A new certificateless aggregate signature scheme

Yao Xiao

January 3, 2020

1 Preliminaries

1.1 Assumptions

Definition 1. Computational Diffie-Hellman(CDH)Problem:Let G be an additive cyclic group with generator P. Given $P, aP, bP \in \mathbb{G}$,for any random numbers $a, b \in \mathbb{Z}_q^*$ compute abP There is an probabilistic polynomial time(PPT) solvable algorithm $\mathcal{A}$ has negligible advantage ϵ in solving CDH problem in $\mathbb{G}$ if the $\Pr[\mathcal{A}(P, aP, bP) = abP] \leq \epsilon$, where ϵ is a small positive integer and the probability is over the selection of $P \in \mathbb{G}$, random numbers $a, b\mathbb{Z}_q^*$ and the security parameter 1^μ. This can be formally presented by the following definition.
Definition 2. Computational Diffe-Hellman(CDH) Assumption: The assumption (t, ϵ) CDH holds in $\mathbb{G}$ if there does not exist any PPT algorithm with running time t has advantage ϵ in solving CDH problem.

1.2 Framework of certificateless aggregate signature

An aggregate signature scheme is a signature scheme which allows an efficient algorithm to aggregate n signatures on n distinct messages from n distinct users into one single signature. The validity of an aggregate signature will convince a verifier that the n users did indeed sign the n original messages.
A CLAS scheme involves a KGC, an aggregating set $\mathcal{U}$ of n users $U_1, ..., U_n$ and an aggregate signature generator. It consists of six algorithms: Setup, Partial-Private-Key-Extract, UserKeyGen, Sign, Aggregate and Aggregate Verify. The description of each algorithm is as follows:
There are 6 parts, the setup is executed by KGC, which accepts security parameters ℓ to generate a master key and a system parameter list. As for aggregate, run by the collective signature generator takes as input the state information Δ, a set $\mathcal{U}$ of n users $U_1, ..., U_n$, the identity ID_i of each user U_i, and the corresponding public key P_i. A signature U_i is present on a message M_1 with status information Δ under the identity ID_i and public key P_i of each user U_i. The output of this algorithm is the aggregated signature σ on messages $M_1, ..., M_n$.

1.3 Adversarial mode of certificateless aggregate signature

Security of CLAS scheme is defined through two games between the adversary $\mathcal{A}_I/\mathcal{A}_{II}$ and challenger $\mathcal{C}$. The two games are defined as:
For Game 1, the adversary $\mathcal{A}_I$ can perform a polynomially bounded number of the following types of queries in an adaptive manner, and following method to win game: σ^* is a valid aggregate signature on messages $M_1^*, ..., M_n^*$ with state information δ^* chosen by $\mathcal{A}_I$.
For Game 2, the adversary $\mathcal{A}_{II}$ can perform a polynomially bounded number of the following types of queries in an adaptive manner, and following method to win game:σ^* is a valid aggregate signature on

messages $M_1^*, ..., M_n^*$ with state information δ^* chosen by $\mathcal{A}_{II}$
If in any of the above two games, the success probability of any polynomial bounded opponent can be ignored, then the CLAS scheme will be unforgeable under adaptive selection message attack.

2 Security proof

2.1 Theorem 1.

In the random oracle model, if there is an opponent $\mathcal{A}_I$ of type I, after a maximum of q_k, within the time t, with the security parameter, the CLAS scheme is forged in the attack modeled in Game 1. Signature has advantages ε. Multiply part-private-key query, g_p multiply by public key query, q_{Hi} multiply by $H_i(i = 1, 2, 3)$ query, q_s multiply by Sign query, then CDH in G1 can be solved in a period problem. It's the basic theorem:

$$t + \mathcal{O}\left(q_{H_1} + q_{H_2} + q_{H_3} + q_K + q_P + q_S\right)\tau_{G_1} \tag{1}$$

Proof: H_1 queries: ℓ maintains a list H_1^{list} of tuples $(ID_j, \alpha_j, Q_j, c_j)$. This list is initially empty. Whenever receiving an H_1 query on ID_i, the same answer from the list H_1^{list}. H_2 queries: ℓ keeps a list H_2^{list} of tuples(δ_j, W_j, β_j). This list is initailly empty. Whenever A_1 issues a query $H_2(\delta_i)$, the same answer from the list H_2. H_3 queries: ℓ keeps a list H_3^{list} of tuples$(\delta_j, M_j, ID_j, P_j, S_j, \gamma_j)$. Whenever A_1 issues a query $(\delta_i||M_i||ID_i||P_i||R_i)$ to H_3, the same answer from the list H_3 will be given if the request has benn asked before.
Otherwise ℓ first makes an H_1 query on ID_i and finds the tuples and does as follows:

1. if c_i=0, abort.
2. Else if there's a tuple (ID_i, x_i, D_i, P_i) on K^{list}, set $D_i = \alpha_i P_T$ and return D_i as answer.
3. Otherwise, compute $D_i = \alpha_i P_T$, set $x_i = P_i = \perp$.

For Public-Key queries: $\mathcal{A}_I$ can choose a new public key for the user whose identity is ID_i.

1. If there's a tuple $(ID_i x_i D_i P_i)$ on $\mathbf{K}^{\text{list}}$ (in this case, the public key P_i of ID_i is $\perp$), choose $x_i' \in Z_q^*$, compute $P_i' = x_i' P$, return P_i' as answer and update $(ID_i x_i, D_i P_i)$ to (ID_i, x_i', D_i, P_i')
2. Otherwise, choose $x_i \in Z_q^*$, compute $P_i = x_i P$, return P_i as answer, set $D_i = \perp$ and add $(ID_i, x_i, D_i P_i)$ to $\mathbf{K}^{\text{list}}$

For Sign queries:

1. If $c_i = 0$, choose $r_i, \gamma_i \in Z_{q_i \text{ set}}^*$ $R_i = r_i P - \gamma_i^{-1} Q_i, \text{set } S_i = \gamma_i P_T$, add$(A_i M_i, ID_i P_i R_i, S_i, \gamma_i)$ to $\mathbf{H}_3^{\text{list}}$ (if there is a tuple $(A_i, M_i, I_i, P_i, R_i - S_i\gamma_i)$ on $\mathbf{H}_3^{\text{list}}$, then redo this step), compute $V_i = \beta_i P_i + r_i \gamma_i P_T$

$$\text{output } \sigma_i = (R_i V_i)$$

2. Else $c_i = 1$, randomly choose $R_i \in G_1$, set $V_i = \alpha_i P_T + \beta_i P_i + \gamma_i R_i$

$$\text{output } \sigma_i = (R_i V_i)$$

In addition, the forged aggregate signature must satisfy:

$$\left(e\left(V^*, P\right) = e\left(P_T, \sum_{i=1}^{n} Q_i^*\right) e\left(W^*, \sum_{i=1}^{n} P_i^*\right) \prod_{i=1}^{n} e\left(S_i^*, R_i^*\right)\right. \tag{2}$$

Because knowing $Q_1^* = \alpha_1^* bP, W^* = \beta^* P, S_1^* = \gamma_1^* P$; and for all i,$2 \le i \le n$, $Q_i^* = \alpha_i^* P, S_i^* = \gamma_i P$;. Hence $\mathcal{C}$ can compute:

$$e\left(V^*, P\right) = e\left(P_T, \sum_{i=1}^{n} Q_i^*\right) e\left(W^*, \sum_{i=1}^{n} P_i^*\right) \prod_{i=1}^{n} e\left(S_i^*, R_i^*\right) \tag{3}$$

$$= e\left(P_T, \sum_{i=1}^{n} \alpha_1^* bP\right) e\left(\beta^* P, \sum_{i=1}^{n} P_i^*\right) \prod_{i=1}^{n} e\left(\gamma_1^* P, R_i^*\right) \tag{4}$$

$$\Rightarrow abP = \alpha_1^{*-1}\left(V^* - \sum_{i=2}^{n}\left(\alpha_i^* P_T + \beta^* P_i^* + \gamma_i^* R_i^*\right) - \beta^* P_1^* - \gamma_1^* R_1^*\right) \tag{5}$$

Probability of success: $\varepsilon' = \Pr[E1 \wedge E2 \wedge E3] \ge (1-\delta)^{q_K}\varepsilon\delta(1-\delta)^{n-1} = \delta(1-\delta)^{(q_K+n-1)}\varepsilon$. When $\delta = \frac{1}{q_K+n}, \delta(1-\delta)^{(9x+n-1)}\varepsilon$ is maximized at $\frac{1}{q_K+n}\left(1-\frac{1}{q_K+n}\right)^{(q_K+n-1)}\varepsilon$ With sufficient large q_K, this probability turns to $\frac{1}{\frac{1}{q_K+n_1}}\varepsilon$. Hence, we have $\varepsilon' \ge \frac{1}{(q_x+n)e}\varepsilon$.

2.2 Theorem 2.

In the random oracle model, if there exists a type II adversary $\mathcal{A}_{II}$ who has an advantage ε in forging a signature of our CLAS scheme in an attack modeled by Game 2 running in time t for a security parameter ℓ and asking at most q_P times Public-Key queries, q_K times Secret-Key queries, H_i times H_i(i = 2,3) queries, q_S times Sign queries, then the CDH problem in G_1

$$t + \mathcal{O}\left(q_{H_2} + q_{H_3} + q_P + q_S\right)\tau_{G_1} \tag{6}$$

Proof: Most of them are the same as Theorem1.
For Sign queries:

1. If $c_i = 0$, choose $r_i \in Z_q^*$, set $R_i = r_i P - \left(\beta_i x_i/\gamma_i\right) bP, S_i = \gamma_i aP$, add $(\Delta_i, M_i, ID_i, P_i, R_i, S_i, \gamma_i)$ to $\mathbf{H}_3^{\text{list}}$ (if there is a tuple $(\Delta_i, M_i, I_i, P_i \ R_i S_i \gamma_i)$ exists on $\mathbf{H}_3^{\text{list}}$, then redo this step), compute $V_i = r_i P + \lambda H_1\left(ID_i\right)$, output $\sigma_i = (R_i, V_i)$

2. Else $c_i = 1$, generate $\sigma_i = (R_i, V_i)$ use the standard Sign algorithm, output $\sigma_i = (R_i, V_i)$

In addition, the forged aggregate signature must satisfy:

$$e\left(V^*, P\right) = e\left(P_T, \sum_{i=1}^{n} Q_i^*\right) e\left(W^*, \sum_{i=1}^{n} P_i^*\right) \prod_{i=1}^{n} e\left(S_i^*, R_i^*\right)$$

Because knowing $P_1^* = x_1^* bP, W^* = \beta^* aP, S_1^* = \gamma_1^* P$; for all $i, 2 \le i \le n, P_i^* = x_i^* P, S_i^* = \gamma_i^* P$;. Hence $\mathcal{C}$ can compute:

$$e\left(V^*, P\right) = e\left(P_T, \sum_{i=1}^{n} Q_i^*\right) e\left(W^*, \sum_{i=1}^{n} P_i^*\right) \prod_{i=1}^{n} e\left(S_i^*, R_i^*\right) \tag{7}$$

$$= e\left(x_T^* bP, \sum_{i=1}^{n} Q_i^*\right) e\left(\beta^* aP, \sum_{i=1}^{n} P_i^*\right) \prod_{i=1}^{n} e\left(\gamma_i^* P, R_i^*\right) \tag{8}$$

$$\Rightarrow abP = \left(x_1^* \beta^*\right)^{-1}\left(V^* - \sum_{i=2}^{n}\left(\lambda \mathrm{Q}_i^* + x_i^* W^* + \gamma_i^* R_i^*\right) - \lambda \mathrm{Q}_1^* - \gamma_1^* R_1^*\right) \tag{9}$$

Probability of success: $\varepsilon' = \Pr[E1 \wedge E2 \wedge E3] \geq \delta(1-\delta)^{(q_K+n-1)}\varepsilon$ When $\delta = \frac{1}{q_{K+n}}, \delta(1-\delta)^{(9x+n-1)}\varepsilon$ is maximized at $\frac{1}{q_K+n}\left(1-\frac{1}{q_K+n}\right)^{(q_K+n-1)}\varepsilon$ With sufficient large q_K, this probability turns to $\frac{1}{(q_K+n)e}\varepsilon$. Hence, we have $\varepsilon' \geq \frac{1}{(q_K+n)e}\varepsilon$

3 Conclusion

An efficient construction of a security model and certificateless aggregate signature scheme is proposed. Assuming that the CDH problem is difficult, the CLAS scheme is proved to be unforgeable under the adaptive selection message attack under the random oracle model. Because CL-PKC has some advantages over traditional PKC and ID-PKC, our CLAS solution may need to compress many different certificateless signatures into one signed application.