International Journal of Security and Its Applications
Vol.8, No.1 (2014), pp.411-422
http://dx.doi.org/10.14257/ijsia.2014.8.1.38

New Construction of Efficient Certificateless Aggregate Signatures

He Liu[1], Sijia Wang[2], Mangui Liang[1] and Yongqian Chen[1]

[1]*Institute of Information Science, Beijing Jiaotong University*
Beijing, 100044, P. R. China
[2]*Beijing Huada Infosec Technology Co.,Ltd*
Beijing, 100044, P. R. China
08112071@bjtu.edu.cn, acsafin@163.com, mgliang58@gmail.com,
09112078@bjtu.edu.cn

Abstract

This paper proposes a novel construction of efficient certificateless aggregate signature (CLAS) scheme. On basis of the computational Diffie-Hellman (CDH) assumption, the proposed scheme can be proven existentially unforgeable against adaptive chosen-message attacks. The new scheme also requires small constant pairing computations for aggregate verification, which is independent of the number of signers. Most importantly, a certain synchronization for aggregating randomness can be avoided by the proposed scheme. All the signers don't need to share the same synchronized clock to generate the aggregate signature, which greatly decreases the implementation complexity in many application scenarios.

Keywords: *provably secure; aggregate signature; certificateless cryptography; authentication*

1. Introduction

The concept of aggregate signature (AS) [1] was introduced by Boneh et al. at Eurocrypt 2003. Its basic idea is that anyone can aggregate n signatures on n distinct messages from n different users into a single AS, and the resulting AS's validity can ensure that the n original messages were indeed signed by the n users. These features can greatly reduce the signature length and the verification time, which is very suitable for wireless applications, *e.g.*, [2-4]. Since Boneh *et al.'s*, work, many AS schemes have been proposed *e.g.*, [5-8]. However, they are mostly based on either public key cryptography (PKC) or identity-based cryptography (ID-PKC) [9-11]. As is known, PKC suffers the heavy certificate management while ID-PKC has the inherent key escrow problem. Consequentially, the theory of AS couldn't avoid the complex certificate overhead and the serious key escrow, either.

To overcome the certificate costs in PKC and the key escrow issue in ID-PKC, Al-Riyami and Paterson [12] introduced certificateless public key cryptosystem (CL-PKC). Similar to ID-PKC, a CL-PKC scheme employs a trusted third party called Key Generation Center (KGC) to help a user to generate his private key. But the KGC only constructs partial private key for a user, and the user's full signing key also includes a secret value (selected by himself) besides the partial private key (obtained from the KGC). Meanwhile, a user can compute his public key with the public system parameters and the full secret information. Then, CL-PKC can solve the certificate overload as well as the key escrow problem. However, certificateless signature (CLS) is more meaningful in practice, and plenty of researches have also been presented, *e.g.*, [13-16].

ISSN: 1738-9976 IJSIA

To integrate the advantages of AS and CLS, a number of researchers have investigated certificateless aggregate signature (CLAS) schemes [17–21]. However, most of the existing CLAS schemes incur low computational efficiency or serious security problems. (1) The schemes of [17-19] have the linear increase of pairing computations in proportional to the number of signers, which goes against the AS's original goals. The scheme of [20] can support constant computations. But it still needs to compute five pairings, which is inefficient. (2) The schemes of [17, 18] have been proven to achieve weak security. Although the schemes of [19, 20] provide strong security, they require the involved signers to share the same synchronized clock. This is hard to achieve in practice. (3) Recently, the scheme of [21] has given an efficient CLAS scheme without a certain synchronization. Unfortunately, this scheme has been proven insecure against Type II adversaries by [22].

This paper proposes a new construction of efficient CLAS scheme. For security, the proposed scheme can be proven secure under the computational Diffie-Hellman (CDH) assumption in the random oracle model. As for efficiency, the proposed scheme requires small constant pairing computations during aggregate verification, which is independent of the number of singers. Most importantly, a certain synchronization for aggregating randomness is eliminated by the proposed scheme. All the signers don't require the same synchronized clock for the generation of AS, which is very useful in practice.

The rest of this paper is organized as follows: Section 2 provides some preliminaries. Section 3 gives the detailed CLAS construction. Section 4 proves the security analysis. Section 5 provides a comparison with the related works. Section 6 concludes this paper.

2. Preliminaries

2.1. Bilinear Maps

Let G_1 be an additive cyclic group of prime order q and G_2 be a multiplicative cyclic group of the same order. P is a generator of G_1. A bilinear map is a map $e: G_1 \times G_1 \to G_2$ with the following properties.

1) Bilinearity: for all $P, Q \in G$ and $a, b \in Z_q^*$, $e(aP, bQ) = e(P, Q)^{ab}$;

2) Non-degeneracy: $e(P, P) \neq 1$;

3) Computability: for all $P, Q \in G$, there exists an efficient algorithm to compute $e(P, Q)$.

2.2. Outline of CLAS Schemes

A typical CLAS scheme consists of six algorithms as given below.

Setup. On input a security parameter ℓ, the KGC runs this algorithm to generate the system parameters params and the master-key.

Partial-Private-Key-Extract. On input the master-key, a user's identity ID_i and the system parameters params, the KGC runs this algorithm to construct the user's partial private key D_i.

UserKeyGen. On input params and a user's identity ID_i, the user runs this algorithm, picks a random $x_i \in Z_q^*$ and outputs his secret value/public key x_i / P_i.

Sign. On input a message $M_i \in \mathcal{M}$, a user's identity ID_i, his full private key (x_i, D_i), his related public key P_i and params, the signer outputs a signature σ_i.

Aggregation. On input n individual message-signature pairs $\{(M_1,\sigma_1),...,(M_n,\sigma_n)\}$ signed by n users with their identities $\{ID_1,...,ID_n\}$ and the related public keys $\{P_1,...,P_n\}$, the aggregate signature generator outputs an AS σ.

Aggregate Verify. On input n distinct messages $\{M_1,...,M_n\}$, an AS σ, params, n users' identities $\{ID_1,...,ID_n\}$ and the related public keys $\{P_1,...,P_n\}$, the verifier runs this algorithm and outputs 1 if the signature is valid; otherwise, 0.

2.3. Complexity Assumption

Definition 1: Given the elements (P, aP, bP), for some random values $a, b \in Z_q^*$, the CDH problem is to compute abP.

2.4. Security Model of CLAS schemes

Two types of adversaries are considered in a CLAS scheme, *i.e.*, Type I adversaries and Type II adversaries. A Type I adversary A_1 can't access the KGC's master-key, but he can replace a user's public key. A Type II adversary A_2 can access the KGC's master-key, but he can't replace a user's public key. The security model for CLAS is defined as the following game between a challenger C and an adversary $A \in \{A_1, A_2\}$.

Setup. C runs this algorithm, takes a security parameter ℓ and publishes the system parameters params. Then,

- Type I: The challenger C keeps the master-key secret. For a user u_i with identity ID_i, A can request a partial private key of ID_i, and C responds D_i. A can pick a secret value x_i' and compute the related public key P_i'. C will take this replacement (x_i', P_i') as valid.
- Type II: C sends the master-key to A.

Queries. For ID_i, A can make a query to C on a message M_i. C responds the signature σ_i that is valid based on the user's public key and records the signed M_i.

Forgery. After the above operations, A outputs a valid AS σ^* which meets the verifying algorithm and there exists at least one message $M_i, i \in [1,n]$ which is not recorded by C.

Definition 2: We say that a CLAS scheme is Type I/II-secure, if there is no probabilistic polynomial time adversary A_1 / A_2 that can win the above games with non-negligible advantage.

3. The Proposed CLAS Scheme

This section constructs the new CLAS scheme.

Setup. Given a security parameter ℓ, the KGC generates the system parameters params and the master-key as follows:

1) Generate (G_1, G_2, e) where G_1 is an additive cyclic group of prime order q, G_2 is a multiplicative cyclic group of the same order and $e: G_1 \times G_1 \to G_2$ is a bilinear map.

2) Pick $\lambda \in Z_q^*$ as the master-key and two random generators P and S of G_1. Then set $P_T = \lambda P$.

3) Choose three cryptographic hash functions $H_0: \{0,1\}^* \to G_1, H_1: \{0,1\}^* \to Z_q^*$ and $H_2: \{0,1\}^* \to Z_q^*$.

4) Publish $\text{params} = (G_1, G_2, e, S, P, P_T, H_0, H_1, H_2)$.

Note: the message space is $\mathcal{M} = \{0,1\}^*$.

Partial-Private-Key-Extract. On input the master-key, a user's identity ID_i and params, the KGC generates the partial private key for the user.

1) Set $Q_i = H_0(ID_i)$.

2) Output the partial private key $D_i = \lambda Q_i$.

UserKeyGen. On input params and a user's identity ID_i, the user chooses a random $x_i \in Z_q^*$ and sets his secret value/public key as $x_i / P_i = x_i P$.

Sign. For using the full private key (x_i, D_i) to sign a message $M_i \in \mathcal{M}$, the signer (whose identity and public key are ID_i and P_i, respectively) performs the following steps:

1) Pick $r_i \in Z_q^*$ and set $R_i = r_i P$.

2) Compute $w_i = H_1(M_i \| P_i \| ID_i \| R_i), t_i = H_2(M_i \| P_i \| ID_i \| R_i)$.

3) Compute $U_i = D_i + w_i x_i S + t_i r_i P_T$.

4) Output the signature $\sigma_i = (R_i, U_i)$.

Aggregation. Anyone can play the role of aggregate signature generator that can aggregate a set of distinct signatures. On input n individual message-signature pairs $\{(M_1, \sigma_1 = (R_1, U_1)), ..., (M_n, \sigma_n = (R_n, U_n))\}$ signed by n users with their identities $\{ID_1, ..., ID_n\}$ and the related public keys $\{P_1, ..., P_n\}$, the aggregate signature generator computes $U = \sum_{i=1}^{n} U_i$ and outputs the AS $\sigma = (R_1, ..., R_n, U)$.

Aggregate Verify. Given an A $\sigma = (R_1, ..., R_n, U)$ signed by n users with $\{ID_1, ..., ID_n\}, \{P_1, ..., P_n\}$, the verifier performs the following steps:

1) Compute $w_i = H_1(M_i \| P_i \| ID_i \| R_i), t_i = H_2(M_i \| P_i \| ID_i \| R_i)$ and $Q_i = H_0(ID_i)$ for all $i (1 \le i \le n)$.

2) Check

$$e(U,P) \stackrel{?}{=} e(\sum_{i=1}^{n}(Q_i + t_i R_i), P_T)e(S, \sum_{i=1}^{n} w_i P_i) \quad (1)$$

If (1) holds, output 1; otherwise, 0.

4. Security Analysis

4.1. Correctness

$$e(U,P) = e(\sum_{i=1}^{n}(D_i + w_i x_i S + t_i r_i P_T), P)$$

$$= e(\sum_{i=1}^{n}(\lambda Q_i + \lambda t_i r_i P), P)e(\sum_{i=1}^{n} w_i x_i S, P)$$

$$= e(\sum_{i=1}^{n}(Q_i + t_i R_i), \lambda P)e(S, \sum_{i=1}^{n} w_i x_i P)$$

$$= e(\sum_{i=1}^{n}(Q_i + t_i R_i), P_T)e(S, \sum_{i=1}^{n} w_i P_i)$$

4.2. Security Proof

This section shows the security proof of the proposed scheme through Theorem 1 and 2.

Theorem 1. In the random oracle model, if a probabilistic polynomial-time Type I adversary A_1 has an advantage ε in forging a signature of the proposed CLAS scheme in an attack after running in time t and making at most q_{H_i} times H_i $(H_i = H_0, H_1, H_2)$ queries, q_K times Partial-Private-Key queries, q_P times Public-Key queries and q_S times Sign queries, then the CDH problem in G_1 can be solved within probability

$$\varepsilon' \geq \varepsilon / (q_k e + ne)$$

where n is the size of AS set.

Proof. Assume that there exists a Type I adversary A_1 who has an advantage in attacking the proposed CLAS scheme. Let C be a CDH attacker who receives a CDH instance (P, aP, bP) in G_1. Afterwards, the attaker C can interact with the adversary A_1 as modeled to solve the CDH problem (i.e. to compute abP) in the following game.

Setup. In the beginning, the attacker C sets $P_T = aP$ and picks the system parameters $\text{params} = (G_1, G_2, e, S, P, P_T, H_0, H_1, H_2)$. Then, C sends params to A_1.

Attack. These hash functions H_0, H_1, H_2 are considered as random oracles here. Suppose that A_1 can ask at most q_{H_i} times H_i $(H_i = H_0, H_1, H_2)$ queries, q_K times Partial-Private-Key queries, q_P times Public-Key queries and q_S times Sign queries. Then, A_1 can run the following queries in an adaptive way.

H_0 queries. C manages a list LH_0 of tuples $(ID_i, \alpha_i, Q_i, c_i)$ which is initially empty. On receiving an H_0 query on ID_i, C returns the same answer from LH_0 if it has been made before. Otherwise, C picks a random value $\alpha_i \in Z_q^*$, and then flips a coin $c_i \in \{0,1\}$ (that yields 0 with probability δ and 1 with probability $1-\delta$).

1) If $c_i = 0$, C set $Q_i = \alpha_i bP$, returns Q_i as answer and adds $(ID_i, \alpha_i, Q_i, c_i)$ to LH_0.

2) Else $c_i = 1$, C sets $Q_i = \alpha_i P$, returns Q_i as answer and adds $(ID_i, \alpha_i, Q_i, c_i)$ to LH_0.

H_1 queries. C manages a list LH_1 of tuples $(M_i, P_i, ID_i, R_i, w_i)$ which is initially empty. On receiving an H_1 query, C returns the same answer from LH_1 if it has been made before. Otherwise, C picks a random $w_i \in Z_q^*$, returns w_i as answer and adds $(M_i, P_i, ID_i, R_i, w_i)$ to LH_1.

H_2 queries. C manages a list LH_2 of tuples $(M_i, P_i, ID_i, R_i, t_i)$ that is initially empty. On receiving an H_2 query, C returns the same answer from LH_2 if it has been made before. Otherwise, C picks a random $t_i \in Z_q^*$, returns t_i as answer and adds $(M_i, P_i, ID_i, R_i, t_i)$ to LH_1.

Partial-Private-Key queries. C manages a list K^{List} of tuples (ID_i, x_i, D_i, P_i) which is initially empty. On receiving a Partial-Private-Key query (i.e. $PPK(ID_i)$), C returns the same answer from K^{List} if it has been made before. Otherwise, C first make an H_0 query to recover the corresponding tuple $(ID_i, \alpha_i, Q_i, c_i)$ from LH_0, and then does as below.

1) If $c_i = 0$, C aborts.

2) Else if there is a tuple (ID_i, x_i, D_i, P_i) on K^{List}, C sets $D_i = \alpha_i P_T$ and returns D_i.

3) Otherwise, C computes $D_i = \alpha_i P_T$, sets $x_i = P_i = \perp$, returns D_i as answer and adds (ID_i, x_i, D_i, P_i) to K^{List}.

Public-Key queries. When receiving a Public-Key query $PK(ID_i)$, C returns the same answer from K^{List} if the query has been made before. Otherwise, C does as below.

1) If there's a tuple (ID_i, x_i, D_i, P_i) on K^{List} (where the public key P_i is $\perp$ in this case), C selects a random $x_i' \in Z_q^*$, sets $P_i' = x_i' P$, returns P_i' as answer and updates (ID_i, x_i, D_i, P_i) to (ID_i, x_i', D_i, P_i').

2) Else, C picks $x_i \in Z_q^*$, computes $P_i = x_i P$, returns P_i as answer, sets $D_i = \perp$ and adds (ID_i, x_i, D_i, P_i) to K^{List}.

Secret-Value queries. On receiving a Secret-Value query $SV(ID_i)$, C firstly makes $PK(ID_i)$ to recover (ID_i, x_i, D_i, P_i) from K^{list}, and then returns x_i.

Public-Key-Replacement queries. On receiving a $PKR(ID_i, P_i')$ query, C first searches (ID_i, x_i, D_i, P_i) on K^{list}. If there's such a tuple and $P_i \neq \perp$, C updates P_i to P_i'; otherwise, C makes $PK(ID_i)$ and updates P_i to P_i'.

Sign queries. For a $Sign(ID_i, M_i, P_i)$ query, C makes $H_0, ..., H_2$ queries to recover $(ID_i, \alpha_i, Q_i, c_i)$ from LH_0, $(M_i, P_i, ID_i, R_i, w_i)$ from LH_1 and $(M_i, P_i, ID_i, R_i, t_i)$ from LH_2, then constructs the signature.

1) If $c_i = 0$, C picks $r_i \in Z_q^*$, sets $R_i = r_i P - t_i^{-1} \alpha_i bP$, computes $U_i = w_i x_i S + t_i r_i P_T$ and outputs the signature $\sigma_i = (R_i, U_i)$.

2) If $c_i = 1$, C picks $r_i \in Z_q^*$, sets $R_i = r_i P$, computes $U_i = \alpha_i P_T + w_i x_i S + t_i r_i P_T$ and outputs the signature $\sigma_i = (R_i, U_i)$.

Forgery. A_1 outputs a forged AS $\sigma^* = (R_1^*, ..., R_n^*, U^*)$ on the message set $L_m^* = \{m_1^*, ..., m_n^*\}$, the identity set $L_{ID}^* = \{ID_1^*, ..., ID_n^*\}$ of n users and the related public key set $L_P^* = \{P_1^*, ..., P_n^*\}$. Ahead, C recovers $(ID_i, \alpha_i, Q_i, c_i)$ from LH_0, $(M_i, P_i, ID_i, R_i, w_i)$ from LH_1 and $(M_i, P_i, ID_i, R_i, t_i)$ from LH_2. Also, it's required that there exists an $i \in [1, n]$ so that $c_i^* = 0$ and A_1 has never made a $Sign(ID_i^*, M_i^*, P_i^*)$ query. Without loss of generality, we let $i = 1$ here. Then, the forged AS σ^* must satisfy

$$e(U^*, P) = e(\sum_{i=1}^{n}(Q_i^* + t_i^* R_i^*), P_T)e(S, \sum_{i=1}^{n} w_i^* P_i^*) \quad (2)$$

Otherwise, C aborts.

If (2) holds, we can have $e(Q_1^* + t_1^* R_1^*, P_T) = (e(\sum_{i=2}^{n}(Q_i^* + t_i^* R_i^*), P_T)e(S, \sum_{i=1}^{n} w_i^* P_i^*))^{-1} e(U^*, P)$. According to our settings, $Q_1^* = \alpha_1^* bP$; for all $i (i \in [2, n])$, $Q_i^* = \alpha_i^* P$. Then, we can get

$$(U^* - \sum_{i=2}^{n}(\alpha_i^* P_T + t_i^* r_i^* P_T) - \sum_{i=1}^{n} w_i^* x_i^* S - t_1^* r_1^* P_T)\alpha_1^{*-1} = abP$$

To finish the proof, we'll show that C can solve the given instance of CDH problem with the probability at least $\varepsilon' \geq \varepsilon / (q_k e + ne)$.

Similar to the analysis in [19, 20], we evaluate C 's success probability. Firstly, three events are given as below

- E1: C doesn't abort as a result of A_1's $PPK(ID_i)$ queries.
- E2: C generates a valid and nontrivial AS forgery.
- E3: Event E2 occurs, it's required that $c_1^* = 0$ and $c_i^* = 0$ for all $i \in [2, n]$.

C succeeds if all the events occur. Then, we can easily get the probability $\Pr[E1 \wedge E2 \wedge E3] = \Pr[E1]\Pr[E2|E1]\Pr[E3|E2 \wedge E1]$. According to [19, 20], we can get the probability at least $(1-\delta)^{q_k}$ for C not to fail in $PPK(ID_i)$ queries (*i.e.*, $\Pr[E1]$), the probability at least ε for C not to fail in sign queries based on the success of $PPK(ID_i)$ queries (*i.e.*, $\Pr[E2|E1]$) and the probability at least $\delta(1-\delta)^{n-1}$ for C not to fail in sign queries based on the success of E1 and E2 (*i.e.*, $\Pr[E3|E2 \wedge E1]$). As a result, we can obtain $\varepsilon' = \Pr[E1 \wedge E2 \wedge E3] \geq \delta(1-\delta)^{q_K + n - 1}\varepsilon$. When q_K is large enough and $\delta = 1/(q_K + n)$, $\delta(1-\delta)^{q_K + n - 1}\varepsilon$ can be maximized as $\varepsilon / (q_K e + ne)$, *i.e.*, $\varepsilon' \geq \varepsilon / (q_K e + ne)$.

Hence, if a Type I adversary that can break the proposed CLAS scheme exists, an attacker that can solve the CDH problem exists.

Theorem 2. In the random oracle model, if a probabilistic polynomial-time Type II adversary A_2 has an advantage ε in forging a signature of the proposed CLAS scheme in an attack after running in time t and making at most q_{H_i} times H_i $(H_i = H_0, H_1, H_2)$ queries, q_K times Partial-Private-Key queries, q_P times Public-Key queries and q_S times Sign queries, then the CDH problem in G_1 can be solved within probability

$$\varepsilon' \geq \varepsilon / (q_k e + ne)$$

where n is the size of AS set.

Proof. Assume that there exists a Type II adversary A_2 who has an advantage in attacking the proposed CLAS scheme. Let C be a CDH attacker who gets a CDH instance (P, aP, bP) in G_1. Afterwards, C interacts with A_2 as modeled to solve the CDH problem (i.e. to compute abP) in the following game.

Setup. Firstly, C picks $\lambda \in Z_q^*$ as the master-key and sets $P_T = \lambda P$ and $S = aP$. After that, C randomly chooses the system parameters $\text{params} = (G_1, G_2, e, S, P, P_T, H_0, H_1, H_2)$. Once the game is started, A_2 is given the master-key λ and params. Due to the access of λ, A_2 can make Partial-Private-Key queries himself.

Attack. These hash functions H_0, H_1, H_2 are considered as random oracles here. Suppose that A_2 can ask at most q_{H_i} times H_i $(H_i = H_0, H_1, H_2)$ queries, q_K times Partial-Private-Key queries, q_P times Public-Key queries and q_S times Sign queries. A_2 can run the following queries in an adaptive way.

H_0 queries. C manages a list LH_0 of tuples (ID_i, Q_i) that is initially empty. On receiving an H_0 query on ID_i, C returns the same answer from LH_0 if it has been made before. Otherwise, C picks a random $Q_i \in G_1$, returns Q_i and add (ID_i, Q_i) to LH_0.

H_1 queries. C manages a list LH_1 of tuples $(M_i, P_i, ID_i, R_i, w_i)$ that is initially empty. On receiving an H_1 query, C returns the same answer from LH_1 if it has been made before. Otherwise, C picks $w_i \in Z_q^*$, returns w_i as answer and adds $(M_i, P_i, ID_i, R_i, w_i)$ to LH_1.

H_2 queries. C manages a list LH_2 of tuples $(M_i, P_i, ID_i, R_i, t_i)$ that is initially empty. On receiving an H_2 query, C returns the same answer from LH_2 if it has been made before. Otherwise, C picks a random $t_i \in Z_q^*$, returns t_i as answer and adds $(M_i, P_i, ID_i, R_i, t_i)$ to LH_1.

Public-Key queries. C manages a list K^{List} of tuples (ID_i, x_i, P_i, c_i) that is initially empty. When receiving a Public-Key query $PK(ID_i)$, C returns the same answer from K^{List} if the query has been made before. Otherwise, C picks a random value $x_i \in Z_q^*$, flips a coin $c_i \in \{0,1\}$ (that yields 0 with probability δ and 1 with probability $1-\delta$) and does the following steps.

1) If $c_i = 0$, C returns $P_i = x_i bP$ as answer and adds (ID_i, x_i, P_i, c_i) to K^{List}.

2) Else $c_i = 1$, C returns $P_i = x_i P$ as answer and adds (ID_i, x_i, P_i, c_i) to K^{List}.

Secret-Value queries. On receiving a Secret-Value query $SV(ID_i)$, C firstly makes $PK(ID_i)$ to recover (ID_i, x_i, D_i, P_i) from K^{list}. If $c_i = 0$, C aborts. Otherwise, C returns x_i as answer.

Sign queries. For a $Sign(ID_i, M_i, P_i)$ query, C makes $H_0, ..., H_2$ queries to recover (ID_i, Q_i) from LH_0, $(M_i, P_i, ID_i, R_i, w_i)$ from LH_1 and $(M_i, P_i, ID_i, R_i, t_i)$ from LH_2, then constructs the signature.

1) If $c_i = 0$, C picks a random $r_i \in Z_q^*$, sets $R_i = r_i P - t_i^{-1} w_i x_i bP$, computes $U_i = \lambda Q_i + t_i r_i P_T$ and returns $\sigma_i = (R_i, U_i)$.

2) Else $c_i = 1$, C picks a random $r_i \in Z_q^*$, sets $R_i = r_i P$, computes $U_i = \lambda Q_i + w_i x_i S + t_i r_i P_T$ and returns $\sigma_i = (R_i, U_i)$.

Forgery. A_2 outputs a forged AS $\sigma^* = (R_1^*, ..., R_n^*, U^*)$ on the message set $L_m^* = \{m_1^*, ..., m_n^*\}$, the identity set $L_{ID}^* = \{ID_1^*, ..., ID_n^*\}$ of n users and the related public key set $L_P^* = \{P_1^*, ..., P_n^*\}$. Ahead, C recovers (ID_i, Q_i) from LH_0, $(M_i, P_i, ID_i, R_i, w_i)$ from LH_1 and $(M_i, P_i, ID_i, R_i, t_i)$ from LH_2. Also, it's required that there exists an $i \in [1, n]$ so that $c_i^* = 0$ and A_1 has never made a $Sign(ID_i^*, M_i^*, P_i^*)$ query. Without loss of generality, we let $i = 1$ here. Then, the forged AS σ^* must satisfy

$$e(U^*, P) = e(\sum_{i=1}^{n}(Q_i^* + t_i^* R_i^*), P_T) e(S, \sum_{i=1}^{n} w_i^* P_i^*) \quad (3)$$

Otherwise, C aborts.

If (3) holds, we have $e(S, w_1^* P_1^*) = e(U^*, P)\ (e(\sum_{i=1}^{n}(Q_i^* + t_i^* R_i^*), P_T) e(S, \sum_{i=2}^{n} w_i^* P_i^*))^{-1}$. According to our settings, $P_1^* = x_1^* bP$; for all $i (i \in [2, n])$, $P_i^* = x_i^* P$. We can get

$$(U^* - \sum_{i=1}^{n}(\lambda Q_i^* + t_i^* r_i^* P_T) - \sum_{i=2}^{n} w_i^* x_i^* S)(x_1^* w_1^*)^{-1} = abP$$

To finish the proof, we'll show that C can solve the given instance of CDH problem with the probability at least $\varepsilon' \geq \varepsilon / (q_k e + ne)$.

Similar to the analysis in [19, 20], we evaluate C's probability of success. Firstly, three events are given as below.

- E1: C doesn't abort as a result of A_2's $SV(ID_i)$ queries.
- E2: C generates a valid and nontrivial AS forgery.
- E3: Event E2 occurs, it's required that $c_1^* = 0$ and $c_i^* = 0$ for all $i \in [2, n]$.

C succeeds if all the above events occur. Then, we can easily get the probability $\Pr[E1 \wedge E2 \wedge E3] = \Pr[E1]\Pr[E2|E1]\Pr[E3|E2 \wedge E1]$. According to [19, 20], we can get the probability at least $(1-\delta)^{q_k}$ for C not to fail in $SV(ID_i)$ queries (*i.e.*, $\Pr[E1]$), the probability at least ε for C not to fail in sign queries based on the success of $SV(ID_i)$ queries (*i.e.*, $\Pr[E2|E1]$) and the probability at least $\delta(1-\delta)^{n-1}$ for C not to fail in sign queries based on the success of E1 and E2 (*i.e.*, $\Pr[E3|E2 \wedge E1]$). As a result, we can obtain $\varepsilon' = \Pr[E1 \wedge E2 \wedge E3] \geq \delta(1-\delta)^{q_K + n - 1}\varepsilon$. When q_K is large enough and $\delta = 1/(q_K + n)$, $\delta(1-\delta)^{q_K + n - 1}\varepsilon$ can be maximized as $\varepsilon / (q_K e + ne)$, i.e. $\varepsilon' \geq \varepsilon / (q_K e + ne)$.

Hence, if a Type II adversary that can break the proposed CLAS scheme exists, an attacker that can solve the CDH problem exists.

5. Comparison

This section compares the proposed CLAS scheme with the related works [18-21]. (Note that pairing computation is the most time consuming while some pre-computed values are omitted here.) Table 1 provides the comparison results. It can be seen that, compared to [18-

20], the proposed CLAS scheme is less efficient than [18-20] in AS length but wins the most efficient aggregate verification owing to only three involved pairings. Meanwhile, the proposed scheme doesn't require the synchronized clock. As for [21], the proposed scheme almost share the same efficiency but wins more security, because [21] is insecure against type II adversary. In fact, the practicability of a CLAS scheme mainly depends on two factors, *i.e.*, aggregate verifying efficiency and security level. Then, according to the above analysis, we claim that our CLAS is secure and efficient.

Table 1. Comparison of the Schemes

scheme	Security Proof	a certain synchronization	Signing costs	Aggregate Verifying costs	AS length
Scheme 1 in [18]	wI,wII	No	2S,H	(4n+1)P,2nH	(n+1)G
Scheme 2 in [18]	wI,wII	No	3S,H	(3n+2)P,nS,3nH	2G
[19]	sI, sII	Yes	3S,2H	(n+31)P,(2n+1)H	(n+1)G
[20]	sI, sII	Yes	5S,3H	5P,2nS,(2n+3)H	2G
[21]	sI, wII	No	3S,H	3P,2nS,(2n+2)H	(n+1)G
Our scheme	sI, sII	No	3S	3P,2nS,nH	(n+1)G

wI/wII: weak security proof under type I/II adversary; sI/sII: strong security proof under type I/II adversary;

S: Scale Multiplication in G_1 ; H: MapToPoint Hash; P: Pairing ; G: a point length in G_1 .

6. Conclusions

This paper presented a novel CLAS scheme based on the CDH problem in the oracle model. The new scheme has many advantages, such as efficient aggregate verification, strong security, *etc*. In particular, the proposed scheme doesn't require a certain synchronization. A comparison of the proposed scheme with the related works is also provided, which shows that the proposed scheme is secure and efficient.

Acknowledgements

This work was supported (1) the State 863 Plan under grant 2007AA01Z203; (2) Beijing Jiaotong University under grant 2006XZ002; (3) China 973 program 2007CB307101; (4) the Natural Grant 60772039.

References

[1] D. Boneh, C. Gentry, B. Lynn and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps", Proc. EUROCRYPT 03, LNCS 3027, **(2003)** pp. 416-432.
[2] D. He, C. Chen, S. Chan and J. Bu, "Secure and Efficient Handover Authentication Based on Bilinear Pairing Functions", IEEE Transactions on Wireless Communications, vol. 11, no. 1, **(2012)**, pp. 48-53.
[3] C. Sur, Y. Park and K. Sakurai, "Providing Secure Location-Aware Services for Cooperative Vehicular Ad Hoc Networks", Journal of Internet Technology, vol. 13, no. 4, **(2012)**, pp. 631-643.
[4] A. Wasef, Y. Jiang and X. Shen, "DCS: An Efficient Distributed-Certificate-Service Scheme for Vehicular Networks', IEEE Transactions on Vehicular Technology, vol. 59, no. 2, **(2010)**, pp. 533-549.
[5] S. Lim, E. Lee and CM Park, "A short redactable signature scheme using pairing", Security and Communication Networks, DOI: 10.1002/sec.346, vol. 5, no. 5, **(2012)**, pp. 523-534.
[6] A. Bagherzandi, S. Jarecki, "Identity-based aggregate and multi-signature schemes based on RSA", P.Q. Nguyen, D. Pointcheval (Eds.), Public Key Cryptography (PKC '2010), LNCS 6056, **(2010)**, pp. 480-498.
[7] A. Lysyanskaya, S. Micali, L. Reyzin and H. Shacham, "Sequential aggregate signatures from trapdoor permutations", EUROCRYPT 2004, LNCS 3027, **(2004)**, pp. 74-90.
[8] G. Neven, "Efficient sequential aggregate signed data", N. Smart (Ed.), Advances in Cryptology-EUROCRYPT 2008, LNCS, vol. 4965, Springer Verlag, **(2008)**, pp. 52-69.

[9] A. Shamir, "Identity-based cryptosystems and signature schemes", Proc. CRYPTO 84 on Advances in cryptology (1984), LNCS, 196, **(1984)**, pp. 47-53.
[10] H. Liu, S. J. Wang and M. Liang, "New Construction of Efficient Hierarchical Identity Based Signature in the Standard Model", International Journal of Security and Its Applications, In press, **(2013)**.
[11] H. Liu and M. Liang, "Efficient identity-based hierarchical access authentication protocol for mobile network", Security Comm. Networks. in press. doi: 10.1002/sec.412, **(2012)**.
[12] S. Al-Riyami and K. Paterson, "Certificateless public key cryptography", Proc. ASIACRYPT 2003, **(2003)**, pp. 452-473
[13] X. Huang, "On the security of a Certificateless signature scheme", Proc. CANS 2005, LNCS, **(2005)**, pp. 13-25.
[14] K. Y. Choi, J. H. Park and D. H. Lee, "A new provably secure certificateless short signature scheme", Computers & Mathematics with Applications, vol. 61, **(2011)**, pp. 1760-1768.
[15] X. Ren, "Provably Secure Aggregate Signcryption Scheme", ETRI Journal, vol. 34, no. 3, **(2012)**, pp. 421-428.
[16] X. Huang, "Certificateless Signatures: New Schemes and Security Models", The Computer Journal, vol. 55, **(2012)**.
[17] R. Castro and R. Dahab, "Efficient certificateless signatures suitable for aggregation", Cryptology ePrint Archive <http://eprint.iacr.org/2007/454>.
[18] Z. Gong, Y. Long, X. Hong and K. Chen, "Two certificateless aggregate signatures from bilinear maps", Proc. IEEE SNPD 2007, vol. 3, **(2007)**, pp. 188-193.
[19] L. Zhang and F. Zhang, "A New Certificateless Aggregate Signature Scheme", Computer Communications, vol. 36, **(2009)**, pp. 1079-1085.
[20] L. Zhang, "Efficient many-to-one authentication with certificateless aggregate signatures", Computer Networks, vol. 54, **(2010)**, pp. 2482-2491.
[21] H. Xiong, Z. Guan, Z. Chen and F. Li, "An Efficient certificateless aggregate signature with constant pairing computations", Information Science, vol. 219, **(2013)**, pp. 225–235.
[22] D. He and M. Tian, "A note on 'An efficient certificateless aggregate signature with constant pairing computations", Cryptology ePrint Archive.<https://eprint.iacr.org/2012/445.pdf.

Authors

He Liu received his B.S. degree in Computer Science from Beijing Jiaotong University in 2008. From 2008, he is a Master-Doctoral Program student in Institute of Information Science, Department of Computer Science, Beijing Jiaotong University. His research interests include many areas of information security.

Sijia Wang received the B.S and M.S. degree from Beijing Jiaotong University. From 2008, he works at Beijing Huada Infosec Technology Co.,Ltd as the R&D Engineer. His research interests include many areas of network security.

Mangui Liang received the B.S. degree in Electrical Engineering from NCEPU in 1982, M.S. and Ph.D. in Telecommunication from BJTU in 1985 and 1988 respectively. During 1989-2009, he stayed in BJTU and now is a professor in Information and Communications Technology. He is the member of the council of CIE, the member of the board of ASC and the director of SHMC which is a branch of ASC.

Yongqian Chen received the B.S. degree in Dalian University of Technology in 2001, he receive M.S. degree in Inha University, Korea in 2006. From 2009, he is a Phd candidate in Institute of Information Science, Beijing Jiaotong University. His research interests consists of networking performanc analysis, routing protoco performance analysis, Qos.