An Efficient Proxy Re-Signature Over Lattices

Mingming Jiang[1], Jinqiu Hou[1], Yuyan Guo[1(✉)], Yan Wang[2], and Shimin Wei[1]

[1] School of Computer Science and Technology, Huaibei Normal University, Huaibei 235000, China
guoyuyan428@163.com
[2] School of Mathematics Science, Huaibei Normal University, Huaibei 235000, China

Abstract. In 2008, Libert and Vergnaud constructed the first multi-use unidirectional proxy re-signature scheme. In this scheme, the proxy can translate the signatures several times but only in one direction. Thus, two problems remain open. That is, to construct a multi-use unidirectional proxy re-signature scheme based on classical hardness assumptions, and to design a multi-use unidirectional proxy re-signature scheme with the size of signatures and the verification cost growing sub-linearly with the number of translations. This paper solves the first problem and sharply reduces the verification costs. We use the preimage sampleable algorithm to develop a multi-use unidirectional proxy re-signature scheme based on lattices, namely, the hardness of the Small Integer Solution (SIS) problem. The verification cost does not grow with the number of translations and the size of signatures grows linearly with the number of translations in this scheme. Furthermore, the proposal is secure in quantum environment.

Keywords: Lattice cryptography · Proxy re-signature scheme · Small Integer Solution (SIS) problem · Gaussian Sample · Multi-use

1 Introduction

Proxy re-signature is proposed by Blaze, Bleumer, and Strauss [1]. In a proxy re-signature scheme, a semi-trusted proxy is given some information that allows it to transform Alice's signature into Bob's signature on the same message, but the proxy cannot generate signatures for Alice or Bob on its own. In [1], the first proxy re-signature scheme is constructed and is proven to be multi-use and bidirectional. However, the proxy re-signature primitive was seldom noticed until 2005. In 2005, Ateniese and Hohenberger [2] formalized the definition of security and illustrated the applications of proxy re-signature schemes. What follows presents some properties that will be taken into account in a proxy re-signature scheme.

1. Unidirectional: the proxy only can turn the Alice's signatures into the Bob's signatures, but the reverse is not true.
2. Multi-use: a signature can be re-signed many times;
3. Private Proxy: re-signature keys are kept secret;
4. Transparent: we can not distinguish the re-signatures from the original signatures;

B. Shen et al. (Eds.): FCS 2019, CCIS 1105, pp. 145–160, 2019.
https://doi.org/10.1007/978-981-15-0818-9_10

5. Key optimal: a user is only required to store a constant amount of secret data;
6. Non-interactive: the delegatee does not participate in the process of the generation of the proxy re-signature key;
7. Non-transitive: the re-signing rights cannot be re-delegated by the proxy;
8. Unlinkable: a re-signature cannot be linked to the one from which it was generated.

In [2], three proxy re-signature schemes were proposed: the first one is multi-use and bidirectional with a private re-signature key; the second one is single-use and unidirectional with a public re-signature key; the third one is single-use and unidirectional with a private re-signature key. The possible applications of a re-signature scheme may include the space-efficient proof, group signatures management, simplification of certificate management. However, it remains an open problem to design a multi-use unidirectional re-signature scheme. To solve this problem, Labert and Vergnaud [3] proposed two multi-use and unidirectional schemes with a private re-signature key based on the *l*-FlexDH assumption (in the random oracle model and the standard model, respectively). However, we are confronted with two open problems: one is to construct a multi-use unidirectional proxy re-signature scheme under the standard hardness assumptions; the other is to reduce the size of signatures and the verification costs. Sunitha and Amberker [4] proposed another multi-use unidirectional proxy re-signature scheme, but the scheme only obtains a forward security, and hence is not provably secure. Sunitha [5] constructed a proxy signature schemes that translates Alice's Schnorr/ElGamal/RSA signature to Bob's RSA signature, but failed to prove the security. Shao et al. [6] proposed the first multi-use bidirectional proxy re-signature scheme in the standard model and extended it to the ID-based case. Shao et al. [7] proposed the first unidirectional identity based proxy re-signature in the random oracle based on the Schnorr's signature and the Libert-Vergnaud proxy re-signature. Shao et al. [8] analyzed and improved the previous security model [2] and gave a unidirectional proxy re-signature scheme to meet the new security model. Yang et al. [9] first defined the security model for threshold proxy re-signature scheme, and then proposed two threshold proxy re-signature schemes based on the Ateniese-Hohenberger's and the Shao-Cao-Wang-Liang's approach. However, the four proposals were built from the intractability assumptions for factoring large integers or solving discrete logarithms. Thus, they are not secure in the quantum setting and hence it is meaningful to construct a proxy re-signature scheme secure in the quantum setting.

As an important class of post-quantum cryptography, lattice cryptography attracts more and more attentions in the cryptographic literature in recent years due to the elegant cryptographic properties. First, lattice cryptography only involves some linear operations on small integers, and hence results in an asymptotically low computational complexity. Second, the security is supported by the worst-case to average-case equivalence connections. Since the first proposals of a provably secure lattice signature scheme and a lattice IBE scheme due to Gentry et al. [10], we are witnessing a rapid development of lattice cryptography. Many lattice schemes are constructed, such as the lattice-based public key encryption schemes [11–14], identity-based encryption schemes [10, 15–17], fully homomorphic encryption [18–21] and lattice-based signatures schemes [10, 22] and signature schemes with particular properties [23–25].

1.1 Contributions

We aim at the open problems left by Libert and Vergnaud over lattices. In our scheme, the proxy re-signature key is generated by the Gaussian Sample algorithm. First, given two public keys $pk_1 = \boldsymbol{A}_1$, $pk_2 = \boldsymbol{A}_2$ of users 1 and 2 and the secret key of user 2, use the Gaussian Sample algorithm to generate the proxy re-signature key $\boldsymbol{S}_{1\to 2}$, such that $\boldsymbol{A}_2\boldsymbol{S}_{1\to 2} = \boldsymbol{A}_1 \bmod q$. Second, gives an original signature $\boldsymbol{e}_1$ of user 1, and the re-signature $\boldsymbol{e}_2 = \boldsymbol{S}_{1\to 2}\boldsymbol{e}_1$. We know that the proxy re-signature key $\boldsymbol{S}_{1\to 2}$ has two properties: (1) its norm is small; (2) its distribution is statistically close to a Gaussian distribution. Then the distribution of the re-signature is statistically close to a Gaussian distribution and its norm is also small. Thus, the proxy re-signature has the same properties as the original signature.

1.2 Organization

In Sect. 2, we formalize the related notations, review the definitions of lattice and Gaussian distribution, introduce the lattice basis delegation technique, and define the Small Integer Solution hardness assumption on which the security of our scheme is based. We describe the definition and security model of a Proxy Re-Signature scheme in Sect. 3. In Sect. 4, we propose a Multi-Use Unidirectional Proxy Re-Signature scheme based on lattice in the random model. The scheme in the standard model is constructed in Sect. 5. Finally, the conclusion is given in Sect. 6.

2 Preliminaries

2.1 Notation

We denote sets of real numbers by $\mathbb{R}$ and the integers by $\mathbb{Z}$, respectively. Vectors are written as bold italic lower-case letters, e.g. $\boldsymbol{x}$. The i-th component of $\boldsymbol{x}$ is denoted by x_i. Matrices are written as bold italic capital letters, e.g. $\boldsymbol{X}$, and the i-th column vector of a matrix $\boldsymbol{X}$ is denoted $\boldsymbol{x}_i$. The Euclidean norm l_2 norm of a vector x is denoted as $\|\boldsymbol{x}\|_2 = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x}\rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}$. Generally, we abbreviate $\|\boldsymbol{x}\|_2$ as $\|\boldsymbol{x}\|$. The length of a matrix is defined as the norm of the longest column, namely, $\|\boldsymbol{X}\| = \max_i \|\boldsymbol{x}_i\|$, for $1 \le i \le k$.

2.2 Lattice

Let $\boldsymbol{B} = \{\boldsymbol{b}_1, \cdots, \boldsymbol{b}_m\} \in \mathbb{R}^{m\times m}$ be an $m \times m$ matrix whose columns are linearly independent vectors $\boldsymbol{b}_1, \cdots, \boldsymbol{b}_m \in \mathbb{R}^m$. The m-dimensional lattice Λ generated by $\boldsymbol{B}$,

$$\Lambda = \mathcal{L}(\boldsymbol{B}) = \left\{ \boldsymbol{y} \in \mathbb{R}^m \;\; s.t. \;\; \exists \boldsymbol{x} \in \mathbb{Z}^m, \;\; \mathbf{y} = \boldsymbol{B}\boldsymbol{x} = \sum_{i=1}^{m} x_i \boldsymbol{b}_i \right\} \tag{1}$$

Here, we focus on inter lattices, i.e., $\mathcal{L}$ is contained in $\mathbb{Z}^m$.

Definition 1. For q prime, $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, $\boldsymbol{u} \in \mathbb{Z}_q^n$, define:

$$\Lambda_q^{\perp}(\boldsymbol{A}) := \{\boldsymbol{e} \in \mathbb{Z}^m \ \ s.t. \ \ \boldsymbol{Ae} = \boldsymbol{0}(\mathrm{mod}q)\} \tag{2}$$

$$\Lambda_q^{\boldsymbol{u}}(\boldsymbol{A}) := \{\boldsymbol{e} \in \mathbb{Z}^m \ \ s.t. \ \ \boldsymbol{Ae} = \boldsymbol{u}\,(\mathrm{mod}q)\} \tag{3}$$

Observe that if $\boldsymbol{t} \in \Lambda_q^{\boldsymbol{u}}(\boldsymbol{A})$, then $\Lambda_q^{\boldsymbol{u}}(\boldsymbol{A}) = \Lambda_q^{\perp}(\boldsymbol{A}) + \boldsymbol{t}$, hence $\Lambda_q^{\boldsymbol{u}}(\boldsymbol{A})$ is a shift of $\Lambda_q^{\perp}(\boldsymbol{A})$.

Lemma 1 [26]. Let $q \geq 3$ be odd and $m = \lceil 6n \log q \rceil$. There is a probabilistic polynomial-time algorithm TrapGen(q, n) that outputs two matrixes $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$ and $\boldsymbol{T} \in \mathbb{Z}_q^{m\times m}$ such that $\boldsymbol{A}$ is statistically close to a uniform matrix in $\mathbb{Z}_q^{n\times m}$ and $\boldsymbol{T}$ is a basis for $\Lambda_q^{\perp}(\boldsymbol{A})$ satisfying

$\|\tilde{\boldsymbol{T}}\| \leq O(\sqrt{n \log q})$ and $\|\boldsymbol{T}\| \leq O(n \log q)$ with all but negligible probability in n.

2.3 Discrete Gaussians

We briefly recall Discrete Gaussian Distributions over lattices.

For any positive parameter $\sigma > 0$ define the Gaussian function on $\mathbb{R}^m$ centered at $\boldsymbol{c}$:

$$\forall \boldsymbol{x} \in \mathbb{R}^m, \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) = \exp\left(-\pi\|\boldsymbol{x} - \boldsymbol{c}\|^2 \Big/ \sigma^2\right) \tag{4}$$

For any $\boldsymbol{c} \in \mathbb{R}^m$, real $\sigma > 0$, and an m-dimensional Λ, define the Discrete Gaussian Distribution over Λ as:

$$\forall \boldsymbol{x} \in \mathbb{R}^m, D_{\Lambda,\sigma,\boldsymbol{c}}(\boldsymbol{x}) = \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})}{\rho_{\sigma,\boldsymbol{c}}(\Lambda)} = \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})}{\sum_{x\in\Lambda} \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})} \tag{5}$$

Lemma 2 [10]. Let $q \geq 2$ and a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, $m > n$. Let $\boldsymbol{T_A}$ be a basis for $\Lambda_q^{\perp}(\boldsymbol{A})$, $\sigma \geq \|\tilde{\boldsymbol{T}}_{\boldsymbol{A}}\| \cdot \omega(\sqrt{\log m})$. Then for $\boldsymbol{c} \in \mathbb{R}^m$, $\boldsymbol{u} \in \mathbb{Z}_q^n$:

1. $\Pr\left[\boldsymbol{x} \sim D_{\Lambda_q^{\perp}(\mathbf{A}),\sigma} : \|\boldsymbol{x}\| > \sigma\sqrt{m}\right] \leq negl(n)$.
2. There is a polynomial-time algorithm SampleGaussian $(\boldsymbol{A}, \boldsymbol{T_A}, \sigma, \boldsymbol{c})$ that returns $\boldsymbol{x} \in \Lambda_q^{\perp}(\boldsymbol{A})$ drawn from a distribution statistically close to $D_{\Lambda_q^{\perp}(\boldsymbol{A}),\sigma,\boldsymbol{c}}$.
3. There is a polynomial-time algorithm SamplePre $(\boldsymbol{A}, \boldsymbol{T_A}, \boldsymbol{u}, \sigma)$ that returns $\boldsymbol{x} \in \Lambda_q^{\boldsymbol{u}}(\boldsymbol{A})$ sampled from a distribution statistically close to $D_{\Lambda_q^{\boldsymbol{u}}(\boldsymbol{A}),\sigma,\boldsymbol{c}}$.

Definition 2. For any m-dimensional lattice Λ and positive real $\epsilon > 0$, the smoothing parameter η_ϵ is the smallest real $\sigma > 0$ such that $\rho_{1/\sigma}(\Lambda^* \backslash \{0\}) \leq \epsilon$.

Lemma 3 [27]. Let $\Lambda \subseteq \mathbb{Z}^m$ be a lattice and $\sigma \in \mathbb{R}$. For $i = 1, \cdots, k$, $\boldsymbol{v}_i \in \mathbb{Z}^m$ and let X_i be mutually independent random variables sampled from $D_{\Lambda+\boldsymbol{v}_i,\sigma}$. Let $\boldsymbol{c} = (c_1, \cdots,$

$c_k) \in \mathbb{Z}^k$, and define $g := \gcd(c_1, \cdots, c_k)$, and $\boldsymbol{v} := \sum_{i=1}^{k} c_i \boldsymbol{v}_i$. Suppose that $\sigma > \|\boldsymbol{c}\| \cdot \eta_\epsilon(\Lambda)$ for some negligible ϵ. Then $Z = \sum_{i=1}^{k} c_i X_i$ is statistically close to $D_{g\Lambda+\boldsymbol{v},\|\boldsymbol{c}\|\sigma}$.

Definition 3. We say that a matrix $\boldsymbol{A}$ in $\mathbb{Z}^{m\times m}$ is $\mathbb{Z}_q$-invertible if $\boldsymbol{A} \bmod q$ is invertible as a matrix in $\mathbb{Z}_q^{m\times m}$.

Algorithm 1. [16] Sample$\boldsymbol{S}(1^m)$
Let $\sigma_s = O(\sqrt{n \log q}) \cdot \omega(\log m) \cdot \sqrt{m}$

1. Let $\boldsymbol{T}_0$ be the canonical basis of the lattice $\mathbb{Z}^m$;
2. For $i = 1, \cdots, m$ do $\boldsymbol{s}_i \xleftarrow{R}$ SampleGaussian$(\mathbb{Z}^m, \mathbf{T}_0, \sigma_s, \boldsymbol{0})$;
3. If $\boldsymbol{S}$ is $\mathbb{Z}_q$-invertible, output $\boldsymbol{S}$; otherwise repeat step 2.

2.4 The SIS Problem

In this section, we recall the Small Integer Solution problem, which is essentially the knapsack problem over elements in $\mathbb{Z}_q^n$. We focus on $l_2 - \mathrm{SIS}_{q,n,m,\beta}$ problem.

Definition 4 ($l_2 - \mathrm{SIS}_{q,n,m,\beta}$ problem). Given an integer q, a random matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$ and a real β, find a vector $\boldsymbol{v} \in \mathbb{Z}^m \backslash \{0\}$ such that $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$ and $||\boldsymbol{v}|| \le \beta$.

The following lemma shows that $l_2 - \mathrm{SIS}_{q,n,m,\beta}$ problem is as hard as approximating certain worst-case problems on lattice.

Lemma 4 [10]. For any poly-bounded m, $\beta = poly(n)$ and for any prime $q \ge \beta \cdot \omega(\sqrt{n \log n})$, the average-case problem $l_2 - \mathrm{SIS}_{q,n,m,\beta}$ is as hard as approximating the SIVP problem in the worst-case to within certain $\gamma = \beta \cdot \tilde{O}(\sqrt{n})$.

Lemma 5 [16]. Let $q > 2$, $m > 2n \log q$ and $\sigma > ||\tilde{\boldsymbol{T}}_{\boldsymbol{A}}|| \cdot \omega(\sqrt{\log 2m})$. Then there exists a polynomial-time algorithm *SampleBasisLeft*$(\boldsymbol{A}, \boldsymbol{M}, \boldsymbol{T}_{\boldsymbol{A}})$ takes $\boldsymbol{A}, \boldsymbol{M} \in \mathbb{Z}_q^{n\times m}$ and a basis $\boldsymbol{T}_{\boldsymbol{A}}$ of $\Lambda_q^{\perp}(\boldsymbol{A})$ as inputs, outputs a basis $\boldsymbol{T}_{\boldsymbol{F}}$ of $\Lambda_q^{\perp}(\boldsymbol{F})$ with $||\tilde{\boldsymbol{T}}_{\boldsymbol{A}}|| = ||\tilde{\boldsymbol{T}}_{\boldsymbol{F}}||$, where $\boldsymbol{F} = (\boldsymbol{A}|\boldsymbol{M})$.

3 Proxy Re-Signature: Definition and Security Model

3.1 Definition of Unidirectional Proxy Re-Signature

In this section we recall the definition of the unidirectional proxy re-signature schemes. The unidirectional proxy re-signature scheme for L levels consists of five algorithms (KeyGen, ReKeyGen, Sign, ReSign, Verify)

KeyGen: This algorithm takes as input a security parameter n and returns a user's private/public key pair (sk, pk).

ReKeyGen: This algorithm takes as input user i's public key pk_i, user j's private key sk_j and returns a re-signature key $rk_{i\to j}$ that allows translating i's signatures into j's signatures. The re-signature key $rk_{i\to j}$ is secret.

Sign: This algorithm takes as input a message μ, a private key sk_i, an integer $l \in [L]$ and returns a signature θ on behalf of user i at level l.

ReSign: This algorithm takes as input public parameters, a level l signature θ for message μ from user i, a re-signature key $rk_{i\to j}$ and checks that θ is valid. If so, it returns a signature θ' which verifies at level $l+1$ under public key pk_j.

Verify: This algorithm takes as input public parameters, an integer $l \in [L]$, a message μ, a signature θ', a public key pk_j and returns 0 or 1.

Here, we explain that why the definition contains the level. In a proxy re-signature scheme, if we can distinguish the re-signatures from the original signatures. Without loss of generality, we say that original signatures are the Bob's first-level signatures and the re-signatures are the Bob's second-level signatures. We know that Alice and proxy can produce Bob's re-signatures (second-level signatures). Then it is a secure problem that the first-level signatures are generated by Alice and proxy. If we cannot distinguish the re-signatures from the original signatures, i.e. the first-level signatures and second-level signatures are indistinguishable, the level is not considered.

3.2 Security Model of Unidirectional Proxy Re-Signature

The security model of unidirectional proxy re-signature of [2] considers the following notions termed as external and internal security.

External Security: It is the security against adversaries except the proxy and delegation partners. Formally, for the security parameter n and all probability polynomial time adversaries $\mathcal{A}$:

$$\begin{aligned}\Pr[\{(pk_i, sk_i) \leftarrow KeyGen(1^n)\}_{i\in[1,k]},\\ (t,\mu,\theta) \leftarrow \mathcal{A}^{\mathcal{O}_{sign}(\cdot,\cdot),\mathcal{O}_{resign}(\cdot,\cdot,\cdot,\cdot)}(\{pk_i\}_{i\in[1,k]}) :\\ Verify(pk_t,\mu,\theta) = 1 \wedge (1 \le t \le k) \wedge (t,\mu,\theta) \notin Q] < 1/poly(n)\end{aligned} \tag{6}$$

where the oracle $\mathcal{O}_{sign}$ takes as input an index $i \in [1,k]$ and a message $\mu \in M$ and outputs a signature $\theta \leftarrow Sign(sk_j, \mu)$. The oracle $\mathcal{O}_{resign}$ takes as input two distinct indices $1 \le i, j \le k$, a message μ and a signature θ and outputs a re-signature $\theta' \leftarrow ReSign(rk_{i\to j}, pk_i, \theta, \mu)$. Let Q denotes the set of tuples (t,μ,θ) where $\mathcal{A}$ obtained a signature θ on μ under public key pk_t by querying $\mathcal{O}_{sign}$ on (t,μ) or $\mathcal{O}_{resign}(\cdot, t, \mu, \cdot)$.

Internal Security: This security model can be against the collusion attack (dishonest proxies and colluding delegation partners). The model contains three security guarantees.

1. **Limited Proxy:** This notion protects the honest delegator and delegatee, namely, the proxy can not forge the signatures of the delegatee or delegator unless the message was first signed by one of the latter's delegates. Formally, for the security parameter n and all probability polynomial time adversaries $\mathcal{A}$:

$$\Pr[\{(pk_i, sk_i) \leftarrow KeyGen(1^n)\}_{i\in[1,k]}, \\ (t, \mu, \theta) \leftarrow \mathcal{A}^{\mathcal{O}_{sign}(\cdot,\cdot),\mathcal{O}_{rekey}(\cdot,\cdot)}(\{pk_i\}_{i\in[1,k]}) : \\ Verify(pk_t, \mu, \theta) = 1 \wedge (1 \le t \le k) \wedge (t, \mu) \notin Q] < 1/poly(n) \tag{7}$$

where the oracle $\mathcal{O}_{sign}$ takes as input an index $i \in [1, k]$ and a message $\mu \in M$ and outputs a signature $\theta \leftarrow Sign(sk_j, \mu)$. The oracle $\mathcal{O}_{rekey}$ takes as input two distinct indices $1 \le i, j \le k$ and outputs the re-signature key $rk_{i\to j} \leftarrow ReKey(pk_i, pk_j, sk_j)$. Let Q denotes the set of tuples (t, μ) where $\mathcal{A}$ obtained a signature on μ under public key pk_t or one of its delegate key's by querying $\mathcal{O}_{sign}$.

2. **Delegatee Security:** This notion protects the delegate, i.e., it can be against the collusion attack from delegator and proxy. We associate the index 0 to the delegatee. Formally, for the security parameter n and all probability polynomial time adversaries $\mathcal{A}$:

$$\Pr[\{(pk_i, sk_i) \leftarrow KeyGen(1^n)\}_{i\in[1,k]}, \\ (\mu, \theta) \leftarrow \mathcal{A}^{\mathcal{O}_{sign}(0,\cdot),\mathcal{O}_{rekey}(\cdot,\star)}(pk_0, \{pk_i, sk_i\}_{i\in[1,k]}) : \\ Verify(pk_0, \mu, \theta) = 1 \wedge (\mu, \theta) \notin Q] < 1/poly(n) \tag{8}$$

where $\star \neq 0$ and Q is the set of pairs (μ, θ) such that $\mathcal{A}$ queried $\mathcal{O}_{sign}(0, \mu)$ and obtained θ.

3. **Delegator Security:** This notion protects the delegator, i.e., it can be against the collusion attack from delegatee and proxy. That is, there are distinguishable signatures for a user based on whether she used her strong secret key or her weak secret key. The colluding delegate and proxy cannot produce strong signatures (first-level signature) on her behalf. We associate the index 0 to the delegator. Formally, for the security parameter n and all probability polynomial time adversaries $\mathcal{A}$:

$$\Pr[\{(pk_i, sk_i) \leftarrow KeyGen(1^n)\}_{i\in[1,k]}, \\ (\mu, \theta) \leftarrow \mathcal{A}^{\mathcal{O}_{sign}(0,\cdot),\mathcal{O}_{rekey}(\cdot,\cdot)}(pk_0, \{pk_i, sk_i\}_{i\in[1,k]}) : \\ Verify(pk_0, \mu, \theta) = 1 \wedge (\mu, \theta) \notin Q] < 1/poly(n) \tag{9}$$

where θ is a first-level signature and Q is the set of pairs (μ, θ) such that $\mathcal{A}$ queried $\mathcal{O}_{sign}(0, \mu)$ and obtained θ.

4 Multi-use Unidirectional Proxy Re-Signature Scheme from Lattice in the Random Oracle Model

4.1 Our Construction

In this section, we use the Gentry, Peikert, and Vaikuntanathan's signature scheme [10] to construct a multi-use unidirectional proxy re-signature scheme. Let n be a security parameter, and $q \geq \beta \cdot \omega(\log n)$ for $\beta = poly(n)$. Let $m \geq 2n \log q$ and a Gaussian parameter $\sigma \geq O(\sqrt{n \log q}) \cdot \omega(\sqrt{\log n})$. There is a collision-resistant secure hash function H that maps $\{0,1\}^*$ to $\mathbb{Z}_q^n$. Our scheme consists of the following algorithms.

KeyGen: On input the security parameter n, run $TrapGen(q,n)$ to generate a random rank n matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ and a trapdoor basis $\mathbf{T}$ of $\Lambda_q^{\perp}(\boldsymbol{A})$ such that $\left\|\tilde{\boldsymbol{T}}\right\| \leq O(\sqrt{n \log q})$. Let the trapdoor function $f_{\boldsymbol{A}}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} \bmod q$. The public key is $pk = \boldsymbol{A}$, the secret key is $sk = \boldsymbol{T}$.

Re-Signature Key Generation: On input public keys of user A and B, $pk_A = \boldsymbol{A}$, $pk_B = \boldsymbol{B}$ and a secret key $sk_B = \boldsymbol{T}_B$. Let $\boldsymbol{A} = (\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_m)^T$, where $\boldsymbol{a}_i \in \mathbb{Z}_q^n$. For every $\boldsymbol{a}_i$, $i = 1, 2, \cdots, m$, use preimage sampleable algorithm $SamplePre(\boldsymbol{B}, \boldsymbol{T}_B, \boldsymbol{a}_i, \sigma)$ which samples a vector $\boldsymbol{s}_i$ such that $\boldsymbol{B}\boldsymbol{s}_i = \boldsymbol{a}_i \bmod q$ and $\|\boldsymbol{s}_i\| \leq \sigma\sqrt{m}$. Let $\boldsymbol{S}_{A \to B} = (\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_m) \in \mathbb{Z}^{m \times m}$, then $\boldsymbol{B}\boldsymbol{S}_{A \to B} = \boldsymbol{A} \bmod q$ and $\|\boldsymbol{S}_{A \to B}\| \leq \sigma\sqrt{m}$. Output the re-signature key $rk_{A \to B} = \boldsymbol{S}_{A \to B}$.

Sign: The first-level signature: on input a secret key $sk = \boldsymbol{T}$ and a message μ, do:

1. Choose a random vector $r \in \{0,1\}^*$ and compute $\boldsymbol{u} = H(\mu \,||\, r) \in \mathbb{Z}_q^n$;
2. Use preimage sampleable algorithm $SamplePre(\boldsymbol{A}, \boldsymbol{T}, \boldsymbol{u}, \sigma)$ samples a vector $\boldsymbol{e}$ such that $\boldsymbol{A}\boldsymbol{e} = \boldsymbol{u} \bmod q$ and $\|\boldsymbol{e}\| \leq s\sqrt{m}$.
3. Output $(\boldsymbol{e}, r)$ as the signature for message μ.

The i-level signature: on input a secret key $sk = \boldsymbol{T}$ and a message μ, do:

4. Choose a random vector $r \in \{0,1\}^*$ and compute $\boldsymbol{u} = H(\mu \,||\, r) \in \mathbb{Z}_q^n$;
5. Use preimage sampleable algorithm $SamplePre(\boldsymbol{A}, \boldsymbol{T}, \boldsymbol{u}, \sigma^i m^{(i-1)/2})$ to sample a vector $\boldsymbol{e}$ such that $\boldsymbol{A}\boldsymbol{e} = \boldsymbol{u} \bmod q$ and $\|\boldsymbol{e}\| \leq \sigma^i m^{i/2}$.
6. Output $(\boldsymbol{e}, r)$ as the signature for message μ.

Re-Signature: On input re-signature key $rk_{A \to B} = \boldsymbol{S}_{A \to B}$, a public key $pk_A = \boldsymbol{A}$, a message μ and a first-level signature $(\boldsymbol{e}_A, r)$, check that $\boldsymbol{A}\boldsymbol{e}_A = \boldsymbol{u} \bmod q$ and $\|\boldsymbol{e}_A\| \leq \sigma\sqrt{m}$. If $\boldsymbol{e}_A$ is not a signature for μ, output $\perp$; otherwise compute re-signature $\boldsymbol{e}_B = \boldsymbol{S}_{A \to B}\boldsymbol{e}_A$. $(\boldsymbol{e}_B, r)$ is the re-signature for $A \to B$.

The algorithm ReSign can transform an l-level signature into $(l + 1)$-level signature as first-level re-signature.

Verify: On input a public key $pk_B = \boldsymbol{B}$, a message μ and a re-signature $(\boldsymbol{e}_B, r)$ for $A \to B$. If $\boldsymbol{B}\boldsymbol{e}_B = \boldsymbol{u} \bmod q$ and $\|\boldsymbol{e}_B\| \leq \sigma^2 m$, output 1; otherwise output 0.

4.2 Security and Good Properties

Theorem 1 (Multi-use). The scheme is multi-use correct.

Proof: Consider the users $1, \cdots, k$. Suppose $(\boldsymbol{e}_1, r)$ is a valid signature of user 1, i.e., $\boldsymbol{A}_1\boldsymbol{e}_1 = H(\mu \,||\, r) \bmod q$ and $\|\boldsymbol{e}_1\| \leq \sigma\sqrt{m}$. Re-signature procedure is performed from 1 to k through 2 to $k-1$. The re-signature procedure is as follows:

$$\begin{aligned} \boldsymbol{e}_k &= \boldsymbol{S}_{k-1\to k}\boldsymbol{e}_{k-1} = \boldsymbol{S}_{k-1\to k}\boldsymbol{S}_{k-2\to k-1}\boldsymbol{e}_{k-2} \\ &= \cdots = \boldsymbol{S}_{k-1\to k}\boldsymbol{S}_{k-2\to k-1}\cdots\boldsymbol{S}_{2\to 1}\boldsymbol{e}_1 \end{aligned} \tag{10}$$

The verification procedure by the public key $\boldsymbol{A}_k$ of user k is as follows:

$$\begin{aligned} \boldsymbol{A}_k\boldsymbol{e}_k &= \boldsymbol{A}_k\boldsymbol{S}_{k-1\to k}\boldsymbol{S}_{k-2\to k-1}\cdots\boldsymbol{S}_{2\to 1}\boldsymbol{e}_1 \\ &= \boldsymbol{A}_{k-1}\boldsymbol{S}_{k-2\to k-1}\cdots\boldsymbol{S}_{2\to 1}\boldsymbol{e}_1 \\ &= \boldsymbol{A}_1\boldsymbol{e}_1 \\ &= \boldsymbol{u} \bmod q \end{aligned} \tag{11}$$

and

$$\begin{aligned} \|\boldsymbol{e}_k\| &= \|\boldsymbol{S}_{k-1\to k}\boldsymbol{S}_{k-2\to k-1}\cdots\boldsymbol{S}_{2\to 1}\boldsymbol{e}_1\| \\ &\leq \|\boldsymbol{S}_{k-1\to k}\|\cdots\|\boldsymbol{S}_{2\to 1}\|\|\boldsymbol{e}_1\| \\ &\leq \sigma^k m^{k/2} \end{aligned} \tag{12}$$

Therefore, the scheme is multi-use correct.

In the following, we analyze the other properties.

Theorem 2. In a random oracle model, the scheme is secure under the $\mathrm{SIS}_{q,n,m,\beta}$ problem, more precisely, given a random rank n matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, if finding a non-zero vector $\boldsymbol{v}$ such that $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{v}\| \leq \beta$ is hard, then the scheme is secure.

Proof: We argue security in two parts, i.e., the external security and the internal security.

External Security: For security, we assume there is a probability poly-time adversary $\mathcal{A}$ which breaks this guarantee with non-negligible probability ε after making at most q_H hash queries, q_s signature queries and q_{rs} re-signature queries. We use $\mathcal{A}$ to construct a poly-time simulator $\mathcal{B}$ that solves the $\mathrm{SIS}_{q,n,m,\beta}$ problem.

System Parameters: On input a random matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, the simulator $\mathcal{B}$ outputs a non-zero vector $\boldsymbol{v}$ such that $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{v}\| \leq \beta$.

Public keys: When $\mathcal{A}$ asks for the creation of user $i \in \{1, \cdots, \kappa\}$, $\mathcal{B}$ needs to prepare κ public keys $\boldsymbol{A}_1, \cdots, \boldsymbol{A}_\kappa$. The procedure is as follows:

(i) Let $\boldsymbol{A} = \boldsymbol{A}_t$. $\mathcal{B}$ uses the algorithm $SampleS(1^m)$ to sample $t-1$ matrices $\boldsymbol{S}_{t-1\to t}, \cdots, \boldsymbol{S}_{1\to 2}$ and computes $\boldsymbol{A}_{t-1} = \boldsymbol{A}_t \boldsymbol{S}_{t-1\to t} \bmod q, \cdots, \boldsymbol{A}_1 = \boldsymbol{A}_2 \boldsymbol{S}_{1\to 2} \bmod q$.
(ii) $\mathcal{B}$ uses $TrapGen(1^n)$ to generate $\kappa - t$ public/secret key pairs $(\boldsymbol{A}_i, \boldsymbol{T}_i)$, $i = t+1, \cdots, \kappa$.

In the following, $\mathcal{B}$ must answer the random oracle H, the signature oracle $\mathcal{O}_{sign}$ and the re-signature oracle $\mathcal{O}_{resign}$. $\mathcal{B}$ simulates these oracles as follows:

Hash queries: $\mathcal{B}$ maintains a list of tuples $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ which is called the H list. For each query to H, if (μ_k, r_k) is in the H list, then $\mathcal{B}$ returns $\boldsymbol{u}_k$ to $\mathcal{A}$. Otherwise, if $i > t$, compute $\boldsymbol{u}_k = H(\mu_k || r_k)$ and use the secret key $\boldsymbol{T}_i$ to sample a vector $\boldsymbol{e}_k \leftarrow SamplePre(\boldsymbol{A}_i, \boldsymbol{T}_i, \boldsymbol{u}_k, \sigma_i)$, store $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ and return $\boldsymbol{u}_k$ to $\mathcal{A}$. If $i \le t$, sample $\boldsymbol{e}_k \leftarrow D_{\mathbb{Z}^m, s_i}$ and compute $\boldsymbol{u}_k = \boldsymbol{A}_i \boldsymbol{e}_k \bmod q$, store $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ and return $\boldsymbol{u}_k$ to $\mathcal{A}$.

Signature queries: For each query to $\mathcal{O}_{sign}$ on input $(i, (\mu_k, r_k))$. We assume that μ_k has already been queried on the random oracle H. $\mathcal{B}$ looks up $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ in the H list and returns $\boldsymbol{e}_k$ to $\mathcal{A}$.

Re-Signature queries: For each query to $\mathcal{O}_{resign}$ on input $(i, j, (\mu_k, r_k), \boldsymbol{e}_k)$, if $j > t$, compute re-signature key $rk_{i\to j} = \boldsymbol{S}_{i\to j}$ by the *Re-Signature key generation algorithm* and compute $\boldsymbol{e'}_k = \boldsymbol{S}_{i\to j} \boldsymbol{e}_k$, and then return $\boldsymbol{e'}_k$ to $\mathcal{A}$. Otherwise, if $j \le t$, compute $rk_{i\to j} = \boldsymbol{S}_{i\to j} = \boldsymbol{S}_{j-1\to j} \cdots \boldsymbol{S}_{i\to i+1}$ and $\boldsymbol{e'}_k = \boldsymbol{S}_{i\to j} \boldsymbol{e}_k$, and then return $\boldsymbol{e'}_k$ to $\mathcal{A}$.

Forgery: Without loss of generality, we assume that $\mathcal{A}$ selects $\boldsymbol{A}_t$ as the challenge public key (the probability is $1/\kappa$) before outputting its forgery $((\mu^*, r^*), \boldsymbol{e}^*)$ and querying H on μ^*. Finally, $\mathcal{A}$ outputs forgery $((\mu^*, r^*), \boldsymbol{e}^*)$.

We now analyze the simulation. First, for each distinct query (μ, r) to H, the value $\boldsymbol{u}$ returned by $\mathcal{B}$ is $\boldsymbol{u} = f_{\boldsymbol{A}}(\boldsymbol{e}) = \boldsymbol{A}\boldsymbol{e} \bmod q$, where $\boldsymbol{e} \leftarrow D_{\mathbb{Z}^m, s}$. Because the distribution of $\boldsymbol{u}$ is uniform, it is identical to the uniformly random value of $H(\mu \,||\, r)$ in the real system. Second, for each query (μ, r) to $\mathcal{O}_{sign}$, $\mathcal{B}$ returns a single value $\boldsymbol{e} \leftarrow D_{\mathbb{Z}^m, s}$ such that $f_{\boldsymbol{A}}(\boldsymbol{e}) = H(\mu \,||\, r)$. In the real system, signature queries on μ are answered by a single value with the same distribution by the algorithm *SamplePre*. Third, for each query to $\mathcal{O}_{resign}$, we know that the re-signature key in $\mathcal{O}_{resign}$ queries is indistinguishable from that in the real system, so the $\mathcal{O}_{resign}$ queries is statistically close to the view of the real system. Thus we claim that the simulation of $\mathcal{B}$ is identical to the real system.

When $\mathcal{A}$ outputs forgery $((\mu^*, r^*), \boldsymbol{e}^*)$, $\mathcal{B}$ looks up $\left((\mu^*, r^*), \boldsymbol{e}_{\mu^*}\right)$ in the H list and outputs $\boldsymbol{v} = \boldsymbol{e}_{\mu^*} - \boldsymbol{e}^*$ as the solution of the $\mathrm{SIS}_{q,n,m,\beta}$ problem $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$. Because $((\mu^*, r^*), \boldsymbol{e}^*)$ and $\left((\mu^*, r^*), \boldsymbol{e}_{\mu^*}\right)$ are both the signatures of μ^*, then

$$\boldsymbol{A}_t \boldsymbol{e}^* \bmod q = H(\mu^* || r^*) \bmod q = \boldsymbol{A}_t \boldsymbol{e}_{\mu^*} \bmod q \tag{13}$$

Therefore, we obtain $\boldsymbol{A}_t(\boldsymbol{e}^* - \boldsymbol{e}_{\mu^*}) = \boldsymbol{0} \bmod q$. Since $\|\boldsymbol{e}^*\|$, $\|\boldsymbol{e}_{\mu^*}\| \leq \sigma\sqrt{m}$ and $\boldsymbol{e}^* \neq \boldsymbol{e}_{\mu^*}$, we have $\|\boldsymbol{e}^* - \boldsymbol{e}_{\mu^*}\| \leq 2\sigma\sqrt{m}$ and $\boldsymbol{e}^* - \boldsymbol{e}_{m^*} \neq \boldsymbol{0}$.

Internal Security: In this scheme, since the first-level signatures belong to the second-level signatures, the colluding delegatee and proxy can produce a first-level signature on delegator's behalf. Thus, the delegator security in our scheme is not satisfied. Internal security refers only to the limited proxy security and delegatee security.

Limited Proxy Security: For security, we assume there is a probability poly-time adversary (proxy) $\mathcal{A}$ which breaks this guarantee with non-negligible probability. We use $\mathcal{A}$ to construct a poly-time simulator $\mathcal{B}$ that solves the $\mathrm{SIS}_{q,n,m,\beta}$ problem.

System Parameters: On input a random matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, the simulator $\mathcal{B}$ outputs a non-zero vector $\boldsymbol{v}$ such that $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{v}\| \leq \beta$.

Public keys: When $\mathcal{A}$ asks for the creation of user $i \in \{1, \cdots, \kappa\}$, $\mathcal{B}$ needs to prepare κ public keys $\boldsymbol{A}_1, \cdots, \boldsymbol{A}_\kappa$. The procedure is as follows:

(i) $\mathcal{B}$ sets $\boldsymbol{A} = \boldsymbol{A}_t$.
(ii) $\mathcal{B}$ uses $TrapGen(1^n)$ to generate $\kappa - 1$ pairs of public/secret keys $(\boldsymbol{A}_i, \boldsymbol{T}_i)$, $i = 1, \cdots, t-1, \cdots, t+1, \cdots \kappa$.

In the following, $\mathcal{B}$ must answer the random oracle H, the signature oracle $\mathcal{O}_{sign}$ and the re-signature key oracle $\mathcal{O}_{rk}$. $\mathcal{B}$ simulates these oracles as follows:

Hash queries: $\mathcal{B}$ maintains a list of tuples $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ which is called the H list. for each query to H, if (μ_k, r_k) is in the H list, then $\mathcal{B}$ returns $\boldsymbol{u}_k$ to $\mathcal{A}$. Otherwise, if $i \neq t$, choose a random vector $r_k \in \{0,1\}^*$, compute $\boldsymbol{u}_k = H(\mu_k \,||\, r_k)$ and use the secret key $\boldsymbol{T}_i$ to sample a vector $\boldsymbol{e}_k \leftarrow SamplePre(\boldsymbol{A}_i, \boldsymbol{T}_i, \boldsymbol{u}_k, \sigma_i)$, store $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ and return $\boldsymbol{u}_k$ to $\mathcal{A}$. If $i = t$, sample $\boldsymbol{e}_k \leftarrow D_{\mathbb{Z}^m, s}$ and compute $\boldsymbol{u}_k = \boldsymbol{A}_i\boldsymbol{e}_k \bmod q$, store $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ and return $\boldsymbol{u}_k$ to $\mathcal{A}$.

Signature queries: For each query to $\mathcal{O}_{sign}$ on input $(i, (\mu_k, r_k))$. We assume that μ_k has already been queried on the random oracle H. $\mathcal{B}$ looks up $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ in the H list and returns $\boldsymbol{e}_k$ to $\mathcal{A}$.

Re-Signature key queries: For each query to $\mathcal{O}_{rk}$ on input (i, j) , if $i = t$ or $j = t$, abort; otherwise, compute re-signature key $rk_{i\to j} = \boldsymbol{S}_{i\to j}$ by the *Re-Signature key generation algorithm* and return $rk_{i\to j} = \boldsymbol{S}_{i\to j}$ to $\mathcal{A}$.

Forgery: Without loss of generality, we assume that $\mathcal{A}$ selects $\boldsymbol{A}_t$ as the challenge public key (the probability is $1/\kappa$) before outputting its forgery $((\mu^*, r^*), \boldsymbol{e}^*)$ and querying H on μ^*. Finally, $\mathcal{A}$ outputs forgery $((\mu^*, r^*), \boldsymbol{e}^*)$.

Simulator $\mathcal{B}$'s simulation of the world for $\mathcal{A}$ is the same as the external security except that the Re-Signature queries is replaced by the Re-Signature key queries.

Delegatee security: For security, we assume there is a probability poly-time adversary (proxy) $\mathcal{A}$ which breaks this guarantee with non-negligible probability. We use $\mathcal{A}$ to construct a poly-time simulator $\mathcal{B}$ that solves the $\mathrm{SIS}_{q,n,m,\beta}$ problem.

System Parameters: On input a random matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, the simulator $\mathcal{B}$ outputs a non-zero vector $\boldsymbol{v}$ such that $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{v}\| \leq \beta$.

Public keys: When $\mathcal{A}$ asks for the creation of user $i \in \{1, \cdots, \kappa\}$, $\mathcal{B}$ needs to prepare κ public keys $\boldsymbol{A}_1, \cdots, \boldsymbol{A}_\kappa$. The procedure is as follows:

(i) $\mathcal{B}$ sets $\boldsymbol{A} = \boldsymbol{A}_1$.
(ii) $\mathcal{B}$ uses $TrapGen(1^n)$ to generate $k-1$ pairs of public/secret keys $(\boldsymbol{A}_i, \boldsymbol{T}_i)$, $i = 2, \cdots, \kappa$.

In the following, $\mathcal{B}$ must answer the random oracle H, the signature oracle $\mathcal{O}_{sign}$ and the re-signature key oracle $\mathcal{O}_{rk}$. $\mathcal{B}$ simulates these oracles as follows:

Hash queries: $\mathcal{B}$ maintains a list of tuples $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ which is called the H list. for each query to H, if μ_k is in the H list, $\mathcal{B}$ returns $\boldsymbol{u}_k$ to $\mathcal{A}$. Otherwise, if $i \neq 1$, choose a random vector $r_k \in \{0,1\}^*$, compute $\boldsymbol{u}_k = H(\mu_k \,||\, r_k)$ and use the secret key $\boldsymbol{T}_i$ to sample a vector $\boldsymbol{e}_k \leftarrow SamplePre(\boldsymbol{A}_i, \boldsymbol{T}_i, \boldsymbol{u}_k, \sigma_i)$, store $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ and return $\boldsymbol{u}_k$ to $\mathcal{A}$. If $i = 1$, sample $\boldsymbol{e}_k \leftarrow D_{\mathbb{Z}^m, s}$ and compute $\boldsymbol{u}_k = \boldsymbol{A}_1 \boldsymbol{e}_k \bmod q$, store $(1, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ and return $\boldsymbol{u}_k$ to $\mathcal{A}$.

Signature queries: For each query to $\mathcal{O}_{sign}$ on input $(i, (\mu_k, r_k))$. We assume that μ_k has already been queried on the random oracle H. $\mathcal{B}$ looks up $(i, \boldsymbol{u}_k, \boldsymbol{e}_k, (\mu_k, r_k))$ in the H list and returns $\boldsymbol{e}_k$ to $\mathcal{A}$.

Re-Signature key queries: For each query to $\mathcal{O}_{rk}$ on input (i,j), if $i = 1$, abort; otherwise, compute re-signature key $rk_{i \to j} = \boldsymbol{S}_{i \to j}$ by the *Re-Signature key generation algorithm* and return $rk_{i \to j} = \boldsymbol{S}_{i \to j}$ to $\mathcal{A}$.

Forgery: Without loss of generality, we assume that $\mathcal{A}$ selects $\boldsymbol{A}_t$ as the challenge public key (the probability is $1/\kappa$) before outputting its forgery $((\mu^*, r^*), \boldsymbol{e}^*)$ and querying H on μ^*. Finally, $\mathcal{A}$ outputs forgery $((\mu^*, r^*), \boldsymbol{e}^*)$.

We know that the simulation is perfect. When $\mathcal{A}$ outputs forgery $((\mu^*, r^*), \boldsymbol{e}^*)$, $\mathcal{B}$ looks up $\left((\mu^*, r^*), \boldsymbol{e}_{\mu^*}\right)$ in the H list and outputs $\boldsymbol{v} = \boldsymbol{e}_{\mu^*} - \boldsymbol{e}^*$ as the solution of the $\mathrm{SIS}_{q,n,m,\beta}$ problem $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0} \bmod q$. Because $((\mu^*, r^*), \boldsymbol{e}^*)$ and $\left((\mu^*, r^*), \boldsymbol{e}_{\mu^*}\right)$ are both the signatures of μ^*, then

$$\boldsymbol{A}_1 \boldsymbol{e}^* \bmod q = H(\mu^* || r^*) \bmod q = \boldsymbol{A}_1 \boldsymbol{e}_{\mu^*} \bmod q \tag{14}$$

Therefore, we obtain $\boldsymbol{A}_1 \left(\boldsymbol{e}^* - \boldsymbol{e}_{\mu^*}\right) = \boldsymbol{0} \bmod q$. Since $\|\boldsymbol{e}^*\|$, $\left\|\boldsymbol{e}_{\mu^*}\right\| \leq \sigma\sqrt{m}$ and $\boldsymbol{e}^* \neq \boldsymbol{e}_{\mu^*}$, we have $\left\|\boldsymbol{e}^* - \boldsymbol{e}_{\mu^*}\right\| \leq 2\sigma\sqrt{m}$ and $\boldsymbol{e}^* - \boldsymbol{e}_{\mu^*} \neq \boldsymbol{0}$.

4.3 Security and Efficiency Comparison

In this section, we compare the security and efficiency of the proposed scheme with that of the scheme of [3] which is the first multi-use unidirectional proxy re-signature scheme. The scheme needs 6 pair operations in the verification of 1-level signature, and 4L + 2 pair operations in the verification of L-level signature. The proposed

construction is based on the Small Integer Solution problem. The verification cost does not grow with the number of translations (only one matrix-vector product operation in any level signature) and the size of signatures also grows linearly with the number of translations. The comparison results are summarized in Table 1.

Table 1. Security and efficiency comparison

Cryptosystem	Underlying problem	The size of signature	Verification cost
The scheme of [3]	*l*-FlexDH assumption	Grows linearly with the number of translations	Grows linearly with the number of translations
The proposed scheme	SIS problem	Grows linearly with the number of translations	Not change with the number of translations

5 Multi-use Unidirectional Proxy Re-Signature Scheme from Lattice in the Standard Model

In this section, we use the signature scheme of [15] to construct a multi-use unidirectional proxy re-signature scheme in the standard model.

KeyGen: On input the security parameter n, run $TrapGen(q,n)$ to generate a random rank n matrix $\boldsymbol{A}_0 \in \mathbb{Z}_q^{n\times m}$ and a trapdoor basis $\mathbf{T}_0$ of $\Lambda_q^{\perp}(\boldsymbol{A}_0)$ such that $\left\|\tilde{\boldsymbol{T}}_0\right\| \le O(\sqrt{n\log q})$.

For each $(b,j) \in \{0,1\} \times [k]$, choose uniformly random and independent $\mathbf{A}_j^{(b)} \in \mathbb{Z}_q^{n\times m}$. Output public key $pk = (\boldsymbol{A}_0, \boldsymbol{A}_j^{(b)})$ and secret key $sk = \boldsymbol{T}_0$.

Re-Signature Key Generation: On input public keys of user 1 and 2, $pk_1 = (\boldsymbol{A}_{10}, \boldsymbol{A}_j^{(b)})$, $pk_2 = (\boldsymbol{A}_{20}, \boldsymbol{A}_j^{(b)})$ and a secret key $sk_2 = \boldsymbol{T}_2$. Let $\boldsymbol{A}_{10} = (\boldsymbol{a}_{11}, \boldsymbol{a}_{12}, \cdots, \boldsymbol{a}_{1m})^T$, where $\boldsymbol{a}_{1i} \in \mathbb{Z}_q^n$. For every $\boldsymbol{a}_{1i}$, $i = 1, 2, \cdots, m$, use preimage sampleable algorithm $SamplePre(\boldsymbol{A}_{20}, \boldsymbol{T}_2, \boldsymbol{a}_{1i}, \sigma)$ which samples a vector $\boldsymbol{s}_i$ such that $\boldsymbol{A}_{20}\boldsymbol{s}_i = \boldsymbol{a}_{1i} \bmod q$ and $\|\boldsymbol{s}_i\| \le \sigma\sqrt{m}$. Let $\boldsymbol{S} = (\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_m) \in \mathbb{Z}^{m\times m}$, then $\boldsymbol{A}_{20}\boldsymbol{S} = \boldsymbol{A}_{10} \bmod q$ and $\|\boldsymbol{S}\| \le s\sqrt{m}$. Let $\boldsymbol{S}_{1\to 2} = \begin{pmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix}$ and output the re-signature key $rk_{1\to 2} = \boldsymbol{S}_{1\to 2}$.

Sign: The first-level signature: on input a secret key $sk = \boldsymbol{T}_0$ and a message $\mu \in \{0,1\}^k$, do:

1. Let $\boldsymbol{A}_\mu = \boldsymbol{A}_0 || \boldsymbol{A}_1^{(\mu_1)} || \cdots || \boldsymbol{A}_k^{(\mu_k)} \in \mathbb{Z}_q^{n\times(k+1)m}$. Use $SampleBasisLeft(\boldsymbol{A}_0, \boldsymbol{A}_i^{(\mu_i)}, \boldsymbol{T}_0)$ to generate the basis $\boldsymbol{T}_\mu$ of $\Lambda^{\perp}(\boldsymbol{A}_\mu)$;
2. Use preimage sampleable algorithm $SamplePre(\boldsymbol{A}_\mu, \boldsymbol{T}_\mu, \boldsymbol{0}, \sigma)$ to sample a vector $\boldsymbol{e}$ such that $\boldsymbol{A}_\mu \boldsymbol{e} = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{e}\| \le \sigma\sqrt{(k+1)m}$.
3. Output $\boldsymbol{e}$ as the signature for message μ.

The i-level signature: on input a secret key $sk = \boldsymbol{T}_0$ and a message μ, do:

1. Let $\boldsymbol{A}_\mu = \boldsymbol{A}_0 \,||\, \boldsymbol{A}_1^{(\mu_1)} \,||\cdots||\, \boldsymbol{A}_k^{(\mu_k)} \in \mathbb{Z}_q^{n\times(k+1)m}$. Use $SampleBasisLeft(\boldsymbol{A}_0, \boldsymbol{A}_i^{(\mu_i)}, \boldsymbol{T}_0)$ to generate the basis $\boldsymbol{T}_\mu$ of $\Lambda^{\perp}(\boldsymbol{A}_\mu)$;
2. Use preimage sampleable algorithm $SamplePre(\boldsymbol{A}_\mu, \boldsymbol{T}_\mu, \boldsymbol{0}, \sigma^i[(k+1)m]^{(i-1)/2})$ to sample a vector $\boldsymbol{e}$ such that $\boldsymbol{A}_\mu \boldsymbol{e} = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{e}\| \le \sigma^i[(k+1)m]^{i/2}$.
3. Output $\boldsymbol{e}$ as the i-level signature for message μ.

Re-Signature: On input re-signature key $rk_{1\to 2} = \boldsymbol{S}_{1\to 2}$, a public key $pk_1 = (\boldsymbol{A}_{10}, \boldsymbol{A}_j^{(b)})$, a message μ and its signature $\boldsymbol{e}_1$, check that $\boldsymbol{A}_{1\mu}\boldsymbol{e}_1 = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{e}_1\| \le s\sqrt{(k+1)m}$, where $\boldsymbol{A}_{1\mu} = \boldsymbol{A}_{10} \,||\boldsymbol{A}_1^{(\mu_1)} \,||\cdots||\, \boldsymbol{A}_k^{(\mu_k)} \in \mathbb{Z}_q^{n\times(k+1)m}$. If $\boldsymbol{e}_1$ is not a signature for μ, output $\perp$; otherwise compute re-signature $\boldsymbol{e}_2 = \boldsymbol{S}_{1\to 2}\boldsymbol{e}_1$. $\boldsymbol{e}_2$ is the re-signature for $1 \to 2$.

Verify: On input a public key $pk_2 = (\boldsymbol{A}_{20}, \boldsymbol{A}_j^{(b)})$, a message μ and a re-signature $\boldsymbol{e}_2$ for $1 \to 2$. If $\boldsymbol{A}_{2\mu}\boldsymbol{e}_2 = \boldsymbol{0} \bmod q$ and $\|\boldsymbol{e}_2\| \le \sigma^2(k+1)m$, where $\boldsymbol{A}_{2\mu} = \boldsymbol{A}_{20} \,||\boldsymbol{A}_1^{(\mu_1)} \,||\cdots||\, \boldsymbol{A}_k^{(\mu_k)} \in \mathbb{Z}_q^{n\times(k+1)m}$, output 1; otherwise output 0.

6 Conclusion

In this paper, we construct the first multi-use unidirectional proxy re-signature scheme based on the hardness of the Small Integer Solution (SIS) problem. In our scheme, the verification cost does not grow with the number of translations which only needs a matrix-vector multiplication. The size of signatures grows linearly with the number of the translations in this scheme. Our scheme only uses one signature algorithm such that the user's i-level signatures contain $(i - 1)$-level signatures, however it does not resist the collusion attack of delegator security.

Acknowledgments. We are thankful to anonymous referees for their helpful comments. This paper is supported by the National Natural Science Foundation of China under Grant No. 61902 140, No. 60573026, the Anhui Provincial Natural Science Foundation under Grant No. 1708085 QF154, No. 1908085QF288, NO. 1808085QF181, the Nature Science Foundation of Anhui Higher Education Institutions under Grant No. KJ2019A0605, No. KJ2018A0398, No. KJ2019 B018.

References

1. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054122

2. Ateniese, G., Hohenberger, S.: Proxy re-signatures: new definitions, algorithms, and applications. In: CCS Proceedings of the 15th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, March 2005, pp. 310–319 (2005). https://doi.org/10.1145/1102120.1102161
3. Libert, B., Vergnaud, D.: Multi-use unidirectional proxy re-signatures. In: CCS Proceedings of the 15th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, October 2008, pp. 511–520 (2008)
4. Sunitha, N.R., Amberker, B.B.: Multi-use unidirectional forward-secure proxy re-signature scheme. In: Proceedings of the 3rd IEEE International Conference on Internet Multimedia Services Architecture and Applications, Bangalore, India, December 2009, pp. 223–228 (2009)
5. Sunitha, N.R.: Proxy re-signature schemes: multi-use, unidirectional and translations. J. Adv. Inf. Technol. **2**(3), 165–176 (2011)
6. Shao, J., Cao, Z., Wang, L., Liang, X.: Proxy re-signature schemes without random oracles. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 197–209. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77026-8_15
7. Shao, J., Wei, G.Y., Ling, Y., Xie, M.D.: Unidirectional identity-based proxy re-signature. In: Proceedings of the IEEE Communications Society, Hangzhou, China, June 2011, pp. 1–5 (2011)
8. Shao, J., Feng, M., Zhu, B., Cao, Z., Liu, P.: The security model of unidirectional proxy re-signature with private re-signature key. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 216–232. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14081-5_14
9. Yang, P.Y., Cao, Z.F., Dong, X.L.: Threshold proxy re-signature. J. Syst. Sci. Complex **2011**(24), 816–824 (2011)
10. Gentry, C., Peikert, C., Vaikuntanathan, V.: How to use a short basis: trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the STOC 2008, Victoria, Canada, May 2008, pp. 197–206 (2008)
11. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), Article 34 (2009)
12. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1
13. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_21
14. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, Kenneth G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_4
15. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_27
16. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
17. Agrawal, S., Boneh, D., Boyen, X.: Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_6
18. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the STOC 2009, Bethesda, Maryland, USA, May 2009, pp. 169–178 (2009)

19. Gentry, C.: Toward basing fully homomorphic encryption on worst-case hardness. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 116–137. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_7
20. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_29
21. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: Proceedings of the FOCS 2011, Palm Springs, CA, USA, October 2011, pp. 97–106 (2011)
22. Lyubashevsky, V.: lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_43
23. Gordon, S.D., Katz, J., Vaikuntanathan, V.: A group signature scheme from lattice assumptions. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 395–412. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_23
24. Rückert, M.: Lattice-based blind signatures. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 413–430. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_24
25. Rückert, M.: Strongly unforgeable signatures and hierarchical identity-based signatures from lattices without random oracles. In: Sendrier, N. (ed.) PQCrypto 2010. LNCS, vol. 6061, pp. 182–200. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12929-2_14
26. Alwen, J., Peiker, C.: Generating shorter bases for hard random lattices. In: Proceedings of the STACS 2009, Freiburg, Germany, February 2009, pp. 75–86 (2009)
27. Boneh, D., Freeman, D.M.: Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 1–16. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_1