extract queries.

H_2 queries:

C maintains a list denoted H_2^{list}, and the structure of H_2^{list} is (P_{pub}, υ, Z), H_2^{list} is initialized to null. When A_2 executes the query with P_{pub}, C checks if a tuple (P_{pub}, υ, Z) exists in H_2^{list}, if exists, C returns Z to A_2; Otherwise, C randomly selects $\upsilon \in Z_q^*$ and computes $Z = \upsilon aP$. Return Z to A_2 and store (P_{pub}, υ, Z) in H_2^{list}.

User-Public-Key queries. C maintains a list denoted P^{list}, and the structure of P^{list} is (ID_i, P_i, ω_i), P^{list} is initialized to null. When A_2 executes the query with the identity ID_i, C checks whether a tuple (ID_i, P_i, ω_i) exists in P^{list}, if exists, C returns P_i to A_2; Otherwise, if $\omega_i = 0$, C accesses x^{list} to get x_i and computes $P_i = x_i.P$; Otherwise, if $\omega_i = 1$, C randomly selects $x_i \in Z_q^*$ and computes $P_i = x_i bP$, where the probability of $\omega_i = 0$ is $\zeta = \frac{1}{q_{H_1}}$ and the probability of $\omega_i = 1$ is $1 - \zeta$. Return P_i to A_2 and store (ID_i, P_i, ω_i) in P^{list}.

Single-Sign queries. When A_1 executes the query with the user U_i's identity ID_i and public key P_i, message m_i, C accesses H_1^{list}, h_1^{list}, h_2^{list} and H_2^{list} to get ω_i, Q_i, α_i, β_i and Z respectively. Furthermore, C randomly selects $W_i \in G_1$, if $\omega_i = 0$, C computes $T_i = \alpha_i\vartheta_i P_{pub} + \upsilon a(\beta_i x_i P + W_i)$; Otherwise, $\omega_i = 1$, C computes $T_i = \alpha_i\vartheta_i P_{pub} + \upsilon a(\beta_i x_i bP + W_i)$. Return $\sigma_i = (W_i, T_i)$ to A_2 as the signature of the message m_i from the user U_i.

Forgery. Finally, A_2 outputs a forged aggregate signature $\sigma^* = (W^*, T^*)$ from n message-identity-public key pairs $(m_i^*, ID_i^*, P_i^*)_{1\le i\le n}$. For each i, if $\omega_i^* = 0$, A_2 aborts; Otherwise, let $ID_{gt} = ID_1$, that is, $\omega_1^* = 1$, $\omega_i^* = 0$ $(2 \le i \le n)$, then the forged aggregate signature should satisfy:

$$e(T^*, P) = e(P_{pub}, \sum_{i=1}^{n}\alpha_i^*Q_i^*)e(Z, \sum_{i=1}^{n}(\beta_i^*x_i^* + w_i^*)P) \quad \textbf{(9)}$$

where $Q_i^* = H_1(ID_i^* \| P_i^*)$, $P_1^* = x_1^* bP$, $P_i^* = x_i^* P$ $(2 \le i \le n)$, $Z = \upsilon aP$, $W^* = \{w_1^*P, w_2^*P, ..., w_n^*P\}$, $T^* = \sum_{i=1}^{n}T_i^*$. Furthermore, the derivation process is shown as follows.

$$e(T^*, P)$$

$$= e(P_{pub}, \sum_{i=1}^{n}\alpha_i^*Q_i^*)e(Z, \sum_{i=2}^{n}\beta_i^*P_i^* + \sum_{i=1}^{n}W_i^*)e(Z, \beta_1^*P_1^*)$$

$$\Rightarrow e(Z, \beta_1^*P_1^*)$$

$$= e(T^*, P)[e(P_{pub}, \sum_{i=1}^{n}\alpha_i^*Q_i^*)e(Z, \sum_{i=2}^{n}\beta_i^*P_i^* + \sum_{i=1}^{n}W_i^*)]^{-1}$$

$$\Rightarrow e(\beta_1^*x_1^*\upsilon abP, P)$$

$$= e(T^*, P)[e(\gamma\sum_{i=1}^{n}\alpha_i^*Q_i^*, P)e(\sum_{i=2}^{n}\beta_i^*x_i^*Z + \sum_{i=1}^{n}w_i^*Z, P)]^{-1}$$

$$\Rightarrow \beta_1^*x_1^*\upsilon abP = T^* - \gamma\sum_{i=1}^{n}\alpha_i^*Q_i^* - (\sum_{i=2}^{n}\beta_i^*x_i^* + \sum_{i=1}^{n}w_i^*)Z$$

$$\Rightarrow abP = (\beta_1^*x_1^*\upsilon)^{-1}[T^* - \gamma\sum_{i=1}^{n}\alpha_i^*Q_i^* - (\sum_{i=1}^{n}w_i^* + \sum_{i=2}^{n}\beta_i^*x_i^*)Z]$$

8 Security Comparisons and Performance Analysis

8.1 Security Comparisons

In this subsection, we compare the security of our proposed CLAS scheme with three recently proposed CLAS schemes [5, 18, 20]. For the convenience of description, let A_1 denotes the type I adversary, A_2 denotes the type II adversary. Furthermore, the two types of adversaries are divided three levels [26],

where B_1 denotes general adversary, B_2 denotes strong adversary, B_3 denotes super adversary respectively. √ denotes it can satisfy the corresponding security requirement. L denotes the weak security under the corresponding attack types, H denotes the strong security under the corresponding attack types. SP denotes the security performance. The security comparisons of the various schemes are listed in Table 1.

Table 1. Security comparisons

	A_1				A_2			
	B_1	B_2	B_3	SP	B_1	B_2	B_3	SP
Gong et al.'s Scheme [18]		√		L	√			L
Xiong et al.'s Scheme [20]			√	H				L
Kang et al.'s Scheme [5]			√	H				L
The proposed scheme			√	H			√	H

According to Table 1, none of the three schemes (i.e., Gong et al.'s scheme [18], Xiong et al.'s scheme [20], Kang et al.'s scheme [5]) can satisfy the strong security requirements (H). In contrast, our proposed CLAS could satisfy the strong security requirements under the corresponding attack types (super adversary in the type I and type II).

8.2 Performance Analysis

We analyze and compare the computation cost of the proposed protocol with Kang et al.'s scheme [5] in this subsection. To get a trusted security level, select q is a 160-bits prime number and p is a 512-bits prime number. A ate pairing $e: G_1 \times G_1 \to G_2$ is used in our experiments, where G_1 is an additive cyclic group, with order q , generated by a point on the super singular elliptic curve $E(F_p): y^2 = x^3 + 1$ defined on the finite field F_p .

For the sake of simplicity, we firstly define the related symbol-operation-execution time as shown in Table 2. Where we have implemented the related operations on a personal computer (Lenovo with Intel I5-3470 3.20G Hz processor, 4G bytes memory and the Window 7 operating system) using the MIRACL library [27].

Table 2. symbol-operation-execution time

Symbol	Operation	Time (ms)
T_{z-sa}	The time of performing an addition operation in Z_q	0.0008
T_{z-sm}	The time of performing an multipli -cation operation in Z_q^*	0.0011
T_{mts}	The time of performing a general hash operation	0.053
T_{mtp}	The time of performing a map-to -point operation in G_1	9.773
P_{ecc-pa}	The time of performing a point addition operation in G_1	0.022
T_{ecc-pm}	The time of performing a point multiplication operation in G_1	3.740
T_{bp}	The time of performing a bilinear pairing operation	11.515

Because *Setup*, *Partial-PrivateKey-Extract* and *User -Key-Gen* phases are executed by KGC or user and all of them are one-time operation, we only laid stress on the comparisons of the computation cost in *single-Sign* phase, *Aggregate-Sign* phase and *Aggregate-Verify* phase.

In the *single-Sign* phase, the signer in Kang et al.*'s* CLAS scheme needs to execute one general hash operation, two map-to-point operations in G_1 , two point addition operations and four point multiplication operations in G_1 . Therefore, the signer's computation cost is $T_{mts} + 2T_{mtp} + 2T_{ecc-pa} + 4T_{ecc-pm} \approx 34.603$ milliseconds. Whereas the signer in our proposed CLAS scheme needs to execute one addition operation in Z_q , one multiplication operation in Z_q^* , two general hash operations, one map-to-point operation in G_1 , one point addition operation and three point multiplication operations in G_1 . Therefore, the signer's computation cost is $T_{z-sa} + T_{z-sm} + 2T_{mts} + T_{mtp} + T_{ecc-pa} + 3T_{ecc-pm} \approx 21.123$ milliseconds.

In the *Aggregate-Sign* phase, we suppose there are 100 signatures needed to be aggregated. The aggregator in Kang et al.'s CLAS scheme needs to execute $n = 100$ point addition operations in G_1 . Whereas the aggregator in our proposed CLAS scheme also needs to execute $n = 100$ point addition operations in G_1. Therefore, we can find that the signer's computation cost of *Aggregate-Sign* phase in the two schemes is equal to $100 \times T_{ecc-pa} \approx 2.2$ milliseconds.

In the *Aggregate-Verify* phase ($n = 100$) the verifier in Kang et al.'s CLAS scheme needs to execute $n = 100$ general hash operations, $n + 2 = 102$ map-to-point operations in G_1 , $3(n-1) = 297$ point addition operations, $n = 100$ point multiplication operations in

G_1 and 4 bilinear pairing operations. Therefore, the verifier's computation cost is $100T_{mts}+102T_{mtp}+297T_{ecc-pa}+100T_{ecc-pm}+4T_{bp}\approx 1428.74$ milliseconds. Whereas the verifier in our proposed CLAS scheme needs to execute $n-1=99$ addition operations in Z_q, $n=100$ multiplication operations in Z_q^*, $2n=200$ general hash operations, $n+1=101$ map-to-point operations in G_1, $n-1=99$ point addition operations, $n+1=101$ point multiplication operations in G_1 and 3 bilinear pairing operations. Therefore, the verifier's computation cost is $99T_{z-sa}+100T_{z-sm}+200T_{mts}+101T_{mtp}+99T_{ecc-pa}+3T_{bp}+101T_{ecc-pm}\approx 1412.33$ milliseconds.

The computation cost comparisons are demonstrated in Table 3 and Figure 1. From the results in Table 3 and Figure 1, our proposed CLAS scheme has the same computation cost as that of Kang et al.'s scheme for *Aggregate-Sign* phase. At the *single-Sign* and *Aggregate-Verify* phases, the percentage reduction for computation cost with our proposed CLAS scheme is 38.96 and 1.15 lower compared with Kang et al.'s scheme respectively. Namely, we enforce the security in a large extent with lower computation cost compared with Kang et al.'s scheme [5].

Table 3. Computation cost comparisons (ms)

	Kang et al.'s Scheme [5]	The proposed scheme
Single -Sign	$2T_{mtp}+2T_{ecc-pa}$ $T_{mts}++4T_{ecc-pm}$ ≈ 34.603	$T_{z-sa}+T_{z-sm}+2T_{mts}$ $+T_{mtp}+T_{ecc-pa}$ $+3T_{bp}\approx 21.123$
Aggregate-Sign	$100\times T_{ecc-pa}\approx 2.2$	$100\times T_{ecc-pa}\approx 2.2$
Aggregate-Verify	$297T_{ecc-pa}+102T_{mtp}$ $+100T_{mts}+100T_{ecc-pm}$ $+4T_{bp}\approx 1428.74$	$99T_{z-sa}+100T_{z-sm}+200T_{mts}$ $+101T_{mtp}+99T_{ecc-pa}$ $+101T_{ecc-pm}+3T_{bp}$ ≈ 1412.33

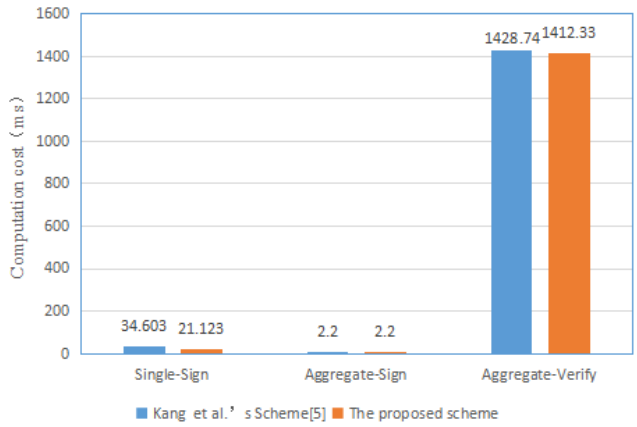


Figure 1. Computation cost comparisons

9 Conclusion

Aggregate signature can greatly reduce the verification cost, that makes it especially suitable for resource-constrained wireless networks. Unfortunately, most exiting schemes were subsequently found to have vulnerabilities or suffer unsatisfactory performance in terms of computational and communication costs. In this paper, we first provide a security analysis for an efficient certificateless aggregate signature scheme, and then present an attack method to demonstrate that their scheme is forgeable. Furthermore, an improved scheme is proposed, and then the security of the scheme is formally proved under the CDH assumption in the random oracle model. We finally evaluate the performance of our proposed scheme. Compared with Kang *et al.*'s scheme, our new scheme achieves a higher level of security assurance with lower computation cost. Therefore, our proposal is more practical and adoptable in real life to solve batch identification problem.

Acknowledgements

The work of was supported by the National Natural Science Foundation of China (Nos. 61472287, 61501333, 61572379), the Natural Science Foundation of Hubei Province of China (Nos. 2015CFA068, 2015CFB257) and the Fundamental Research Funds for the Central Universities..

References

[1] K. Ren, W. Lou, K. Kim, R. Deng, A Novel Privacy Preserving Authentication and Access Control Scheme for Pervasive Computing Environments, *IEEE Transactions on Vehicular Technology*, Vol. 55, No. 4, pp. 1373-1384, July, 2006.

[2] M. Fatemi, S. Salimi, A. Salahi, Anonymous Roaming in Universal Mobile Telecommunication System Mobile Networks, *IET Information Security*, Vol. 4, No. 2, pp. 93-103, June, 2010.

[3] G. Yang, Q. Huang, D. S. Wong, X. Deng, Universal Authentication Protocols for Anonymous Wireless Communications, *IEEE Transactions on Wireless Communications*, Vol. 9, No. 1, pp. 168-174, January, 2010.

[4] D. Boneh, C. Gentry, B. Lynn, H. Shacham, Aggregate and Verifiably Encrypted Signatures from Bilinear Maps, *International Conference on the Theory and Applications of Cryptographic Techniques*, Warsaw, Poland, 2003, pp. 416-432.

[5] B. Kang, M. Wang, D. Jing, An Efficient Certificateless Aggregate Signature Scheme, *Wuhan University Journal of Natural Sciences*,Vol. 22, No. 2, pp. 165-170, April, 2017.

[6] D. He, M. Ma, Y. Zhang, C. Chen, J. Bu, A Strong User Authentication Scheme with Smart Cards for Wireless

Communications, *Computer Communications*, Vol. 34, No. 3, pp. 367-374, March, 2011.

[7] J. Park, J. Go, K. Kim, Wireless Authentication Protocol Preserving User Anonymity, *WPMC'01 the Fourth International Symposium on Wireless Personal Multimedia Communications*, Aalborg, Denmark, 2001, pp. 1153-1158.

[8] A. Shamir, Identity-Based Cryptosystems and Signature Schemes, in: G. R. Blakley, D. Chaum (Eds.), *Advances in Cryptology, CRYPTO 1984, Lecture Notes in Computer Science*, Springer, 1985, pp. 47-53.

[9] S. S. Al-Riyami, K. G. Paterson, Certificateless Public Key Cryptography, *International Conference on the Theory and Application of Cryptology and Information Security*, Taipei, Taiwan, 2003, pp. 452-473.

[10] X. Huang, W. Susilo, Y. Mu, F. Zhang, On the Security of Certificateless Signature Schemes from Asiacrypt 2003, *International Conference on Cryptology and Network Security*, Xiamen, China, 2005, pp. 13-25.

[11] J. Li, X. Huang, Y. Mu, W. Wu, Cryptanalysis and Improvement of an Efficient Certificateless Signature Scheme, *Journal of Communications and Networks*, Vol. 10, No. 1, pp. 10-17, March, 2008.

[12] W. S. Yap, S. H. Heng, B. M. Goi, An Efficient Certificateless Signature Scheme, *Emerging Directions in Embedded and Ubiquitous Computing*, Seoul, Korea, 2006, pp. 322-331.

[13] M. H. Au, Y. Mu, J. Chen, D. S. Wong, J. K. Liu, G. Yang, Malicious KGC Attacks in Certificateless Cryptography, *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security*, Singapore, 2007, pp. 302-311.

[14] A. Dent, B. Libert, K. Paterson, Certificateless Encryption Schemes Strongly Secure in the Standard Model, *11th International Conference on Practice and Theory in Public Key Cryptography*, Barcelona, Spain, 2008, pp. 344-359.

[15] X. Li, K. Chen, L. Sun, Certificateless Signature and Proxy Signature Schemes from Bilinear Pairings, *Lithuanian Mathematical Journal*, Vol. 45, No. 1, pp. 76-83, January, 2005.

[16] J. Liu, M. Au, W. Susilo, Self-generated-certificate Public Key Cryptography and Certificateless Signature/Encryption Scheme in the Standard Model, *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security*, Singapore, 2007, pp. 273-283.

[17] R. Castro, R. Dahab, Efficient Certificateless Signatures Suitable for Aggregation, *IACR Cryptology ePrint Archive*, http://eprint.iacr.org/2007/454.

[18] Z. Gong, Y. Long, X. Hong, K. Chen, Two Certificateless Aggregate Signatures from Bilinear Maps, *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, (SNPD 2007)*, Vol. 3, Qingdao, China, 2007, pp. 188-193.

[19] L. Zhang, F. Zhang, A New Certificateless Aggregate Signature Scheme, *Computer Communications*, Vol. 32, No. 6, pp. 1079-1085, April, 2009.

[20] H. Xiong, Z. Guan, Z. Chen, F. Li, An Efficient Certificateless Aggregate Signature with Constant Pairing Computations, *Information Sciences*, Vol. 219, pp. 225-235, January, 2013.

[21] D. He, M. Tian, J. Chen, Insecurity of an Efficient Certificateless Aggregate Signature with Constant Pairing Computations, *Information Sciences*, Vol. 268, pp. 458-462, June, 2014.

[22] H. Liu, S. Wang, M. Liang, Y. Chen, New Construction of Efficient Certificateless Aggregate Signatures, *International Journal of Security and Its Applications*, Vol. 8, No. 1, pp. 411-422, January, 2014.

[23] L. Cheng, Q. Wen, Z. Jin, H. Zhang, L. Zhou, Cryptanalysis and Improvement of a Certificateless Aggregate Signature Scheme, *Information Sciences*, Vol. 295, pp. 337-346, February, 2015.

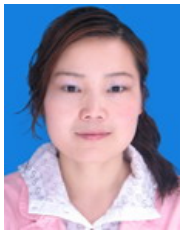
[24] F. Zhang, L. Shen, G. Wu, Notes on the Security of Certificateless Aggregate Signature Schemes, *Information Sciences*, Vol. 287, pp. 32-37, December, 2014.

[25] Y. Zhang, C. Wang, Comment on New Construction of Efficient Certificateless Aggregate Signatures, *International Journal of Security and Its Applications*, Vol. 9, No. 1, pp. 147-154, January, 2015.

[26] D. He, S. Zeadally, L. Wu, Certificateless Public Auditing Scheme for Cloud-assisted Wireless Body Area Networks, *IEEE Systems Journal*, Vol. 12, No. 1, pp. 64-73, March, 2018.

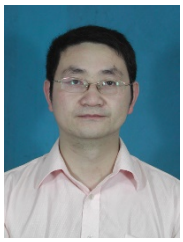
[27] Shamus Software Ltd., *MIRACL Library*, http://www.shamus.ie/index.php?page=home.

Biographies



Zhiyan Xu received her master's degree from Central China normal University in 2007. She is currently pursing a doctor's degree at Computer School, Wuhan university, Wuhan, China. Her research interests are in the areas of digital signatures and secure cloud storage. Email:csxzy@whu.edu.cn

Libing Wu was born in 1972. He received the B.S. and M.S. degrees in computer science from Central China Normal University, Wuhan, China, in 1994 and 2001, respectively. He received his Ph.D. degree in computer science from Wuhan University in 2006. He is now a Professor in the Computer School, Wuhan University, China. He is a senior member of IEEE and CCF. His areas of research interests include distributed computing, trusted software and wireless sensor networks. Email:wu@whu.edu.cn



Yongjun Ren obtained the Ph.D. degree in the computer and science department at the Nanjing University of Aeronautics and Astronautics, China, in 2008. Now he is serving as a full time faculty in the Nanjing University of Information Science and Technology. His research interests include network security and applied cryptography. Email: renyj100@126.com

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University in 2009. He is currently a Professor of the State Key Lab of Software Engineering, Computer School, Wuhan University. His main research interests include cryptography and information security, in particular, cryptographic protocols. E-mail: hedebiao@163.com