Improvement of an Identity-Based Aggregate Signature Protocol from Lattice

Songshou Dong[1,2,3], Yanqing Yao[1,2,3](✉), Yihua Zhou[4,5], and Yuguang Yang[4,5]

[1] State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China
yaoyq@buaa.edu.cn
[2] State Key Laboratory of Cryptology, Beijing 100878, China
[3] Key Laboratory of Aerospace Network Security, Ministry of Industry and Information Technology, School of Cyber Science and Technology, Beihang University, Beijing 100191, China
[4] Faculty of Information Technology, Beijing University of Technology, Bejing 100124, China
{zhouyh,yangyang7357}@bjut.edu.cn
[5] Beijing Key Laboratory of Trusted Computing, Beijing 100124, China

Abstract. In 2022, Li et al. [1] proposed a quantum secure and non-interactive identity-based aggregate signature protocol from lattices. In the end of their paper, they claimed that their scheme has key escrow problem. Based on this fact, we improve their scheme and propose a lattice-based certificateless aggregate signature protocol (L-CASP). Furthermore, our scheme has same signature size as Li et al. scheme and can avoid key escrow problem. Finally, we prove that our scheme is existentially unforgeable against adaptive chosen message attacks (EUF-CMA) under type I adversary and a type II adversary in the random oracle model (ROM).

Keywords: Lattice · certificateless · key escrow problem · signature · security

1 Introduction

Digital signatures can generally be divided into three categories: certificate-based, identity-based, certificateless.

For certificate-based signature schemes, public key infrastructure (PKI) is an important mechanism to maintain certificates and spread trust information among users in a hierarchical manner. As we all know, due to the generation and management of so many certificates are quite time-consuming, PKI is very expensive to deploy in practice and is very troublesome to use. Therefore, the introduction of the certification authority (CA) and PKI into the network coding system and the additional verification of each certificate will greatly reduce its performance.

For identity-based signature schemes, it can also be easily modified to be used in network coding to simplify the management of public keys. However, the inherent problem of any identity-based primitive lies in key escrow, which means Key-Generation-Center (KGC) entity knows the private key of any signer. Therefore, KGC can forge

S. Goel and P. R. Nunes de Souza (Eds.): ICDF2C 2023, LNICST 571, pp. 258–263, 2024.
https://doi.org/10.1007/978-3-031-56583-0_17

any signature of any signer as needed. Furthermore, the network coding system based on identity-based homomorphic signature also encounters the same problem and is only suitable for small private networks with low-security requirements.

For a certificateless signature scheme, it does not require key escrow, can avoid malicious KGC, and can better ensure the security requirements in network communications. In 2022, Li et al. [1] proposed a quantum secure and non-interactive identity-based aggregate signature protocol from lattices. In the end of their paper, they claimed that their scheme has key escrow problem. Based on this fact, we improve their scheme and propose a lattice-based certificateless aggregate signature protocol (L-CASP).

2 System Model, Algorithm Model and Security Model

Different from Li et al.'s scheme [1], our scheme is certificateless which avoids the key escrow problem. The following is the algorithm model and security model of our scheme.

2.1 Algorithm Model

1) Setup: Given a security parameter 1^n, KGC will generate system parameters *params* and secret value *msk*.
2) Generate public key and secret value: Given a signer's identity $ID \in \{0, 1\}^*$, outputs the public key $\boldsymbol{p_{ID}}$ and secret value $\boldsymbol{c_{ID}}$;
3) Generate a partial private key: Given a signer's identity $ID \in \{0, 1\}^*$ and a public key $\boldsymbol{p_{ID}}$, *ID* outputs the partial private key $\boldsymbol{s_{ID}}$.
4) Generate private key: Given the signer's identity $ID \in \{0, 1\}^*$ and the partial private key $\boldsymbol{c_{ID}}$, the user outputs the private key $\boldsymbol{sk_{ID}}$ and key $\boldsymbol{pk_{ID}}$.
5) Signature: Given a triple $(\boldsymbol{m_{ID}}, ID, \boldsymbol{sk_{ID}})$, the user generates a signature $\boldsymbol{Sig_{ID}} = \boldsymbol{z_{ID}}$.
6) Aggregation: Given a triple $(m_i, \boldsymbol{PK} = (\boldsymbol{pk_{ID_i}}), \boldsymbol{Sig_i}(i \in [t]))$, Aggregator outputs the aggregated signature $\boldsymbol{Zag}$;
7) Aggregate verification: input triplet $(\boldsymbol{m_i}, \boldsymbol{PK} = (\boldsymbol{pk_{ID_i}}), (i \in [t]), \boldsymbol{Zag})$, if $\boldsymbol{Zag}$ is a valid aggregate signature, verifier output 1, otherwise output 0.

2.2 Security Model

There are two types of adversaries in certificateless cryptosystems: type I and type II (A_1 and A_2). For signer *ID*, A_1: If you know the secret value $\boldsymbol{c_{ID}}$, but you don't know the master key $\boldsymbol{B}$, you can replace the user public key $\boldsymbol{pk_{ID}}$. A_2: Know the master key $\boldsymbol{B}$, but cannot replace the user public key $\boldsymbol{pk_{ID}}$. According to different classifications of adversaries, two types of games are defined as Game I and Game II.

The security requirements of the lattice-based certificateless aggregate signature protocol (L-CASP) for cloud healthcare are as follows:

Definition 1. The L-CASP scheme satisfies unforgeability if the adversary wins in probabilistic polynomial time with a non-negligible advantage in the next two games.

Game I. Challenger *C* plays an interactive game with type I adversary A_1.

1) *Initialization* : Challenger C runs the setting algorithm to generate secret value *msk* and system parameter *params*, keeps *msk* secret, and sends system parameter params to adversary A_1.
2) *Query* : Adversary A_1 can perform polynomial number of queries. For the signer *ID*, the adversary A_1 performs the following query:
   (1) $PK-Query$: Adversary A_1 queries the public key of *ID*, and challenger C outputs the corresponding public key $\boldsymbol{pk_{ID}}$.
   (2) $Hash-Query$: Adversary A_1 can query any hash function value.
   (3) $PK-Replace$: Adversary A_1 sends $\boldsymbol{pk_{ID}^{'}}$ to challenger C, and challenger C replaces $\boldsymbol{pk_{ID}}$ with $\boldsymbol{pk_{ID}^{'}}$.
   (4) $PPK-Query$: Adversary A_1 queries the partial private key of *ID*, and the challenger C outputs the partial private key $\boldsymbol{s_{ID}}$. If the private key $\boldsymbol{sk_{ID}}$ of *ID* is replaced, A_1 will not be able to execute $PPK-Query$.
   (5) $SV-Query$: Adversary A_1 queries the secret value of *ID*, challenger C returns $\boldsymbol{c_{ID}}$ to adversary A_1. If $\boldsymbol{c_{ID}}$ is replaced, adversary A_1 will not be able to perform secret value lookup.
   (6) $Sign-Query$: Adversary A_1 inputs the tuple $(\boldsymbol{m_{ID}}, ID, \boldsymbol{sk_{ID}})$, and the challenger C outputs the signature $\boldsymbol{Sig_{ID}} = (z_{\boldsymbol{ID}})$.
3) *Forge* : adversary A_1 output tuple $(\boldsymbol{M} = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots \boldsymbol{m_t}\}, \boldsymbol{P} = \{(ID_1, \boldsymbol{pk}_1), (ID_2, \boldsymbol{pk}_2), \cdots (ID_t, \boldsymbol{pk_t}), \}, \boldsymbol{Zag})$. The opponent wins if the following requirements are met:
   (1) $\boldsymbol{Sig}$ is not obtained through $Sign-Query$;
   (2) Aggregate verification $(\boldsymbol{M}, \boldsymbol{P}, \boldsymbol{Zag}) = 1$;
   (3) There is at least one identity *ID* for which the adversary A_1 did not perform $PPK-Query$ or replace the main private key $\boldsymbol{B}$.

The advantage of adversary A_1 winning is defined as: $Adv_{A_1}^{\mathrm{L-CASP}} = \Pr[A_1 wins]$.

Game II. Challenger C and the second type of opponent A_2 start an interactive game.

1) *Initialization* : Challenger C runs the setup algorithm to generate the secret value msk and system parameters params, and then sends them to the adversary A_2.
2) *Query* : Adversary A_2 performs the same polynomial query as in game I.
3) *Forge* : the adversary A_2 output tuple $(\boldsymbol{M} = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots \boldsymbol{m_t}\}, \boldsymbol{P} = \{(ID_1, \boldsymbol{pk}_1), (ID_2, \boldsymbol{pk}_2), \cdots (ID_t, \boldsymbol{pk_t}), \}, \boldsymbol{Zag})$. The opponent wins if the following requirements are met:
   (1) $\boldsymbol{Sig}$ is not obtained through $Sign-Query$;
   (2) Aggregate verification $(\boldsymbol{M}, \boldsymbol{P}, \boldsymbol{Zag}) = 1$;
   (3) There is at least one identity *ID* for which the adversary A_2 did not perform $SV-Query$ or replace the secret value $\boldsymbol{c}_{ID}$.

The advantage of adversary A_2 winning is defined as: $Adv_{A_2}^{\mathrm{L-CASP}} = \Pr[A_2 wins]$.

3 Our Protocol

In this section, we will introduce our L-CASP in detail. In our scheme, we use the *TrapGen* algorithm [2] and the *SamplePre* algorithm [3]. Additionally, all polynomial computations of our protocol are defined on $R_q = Z_q[X]/(X^n + 1)$. The following are

details. Figure 1 describes the process of key generation (This is the core step of our improvement, as you can see, the user's private key is generated in two parts).

Fig. 1. The process of key extraction

(1) Setup Phase

The Setup Phase is basically the same as the reference [1], except that we add a hash function $H_3 : \{0, 1\}^* \rightarrow (B_\beta)^*$, $B_\beta = \{\boldsymbol{r} \in \mathrm{R} : \|\boldsymbol{r}\| \leq \beta_3\}$.

(2) Key Extraction Phase

1) The user *ID* selects polynomial $\boldsymbol{c_{ID}} R_q$, calculates $\boldsymbol{p_{ID}} = \boldsymbol{h} * \boldsymbol{c_{ID}}$ and send $\boldsymbol{p_{ID}}$ to KGC;
2) KGC computes $\boldsymbol{k_{ID}} = H_3(ID, \boldsymbol{p_{ID}})$, gets $\boldsymbol{s_{ID}} = (\boldsymbol{s_{ID1}}, \boldsymbol{s_{ID2}})^T$ (which meets $H_3(ID, \boldsymbol{p_{ID}}) = \boldsymbol{s_{ID1}} + \boldsymbol{h}\boldsymbol{s_{ID2}}$, $\|(\boldsymbol{s_{ID1}}, \boldsymbol{s_{ID2}})\| \leq \sigma\sqrt{2n}$) through $SamplePre(\boldsymbol{h}, \boldsymbol{B}, \sigma, (\boldsymbol{k_{ID}}, 0))$ and sends $\boldsymbol{s_{ID}}$ to the user *ID*;
3) User *ID* computes the private key $\boldsymbol{sk_{ID}} = \boldsymbol{c_{ID}} + \boldsymbol{s_{ID1}} + \boldsymbol{s_{ID2}}$, lets public key $\boldsymbol{pk_{ID}} = (\boldsymbol{h}, \boldsymbol{p_{ID}})$.

(3) Sign Phase

The user *ID* can generate a valid signature on any message $\boldsymbol{m_{ID}} \in \{0, 1\}^*$ as follows. *ID* gets the random polynomial $\boldsymbol{w_{ID}}$ by the seed and the algorithm *GaussFunction* [1], selects two polynomial $\boldsymbol{y_{ID,2}} \leftarrow D_{R,s}$, $\boldsymbol{y_{ID,3}} \leftarrow D_{R,s}$, computes $\boldsymbol{y_{ID,1}} = \boldsymbol{w_i} - \boldsymbol{h}\boldsymbol{y_{ID,2}} - \boldsymbol{h}\boldsymbol{y_{ID,3}}$, $\boldsymbol{e_{ID}} = H_1(\boldsymbol{PK}, \boldsymbol{w_{ID}}, \boldsymbol{m_{ID}})$ ($\boldsymbol{PK} = (\boldsymbol{pk_{ID_i}})$, $i \in [N]$), $\boldsymbol{z_{ID,1}} = \boldsymbol{y_{ID,1}} + \boldsymbol{s_{ID,1}}\boldsymbol{e_{ID}}$, $\boldsymbol{z_{ID,2}} = \boldsymbol{y_{ID,2}} + \boldsymbol{s_{ID,2}}\boldsymbol{e_{ID}}$, $\boldsymbol{z_{ID,3}} = \boldsymbol{y_{ID,3}} + \boldsymbol{c_{ID}}\boldsymbol{e_{ID}}$ and outputs $\boldsymbol{z_{ID,1}}$ with the probability $\min(\frac{D_{R,\sigma}(\boldsymbol{z_{ID,1}})}{M_1 D_{R,\sigma,\boldsymbol{s_{ID,1}}\boldsymbol{e_{ID}}}(\boldsymbol{z_{ID,1}})}, 1)$ ($M_1 \approx e$), $\boldsymbol{z_{ID,2}}$ with the probability $\min(\frac{D_{R,\sigma}(\boldsymbol{z_{ID,2}})}{M_2 D_{R,\sigma,\boldsymbol{c_{ID}}\boldsymbol{e_{ID}}}(\boldsymbol{z_{ID,2}})}, 1)$($M_2 \approx e$), $\boldsymbol{z_{ID,3}}$ with the probability $\min(\frac{D_{R,\sigma}(\boldsymbol{z_{ID,3}})}{M_3 D_{R,\sigma,\boldsymbol{c_{ID}}\boldsymbol{e_{ID}}}(\boldsymbol{z_{ID,3}})}, 1)$($M_3 \approx e$). Next *ID* could generate a signature $\boldsymbol{z_{ID}} = \boldsymbol{h}^{-1}\boldsymbol{z_{ID,1}} + \boldsymbol{z_{ID,2}} + \boldsymbol{z_{ID,3}}$ and sends it to the aggregator.

(4) Sign-Agg Phase

The aggregator get all $\boldsymbol{w_{ID_i}}$, $i \in [N]$ by the seed and the algorithm *GaussFunction* [1], the set of public keys $\boldsymbol{PK} = (\boldsymbol{pk_{ID_i}})$, $i \in [N]$, all messages $\boldsymbol{m_{ID_i}}$ and $\boldsymbol{w_{ID_i}}$, $i \in [N]$, calculates the hash value $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x_N}) \leftarrow H_2(\boldsymbol{w_{ID_1}}, \boldsymbol{pk_{ID_1}}, \boldsymbol{m_{ID_1}}, \cdots, \boldsymbol{w_{ID_N}}, \boldsymbol{pk_{ID_N}}, \boldsymbol{m_{ID_N}})$, and an aggregate signature $\boldsymbol{Zag} = \sum_{i=1}^{N} \boldsymbol{x_i}\boldsymbol{z_{ID_i}}$.

(5) Verify Phase

Given public keys $\boldsymbol{PK} = (\boldsymbol{pk_{ID_i}})$, $i \in [N]$, the message set $\boldsymbol{m_{ID_i}}$, $i \in [N]$, and the aggregate signature $\boldsymbol{Zag}$, the verifier gets all $\boldsymbol{w_{ID_i}}$, $i \in [N]$ by algorithm *GaussFunction* with seed, computes $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x_N}) \leftarrow H_2(\boldsymbol{w_{ID_1}}, \boldsymbol{pk_{ID_1}}, \boldsymbol{m_{ID_1}}, \cdots,$

$\boldsymbol{w_{ID_N}}$, $\boldsymbol{pk_{ID_N}}$, $\boldsymbol{m_{ID_N}}$) and $\boldsymbol{e_{ID_i}}$=$H_1(\boldsymbol{PK}, \boldsymbol{w_{ID_i}}, \boldsymbol{m_{ID_i}})$, $i \in [N]$, and verifies the correctness of the following conditions:

$$\boldsymbol{h}*\boldsymbol{Zag} = \sum_{i=1}^{N} \boldsymbol{x_i}\left(\boldsymbol{w_{ID_i}} + \boldsymbol{e_{ID_i}}\left(\boldsymbol{H}_3\left(\boldsymbol{ID_i}, \boldsymbol{p_{ID_i}}\right) + \boldsymbol{p_{ID_i}}\right)\right)$$

$$\boldsymbol{Zag} \leq \beta_{ver}\left(\beta_{ver} \geq N\beta_2\left(\sigma\sqrt{n}+2\right)\left(3\sigma\sqrt{n}+nq\beta_1\right)\right)$$

4 Correctness Analysis

In this section, we analyze the correctness of our scheme. The details are as follows.

$$\begin{aligned}
&\boldsymbol{h} * \boldsymbol{Zag} = \boldsymbol{h}\sum_{i=1}^{N} \boldsymbol{x_i}\boldsymbol{z_{ID_i}} \\
&= \boldsymbol{h}\sum_{i=1}^{N} \boldsymbol{x_i}\left(\boldsymbol{h}^{-1}\boldsymbol{z_{ID_i,1}} + \boldsymbol{z_{ID_i,2}} + \boldsymbol{z_{ID_i,3}}\right) \\
&= \sum_{i=1}^{N} \boldsymbol{x_i}\boldsymbol{h}\left(\boldsymbol{h}^{-1}\boldsymbol{z_{ID_i,1}} + \boldsymbol{z_{ID_i,2}} + \boldsymbol{z_{ID_i,3}}\right) \\
&= \sum_{i=1}^{N} \boldsymbol{x_i}\left(\boldsymbol{z_{ID_i,1}} + \boldsymbol{h}\boldsymbol{z_{ID_i,2}} + \boldsymbol{h}\boldsymbol{z_{ID_i,3}}\right) \\
&= \sum_{i=1}^{N} \boldsymbol{x_i}\left(\boldsymbol{y_{ID_i,1}} + \boldsymbol{s_{ID_i,1}}\boldsymbol{e_{ID_i}} + \boldsymbol{h}\left(\boldsymbol{y_{ID_i,2}} + \boldsymbol{s_{ID_i,2}}\boldsymbol{e_{ID_i}}\right) + \boldsymbol{h}\left(\boldsymbol{y_{ID_i,3}} + \boldsymbol{c_{ID_i}}\boldsymbol{e_{ID_i}}\right)\right) \\
&= \sum_{i=1}^{N} \boldsymbol{x_i}\left(\boldsymbol{y_{ID_i,1}} + \boldsymbol{h}\boldsymbol{y_{ID_i,2}} + \boldsymbol{h}\boldsymbol{y_{ID_i,3}} + \boldsymbol{e_{ID_i}}\left(\boldsymbol{s_{ID_i,1}} + \boldsymbol{h}\boldsymbol{s_{ID_i,2}} + \boldsymbol{h}\boldsymbol{c_{ID_i}}\right)\right) \\
&= \sum_{i=1}^{N} \boldsymbol{x_i}\left(\boldsymbol{w_{ID_i}} + \boldsymbol{e_{ID_i}}\left(\boldsymbol{H}\left(\boldsymbol{ID}, \boldsymbol{p_{ID_i}}\right) + \boldsymbol{p_{ID_i}}\right)\right)
\end{aligned}$$

Firstly, according to the Gaussian sampling [4], $\|\boldsymbol{y_{ID_i,1}}\|, \|\boldsymbol{y_{ID_i,2}}\|, \|\boldsymbol{y_{ID_i,3}}\| \leq 2\sigma\sqrt{n}$ with overwhelming probability.

Because $\|\boldsymbol{s_{ID_i,1}}\|, \|\boldsymbol{s_{ID_i,2}}\|, \|\boldsymbol{c_{ID_i}}\| \leq nq$, and $\|\boldsymbol{e_{ID_i}}\| \leq \beta_1$, $\|\boldsymbol{z_{ID_i,1}}\|, \|\boldsymbol{z_{ID_i,2}}\|, \|\boldsymbol{z_{ID_i,3}}\| \leq 3\sigma\sqrt{n} + nq\beta_1$.

So, the signature $\boldsymbol{z_{ID_i}}, i \in [N]$ satisfies the inequality $\|\boldsymbol{z_{ID_i}}\| \leq (\sigma\sqrt{n}+2)(3\sigma\sqrt{n} + nq\beta 1)$.

Due to $\boldsymbol{Zag} = \sum_{i=1}^{N} \boldsymbol{x_i}\boldsymbol{z_{ID_i}}$,

$\|\boldsymbol{Zag}\| \leq N\beta_2(\sigma\sqrt{n}+2)(3\sigma\sqrt{n} + nq\beta 1)$.

5 Existentially Unforgeable Against Adaptive Chosen Message Attacks (EUF-CMA)

Normally, the security of a L-CASP usually needs to be proven from two aspects.

1) Type I attack: Adversary can only replace the signer's public key, but the partial private key of any signer cannot be got;
2) Type II attack: Adversary can only obtain the private key of KGC, but cannot replace the public key of any signer.

The proof method is the same as the proof method in Ref. [5], we will not elaborate here.

6 Conclusion

Through comparison, our scheme has same signature size as Li et al.'s scheme and can avoid key escrow problem. And we prove that our scheme is EUF-CMA under type I adversary and a type II adversary in the random oracle model (ROM).

Acknowledgement. This work is supported by the National Key Research and Development Program of China (No. 2021YFB3100400),the National Natural Science Foundation of China (grant no. 62072023), the Open Project Fund of the State Key Laboratory of Cryptology (grant no. MMKFKT202120), Beijing Municipal Natural Science Foundation, the Exploratory Optional Project Fund of the State Key Laboratory of Software Development Environment, and the Fundamental Research Funds of Beihang University (grant nos. YWF-20-BJ-J-1040, YWF-21-BJ-J-1041, etc.).

References

1. Li, Q., Luo, M., Hsu, C., et al.: A quantum secure and noninteractive identity-based aggregate signature protocol from lattices. IEEE Syst. J. **16**(3), 4816–4826 (2021)
2. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_4
3. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, Victoria, pp. 197–206. ACM (2008)
4. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 40–56. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_3
5. Dong, S.S., Zhou, Y.H., Yang, Y.G., et al.: A certificateless ring signature scheme based on lattice. Concurr. Comput. Pract. Exp. **34**(28), e7385 (2022)