Provably Secure Aggregate Signcryption Scheme

Xun-Yi Ren, Zheng-Hua Qi, and Yang Geng

An aggregate signature scheme is a digital signature scheme that allows aggregation of *n* distinct signatures by *n* distinct users on *n* distinct messages. In this paper, we present an aggregate signcryption scheme (ASC) that is useful for reducing the size of certification chains (by aggregating all signatures in the chain) and for reducing message size in secure routing protocols. The new ASC scheme combines identity-based encryption and the aggregation of signatures in a practical way that can simultaneously satisfy the security requirements for confidentiality and authentication. We formally prove the security of the new scheme in a random oracle model with respect to security properties IND-CCA2, AUTH-CMA2, and EUF-CMA.

Keywords: Identity-based cryptography, signcryption, aggregate signature, bilinear pairing.

I. Introduction

An aggregate signature scheme is a digital signature scheme, the concept of which was first proposed by Boneh and others [1]. Aggregate signature allows aggregation of different signatures by n different users ID$_i$ on different messages m_i. The primary objective of an aggregate signature scheme is to achieve both computation and communication efficiency. In aggregate signature, multiple signatures from various users are combined into a single compact signature. Aggregation can be used to reduce the certificate chains in public key infrastructure (PKI) settings. The aggregate signature has many real world applications ranging from traffic control to documents signed by directors of a company for official purpose.

In certain scenarios, one may need to hide the sending message so that only the receiver will be able to get back the message. In such cases, signcryption comes into the picture. Consider the scenario of an online opinion poll. The verifier has to ensure that all the concerned persons have polled their votes in an efficient way, but one may want his opinion to be secret. Only the verifier will be able to decrypt the messages and get the opinions. Consider another scenario where the directors of a company have to vote on some controversial issue. Each of them wants their vote to be hidden from others since it may disrupt the friendly atmosphere prevailing in the company. In both of these cases, aggregate signcryption can be used to increase efficiency, provide secrecy, and decrease the communication overhead. Aggregate signcryption also has applications in military communication.

The concept of public key signcryption was proposed by Zheng [2]. The idea of this kind of cryptographic primitive is to perform encryption and signature in a single logical step to obtain confidentiality, integrity, authentication, and non-repudiation more efficiently than the sign-then-encrypt approach. Identity-based cryptosystems (IBCs) were first introduced by Shamir in 1984 [3]. IBCs eliminate trust problems

Manuscript received Apr. 10, 2011; revised Jan. 1, 2012; accepted Jan. 18, 2012.

This work was supported by the National Natural Science Foundation of China (61073188), China Postdoctoral Science Foundation (20100471355).

Xun-Yi Ren (phone: +86 13611586255, renxy@njupt.edu.cn), Zheng-Hua Qi (qizh@njupt.edu.cn), and Yang Geng (yangg@njupt.edu.cn) are with the Computer College, Nanjing University of Posts and Telecommunications, Jiang Su, China.

http://dx.doi.org/10.4218/etrij.12.0111.0215

encountered in certificate-based PKIs: There is no need to bind a public key to its owner's identity since they function as a singular element. Malone-Lee [4] developed the first identity-based signcryption scheme. Selvi and others [5] proposed an identity-based threshold signcryption scheme and formally proved its security in the existing security model. Muniz and Laud [6] proposed the first strong forward-secure identity-based signcryption scheme.

Gentry and Ramzan [7] proposed an efficient identity-based aggregate signature scheme. This scheme achieves both full aggregation and also a constant number of pairing operations during signature verification. Selvi and others [8], [9] analyzed the security in some of the existing aggregate signature schemes [10]-[12] and proposed two identity-based aggregate signature schemes. However, Selvi and others [9] presented the security model for unforgeability but not the security proof. The scheme proposed in [9] cannot be considered an identity-based system because the user's public key in Selvi and others' scheme is not an identity-based public key.

Selvi and others [13] proposed the first identity-based aggregate signcryption along with a formal security model and a formal security proof. However, where the aggregate signature V_{agg} in the "IBAS-1 Unsigncrypt" algorithm is the sum of unknown signature V_i, they did not explain how to recover V_i from V_{agg}.

In our previous paper [14], a new signcryption scheme (IBRSC) was presented based on identity and ring signcryption from pairings, but the essence of IBRSC is to ensure the anonymity of a user when they need to send a message confidentially and authentically to a specific receiver. The scheme cannot allow aggregation of n distinct signatures by n distinct users on n distinct messages.

In this paper, we propose an aggregate signcryption scheme (ASC) in which signature is a modification of the aggregate signature schemes in [9]. Also, we use the added advantage that identity-based cryptosystems provide an effective remedy to the key escrow problem, which is an inherent issue in IBC. In our ASC scheme, we eliminate the interaction among the senders (signers) before the signcryption generation, which reduces the communication complexity to a large extent. However, in this scheme, we are able to achieve only partial aggregation, not full aggregation. The ASC we propose can effectively improve computation and communication efficiency and has been formally proven to satisfy confidentiality and unforgeability in the random oracle model.

II. Preliminaries

1. Computation Assumptions

There are some computation assumptions about preliminaries related to an ASC, such as bilinear pairing, the bilinear Diffie-Hellman (BDH) problem, the decisional bilinear Diffie-Hellman (DBDH) problem, and the discrete logarithm (DL) problem. A bilinear pairing is a map $\hat{e}: G_1 \times G_1 \rightarrow G_2$ with the bilinearity, non-degeneracy, and computability properties, where G_1 is an additive cyclic group and G_2 is a multiplicative cyclic group of the same order q. Bilinearity means that given elements $P, R, Q \in G_1$, then $\hat{e}(P + Q, R) = \hat{e}(P, R)\, \hat{e}(Q, R)$ and $\hat{e}(P, Q + R) = \hat{e}(P, Q)\, \hat{e}(P, R)$. In particular, for $a, b \in \mathbb{Z}_q^*$, $\hat{e}(P^a, Q^b) = \hat{e}(P, Q)^{ab} = \hat{e}(P^{ab}, Q) = \hat{e}(P, Q^{ab})$. Non-degeneracy means that there exist $P, Q \in G_1$, such that $\hat{e}(P, Q) \neq I_{G2}$, where I_{G2} is the identity element of G_2. Computability means that there exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P, Q \in G_1$. The above properties can be derived from Weil or Tate pairing on an elliptic curve over a finite field [15]. For any probabilistic polynomial time algorithm $\mathcal{A}$, the BDH problem in G_1 is to compute $\hat{e}(g, g)^{abc}$, and the advantage in solving the BDH problem is defined as $Adv_{\mathcal{A}}^{BDH} = Pr[\mathcal{A}(g, g^a, g^b, g^c) = \hat{e}(P, P)^{abc} \mid a, b, c \in \mathbb{Z}_q^*]$. The DBDH problem is to decide if $w = \hat{e}(P, P)^{abc}$. And the advantage in solving DBDH problem is defined as $Adv_{\mathcal{A}}^{DBDH} = \left| Pr[\mathcal{A}(g, g^a, g^b, g^c, w, \hat{e}(P, P)^{abc})=1] - Pr[\mathcal{A}(g, g^a, g^b, g^c, w)=1] \right|$. The DL problem is to find x. And the advantage in solving the DL problem is defined as $Adv_{\mathcal{A}}^{DL} = Pr[\mathcal{A}(g, h) = x]$ when g and h are given.

2. Framework of ASC

An ASC consists of the following probabilistic polynomial time algorithms.

Setup(k). Given the security parameter of the system k, the private key generator (PKG) generates the set of public parameters π and the master secret key s of the system.

Key Extract(ID_i). Given an identity ID_i, the PKG, using the set of public parameters π and the master secret key s, computes the corresponding private key $<s_i, d_i>$, which is transmitted to ID_i in a secure way, and the public key $<X_i, q_i>$.

Signcrypt($m_i, X_i, d_i, \text{ID}_i, \text{ID}_B$). Let $\mathcal{M}$ be the message space, $\mathcal{W}$ - the signcrypted message space, and $\mathcal{R}$ - the space of senders. We will identify any member $X \in \mathcal{R}$ by its identity ID_X.

For any $m_i \in \mathcal{M}$, i $(1 \le i \le n)$ $(n \in \mathbb{Z}^+)$ is an arbitrary fixed integer, the algorithm *Signcrypt*($m_i, X_i, d_i, \text{ID}_i, \text{ID}_B$) is defined as follows:

The sender ID_i having a private key $<s_i, d_i>$ runs this algorithm to generate a signcryption on message m_i that will be aggregated and send it to a receiver with identity ID_B. The output is a cipertext $\sigma_i \in \mathcal{W}$.

Aggregate($\{\sigma_i, \text{ID}_i\}_{i=1,\ldots,n}$). Let $\mathcal{W}'$ be the aggregate signcrypted message space. Given a set of n signcryptions $\{\sigma_i\}_{i=1,\ldots,n}$ and the corresponding identity ID_i, this algorithm outputs the final aggregate signcryption $\sigma_{\mathrm{agg}}\in\mathcal{W}'$.

Unsigncrypt$(\sigma_{\mathrm{agg}}, s_{\mathrm{B}}, d_{\mathrm{B}})$. For any $\sigma_{\mathrm{agg}}\in\mathcal{W}'$ and a receiver with identity ID_{B} with the private key $<s_{\mathrm{B}}, d_{\mathrm{B}}>$, the algorithm *Unsigncrypt*$(\sigma_{\mathrm{agg}}, s_{\mathrm{B}}, d_{\mathrm{B}})$ is defined as follows:

The receiver ID_{B} receives σ_{agg} and runs *Unsigncrypt*$(\sigma_{\mathrm{agg}}, s_{\mathrm{B}}, d_{\mathrm{B}})$. If σ_{agg} is a valid aggregate signcryption from $\{\mathrm{ID}_i\}_{i=1,\dots,n}$ to ID_{B}, then the output is a plaintext $\{m_i\}_{i=1,\dots,n}$ $(m_i\in\mathcal{M})$; otherwise, the output is "*Invalid.*"

3. Formal Security Model for Aggregate Signcryption

Three security properties that are desired out of any ASC scheme are *message confidentiality*, *ciphertext authentication,* and *signature non-repudiation*.

Definition 1. An ASC is said to be semantically secure against indistinguishability under adaptive chosen ciphertext attack (IND-ASC-CCA2) if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

Start. The simulator $\mathcal{C}$ runs *Setup*(k) and sends the set of public parameters π to the adversary $\mathcal{A}$.

Phase 1. The adversary $\mathcal{A}$ makes a polynomially bounded number of queries to the simulator $\mathcal{C}$.

--*Keygen queries.* The adversary $\mathcal{A}$ produces an identity ID_i and obtains the corresponding secret key of ID_i.

--*Signcrypt queries.* $\mathcal{A}$ produces a message $m_i\in\mathcal{M}$, a signer identity ID_i, and a target identity ID_{B}. Then, $\mathcal{C}$ returns the signcrypted ciphertext $\sigma_i = \mathit{Signcrypt}(m_i, X_i, d_i, \mathrm{ID}_i, \mathrm{ID}_{\mathrm{B}})$ to $\mathcal{A}$, where the private key d_i is generated by querying the *Keygen* oracle.

--*Unsigncrypt queries.* $\mathcal{A}$ produces a receiver identity $\mathrm{ID}_{\mathrm{B}}\notin\{\mathrm{ID}_i\}_{i=1,\dots,n}$ and an aggregate signcryption σ_{agg}. The simulator $\mathcal{C}$ generates the private key s_{B} by querying the *Keygen* oracle. $\mathcal{C}$ returns the result of *Unsigncrypt*$(\sigma_{\mathrm{agg}}, s_{\mathrm{B}}, d_{\mathrm{B}})$ to $\mathcal{A}$. The result returned is $\perp$ if σ is an invalid signcrypted ciphertext from $\{\mathrm{ID}_i\}_{i=1,\dots,n}$ to ID_{B}.

Selection. $\mathcal{A}$ produces two messages sets m_{i0} and m_{i1} with equal length from the message space $\mathcal{M}$, identities $\{\mathrm{ID}_i\}_{i=1,\dots,n}$, and a final receiver identity $\mathrm{ID}_{\mathrm{B}}^*\notin\{\mathrm{ID}_i\}_{i=1,\dots,n}$, and sends them to $\mathcal{C}$. The adversary $\mathcal{A}$ must not have queried the private key corresponding to $\mathrm{ID}_{\mathrm{B}}^*\notin\{\mathrm{ID}_i\}_{i=1,\dots,n}$ in the first phase.

Challenge. The simulator $\mathcal{C}$ chooses randomly a bit $b^*\leftarrow\{0, 1\}$ and obtains the challenge aggregate signcryption σ_{agg} by running $\sigma_i^* = \mathit{Signcrypt}(m_{ib^*}, X_i, d_i, \mathrm{ID}_i, \mathrm{ID}_{\mathrm{B}}^*)$ and $\mathit{Aggregate}(\{\sigma_i^*, \mathrm{ID}_i\}_{i=1,\dots,n})$, and returns σ_{agg}^* to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ is allowed to make polynomially bounded number of new queries as in *Phase* 1 with the restrictions that it should not have made the *Unsigncryption queries* for the unsigncryption of σ_{agg}^* or the *Keygen queries* for the private keys of $\mathrm{ID}_{\mathrm{B}}^*$.

Response. $\mathcal{A}$ outputs a bit b' and wins the game if $b' = b^*$. The advantage of $\mathcal{A}$ is defined as

$$Adv_{\mathcal{A}}^{\text{IND-ASC-CCA2}} = |2Pr\,[\,b^* = b'\,] - 1\,|.$$

Definition 2. An ASC is said to be existentially ciphertext unforgeable under adaptive chosen message outsider attack, or AUTH-ASC-CMA2 secure, if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

Start. The simulator $\mathcal{C}$ runs *Setup*(k) and sends the set of public parameters π to the adversary $\mathcal{A}$.

Query. The adversary $\mathcal{A}$ makes a polynomially bounded number of queries to the simulator $\mathcal{C}$. The attack may be conducted adaptively, and allows the same queries as in the IND-ASC-CCA2 game, namely, *Keygen queries*, *Signcrypt queries*, and *Unsigncrypt queries*.

Forgery. $\mathcal{A}$ produces a new aggregate signcryption σ_{agg} sent from a set $\{\mathrm{ID}_i\}_{i=1,\dots,n}$ of n users on messages $\{m_i\}_{i=1,\dots,n}$ to a final receiver $\mathrm{ID}_{\mathrm{B}}\notin\{\mathrm{ID}_i\}_{i=1,\dots,n}$, where the private keys of the users in $\{\mathrm{ID}_i\}_{i=1,\dots,n}$ was not queried in query phase and σ_i is not the output of a previous query to the *Signcrypt queries*.

Outcome. The adversary $\mathcal{A}$ wins the game if $\perp$ is not returned by *Unsigncrypt*$(\sigma_{\mathrm{agg}}, s_{\mathrm{B}}, d_{\mathrm{B}})$.

Definition 3. An ASC is said to be existentially signature unforgeable against chosen message insider attack, or EUF-ASC-CMA secure, if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

Start. The simulator $\mathcal{C}$ runs *Setup*(k) and sends the set of public parameters π to the adversary $\mathcal{A}$.

Query. The adversary $\mathcal{A}$ makes a polynomially bounded number of queries to the simulator $\mathcal{C}$. The attack may be conducted adaptively, and allows the same queries as in the IND-ASC-CCA2 game, namely, *Keygen queries*, *Signcrypt queries*, and *Unsigncrypt queries*.

Forgery. The adversary returns a recipient identity ID_{B} and a ciphertext σ_i.

Outcome. The adversary $\mathcal{A}$ wins the game if: Under the private key of ID_{B}, the ciphertext σ_i is decrypted as a signed message $(\mathrm{ID}_i, \hat{m}_i, \hat{V}_i)$ that satisfies $\mathrm{ID}_i\neq\mathrm{ID}_{\mathrm{B}}$, $\mathrm{ID}_i\in\{\mathrm{ID}_i\}_{i=1,\dots,n}$; $\perp$ is not returned by *Unsigncrypt*$(\sigma_{\mathrm{agg}}, s_{\mathrm{B}}, d_{\mathrm{B}})$; provided that: (1) the private key of the user $\mathrm{ID}_i\in\{\mathrm{ID}_i\}_{i=1,\dots,n}$ was not queried in query phase; (2) σ_i is not the output of a previous query to the *Signcrypt* queries that involved m_i, ID_i, and recipient $\mathrm{ID}_{\mathrm{B}}'$, and resulted in a ciphertext σ_i' whose decryption under the private key of $\mathrm{ID}_{\mathrm{B}}'$ is the claimed forgery $(\mathrm{ID}_i, \hat{m}_i, \hat{V}_i)$.

III. ASC

We propose an ASC scheme in this section. We follow the framework of ASC that we presented in section II.2.

Setup(k). Given the security parameter of the system k, the PKG chooses two groups, G_1 and G_2, of the same prime order q, the generator g of G_1 and a bilinear map $\hat{e}$: $G_1 \times G_1 \rightarrow G_2$. The PKG then chooses four cryptographic hash functions H_0: $\{0, 1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$, H_1: $G_2 \rightarrow \{0, 1\}^l \times \mathbb{Z}_q^*$, H_2: $\{0, 1\}^l \times \{0, 1\}^* \times G_1 \times G_1 \times \{0, 1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$, H_3: $\{0, 1\}^l \times \{0, 1\}^* \times G_1 \times \mathbb{Z}_q^* \times G_1 \times \{0, 1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$, and $\mathcal{M} = \{0, 1\}^l$, where $l \in \mathbb{Z}^+$ is arbitrarily fixed. The PKG chooses its secret key $s \in \mathbb{Z}_q^*$. The set of system public parameters is $\pi = \{q, G_1, G_2, \hat{e}, g, g^s, H_0, H_1, H_2, H_3\}$.

Keygen(ID_i). For an identity $\mathrm{ID}_i \in \{0, 1\}^*$, the algorithm does the following:

(i) Chooses a random $x_i \in \mathbb{Z}_q$ and computes $X_i = g^{x_i}$, $q_i = H_0(\mathrm{ID}_i, X_i)$;

(ii) Computes $s_i = X_i^s$, $d_i = (x_i + sq_i) \bmod q$;

(iii) The PKG sends the corresponding private key $<s_i, d_i>$, which is transmitted in a secure way, and the public key $<X_i, q_i>$ to the user ID_i.

Signcrypt$(m_i, X_i, d_i, \mathrm{ID}_i, \mathrm{ID}_\mathrm{B})$. To send a message m_i to the receiver ID_B with the public key $<X_\mathrm{B}, q_\mathrm{B}>$, the sender ID_i with the private key $<s_i, d_i>$ and the public key $<X_i, q_i>$ does the following:

(i) Chooses a random $r_i \in \mathbb{Z}_q^*$ and computes $W_i = g^{r_i}$, $w_i = \hat{e}(g^s, X_B)^{r_i}$;

(ii) Lets $h_{1i} = H_1(w_i)$; $h_{2i} = H_2(m_i, \mathrm{ID}_i, X_i, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B})$;

(iii) Lets $h_{3i} = H_3(m_i, ID_i, X_i, h_{2i}, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B})$;

(iv) Computes $V_i = (r_i h_{2i} + h_{3i} d_i) \bmod q$;

(v) Sets $c_i = (m_i \| V_i) \oplus h_{1i}$; $Z_i = g^{V_i}$;

(vi) Outputs $\sigma_i = < c_i, W_i, Z_i, X_i >$ as the signcryption of ID_i on message m_i.

Aggregate$(\{\sigma_i, \mathrm{ID}_i\}_{i=1,\ldots,n})$. On input, a set of n signcryptions $\sigma_i = < c_i, W_i, Z_i, X_i >$, i=1 to n, and the corresponding identity ID_i (such that $\forall i$=1 to n, σ_i is the signcryption on message m_i by ID_i):

(i) Let $Z_\mathrm{agg} = \prod_{i=1}^{n} Z_i$;

(ii) Output the final aggregate signcryption $\sigma_\mathrm{agg} = <\{c_i, W_i, X_i, ID_i\}_{i=1,\ldots,n}, Z_\mathrm{agg}>$.

The aggregation can be done by any of the senders or by any third party.

Unsigncrypt$(\sigma_\mathrm{agg}, s_\mathrm{B}, d_\mathrm{B})$. On input, the aggregate signcryption $\sigma_\mathrm{agg} = <\{c_i, W_i, X_i, \mathrm{ID}_i\}_{i=1,\ldots,n}, Z_\mathrm{agg}>$, the receiver with identity ID_B, the public key $<X_\mathrm{B}, q_\mathrm{B}>$, and the private key $<s_\mathrm{B}, d_\mathrm{B}>$, do the following:

(i) Compute $w_i = \hat{e}(W_i, s_\mathrm{B})$, recover $m_i \| V_i = c_i \oplus H_1(w_i)$;

(ii) For i=1 to n, compute $h_{2i} = H_2(m_i, \mathrm{ID}_i, X_i, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B})$, $h_{3i} = H_3(m_i, \mathrm{ID}_i, X_i, h_{2i}, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B})$;

(iii) Check if $Z_\mathrm{agg} = g^{\sum_{i=1}^{n} V_i}$, and

$$Z_\mathrm{agg} = \prod_{i=1}^{n} (W_i)^{h_{2i}} \prod_{i=1}^{n} (X_i)^{h_{3i}} (g^s)^{\sum_{i=1}^{n} q_i h_{3i}}.$$

If the check succeeds, the output is m_i, which has been decrypted; otherwise, the output reads "Invalid."

IV. Analysis of Proposed Method

1. Correctness

Correctness of the unsigncryption algorithm:

$$\begin{aligned} w_i &= \hat{e}(W_i, s_\mathrm{B}) \\ &= \hat{e}(g^{r_i}, X_\mathrm{B}{}^s) = \hat{e}(g^s, X_\mathrm{B})^{r_i}, \\ Z_i &= g^{V_i} = g^{r_i h_{2i} + h_{3i} d_i} \\ &= (g^{r_i})^{h_{2i}} g^{h_{3i}(x_i + sq_i)} \\ &= (W_i)^{h_{2i}} (X_i)^{h_{3i}} (g^s)^{q_i h_{3i}}, \\ Z_\mathrm{agg} &= \prod_{i=1}^{n} Z_i = \prod_{i=1}^{n} ((W_i)^{h_{2i}} (X_i)^{h_{3i}} (g^s)^{q_i h_{3i}}) \\ &= \prod_{i=1}^{n} (W_i)^{h_{2i}} \cdot \prod_{i=1}^{n} (X_i)^{h_{3i}} \cdot (g^s)^{\sum_{i=1}^{n} q_i h_{3i}}. \end{aligned}$$

2. Proof of Confidentiality

Theorem 1. Assume there is an IND-ASC-CCA2 adversary $\mathcal{A}$ that is able to distinguish two valid ciphertexts during the game defined in Definition 1 with a non-negligible advantage and asks *Keygen queries*, *Signcrypt queries*, and *Unsigncrypt queries*; then, there exists a simulator $\mathcal{C}$ that can solve an instance of the DBDH problem with a non-negligible advantage.

Proof. The proof proceeds in the IND-ASC-CCA2 game. Assume that the simulator $\mathcal{C}$ receives a random DBDH problem instance (g, g^a, g^b, g^c, w); then, the goal is to decide whether $w = \hat{e}(g, g)^{abc}$ or not. $\mathcal{C}$ will run the adversary $\mathcal{A}$ as a subroutine and act as the adversary's challenger in the IND-ASC-CCA2 game.

Start. The simulator $\mathcal{C}$ sets the master public key $g^s = g^a$ and gives the system public parameters to $\mathcal{A}$.

Phase 1. The simulator $\mathcal{C}$ will set the random oracles of H_0, H_1, H_2, H_3, *Keygen*, *Signcrypt*, and *Unsigncrypt*. To maintain consistency, the simulator $\mathcal{C}$ will maintain four lists: H_0-List, H_1-List, H_2-List, and H_3-List, which will be detailed later. $\mathcal{C}$ will also simulate all oracles required during the game and control the H_0 random oracle. The adversary $\mathcal{A}$ outputs and threatens to attack the identity ID_B.

H_0 *Oracle*. When the H_0 oracle is queried with $\mathrm{ID}_i \in \{0,1\}^*$, $\mathcal{C}$ does the following: checks the H_0-List$< \mathrm{ID}_i, X_i, q_i, x_i>$, $x_i \in \mathbb{Z}_q^*$, if $\mathrm{ID}_i = \mathrm{ID}_\mathrm{B}$; selects a new random $\lambda_i \in \mathbb{Z}_q^*$; sets $X_i = g^b$, $q_i = \lambda_i$; adds the tuple $< \mathrm{ID}_i, X_i, q_i, * >$ to the H_0-List; and returns q_i. Otherwise, $\mathcal{C}$ selects a new random $\lambda_i \in \mathbb{Z}_q^*$, $x_i \in \mathbb{Z}_q$, sets $X_i = g^{x_i}$, $q_i = \lambda_i$, adds the tuple $< \mathrm{ID}_i, X_i, q_i, x_i >$ to the H_0-List, and returns q_i.

H_1 Oracle. When the H_1 oracle is queried with an input w_i, $\mathcal{C}$ checks the H_1-List. If there exists a tuple $<w_i, h_{1i}>$ in H_1-List, $\mathcal{C}$ returns h_{1i}. Otherwise, $\mathcal{C}$ selects a new random $h_{1i} \in \mathbb{Z}_q^*$, adds the tuple $<w_i, h_{1i}>$ to the H_1-List, and returns h_{1i}.

H_2 Oracle. When the H_2 oracle is queried with an input (m_i, ID_i, X_i, w_i, ID_B, X_B), $\mathcal{C}$ checks the H_2-List. If there exists a tuple $<m_i, \mathrm{ID}_i, X_i, W_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B}, h_{2i}>$ in the H_2-List, $\mathcal{C}$ returns h_{2i}. Otherwise, $\mathcal{C}$ chooses a new random $h_{2i} \in \mathbb{Z}_q^*$, adds the tuple $<m_i, \mathrm{ID}_i, X_i, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B}, h_{2i}>$ to H_2-List, and returns h_{2i}.

H_3 Oracle. When the H_3 oracle is queried with an input (m_i, ID_i, X_i, h_{2i}, w_i, ID_B, X_B), $\mathcal{C}$ checks the H_3-List. If there exists a tuple $<m_i, \mathrm{ID}_i, X_i, h_{2i}, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B}, h_{3i}>$ in the H_3-List, $\mathcal{C}$ returns h_{3i}. Otherwise, $\mathcal{C}$ chooses a new random $h_{3i} \in \mathbb{Z}_q^*$, adds the tuple $<m_i, \mathrm{ID}_i, X_i, h_{2i}, w_i, \mathrm{ID}_\mathrm{B}, X_\mathrm{B}, h_{3i}>$ to the H_3-List, and returns h_{3i}.

Keygen Oracle. When $\mathcal{A}$ makes a *Keygen query* with ID_i as the input, $\mathcal{C}$ checks the H_0-List to verify whether or not there is an entry for ID_i. If the H_0-List does not contain an entry for ID_i, return $\perp$. Otherwise, if $\mathrm{ID}_i = \mathrm{ID}_\mathrm{B}$, $\mathcal{C}$ recovers the tuple $< \mathrm{ID}_i, X_i, q_i, x_i >$ from the H_0-List and returns $< X_i, q_i, *, * >$; if $\mathrm{ID}_i \neq \mathrm{ID}_\mathrm{B}$, $\mathcal{C}$ recovers the tuple $<\mathrm{ID}_i, X_i, q_i, x_i>$ from the H_0-List and returns $< X_i, q_i, s_i, d_i >$, where $s_i = (g^a)^{x_i} = (g^{x_i})^a = (X_i)^a$, and $d_i \in \mathbb{Z}_q$ is randomly selected.

Signcrypt Oracle. When $\mathcal{A}$ makes a *Signcrypt query* with ID_i as the input, $\mathcal{C}$ checks the H_0-List to verify whether or not there is an entry for ID_i. If the H_0-List does not contain an entry for ID_i, return $\perp$. Otherwise, $\mathcal{C}$ executes *Signcrypt*(m_i, X_i, d_i, ID_i, ID_B) as usual and returns what the *signcrypt* algorithm returns.

Unsigncrypt Oracle. When $\mathcal{A}$ makes an *Unsigncrypt query* with $\sigma_\mathrm{agg} = <\{ c_i, W_i, X_i, \mathrm{ID}_i \}_{i=1,\ldots,n}, Z_\mathrm{agg} >$ and the receiver with identity ID_B, $\mathcal{C}$ first verifies whether or not there are entries for ID_i ($\mathrm{ID}_i \neq \mathrm{ID}_\mathrm{B}$) and ID_B in H_0-List and there is an entry of the form $<\mathrm{ID}_i, X_i, q_i, \lambda_i>$. If at least one of these conditions is not satisfied, $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ executes *Unsigncrypt*(σ_agg, s_B, d_B) in the normal way and returns what the *Unsigncrypt* algorithm returns.

Challenge. After getting sufficient training, $\mathcal{A}$ submits two equal length messages m_{i0} and m_{i1}. $\mathcal{C}$ randomly chooses a bit $b^* \leftarrow \{0, 1\}$ and obtains the challenge signcrypted ciphertext by running *Signcrypt*(m_{ib*}, X_i, d_i, ID_i, ID_B) and *Aggregate*($\{\sigma_i^*, \mathrm{ID}_i\}_{i=1,\ldots,n}$), then returns σ_agg^* to $\mathcal{A}$.

Phase 2. This phase is similar to *Phase* 1. However, in *Phase* 2, $\mathcal{A}$ cannot ask for *Unsigncrypt* on the challenge aggregate signcrypt $\sigma_\mathrm{agg}^* = <\{c_i, W_i, X_i, ID_i\}_{i=1,\ldots,n}, Z_\mathrm{agg}^*>$ or the *Keygen queries* for the secret keys of ID_B.

Output. After $\mathcal{A}$ has made a sufficient number of queries, $\mathcal{A}$ returns its guess: a bit b'. If $b' = b^*$, then $\mathcal{C}$ outputs 1 as the answer to the DBDH problem. Otherwise, it outputs 0. Since the adversary is denied access to the *Unsigncrypt* oracle with the challenge signcryption, for $\mathcal{A}$ to find that σ_i is not a valid ciphertext, $\mathcal{A}$ should have queried the *H_1 Oracle* with $w_i = \hat{e}(W_i, s_\mathrm{B})$. Here, s_B is the private key of the receiver, and it is $(X_\mathrm{B})^a = (g^b)^a = g^{ab}$. Also, $\mathcal{C}$ has set $W_i = g^c$. We have $w_i = \hat{e}(W_i, s_\mathrm{B}) = \hat{e}(g^c, g^{ab}) = \hat{e}(g, g)^{abc}$. □

3. Proof of Authentication

Theorem 2. The ASC proposed is secure against any probabilistic polynomial-time AUTH-ASC-CMA2 adversary $\mathcal{A}$ under the random oracle model if the DL problem is hard in G_1.

Proof. On getting a DL problem instance (g, $W_r = g^{r_r}$) and (g, g^{d_r}) as a challenge in the AUTH-ASC-CMA2 game defined in Definition 2, the simulator $\mathcal{C}$ uses $\mathcal{A}$ to solve the DL problem. The goal of $\mathcal{C}$ is to determine r_r and d_r. The simulator $\mathcal{C}$ gives the system public parameters to $\mathcal{A}$. $\mathcal{A}$ knows g^{d_r} from computing $W_r(g^s)^{q_r}$ ($g^{d_r} = g^{x_r + sq_r} = W_r(g^s)^{q_r}$). The proof of Theorem 2 is similar to the proof of Theorem 1, with some changes given in the following random oracles.

H_0 Oracle. When the H_0 oracle is queried with an input $\mathrm{ID}_i \in \{0,1\}^*$, $\mathcal{C}$ checks the H_0-List; if the tuple $< \mathrm{ID}_i, X_i, q_i, x_i >$ exists, $\mathcal{C}$ returns q_i. Otherwise, $\mathcal{C}$ selects new random $\lambda_i \in \mathbb{Z}_q^*$, $x_i \in \mathbb{Z}_q$, sets $X_i = g^{x_i}$, $q_i = \lambda_i$, adds the tuple $< \mathrm{ID}_i, X_i, q_i, x_i >$ to H_0-List, and returns q_i.

Keygen Oracle. When $\mathcal{A}$ makes a *Keygen query* with ID_i as the input, $\mathcal{C}$ checks the H_0-List to verify whether or not there is an entry for ID_i. If the H_0-List does not contain an entry for ID_i, return $\perp$. Otherwise, if $\mathrm{ID}_i \in \{\mathrm{ID}_i\}_{i=1,\ldots,n}$ (the corresponding senders identities set), $\mathcal{C}$ recovers the tuple $< \mathrm{ID}_i, X_i, q_i, x_i >$ from the H_0-List and returns $< X_i, q_i, *, * >$; if $\mathrm{ID}_i \notin \{\mathrm{ID}_i\}_{i=1,\ldots,n}$ (the corresponding senders identities set), $\mathcal{C}$ recovers the tuple $< \mathrm{ID}_i, X_i, q_i, x_i >$ from the H_0-List and returns $< X_i, q_i, s_i, d_i >$, where $s_i = (g^s)^{x_i}$, and $d_i \in \mathbb{Z}_q$ is randomly selected.

Forgery. $\mathcal{A}$ chooses the corresponding senders identities set $\{\mathrm{ID}_i\}_{i=1,\ldots,n}$ and the receiver identity ID_B and outputs a forged signcryption $\sigma_r^* = < c_r^*, W_r^*, Z_r^*, X_r^* >$ on message m_r^* from $\mathrm{ID}_r \in \{\mathrm{ID}_i\}_{i=1,\ldots,n}$ to $\mathcal{C}$. The simulator $\mathcal{C}$ retrieves the entry corresponding to ID_B in the H_0-List and uses S_B to execute *Unsigncrypt*(σ_agg^*, s_B, d_B). If σ_r^* is a valid signcryption from ID_r to receiver ID_B, that is, a message m_r^* is returned by the *Unsigncrypt* algorithm, then $\mathcal{C}$ applies the oracle replay technique to produce two valid signcryptions $\sigma_r' = (c_r', W_r', Z_r',$

X_r') and $\sigma_r'' = (c_r'', W_r'', Z_r'', X_r'')$ on message m_r from the ID$_r$ to receiver ID$_B$. This is achieved by running the turing machine again with the same random tape but with a different hash value. $\mathcal{C}$ obtains the 'signatures' $V_r' = r_r\, h_{2r}' + h_{3r}'d_r$ and $V_r'' = r_r h_{2r}'' + h_{3r}''d_r$ with $h_{2r}' \neq h_{2r}''$ and $h_{3r}' \neq h_{3r}''$. $\mathcal{C}$ successfully computes r_r and d_r. Indeed, from $V_r' = r_r\, h_{2r}' + h_{3r}'d_r$ and $V_r'' = r_r\, h_{2r}'' + h_{3r}''d_r$, we have

$$r_r = \frac{V_r' h_{3r}'' - V_r'' h_{3r}'}{h_{2r}' h_{3r}'' - h_{2r}'' h_{3r}'}, \quad h_{2r}' h_{3r}'' - h_{2r}'' h_{3r}' \neq 0,$$

$$d_r = \frac{V_r' h_{2r}'' - V_r'' h_{2r}'}{h_{3r}' h_{2r}'' - h_{3r}'' h_{2r}'}, \quad h_{3r}' h_{2r}'' - h_{3r}'' h_{2r}' \neq 0. \qquad \square$$

4. Proof of Non-repudiation

Theorem 3. The ASC proposed is secure against any probabilistic polynomial-time EUF-ASC-CMA adversary $\mathcal{A}$ under the random oracle model if the DBDH problem is hard in G_1.

Proof. The proof of Theorem 3 is similar to the proof of Theorem 2, with some changes in the following random oracles. We only provide the changes.

Keygen Oracle. If the H_0-List does not contain an entry for ID_i, $\mathcal{C}$ return $\perp$. Otherwise, if ID$_i$ =ID$_r$ (this corresponding identity ID$_r$ is called the "guessed" sender), $\mathcal{C}$ recover the tuple <ID$_i$, X_i, q_i, x_i > from the H_0-List and return < X_i, q_i,*,* >. If ID$_i \neq$ ID$_r$, recover the tuple < ID$_i$, X_i, q_i, x_i > from the H_0-List and return < X_i, q_i, s_i, d_i >, where $s_i = (g^s)^{x_i}$, and $d_i \in \mathbb{Z}_q$ is randomly selected.

Eventually, $\mathcal{A}$ returns a forgery, consisting of a ciphertext σ_i and a recipient identity ID$_B$. $\mathcal{C}$ decrypts the ciphertext for ID$_B$ (by invoking its own decryption oracle), which causes the plaintext forgery (ID$_i$, m_i, V_i) to be revealed. Note that if $\mathcal{C}$ has made the correct guess, that is, ID$_i$ = ID$_r$, then ID$_B \neq$ID$_r$ and the decryption works.

If σ_i is a valid signcryption from ID$_i$ to receiver ID$_B$, that is, a message m_i is returned by the *Unsigncrypt* algorithm, then $\mathcal{C}$ applies the oracle replay technique to produce two valid signed messages (ID$_i$, m_i, V_i) and (ID$_i$, m_i, V_i') on a message m_i from the ID$_i$ to receiver ID$_B$. This is achieved by running the turing machine again with the same random tape but with a different hash value. $\mathcal{C}$ obtains the 'signatures' $V_r' = r_r\, h_{2r}' + h_{3r}'d_r$ and $V_r'' = r_r\, h_{2r}'' + h_{3r}''d_r$ with $h_{2r}' \neq h_{2r}''$ and $h_{3r}' \neq h_{3r}''$. □

5. Efficiency

The primary objective of the aggregate signature scheme is to achieve both computation and communication efficiency. Using aggregate signature schemes, signatures from different users on different messages can be aggregated into a single compact signature. We eliminate the interaction among the senders (signers) before signcryption generation, which reduces the communication complexity to a large extent. Also, our ASC is more efficient than the sign-then-encrypt approach. We have compared our scheme (ASC) with the sign-then-encrypt scheme (AS and BF) in Table 1, where the aggregate signature scheme (AS) is proposed in [9] and the encryption scheme (BF) is proposed in [15]. In the comparison table, $|M|$ represents the length of a message. $|G_1|$ is the number of G_1 elements.

Table 1. Ciphertext size.

Scheme	Ciphertext size for sender
ASC	$\lvert M\rvert + \lvert Z_q^*\rvert + 3\lvert G_1\rvert$
AS and BF	$\lvert M\rvert + \lvert Z_q^*\rvert + 4\lvert G_1\rvert$

The major parameters involved in our ASC scheme are the computation costs for *Signcrypt* and *Unsigncrypt* operations. For computation cost, we mainly consider the number of pairing $\hat{e}$ computations performed, as they are the costliest operations involved. In our ASC scheme, the sender only performs one pairing $\hat{e}$ computation, and the receiver only performs n pairings $\hat{e}$ computations. In the IBAS scheme of Selvi and others [13], each sender only performs one pairing $\hat{e}$ computation, but the receiver performs $2n$+3 pairings $\hat{e}$ computations. Otherwise, the *Unsigncrypt* algorithm in Selvi and others' IBAS scheme [13] does not explain how to recover the unknown users' signatures from the aggregate signature. Thus, our ASC is more efficient than the IBAS scheme of Selvi and others [13].

We conducted five experiments for one pairing $\hat{e}$ operation using a pairing-based cryptography library [16], running each experiment 1,000 times. The results of the five experiments were 8.258133 s, 8.265714 s, 8.243305 s, 8.394295 s, and 8.384370 s. Therefore, the average time of one pairing $\hat{e}$ operation is about 0.008309 s.

V. Conclusion

We studied an ASC built upon the identity-based aggregate signature scheme proposed by Selvi and others [9]. Our proposed ASC was formally proven to be secure with respect to its IND-CCA2, AUTH-CMA2, and EUF-CMA security properties in a random oracle model. The ASC does not need the interaction among the signers, which is a requirement in existing efficient aggregate signature schemes, and it is efficient in pairing $\hat{e}$ computations.

Appendix:

In Table A1, we summarize all the notations of our paper.

Table A1. Definition and theorem.

Name	Content
Definition 1	An ASC is said to be semantically secure against indistinguishability under adaptive chosen ciphertext attack (IND-ASC-CCA2) if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.
Definition 2	An ASC is said to be existentially ciphertext unforgeable under adaptive chosen message outsider attack, or AUTH-ASC-CMA2 secure, if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.
Definition 3	An ASC is said to be existentially signature unforgeable against chosen message insider attack, or EUF-ASC-CMA secure, if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.
Theorem 1	Assume there is an IND-ASC-CCA2 adversary $\mathcal{A}$ that is able to distinguish two valid Ciphertexts during the game defined in definition 1 with non-negligible advantage and asking *Keygen queries*, *Signcrypt queries,* and *Unsigncrypt queries*, then there exists a simulator $\mathcal{C}$ that can solve an instance of the DBDH problem with non-negligible advantage.
Theorem 2	The ASC proposed is secure against any probabilistic polynomial-time AUTH-ASC-CMA2 adversary $\mathcal{A}$ under the random oracle model if the DL problem is hard in G_1.
Theorem 3	The ASC proposed is secure against any probabilistic polynomial-time EUF-ASC-CMA adversary $\mathcal{A}$ under the random oracle model if the DBDH problem is hard in G_1.

References

[1] D. Boneh et al., "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," *EUROCRYPT*, *LNCS*, vol. 2656, 2003, pp. 416-432.

[2] Y.L. Zheng, "Digital Signcryption or How to Achieve Cost (Signature&Encryption)<<Cost(Signature) + Cost(Encryption)," *CRYPTO*, *LNCS*, vol. 1294, 1997, pp. 165-179.

[3] A. Shamir, "Identity-Based Cryptosystem and Signature Schemes," *Proc. CRYPTO'84 Adv. Cryptology*, *LNCS*, vol. 196, 1985, pp. 47-53.

[4] J. Malone-Lee, "Identity-Based Signcryption," Cryptology ePrint Archive, Report 2002/098. http://eprint.iacr.org/2002/098

[5] S.S.D. Selvi et al., "Provably Secure ID-Based Broadcast Signcryption (IBBSC) Scheme," Cryptology ePrint Archive, Report 2008/225. http://eprint.iacr.org/2008/225

[6] M.G. Muniz and P. Laud, "Strong Forward Security in Identity-Based Signcryption," Cryptology ePrint Archive, Report 2011/156. http://eprint.iacr.org/2011/156

[7] C. Gentry and Z. Ramzan, "Identity-Based Aggregate Signatures," *Public Key Cryptography*, *LNCS*, vol. 3958, 2006, pp. 257-273.

[8] S.S.D. Selvi et al., "Security Analysis of Aggregate Signature and Batch Verification Signature Schemes," Cryptology ePrint Archive, Report 2009/290. http://eprint.iacr.org/2009/290

[9] S.S.D. Selvi et al., "Efficient and Provably Secure Identity Based Aggregate Signature Schemes with Partial and Full Aggregation," Cryptology ePrint Archive, Report 2010/461. http://eprint.iacr.org/2010/461

[10] S.-H. Seo et al., "Identity-Based Universal Designated Multi-Verifiers Signature Schemes," *Comput. Stand. Interfaces*, vol. 30, no. 5, 2008, pp. 288-295.

[11] Z. Wang et al., "Practical Identity-Based Aggregate Signature from Bilinear Maps," *J. Sci. Shanghai Jiao Tong University,* vol. 13, no. 6, 2008, pp. 684-687.

[12] Y. Wen and J. Ma, "An Aggregate Signature Scheme with Constant Pairing Operations," *CSSE*, vol. 3, 2008, pp 830-833.

[13] S.S.D. Selvi et al., "Identity Based Aggregate Signcryption Schemes," *INDOCRYPT*, *LNCS*, vol. 5922, 2009, pp. 378-397.

[14] Zheng-hua Qi et al., "An ID-Based Ring Signcryption Scheme for Wireless Sensor Networks," *IET Int. Conf. Wireless Sensor Netw.*, Beijing, China, Nov. 2010, pp. 368-373.

[15] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing," *SIAM J. Comput.*, vol. 32, no. 3, 2003, pp. 586-615.

[16] B. Lynn, "On the Implementation of Pairing-Based Cryptosystems," PhD thesis, Stanford, 2008.

Xun-Yi Ren received his BS in computer application from the University of Science and Technology, Beijing, China, in 1998, and his MS in management science and PhD in information networks, from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2004 and 2008, respectively. At present, he is an associate professor in the Department of Information Security, in the College of Computer, Nanjing University of Posts and Telecommunications, Nanjing, China. His current research interests include information security, grid & cloud computing, and machine learning.

Zheng-Hua Qi received her BS in computer application from the University of Science and Technology, Beijing, China, in 1998, and her MS in computer science from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2004. She is currently a PhD candidate in information security at the same university. Her main research interests include cryptography and information security.

Yang Geng received his PhD in computer science from Université Laval, Québec, Canada, in 1994. From 1995 to 1996, he worked with the Application Research Center, Canada University of Computing Technology, Montreal, Canada, as a postdoctoral researcher. He is currently a professor and doctorate supervisor in the College of Computer, Nanjing University of Posts and Telecommunications, Nanjing, China. His research interests include information security, parallel and distributed computing, and wireless sensor networks.