SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2026, Vol. 69, Iss. 9, 192105:1–192105:16
https://doi.org/10.1007/s11432-025-4692-7

A non-interactive identity-based accountable-subgroup multi-signature scheme from lattices

Xinjian CHEN[1], Pingbin LUO[1], Hongbo LI[1], Willy SUSILO[2] & Qiong HUANG[3,1*]

[1]*College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China*
[2]*School of Computing and Information Technology, University of Wollongong, Wollongong NSW 2522, Australia*
[3]*School of Computer Science, Guangdong University of Finance, Guangzhou 510521, China*

Received 21 June 2025/Revised 24 September 2025/Accepted 20 November 2025/Published online 12 August 2026

Abstract As blockchain technology advances rapidly, Bitcoin transactions now support more complex t-out-of-N multi-signature transactions. Accountable subgroup multi-signatures are well-suited for this requirement, allowing any subgroup of possible signers to collaboratively sign a message. However, traditional multi-signature schemes face potential risks from quantum computing attacks due to their compromised security assumptions. Alternatively, lattice-based cryptography offers a reliable countermeasure. In this paper, we introduce a non-interactive identity-based accountable-subgroup multi-signature scheme (IBASM) from lattices, which supports flexible accountable-subgroup multi-signatures, aligning better with the t-out-of-N multi-signature transactions on the Bitcoin blockchain. Moreover, our non-interactive signing algorithm eliminates the need for complex broadcast protocols, reducing communication overhead in the Bitcoin blockchain. Additionally, if any participant needs to restart the signing algorithm, it will not influence other participants, improving multi-signature generation efficiency. Furthermore, the public keys are generated based on users' identity information, making the scheme practical for real-world applications. Finally, experimental results demonstrate that our approach offers more efficient multi-signature generation and comparable verification performance compared to existing schemes. Specifically, in our scheme, the multi-signature generation efficiency improves by around 50% and 66.7%, respectively, in two sets of parameters.

Keywords Bitcoin blockchain, accountable-subgroup multi-signature, lattice-based cryptography, non-interactive signing algorithm, t-out-of-N multi-signature transactions

Citation Chen X J, Luo P B, Li H B, et al. A non-interactive identity-based accountable-subgroup multi-signature scheme from lattices. Sci China Inf Sci, 2026, 69(9): 192105, https://doi.org/10.1007/s11432-025-4692-7

1 Introduction

The Bitcoin blockchain [1] serves as a decentralized digital ledger that records all Bitcoin transactions. Its fundamental characteristic is the series of interconnected blocks to guarantee data integrity and transparency. Each block holds transaction information along with the hash of the preceding block, forming a chain structure. Bitcoin utilizes digital signature technology [2] to authenticate the legitimacy of transactions. As Bitcoin blockchain technology continues to evolve, some transactions now require multiple signatures for authentication [3], with different nodes collaborating to confirm the validity of the transactions. This approach helps prevent single points of failure and enhances security. However, a potential drawback arises: each transaction involves a bunch of individual signatures, all of which must be recorded on the blockchain, leading to substantial storage consumption and placing a heavy burden on the system.

A similar situation can be found in blockchain-based voting systems [4]. In such a system, suppose a proposal is up for voting, and the system typically requires a majority of votes in favor for the proposal to pass. During the voting process, users cast their votes in favor and sign them. A vote is only considered valid if its signature is verified. Ultimately, the supporting votes for the proposal come from different nodes on the blockchain. Through verification and calculation, the total number of supporting votes is determined. When the number reaches or surpasses a certain threshold, the proposal gets approved. Finally, all the data, including the signatures of the supporting votes, is stored on the blockchain, which consumes substantial storage space and places a considerable burden on the e-voting system.

To address the aforementioned issue, an effective solution is to use multi-signature schemes to reduce the storage requirements of the blockchain. Multi-signature [5] enables multiple signers, each with their own public and private

* Corresponding author (email: qhuang@scau.edu.cn)

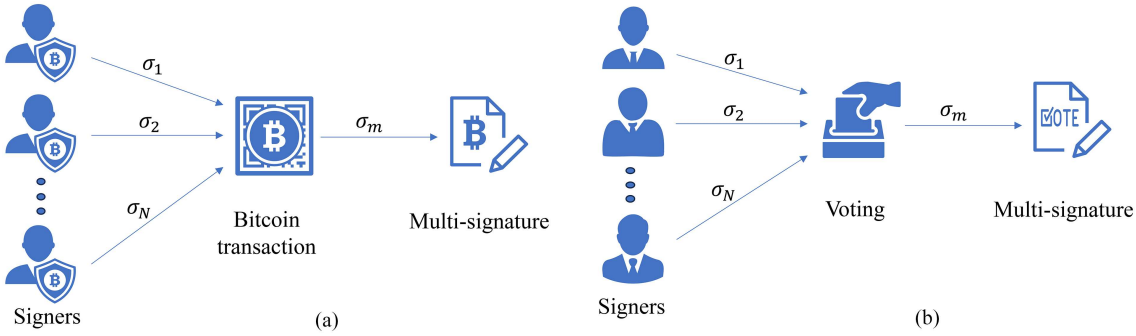

Figure 1 (Color online) Multi-signature applications. (a) Bitcoin multi-signature; (b) voting multi-signature.

key pairs, to jointly create a single, compressed signature for the same message, which can be applied to Bitcoin transactions and e-voting systems, as shown in Figure 1. With this approach, only the generated multi-signature is stored on the blockchain, rather than a bunch of individual signatures, thereby alleviating the storage burden on the blockchain. Furthermore, during the verification process, if the multi-signature is successfully verified, it indicates that all participants have signed the same message.

Moreover, in practical blockchain transaction scenarios, Bitcoin transactions typically support more complex multi-signature transactions, specifically t-out-of-N multi-signature transactions [3]. In this setup, among N possible validating nodes, if at least t nodes authenticate the transaction and provide valid signatures, the transaction is confirmed and entered into the blockchain. A similar principle applies in voting systems. In most voting contexts, a proposal is approved as long as the majority votes in favor, rather than requiring unanimous agreement. This technique, which incorporates a threshold policy into multi-signatures, is known as t-out-of-N multi-signatures [6] or accountable-subgroup multi-signatures (ASM) [7]. An ASM [7] enables any subgroup of the N parties (each of which has its own public and private keys) to jointly sign a message. Threshold signature [8] is a closely related concept, where a signing key is shared across the N parties, allowing any threshold number of those parties to jointly issue a signature. The key difference between ASMs and threshold signatures is accountability. With ASMs, it is possible to trace which subgroup of signers provided the signature, as all participants in the signing process are recorded. In contrast, threshold signatures do not provide such accountability, as they prioritize anonymity and do not reveal which specific members participated in the signing. To preserve this accountability and avoid the complex distributed key generation process associated with threshold signatures, we primarily focus on accountable-subgroup multi-signatures in this paper, where the threshold t denotes the minimum required number of subgroup signers. This means that a valid multi-signature can only be created when at least t signers from the subgroup have signed.

Initially, the public key infrastructure (PKI) cryptosystem was employed to create multi-signature schemes. In this system, each user's public key is generated randomly and does not have any specific meaning related to the user. Therefore, to establish the correspondence between each user and their respective public key, a trusted entity, known as the certificate authority (CA), plays a key role by issuing a certificate that confirms the one-to-one correspondence between the two. However, as the user count continues to grow, managing certificates becomes increasingly cumbersome, which can lead to complex certificate management problems. To address this issue, Shamir [9] introduced a groundbreaking concept in 1984, known as identity-based cryptography (IBC). In this new cryptosystem, a user's identity information (e.g., ID card number or email address) is used to create the public key, while the key generation center (KGC) is in charge of producing the associated secret key. Since the public keys are directly linked to user's identity information in the system, which naturally establishes a clear correspondence, there is no need for certificates to be issued by the CA, thus effectively getting rid of the complex certificate management problem. Consequently, this paper focuses primarily on identity-based accountable-subgroup multi-signatures.

On the other hand, it should be noted that traditional multi-signature schemes primarily rely on classical cryptographic problems, including the integer factoring and discrete logarithm problems. As quantum computing technology advances rapidly, these schemes are becoming insecure due to the weakening of their underlying security assumptions. With the aid of quantum computers, it is possible to efficiently solve certain integer factoring and discrete logarithm problems within polynomial time by using Shor's algorithms [10]. Additionally, Grover's quantum search algorithm [11] can significantly enhance search efficiency, posing a serious security threat to traditional multi-signature schemes. In contrast, lattice-based cryptographic schemes are widely recognized for their quantum resistance, as lattice problems are extremely difficult to solve in polynomial time, even with the power of quantum computers [12–18]. Therefore, this paper mainly focuses on constructing a lattice-based identity-based accountable-subgroup multi-signature protocol.

Our contributions. In this paper, we make the following contributions.

• We propose a non-interactive identity-based accountable-subgroup multi-signature scheme (IBASM) from lattices. Unlike previous lattice-based multi-signature protocols, our signing algorithm is non-interactive, allowing each signer to independently generate their signature without requiring the complex multi-round interactions typical of interactive schemes. This significantly reduces the communication overhead of our scheme. Furthermore, if any signer needs to restart the signing algorithm, it will not affect the other signers. In contrast, in interactive schemes, all signers must restart if any signer restarts, leading to an exponential increase in the runtime of the signing algorithm and severely hampering signing efficiency. Therefore, our IBASM scheme offers superior signing efficiency compared to related schemes.

• The multi-signature in our IBASM scheme is generated when the number of subgroup signers meets the required threshold. This multi-signature can identify which subgroup of signers generated it, ensuring accountability. Unlike current lattice-based multi-signature schemes, which require all participants to be predetermined, our scheme is more flexible and better suited for t-out-of-N multi-signature transactions on the Bitcoin blockchain.

• By leveraging the ring version of the small integer solution (Ring-SIS) problem, a key lattice-based cryptographic security assumption, we demonstrate the existential unforgeability of our approach within the random oracle model. As a result, the IBASM scheme is post-quantum secure, since lattice-based cryptographic systems are widely regarded as resistant to quantum attacks [12, 13].

• Our IBASM scheme allows each user to designate their own system identity (e.g., ID card number or email address) as their public key compared to PKI-based schemes, making it more compatible with practical applications.

• Finally, we conduct experiments to compare our scheme with other related approaches. The results indicate that our scheme demonstrates superior performance compared to other relevant schemes. Specifically, our scheme achieves faster multi-signature generation, with improvements of approximately 50% and 66.7% in two parameter combinations, while the multi-signature verification efficiency is comparable to other related solutions.

Technical overview. Our IBASM scheme, like other practical lattice-based schemes, works over the polynomial ring $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n+1)$, where n is a power of 2 and q is a prime congruent to 1 modulo $2n$. In an identity-based signature scheme, the public key of a user is simply his identity (e.g., his email or IP address), and a trusted KGC assigns each user a corresponding secret signing key. We now give a brief description of our IBASM scheme, which is built on NTRU lattices.

The master public key is a polynomial h, and the secret key consists of a "nice basis" for the $2n$-dimensional lattice (see Definition 2). A user with identity ID generates a public key $t = H_1(ID)$, where H_1 is a publicly-known cryptographic hash function mapping into $\mathbb{Z}_q[x]/(x^n+1)$. The user's secret key consists of a small polynomial s_2 such that $s_1 + s_2 \cdot h = t$, where s_1 is another small polynomial.

The basic signature scheme follows the Fiat-Shamir paradigm for lattice-based signatures. Specifically, a signature is denoted as $\sigma = (\mathbf{z}, r) \in R_q^2 \times R_q$, where r is a commitment that, via a hash function H_2, defines a challenge c, and $\mathbf{z}$ is the response to this challenge. The challenge c is a polynomial with coefficients in $\{-1, 0, 1\}$. For verification, one checks that $\mathbf{z}$ is short and that the equation $[1|h] \cdot \mathbf{z} = t \cdot c + r$ holds, where $c = H_2(ID, r, \mathbf{ID}, m)$ for the identity ID, commitment r, identity list of signers $\mathbf{ID}$, and message m.

When the number of validated individual signatures reaches or exceeds the threshold T, we generate a multi-signature. For ease of description, we define L_m as the set of indices corresponding to signers with valid signatures. An intuitive approach to combine individual signatures $(\sigma_j = (\mathbf{z}_j, r_j))_{j \in L_m}$ is to compute the sum of all components $(\sum_j \mathbf{z}_j, \sum_j r_j)$. However, this multi-signature cannot be verified as we cannot recompute the different challenges c_j for the commitments r_j, originally used by the signing parties. As a result, we can only sum the $\mathbf{z}_j$ components and transmit all r_j, which produces a multi-signature of the form $\sigma_m = (\sum_j \mathbf{z}_j, (r_j)_j)$. To prevent rogue attacks, we use a (randomized) linear combination of individual signatures $\mathbf{z} = \sum_j e_j \mathbf{z}_j$ (instead of the simple sum), where the random coefficients e_j are derived from a hash oracle H_3, which is queried on all challenges c_j for each $j \in L_m$.

To verify the multi-signature σ_m, the verifier first reconstructs the challenges c_j for $j \in L_m$ using the identity ID_j, commitment r_j, identity list $\mathbf{ID}$ and message m. It then computes $e_j = H_3((c_i)_{i \in L_m}, j)$ for each $j \in L_m$. It checks that the norm of $\mathbf{z}$ is within the correct bound and that the equation $[1|h] \cdot \mathbf{z} = \sum_j e_j (H_1(ID_j) \cdot c_j + r_j)$ holds. Finally, by applying (twice) the general forking lemma [6], we show that four distinct responses from a successful adversary can be used to solve an instance of Ring-SIS.

2 Related work

It was Itakura and Nakamura [5] who first put forward the multi-signature concept, which allows multiple signers to jointly generate a single, compact multi-signature for a message, thereby effectively reducing the required storage

space compared to storing a set of individual signatures. Since then, numerous multi-signature protocols [6, 19–22], based on the framework of Schnorr signature [23], have been introduced. However, these schemes have been found to be susceptible to concurrent attacks [24, 25], prompting the development of more secure alternatives [24, 26–29]. Regarding identity-based multi-signatures, several related schemes [30–35] have been developed to address certificate management challenges in PKI settings.

ASM was initially introduced by Micali et al. [36] in 2001, and is sometimes referred to as a t-out-of-N multi-signature, where t is the threshold number of signatures and N denotes the total potential participants involved. However, the key generation protocol in this scheme is highly complex and cumbersome, leading to significant overhead. In 2018, Boneh et al. [7] constructed an ASM protocol aimed at shrinking the size of the Bitcoin blockchain, which features a non-interactive signing process. Recently, Kara et al. [37] designed a novel ASM construction based on the RSA assumption, which enables a valid multi-signature to disclose which subset produced the signature. Their approach employs a two-round interactive protocol for signing.

However, the aforementioned schemes are built upon traditional security assumptions that may become vulnerable in the face of rapid advancements in quantum computing. Consequently, a series of lattice-based multi-signature solutions have emerged. The first such scheme, introduced by Bansarkhani and Sturm [38] in 2016, offers a multi-signature length comparable to a single signature. However, its performance in signing and verification scales linearly with the number of participants, limiting the scalability. In 2019, Ma and Jiang [39] presented a lattice-based multi-signature protocol with public key aggregation. Compared with [38], this new scheme highlights a reduction in restart times during the signature algorithm, and offers shorter lengths of both public keys and signatures. However, it suffers from substantial communication overhead and imposes restrictions on the number of participants in the signing process. Fukumitsu and Hasegawa [40] introduced a multi-signature scheme which utilizes lattices and the ring version of the learning with error problem to achieve enhanced security. Subsequently, Fukumitsu and Hasegawa [41] introduced an improved scheme based on [40], making it more secure under the quantum random oracle model (QROM). In 2020, Peng et al. [42] introduced a simultaneous multi-signature approach that combines the signature permutation of signers with the NIWI proof [43] as a multi-signature. As a result, the scheme's scalability is restricted due to the linear growth of multi-signature size with the number of participants. A single-round multi-signature scheme from lattices was proposed by Kansal and Dutta [44], supporting public key aggregation and simplifying the verification process to solely examine the correlation between the multi-signature and the aggregated public key. Nevertheless, this scheme suffers from significant storage space consumption. Based on [44], Kansal et al. [45] presented a two-round lattice-based solution with public key aggregation, which reduces both storage space and communication bandwidth requirements.

In 2021, Damgård et al. [46] introduced a two-round multi-signature scheme, named DOTT, which is built upon lattice structures and the trapdoor commitment technique [47]. In 2022, Boschini et al. [48] introduced MuSig-L, another lattice-based multi-signature protocol. Unlike DOTT [46], MuSig-L does not rely on the additional cryptographic primitive of lattice-based trapdoor commitments for its security proof, and it also features a smaller signature size. However, the scheme suffers from high communication complexity. Fleischhacker et al. [49] designed a synchronized multi-signature system that generates a signature by augmenting the current timestamp as input and compresses signatures on identical messages and timestamps into a multi-signature. Subsequently, Chen et al. presented two lattice-based multi-signature protocols: identity-based multi-signature (IBMS) scheme [50] and certificateless multi-signature (CLMS) scheme [51]. Both schemes effectively address the certificate management issue often associated with PKI-based schemes. In 2023, Liang et al. [52] designed a novel multi-signature system by leveraging the BLISS scheme [53], which demonstrates relatively stable multi-signature sizes under different security levels. Recently, Chen [54] proposed DualMS, a trapdoor-free lattice-based multi-signature protocol that offers smaller signature sizes and reduced communication costs. However, it is unable to reduce interaction times through offline preprocessing.

Existing lattice-based multi-signature schemes, to the best of our knowledge, typically fix the entire signer group involved in the multi-signature process in advance, and there is currently no lattice-based ASM scheme. Meanwhile, most current lattice-based multi-signature approaches rely on broadcasting signature protocols, where different signers must interact with each other to collaboratively generate a multi-signature, which obviously increases communication overhead. To address this issue, Phalakarn et al. [55], Li et al. [56] and Chen et al. [57] introduced non-broadcasting approaches. However, their schemes require a reliable third-party server with a trapdoor, and each signer must interact with the server to generate their individual signatures. Therefore, to further improve efficiency and flexibility, we aim to eliminate the need for the third-party server in their signing algorithms and design a non-interactive identity-based ASM scheme on lattices in this paper.

3 Preliminaries

3.1 Notations

We define n as a positive power of 2 and q as a prime number satisfying $q = 1 \bmod 2n$. The sets denoted by $\mathbb{Z}$ and $\mathbb{R}$ correspond to all integers and all real numbers, respectively. $\mathbb{Z}_q$ represents the set of integers within the range $[-q/2, q/2]$. $\mathcal{R}$ represents the quotient ring $\mathbb{Z}[x]/(x^n+1)$, and $\mathcal{R}_q$ refers to the ring $\mathbb{Z}_q[x]/(x^n+1)$, whose elements are polynomials with coefficients in $\mathbb{Z}_q$ and degree at most $n-1$. $\|\cdot\|_p$ is used to indicate the ℓ_p norm of a vector (or matrix), and $\|\cdot\|$ typically refers to the ℓ_2 norm. $[N]$ is used to symbolize the index set $\{1, 2, \cdots, N\}$. Additionally, matrices are written in bold uppercase letters, vectors in bold lowercase letters, and polynomials in lowercase. $\widetilde{\mathbf{A}}$ indicates the Gram-Schmidt orthogonalization of $\mathbf{A}$.

3.2 Anticirculant matrix

Definition 1 (Anticirculant matrix)**.** Given a polynomial $f = \sum_{i=0}^{n-1} f_i x^i$, the corresponding anticirculant matrix can be constructed from the coefficients of this polynomial as below:

$$A(f) = \begin{pmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ -f_{n-1} & f_0 & \cdots & f_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ -f_1 & -f_2 & \cdots & f_0 \end{pmatrix} = \begin{pmatrix} (f) \\ (x \cdot f) \\ \vdots \\ (x^{n-1} \cdot f) \end{pmatrix}.$$

Lemma 1 ([53])**.** Given two polynomials $f, g \in \mathcal{R}$, $A(f+g) = A(f) + A(g)$ and $A(f \cdot g) = A(f) \times A(g)$.

3.3 NTRU lattice

Definition 2 (NTRU lattice)**.** Given a polynomial degree n, a prime q, and polynomials $f, g \in \mathcal{R}$ with $h = g \cdot f^{-1} \bmod q$, the NTRU lattice can be defined as follows:

$$\mathbf{\Lambda}_{h,q} = \left\{ (v_1, v_2) \in \mathcal{R}^2 \middle| \, v_1 + h \cdot v_2 = 0 \bmod q \right\}.$$

The basis of $\mathbf{\Lambda}_{h,q}$ is

$$\mathbf{A}_{h,q} = \begin{pmatrix} -A(h) & I_n \\ qI_n & O_n \end{pmatrix},$$

where O_n and I_n denote the zero and identity matrices, respectively.

$\mathbf{A}_{h,q}$, however, is not a good basis, as it demands a complicated orthogonalization procedure. Consequently, Hoffstein et al. [58] suggested an alternative short basis $\mathbf{B}_{f,g} = \begin{pmatrix} A(g) & -A(f) \\ A(G) & -A(F) \end{pmatrix}$, with $F, G \in \mathcal{R}$ that satisfy $f \cdot G - g \cdot F = q$. According to [53], these required F and G can be computed efficiently.

Definition 3 (Ring-SIS problem)**.** Considering $\mathbf{a} = (a_1, a_2, \cdots, a_l)^{\mathrm{T}} \in \mathcal{R}_q^l$ and a real number $\beta > 0$, determine a vector $\mathbf{x} = (x_1, x_2, \cdots, x_l)^{\mathrm{T}} \in \mathcal{R}^l$ that satisfies both $\mathbf{a}^{\mathrm{T}}\mathbf{x} = \sum_{i=1}^{l} a_i \cdot x_i = 0 \bmod q$ and $0 < \|\mathbf{x}\| \leqslant \beta$.

3.4 Discrete Gaussian distribution

Definition 4 (Gaussian distribution)**.** Assuming $\mathbf{c} \in \mathbb{R}^n$ and $s > 0$, the Gaussian distribution on $\mathbb{R}^n$ is characterized by

$$\rho_{s,\mathbf{c}}(\mathbf{x}) = \exp\left(\frac{-\pi\|\mathbf{x} - \mathbf{c}\|^2}{s^2}\right)$$

with $\mathbf{x} \in \mathbb{R}^n$.

Let $\rho_{s,\mathbf{c}}(\Lambda)$ denote the total sum of $\rho_{s,\mathbf{c}}(\mathbf{x})$ over all $\mathbf{x} \in \Lambda$. Then with $\mathbf{c} \in \mathbb{R}^n$ and $s > 0$, the discrete Gaussian distribution on Λ is characterized by $\mathcal{D}_{\Lambda,s,\mathbf{c}}(\mathbf{x}) = \rho_{s,\mathbf{c}}(\mathbf{x})/\rho_{s,\mathbf{c}}(\Lambda)$. To make things easier, we define the two distributions by ρ_s and $\mathcal{D}_{\Lambda,s}$ when $\mathbf{c} = \mathbf{0}$ in this paper.

According to [59, 60], there are two critical characteristics associated with the discrete Gaussian distribution $\mathcal{D}_\sigma^n$ on $\mathbb{Z}^n$.

Lemma 2 ([59]). Assuming $\sigma > 0$ and $n \in \mathbb{Z}^+$, the subsequent facts hold true:
(1) $\Pr[\mathbf{x} \leftarrow \mathcal{D}_\sigma^n : \|\mathbf{x}\| > 2\sigma\sqrt{n}] < 2^{-n}$;
(2) $\Pr\left[x \leftarrow \mathcal{D}_\sigma^1 : |x| > 12\sigma\right] < 2^{-100}$.

Lemma 3 ([60]). Given $\mathbf{v} \in \mathbb{Z}^n$ and $\alpha > 0$, with the condition $\sigma = \omega(\|\mathbf{v}\|\sqrt{\log n})$, we can conclude that

$$\Pr\left[\mathbf{x} \leftarrow \mathcal{D}_\sigma^n : \mathcal{D}_\sigma^n(\mathbf{x})/\mathcal{D}_{\sigma,\mathbf{v}}^n(\mathbf{x}) = O(1)\right] = 1 - 2^{-\omega(\log n)},$$

and precisely, under the condition that $\sigma = \alpha\|\mathbf{v}\|$, we acquire

$$\Pr\left[\mathbf{x} \leftarrow \mathcal{D}_\sigma^n : \mathcal{D}_\sigma^n(\mathbf{x})/\mathcal{D}_{\sigma,\mathbf{v}}^n(\mathbf{x}) < e^{12/\alpha + 1/2\alpha^2}\right] > 1 - 2^{-100}.$$

With $\alpha = 12$ and $M \approx e$, we have $\mathcal{D}_\sigma^n(\mathbf{x}) \leqslant M \cdot \mathcal{D}_{\sigma,\mathbf{v}}^n(\mathbf{x})$ for $\mathbf{x} \in \mathcal{D}_\sigma^n$ with high probability.

3.5 Lattice algorithms

In this section, we emphasize two important algorithms on NTRU lattices according to [53], which are often employed in lattice-based signature schemes.

Lemma 4 ([53]). Given a prime q and a polynomial degree n, the algorithm $\mathsf{TrapGen}(n, q)$ executes in probabilistic polynomial time (PPT) and outputs a polynomial $h \in \mathcal{R}_q$ along with a basis $\mathbf{B}_{f,g}$ for $\mathbf{\Lambda}_{h,q}$.

Lemma 5 ([53]). Given a basis $\mathbf{B}_{f,g}$ of $\mathbf{\Lambda}_{h,q}$, Gaussian parameters $s > 0$ and $\mathbf{c} \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$, the PPT algorithm $\mathsf{Gaussian\_Sampler}(\mathbf{B}_{f,g}, s, \mathbf{c})$ outputs a vector $\mathbf{e} = (e_1, e_2) \in \mathcal{D}_{\mathbf{\Lambda}_{h,q},s,\mathbf{c}}$.

3.6 General forking lemma

Here, we recall the general forking lemma from [19], which is frequently used to prove the security of lattice-based signature schemes.

Lemma 6 ([19]). Let $q \geqslant 1$ be an integer and H a set of size $h \geqslant 2$. Let $\mathcal{A}$ be a randomized algorithm that on input $x, h_1, \ldots, h_q$ returns a pair, the first element of which is an integer in the range $0, \ldots, q$ and the second element of which we refer to as a side output. Let IG be a randomized algorithm that we call the input generator. The accepting probability of $\mathcal{A}$, *acc*, is defined by

$$acc = \Pr\left[J \geqslant 1 : \ x \xleftarrow{\$} \mathsf{IG};\ h_1, \ldots, h_q \xleftarrow{\$} H;\ (J, \sigma) \xleftarrow{\$} \mathcal{A}(x, h_1, \ldots, h_q)\right].$$

The forking algorithm $\mathsf{F}_\mathcal{A}$ associated to $\mathcal{A}$ is the randomized algorithm that takes input x and proceeds as Algorithm 1. Let $frk = \Pr\left[b = 1 : \ x \xleftarrow{\$} \mathsf{IG};\ (b, \sigma, \sigma') \xleftarrow{\$} \mathsf{F}_\mathcal{A}(x)\right]$. Then $frk \geqslant acc \cdot \left(\frac{acc}{q} - \frac{1}{h}\right)$.

Algorithm 1 Forking algorithm $\mathsf{F}_\mathcal{A}(x)$.

```
Pick coins ρ for A at random;
h_1, ..., h_q ←$ H; (I, σ) ←$ A(x, h_1, ..., h_q; ρ);
if I = 0 then
    return (0, ε, ε);
end if
h'_I, ..., h'_q ←$ H; (I', σ') ←$ A(x, h_1, ..., h_{I-1}, h'_I, ..., h'_q; ρ);
if I = I' and h_I ≠ h'_I then
    return (1, σ, σ');
else
    return (0, ε, ε);
end if
```

4 System model, definition, and security model of IBASM

4.1 System model of IBASM

The system model of the IBASM, as shown in Figure 2, involves five distinct parties: a trustworthy KGC, a message sender, multiple signers, a combiner, and a verifier. The responsibilities of each party in the IBASM system are described below.

Key generation center: In the IBASM system, the KGC is a reliable entity.

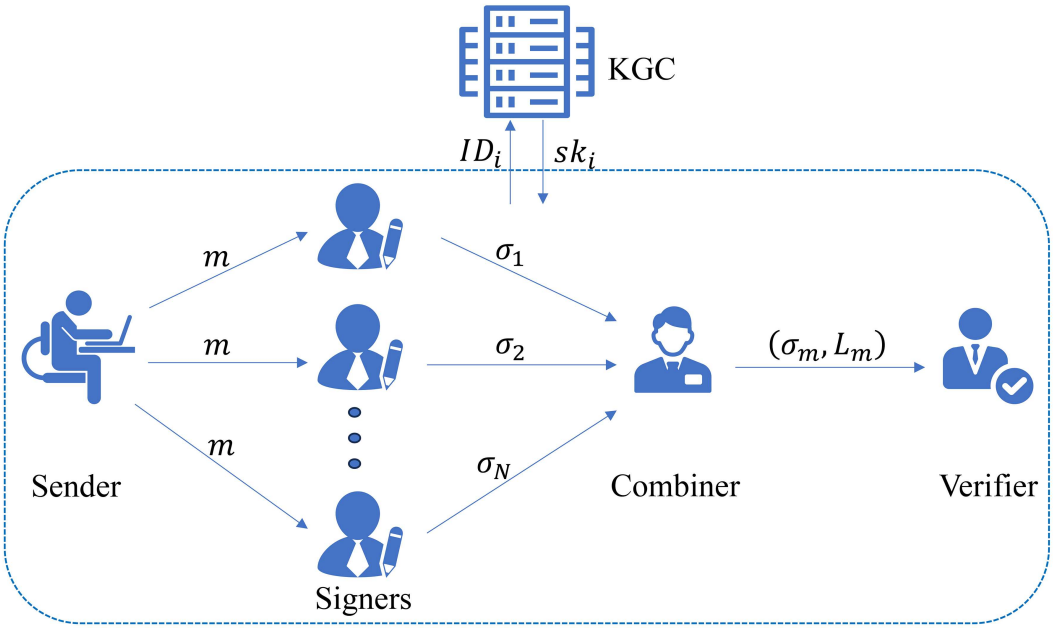

Figure 2 (Color online) System model of the IBASM.

• The KGC holds the responsibility for producing both the master public and private keys for the system. The master private key remains confidential within the KGC, while the master public key is made public to other participants.

• The KGC generates individual secret keys sk_i based on each user's identity ID_i and securely transmits these keys to the respective users.

Sender: There is a sender in the IBASM system.

• The sender is responsible for initiating the message or transaction that requires signing.

• The sender sends the message m to different signers to undergo the multi-signature process. Specifically, the sender preprocesses the original message (such as appending a timestamp and computing the hash value to ensure integrity), and then broadcasts it in parallel to each signer using methods such as the publish-subscribe model or network multicast. To ensure secure transmission, the process typically integrates secure channels, digital signatures, timestamps, and hash functions, which serve to prevent eavesdropping and tampering, verify the authenticity and integrity of the message, and effectively defend against replay attacks.

• In Bitcoin transactions, the sender is typically the party who initiates the transaction. This could be a buyer intending to pay with Bitcoin, or the originator of a transfer involving funds managed by multiple parties. They construct a raw transaction that requires multi-signatures for spending.

Signers: Multiple signers participate in the IBASM system.

• Each signer uses their individual secret key to generate a signature σ_i for the message and sends it to the combiner for the multi-signature process.

• Multiple signers must collaborate in the multi-signature process to ensure the legitimacy and trustworthiness of the document or transaction.

Combiner: The combiner plays a key role in the IBASM system.

• The combiner is responsible for collecting the resulting signatures from each participant and generating the final multi-signature σ_m.

• A valid multi-signature is only produced when the number of valid signatures meets or exceeds the required minimum number of subgroup signers.

• In Bitcoin transactions, the role of the combiner is typically implicit, handled automatically by wallet software (such as Electrum), hardware wallets, or libraries that manage the signing process. When using these tools to sign a multi-signature transaction, they assist in collecting signatures, constructing the complete script (with the necessary multi-signatures), and sending the multi-signature transaction to the network.

Verifier: The verifier is responsible for confirming the validity of multi-signatures within the IBASM system.

• The verifier checks that the length of the public key list L_m from users providing valid signatures meets or exceeds the required threshold T.

• The verifier is responsible for validating the multi-signature σ_m to ensure its authenticity.

4.2 Definition of IBASM

An IBASM scheme mainly consists of six algorithms as described below.

• **Setup**$(1^\lambda) \rightarrow pp$. Upon receiving the security parameter 1^λ, the initialization procedure creates a master secret key msk kept secretly by the KGC, and produces the public system parameters pp as its output, including the master public key.

• **Extract**$(pp, msk, ID_i) \rightarrow sk_i$. Upon receiving the system public parameters pp, master secret key msk and a user's identity ID_i, the extraction mechanism derives a secret key sk_i and transmits it to the user over a protected communication channel. (A secure encrypted communication channel is established between the KGC and the user, typically implemented through well-known security protocols like TLS/SSL (transport layer security/secure sockets layer). Before communication begins, both the KGC and the user perform mutual authentication to ensure that the user is connecting to the legitimate KGC, and that the KGC confirms the user's valid identity. They then negotiate a session key. The KGC uses its master secret key and the user's ID to generate the corresponding user private key, which is then encrypted with the session key and transmitted to the user over the established secure channel. Upon receiving the encrypted private key, the user decrypts it using the session key to obtain the corresponding private key, and securely stores the private key.)

• **Sign**$(pp, \mathbf{ID}, m, ID_i, sk_i) \rightarrow \sigma_i$. By inputting the system public parameters pp, identity list $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$, message m, along with a user's identity ID_i and secret key sk_i, the signature generation process computes a signature σ_i and delivers it to the combiner.

• **Verify**$(pp, \mathbf{ID}, m, ID_i, \sigma_i) \rightarrow 1/0$. By inputting the system public parameters pp, identity list $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$, message m, along with a user's identity ID_i and the corresponding signature σ_i, the verification algorithm checks the validity of σ_i and returns 1 ("accept") if valid, and 0 ("reject") if not.

• **ComSign**$(pp, \mathbf{ID}, m, T, \Sigma) \rightarrow \sigma_m$. Given the system public parameters pp, a list of signer identities $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$, a message m, the required minimum number T of subgroup signers, and a signature list $\Sigma = \{\sigma_1, \sigma_2, \cdots, \sigma_N\}$, the signature combination algorithm derives a multi-signature σ_m if the number of valid signatures meets or exceeds T, which is then forwarded to the verifier. Otherwise, return $\perp$.

• **MVerify**$(pp, \mathbf{ID}, m, T, \sigma_m) \rightarrow 1/0$. When provided with the system public parameters pp, a list of signer identities $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$, a message m, the required minimum number T of subgroup signers, and a multi-signature σ_m, the verification algorithm determines the validity of σ_m and produces an output of 1 ("accept") for a valid multi-signature, or 0 ("reject") for an invalid one.

4.3 Security model of IBASM

According to [19], even in the presence of only a single honest participant within the group, multi-signature schemes must remain secure against any PPT adversary $\mathcal{A}$ attempting to construct a forged valid multi-signature. According to [19, 56], we aim to achieve the existential unforgeability for our IBASM protocol, which is described by the following game.

Init. By executing the **Setup** algorithm, the challenger $\mathcal{C}$ generates pp and msk for the system. Subsequently, $\mathcal{C}$ securely keeps msk and outputs pp.

Queries. Proceeding adaptively, $\mathcal{A}$ can submit polynomially many queries as follows.

• *Hash Query.* $\mathcal{C}$ provides the corresponding hash value as the response, when $\mathcal{A}$ makes a hash oracle query with certain input parameters.

• *Key Extraction Query.* $\mathcal{C}$ performs the **Extract** algorithm to obtain the secret key sk_i and returns it to $\mathcal{A}$, when $\mathcal{A}$ queries for the secret key corresponding to identity ID_i.

• *Signing Query.* $\mathcal{A}$ can make a signing query with message m, identity list $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$, index subset $L_m \subseteq \{1, 2, \cdots, N\}$ and required minimum number T of subgroup signers. $\mathcal{C}$ first checks whether $L_m.length \geqslant T$ and returns $\perp$ if $L_m.length < T$. Otherwise, $\mathcal{C}$ computes the multi-signature σ_m (corresponding to the subset L_m) and returns it to the adversary $\mathcal{A}$, by executing the **Sign** and **ComSign** algorithms. Let I be the last index in L_m and ID_I be the identity that $\mathcal{A}$ chooses for the honest signer, without loss of generality. $\mathcal{C}$ simulates the role of ID_I, while $\mathcal{A}$ acts as all other signers (whose identities are indexed in $L_m \backslash \{I\}$). Note that not all identities (indexed in $L_m \backslash \{I\}$) need to be compromised and $\mathcal{A}$ can attempt to simulate them without their private keys.

Forgery. The adversary $\mathcal{A}$ outputs an identity set $\mathbf{ID}^* = \{ID_1^*, ID_2^*, \cdots, ID_N^*\}$, an index subset $L_m^* \subseteq \{1, 2, \cdots, N\}$, a message m^*, required minimum number T^* of subgroup signers, and a forged multi-signature

σ_m^*. $\mathcal{A}$ wins the game if (1) $\mathcal{A}$ never submitted m^* to the signing oracle, (2) L_m^* contains the index of at least one uncorrupted identity, and (3) **MVerify** $(pp, \mathbf{ID}^*, m^*, T^*, \sigma_m^*) = 1$.

Definition 5 (EUF-AID-CMA)**.** If the probability of any PPT adversary $\mathcal{A}$ winning the described game is negligible, we can conclude the IBASM scheme is existentially unforgeable against adaptively chosen-identity and chosen-message attacks within the random oracle model.

5 Our construction

This section outlines the details of our IBASM scheme. It is worth mentioning that while using simple summation to generate multi-signatures can reduce the computational overhead of our scheme, it also makes the scheme vulnerable to rogue attacks. Therefore, we employ a randomized linear combination of individual signatures instead of simple summation.

Setup: Given the security parameter 1^λ, our setup algorithm works as follows.

- Choose a positive integer n, two positive real numbers s and σ, and a prime number $q > 2$.
- Perform $\mathsf{TrapGen}(n, q) \to (h, \mathbf{B}_{f,g})$ such that $h = g \cdot f^{-1} \bmod q$ and $\mathbf{B}_{f,g} = \begin{pmatrix} A(g) & -A(f) \\ A(G) & -A(F) \end{pmatrix}$, where $f, g, F, G \in \mathcal{R}$ and $f \cdot G - g \cdot F = q$.
- Set $msk = \mathbf{B}_{f,g}$ and $mpk = h$ for the master secret and public keys, respectively.
- Select three hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q^n$, and $H_2, H_3 : \{0,1\}^* \to \{\mathbf{e} | \mathbf{e} \in \{-1, 0, 1\}^n \wedge \|\mathbf{e}\|_1 \leqslant d\}$.
- Output the system public parameters $pp = (h, H_1, H_2, H_3, q, n, s, \sigma)$.

Extract: According to a signer's identity ID_i, the KGC creates the secret key for the signer as below.

- $t_i \leftarrow H_1(ID_i) \in \mathcal{R}_q$.
- Perform $(t_i, 0) - \mathsf{Gaussian\_Sampler}(\mathbf{B}_{f,g}, s, (t_i, 0)) \to (s_{i,1}, s_{i,2})$ such that $s_{i,1} + s_{i,2} \cdot h = t_i$ and $\|(s_{i,1}, s_{i,2})\| \leqslant s\sqrt{2n}$.
- Set $sk_i = (s_{i,1}, s_{i,2})$ as the secret key and transmit it to the signer through a protected communication channel. In the practical applications, we only need to save $s_{i,2}$, as $s_{i,1}$ can be computed with $s_{i,2}$.

Sign: Our signing algorithm enables each signer to independently produce their signature, without the need for mutual interaction. For instance, the signature process for signer i is shown as follows.

- Set $\sigma_i = H_2(\mathbf{ID}, m)$ if signer i abstains or disagrees, where $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$ is a group of signer identities and m is a message.
- If signer i agrees to sign for message m, the signing algorithm works as below.

(1) $y_{i,1}, y_{i,2} \leftarrow \mathcal{D}_\sigma^n \times \mathcal{D}_\sigma^n$.
(2) $r_i = y_{i,1} + h \cdot y_{i,2} \in \mathcal{R}_q$.
(3) $c_i \leftarrow H_2(ID_i, r_i, \mathbf{ID}, m)$.
(4) Compute $z_{i,1} = s_{i,1} \cdot c_i + y_{i,1}$ with probability $\min(\frac{\mathcal{D}_\sigma^n(z_{i,1})}{M_1 \cdot \mathcal{D}_{\sigma, s_{i,1} c_i}^n(z_{i,1})}, 1)$.
(5) Compute $z_{i,2} = s_{i,2} \cdot c_i + y_{i,2}$ with probability $\min(\frac{\mathcal{D}_\sigma^n(z_{i,2})}{M_2 \cdot \mathcal{D}_{\sigma, s_{i,2} c_i}^n(z_{i,2})}, 1)$.
(6) Output the signature $\sigma_i = (\mathbf{z}_i, r_i)$, where $\mathbf{z}_i = (z_{i,1}, z_{i,2})$.

Verify: To verify signer's signature σ_i, our verification algorithm works as below.

- $c_i \leftarrow H_2(ID_i, r_i, \mathbf{ID}, m)$.
- If $z_{i,1} + h \cdot z_{i,2} = H_1(ID_i) \cdot c_i + r_i$ and $\|\mathbf{z}_i\| \leqslant 2\sigma\sqrt{2n}$, return 1 ("accept"); otherwise, return 0 ("reject").

ComSign: Run by the combiner, the multi-signature generation process is outlined as follows.

- For each σ_i in the signature list $\Sigma = (\sigma_1, \sigma_2, \cdots, \sigma_N)$,

(1) if $\sigma_i = H_2(\mathbf{ID}, m)$, the signer abstains or disagrees;
(2) if $\sigma_i = (\mathbf{z}_i, r_i)$ and $\mathbf{Verify}(pp, \mathbf{ID}, m, ID_i, \sigma_i) = 1$, save the corresponding index i into L_m.

- If $L_m.length < T$, abort and return $\perp$.
- Compute $c_i \leftarrow H_2(ID_i, r_i, \mathbf{ID}, m)$ for all $i \in L_m$.
- Compute $e_j \leftarrow H_3((c_i)_{i \in L_m}, j)$ for all $j \in L_m$.
- $\mathbf{z} = \Sigma_{j \in L_m} e_j \mathbf{z}_j$.
- Output the multi-signature $\sigma_m = (\mathbf{z}, (r_i)_{i \in L_m}, L_m)$.

MVerify: The following steps are executed by the verifier to validate the multi-signature σ_m.

- If $L_m.length < T$, return 0.
- Compute $c_i \leftarrow H_2(ID_i, r_i, \mathbf{ID}, m)$ for all $i \in L_m$.
- Compute $e_j \leftarrow H_3((c_i)_{i \in L_m}, j)$ for all $j \in L_m$.

- If $\mathbf{z}^{(1)} + h \cdot \mathbf{z}^{(2)} = \Sigma_{j \in L_m} e_j (H_1(ID_j) \cdot c_j + r_j)$ and $\|\mathbf{z}\| \leqslant 2N\sigma\sqrt{2nd}$, return 1 ("accept").
- Otherwise, return 0 ("reject").

Correctness: Below, we demonstrate the correctness for our IBASM protocol.

$$\begin{aligned}\mathbf{z}^{(1)} + h \cdot \mathbf{z}^{(2)} &= \Sigma_{j \in L_m} e_j (s_{j,1} c_j + y_{j,1} + h \cdot (s_{j,2} c_j + y_{j,2})) \\ &= \Sigma_{j \in L_m} e_j ((s_{j,1} + h \cdot s_{j,2}) \cdot c_j + r_j) \\ &= \Sigma_{j \in L_m} e_j (H_1(ID_j) \cdot c_j + r_j).\end{aligned}$$

Next, we discuss the Euclidean norm of the final multi-signature. We know that a signer's individual signature $\mathbf{z}_i$ follows the discrete Gaussian distribution $\mathcal{D}_\sigma^n \times \mathcal{D}_\sigma^n$ in the signature algorithm, so we obtain $\|\mathbf{z}_i\| \leqslant 2\sigma\sqrt{2n}$. Since the final multi-signature $\mathbf{z} = \Sigma_{j \in L_m} e_j \mathbf{z}_j$, we then have $\|\mathbf{z}\| \leqslant 2N\sigma\sqrt{2nd}$.

6 Security analysis

Theorem 1. Assuming that any PPT adversary $\mathcal{A}$ makes no more than N_H queries to random oracle (including all hash oracles), N_K to key extraction oracle, and N_S to signing oracle, and achieves a probability ϵ of forging a multi-signature, we can design a simulator $\mathcal{B}$ to solve the Ring-SIS problem with probability $\frac{\epsilon^4}{256 N_K^4 (N_H + N_S)^3}$.

Proof. Given $\mathbf{A} = [1|h] \in \mathcal{R}_q^{1\times 2}$, the pair $(\mathbf{A}, t) \in \mathcal{R}_q^{1\times 3}$ defines an instance of Ring-SIS, where t is a uniformly random polynomial in $\mathcal{R}_q$. Now we aim to construct a simulator $\mathcal{B}$ that can be used to solve this instance of Ring-SIS. Assuming Ch denotes the same challenge space for H_2 and H_3. $\mathcal{B}$ can randomly choose $h_j, h'_j \leftarrow U(Ch)$ for $j \in [N_H + N_S]$. For the random oracles H_1, H_2 and H_3, $\mathcal{B}$ maintains three hash tables HT_1, HT_2 and HT_3, which are initially empty. Additionally, $\mathcal{B}$ sets two counters ctr_1 and ctr_2, and initializes them to 0.

Init. $\mathcal{B}$ sets the system public parameters $pp = (h, H_1, H_2, H_3, q, n, s, \sigma)$, where h represents the master public key. $\mathcal{B}$ then responds pp to $\mathcal{A}$.

Queries. Proceeding adaptively, $\mathcal{A}$ can submit polynomially many queries as follows.

- *H_1 query.* When $\mathcal{A}$ makes an H_1 query with identity ID_i, $\mathcal{B}$ returns p_i to the adversary $\mathcal{A}$ if $HT_1[ID_i] = (k_i, \mathbf{s}_i, p_i)$. Otherwise, $\mathcal{B}$ randomly chooses $\mathbf{s}_i = (s_{i,1}, s_{i,2}) \in \mathcal{D}_s^n \times \mathcal{D}_s^n$ and flips a coin k_i such that $k_i = 0$ with probability δ and $k_i = 1$ with probability $1 - \delta$. If $k_i = 0$, $\mathcal{B}$ computes $p_i = s_{i,1} + h \cdot s_{i,2}$. If $k_i = 1$, $\mathcal{B}$ sets $p_i = t - (s_{i,1} + h \cdot s_{i,2})$. Finally, $\mathcal{B}$ saves $HT_1[ID_i] \leftarrow (k_i, \mathbf{s}_i, p_i)$ and returns p_i to $\mathcal{A}$.
- *H_2 query.* When $\mathcal{A}$ makes an H_2 query on $x = (ID_i, r_i, \mathbf{ID}, m)$, $\mathcal{B}$ returns c_i to the adversary $\mathcal{A}$ if $HT_2[x] = c_i$. Otherwise, if $HT_1[ID_i]$ is already defined and $k_i = 1$, $\mathcal{B}$ increases ctr_1 and sets $HT_2[x] = h_{ctr_1}$. Else, $\mathcal{B}$ randomly samples $HT_2[x] \leftarrow U(Ch)$. Finally, $\mathcal{B}$ returns $c_i = HT_2[x]$ to $\mathcal{A}$.
- *H_3 query.* When $\mathcal{A}$ sends an H_3 query with $x = ((c_i)_{i \in L_m}, j)$ for $j \in L_m$ (L_m is a subset of indices such that $L_m \subseteq \{1, 2, \cdots, N\}$), $\mathcal{B}$ returns e_j to the adversary $\mathcal{A}$ if $HT_3[x] = e_j$. Otherwise, assume that I is the last index in L_m, without loss of generality. For all $j \in L_m \backslash \{I\}$, $\mathcal{B}$ randomly selects $HT_3[x_j] \leftarrow U(Ch)$ for $x_j = ((c_i)_{i \in L_m}, j)$. It then increments ctr_2 and sets $HT_3[x_I] = h'_{ctr_2}$ for $x_I = ((c_i)_{i \in L_m}, I)$. Finally, $\mathcal{B}$ returns $e_j = HT_3[x]$ (where $x = x_j$ for index $j \in L_m$) to $\mathcal{A}$.
- *Key extraction query.* When $\mathcal{A}$ requests the secret key for identity ID_i, $\mathcal{B}$ first issues an H_1 query with ID_i and looks up $HT_1[ID_i] = (k_i, \mathbf{s}_i, p_i)$. If $k_i = 1$, $\mathcal{B}$ assigns $bad \leftarrow \mathsf{true}$ and ends the process. If $k_i = 0$, $\mathcal{B}$ returns $\mathbf{s}_i = (s_{i,1}, s_{i,2})$ to $\mathcal{A}$.
- *Signing query.* Upon receiving a signing query from $\mathcal{A}$ with message m, identity list $\mathbf{ID} = \{ID_1, ID_2, \cdots, ID_N\}$, an index subset $L_m \subseteq \{1, 2, \cdots, N\}$ and required minimum number T of subgroup signers, $\mathcal{B}$ first checks whether $L_m.length \geqslant T$ and returns $\perp$ if $L_m.length < T$. Otherwise, suppose that I is the final index in L_m and ID_I is the identity designated by $\mathcal{A}$ for the honest signer, without loss of generality. $\mathcal{B}$ then acts as ID_I, whereas $\mathcal{A}$ takes on the role of all other signers (whose indices are present in $L_m \backslash \{I\}$). Subsequently, $\mathcal{B}$ performs an H_1 query with ID_I and looks up $HT_1[ID_I] = (k_I, \mathbf{s}_I, p_I)$. If $k_I = 0$, $\mathcal{B}$ honestly follows the real **Sign** algorithm and produces a partial signature σ_I with the secret key $\mathbf{s}_I = (s_{I,1}, s_{I,2})$. If $k_I = 1$, $\mathcal{B}$ increases ctr_1 and sets $c_I = h_{ctr_1}$. It randomly samples $\mathbf{z}_I = (z_{I,1}, z_{I,2}) \leftarrow \mathcal{D}_\sigma^n \times \mathcal{D}_\sigma^n$ and computes $r_I = z_{I,1} + h \cdot z_{I,2} - H_1(ID_I) \cdot c_I$. $\mathcal{B}$ then stores $HT_2[x] = c_I$ with $x = (ID_I, r_I, \mathbf{ID}, m)$ and generates the partial signature $\sigma_I = (\mathbf{z}_I = (z_{I,1}, z_{I,2}), r_I)$. After receiving the other partial signatures σ_i for $i \in L_m \backslash \{I\}$ generated by $\mathcal{A}$ itself, $\mathcal{B}$ performs the **ComSign** algorithm to derive the multi-signature σ_m (corresponding to L_m) and returns σ_m to $\mathcal{A}$.

Forgery. The adversary $\mathcal{A}$ outputs an identity list $\mathbf{ID}^* = \{ID_1^*, ID_2^*, \cdots, ID_N^*\}$, an index subset $L_m^* \subseteq \{1, 2, \cdots, N\}$, along with message m^*, required minimum number T^* of subgroup signers, and a forged multi-signature $\sigma_m^* = (\mathbf{z}^*, (r_i^*)_{i \in L_m^*}, L_m^*)$. $\mathcal{B}$ checks the validity of the forged multi-signature σ_m^*, i.e., (1) $\mathcal{A}$ never

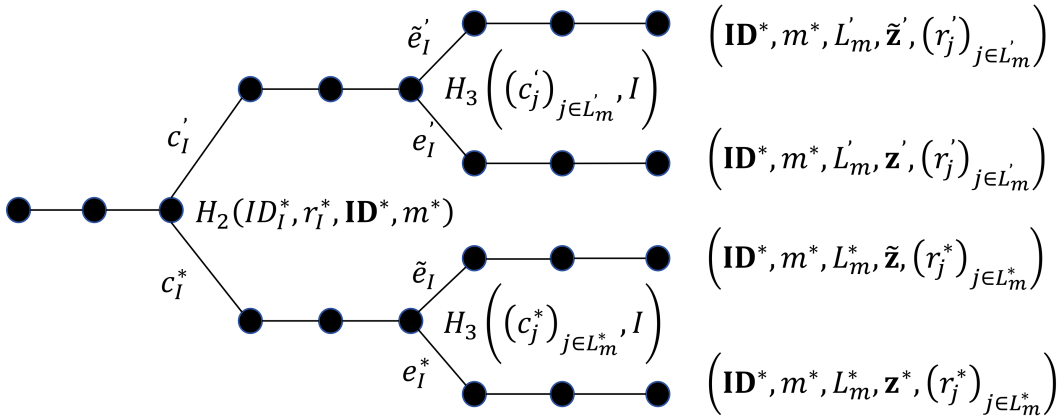

Figure 3 Double forking lemma.

submitted m^* to the signing oracle, (2) L_m^* contains the index of at least one uncorrupted identity, and (3) $\mathbf{MVerify}(pp, \mathbf{ID}^*, m^*, T^*, \sigma_m^*) = 1$.

If any of the above conditions do not hold, $\mathcal{B}$ outputs $(0, 0, \perp)$. Otherwise, assume that I represents the final index in L_m^* and ID_I^* is an uncorrupted identity, without loss of generality. Suppose that J and J' denote the counter indices such that $c_I^* = h_J$ and $e_I^* = h'_{J'}$, respectively. Subsequently, $\mathcal{B}$ outputs $(J, J', (\mathbf{ID}^*, L_m^*, m^*, \sigma_m^*, \mathbf{C}^*, \mathbf{E}^*))$, where $\mathbf{C}^* = (c_j^*)_{j\in L_m^*}$ such that $c_j^* = H_2(ID_j^*, r_j^*, \mathbf{ID}^*, m^*)$ and $\mathbf{E}^* = (e_j^*)_{j\in L_m^*}$ such that $e_j^* = H_3(\mathbf{C}^*, j)$. Finally, we apply the double forking lemma (as shown in Figure 3) to complete the security analysis.

Let $out = (J, \mathbf{ID}^*, L_m^*, m^*, \sigma_m^*, \mathbf{C}^*, \mathbf{E}^*)$. we employ the forking lemma to derive another output $\widetilde{out} = (\widetilde{J}, \widetilde{\mathbf{ID}}, \widetilde{L}_m, \widetilde{m}, \widetilde{\sigma}_m, \widetilde{\mathbf{C}}, \widetilde{\mathbf{E}})$, where the forking point occurs at the counter index ctr_2. According to the forking lemma, both simulation environments provided by $\mathcal{B}$ to $\mathcal{A}$ are the same before reaching the fork. Therefore, we have $J = \widetilde{J}$, $\mathbf{ID}^* = \widetilde{\mathbf{ID}}$, $L_m^* = \widetilde{L}_m$, $m^* = \widetilde{m}$, and $c_j^* = \widetilde{c}_j$ for all $j \in L_m^*$ and $e_j^* = \widetilde{e}_j$ for all $j \in L_m^* \backslash \{I\}$. Since both generated multi-signatures are valid, we know that $\mathbf{z}^{*(1)} + h \cdot \mathbf{z}^{*(2)} = \Sigma_{j\in L_m^*} e_j^*(H_1(ID_j^*) \cdot c_j^* + r_j^*)$ and $\widetilde{\mathbf{z}}^{(1)} + h \cdot \widetilde{\mathbf{z}}^{(2)} = \Sigma_{j\in \widetilde{L}_m} \widetilde{e}_j \left(H_1\left(\widetilde{ID}_j\right) \cdot \widetilde{c}_j + \widetilde{r}_j\right)$. Given $\mathbf{A} = [1|h]$, we can compute

$$\mathbf{A}(\mathbf{z}^* - \widetilde{\mathbf{z}}) = (e_I^* - \widetilde{e}_I)(H_1(ID_I^*) \cdot c_I^* + r_I^*). \tag{1}$$

As can be noticed, Eq. (1) alone does not provide a complete solution for the Ring-SIS problem. Next, we will apply the forking lemma for the second time. Let $out = (\mathbf{ID}^*, L_m^*, m^*, \sigma_m^*, \mathbf{C}^*, \mathbf{E}^*)$. By performing the forking lemma again, we can attain another output $out' = (\mathbf{ID}', L'_m, m', \sigma'_m, \mathbf{C}', \mathbf{E}')$. This time, the forking point occurs at the counter index ctr_1. In the same manner, both simulation environments are identical before reaching the fork. Hence, we have $\mathbf{ID}^* = \mathbf{ID}', m^* = m'$, and $r_I^* = r'_I$, along with $c_I^* \neq c'_I$. Therefore, we can derive

$$\mathbf{A}(\mathbf{z}' - \widetilde{\mathbf{z}}') = (e'_I - \widetilde{e}'_I)(H_1(ID'_I) \cdot c'_I + r'_I) = (e'_I - \widetilde{e}'_I)(H_1(ID_I^*) \cdot c'_I + r_I^*). \tag{2}$$

Assume $\zeta = (e_I^* - \widetilde{e}_I)$ and $\zeta' = (e'_I - \widetilde{e}'_I)$. Multiply Eq. (1) by ζ' and Eq. (2) by ζ. Combining both equations, we can deduce

$$\mathbf{A}(\zeta'(\mathbf{z}^* - \widetilde{\mathbf{z}}) - \zeta(\mathbf{z}' - \widetilde{\mathbf{z}}')) = \zeta\zeta'(H_1(ID_I^*)(c_I^* - c'_I)).$$

Given $H_1(ID_I^*) = t - \mathbf{A}\mathbf{s}_I^* = t - (s_{I,1}^* + h \cdot s_{I,2}^*)$, we can obtain

$$\mathbf{A}(\zeta'(\mathbf{z}^* - \widetilde{\mathbf{z}}) - \zeta(\mathbf{z}' - \widetilde{\mathbf{z}}') + \zeta\zeta' \cdot \mathbf{s}_I^*(c_I^* - c'_I)) = \zeta\zeta' \cdot t(c_I^* - c'_I).$$

Suppose $\mathbf{A}' = [\mathbf{A}|t] \in \mathcal{R}_q^{1\times 3}$, we can compute $\mathbf{x} = (\zeta'(\mathbf{z}^* - \widetilde{\mathbf{z}}) - \zeta(\mathbf{z}' - \widetilde{\mathbf{z}}') + \zeta\zeta' \cdot \mathbf{s}_I^*(c_I^* - c'_I), \zeta\zeta'(c'_I - c_I^*))^{\mathrm{T}}$, a solution to the Ring-SIS problem for $\mathbf{A}'$. Subsequently, we can calculate

$$\|\mathbf{x}\| \leqslant 4 \cdot \|\zeta\mathbf{z}^*\| + 2 \cdot \|\zeta^2\mathbf{s}_I^*c_I^*\| + 2 \cdot \|\zeta^2 c_I^*\| \leqslant 4 \cdot 4d \cdot 2N\sigma\sqrt{2nd} + 2 \cdot s\sqrt{2n} \cdot 4d^{\frac{3}{2}} + 2 \cdot 4d^{\frac{3}{2}} = \beta.$$

Let ϵ be the probability of $\mathcal{A}$ succeeding in deriving a forgery. Since L_m^* contains the index of at least one uncorrupted identity and $\mathcal{A}$ submits at most N_K key queries, we can compute $\Pr[\neg bad] \geqslant \delta^{N_K}(1 - \delta)$. When the condition $\delta = \frac{N_K}{N_K+1}$ is satisfied, $\Pr[\neg bad]$ can make the maximum value. Hence, we can obtain

$$\Pr[\neg bad] \geqslant \frac{1}{N_K + 1} \cdot \left(\frac{N_K}{N_K + 1}\right)^{N_K} = \frac{1}{N_K} \cdot \left(1 - \frac{1}{N_K + 1}\right)^{N_K+1} \geqslant \frac{1}{4N_K}.$$

Table 1 Comparison among lattice-based multi-signature schemes. n is the system parameter; k, ℓ, and ℓ' are the dimensions of modular matrices; $m \times n$ represents the dimensions of integer matrices; N is the number of signers; $|sk|$ indicates user's secret key size; $|pk|$ denotes user's public key size; $|\sigma_m|$ represents the multi-signature size; $|ID|$ signifies the size of user's identity; $\widetilde{\mathcal{O}}(g(n))$ denotes the complexity of $\mathcal{O}(g(n)\log^c(g(n)))$. EUF-CMA: existentially unforgeable under chosen-message attacks. EUF-AID-CMA: existentially unforgeable under adaptive-identity and chosen-message attacks. EUF-SID-CMA: existentially unforgeable under selective-identity and chosen-message attacks.

Scheme	Storage consumption			IBC	NIS	ASM	Lattice type	Security
	$\|sk\|$	$\|pk\|$	$\|\sigma_m\|$					
DOTT [46]	$(\ell+k)\mathcal{O}(n)$	$k\cdot\widetilde{\mathcal{O}}(n)$	$(\ell+3k+3k\ell)\widetilde{\mathcal{O}}(n)$	×	×	×	Modular lattice	EUF-CMA
MuSig-L [48]	$(\ell+k)\mathcal{O}(n)$	$k\cdot\widetilde{\mathcal{O}}(n)$	$(\ell+2k)\widetilde{\mathcal{O}}(n)$	×	×	×	Modular lattice	EUF-CMA
DualMS [54]	$(\ell+k)\mathcal{O}(n)$	$k\cdot\widetilde{\mathcal{O}}(n)$	$(\ell+\ell'+2k)\widetilde{\mathcal{O}}(n)$	×	×	×	Modular lattice	EUF-CMA
MSig [55]	$\mathcal{O}(mn)$	$\widetilde{\mathcal{O}}(n^2)$	$\widetilde{\mathcal{O}}(m)$	×	×	×	Euclidean lattice	EUF-CMA
IBMS-pka [57]	$\widetilde{\mathcal{O}}(mn)$	$\|ID\|$	$\widetilde{\mathcal{O}}(m)$	√	×	×	Euclidean lattice	EUF-AID-CMA
MS [45]	$\mathcal{O}(n)$	$\widetilde{\mathcal{O}}(n)$	$(N+2)\mathcal{O}(n)$	×	×	×	Ideal lattice	EUF-CMA
IBMS [50]	$\widetilde{\mathcal{O}}(n)$	$\|ID\|$	$\widetilde{\mathcal{O}}(n)+N\cdot\mathcal{O}(n)$	√	×	×	Ideal lattice	EUF-SID-CMA
IBAS [56]	$\widetilde{\mathcal{O}}(n)$	$\|ID\|$	$(N+1)\widetilde{\mathcal{O}}(n)$	√	×	×	Ideal lattice	EUF-AID-CMA
Our IBASM	$\widetilde{\mathcal{O}}(n)$	$\|ID\|$	$(N+2)\widetilde{\mathcal{O}}(n)$	√	√	√	Ideal lattice	EUF-AID-CMA

Now, the accepting probability is $acc \geqslant \epsilon \cdot \Pr[\neg bad] \geqslant \frac{\epsilon}{4N_K}$.

Based on the forking lemma [19], the successful probability of the first fork is

$$frk_1 \geqslant acc \cdot \left(\frac{acc}{N_H+N_S} - \frac{1}{|Ch|}\right) \geqslant \frac{\epsilon^2}{16N_K^2(N_H+N_S)} - \frac{1}{|Ch|} \approx \frac{\epsilon^2}{16N_K^2(N_H+N_S)}.$$

The successful probability of the second fork is

$$frk_2 \geqslant frk_1 \cdot \left(\frac{frk_1}{N_H+N_S} - \frac{1}{|Ch|}\right) \geqslant \frac{\left(\frac{\epsilon^2}{16N_K^2(N_H+N_S)}\right)^2}{N_H+N_S} - \frac{1}{|Ch|} \approx \frac{\epsilon^4}{256N_K^4(N_H+N_S)^3}.$$

Therefore, the probability is at least $\frac{\epsilon^4}{256N_K^4(N_H+N_S)^3}$ to solve the Ring-SIS problem. Furthermore, it is non-negligible if ϵ, the probability of $\mathcal{A}$ successfully generating a forgery, is non-negligible. However, since the Ring-SIS problem is widely recognized as a difficult problem on lattices, our IBASM scheme remains secure within the random oracle model.

7 Performance evaluation

7.1 Theoretical evaluation

We evaluate our IBASM scheme against several related lattice-based multi-signature schemes [45, 46, 48, 50, 54–57], as presented in Table 1. The comparison covers the following aspects: the sizes of user's public and private keys, the size of multi-signature, support for identity-based cryptographic design (IBC), non-interactive signature protocol (NIS), and ASM, lattice type, and security.

Regarding storage consumption, the DOTT, MuSig-L, and DualMS schemes [46, 48, 54] are based on modular lattices, and their storage costs for public keys, private keys, and multi-signatures depend not only on the degree of polynomials but also on the dimensions of modular matrices. Both the MSig [55] and IBMS-pka [57] schemes are based on Euclidean lattices, and their public key, private key, and multi-signature sizes are primarily determined by the dimensions of integer matrices, which generally leads to large storage overhead. In contrast, the MS, IBMS, and IBAS schemes [45, 50, 56] are designed based on ideal lattices (the NTRU lattice is also an ideal lattice). Since the number of polynomials is constant, the storage requirements for public keys and private keys in these schemes are primarily influenced by the polynomial degree. Additionally, the IBMS [50], IBAS [56], and our IBASM schemes are constructed based on IBC (identity-based cryptography) and NTRU lattices, meaning the public key size is mainly determined by user's identity information, with the private key size being the same. However, since our scheme focuses on non-interactive and accountable-subgroup multi-signatures, the storage overhead for multi-signatures is higher compared to other schemes. To make it more intuitive, we present specific sizes of secret keys, public keys, and multi-signatures for the IBMS [50], IBAS [56], and our scheme, as shown in Table 2. The experimental parameters are set to $n = 512$, $q = 2^{20}$ and $N = 5$, with the public key size representing the evaluation of the publicly-known cryptographic hash function for user identity. From Table 2, it can be seen that the specific storage consumption for each scheme is consistent with the analysis above.

Table 2 Storage cost of keys and multi-signatures ($n = 512$, $q = 2^{20}$ and $N = 5$).

Scheme	Secret keys (KB)	Public keys (KB)	Multi-signatures (KB)
IBMS [50]	1.25	1.25	3.75
IBAS [56]	1.25	1.25	7.5
Our IBASM	1.25	1.25	8.75

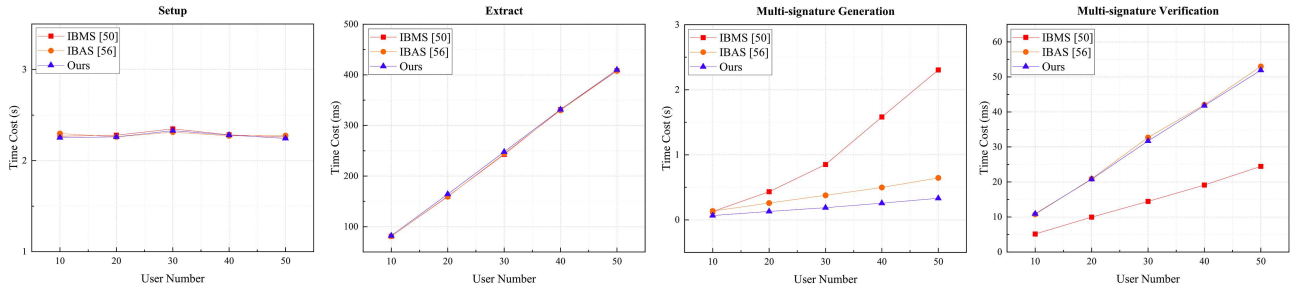

Figure 4 (Color online) Time cost of each algorithm ($n = 512, q = 2^{20}$).

From the perspective of features, our IBASM scheme is an identity-based multi-signature protocol, where each signer's public key is linked to their identity (e.g., email address), eliminating the need to issue certificates for each user's public key. This effectively resolves the complex certificate management issues associated with PKI systems. Moreover, our signature protocol is non-interactive, meaning there is no need for signers to interact with each other to generate multi-signatures, significantly reducing the communication overhead. Additionally, if any signer needs to restart the signature process, it will not affect other participants. In contrast, in broadcasting multi-signature protocols, if any participant needs to restart the signing algorithm, all signers must restart, which severely impacts the overall efficiency of the signing process. Finally, our approach supports flexible accountable-subgroup multi-signatures, making it suitable for practical t-out-of-N scenarios, whereas the entire group of signers is predetermined in other schemes.

In terms of security, both the IBAS scheme [56] and our scheme are EUF-AID-CMA secure, while the IBMS scheme [50] is EUF-SID-CMA secure. Therefore, our scheme provides higher security compared to the IBMS scheme. Specifically, we grant the adversary stronger attack capabilities, but our scheme remains existentially unforgeable against adaptively chosen-identity and chosen-message attacks. On the other hand, the IBMS scheme [50] requires the adversary to select the target identity in advance, making it secure only against selective identity attacks, which results in a lower level of security.

7.2 Experimental evaluation

As mentioned in [61], the NTRU lattice-based signature scheme, integrated with fast Fourier transform (FFT) technology, demonstrates high performance in both bandwidth and efficiency, as FFT can significantly improve the efficiency of polynomial multiplication. Therefore, our approach is built upon NTRU lattices. So are the IBMS [50] and IBAS [56] schemes. To evaluate the practical performance of our approach, we implemented these three schemes for a comprehensive comparison. We conducted the experiments using a personal computer (PC) with 8 GB of memory, an Intel Core i7-8550U processor, and a Linux operating system. The IBMS, IBAS, and our IBASM schemes were implemented using the C++ programming language and the NTL library [62]. Additionally, we set two parameter combinations for comparison in the experiments: $n = 512$, $q = 2^{20}$ and $n = 1024$, $q = 2^{30}$. The corresponding results are presented in Figures 4 and 5.

Setup algorithm. The setup algorithm primarily involves executing the TrapGen algorithm to produce the system's master public and private keys. Since the computational cost does not change with the number of users, the time consumption of the setup algorithm remains relatively stable.

Extract algorithm. The key extraction algorithm is designed to create a secret key for each user based on their identity information. Clearly, the computational effort increases in direct proportion to the number of users. Consequently, an increase in the number of users results in a corresponding rise in the time cost.

Multi-signature generation. The IBMS scheme [50] is a broadcasting multi-signature protocol, where if any participant needs to restart the signing process, all signers must restart as well, leading to an exponential increase in computational cost with respect to the number of participants. This results in a highly inefficient system, particularly when the number of signers is large, as the computational cost becomes significantly burdensome. The IBAS scheme [56], a non-broadcasting multi-signature protocol, introduces a trusted third party. In this scheme,

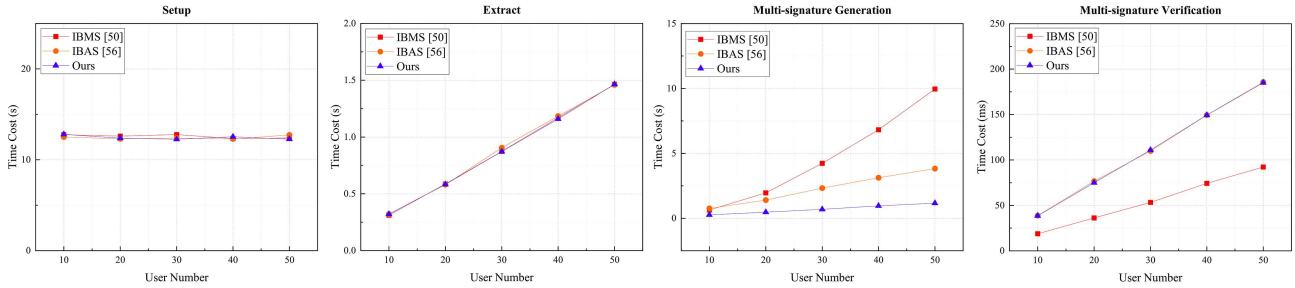

Figure 5 (Color online) Time cost of each algorithm ($n = 1024, q = 2^{30}$).

signers engage only with the trusted third party, and the computational effort for generating multi-signatures increases proportionally to the number of participants. However, the involvement of the trusted third party causes some additional time costs. Finally, our scheme is a non-interactive multi-signature approach, requiring neither broadcasting interactions between signers nor a trusted third party. Consequently, our multi-signature generation is significantly quicker compared to both the IBMS and IBAS schemes. In particular, when compared to the IBAS scheme, our multi-signature generation efficiency is enhanced by approximately 50% and 66.7%, respectively, in two different parameter combinations.

Multi-signature verification. As observed, the time required for multi-signature verification in our approach is comparable to that in the IBAS scheme [56]. Additionally, since our scheme computes multi-signatures using a weighted sum rather than direct summation (to ensure security), the verification process involves more polynomial multiplications, leading to a slightly higher time cost than the IBMS scheme [50]. However, this is still reasonable and acceptable, considering more functionality our approach provides.

In summary, our scheme demonstrates superior overall performance compared to other schemes, particularly in terms of multi-signature generation efficiency. As more signers are involved, the advantage of our multi-signature generation becomes even more pronounced.

8 Conclusion

To make it more suitable for practical applications like the Bitcoin blockchain, we proposed a non-interactive identity-based accountable-subgroup multi-signature scheme based on lattices in this paper. We have demonstrated its security under the random oracle model. Our scheme supports accountable-subgroup multi-signatures, allowing users to independently decide whether to sign for Bitcoin transactions. Unlike previous lattice-based multi-signature schemes, which require all participants to be determined in advance, our scheme is more flexible and better suited for practical t-out-of-N transactions in blockchain environments. Moreover, our signing algorithm is non-interactive, significantly improving the efficiency of multi-signature generation. Additionally, if any participant needs to restart the signing process, it does not affect other participants. Furthermore, our scheme is based on IBC, allowing the derivation of public keys from personal identity details (e.g., ID card numbers or email addresses), making it more appropriate for practical system development and deployment.

Finally, as part of our future research, we aim to shrink the size of multi-signatures and design more efficient lattice-based multi-signatures and aggregate signatures.

Acknowledgements This work was supported by Major Program of Guangdong Basic and Applied Research (Grant No. 2019B030302008), National Natural Science Foundation of China (Grant No. 62272174), and Science and Technology Program of Guangzhou (Grant No. 2024A04J6542). Willy SUSILO was supported by Australian Research Council (ARC) Laureate Fellowship (Grant No. FL230100033).

References

1 Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. 2008. http://bitcoin.org/bitcoin.pdf
2 Diffie W, Hellman M. New directions in cryptography. IEEE Trans Inform Theor, 1976, 22: 644–654
3 Andresen G. M-of-N standard transactions. Bitcoin improvement proposal, 2011. https://github.com/bitcoin/bips/blob/master/bip-0011.mediawiki
4 Hanifatunnisa R, Rahardjo B. Blockchain based e-voting recording system design. In: Proceedings of the 11th International Conference on Telecommunication Systems Services and Applications (TSSA), 2017. 1–6
5 Itakura K, Nakamura K. A public-key cryptosystem suitable for digital multisignatures. NEC Res Dev, 1983, 71: 1–8
6 Maxwell G, Poelstra A, Seurin Y, et al. Simple Schnorr multi-signatures with applications to Bitcoin. Des Codes Cryptogr, 2019, 87: 2139–2164

7 Boneh D, Drijvers M, Neven G. Compact multi-signatures for smaller blockchains. In: Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, 2018. 435–464

8 Desmedt Y. Threshold cryptosystems. In: Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques, 1993. 1–14

9 Shamir A. Identity-based cryptosystems and signature schemes. In: Proceedings of the Annual International Cryptology Conference, 1984. 47–53

10 Shor P. Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings of the 35th Annual Symposium on Foundations of Computer Science, 1994. 124–134

11 Grover L K. Quantum mechanics helps in searching for a needle in a haystack. Phys Rev Lett, 1997, 79: 325–328

12 Ajtai M. Generating hard instances of lattice problems (extended abstract). In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing, 1996. 99–108

13 Ajtai M, Dwork C. A public-key cryptosystem with worst-case/average-case equivalence. In: Proceedings of the 29th Annual ACM Symposium on Theory of Computing, 1997. 284–293

14 Wang J B, Liu L, Lyu S X, et al. Quantum-safe cryptography: crossroads of coding theory and cryptography. Sci China Inf Sci, 2022, 65: 111301

15 Pan J, Zhang J, Zhang F G, et al. Lattice-based group encryptions with only one trapdoor. Sci China Inf Sci, 2022, 65: 152304

16 Yao L S, Hou L, Weng J, et al. Provably secure attribute-based authenticated encryption with keyword search from ideal lattices. Sci China Inf Sci, 2024, 67: 119101

17 Xie W L, Weng J, Tong Y, et al. Lattice-based autonomous path proxy re-encryption in the standard model. Sci China Inf Sci, 2023, 66: 202101

18 Wang M, Long G-L. Lattice-based access authentication scheme for quantum communication networks. Sci China Inf Sci, 2024, 67: 222501

19 Bellare M, Neven G. Multi-signatures in the plain public-key model and a general forking lemma. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, 2006. 390–399

20 Bagherzandi A, Cheon J H, Jarecki S. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: Proceedings of the 15th ACM Conference on Computer and Communications Security, 2008. 449–458

21 Ma C, Weng J, Li Y, et al. Efficient discrete logarithm based multi-signature scheme in the plain public key model. Des Codes Cryptogr, 2009, 54: 121–133

22 Syta E, Tamas I, Visher D, et al. Keeping authorities "honest or bust" with decentralized witness cosigning. In: Proceedings of the IEEE Symposium on Security and Privacy (SP), 2016. 526–545

23 Schnorr C P. Efficient signature generation by smart cards. J Cryptology, 1991, 4: 161–174

24 Drijvers M, Edalatnejad K, Ford B, et al. On the security of two-round multi-signatures. In: Proceedings of the IEEE Symposium on Security and Privacy (SP), 2019. 1084–1101

25 Benhamouda F, Lepoint T, Loss J, et al. On the (in)Security of ROS. J Cryptol, 2022, 35: 25

26 Nick J, Ruffing T, Seurin Y, et al. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, 2020. 1717–1731

27 Alper H K, Burdges J. Two-round trip Schnorr multi-signatures via delinearized witnesses. In: Proceedings of the Annual International Cryptology Conference, 2021. 157–188

28 Nick J, Ruffing T, Seurin Y. MuSig2: simple two-round Schnorr multi-signatures. In: Proceedings of the Annual International Cryptology Conference, 2021. 189–221

29 Bellare M, Dai W. Chain reductions for multi-signatures and the HBMS scheme. In: Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, 2021. 650–678

30 Bellare M, Neven G. Identity-based multi-signatures from RSA. In: Proceedings of the Cryptographers' Track at the RSA Conference, 2006. 145–162

31 Bagherzandi A, Jarecki S. Identity-based aggregate and multi-signature schemes based on RSA. In: Proceedings of the International Workshop on Public Key Cryptography, 2010. 480–498

32 Islam S H, Das A K, Khan M K. Design of a provably secure identity-based digital multi-signature scheme using biometrics and fuzzy extractor. Security Comm Netws, 2016, 9: 3229–3238

33 Wei L, Zhang L, Huang D, et al. PSDAAP: provably secure data authenticated aggregation protocols using identity-based multi-signature in marine WSNs. Sensors, 2017, 17: 2117

34 Babu A R, Gayathri N B, Reddy P V. Efficient ID-based key-insulated multi-signature scheme without pairings. In: Proceedings of the Innovations in Power and Advanced Computing Technologies (i-PACT), 2019. 1–6

35 Liu H, Han D, Cui M, et al. IdenMultiSig: identity-based decentralized multi-signature in Internet of Things. IEEE Trans Comput Soc Syst, 2023, 10: 1711–1721

36 Micali S, Ohta K, Reyzin L. Accountable-subgroup multisignatures: extended abstract. In: Proceedings of the 8th ACM Conference on Computer and Communications Security, 2001. 245–254

37 Kara M, Laouid A, Hammoudeh M. An efficient multi-signature scheme for blockchain. 2023. https://eprint.iacr.org/2023/078

38 Bansarkhani R E, Sturm J. An efficient lattice-based multisignature scheme with applications to bitcoins. In: Proceedings of the International Conference on Cryptology and Network Security, 2016. 140–155

39 Ma C, Jiang M. Practical lattice-based multisignature schemes for blockchains. IEEE Access, 2019, 7: 179765

40 Fukumitsu M, Hasegawa S. A tightly-secure lattice-based multisignature. In: Proceedings of the 6th ASIA Public-Key Cryptography Workshop, 2019. 3–11

41 Fukumitsu M, Hasegawa S. A lattice-based provably secure multisignature scheme in quantum random oracle model. In: Proceedings of the International Conference on Provable and Practical Security, 2020. 45–64

42 Peng C, Du X. New lattice-based digital multi-signature scheme. In: Proceedings of the 6th International Conference of Pioneering Computer Scientists, Engineers and Educators, 2020. 129–137

43 Ajtai M. Generating hard instances of the short basis problem. In: Proceedings of the 26th International Colloquium on Automata, Languages, and Programming, 1999. 1–9

44 Kansal M, Dutta R. Round optimal secure multisignature schemes from lattice with public key aggregation and signature compression. In: Proceedings of the 12th International Conference on Cryptology in Africa, 2020. 281–300

45 Kansal M, Singh A K, Dutta R. Efficient multi-signature scheme using lattice. Comput J, 2021, 65: 2421–2429

46 Damgård I, Orlandi C, Takahashi A, et al. Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. In: Proceedings of the International Workshop on Public Key Cryptography, 2021. 99–130

47 Baum C, Damgård I, Lyubashevsky V, et al. More efficient commitments from structured lattice assumptions. In: Proceedings of the International Conference on Security and Cryptography for Networks, 2018. 368–385

48 Boschini C, Takahashi A, Tibouchi M. MuSig-L: lattice-based multi-signature with single-round online phase. In: Proceedings of the Annual International Cryptology Conference, 2022. 276–305

49 Fleischhacker N, Simkin M, Zhang Z. Squirrel: efficient synchronized multi-signatures from lattices. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, 2022. 1109–1123

50 Chen X, Huang Q, Li H, et al. A novel identity-based multi-signature scheme over NTRU lattices. Theor Comput Sci, 2022, 933: 163–176

51 Chen X, Huang Q, Huang J. A novel certificateless multi-signature scheme over NTRU lattices. In: Proceedings of the 16th International Conference on Information Security Practice and Experience, 2021. 345–362

52 Liang X, Wang X, Zhang Q, et al. A lattice-based multisignature scheme for blockchain-enabled systems. In: Proceedings of the International Conference on Emerging Networking Architecture and Technologies, 2023. 336–346

53 Ducas L, Durmus A, Lepoint T, et al. Lattice signatures and bimodal Gaussians. In: Proceedings of the Annual International Cryptology Conference, 2013. 40–56

54 Chen Y. DualMS: efficient lattice-based two-round multi-signature with trapdoor-free simulation. In: Proceedings of the Annual International Cryptology Conference, 2023. 716–747

55 Phalakarn K, Suppakitpaisarn V, Hasan M A. Single-round lattice-based multisignatures. In: Proceedings of the 9th International Symposium on Computing and Networking Workshops (CANDARW), 2021. 365–371

56 Li Q, Luo M, Hsu C, et al. A quantum secure and noninteractive identity-based aggregate signature protocol from lattices. IEEE Syst J, 2022, 16: 4816–4826

57 Chen X, Huang J, Xiao K, et al. A non-interactive identity-based multi-signature scheme on lattices with public key aggregation. IEEE Trans Dependable Secure Comput, 2025, 22: 4189–4199

58 Hoffstein J, Howgrave-Graham N, Pipher J, et al. NTRUSign: digital signatures using the NTRU lattice. In: Proceedings of the Cryptographers' Track at the RSA Conference, 2003. 122–140

59 Tian M, Huang L. Identity-based signatures from lattices: simpler, faster, shorter. Fundamenta Informaticae, 2016, 145: 171–187

60 Lyubashevsky V. Lattice signatures without trapdoors. In: Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2012. 738–755

61 Yu Y, Jia H, Wang X. Compact lattice gadget and its applications to hash-and-sign signatures. In: Proceedings of the Annual International Cryptology Conference, 2023. 390–420

62 Shoup V. NTL: a library for doing number theory. 2001. https://www.shoup.net/ntr/