Certificateless Aggregate Signcryption Algorithm for IoT Communication Postprint

Authors: Hu Ronglei, Li Wenjing, Jiang Hua, Zeng Ping, Wang Qingrui, Chen Lei

Date: 2018-11-29T00:00:00+00:00

Abstract

To address the low computational efficiency of existing certificateless aggregate signcryption (CLASC) schemes, this paper proposes a pairing-free aggregate signcryption scheme suitable for the Internet of Things (IoT). Compared with the current best scheme [15], the proposed scheme achieves nearly a sixfold improvement in operational efficiency, requiring only (2n+1) point multiplication operations in the aggregate signcryption phase and (5n+1) point multiplication operations in the aggregate de-signcryption phase. Based on the discrete logarithm problem, the scheme is proven to satisfy both confidentiality and unforgeability in the random oracle model. During the aggregate signcryption verification phase, the scheme does not require any third-party secret information, thereby fulfilling public verifiability. Finally, it is demonstrated that the scheme can achieve high security at a relatively low computational cost, making it more suitable for IoT applications.

Full Text

Certificateless Aggregate Signcryption Scheme for Internet of Things Communication

Hu Ronglei, Li Wenjing, Jiang Hua, Zeng Ping, Wang Qingrui, Chen Lei

Dept. of Communication Engineering, Beijing Electronic Science & Technology Institute, Beijing 100070, China

Abstract

Aiming at the problem of low computational efficiency in current certificateless aggregate signcryption (CLASC) schemes, this paper proposes an aggregate

signcryption scheme suitable for the Internet of Things (IoT) that eliminates bilinear pairings. Compared with the state-of-the-art scheme [15], the proposed scheme improves operational efficiency by nearly sixfold, requiring only (2n+1) point multiplication operations in the aggregate signcryption phase and (5n+1) point multiplication operations in the aggregate unsigncryption phase. Based on the discrete logarithm problem, we prove that the scheme satisfies confidentiality and unforgeability in the random oracle model. During the aggregate signcryption verification phase, no secret information from third parties is required, thus the scheme satisfies public verifiability. Finally, we demonstrate that the scheme can achieve higher security at lower computational cost, making it more suitable for IoT applications.

Keywords: certificateless aggregate signcryption; Internet of Things; without bilinear pairings; random oracle model; public verifiability

0 Introduction

The Internet of Things (IoT) concept was first proposed by the International Telecommunication Union in 2005 [1]. Since then, increasing numbers of countries have invested in IoT research, yielding fruitful results [2–5]. IoT has been widely applied in food safety, public security, health monitoring, intelligent transportation, security, environmental protection, and numerous other industries. As network scales expand from single laboratories to entire buildings and communities, and as different systems converge, IoT systems face challenges: IoT terminals are widely distributed and structurally simpler than traditional computer network terminals, making them vulnerable to attacks and lacking in functionality. Applications such as food safety and intrusion detection require IoT systems to provide rapid, accurate responses to emergencies, users, and administrators, enabling accurate communication between people and things while ensuring economical deployment of network infrastructure. This necessitates system coordination in an efficient, reliable, and secure manner, making cryptographic technology essential for designing secure and efficient algorithms and protocols.

Modern cryptography serves as the core technology for ensuring information security, guaranteeing confidentiality, integrity, availability, and non-repudiation in network environments. While confidentiality can be achieved through encryption and authentication through digital signatures, traditional public-key cryptography implements both through "sign-then-encrypt," which suffers from low efficiency. In 1997, Zheng et al. [6] proposed the concept of signcryption and presented a concrete scheme. Baek et al. [7] first defined the security model for signcryption schemes in 2002. In practical applications, when many users perform signcryption, receivers must verify multiple ciphertexts simultaneously. To improve batch verification efficiency, Selvi et al. [8] combined the advantages of aggregate signatures [9] and proposed the concept of aggregate signcryption

in 2009.

In 2003, Al-Riyami et al. [10] first introduced certificateless cryptography, which avoids public key certificate management and verification issues while effectively solving the key escrow problem. Barbosa et al. [11] first proposed a certificateless signcryption scheme and its security model in 2008. Subsequently, Eslami et al. [12], Liu Jianhua et al. [13], Niu Shufen et al. [14], and Han et al. [15] each proposed certificateless signcryption schemes.

Through research on aggregate cryptography, this paper proposes a certificateless signcryption scheme for IoT systems that eliminates bilinear pairing operations. The absence of complex bilinear pairing operations gives the scheme high computational efficiency. We prove that the scheme satisfies unforgeability, confidentiality, and public verifiability.

[Figure 1: see original paper] IoT architecture of WSN

1 Preliminaries

1.1 Discrete Logarithm Problem

Discrete Logarithm Problem (DLP): Let G be an additive cyclic group of order q, and P be a generator of G. For a given $aP \in G$, computing $a \in \mathbb{Z}_q^*$ is computationally hard.

1.2 Computational Diffie-Hellman Problem

Computational Diffie-Hellman Problem (CDHP): For unknown $a, b \in \mathbb{Z}_q^*$, given (P, aP, bP), computing abP is computationally hard.

1.3 System Model

A complete IoT system consists of sensory nodes (SN_i, $1 \leq i \leq n$), a gateway node (GN), a cloud platform server (CPS), and an application terminal (AT), as shown in Figure 1. Sensory nodes collect data and transmit it hop-by-hop to the gateway node. The gateway node automatically stores data and periodically transmits the collected data to the internet cloud platform server at regular intervals. The cloud platform server sends the received data to the application terminal for decryption and analysis. The CPS is honest and reliable, responsible for system management and maintenance, including registration of SN, GN, and AT, and private key distribution. Communication occurs between CPS and GN, GN and SN, and GN and GN via wireless channels.

The concrete implementation process is as follows:

a) System Initialization: This algorithm is executed by GN. Inputting security parameter k, it selects a large prime q ($q > 2^k$), defines G as a cyclic group on an elliptic curve, with the elliptic curve equation defined on $\mathbb{F}_p$ (where $\mathbb{F}_p$ denotes a finite field with p elements, p is prime and $p > 3$). Let P be a

generator on the elliptic curve. Select three collision-resistant hash functions: $H_1 : \{0,1\}^* \times G \times G \to \mathbb{Z}_q^*$, $H_2 : G \times G \to \mathbb{Z}_q^*$, $H_3 : G \times G \times G \times G \to \mathbb{Z}_q^*$, and $H_4 : G \times G \times G \times G \to \mathbb{Z}_q^*$. CPS randomly selects master key $s \in \mathbb{Z}_q^*$ and keeps it secret, computes the master public key $P_{pub} = sP$. CPS publishes system public parameters $params = \{G, q, P, P_{pub}, H_1, H_2, H_3, H_4\}$. The set of all solutions on the elliptic curve plus a point at infinity is denoted as $\mathcal{O}$, i.e., $\mathcal{O} = \{(x, y)|x, y \in \mathbb{F}_p\}$ satisfying the equation $y^2 = x^3 + ax + b \pmod p$. The number of points on the curve is denoted by q, which becomes the order of the elliptic curve.

b) Key Generation: This algorithm is executed by sensory node SN_i. SN_i randomly selects secret value $x_i \in \mathbb{Z}_q^*$ and saves it, computes $X_i = x_{iP}$, and sends (ID_i, X_i) to CPS, where ID_i is the identity. The private key is x_i and the public key is X_i.

c) Partial Private Key Generation: This algorithm is executed by CPS. CPS randomly selects secret $r_i \in \mathbb{Z}_q^*$ and sends (R_i, D_i) to each sensory node SN_i through a secure channel, where $R_i = r_{iP}$, $h_i = H_1(ID_i, R_i, X_i)$, and $D_i = r_i + s \cdot h_i$. Here, R_i serves as the user' s partial public key and D_i as the user' s partial private key.

Thus, the private key of SN_i is $SK_i = (D_i, x_i)$ and the public key is $PK_i = (R_i, X_i)$. Similarly, the application terminal AT' s private key is $SK_B = (D_B, x_B)$ and public key is $PK_B = (R_B, X_B)$.

d) Individual Signcryption: This algorithm is executed by sensory node SN_i. To signcrypt message m_i sent to AT, SN_i performs the following steps:

1. SN_i randomly selects $k_i, t_i \in \mathbb{Z}_q^*$, computes $K_i = k_{iP}$, $T_i = t_{iP}$, $Q_i^B = k_{iX}B$, and $Q_i^{B'} = t_i(R_B + P_{pub} \cdot H_1(ID_B, R_B, X_B))$.
2. Compute $h_2^i = H_2(Q_i^B, Q_i^{B'})$, $C_i = m_i \oplus H_3(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i)$, and $h_4^i = H_4(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i)$.
3. Compute $S_i = k_i + t_i + D_i + h_2^i \cdot x_i$.

The signcryption of message m_i from SN_i to AT is: $\sigma_i = (C_i, K_i, T_i, S_i)$.

e) Aggregate Signcryption: This algorithm is executed by the gateway node CN. Upon receiving n signcryption messages $(ID_i, C_i, K_i, T_i, S_i)$ from n signcrypters, the aggregator CN computes $S = \sum_{i=1}^{n} S_i$. The aggregate signcryption is $\sigma = \{ID_i, C_i, K_i, T_i, S\}_{i=1}^n$, which is sent to AT.

f) Unsigncryption: This algorithm is executed by the application terminal AT. To unsigncrypt $\sigma_i = (C_i, K_i, T_i, S_i)$ sent by SN_i, AT performs:

1. Compute $Q_i^B = x_B K_i$ and $Q_i^{B'} = D_B \cdot H_1(ID_B, R_B, X_B) + R_B$.
2. Recover plaintext: $m_i = C_i \oplus H_3(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i)$.

3. Verify signature correctness: Check if $S_{iP} = K_i + T_i + (R_i + P_{pub} \cdot H_1(ID_i, R_i, X_i)) + h_2^i X_i$ holds. If valid, output m_i; otherwise output "false" .

g) Aggregate Unsigncryption: This algorithm is executed by the application terminal AT. To unsigncrypt $\sigma = \{ID_i, C_i, K_i, T_i, S\}_{i=1}^n$ sent by SN_i, AT:

1. Compute $Q_i^B = x_B K_i$ and $Q_i^{B'} = D_B \cdot H_1(ID_B, R_B, X_B) + R_B$ for each i.
2. Recover plaintext: $m_i = C_i \oplus H_3(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i)$.
3. Verify signature correctness: Check if $SP = \sum_{i=1}^n [K_i + T_i + (R_i + P_{pub} \cdot H_1(ID_i, R_i, X_i)) + h_2^i X_i]$ holds. If valid, output m_i; otherwise output "false" .

3 Security Analysis

3.1 Correctness

Theorem 1. The receiver can verify the correctness of signcryption and aggregate signcryption, and obtain the correct decrypted plaintext.

Proof. a) AT can verify the correctness of signcryption $\sigma_i = (C_i, K_i, T_i, S_i)$:

Since $Q_i^B = x_B K_i = k_i X_B$ and $Q_i^{B'} = D_B \cdot H_1(ID_B, R_B, X_B) + R_B = t_i(R_B + P_{pub} \cdot H_1(ID_B, R_B, X_B))$, we have $h_2^i = H_2(Q_i^B, Q_i^{B'})$. Then:

$$\begin{aligned} S_{iP} &= (k_i + t_i + D_i + h_2^i \cdot x_i)P \\ &= k_{iP} + t_{iP} + (r_i + s \cdot H_1(ID_i, R_i, X_i))P + h_2^i \cdot x_{iP} \\ &= K_i + T_i + (R_i + P_{pub} \cdot H_1(ID_i, R_i, X_i)) + h_2^i X_i \end{aligned}$$

b) AT can verify the correctness of aggregate signcryption:

$$\begin{aligned} SP &= \sum_{i=1}^n S_{iP} \\ &= \sum_{i=1}^n [K_i + T_i + (R_i + P_{pub} \cdot H_1(ID_i, R_i, X_i)) + h_2^i X_i] \end{aligned}$$

c) AT can obtain the correct decrypted plaintext. Since $Q_i^B = x_B K_i$ encrypts the plaintext and AT decrypts the ciphertext using $Q_i^B = k_i X_B$, AT can correctly recover the plaintext.

3.2 Unforgeability

Theorem 2. In the random oracle model and under the assumption that DLP is hard, the proposed CLASC scheme is existentially unforgeable against adaptive chosen-message attacks (EUF-CLASC-CMA).

Lemma 1. In the random oracle model, if there exists a probabilistic polynomial-time adversary $\mathcal{A}_I$ that wins the game with non-negligible probability, then there exists an algorithm $\mathcal{C}$ that can solve the DLP problem (where $\mathcal{A}_I$ makes at most q_{H_i} queries to H_i for $i = 1, 2, 3, 4$, q_{SK} private key queries, q_{PSK} partial private key queries, q_{PK} public key queries, and q_{SC} signcryption queries, with n users in the aggregate signature).

Proof. Assume algorithm $\mathcal{C}$ is a DLP solver that receives input tuple (P, bP) where $b \in \mathbb{Z}_q^*$ is unknown, with the goal of computing b. $\mathcal{C}$ uses $\mathcal{A}_I$ as a subroutine. $\mathcal{C}$ maintains six lists $L_1, L_2, L_3, L_4, L_{ID}, L_{SC}$ to record queries to oracles H_1, H_2, H_3, H_4, user creation, and signcryption. All lists are initially empty.

1. **System Initialization Phase:** $\mathcal{C}$ generates system parameters $params = \{G, q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ (here $P_{pub} = bP$ serves as the system master key, kept secret from $\mathcal{A}_I$) and sends them to $\mathcal{A}_I$.
2. **Query Phase:** $\mathcal{A}_I$ executes polynomially bounded queries.

- H_1 **Query:** $\mathcal{C}$ maintains list L_1. On input (ID_i, R_i, X_i), if the query exists in L_1, return the corresponding h_i; otherwise, randomly select $h_i \in \mathbb{Z}_q^*$, add (ID_i, R_i, X_i, h_i) to L_1, and return h_i.
- H_2 **Query:** $\mathcal{C}$ maintains list L_2. On input $(Q_i^B, Q_i^{B'})$, if the query exists, return h_2^i; otherwise, randomly select $h_2^i \in \mathbb{Z}_q^*$, add $(Q_i^B, Q_i^{B'}, h_2^i)$ to L_2, and return h_2^i.
- H_3 **Query:** $\mathcal{C}$ maintains list L_3. On input $(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i)$, if the query exists, return h_3^i; otherwise, randomly select $h_3^i \in \mathbb{Z}_q^*$, add $(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i, h_3^i)$ to L_3, and return h_3^i.
- H_4 **Query:** $\mathcal{C}$ maintains list L_4. On input $(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i)$, if the query exists, return h_4^i; otherwise, randomly select $h_4^i \in \mathbb{Z}_q^*$, add $(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i, h_4^i)$ to L_4, and return h_4^i.
- **User Creation Query:** $\mathcal{C}$ maintains initially empty list L_{ID}. On submitted user identity ID_i, if ID_i already exists in L_{ID}, ignore; otherwise, $\mathcal{C}$ executes H_1 query to obtain h_i, then:
  - If $i \neq j$, randomly select $r_i, x_i \in \mathbb{Z}_q^*$, compute $R_i = r_{iP}$, $X_i = x_{iP}$, and insert $(ID_i, h_i, D_i, r_i, R_i, x_i, X_i)$ into L_{ID}.
  - If $i = j$, randomly select $r_i, x_i \in \mathbb{Z}_q^*$, compute $R_i = r_{iP}$, $X_i = x_{iP}$, $D_i = r_i + b \cdot h_i$, and insert $(ID_i, h_i, D_i, r_i, R_i, x_i, X_i)$ into L_{ID}.
- **Partial Private Key Query:** On submitted user identity ID_i, return the corresponding D_i.
- **Private Key Query:** On submitted user identity ID_i:
  - If $i = j$, terminate the game.
  - Otherwise, return $SK_i = (D_i, x_i)$ to $\mathcal{A}_I$.

- **Public Key Query:** On submitted user ID_i, return the corresponding public key (R_i, X_i).
- **Public Key Replacement Query:** On submitted user identity ID_i and replacement public key (R_i', X_i'), replace the original public key of legitimate signcrypter ID_i with (R_i', X_i').
- **Signcryption Query:** $\mathcal{C}$ maintains initially empty list L_{SC}. On submitted message m_i, sender identity ID_i, and receiver identity ID_B:
  - Randomly select $S_i, h_3^i, h_4^i \in \mathbb{Z}_q^*$ and $K_i, T_i \in G$.
  - Query lists L_3 and L_4. If $(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i, h_3^i)$ exists in L_3 or $(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i, h_4^i)$ exists in L_4, reselect values.
  - Otherwise, compute $C_i = m_i \oplus h_3^i$ and return ciphertext $\sigma_i = (C_i, K_i, T_i, S_i)$.
  - Finally, insert $(ID_i, m_i, ID_B, K_i, T_i, h_3^i, h_4^i, S_i, c)$ into L_{SC}, and insert $(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i, h_3^i)$ and $(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i, h_4^i)$ into L_3 and L_4 respectively.

3. **Forgery Phase:** After the query phase, $\mathcal{A}_I$ submits challenge user identity (ID_i, ID_j), message m_i, and its signcryption ciphertext $\sigma_i = (C_i, K_i, T_i, S_i)$. $\mathcal{C}$ decrypts the message. According to the forking lemma [17], $\mathcal{C}$ can obtain two valid signatures using oracle replay attack techniques. Then $\mathcal{C}$ computes:

$$b = \frac{(S_j - S_j') - (h_3^j - h_3^{j'}) \cdot D_j}{(h_3^j - h_3^{j'}) \cdot H_1(ID_j, R_j, X_j)}$$

Thus, $\mathcal{C}$ successfully obtains an instance of the DLP hard problem. The advantage of $\mathcal{C}$ in successfully solving the DLP problem is:

$$\varepsilon' \geq \frac{\varepsilon}{q_{PSK}+1}\left(1 - \frac{q_{PSK}}{q}\right)\left(1 - \frac{q_{PSK}+q_{SC}}{q}\right)$$

Lemma 2. In the random oracle model, if there exists a probabilistic polynomial-time adversary $\mathcal{A}_{II}$ that wins the game with non-negligible probability, then there exists algorithm $\mathcal{C}$ that can solve the DLP problem (where $\mathcal{A}_{II}$ makes at most q_{H_i} queries to H_i for $i = 1, 2, 3, 4$, q_{SK} private key queries, q_{PSK} partial private key queries, q_{PK} public key queries, and q_{SC} signcryption queries, with n users in the aggregate signature).

Proof: Assume algorithm $\mathcal{C}$ is a DLP solver receiving input tuple (P, bP) where $b \in \mathbb{Z}_q^*$ is unknown, with the goal of computing b. $\mathcal{C}$ uses $\mathcal{A}_{II}$ as a subroutine. $\mathcal{C}$ maintains six lists $L_1, L_2, L_3, L_4, L_{ID}, L_{SC}$ to record queries to oracles H_1, H_2, H_3, H_4, user creation, and signcryption. All lists are initially empty.

1. **System Initialization Phase:** $\mathcal{C}$ generates system parameters $params = \{G, q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ where $P_{pub} = sP$ and $s \in \mathbb{Z}_q^*$ is randomly selected. $\mathcal{C}$ sends $(params, s)$ to $\mathcal{A}_{II}$.

2. **Query Phase:** $\mathcal{A}_{II}$ executes polynomially bounded queries.

- H_1 **Query:** $\mathcal{C}$ maintains list L_1. On input (ID_i, R_i, X_i), if the query exists, return h_i; otherwise, randomly select $h_i \in \mathbb{Z}_q^*$, add $(ID_i, h_i, D_i, r_i, R_i, x_i, X_i)$ to L_1, and return h_i.
- H_2 **Query:** Same as Lemma 1.
- H_3 **Query:** Same as Lemma 1.
- H_4 **Query:** Same as Lemma 1.
- **User Creation Query:** $\mathcal{C}$ maintains initially empty list L_{ID}. On submitted user identity ID_i, if ID_i exists in L_{ID}, ignore; otherwise:
  - If $i \neq j$, randomly select $r_i, x_i \in \mathbb{Z}_q^*$, compute $R_i = r_{iP}$, $X_i = x_{iP}$, $D_i = r_i + s \cdot H_1(ID_i, R_i, X_i)$, and insert $(ID_i, h_i, D_i, r_i, R_i, x_i, X_i)$ into L_{ID}.
  - If $i = j$, set $X_i = bP$, randomly select $r_i \in \mathbb{Z}_q^*$, compute $R_i = r_{iP}$, and insert $(ID_i, h_i, D_i, r_i, R_i, \perp, X_i)$ into L_{ID}.
- **Partial Private Key Query:** On submitted user identity ID_i, return the corresponding D_i.
- **Private Key Query:** On submitted user identity ID_i, return the corresponding $SK_i = (D_i, x_i)$.
- **Public Key Query:** On submitted user ID_i, return the corresponding public key (R_i, X_i).
- **Public Key Replacement Query:** On submitted user identity ID_i and replacement public key (R_i', X_i'), replace the legitimate signcrypter ID_i' s original public key (R_i, X_i) with (R_i', X_i').
- **Signcryption Query:** $\mathcal{C}$ maintains initially empty list L_{SC}. On submitted message m_i, sender identity ID_i, and receiver identity ID_B:
  - Randomly select $S_i, h_3^i, h_4^i \in \mathbb{Z}_q^*$ and $K_i, T_i \in G$.
  - Query lists L_3 and L_4. If $(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i, h_3^i)$ exists in L_3 or $(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i, h_4^i)$ exists in L_4, reselect values.
  - Otherwise, compute $C_i = m_i \oplus h_3^i$ and return ciphertext $\sigma_i = (C_i, K_i, T_i, S_i)$.
  - Finally, insert $(ID_i, m_i, ID_B, K_i, T_i, h_3^i, h_4^i, S_i, c)$ into L_{SC}, and insert $(ID_i, ID_B, Q_i^B, Q_i^{B'}, K_i, h_3^i)$ and $(ID_i, ID_B, Q_i^B, Q_i^{B'}, T_i, h_4^i)$ into L_3 and L_4 respectively.

3. **Forgery Phase:** After the query phase, $\mathcal{A}_{II}$ submits challenge user identity (ID_i, ID_j), message m_i, and its signcryption ciphertext $\sigma_i =$

(C_i, K_i, T_i, S_i). $\mathcal{C}$ decrypts the message. According to the forking lemma [17], $\mathcal{C}$ can obtain two valid signatures using oracle replay attack techniques. Then $\mathcal{C}$ computes:

$$b = \frac{(S_j - S_j') - (h_4^j - h_4^{j'}) \cdot t_j}{(h_4^j - h_4^{j'})}$$

Thus, $\mathcal{C}$ successfully obtains an instance of the DLP hard problem. The advantage of $\mathcal{C}$ in successfully solving the DLP problem is:

$$\varepsilon' \geq \frac{\varepsilon}{q_{PSK}+1}\left(1 - \frac{q_{PSK}}{q}\right)\left(1 - \frac{q_{PSK}+q_{SC}}{q}\right)$$

3.3 Confidentiality

Theorem 3. In the random oracle model and based on the CDHP, the proposed CLASC scheme is indistinguishable against adaptive chosen-ciphertext attacks, i.e., IND-CLASC-CCA2 secure.

Lemma 3. In the random oracle model, if there exists a probabilistic polynomial-time adversary $\mathcal{A}_I$ (or $\mathcal{A}_{II}$) that wins the game with non-negligible probability, then there exists challenger $\mathcal{C}$ that can solve a CDHP instance with non-negligible probability.

Proof: The proof method is similar to the confidentiality proof in reference [12] and is omitted here due to space limitations.

3.4 Public Verifiability

In this scheme, when disputes occur between signcryption senders and receivers regarding the authenticity of aggregate signcryption ciphertexts, any third party can verify the following equation:

$$SP = \sum_{i=1}^{n}[K_i + T_i + (R_i + P_{pub} \cdot H_1(ID_i, R_i, X_i)) + h_2^i X_i]$$

Since this verification requires no participation from the receiver and needs no secret information from any signcrypter, the scheme provides public verifiability.

3.5 Performance Analysis

To facilitate comparison of signcryption scheme computational efficiency, assume n users participate in signcryption. We consider three operations: exponentiation (denoted as e), group multiplication (denoted as G), and bilinear pairing (denoted as p). Compared with these three operations, the time cost of hash and XOR operations has negligible impact on overall efficiency. In

the proposed scheme' s signcryption phase, n signcrypters computing $Q_i^{B'} = t_i(R_B + P_{pub} \cdot H_1(ID_B, R_B, X_B))$ require $(2n+1)$ multiplication operations (the value $R_B + P_{pub} \cdot H_1(ID_B, R_B, X_B)$ is fixed and only needs to be computed once). In the unsigncryption phase, verifying the equation requires $(5n + 1)$ multiplication operations. Reference [18] provides experimental data showing the time costs (in milliseconds) for the three operations are approximately: $t_e \approx 20.01$, $t_G \approx 11.20$, and $t_p \approx 0.83$.

Comparison of computation and security performance of aggregation signcryption

As shown in Table 1, when signcrypting the same number of messages, the proposed scheme significantly improves computational efficiency compared to schemes in references [12–15], achieving nearly six times higher efficiency than the relatively efficient scheme. In terms of security performance, only reference [13] and this scheme satisfy public verifiability. Considering both computational efficiency and security, the proposed scheme is superior to the above four schemes.

4 Conclusion

Aggregate signcryption offers significant application value in IoT environments due to its encryption, signature, and batch processing capabilities. To improve the computational efficiency of certificateless aggregate signcryption, this paper proposes a pairing-free aggregate signcryption scheme based on the random oracle model. The scheme is proven to satisfy confidentiality, unforgeability, and public verifiability. Compared with existing schemes, the proposed scheme achieves faster computation speed and is more suitable for IoT applications.

References

[1] Presser M, Barnaghi P, Eurich M, et al. The sensei project: integrating the physical world with the digital world of the network of the future [J]. IEEE Communications Magazine, 2009, 47 (4): 1-4.

[2] Han Jinsoo, Park W K, Lee I, et al. Home-to-home communications for smart community with Internet of things [C]// Proc of Consumer Communications & NetworkinG Conference. Piscataway, NJ: IEEE Press, 2017: 720-723.

[3] Lanotte R, Merro M. A semantic theory of the Internet of things [C]// Proc of International Conference on Coordination Languages and Models. Berlin: Springer, 2016: 157-174.

[4] Palade A, Cabrera C, Li Fan, et al. Middleware for Internet of things: an evaluation in a small-scale IoT environment [J]. Journal of Reliable Intelligent Environments, 2018, 1 (4): 1-21.

[5] Baldini G, Skarmeta A, Fourneret E, et al. Security certification and labelling in internet of things [C]// Proc of the 3rd IEEE World Forum on Internet of

Things. Piscataway, NJ: IEEE Press, 2016: 627-632.

[6] Zheng Yuliang. Digital signcryption or how to achieve cost (signature & encryption) cost (signature) +cost (encryption) [C]// Proc of International Cryptology Conference on Advances in Cryptology. Berlin: Springer, 1997: 165-179.

[7] Baek J, Steinfeld R, Zheng Yuliang. Formal proofs for the security of signcryption [J]. Journal of Cryptology, 2007, 20 (2): 203-235.

[8] Selvi D S, Vivek S S, Shriram J, et al. Identity based aggregate signcryption schemes [C]// International Conference on Progress in Cryptology-indocrypt. Berlin: Springer, 2009: 378-397.

[9] 张玉磊, 李臣意, 王彩芬, 等. 无证书聚合签名方案的安全性分析和改进 [J]. 电子与信息学报, 2015, 37 (8): 1994-1999. (Zhang Yulei, Li Chenyi, Wang Caifen, et al. Security analysis and improvements of certificateless aggregate signature schemes [J]. Journal of Electronics & Information Technology, 2015, 37 (8): 1994-1999.)

[10] Al-Riyami S S, Paterson K G. Certificateless public key cryptography [J]. Asiacrypt, 2003, 2894 (2): 452-473.

[11] Barbosa M, Farshim P. Certificateless signcryption [C]// Proc of ACM Symposium on Information, Computer and Communications Security. New York: ACM Press, 2008: 369-372.

[12] Eslami Z, Pakniat N. Certificateless aggregate signcryption: security model and a concrete construction secure in the random oracle model [J]. Journal of King Saud University-Computer and Information Sciences, 2014, 26 (3): 276-286.

[13] 刘建华, 赵长啸, 毛可飞. 高效的无证书聚合签密方案 [J]. 计算机工程与应用, 2016, 52 (12): 131-135. (Liu Jianhua, Zhao Changxiao, Mao Kefei. Efficient certificateless aggregate signcryption scheme based on XOR [J]. Computer Engineering and Applications, 2016, 52 (12): 131-135.)

[14] 牛淑芬, 牛灵, 王彩芬, 等. 一种可证安全的异构聚合签密方案 [J]. 电子与信息学报, 2017, 39 (5): 1213-1218. (Niu Sufen, Niu Ling, Wang Caifen, et al. A provable aggregate signcryption for heterogeneous systems [J]. Journal of Electronics & Information Technology, 2017, 39 (5): 1213-1218.)

[15] Han Yiliang, Chen Fei. The multilinear maps based certificateless aggregate signcryption scheme [C]// Proc of International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery. Piscataway, NJ: IEEE Press, 2015: 92-99.

[16] Lin Jie, Yu Wei, Zhang Nan, et al. A survey on internet of things: architecture, enabling technologies, security and privacy, and applications [J]. IEEE Internet of Things Journal, 2017, 4 (5): 1125-1142.

[17] Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures [J]. Journal of Cryptology, 2000, 13 (3): 361-396.

[18] Deng Lunzhi, Zeng Jiwen, Qu Yunyun. Certificateless proxy signature from rsa [J]. Mathematical Problems in Engineering, 2014, 2014 (9): 1-10.

Note: Figure translations are in progress. See original paper for figures.