Identity-Based Signature from Lattices without Trapdoors

Pingbin Luo[1], Xinjian Chen[1], Willy Susilo[3], Qiong Huang[1,2] ⋆

[1] College of Mathematics and Informatics, South China Agricultural University
[2] Guangzhou Key Laboratory of Intelligent Agriculture, Guangzhou 510642, China
[3] School of Computing and Information Technology, University of Wollongong
robin2022@stu.scau.edu.cn, wsusilo@uow.edu.au, qhuang@scau.edu.cn

Abstract. Currently, the majority of lattice-base identity-based signature (LB-IBS) schemes rely on lattice trapdoor technique introduced by Gentry, Peikert, and Vaikuntanathan. However, this approach significantly impacts computational efficiency. In light of this, we present an alternative method for constructing LB-IBS that does not rely on lattice trapdoors and preimage sampling function. We construct an IBS scheme by employing Lyubashevsky's signature twice, which is essentially a public key delegation from concatenated Schnorr signatures. In order to optimize the signature size, we describe an adaptation of our scheme in module lattices setting and compression technique. Moreover, our new LB-IBS can combine with Dilithium naturally so that it could carry the advantages of the latter such as side-channel attacks protection. We demonstrate that our construction is provably secure against existential forgery on adaptively chosen message and identity attacks in random oracle model. In comparison with the most efficient LB-IBS scheme (to the best of our knowledge), the signature size of our scheme is 52% smaller and runtime of our scheme is up to 32% faster.

Keywords: identity-based signature; lattices signature; module lattices; post-quantum cryptograhpy; trapdoors

1 Introduction

In the realm of public key encryption, digital signatures serve as essential building blocks. They authenticate the source of a digital document and its content's integrity, while also ensuring the source cannot later deny the document. The concept of identity-based signatures (IBS) was introduced by Shamir [1] as a viable alternative to traditional certificate-based cryptosystems. In an identity-based cryptosystem, the user's private key is generated by a private key generator (PKG), while the public key is derived from the user's public information, such as their identity and email address. The absence of authentication requirements for public keys has contributed to the widespread adoption of identity-based cryptosystems, with IBS being a prominent application within this field.

⋆ Corresponding Author.

However, the advancement of quantum computing poses a substantial risk to contemporary encryption techniques. Shor's research demonstrated that quantum computers enable the use of a probabilistic polynomial-time (PPT) algorithm for effectively breaching conventional number theory challenges, such as integer factorization and discrete logarithms. On the other hand, lattice hard problems, like learning with errors (LWE) and short integer solution (SIS), are presumed to withstand quantum computational attacks, capturing significant interest within the field. Due to their benefits, including average-case to worst-case reduction and straightforward implementation, lattice hard problems are a focus of developing post quantum cryptographic systems.

After Gentry, Peikert, and Vaikuntanathan(GPV) introduced their groundbreaking lattice trapdoors paper [2], lattice cryptography entered a period of vigorous development. [2] mentioned a method of constructing identity-based encryption schemes using preimage sampling functions, which also became the core of subsequent identity-based lattice cryptographic schemes. Rückert [3] was the first to propose two IBS schemes based on lattices, considering both the random oracle model and the standard model. Subsequently, numerous lattice-based IBS schemes have been developed [4,5].

On the other hand, the development of lattice signatures split into two directions. One is based on the preimage sampling function, continuing the hash-and-sign paradigm of GPV [2]. The other is based on the rejection sampling introduced by Lyubashevsky, following the Fiat-Shamir with aborts paradigm [6,7].

Additionally, Tian and Huang introduced an IBS scheme combined with GPV scheme and Lyubashevsky's signature [8]. Roughly, it uses preimage sampling function to extract a small coefficients matrix $\mathbf{S}$ satisfying $H(id) = \mathbf{AS}$ as the signing key for user id, where $\mathbf{A}$ is the master pubic key and its trapdoor $\mathbf{T}$ is the master secret key. The equation $H(id) = \mathbf{AS}$ form a SIS problem instance so that a user could use $\mathbf{A}$ and $\mathbf{S}$ to implement Lyubashevsky signature. And one can easily compute $H(id)$ to get the public key of user id. This method became the mainstream framework of later LB-IBS schemes.

In 2014, Ducas et al. proposed an efficient implementation of GPV scheme over NTRU lattices [9]. This approach brings many lattice-based cryptographic schemes into practical fields. With Tian's framework [8] and technique in [9], Xie et al. proposed an efficient IBS scheme over NTRU lattices [10]. Furthermore, Chen et al. put forward an innovative IBS scheme in 2021 that does not rely on Gaussian sampling [11]. After 2021, several IBS schemes in quantum random oracle model with tight security have been proposed [12–14]. Moreover, due to the inherent advantages of identity-based cryptosystems, several signature schemes with unique properties, such as identity-based ring signatures and multi-signatures, have been proposed [15,16].

Although Chen et al.'s scheme [11] is novel and shares a similar structure with our own, in our analysis, it possesses critical security vulnerabilities that one can easily expose user private keys. On the other hand, NTRU trapdoor based IBS enjoy high computational efficiency and small parameter size, but it has several own problems.

Currently, the most efficient GPV instantiation comes from Falcon signature scheme [17], which combined NTRU trapdoors [9] and fast Fourier sampler [18]. By its high efficiency and performance in terms of bandwidth, Falcon is now one of the NIST postquantum standardization. However, the algorithms of Falcon are rather complex so that it may be difficult to implement in constrained environments. For seeking simplification, several variant of Falcon such as Mitaka [19] has been proposed [20,21]. Furthermore, the security of NTRU is shown to be significantly reduced in the overstretched parameter regime [22, 23]. The fatigue of NTRU is estimated to be $q > 0.004 \cdot n^{2.484}$ for $n > 100$, which covers almost all the NTRU-based IBS and IBE. Therefore, it is somewhat desirable to avoid relying on the assumption that finding short vectors in the NTRU lattice is computationally hard, as long as the efficiency trade-off is not excessively high.

Moreover, two of the NIST postquantum standardization Crystal-Kyber [24] and Crystal-Dilithium [25] are both based on module lattices. Algebraic structure lattices could take well efficiency for lattice-based cryptographic schemes, but its geometric structure still face the risk of being attacked. Module lattices is seen as an "intermediate Road" between strand lattices and algebraic structure lattices since Module-LWE/SIS is proved not easier than Ring-LWE/SIS [26]. But once we used NTRU-based trapdoor to construct key extract algorithm, the unforgeability of IBS will inevitably based on Ring-LWE/SIS assumption [9]. A way to solve this problem is to find another preimage sampling function based on better hardness assumption. Gadget based GPV instantiation proposed by Micciancio and Peikert [27] offers significant advantages in terms of implementation and turns out to be extremely versatile for the constructions of advanced primitives [28]. However, gadget based schemes suffer from rather large preimage and public key sizes. In Crypto 2023, Yu et al. proposed a practical and compact gadget based framework [29], but how to instantate it in module lattices setting remain a question.

Another point deserving to be mentioned is that generating secret randomness from the discrete Gaussian distribution in signature scheme should be careful. Generating such samples in secure against side-channel attacks way is highly important and can easily lead to insecure implementations [19, 30]. Since famous lattice-based signature scheme BLISS [31] has been attacked, side-channel protection in lattice-based cryptography became unnegligible [32]. A good alterative to achieve this security requirement is only using uniform distribution like Dilithium [25], but discrete Gaussian sampling is absolutely necessary in preimage sampling function to provide zero-knowledgeness [2].

As aforementioned, current LB-IBS has its own limitation. The combination of lattice trapdoors technique and Fiat-Shamir with aborts signature to construct LB-IBS not only brings their advantages, but also inherits their disadvantages. Since the goal is to create a widely deployable scheme, it is highly desirable to have a more efficient scheme that does not rely on complex cryptographic primitives.

Our Contributions

Inspired by Lyubashevsky's and Galindo-Garcia's signature schemes [6,7,33,34], we propose a novel LB-IBS scheme without using lattice trapdoor technique. Our scheme is simple and easy to instantiate. This makes it attractive for application in resource-constrained environments where cost saving in computation, communication, and implementation code area are a premium.

Technical Overview: To introduce our delegation of public keys, we start from revisiting Lyubashesvky signature. The secret key is a matrix $\mathbf{S} \in \mathbb{Z}_q^{m\times k}$ with small coefficients, and the public key consists of the matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times k}$ and $\mathbf{T} = \mathbf{AS} \bmod q$. To sign a message μ, the signer first pick a random vector $\mathbf{y}$ from a public distribution and then calculates $\mathbf{z} = \mathbf{y} + \mathbf{Sc}$ where $\mathbf{c}$ is a hash value from $\mathbf{w} = \mathbf{Ay} \bmod q$ and message μ. After the approach called *reject sampling* [6], the distribution of $\mathbf{z}$ could independent of secret key $\mathbf{S}$ in overwhelming probability. Finally vectors $(\mathbf{z}, \mathbf{c})$ will be output as the signature on the message, or like Schnorr Signature, output $(\mathbf{z}, \mathbf{w})$ as the signature. If we do as the latter, a signature will be verified by checking whether $\mathbf{z}$ is small and $\mathbf{Az} - \mathbf{w}$ is equal to $\mathbf{T} \cdot H(\mathbf{w}, \mu)$.

Notice that $\mathbf{w} + \mathbf{T} \cdot H(\mathbf{w}, \mu)$ is pseudo-random and non priori under random oracle model. And if we enlarge modulus q appropriately, $(\mathbf{A}, \mathbf{w} + \mathbf{T} \cdot H(\mathbf{w}, \mu))$ and $\mathbf{z}$ could form a new SIS problem instance. By "Signing" a user's *id*, we could extract a signing key for the user in above way.

If we construct an IBS system in this way, the extract algorithm and sign algorithm are both essentially a Fait-Shamir with aborts paradigm. But the final signature for a message would have a relatively large coefficients about $\tilde{O}(n^2)$ cause we employing reject sampling twice. Profit by the pure structure of Lyubashesvkys signature, the IBS could combine the idea in Dilithium or Dilithium-G naturally.

First if the key is generated based on LWE problem, by throwing away the least significant bits of $\mathbf{Ay}$ to get a hash value $\mathbf{c}$, the signature $\mathbf{z}$ could compress a half size [35]. Another optimization way to replace the low-order bits of user's "public key" $\mathbf{w}$ as a one bit carries hints [36].

All computational acceleration techniques presented in Dilithium can be used with our scheme. Overall, we obtain signature of sizes smaller than NTRU-based IBS [10] while our scheme does not rely on complex cryptographic primitives and is easy to implement.

We believe that our schemes can still be improved since we also notice that the most efficient Fait-Shamir paradigm signatures come from bimodal Gaussians setting such as HAETAE [37] and G+G [38]. An interesting open problem would be study the public keys delegation in bimodal Gaussians signature.

2 Preliminaries

Throughout the paper, we will assume that q is a prime number and elements in $\mathbb{Z}_q$ are represented by integers in the range $(-\frac{q}{2}, \frac{q}{2}]$. We will represent vectors

by bold-face letters, and matrices by bold-face capital letters. We will assume that all vectors are column vectors, and the l_i norm of a vector $\mathbf{v}$ is denoted by $||\mathbf{v}||_i$. And we define $D_\eta^n = \{\mathbf{v} : \mathbf{v} \in \mathbb{Z}_q^n, ||\mathbf{v}||_\infty \leq \eta\}$.

2.1 Hardness Assumptions

The security of our scheme relies on two lattice hardness problems, namely the decisional Learning with Errors problem and the Short Integer Solution problem.

Definition 1 (LWE). *Let $n, m \geq 0$ and $q \geq 2$.Let χ be a distribution over $\mathbb{Z}$. The $\mathsf{LWE}_{n,m,q,\chi}$ problem refers to distinguishing $(\mathbf{A}, \mathbf{As} + \mathbf{e})$ from $(\mathbf{A}, \mathbf{u})$ where $\mathbf{A} \leftarrow U(\mathbb{Z}_q^{n \times m})$, $\mathbf{u} \leftarrow U(\mathbb{Z}_q^n)$ and $[\mathbf{s}^\mathsf{T}|\mathbf{e}^\mathsf{T}]^\mathsf{T} \leftarrow \chi^{m+n}$.*

Definition 2 (SIS). *Let $n, m, d \geq 0$ and $q \geq 2$ be a modules. The $\mathsf{SIS}_{n,m,q,d}$ problem refers to finding a $\mathbf{s} \in \mathbb{Z}^m$ such that $\mathbf{As} \equiv \mathbf{0}$ mod q and $0 < ||\mathbf{s}||_\infty \leq d$ for a given $\mathbf{A} \leftarrow U(\mathbb{Z}_q^{n \times m})$.*

2.2 Identity-Based Signature

An identity-based signature scheme is a tuple of PPT algorithms (Setup, $\mathsf{Extract}$, Sign, Verify) described as follows:

Setup. The Setup algorithm that takes as input a security parameter λ, and outputs PKGs secret key msk and public parameters mpk. We denote it by $(mpk, msk) \leftarrow \mathsf{Setup}(1^\lambda)$.

$\mathsf{Extract}$. The extraction algorithm that takes as input the public parameters params, the PKGs secret key msk and an identity id, and outputs a signing key sk_{id} associated with id. We denote it by $sk_{id} \leftarrow \mathsf{Extract}(mpk, msk, id)$.

Sign. The signing algorithm that takes as input the public parameters params, a message μ and a signing key sk_{ID} of the user with identity id, and outputs a signature σ. We denote it by $\sigma \leftarrow \mathsf{Sign}(mpk, sk_{id}, id, \mu)$.

Verify. The verification algorithm that takes as input the public parameters params, a signature σ, a message μ and an identity id, and outputs 1 if the signature sig is valid and 0 otherwise. We denote it by $b \leftarrow \mathsf{Verify}(mpk, \sigma, \mu, id)$.

Generally, the correctness of an identity-based signature scheme requires if a signature is correctly generated, it could always be verified. That is, for any $m \in \{0,1\}^*$, we have that

$$\Pr[\mathsf{Verify}(mpk, \mathsf{Sign}(\mathsf{Extract}(mpk, msk, id), \mu), \mu, id) = 1] = 1.$$

EUF-ID-CMA. Let (msk, mpk) be the output of $\mathsf{Setup}(1^\lambda)$. For correctness, we require that for any message μ and any identity id if $sig = \mathsf{Sign}(mpk, \mu, sk_{id})$ where $sk_{id} = \mathsf{Extract}(mpk, SK, id)$, then we have $\mathsf{Verify}(mpk, sig, \mu, id) = 1$ with overwhelming probability. The security model of IBS schemes is described via the following game which is played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

- **Setup**. The challenger $\mathcal{C}$ runs the algorithm $\mathsf{Setup}(1^\lambda)$ to generate public parameters params, and sends params to the adversary $\mathcal{A}$.
- **Query**. The adversary $\mathcal{A}$ can issue the following types of queries adaptively.
  Extract-Query. To get the signing key of the user with identity id, the adversary $\mathcal{A}$ issues such a query on the identity id. In response, the challenger $\mathcal{C}$ runs the algorithm $\mathsf{Extract}(mpk, msk, id)$ and returns a signing key sk_{id}.
  Sign-Query. When the adversary $\mathcal{A}$ issues such a query on message μ and identity id, the challenger $\mathcal{C}$ returns a signature sig as a response.
- **Forgery**. The adversary $\mathcal{A}$ outputs a signature sig of message μ with respect to identity id. $\mathcal{A}$ wins the game if: (i) $\mathsf{Verify}(mpk, sig, \mu, id) = 1$, (ii) the identity id has never been submitted to **Extract-Query**, and (iii) (μ, id) has never been submitted to **Sign-Query**.

Definition 3. *An identity-based signature scheme is said to be existential unforgeable against adaptive chosen message and identity attacks(EUF-ID-CMA) if for any PPT adversary $\mathcal{A}$ the advantage $\mathbf{Adv}_{\mathcal{A}}(n)$ in the above game is negligible.*

3 Our LB-IBS without Trapdoors

Algorithm 1 to **Algorithm 4** consist of the signature scheme based on on the average-case hardness of the SIS problem.

We first choose two hash functions: $H_1 : \{0,1\}^* \to \{\mathbf{W} : \mathbf{W} \in \{-1,0,1\}^{k\times k}, ||\mathbf{W}||_1 = \kappa\}$ and $H_2 : \{0,1\}^* \to \{\mathbf{w} : \mathbf{w} \in \{-1,0,1\}^k, ||\mathbf{w}||_1 = \kappa\}$. The master secret key is an $m \times k$ matrix $\mathbf{S}$ of random integers of absolute value at most η, and the master public key consists of a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and another matrix $\mathbf{T} \in \mathbb{Z}_q^{n\times k}$ which is equal to $\mathbf{AS}$.

Algorithm 1: Setup

Input: Security parameters

1 $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$
2 $\mathbf{S} \leftarrow D_\eta^{m\times k}$
3 $\mathbf{T} = \mathbf{AS} \bmod q$
4 **Return** $(mpk, msk) = ((\mathbf{A}, \mathbf{T}), \mathbf{S})$

To extract a signing key for user ID , KGC first picks an $m \times k$ masking matrix $\mathbf{Y}_{id}$ from the distribution $D_{\gamma_1}^{m\times k}$. The parameter γ_1 is set strategically it is large enough that the eventual signature does not reveal the secret key (i.e. the signing algorithm is zero-knowledge), yet small enough so that the signing key is not easily forged. Then KGC computes $\mathbf{C}_{id} = H_1(\mathbf{W}_{id} = \mathbf{AY}_{id}, id)$, and finally computes $\mathbf{Z}_{id} = \mathbf{SC}_{id} + \mathbf{Y}_{id}$ (no reduction modulo q). Actually, H_1 could be implemented based on H_2 by letting each vector i in the matrix $\mathbf{W}_{id}$ call $\mathbf{C}_{id}[i] = H_2(\mathbf{W}_{id}[i], id)$, $i \in [0, k-1]$. The potential signing key sk_{id} which KGC outputs is $(\mathbf{Z}_{id}, \mathbf{W}_{id})$, but KCG only outputs it when each coefficient of $\mathbf{Z}_{id}$ not greater than $\gamma_1 - \beta_1$. It has a repetition expectation of M.

Algorithm 2: Extract

Input: mpk, msk, id

$\mathbf{Y}_{id} \leftarrow D_{\gamma_1}^{m \times k}$

$\mathbf{W}_{id} = \mathbf{A}\mathbf{Y}_{id} \bmod q$

$\mathbf{C}_{id} = H_1(\mathbf{W}_{id}, id)$

$\mathbf{Z}_{id} = \mathbf{S}\mathbf{C}_{id} + \mathbf{Y}_{id}$

If $||\mathbf{Z}_{id}||_\infty > \gamma_1 - \beta_1$ **then Restart**

Return $sk_{id} = (\mathbf{S}_{id} = \mathbf{Z}_{id}, \mathbf{W}_{id})$

The correctness of sk_{id} can be verified by checking if $||\mathbf{Z}_{id}||_\infty < \gamma_1 - \beta_1$ and $\mathbf{A}\mathbf{Z}_{id} = \mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id)$. Note that $\mathbf{W}_{id}$ is actually public information even though it is part of the signing key.

The Sign algorithm is similar to the Extract algorithm. While user ID want to sign a message μ, the signer just picks an m-dimensional vector y from uniform distribution $D_{\gamma_2}^m$ (not matrix now), then computes $\mathbf{c} = H_2(id, \mathbf{w}', \mu)$ and $\mathbf{z} = \mathbf{Z}_{id}\mathbf{c} + \mathbf{y}$. Finally signer output signature $(\mathbf{z}, \mathbf{c}, \mathbf{W}_{id})$ when each coefficient of $\mathbf{z}$ not greater than $\gamma_2 - \beta_2$.

Algorithm 3: Sign

Input: mpk, sk_{id}, μ, id

$\mathbf{y} \leftarrow D_{\gamma_2}^m$

$\mathbf{w} = \mathbf{A}\mathbf{y} \bmod q$

$\mathbf{c} = H_2(id, \mathbf{w}, \mu)$

$\mathbf{z} = \mathbf{S}_{id}\mathbf{c} + \mathbf{y}$

If $||\mathbf{z}||_\infty \geq \gamma_2 - \beta_2$ **then Restart**

Return $sig = (\mathbf{z}, \mathbf{c}, \mathbf{W}_{id})$

By the rejection sampling theorem [6], the distribution of the signature $(\mathbf{z}, \mathbf{c})$, signing key $(\mathbf{S}_{id}, \mathbf{W}_{id})$ and master secret key $\mathbf{S}$ are independent of each others, so that we can use a forger who successfully breaks the IBS to solve the SIS problem.

Algorithm 4: Verify

Input: mpk, μ, sig, id

$\mathbf{w}' = \mathbf{A}\mathbf{z} - (\mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id))\mathbf{c}$

Return $||\mathbf{z}||_\infty < \gamma_2 - \beta_2$ and $\mathbf{c} = H_2(id, \mathbf{w}', \mu)$

3.1 Correctness Analysis

In this section, we prove the correctness of the signature scheme.

According to the construction of the IBS scheme, we know that

$$\mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{S}_{id}\mathbf{c} + \mathbf{A}\mathbf{y} = \mathbf{A}\mathbf{S}_{id}\mathbf{c} + \mathbf{w}.$$

Since $\mathbf{A}\mathbf{S}_{id} = \mathbf{A}\mathbf{Z}_{id} = \mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id)$, we have

$$\mathbf{A}\mathbf{z} = (\mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id))\mathbf{c} + \mathbf{w}.$$

Then we can get $\mathbf{w} = \mathbf{A}\mathbf{z} - (\mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id))\mathbf{c}$.

3.2 Security Proof

Now we show two algorithm **Extract Oracle** and **Hybrid Sign**:

Algorithm 5: Extract Oracle

Input: id

$\mathbf{C}_{id} \leftarrow \{\mathbf{W} : \mathbf{W} \in \{-1, 0, 1\}^{k\times k}, ||\mathbf{W}||_1 = \kappa\}$

$\mathbf{Z}_{id} \leftarrow D^{n\times k}_{\gamma_1-\beta_1}$

$\mathbf{W}_{id} = \mathbf{A}\mathbf{Z}_{id} - \mathbf{T}\mathbf{C}_{id}$

Restart with probability $1 - 1/M$

Program $H_1(\mathbf{W}_{id}, id) = \mathbf{C}_{id}$

Return $sk_{id} = (\mathbf{S}_{id} = \mathbf{Z}_{id}, \mathbf{W}_{id})$

Algorithm 6: Sign Oracle

Input: id, μ

$(\mathbf{Z}_{id}, \mathbf{W}_{id}) = \mathsf{Extract\ Oracle}(id)$

$\mathbf{c} \leftarrow \{\mathbf{w} : \mathbf{w} \in \{-1, 0, 1\}^k, ||\mathbf{w}||_1 = \kappa\}$

$\mathbf{z} \leftarrow D^n_{\gamma_2-\beta_2}$

Restart with probability $1 - 1/M$

Program $H_2(id, \mathbf{A}\mathbf{z} - (\mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id))\mathbf{c}, \mu) = \mathbf{c}$

Return $sig = (\mathbf{z}, \mathbf{c}, \mathbf{W}_{id})$

By the Lemma 5.3 in [7], the advantage of distinguishing the actual Extract algorithm and Sign algorithm from **Extract Oracle** and **Hybrid Sign** is negligible.

Theorem 1. *If there is a polynomial-time forger , which makes at most s queries to the Extract and Sign oracle, and h queries to the random oracle H_1 and H_2, breaks the IBS with probability δ, then there is a polynomial-time algorithm which can solve the $\mathsf{SIS}_{n,m,q,d}$ problem for $d = 2\gamma_2 + 2\kappa\beta_1 = \tilde{O}(dn^2)$ with at least probability $\approx \frac{\delta^2}{(s+h)^2}$.*

Proof. Let $D_H = \{\mathbf{c} : \mathbf{c} \in \{-1, 0, 1\}^k, ||\mathbf{c}||_1 = \kappa\}$ denote the range of the random oracle. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ to be challenged, we randomly pick $\mathbf{S} \in \mathbb{Z}_\eta^{m\times k}$ and then compute $\mathbf{T} = \mathbf{A}\mathbf{S}$. Let $t = h + s$ be the bound on the number of times the the random oracle is called or programmed during $\mathcal{F}$s attack. We then pick random coins ϕ for the forger and ψ for the signer, and we also pick $(\mathbf{r}_1, ..., \mathbf{r}_t) \leftarrow D_H$, which will correspond to the responses of the random oracle. And we construct a polynomial-time algorithm $\mathcal{A}$ run as follow:

- **Setup**. Given $(\mathbf{A}, \mathbf{T}, \phi, \psi, \mathbf{r}_1, ..., \mathbf{r}_t)$ as input, $\mathcal{A}$ sends the public parameters and random coins $(\mathbf{A}, \mathbf{T}, \phi)$ to $\mathcal{F}$ for initialization.
- **Extract hash query**. The forger $\mathcal{F}$ can make queries to the random oracle H_1, in which case $\mathcal{A}$ will pick the first unused $\mathbf{r}_i$ in the list $(\mathbf{r}_1, ..., \mathbf{r}_t)$. Then A will first extend $\mathbf{r}_i$ into $\mathbf{R}_i \in D_H^k$ and take it as reply. Note that $\mathcal{A}$ will keep a table of all the queries to H_1, so in case the same query is made twice, it will reply with the previous answer.

- **Sign hash query**. The forger $\mathcal{F}$ can make queries to the random oracle H_2, in which case $\mathcal{A}$ will pick the first unused $\mathbf{r}_i$ in the list $(\mathbf{r}_1, ..., \mathbf{r}_t)$ and take it as reply. Also $\mathcal{A}$ will keep a table of all the queries to H_2, so in case the same query is made twice, it will reply with the previous answer.
- **Extract query**. When $\mathcal{F}$ wants some sign keys extracted, $\mathcal{A}$ runs the **Extract Oracle** algorithm using the signers random coins ψ to produce a sign key. During extracting, the random oracle H_1 will be programmed. Also, $\mathcal{A}$ will keep a table of all queries to **Extract Oracle**, it will reply with the previous answer if the same query is made twice.
- **Sign query**. When $\mathcal{F}$ wants some message signed by a given id, $\mathcal{A}$ runs the **Sign Oracle** algorithm using the signers random coins ψ to produce a signature. During signing, the random oracle H_2 will be programmed.
- **Forgery**. Once $\mathcal{F}$ finishes running and outputs a forgery with probability δ, $\mathcal{A}$ simply outputs $\mathcal{F}$s output.

$\mathcal{F}$ will outputs a valid but non-trivial forgery $(\mathbf{z}, \mathbf{c}, \mathbf{W}_{id})$ of identity id on message μ with probability δ such that $||\mathbf{z}||_\infty < \gamma_2 - \beta_2$ and

$$\mathbf{c} = H_2(id, \mathbf{Az} - (\mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id))\mathbf{c}, \mu) \tag{3.1}$$

Notice that if the random oracle H_2 was not queried or programmed on some input $\mathbf{w} = \mathbf{Az} - (\mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id))\mathbf{c}$, then $\mathcal{F}$ only has $1/|D_H|$ chance of producing a $\mathbf{c}$ such that $c = H_2(id, \mathbf{w}, \mu)$. So with probability $\delta - 1/|D_H|$, $\mathcal{F}$ will succeeds in a forgery and $\mathbf{c}$ is one of the $\mathbf{r}_i$. Now let's divide the forged signatures $(\mathbf{z}, \mathbf{c}, \mathbf{W}_{id})$ into two cases $\mathbf{E}$ and $\dot{\mathbf{E}}$ as follow:

- $\mathbf{E}$: $\mathcal{F}$ makes at least one sign query on id and $\mathbf{W}_{id}$ was returned by $\mathcal{A}$ as part of the output to a signature query on id.
- $\dot{\mathbf{E}}$: Either $\mathcal{F}$ does not any make sign queries on id or $\mathbf{W}_{id}$ was never returned by $\mathcal{A}$ as part of the output to a signature query on id.

We start by dealing with the simpler event $\mathbf{E}$ and set $\mathbf{T}' = \mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id)$. In light of the discussion above, $\mathbf{c}$ is one of the $\mathbf{r}_i$. One possibility is that $\mathbf{r}_i = \mathbf{c}$ was programmed during signing. It means that we already know another $(\mathbf{z}', \mathbf{c}, \mathbf{W}_{id})$ such that $H_2(id, \mathbf{Az}' - \mathbf{T}'\mathbf{c}, \mu') = \mathbf{c} = H_2(id, \mathbf{Az} - \mathbf{T}'\mathbf{c}, \mu)$. This implies that $\mathbf{Az} - \mathbf{T}'\mathbf{c} = \mathbf{Az}' - \mathbf{T}'\mathbf{c}$ and $\mu = \mu'$, or else we found a second pre-image for $\mathbf{c}$. Thus $\mathbf{A}(\mathbf{z} - \mathbf{z}') = 0$ with $\mathbf{z} \neq \mathbf{z}'$, and since $||\mathbf{z}||_\infty, ||\mathbf{z}'||_\infty < \gamma_2$, we have $||\mathbf{z} - \mathbf{z}'||_\infty < 2\gamma_2$.

Another case is that $\mathbf{c} = \mathbf{r}_i$ was a response to H_2 directly query made by $\mathcal{F}$. In this case, we pick fresh random elements $(\mathbf{r}'_i, ..., \mathbf{r}'_t) \leftarrow D_H$ ans take $(\mathbf{A}, \mathbf{T}, \phi, \psi, \mathbf{r}_1, ..., \mathbf{r}_{i-1}, \mathbf{r}'_i, ..., \mathbf{r}'_t)$ as input to run $\mathcal{A}$ again. By the General Forking Lemma of Bellare and Neven [39], we can get a forgery signature $(\mathbf{z}', \mathbf{c}', W_{id})$ on μ and $\mathbf{Az} - \mathbf{T}'\mathbf{c} = \mathbf{Az}' - \mathbf{T}'\mathbf{c}'$ which $\mathbf{c}' \neq \mathbf{c}$ with probability at least

$$(\delta - \frac{1}{|D_H|})(\frac{\delta - 1/|D_H|}{t} - \frac{1}{|D_H|}).$$

Cause $\mathbf{T}' = \mathbf{W}_{id} + \mathbf{T}H_1(\mathbf{W}_{id}, id) = \mathbf{AZ}_{id}$, rearrang terms in $\mathbf{Az} - \mathbf{T}'\mathbf{c} = \mathbf{Az}' - \mathbf{T}'\mathbf{c}'$ then we can get $\mathbf{A}(\mathbf{z}' - \mathbf{z} + \mathbf{Z}_{id}\mathbf{c}' - \mathbf{Z}_{id}\mathbf{c}) = 0$. Since $||\mathbf{z}||_\infty, ||\mathbf{z}'||_\infty < \gamma_2$ and $||\mathbf{Z}_{id}\mathbf{c}||_\infty, ||\mathbf{Z}_{id}\mathbf{c}'||_\infty < \beta_2$ we know that $||\mathbf{z}' - \mathbf{z} + \mathbf{Z}_{id}\mathbf{c}' - \mathbf{Z}_{id}\mathbf{c}|| < 2\gamma_2 + 2\beta_2$.

Now we turn to event $\mathbf{E}'$. In order to come up with the forgery signature $(\mathbf{z}, \mathbf{c}, \mathbf{W}_{id})$ with a non-negligible probability, the adversary, at some point during its execution, has to make the two random oracle calls: $H_1(\mathbf{W}_{id}, id)$ and $H_2(id, \mathbf{W}_{id}, \mu)$. Let j be such that $H_2(id, \mathbf{W}_{id}, \mu)$ was the extendsion of $\mathbf{r}_j$ and $\mathbf{r}_i$ was the return of $H_1(\mathbf{W}_{id}, id)$. Setting $H_2(id, \mathbf{W}_{id}, \mu) = \mathbf{R}$ and we can obtain $\mathbf{c} = \mathbf{r}_i = H_2(id, \mathbf{Az} - (\mathbf{W}_{id} + \mathbf{TR})\mathbf{r}_i, \mu)$. Next we pick fresh random elements $(\mathbf{r}'_j, ..., \mathbf{r}'_t) \leftarrow D_H$ but keep $\mathbf{r}'_i = \mathbf{r}_i$ if $j < i$. Then take $(\mathbf{A}, \mathbf{T}, \phi, \psi, \mathbf{r}_1, ..., \mathbf{r}_{j-1}, \mathbf{r}'_j, ..., \mathbf{r}'_t)$ as input to run $\mathcal{A}$ again. Using the Multiple-Forking Lemma of Boldyreva, Palacio and Warinschi [40], which is a further generalization of the General Forking Lemma to multiple random oracles and signatures, with probability

$$(\delta - \frac{1}{|D_H|})(\frac{\delta - 1/|D_H|}{t^2} - \frac{1}{|D_H|}),$$

we can get another signature $(\mathbf{z}', \mathbf{c}, \mathbf{W}_{id})$ and $\mathbf{Az} - (\mathbf{W}_{id} + \mathbf{TR})\mathbf{c} = \mathbf{Az}' - (\mathbf{W}_{id} + \mathbf{TR}')\mathbf{c}$ which $\mathbf{R} \neq \mathbf{R}'$. With equality $\mathbf{T} = \mathbf{AS}$, we obtain $\mathbf{A}(\mathbf{z}' - \mathbf{z} + \mathbf{SRc} - \mathbf{SR}'\mathbf{c}) = 0$. Since $||\mathbf{SRc}||_\infty < \kappa\beta_1$, we know that

$$||\mathbf{z}' - \mathbf{z} + \mathbf{SRc} - \mathbf{SR}'\mathbf{c}||_\infty < 2\gamma_2 + 2\kappa\beta_1.$$

□

As Lyubashevsky's signature scheme, our scheme underlies a abortable Σ-protocol with Fiat-Shamir heuristic. If we considering quantum adversaries, it should be granted quantum access to the random oracle (QROM) as it can query the hash function in quantum superposition. In recent years, many works have shown that existing lattice signatures based on Fiat-Shamir with aborts are also secure in QROM setting without or with little modification [41–43].

4 Implementation of Our LB-IBS

Through the inspiration given by Dilithium [25], we modify the scheme to one based on Module-LWE problem and use the high/low bits signature compression technology introduced in [35] and signing key compression technology by hints [36].

The matrixs and vectors in the scheme are all composed of polynomials in $\mathbb{Z}_q[X]/(X^{256} + 1)$ by vector. As in most lattice-based schemes that are based on operations over polynomial rings, the multiplication operation has an efficient implementation via the Number Theoretic Transform (NTT), which is just a version of FFT. Cause we don't use trapdoor technology, so we can have master public key $\mathbf{A}$ generated from some seed ρ using SHAKE-128.

4.1 Preliminaries

Ring Operations. We let R and R_q respectively denote the rings $\mathbb{Z}[X]/(X^n+1)$ and $\mathbb{Z}_q[X]/(X^n+1)$, for q an integer. Throughout this section, the value of n will always be 256 and q will be a prime number. Regular font letters denote elements in R or R_q (which includes elements in $\mathbb{Z}$ and $\mathbb{Z}_q$) and bold lower-case letters represent column vectors with coefficients R or R_q. By default, all vectors will be column vectors. Bold upper-case letters are matrices. The boolean operator "$a == b$" evaluates to 1 if $|a-b|$ is equal to zero, and to 0 otherwise.

For an even (resp. odd) positive integer α, we define $r' = r \bmod^{\pm} \alpha$ to be the unique element r' in the range $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (resp. $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha}{2}$) such that $r' \equiv r \bmod \alpha$.

For $w = w_0 + w_1 X + ... + w_{n-1} X^{n-1} \in R$ and $\mathbf{t} = (t_1, ..., t_k) \in R_q^m$, we define the l_∞ of them:

$$||w||_\infty = max\ |w_i|,\ ||\mathbf{t}||_\infty = max||t_i||.$$

And we define $S_\eta = \{w : w \in R_q, ||w||_\infty \leq \eta\}$.

Security of our modified LB-IBS scheme would be based on the following hardness assumptions.

Definition 4 (MLWE). *Let $n, m \geq 0$ and $q \geq 2$.Let χ be a distribution over R. The $\mathsf{MLWE}_{n,m,q,\chi}$ problem refers to distinguishing $(\mathbf{A}, \mathbf{As}+\mathbf{e})$ from $(\mathbf{A}, \mathbf{u})$ where $\mathbf{A} \leftarrow U(R_q^{n\times m})$, $\mathbf{u} \leftarrow U(R_q^n)$ and $[\mathbf{s}^\mathsf{T}|\mathbf{e}^\mathsf{T}]^\mathsf{T} \leftarrow \chi^{m+n}$.*

Definition 5 (MSIS). *Let $n, m, d \geq 0$ and $q \geq 2$ be a modules. The $\mathsf{MSIS}_{n,m,q,d}$ problem refers to finding a $\mathbf{s} \in R^m$ such that $\mathbf{As} \equiv \mathbf{0}$ mod q and $0 < ||\mathbf{s}||_\infty \leq d$ for a given $\mathbf{A} \leftarrow U(R_q^{n\times m})$.*

High/Low Order Bits and Hints. To reduce the size of the public key, we will need some simple algorithms that extract higher-order and lower-order bits of elements in $\mathbb{Z}_q$. The goal is that when given an arbitrary element $r \in \mathbb{Z}_q$ and another small element $z \in \mathbb{Z}_q$, we would like to be able to recover the higher order bits of $r + z$ without needing to store z. We therefore define algorithms that take r, z and produce a 1-bit hint h that allows one to compute the higher order bits of $r + z$ just using r and h. This hint is essentially the carry caused by z in the addition.

$\mathsf{Power2Round_q}$ and $\mathsf{Decompose_q}$ are two different ways in which we will break up elements in Zq into their highorder bits and low-order bits. The former is the straightforward bit-wise way to break up an element $r = r_1 \cdot 2^d + r_0$ where $r_0 = r \bmod^{\pm} 2^d$ and $r_1 = (r - r_0)/2^d$. The latter one is similar to above but need to round r_1 to 0 when $r_1 \cdot \alpha = q - 1$ so that it can be adapted to use 1-bit hint.

We define the $\mathsf{MakeHint_q}$ and $\mathsf{UseHint_q}$ routines that produce a hint and, respectively, use the hint to recover the high-order bits of the sum. And $\mathsf{HighBits_q}$ and $\mathsf{LowBits_q}$ routines are simply extract r_1 and r_0 of $\mathsf{Decompose_q}$.

Algorithm 7: $\mathsf{Power2Round_q}$

Input: r, d

$r = r \bmod q$

$r_0 = r \bmod^{\pm} 2^d$

return $(r - r_0)/2^d$

Algorithm 8: $\mathsf{Decompose_q}$

Input: r, α

$r = r \bmod q$

$r_0 = r \bmod^{\pm} \alpha$

if $r - r_0 = q - 1$ **then**

  $r_1 = 0; r_0 = r_0 - 1$

else

  $r_1 = (r - r_0)/\alpha$

end

return (r_1, r_0)

Algorithm 9: $\mathsf{HighBits_q}$

Input: r, α

$(r_1, r_0) = \mathsf{Decompose_q}(r, \alpha)$

return r_1

Algorithm 10: $\mathsf{LowBits_q}$

Input: r, α

$(r_1, r_0) = \mathsf{Decompose_q}(r, \alpha)$

return r_0

Algorithm 11: $\mathsf{MakeHint_q}$

Input: z, r, α

$r_1 = \mathsf{HighBits_q}(r, \alpha)$

$v_1 = \mathsf{HighBits_q}(r + z, \alpha)$

return $v_1 \neq r_1$

Algorithm 12: $\mathsf{UseHint_q}$

Input: h, r, α

$m = (q - 1)/\alpha$

$(r_1, r_0) = \mathsf{Decompose_q}(r, \alpha)$

if $h == 1$ *and* $r_0 > 0$ **then**

  return $(r_1 + 1)$ *mod* m

end

if $h == 1$ *and* $r_0 \leq 0$ **then**

  return $(r_1 - 1)$ *mod* m

end

return r_1

Let $\mathbf{z}$ and $\mathbf{r}$ be vectors of elements in R_q and $q > 2\alpha, ||\mathbf{z}||_\infty < \alpha/2$, we have:

Lemma 1. $\mathsf{UseHint_q}(\mathsf{MakeHint_q}(\mathbf{z}, \mathbf{r}, \alpha), \mathbf{r}, \alpha) = \mathsf{HighBits_q}(\mathbf{r} + \mathbf{z}, \alpha)$

Lemma 2. *For any* $\mathbf{h}, \mathbf{h}'$*, if* $\mathsf{UseHint_q}(\mathbf{h}, \mathbf{r}, \alpha) = \mathsf{UseHint_q}(\mathbf{h}', \mathbf{r}, \alpha)$*, then* $\mathbf{h} = \mathbf{h}'$

Lemma 3. *If* $||\mathbf{s}||_\infty < \beta$ *and* $||\mathsf{LowBits_q}(\mathbf{r}, \alpha)||_\infty < \alpha/2 - \beta$*, then* $\mathsf{HighBits_q}(\mathbf{r}, \alpha) = \mathsf{HighBits_q}(\mathbf{r} + \mathbf{s}, \alpha)$

4.2 Dilithium-like Scheme

Algorithm 13 to **Algorithm 16** consist of the signature scheme based on modules lattices.

The Setup algorithm generates a $k \times l$ matrix $\mathbf{A}$ each of whose entries is a polynomial in the ring $R_q = \mathbb{Z}_q[X]/(X^{n+1})$. Afterwards, the algorithm samples random secret key vectors s_1 and s_2. Each coefficient of these vectors is an element of R_q with small coefficients of size at most η. Finally, the second part of the public key is computed as $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$. All algebraic operations in this scheme are assumed to be over the polynomial ring R_q.

Algorithm 13: Setup

Input: Security parameters

$\mathbf{A} \leftarrow R_q^{k \times l}$

$\mathbf{s}_1 \leftarrow S_\eta^l, \mathbf{s}_2 \leftarrow S_\eta^k$

$\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$

Return $(mpk, msk) = ((\mathbf{A}, \mathbf{t}), (\mathbf{s}_1, \mathbf{s}_2))$

The Extract algorithm generates two masking vectors of polynomials y_{id_1} and y_{id_2} with coefficients less than γ_1. KGC then computes $\mathbf{w}_{id} = \mathbf{A}\mathbf{y}_{id_1} + \mathbf{y}_{id_2}$ and drop d lower bits of its which as $\mathbf{w}_{id_1}$. The challenge c is then created as the hash of the id and $\mathbf{w}_{id_1}$. The output c is a polynomial in R_q with exactly κ ± 1's and the rest 0's. The reason for this distribution is that c could has small norm and comes from a domain of entropy $\log_2 \binom{256}{\kappa} + \kappa$, which we would like to be between 128 to 256 bits. The potential signing key is then computed as $\mathbf{z}_{id_1} = \mathbf{s}_1 c_{id} + \mathbf{y}_{id_1}, \mathbf{z}_{id_2} = \mathbf{s}_2 c_{id} + \mathbf{y}_{id_2}$and $\mathbf{w}_{id}$. If it were directly output at this point, then the scheme would be insecure due to the fact that the secret key would be leaked. To avoid the dependency of $\mathbf{z}_{id_i}$ on the secret key, we use rejection sampling. The parameter β_1 is set to be the maximum possible coefficient of $c\mathbf{s}_i$. If any coefficient of $\mathbf{z}_{id_i}$ is larger than $\gamma_1 - \beta_1$, then we reject and restart the signing procedure. The while loop in the signing procedure keeps being repeated until the preceding two conditions are satisfied. The parameters are set such that the expected number of repetitions is not too high. User could compute $\mathbf{w}_{id_1} = \mathsf{Power2Round}_{\mathsf{q}}(\mathbf{w}_{\mathsf{id}}, \mathsf{d})$ and $\mathbf{A}\mathbf{z}_{id_1} + \mathbf{z}_{id_2} = \mathbf{w}_{id} + \mathbf{t}H(\mathbf{w}_{id_1}, id)$ to check the correctness of the signing key. Note that $\mathbf{w}_{id_1}$ is actually public information even though it is part of the signing key.

Algorithm 14: Extract

Input: mpk, msk, id

$\mathbf{y}_{id_1} \leftarrow S_{\gamma_1}^l, \mathbf{y}_{id_2} \leftarrow S_{\gamma_1}^k$

$\mathbf{w}_{id} = \mathbf{A}\mathbf{y}_{id_1} + \mathbf{y}_{id_2}$

$\mathbf{w}_{id_1} = \mathsf{Power2Round}_{\mathsf{q}}(\mathbf{w}_{id}, d)$

$c_{id} = H(\mathbf{w}_{id_1}, id)$

$\mathbf{z}_{id_1} = \mathbf{s}_1\mathbf{c}_{id} + \mathbf{y}_{id_1}, \mathbf{z}_{id_2} = \mathbf{s}_2\mathbf{c}_{id} + \mathbf{y}_{id_2}$

If $||\mathbf{z}_{id_1}, \mathbf{z}_{id_2}||_\infty > \gamma_1 - \beta_1$, then goto **1**

Return $sk_{id} = (\mathbf{s}_{id_1} = \mathbf{z}_{id_1}, \mathbf{s}_{id_2} = \mathbf{z}_{id_2}, \mathbf{w}_{id})$

The Sign algorithm is similar to above. The signer only need generates one masking vector of polynomials $\mathbf{y}$ with coefficients less than γ_2. The signer then computes $\mathbf{A}\mathbf{y}$ and sets $\mathbf{w}'$ to be the high-order bits of the coefficients in this vector. In particular, every coefficient w in $\mathbf{A}\mathbf{y}$ can be written in a canonical way as $w = w' \cdot 2\gamma_3 + x$ where $|x| \leq \gamma_3$. And $\mathbf{w}'$ is the vector comprising all the w'. The potential signature is then computed as $\mathbf{z} = \mathbf{z}_{id_1} c + \mathbf{y}$. By the rejection sampling, If any coefficient of $\mathbf{z}$ is larger than $\gamma_2 - \beta_2$, then we reject and restart the signing procedure. Also, if any coefficient of the low-order bits of $\mathbf{A}\mathbf{z} - (\mathbf{w}_{id} + \mathbf{t}H(\mathbf{w}_{id_1}, id))c$ is greater than $\gamma_3 - \beta_2$, we restart. The first check is necessary for security, while the second is necessary for both security and

correctness. Cause when the verifier computes $\mathbf{w}''$ in Verify algorithm, the high-order bits of $\mathbf{Az}-(\mathbf{w}_{id}+\mathbf{t}H(\mathbf{w}_{id_1},id))c$ do not depend too much on the low order bits of $\mathbf{w}_{id}$ because $\mathbf{w}_{id}$ is being multiplied by a very low-weight polynomial c. So we drop low-order bits of $\mathbf{w}_{id}$ at the beginning. To make up for this, the signer includes some hints as part of the signature, which are essentially the carries caused by adding in the product of c with the missing low-order bits of $\mathbf{w}_{id}$. With this hint, the verifier is able to correctly compute $\mathbf{w}'$.

Algorithm 15: Sign

Input: mpk, sk_{id}, μ

1 $\mathbf{w}_{id_1} = \mathsf{Power2Round}_{\mathsf{q}}(\mathbf{w}_{id}, d)$
2 $\mathbf{w}_{id_0} = \mathbf{w}_{id} - \mathbf{w}_{id_1} \cdot 2^d$
3 $\mathbf{y} \leftarrow S^l_{\gamma_2}$
4 $\mathbf{w} = \mathbf{Ay}$
5 $\mathbf{w}' = \mathsf{HighBits}_{\mathsf{q}}(\mathbf{w}, 2\gamma_3)$
6 $c = H(id, \mathbf{w}', \mu)$
7 $\mathbf{z} = \mathbf{s}_{id_1}\mathbf{c} + \mathbf{y}$
8 $(\mathbf{r}_1, \mathbf{r}_0) = \mathsf{Decompose}_{\mathsf{q}}(\mathbf{w} - \mathbf{s}_{id_2}\mathbf{c}, 2\gamma_3)$
9 **If** $||\mathbf{z}_{id}||_\infty \geq \gamma_2 - \beta_2$ or $||\mathbf{r}_0||_\infty \geq \gamma_3 - \beta_2$ or $||\mathbf{w}_{id_0}c||_\infty \geq \gamma_3$, then goto **3**
10 $\mathbf{h} = \mathsf{MakeHint}_{\mathsf{q}}(-\mathbf{w}_{id_0}\mathbf{c}, \mathbf{w} - \mathbf{s}_{id_2}\mathbf{c} + \mathbf{w}_{id_0}\mathbf{c}, 2\gamma_3)$
11 **Return** $sig = (\mathbf{z}, \mathbf{c}, \mathbf{w}_{id_1}, \mathbf{h})$

In the verification, the verifier first computes $\mathbf{w}''$ to be the high-order bits of $\mathbf{Az} - (\mathbf{w}_{id} + \mathbf{t}H(\mathbf{w}_{id_1}, id))c$ using hints, and then accepts if all the coefficients of $\mathbf{z}$ are less than $\gamma_2 - \beta_2$ and if c is the correct hash value.

Algorithm 16: Verify

Input: mpk, μ, m, id

1 $\mathbf{w}'' = \mathsf{UseHint}_{\mathsf{q}}(\mathbf{h}, \mathbf{Az} - (\mathbf{w}_{id_1} \cdot 2^d + \mathbf{t}H(\mathbf{w}_{id_1}, id))\mathbf{c}, 2\gamma_3)$
2 **Return** $||\mathbf{z}||_\infty < \gamma_2 - \beta_2$ and $c == H(id, \mathbf{w}'', \mu)$

4.3 Correctness Analysis

If $||\mathbf{w}_{id_0}c||_\infty < \gamma_3$, then by the lemmas in 4.1 we know that

$$\mathsf{UseHint}_{\mathsf{q}}(\mathbf{h}, \mathbf{w} - \mathbf{s}_{id_2}c + \mathbf{w}_{id_0}c, 2\gamma_3) = \mathsf{HighBits}_{\mathsf{q}}(\mathbf{w} - \mathbf{s}_{id_2}c, 2\gamma_3).$$

Since $\mathbf{w} = \mathbf{Ay}$ and $\mathbf{As}_{id_1} + \mathbf{s}_{id_2} = \mathbf{w}_{id} + \mathbf{t}H(\mathbf{w}_{id_1}, id)$, we have that

$$\begin{aligned}\mathbf{w} - \mathbf{s}_{id_2}c &= \mathbf{Ay} - \mathbf{s}_{id_2}c \\ &= \mathbf{A}(\mathbf{z} - \mathbf{s}_{id_1}c) - \mathbf{s}_{id_2}c \\ &= \mathbf{Az} - (\mathbf{w}_{id} + \mathbf{t}H(\mathbf{w}_{id_1}, id))c\end{aligned}$$

and $\mathbf{w} - \mathbf{s}_{id_2}c + \mathbf{w}_{id_0}c = \mathbf{Az} - (\mathbf{w}_{id_1} \cdot 2^d + \mathbf{t}H(\mathbf{w}_{id_1}, id))c$, therefore the verifier computes

$$\mathsf{UseHint}_{\mathsf{q}}(\mathbf{h}, \mathbf{Az} - (\mathbf{w}_{id_1} \cdot 2^d + \mathbf{t}H(\mathbf{w}_{id_1}, id))c, 2\gamma_3) = \mathsf{HighBits}_{\mathsf{q}}(\mathbf{w} - \mathbf{s}_{id_2}c, 2\gamma_3).$$

Furthermore, because β_2 is set such that $||\mathbf{z}_{id_2}c||_\infty \leq \beta_2$ and the signer checks in Line (9) that $\mathsf{LowBits}_\mathsf{q}(\mathbf{w} - \mathbf{s}_{id_2}c, 2\gamma_3) \leq \gamma_3 - \beta_2$, this implies that

$$\mathsf{HighBits}_\mathsf{q}(\mathbf{w} - \mathbf{s}_{id_2}\mathbf{c}, 2\gamma_3) = \mathsf{HighBits}_\mathsf{q}(\mathbf{w}, 2\gamma_3) = \mathbf{w}'.$$

Therefore, the $\mathbf{w}''$ computed by the verifier in the input to the hash function is the same as the $\mathbf{w}'$ of the signer. And thus the verification procedure will always accept.

We estimate the BKZ blocksize b required for successful forgery attack against our scheme by Python script provided at https://github.com/pq-crystals/security-estimates. Specifically, the cost of BKZ with blocksize b is estimated as $2^{0.292b}$ [44] in the classical setting and $2^{0.265b}$ [45] in the quantum setting. We offer three different parameter settings for different security environments.

Table 1. parameter settings 1

Security Level	**90 bits**	**136 bits**	**200 bits**
$log_2(q)$[modulus]	34	36	38
(k, l)[dimensions]	(6,5)	(8,7)	(11,10)
κ[# of ± 1 in c]	39	39	49
challenge entropy[$C_{256}^{\kappa} + \kappa$]	192	192	225
η	1	1	1
γ_1[$\mathbf{y}_{id}$ range]	49920	69888	125440
β_1[$\kappa\eta$]	39	39	49
d[dropped bits]	26	27	28
β_2[$\kappa\gamma_1$]	1946880	2725632	6146560
γ_2[$\mathbf{y}$ range]	4984012800	9768665088	31470387200
γ_3[low-order range]	$\gamma_2/2$	$\gamma_2/2$	$\gamma_2/2$
Extract Repetitions	9.02	8.52	8.17
Sign Repetitions	5.47	5.16	4.95
Signature size	6691 bytes	9738 bytes	14680 bytes
BKZ block-size b	312	470	688

4.4 Security Analysis

Now we give a security reduction sketch of the modified scheme.

The MLWE assumption is needed to protect against key-recovery, and the MSIS is the assumption upon which key and message forgery are based.

The simulation of the extract and sign algorithms follows subsection 3.2 and [25]. All the returns of our scheme are based on public information and are therefore simulatable.

If we thus assume that the MLWE problem is hard, where D is the distribution that samples a uniform integer in the range $[-\lambda, \lambda]$, then to prove EUF-ID-CMA security, we only need analyze the hardness of the experiment where

the adversary receives a random ($\mathbf{A}$,$\mathbf{t}$) and outputs a valid identity-message-signature triplet $id, \mu, (\mathbf{z}, \mathbf{h}, \mathbf{w}_1, c)$ such that $||\mathbf{z}||_\infty \leq \gamma_2 - \beta_2$ and

$$H(id, \mathsf{UseHint_q}(\mathbf{h}, \mathbf{Az} - (\mathbf{w}_1 \cdot 2^d + \mathbf{t}H(\mathbf{w}_1, id))c, 2\gamma_3), \mu) = c.$$

By Lemma 1 we can rewrite it into

$$2\gamma_3 \cdot \mathsf{UseHint_q}(\mathbf{h}, \mathbf{Az} - (\mathbf{w}_1 \cdot 2^d + \mathbf{t}H(\mathbf{w}_1, id))c, 2\gamma_3) = \mathbf{Az} - (\mathbf{w}_1 \cdot 2^d + \mathbf{t}H(\mathbf{w}_1, id))c + \mathbf{u},$$

where $||\mathbf{u}||_\infty \leq 2\gamma_3 + 1$. Because $\mathbf{w} = \mathbf{w}_1 \cdot 2^d + \mathbf{w}_0$, we can further rewrite

$$\mathbf{Az} - (\mathbf{w}_1 \cdot 2^d + \mathbf{t}H(\mathbf{w}_1, id))c + \mathbf{u} = \mathbf{Az} - (\mathbf{w} + \mathbf{t}H(\mathbf{w}_1, id))c + \mathbf{u}'.$$

And the the worst-case upper-bound for $\mathbf{u}'$ is

$$\begin{aligned} ||\mathbf{u}'||_\infty &\leq ||\mathbf{w}_0 c||_\infty + ||\mathbf{u}||_\infty \\ &\leq ||\mathbf{w}_0||_\infty \cdot ||c||_1 + ||\mathbf{u}||_\infty \\ &\leq 2^{d-1} \cdot \kappa + 2\gamma_3 + 1. \end{aligned}$$

Thus a adversary who is successful at creating a forgery of a message for a chosen identity is able to find $id, \mu, \mathbf{z}, \mathbf{u}', \mathbf{w}, c$ such that $||\mathbf{z}||_\infty \leq \gamma_2 - \beta_2$, $||\mathbf{u}'||_\infty \leq 2^{d-1} \cdot \kappa + 2\gamma_3 + 1$ and satisfy

$$H(id, [\mathbf{A}|\mathbf{I_k}] \begin{bmatrix} \mathbf{z} \\ \mathbf{u}' \end{bmatrix} - (\mathbf{w} + \mathbf{t}H(\mathbf{w}, id))c, \mu) = c. \tag{4.1}$$

This is actually variation of Eq.(3.1) which could reduce to MSIS problem in random oracle model.

4.5 Comparison

According to the results of our study, the most efficient IBS schemes on lattices in the random oracle model are [10] and are based on NTRU lattices. However, as we mentioned in Subsection 1, finding a short vector in the NTRU lattice may be weaker than initially assumed, particularly for large parameters.

Sageloli et al. [14] and Foo [13] have proposed IBS schemes in hash-and-sign paradigm. Their works focus on constructing an IBS scheme in QROM with tight security.

The scheme [11] is similar to ours. However, their scheme uses the signing key as the masking vector $\mathbf{y}$ and takes a random vector $\mathbf{s}$ in Sign algorithm, while the signature $\mathbf{z} = \mathbf{s}c + \mathbf{y}$ is an addition of $\mathbf{y}$ with large coefficient and $\mathbf{s}c$ with small coefficient, which makes information about the signing key easily leaked.

We simulated [10](for N adapted to 1024) and our Dilithium-like scheme in Python 3.9 under the test environment of Windows 10 operating system and AMD Ryzen 5 5600G 3.90 GHz processor. In similar security conditions, the signature size of our scheme is 52% smaller and Extract algorithm runtime is 32% faster than [10].

Table 2. Comparison

Scheme	hardness assumptions	secure bits	signature	Setup	Extract	Sign	Verify
[13]	SIS	≈120	> 5MB	-	-	-	-
[14]	Ring-SIS	≈120	> 1MB	-	-	-	-
[10]	NTRU,Ring-SIS	60	12918 bytes	11652 ms	70 ms	23 ms	12 ms
ours	MLWE,MSIS	90	6691 bytes	9 ms	47 ms	28 ms	11 ms

5 Conclusion

In this paper, we constructed a new IBS scheme based on lattices. Our scheme does not use the trapdoor sampling technique. Moreover, we optimize our scheme using many different techniques. This makes our scheme having a more compact signature size and faster running time than other LB-IBS schemes. As a future work, we will consider constructing a certificateless encryption and signature schemes without trapdoor.

Acknowledgements

This work is supported by the Major Program of Guangdong Basic and Applied Research (2019B030302008), National Natural Science Foundation of China (62272174),and Science and Technology Program of Guangzhou(2024A04J6542).

References

1. Adi Shamir. Identity-based cryptosystems and signature schemes. In George Robert Blakley and David Chaum, editors, *Advances in Cryptology*, pages 47–53, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.
2. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, page 197206, New York, NY, USA, 2008. Association for Computing Machinery.
3. Markus Rückert. Strongly unforgeable signatures and hierarchical identity-based signatures from lattices without random oracles. In Nicolas Sendrier, editor, *Post-Quantum Cryptography*, pages 182–200, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
4. Zhenhua Liu, Yupu Hu, Xiangsong Zhang, and Fagen Li. Efficient and strongly unforgeable identity-based signature scheme from lattices in the standard model. *Security and Communication Networks*, 6(1):69–77, 2013.
5. Miaomiao Tian, Liusheng Huang, and Wei Yang. Efficient hierarchical identity-based signatures from lattices. *International Journal of Electronic Security and Digital Forensics*, 5(1):1–10, 2013. PMID: 54403.
6. Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, pages 598–616, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

7. Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 738–755, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
8. Miaomiao Tian and Liusheng Huang. Efficient identity-based signature from lattices. In Nora Cuppens-Boulahia, Frédéric Cuppens, Sushil Jajodia, Anas Abou El Kalam, and Thierry Sans, editors, *ICT Systems Security and Privacy Protection*, pages 321–329, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.
9. Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over ntru lattices. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014*, pages 22–41, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.
10. Jia Xie, Yu-pu Hu, Jun-tao Gao, and Wen Gao. Efficient identity-based signature over ntru lattice. *Frontiers of Information Technology & Electronic Engineering*, 17(2):135–142, 2016.
11. Jiang-shan Chen, Yu-pu Hu, Hong-mei Liang, and Wen Gao. Novel efficient identity-based signature on lattices. *Frontiers of Information Technology & Electronic Engineering*, 22(2):244–250, 2021.
12. Jiaxin Pan and Benedikt Wagner. Short identity-based signatures with tight security from lattices. In *Post-Quantum Cryptography: 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20–22, 2021, Proceedings 12*, pages 360–379. Springer, 2021.
13. Ernest Foo and Qinyi Li. Tightly secure lattice identity-based signature in the quantum random oracle model. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *Information Security and Privacy*, pages 381–402, Cham, 2023. Springer Nature Switzerland.
14. Éric Sageloli, Pierre Pébereau, Pierrick Méaux, and Céline Chevalier. Shorter andăfaster identity-based signatures withătight security inătheă(q)rom fromălattices. In Mehdi Tibouchi and XiaoFeng Wang, editors, *Applied Cryptography and Network Security*, pages 634–663, Cham, 2023. Springer Nature Switzerland.
15. Xinjian Chen, Qiong Huang, Hongbo Li, Zhijian Liao, and Willy Susilo. A novel identity-based multi-signature scheme over ntru lattices. *Theoretical Computer Science*, 933:163–176, 2022.
16. Junbin Liang, Qiong Huang, Jianye Huang, Liantao Lan, and Man Ho Allen Au. An identity-based traceable ring signatures based on lattice. *Peer-to-Peer Networking and Applications*, 16(2):1270–1285, 2023.
17. Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon. *Post-Quantum Cryptography Project of NIST*, 2020.
18. Léo Ducas and Thomas Prest. Fast fourier orthogonalization. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '16, page 191198, New York, NY, USA, 2016. Association for Computing Machinery.
19. Thomas Espitau, Pierre-Alain Fouque, François Gérard, Mélissa Rossi, Akira Takahashi, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu. Mitaka: A simpler, parallelizable, maskable variant ofăfalcon. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022*, pages 222–253, Cham, 2022. Springer International Publishing.
20. Thomas Espitau, Thi Thu Quyen Nguyen, Chao Sun, Mehdi Tibouchi, and Alexandre Wallet. Antrag: Annular ntru trapdoor generation. In Jian Guo and Ron Stein-

feld, editors, *Advances in Cryptology – ASIACRYPT 2023*, pages 3–36, Singapore, 2023. Springer Nature Singapore.

21. Chitchanok Chuengsatiansup, Thomas Prest, Damien Stehlé, Alexandre Wallet, and Keita Xagawa. Modfalcon: Compact signatures based on module-ntru lattices. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, ASIA CCS '20, page 853866, New York, NY, USA, 2020. Association for Computing Machinery.
22. Paul Kirchner and Pierre-Alain Fouque. Revisiting lattice attacks on overstretched ntru parameters. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, pages 3–26, Cham, 2017. Springer International Publishing.
23. Léo Ducas and Wessel van Woerden. Ntru fatigue: How stretched is overstretched? In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 3–32, Cham, 2021. Springer International Publishing.
24. Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehle. Crystals - kyber: A cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367, 2018.
25. Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238268, Feb. 2018.
26. Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.
27. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 700–718, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
28. Pauline Bert, Gautier Eberhart, Lucas Prabel, Adeline Roux-Langlois, and Mohamed Sabt. Implementation of lattice trapdoors on modules and applications. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography*, pages 195–214, Cham, 2021. Springer International Publishing.
29. Yang Yu, Huiwen Jia, and Xiaoyun Wang. Compact lattice gadget and its applications to hash-and-sign signatures. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 390–420, Cham, 2023. Springer Nature Switzerland.
30. James Howe, Thomas Prest, Thomas Ricosset, and Mélissa Rossi. Isochronous gaussian sampling: Fromăinception to implementation. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography*, pages 53–71, Cham, 2020. Springer International Publishing.
31. Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, pages 40–56, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
32. Peter Pessl, Leon Groot Bruinderink, and Yuval Yarom. To bliss-b or not to be: Attacking strongswan's implementation of post-quantum signatures. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 18431855, New York, NY, USA, 2017. Association for Computing Machinery.

33. David Galindo and Flavio D. Garcia. A schnorr-like lightweight identity-based signature scheme. In Bart Preneel, editor, *Progress in Cryptology – AFRICACRYPT 2009*, pages 135–148, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
34. Sanjit Chatterjee, Chethan Kamath, and Vikas Kumar. Galindo-garcia identity-based signature revisited. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *Information Security and Cryptology – ICISC 2012*, pages 456–471, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
35. Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *Topics in Cryptology – CT-RSA 2014*, pages 28–47, Cham, 2014. Springer International Publishing.
36. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems – CHES 2012*, pages 530–547, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
37. Jung Hee Cheon, Hyeongmin Choe, Julien Devevey, Tim Güneysu, Dongyeon Hong, Markus Krausz, Georg Land, Marc Möller, Damien Stehlé, and MinJune Yi. Haetae: Shorter lattice-based fiat-shamir signatures. Cryptology ePrint Archive, Paper 2023/624, 2023. https://eprint.iacr.org/2023/624.
38. Julien Devevey, Alain Passelègue, and Damien Stehlé. G+g: A fiat-shamir lattice signature based onăconvolved gaussians. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023*, pages 37–64, Singapore, 2023. Springer Nature Singapore.
39. Mihir Bellare and Gregory Neven. 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*, CCS '06, page 390399, New York, NY, USA, 2006. Association for Computing Machinery.
40. Alexandra Boldyreva, Adriana Palacio, and Bogdan Warinschi. Secure proxy signature schemes for delegation of signing rights. *Journal of Cryptology*, 25:57–115, 2012.
41. Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In *Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part III 37*, pages 552–586. Springer, 2018.
42. Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39*, pages 326–355. Springer, 2019.
43. Julien Devevey, Pouria Fallahpour, Alain Passelègue, and Damien Stehlé. A detailed analysis ofăfiat-shamir withăaborts. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 327–357, Cham, 2023. Springer Nature Switzerland.
44. Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 10–24. SIAM, 2016.
45. Thijs Laarhoven. *Search problems in cryptography: from fingerprinting to lattice sieving*. PhD thesis, Mathematics and Computer Science, February 2016. Proefschrift.