A homomorphic signature scheme for quadratic polynomials

Seiko Arita
Institute of Information Security
Email: arita@iisec.ac.jp

Shunji Kozaki
Institute of Information Security
Email: kzksh2+BITS2017@gmail.com

***Abstract*—Homomorphic signatures can provide a credential of a result which is indeed computed with a given function on a data set by an untrusted third party like a cloud server, when the input data are stored with the signatures beforehand. Boneh and Freeman in EUROCRYPT2011 proposed a homomorphic signature scheme for polynomial functions of any degree, however the scheme is not based on the normal short integer solution (SIS) problems as its security assumption. In this paper, we show a homomorphic signature scheme for quadratic polynomial functions those security assumption is based on the normal SIS problems. Our scheme constructs the signatures of multiplication as tensor products of the original signature vectors of input data so that homomorphism holds. Moreover, security of our scheme is reduced to the hardness of the SIS problems respect to the moduli such that one modulus is the power of the other modulus. We show the reduction by constructing solvers of the SIS problems respect to either of the moduli from any forger of our scheme.**

I. Introduction

Homomorphic signatures can provide a credential of a result which is indeed computed with a given function over a certain data set, when the input data set is stored with signatures beforehand. Let a client store an initial data set $(x_1, \cdots, x_k)$ with its signatures $(\sigma_1, \cdots, \sigma_k)$ in a cloud server, and $f(X_1, \cdots, X_k)$ denotes a given function. Then homomorphic signature enables a server to output a signature $\sigma_{y,f}$ for a result $y = f(x_1, \cdots, x_k)$. Moreover, the signature can be publicly verified without the initial data. Homomorphic signature schemes are considered to be applicable to electronic voting, smart grids and electronic health records [1].

Boneh and Freeman [2] proposed a homomorphic signature scheme for linear functions. That is, their scheme is limited to the only case of $\deg f = 1$. Boneh and Freeman [3] also proposed a homomorphic signature scheme for polynomial functions of any degree. However, [3] was not based on normal short integer solution (SIS) problems as its security assumption. They assume SIS problems related with principal ideals over ideal lattices, which is unclear as the security assumption. Moreover, Gorbunov, Vaikuntanathan and Wichs [4] proposed a homomorphic signature scheme based on normal SIS problems as its security assumption. However the size of signatures for initial data, what we call the initial signatures hereafter, was large as $\tilde{O}(n^2)$ for the security parameter n. Our homomorphic signature scheme is based on normal SIS problems as its security assumption, which can be reduced to the worst-case lattice problems [5]. The size of the initial signatures of our scheme is double as [2], that is $\tilde{O}(n)$. Therefore, our scheme is efficient with respect to storage size for the initial data comparing to [4]. Our homomorphic scheme is applicable to the case up to $\deg f = 2$, however, it is worth enough for statistical computation in the cloud.

Our scheme is based on the linear homomorphic signature scheme in [3], which builds on the GPV signature scheme [6] using trapdoors for lattices. When x_1, x_2 are an initial data, then the initial signatures for x_1, x_2 are short vectors σ_1, σ_2 such that $A\sigma_1 \equiv \alpha$, $B\sigma_2 \equiv \beta$ for certain matrices A, B and given vectors α, β. Our scheme constructs a signature of multiplication x_1x_2 as a tensor product $\sigma_1 \otimes \sigma_2$, what we call the derived signature hereafter. Since tensor products have a property: $(A \otimes B)(\sigma_1 \otimes \sigma_2) = A\sigma_1 \otimes B\sigma_2$, we have $(A \otimes B)(\sigma_1 \otimes \sigma_2) \equiv \alpha \otimes \beta$ for the derived signature verification.

However, we cannot directly reduce a security of our scheme to solving the SIS problem as $(A \otimes B)\sigma \equiv 0$, because $A \otimes B$ is not an uniformly random matrix, whereas A, B are even uniformly random matrices. For that reason, we reduce the security of our scheme to solving the SIS problem with respect to A or B by decomposing the SIS problem with respect to $A \otimes B$ using two moduli such that one modulus is the power of the other modulus.

II. Preliminaries

A. Notations

Let $\mathbb{N}$, $\mathbb{Z}$ and $\mathbb{R}$ be positive integers, integers and real numbers respectively, and $\mathbb{Z}_q$ be a quotient ring with a modulus $q \in \mathbb{N}$. We denote $\mathrm{negl}(n) = 2^{-\omega(\log n)}$ a negligible function with respect to $n \in \mathbb{N}$.

For any $n \times m$ matrices A and B, the tensor product (or Kronecker product) of A and B denoted by $A \otimes B$ is an $n^2 \times m^2$ matrix defined by:

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1m}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \cdots & a_{nm}B \end{pmatrix}, \text{where } A = (a_{ij}).$$

For a $m \times m$ matrix V, let $\mathsf{vec}(V)$ denote a column vector of dimension m^2 formed by stacking each columns $\mathbf{v}_i$ of V:

$$\mathsf{vec}(V) := \begin{pmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_m \end{pmatrix}, \text{where } V = (\mathbf{v}_1 \cdots \mathbf{v}_m).$$

Then we will use the following fact of tensor products and matrix multiplications [7]: $(A \otimes B)\mathsf{vec}(V) = \mathsf{vec}\left(BVA^\top\right)$.

978-1-5090-6517-2/17/$31.00 ©2017 IEEE

In particular, $\mathbf{a} \otimes \mathbf{b} = \mathsf{vec}(\mathbf{b}\mathbf{a}^\top)$ for n-dimensional vectors $\mathbf{a}$, $\mathbf{b}$ when $m = 1$, $V = 1$. We define $\|V\| := \max_i \|\mathbf{v}_i\|$ for a matrix V whose columns are $\mathbf{v}_i$.

Let $\Lambda_q^\perp(A) := \{\mathbf{z} \in \mathbb{Z}^m | A\mathbf{z} = \mathbf{0} \bmod q\}$ be a m-dimensional lattice with a parity-check matrix $A \in \mathbb{Z}_q^{n\times m}$.

For a lattice $L \subset \mathbb{Z}^m$, a real $s > 0$ and a vector $\mathbf{c} \in \mathbb{R}^m$, the discrete Gaussian distribution over L is denoted by $D_{\mathbf{c}+L,s}(\mathbf{x}) := \frac{\exp(-\pi\|\mathbf{x}-\mathbf{c}\|^2/s^2)}{\sum_{\mathbf{v}\in L}\exp(-\pi\|\mathbf{v}-\mathbf{c}\|^2/s^2)}$ for all $\mathbf{x} \in L$.

B. SIS problems

Our scheme uses Short Integer Solution (SIS) problems [5] as security assumption. SIS problems are defined as follows.

Definition 1 ($\mathrm{SIS}_{n,q,\beta}$ problem)**.** Let $n, q, m \in \mathbb{N}$, $\beta \in \mathbb{R}_{>0}$. $\mathrm{SIS}_{n,q,\beta}$ problem is that for a uniformly random chosen $A \in \mathbb{Z}_q^{n\times m}$, find $\mathbf{z} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ such that $\|\mathbf{z}\| \le \beta$ and $A\mathbf{z} = \mathbf{0} \bmod q$.

If $m \ge 2n\log q$ and $\beta \ge \sqrt{m}$ hold, then $\mathrm{SIS}_{n,q,\beta}$ problem has a solution. Moreover, solving $\mathrm{SIS}_{n,q,\beta}$ problem for $\beta = \mathrm{poly}(n)$ and $q \ge \beta\cdot\omega(\sqrt{n\log n})$ is known at least hard as solving approximate Shortest Independent Vector Problem (SIVP), where the SIVP approximate factor is $\beta \cdot \tilde{O}(\sqrt{n})$[6, Proposition 5.7]. Therefore, $\mathrm{SIS}_{n,q,\beta}$ problems are considered as infeasible.

C. Algorithms for generating lattices with trapdoors and sampling short vectors from the lattices

We explain an algorithm $\mathsf{TrapGen}$ for generating parity-check lattices with trapdoors and an algorithm $\mathsf{SamplePre}$ for sampling short vectors from these lattices. These algorithms are used in our scheme as sub algorithms.

Micciancio and Peikert [8, Theorem 5.1] provide an algorithm to output an uniformly random parity-check matrix $A \in \mathbb{Z}_q^{n\times m}$ and a trapdoor R of the lattice $\Lambda_q^\perp(A)$ for given $n, q, m \in \mathbb{N}$ such that $q \ge 2$ and $m = O(n\log q)$. Moreover, they provide an algorithm to output $\mathbf{z} \in \mathbb{Z}^m$ such that $A\mathbf{z} \equiv \alpha \pmod q$, $\|\mathbf{z}\| = O(n\log q)$ and its output distribution is Gaussian over a coset of $\Lambda_q^\perp(A)$ with the deviation $s\cdot\omega(\sqrt{\log n})$ for given $\alpha \in \mathbb{Z}_q^n$, $s = O\left(\sqrt{n\log q}\right)$.

By using these algorithms, we can obtain an algorithm $\mathsf{TrapGen}(n, q, m)$ to output a matrix $A \in \mathbb{Z}_q^{n\times m}$ and a basis T of $\Lambda_q^\perp(A)$ such that $\|T\| = O(n\log q)$. Also, according to [6], we can obtain an algorithm $\mathsf{SamplePre}(\Lambda, T, \mathbf{t}, s)$ to output a vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\mathbf{z} \in \mathbf{t} + \Lambda$ for a given m-dimensional lattice Λ, its short basis T, $\mathbf{t} \in \mathbb{Z}^m$ and $s \ge \|T\|\cdot\omega(\sqrt{\log n})$, then the output satisfies $\|\mathbf{z}\| \le s\sqrt{m}$ with overwhelming probability.

D. Hensel lifting algorithm

We describe the algorithm $\mathsf{HenselLift}$ which is used in our scheme. Let $B \in \mathbb{Z}_q^{n\times m}$ be a parity-check matrix and $\sigma_1 \in \mathbb{Z}^m$ such that $B\sigma_1 \equiv \beta \pmod q$ for some β. Then the algorithm $\mathsf{HenselLift}$ outputs a vector $\sigma \in \mathbb{Z}^m$ such that $\sigma \equiv \sigma_1 \pmod q$ and $B\sigma_1 \equiv \beta \pmod{q^r}$ for a given $r \in \mathbb{N}$. Namely the algorithm $\mathsf{HenselLift}$ lifts up σ_1 in a coset of $\Lambda_q^\perp(B)$ to σ in a coset of $\Lambda_{q^r}^\perp(B)$, where B is considered in $\mathbb{Z}_{q^r}^{n\times m}$. The following algorithm 1 describes $\mathsf{HenselLift}$.

Algorithm 1 $\mathsf{HenselLift}(q, r, \beta, B, \sigma_1)$

IN: $q \in \mathbb{N}$: prime, $r \in \mathbb{N}$, $\beta \in \mathbb{Z}_q^n$, $B = [B_1|B_2] \in \mathbb{Z}_q^{n\times m}$ s.t. $^\exists B_1^{-1} \in \mathbb{Z}_q^{n\times n}$, $\sigma_1 \in \mathbb{Z}^m$ s.t. $B\sigma_1 \equiv \beta \pmod q$

OUT: $\sigma \in \mathbb{Z}^m$ s.t. $\sigma \equiv \sigma_1 \pmod q$, $B\sigma \equiv \beta \pmod{q^r}$

1: **for** $i = 2$ to r **do**
2: $\quad\mathbf{v} \leftarrow (\beta - B\sigma_{i-1})/q^{i-1} \in \mathbb{Z}^n$
$\quad/\star B\sigma_{i-1} \equiv \beta \pmod{q^{i-1}}\star/$
3: $\quad\mathbf{x}_2 \xleftarrow{\$} \mathbb{Z}_q^{m-n}$
4: $\quad\mathbf{x}_1 \leftarrow B_1^{-1}(\mathbf{v} - B_2\mathbf{x}_2) \bmod q$
5: $\quad\sigma_i \leftarrow \sigma_{i-1} + q^{i-1}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix}$ $/\star B\sigma_i \equiv \beta \pmod{q^i}\star/$
6: **return** σ_r

III. Our scheme

In the following, we explain the outline of our scheme.

Let $\mathbf{x}_i \in \mathbb{Z}^m$ be a vector whose first coefficient has each message x_i in the data set. Let p, q be primes s.t. $(p, q) = 1$ and $r \in \mathbb{N}$, and $A \in \mathbb{Z}_{q^r}^{n\times m}$, $B \in \mathbb{Z}_q^{n\times m}$ denote parity-check matrices. Let $\alpha_i \in \mathbb{Z}_{q^r}^n$, $\beta_i \in \mathbb{Z}_q^n$ be hash values for the identity function X_i of the message x_i.

By the Chinese remainder theorem and using the trapdoors as in [3], we can obtain a short vector $\sigma_{A,i} \in \mathbb{Z}^m$ such that $\sigma_{A,i} \equiv \mathbf{x}_i \pmod p$ and $A\sigma_{A,i} \equiv \alpha_i \pmod{q^r}$. Similarly, we can obtain a short vector $\sigma_{B,i} \in \mathbb{Z}^m$ such that $\sigma_{B,i} \equiv \mathbf{x}_i \pmod p$ and $B\sigma_{B,i} \equiv \beta_i \pmod{q^r}$. We set the initial signature for x_i as $(\sigma_{A,i}, \sigma_{B,i}) \in \mathbb{Z}^{2m}$.

We construct the derived signature σ of multiplication x_ix_j as a tensor product $\sigma = \sigma_{A,i} \otimes \sigma_{B,j} \in \mathbb{Z}^{m^2}$. Then the first coefficient of $\sigma \bmod p$ is equivalent to x_ix_j. Also, $(A \otimes B)\sigma \bmod q^r$ is equivalent to $\alpha_i \otimes \beta_j$. Hence we can verify the signature by these formulas and how short σ is.

The latter formula is considered as a verification with the parity-check matrix $A \otimes B$, however, $A \otimes B$ is not uniformly random in $\mathbb{Z}^{m^2\times n^2}$ although A and B are uniformly random in $\mathbb{Z}^{m\times n}$. Therefore, the security of our scheme cannot be simply reduced to the SIS problem with the parity-check matrices in $\mathbb{Z}^{m^2\times n^2}$.

We break down the problem into two SIS problems by using the following observation. Let $V := \sigma_{B,j}\sigma_{A,i}^\top \in \mathbb{Z}^{m\times m}$ and $W := (BV)^\top \in \mathbb{Z}^{m\times n}$, then, since $\mathsf{vec}(V) = \sigma_{A,i} \otimes \sigma_{B,j} = \sigma$, $BV = W^\top$ and $W^\top A^\top = (AW)^\top$, we have

$$(A \otimes B)\sigma = \mathsf{vec}\left(BVA^\top\right) = \mathsf{vec}\left((AW)^\top\right).$$

Therefore, if σ is a nonzero solution for $(A \otimes B)\sigma \equiv 0$, then the nonzero matrix V satisfies $BV \equiv O$ or the nonzero matrix W satisfies $AW \equiv O$. Since each of the column vectors of V is short comparing to q, if there exists the nonzero column vector then it is a solution for the SIS problem with respect to B. On the other hand, each of the column vectors of W has the relatively large size comparing to the modulus q of B. For this reason, by using the larger modulus q^r for A, we make the column vectors of W to be short relatively. Thus, we can

reduce the security of our scheme to solve the SIS problem with respect to B or A.

We show the construction of our scheme in the following. Let a security parameter $n \in \mathbb{N}$, a maximum data set size $k \in \mathbb{N}$, different primes p, q, positive integers $r, m \in \mathbb{N}$, Gaussian parameters $s_1, s_2 \in \mathbb{R}$ to be used in the signing, and parameters $\delta_1, \delta_2 \in \mathbb{R}$ for the verification. By using the sub algorithms TrapGen, SamplePre, HenselLift in the section II-C, II-D, we describe our homomorphic signature scheme $\mathcal{S}$ consisting of four algorithms $(\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Eval})$.

- $\mathsf{Setup}(1^n, k) \to (pk, sk)$.
  1) $x_{k+1} \xleftarrow{\cup} \mathbb{F}_p$: a value for a dummy variable,
     $H_0 : \{0,1\}^* \to \mathbb{F}_p$: hash function on tags and functions,
     $H_1 : \{0,1\}^* \to \mathbb{Z}_{q^r}^n$: hash function on data sets,
     $H_2 : \{0,1\}^* \to \mathbb{Z}_q^n$: hash function on data sets.
  2) $(A, T_A) \leftarrow \mathsf{TrapGen}(n, q^r, m)$,
     /*$A \in \mathbb{Z}_{q^r}^{n \times m}$, T_A : a short basis for $\Lambda_{q^r}^{\perp}(A)$*/
     $(B, T_B) \leftarrow \mathsf{TrapGen}(n, q, m)$,
     /*$B \in \mathbb{Z}_q^{n \times m}$, T_B : a short basis for $\Lambda_q^{\perp}(B)$*/
  3) return $pk = (A, B, x_{k+1}, H_0, H_1, H_2)$, $sk = (T_A, T_B)$.

The next algorithm is signing for an initial data set $(x_1, \cdots, x_k) \in (\mathbb{F}_p)^k$ tagged with $\tau \in \{0,1\}^n$.

- $\mathsf{Sign}\,(sk, pk, \tau, (x_1, \cdots, x_k)) \to (\sigma_1, \cdots, \sigma_{k+1})$.
  1) $\mathbf{x}_i := (x_i, 1, \ldots, 1)^\top \in \mathbb{F}_p^m$ for $i \in \{1, \ldots, k+1\}$
  2) for $i = 1, \cdots, k+1$:
     $\alpha_i \leftarrow H_1(\tau \| i)$,
     Compute $\mathbf{t} \in \mathbb{Z}^m$ s.t. $\mathbf{t} \equiv \mathbf{x}_i \pmod p$ and $A\mathbf{t} \equiv \alpha_i \pmod{q^r}$ using CRT,
     $\sigma_{A,i} \leftarrow \mathsf{SamplePre}(p\Lambda_{q^r}^{\perp}(A), pT_A, \mathbf{t}, s_1)$,
     /* $\sigma_{A,i} \equiv \mathbf{x}_i \pmod p \wedge A\sigma_{A,i} \equiv \alpha_i \pmod{q^r}$*/
     $\beta_i \leftarrow H_2(\tau \| i)$,
     Compute $\mathbf{t} \in \mathbb{Z}^m$ s.t. $\mathbf{t} \equiv \mathbf{x}_i \pmod p$ and $A\mathbf{t} \equiv \beta_i \pmod q$ using CRT,
     $\sigma_{B,i}^{(1)} \leftarrow \mathsf{SamplePre}(p\Lambda_q^{\perp}(B), pT_B, \mathbf{t}, s_2)$,
     /* $\sigma_{B,i}^{(1)} \equiv \mathbf{x}_i \pmod p \wedge B\sigma_{B,i}^{(1)} \equiv \beta_i \pmod q$*/
     $\sigma_{B,i} \leftarrow \mathsf{HenselLift}(q, r, \beta_i, B, \sigma_{B,i}^{(1)})$
     /* $\sigma_{B,i} \equiv \mathbf{x}_i \pmod p \wedge \sigma_{B,i} \equiv \sigma_{B,i}^{(1)} \pmod q \wedge B\sigma_{B,i} \equiv \beta_i \pmod{q^r}$ */
     $\sigma_i := (\sigma_{A,i}, \sigma_{B,i}) \in \mathbb{Z}^{2m}$.
  3) return $(\sigma_1, \ldots, \sigma_{k+1})$

The next algorithm generates the derived signature σ for the result $y = f(x_1, \ldots, x_k)$ of a function $f(X_1, \ldots, X_k) = \sum_{1 \le i,j \le k} f_{ij} X_i X_j, f_{ij} \in \mathbb{F}_p, f_{ij} = f_{ji}$ on the input data set tagged with τ.

- $\mathsf{Eval}(pk, \tau, (\sigma_1, \cdots, \sigma_{k+1}), f) \to \sigma$ for $y = f(x_1, \ldots, x_k) \in \mathbb{F}_p$ with τ.
  1) $g \leftarrow H_0(\tau \| f)$.
  2) $\mathbf{f} := \sum_{1 \le i,j \le k} f_{ij} \cdot (X_i \otimes X_j) + g \cdot (X_{k+1} \otimes X_{k+1})$.
  3) $\sigma \leftarrow \mathbf{f}\big|_{X_i \otimes X_j = \sigma_{A,i} \otimes \sigma_{B,j}} \in \mathbb{Z}^{m^2}$.

The next algorithm verifies the result y of the function f and the input data tagged with τ using the signature σ.

- $\mathsf{Verify}(pk, \sigma, \tau, y, f) \to$ accept or reject.
  1) $g \leftarrow H_0(\tau \| f)$.
  2) $\mathbf{f} := \sum_{1 \le i,j \le k} f_{ij} \cdot (X_i \otimes X_j) + g \cdot (X_{k+1} \otimes X_{k+1})$.
  3) for $i = 1, \cdots, k+1$:
     $\alpha_i \leftarrow H_1(\tau \| i)$, $\beta_i \leftarrow H_2(\tau \| i)$
  4) If the following three conditions hold, then output accept. Otherwise output reject.
     a) Set $V \in \mathbb{Z}^{m \times m}$ s.t. $\sigma = \mathsf{vec}(V)$, $W := (BV)^\top$.
        i) $\|W\| < \delta_1$,
        ii) $\|V \bmod q\| < \delta_2$.
     b) $\sigma \equiv (y + gx_{k+1}^2, *, \cdots, *)^\top \pmod p$.
     c) $(A \otimes B)\sigma \equiv \mathbf{f}\big|_{X_i \otimes X_j = \alpha_i \otimes \beta_j} \in \mathbb{Z}^{n^2} \pmod{q^r}$

The correctness of homomorphic signature schemes is defined as follows.

Definition 2. Let a message space be $\mathcal{M}$ and a target function space be $\mathcal{F}$. An homomorphic signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Eval})$ is correct, if for any (pk, sk) generated by $\mathsf{Setup}(1^n, k)$, any data set $(x_1, \cdots, x_k) \in \mathcal{M}^k$ tagged with $\tau \in \{0,1\}^n$ and any function $f(X_1, \ldots, X_k) \in \mathcal{F}$, the following formula holds:

$$\mathsf{Verify}\left(pk, \mathsf{Eval}\left(pk, \tau, \mathsf{Sign}\left(sk, pk, \tau, (x_1, \cdots, x_k)\right), f\right), \tau, y = f\left(x_1, \ldots, x_k\right), f\right) = 1.$$

Our proposed homomorphic signature scheme satisfies the following lemma.

Lemma 1 (Correctness)**.** The above homomorphic signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Eval})$ with parameters $n, k, r, m \in \mathbb{N}$, primes $p, q \in \mathbb{N}$, $s_1, s_2, \delta_1, \delta_2 \in \mathbb{R}$, is correct if $(k+1)^2 pq s_1 \sqrt{m} \le \delta_1$ and $(k+1)^2 p s_1 s_2 m \le \delta_2$.

Proof. Let (pk, sk) be an output by $\mathsf{Setup}(1^n, k)$, a data set $(x_1, \cdots, x_k) \in (\mathbb{F}_p)^k$ tagged with $\tau \in \{0,1\}^n$ and signatures $(\sigma_1, \ldots, \sigma_{k+1}) \leftarrow \mathsf{Sign}\,(sk, pk, \tau, (x_1, \cdots, x_k))$ such that $\sigma_i = (\sigma_{A,i}, \sigma_{B,i})$, $1 \le i \le k+1$. Let $\sigma \leftarrow \mathsf{Eval}\,(pk, \tau, (\sigma_1, \ldots, \sigma_{k+1}), f)$ for a function $f(X_1, \ldots, X_k) = \sum_{1 \le i,j \le k} f_{ij} X_i X_j, (f_{ij} \in \mathbb{F}_p, f_{ij} = f_{ji})$. Then we have

$$\begin{aligned}\sigma &= \mathbf{f}\big|_{X_i \otimes X_j = \sigma_{A,i} \otimes \sigma_{B,j}} \\ &= \sum_{1 \le i,j \le k} f_{ij} \cdot (\sigma_{A,i} \otimes \sigma_{B,j}) + g \cdot (\sigma_{A,k+1} \otimes \sigma_{B,k+1}) \quad (1)\end{aligned}$$

In the following, we will check the conditions (a), (b), (c) in Verify.

(a) Let V be a matrix s.t. $\sigma = \mathsf{vec}(V)$, $W := (BV)^\top$. Since $\sigma_{A,i} \otimes \sigma_{B,j} = \mathsf{vec}\left(\sigma_{B,j}\sigma_{A,i}^\top\right)$ for $1 \le i, j \le k+1$, we have

$$\begin{aligned}V &= \sum_{1 \le i,j \le k} f_{ij} \cdot \sigma_{B,j}\sigma_{A,i}^\top + g \cdot \sigma_{B,k+1}\sigma_{A,k+1}^\top, \\ W &= \left(\sum_{1 \le i,j \le k} f_{ij} \cdot B\sigma_{B,j}\sigma_{A,i}^\top + g \cdot B\sigma_{B,k+1}\sigma_{A,k+1}^\top\right)^\top \\ &= \sum_{1 \le i,j \le k} f_{ij} \cdot \sigma_{A,i}(B\sigma_{B,j})^\top + g \cdot \sigma_{A,k+1}(B\sigma_{B,k+1})^\top\end{aligned}$$

(i) By the triangular inequality,

$$\|W\| \leq \sum_{1\leq i,j\leq k} \|f_{ij}\| \left\|\sigma_{A,i}(B\sigma_{B,j})^\top\right\| + \|g\| \left\|\sigma_{A,k+1}(B\sigma_{B,k+1})^\top\right\|.$$

Since each $f_{i,j}, g < p$ is satisfied for $1 \leq i,j \leq k$,

$$\|W\| < \sum_{1\leq i,j\leq k+1} p \left\|\sigma_{A,i}(B\sigma_{B,j})^\top\right\|.$$

Since each $\sigma_{B,j}$ is outputted by HenselLift, $B\sigma_{B,j} \equiv \beta_j \pmod{q^r}$ for $1 \leq j \leq k+1$. Also, since each β_j is outputted by the hash function H_2, each coefficient of $\beta_j \in \mathbb{Z}_q^n$ is less than q. These imply

$$\|W\| < \sum_{1\leq i,j\leq k+1} pq \|\sigma_{A,i}\|.$$

Moreover, since each $\sigma_{A,i}$ is outputted by SamplePre with Gaussian parameter s_1, $\|\sigma_{A,i}\| \leq s_1\sqrt{m}$ hold for $1 \leq i \leq k+1$ with overwhelming probability. Therefore, we have $\|W\| < (k+1)^2 pqs_1\sqrt{m}$. Consequently, $\|W\| < \delta_1$ by the assumption.

(ii) By the triangular inequality and $\sigma_{B,i} \equiv \sigma_{B,i}^{(1)} \pmod{q}$ according to HenselLift,

$$\|V \bmod q\| \leq \sum_{1\leq i,j\leq k} \|f_{ij}\| \left\|\sigma_{B,j}^{(1)}\sigma_{A,i}^\top\right\| + \|g\| \left\|\sigma_{B,k+1}^{(1)}\sigma_{A,k+1}^\top\right\|.$$

Since each $f_{i,j}, g < p$ for $1 \leq i,j \leq k$, we have

$$\|V \bmod q\| \leq \sum_{1\leq i,j\leq k+1} p \left\|\sigma_{B,j}^{(1)}\sigma_{A,i}^\top\right\|.$$

Since each $\sigma_{B,j}^{(1)}$ is outputted by SamplePre with Gaussian parameter s_2, $\left\|\sigma_{B,j}^{(1)}\right\| \leq s_2\sqrt{m}$ hold for $1 \leq j \leq k+1$. Moreover, as in the above (i), $\|\sigma_{A,i}\| \leq s_1\sqrt{m}$ hold for $1 \leq i \leq k+1$. This implies that each coefficient of $\sigma_{A,i}$ is at most $s_1\sqrt{m}$. Therefore, we have $\|V \bmod q\| < (k+1)^2 ps_1s_2m$. Consequently, $\|V \bmod q\| < \delta_2$ by the assumption.

(b) Since the initial signatures outputted by Sign algorithm satisfy $\sigma_{A,i}, \sigma_{B,i} \equiv (x_i, 1, \ldots, 1)^\top \pmod{p}$ for $1 \leq i \leq k+1$, the first coefficient in the formula (1) satisfies

$$\begin{aligned}
&\sigma \bmod p \\
&= \sum_{1\leq i,j\leq k} f_{ij} \cdot (\sigma_{A,i} \otimes \sigma_{B,j}) \\
&\quad + g \cdot (\sigma_{A,k+1} \otimes \sigma_{B,k+1}) \bmod p \\
&= \left(\sum_{1\leq i,j\leq k} f_{ij}x_ix_j + gx_{k+1}^2, *, \cdots, * \right)^\top \\
&= \left(f(x_1, \ldots, x_k) + gx_{k+1}^2, *, \cdots, *\right)^\top \\
&= (y + gx_{k+1}^2, *, \cdots, *)^\top.
\end{aligned}$$

(c) By using the fact $(A \otimes B)(\sigma_{A,i} \otimes \sigma_{B,j}) = A\sigma_{A,i} \otimes B\sigma_{B,j}$ for $1 \leq i,j \leq k+1$, the formula (1) implies

$$\begin{aligned}
&(A \otimes B)\sigma \\
&= (A \otimes B) \left(\sum_{1\leq i,j\leq k} f_{ij} \cdot (\sigma_{A,i} \otimes \sigma_{B,j}) \right. \\
&\quad \left. + g \cdot (\sigma_{A,k+1} \otimes \sigma_{B,k+1}) \right) \\
&= \sum_{1\leq i,j\leq k} f_{ij} \cdot (A \otimes B)(\sigma_{A,i} \otimes \sigma_{B,j}) \\
&\quad + g \cdot (A \otimes B)(\sigma_{A,k+1} \otimes \sigma_{B,k+1}) \\
&= \sum_{1\leq i,j\leq k} f_{ij} \cdot (A\sigma_{A,i} \otimes B\sigma_{B,j}) \\
&\quad + g \cdot (A\sigma_{A,k+1} \otimes B\sigma_{B,k+1}).
\end{aligned}$$

Since the initial signatures outputted by Sign algorithm satisfy $A\sigma_{A,i} \equiv \alpha_i \pmod{q^r}$ for $1 \leq i \leq k+1$ and $B\sigma_{B,j} \equiv \beta_j \pmod{q^r}$ for $1 \leq j \leq k+1$, we have

$$\begin{aligned}
(A \otimes B)\sigma &= \sum_{1\leq i,j\leq k} f_{ij} \cdot (\alpha_i \otimes \beta_j) + g \cdot (\alpha_{k+1} \otimes \beta_{k+1}) \\
&= \mathbf{f}\big|_{X_i \otimes X_j = \alpha_i \otimes \beta j} \bmod q^r.
\end{aligned}$$

□

IV. Security

This section discusses the security of our homomorphic signature scheme.

Let a security parameter $n \in \mathbb{N}$. For a homomorphic signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Eval})$, the advantage $\mathrm{Adv}_{\mathcal{A}}^{\mathcal{S}}(n)$ is the probability that any probabilistic polynomial time algorithm $\mathcal{A}$ wins the following game:

1) A challenger $\mathcal{C}$ generates (pk, sk) by using $\mathsf{Setup}(1^n, k)$ and sends the public key pk to $\mathcal{A}$.
2) Sign queries: the next Steps a),b) are repeated for $\ell \in \{1, \ldots, L\}$.
   a) $\mathcal{A}$ asks $\mathcal{C}$ the signatures for a data set $(x_1^{(\ell)}, \ldots, x_k^{(\ell)})$.
   b) $\mathcal{C}$ tags the data set with $\tau^{(\ell)} \leftarrow \{0,1\}^n$ and generates signatures $(\sigma_1^{(\ell)}, \cdots, \sigma_{k+1}^{(\ell)})$ by using $\mathsf{Sign}(sk, pk, \tau^{(\ell)}, (x_1^{(\ell)}, \ldots, x_k^{(\ell)}))$. Then $\mathcal{C}$ answers the signatures to $\mathcal{A}$.
3) $\mathcal{A}$ outputs a tag $\tilde{\tau}$, a message $\tilde{m}$, a function $\tilde{f}$, and a signature $\tilde{\sigma}$.

The adversary $\mathcal{A}$ wins if $\mathsf{Verify}(pk, \tilde{\sigma}, \tilde{\tau}, \tilde{m}, \tilde{f}) = 1$ and either

Type 1: $\tilde{\tau} \neq \tau^{(\ell)}$ for $^\forall\ell$,

or

Type 2: $\tilde{\tau} = \tau^{(\ell)}$ for some ℓ, $\tilde{m} \neq \tilde{f}(x_1^{(\ell)}, \ldots, x_k^{(\ell)})$.

The security of a homomorphic signature scheme is defined as follows.

Definition 3 (Unforgeability)**.** A homomorphic signature scheme $\mathcal{S}$ with a security parameter n is unforgeable if $\mathrm{Adv}_{\mathcal{A}}^{\mathcal{S}}(n) \leq \mathrm{negl}(n)$ for all probabilistic polynomial time algorithm $\mathcal{A}$.

Let the advantage $\mathrm{Adv}_{\mathcal{S}}^{\mathrm{SIS}_{n,q,\beta}}(n)$ be the probability to solve the $\mathrm{SIS}_{n,q,\beta}$ problem defined in the section II-B. The following lemma holds for our scheme.

Lemma 2. Let $\mathcal{S}$ be the homomorphic signature scheme described in III with the parameters $k, n, m, p, q, r \in \mathbb{N}$, $s_1, s_2, \delta_1, \delta_2 \in \mathbb{R}$, and let $\gamma = \delta_1\sqrt{m}$, and let h be a number of queries to the hash oracle.

For any Type 1 adversary $\mathcal{A}_1$ of $\mathcal{S}$, there exist the solvers $\mathcal{S}$, $\mathcal{S}'$ for the SIS problems such that
$\mathrm{Adv}^{\mathcal{S}}_{\mathcal{A}_1}(n) \leq h \cdot \mathrm{Adv}^{\mathrm{SIS}_{n,q^r,\gamma}}_{\mathcal{S}}(n) + \mathrm{Adv}^{\mathrm{SIS}_{n,q,\delta_2}}_{\mathcal{S}'}(n) + \mathrm{negl}(n)$.

For any Type 2 adversary $\mathcal{A}_2$ of $\mathcal{S}$, there exist the solvers $\mathcal{S}$, $\mathcal{S}'$ for the SIS problems such that
$\mathrm{Adv}^{\mathcal{S}}_{\mathcal{A}_2}(n) \leq \mathrm{Adv}^{\mathrm{SIS}_{n,q^r,\gamma}}_{\mathcal{S}}(n) + \mathrm{Adv}^{\mathrm{SIS}_{n,q,\delta_2}}_{\mathcal{S}'}(n) + \mathrm{negl}(n)$.

Proof. At first, we construct the solver for the problem $\mathrm{SIS}_{n,q^r,\gamma}$ using the Type 1 adversary $\mathcal{A}_1$. For an input $A \xleftarrow{\cup} \mathbb{Z}_{q^r}^{n\times m}$ of the SIS problem, set $A' \in \mathbb{Z}_{q^r}^{n\times(m-1)}$ and $\mathbf{a} = (a_1, \ldots, a_n)^\top \in \mathbb{Z}_{q^r}^n$ such that $A = [A'|\mathbf{a}]$. The solver $\mathcal{S}_A$ for $\mathrm{SIS}_{n,q^r,\gamma}$ is constructed as follows.

1) Setup':
   By using $\mathsf{Setup}(1^n, k)$, $\mathcal{S}_A$ can honestly generate $pk' := (A', B, x_{k+1}, H_0, H_1, H_2)$ and $sk' := (T_B)$ except $T_{A'}$. Then $\mathcal{S}_A$ passes pk' to $\mathcal{A}_1$.

   For sign queries by $\mathcal{A}_1$, $\mathcal{S}_A$ answers the signatures generated by the following Step 2 and 3:
2) $\mathcal{A}_1$ asks $\mathcal{S}_A$ for signing the data set $(x_1^{(\ell)}, \ldots, x_k^{(\ell)})$.
3) $\mathcal{S}_A$ tags the data set with $\tau^{(\ell)} \xleftarrow{\cup} \{0,1\}^n$, and
   Sign':
   for $i \in \{1, \cdots, k+1\}$:
   $\sigma^{(\ell)}_{A',i} \leftarrow D_{\mathbb{Z}^m, s_1} \in \mathbb{Z}^{m-1}$,
   $H_1[\tau^{(\ell)}\|i] := \alpha_i^{(\ell)} \leftarrow A'\sigma^{(\ell)}_{A',i} \bmod q^r \in \mathbb{Z}^n_{q^r}$.
   $\mathcal{S}_A$ can generate $\sigma^{(\ell)}_{B,i} \in \mathbb{Z}^{m-1}$ by using Sign with T_B,
   $\sigma_i^{(\ell)} := (\sigma^{(\ell)}_{A',i}, \sigma^{(\ell)}_{B,i}) \in \mathbb{Z}^{2(m-1)}$.
   $\mathcal{S}_A$ answers $\tau^{(\ell)}, (\sigma_1^{(\ell)}, \cdots, \sigma_{k+1}^{(\ell)})$ to $\mathcal{A}_1$.

   For hash queries except those in Sign' by $\mathcal{A}_1$, $\mathcal{S}_A$ guesses the output $(\tilde{\tau} \notin \{\tau^{(\ell)}\}, \tilde{f})$ of $\mathcal{A}_1$ and answers the hash values according to the following Step 4 with respect to the guess. Except for the guess, $\mathcal{S}_A$ answers random values in the domain of the hash function.
4) $\mathcal{A}_1$ asks $\mathcal{S}_A$ for hash values of $(\tilde{\tau}, \tilde{f} = \sum_{1\leq i,j\leq k} f_{ij}(X_i \otimes X_j))$, then
   $H_0[\tilde{\tau}\|\tilde{f}] := g \xleftarrow{\cup} \mathbb{F}_p$.
   for $i \in \{1, \ldots, k\}$:
   $H_1[\tilde{\tau}\|i] := \alpha_i \xleftarrow{\cup} \mathbb{Z}^n_{q^r}$.
   for $j \in \{1, \ldots, k+1\}$:
   $H_2[\tilde{\tau}\|j] := \beta_j = (\beta_{j,1} \ldots \beta_{j,n})^\top \xleftarrow{\cup} \mathbb{Z}^n_q$.
   $\alpha_{k+1} \leftarrow \left(\mathbf{a} - \sum_{1\leq i,j\leq k} f_{ij}\beta_{j,1}\alpha_i\right)/(g\beta_{k+1,1})$.
   $H_1[\tilde{\tau}\|k+1] := \alpha_{k+1} \in \mathbb{Z}^n_{q^r}$.
   $\mathcal{S}_A$ answers $H_0[\tilde{\tau}\|\tilde{f}]$, $H_1[\tilde{\tau}\|1], \ldots, H_1[\tilde{\tau}\|k+1]$, $H_2[\tilde{\tau}\|1], \ldots, H_2[\tilde{\tau}\|k+1]$ to $\mathcal{A}_1$.
5) $\mathcal{A}_1$ outputs $(\tilde{\tau}, \tilde{m}, \tilde{\sigma}, \tilde{f})$. If $\tilde{\tau}$ is not the guessed value in the above Step 4, then abort.
6) Let the $(m-1)\times(m-1)$ matrix V such that $\mathsf{vec}(V) = \tilde{\sigma}$, and let the $(m-1)\times n$ matrix $W := (BV)^\top$. If $W \equiv O \pmod q$, then $\mathrm{abort}_{\mathcal{S}_A}$. Otherwise, there exists the nonzero column $\mathbf{w}$ in W. $\mathcal{S}_A$ can output the column vector $\mathbf{z} := (\mathbf{w}^\top \mid -1)^\top \in \mathbb{Z}^m$.

Since $A'\mathbf{w} \equiv \mathbf{a} \pmod{q^r}$, the output $\mathbf{z}$ satisfies $A\mathbf{z} = A'\mathbf{w} - \mathbf{a} \equiv 0 \pmod{q^r}$. Since $\|W\| < \delta_1$, $\|\mathbf{z}\| < \sqrt{(m-1)\delta_1^2 + 1} < \gamma$ if $\delta_1 > 1$. Therefore, $\mathcal{S}_A$ outputs a solution for $\mathrm{SIS}_{n,q^r,\gamma}$ with the input A.

Next, we construct the solver $\mathcal{S}_B$ for the problem $\mathrm{SIS}_{n,q,\delta_2}$ with an input $B \xleftarrow{\cup} \mathbb{Z}_q^{n\times m}$ by using the Type 1 adversary $\mathcal{A}_1$ as follows.

1) Setup':
   By using $\mathsf{Setup}(1^n, k)$, $\mathcal{S}_B$ can honestly generate $pk' := (A, B, x_{k+1}, H_0, H_1, H_2)$ and $sk' := (T_A)$ except T_B. Then $\mathcal{S}_B$ passes pk' to $\mathcal{A}_1$.

   For sign queries by $\mathcal{A}_1$, $\mathcal{S}_B$ answers the signatures generated by the following Step 2 and 3:
2) $\mathcal{A}_1$ asks $\mathcal{S}_B$ for signing the data set $(x_1^{(\ell)}, \ldots, x_k^{(\ell)})$.
3) $\mathcal{S}_B$ tags the data set with $\tau^{(\ell)} \xleftarrow{\cup} \{0,1\}^n$, and
   Sign':
   for $i \in \{1, \cdots, k+1\}$:
   $\mathcal{S}_B$ can generate $\sigma^{(\ell)}_{A,i} \in \mathbb{Z}^m$ by using Sign with T_A,
   $\sigma^{(\ell)}_{B,i} \leftarrow \mathsf{HenselLift}(q, r, \beta_i, B, \sigma' \leftarrow D_{\mathbb{Z}^m, s_2})$
   $H_2[\tau^{(\ell)}\|i] := \beta_i^{(\ell)} \leftarrow B\sigma^{(\ell)}_{B,i} \bmod q \in \mathbb{Z}^n_q$,
   $\sigma_i^{(\ell)} = (\sigma^{(\ell)}_{A,i}, \sigma^{(\ell)}_{B,i}) \in \mathbb{Z}^{2m}$.
   $\mathcal{S}_B$ answers $\tau^{(\ell)}, (\sigma_1^{(\ell)}, \cdots, \sigma_{k+1}^{(\ell)})$ to $\mathcal{A}_1$.

   For hash queries except those in Sign' by $\mathcal{A}_1$, $\mathcal{S}_B$ answers random values in the domain of the hash function.
4) $\mathcal{A}_1$ outputs $(\tilde{\tau}, \tilde{m}, \tilde{\sigma}, \tilde{f})$.
5) Let the $m \times m$ matrix V such that $\mathsf{vec}(V) = \tilde{\sigma} (\neq 0)$. If $BV \not\equiv O \pmod q$, then $\mathrm{abort}_{\mathcal{S}_B}$. Otherwise, there exists the nonzero column $\mathbf{v}$ in V. $\mathcal{S}_B$ can output $\mathbf{v} \in \mathbb{Z}^m$.

The output $\mathbf{v}$ satisfies $B\mathbf{v} \equiv 0 \pmod q$ and $\|\mathbf{v} \bmod q\| < \delta_2$. Therefore, $\mathcal{S}_B$ solves $\mathrm{SIS}_{n,q,\delta_2}$ with the input B.

By using the above solvers $\mathcal{S}_A$ and $\mathcal{S}_B$, we will evaluate the probability $\mathrm{Adv}^{\mathcal{S}}_{\mathcal{A}_1}(n)$. If $\mathcal{S}_A$ or $\mathcal{S}_B$ chooses the same tag for the different queries in Step 3, then the simulation fails. But the failure probability is $\mathrm{negl}(n)$. Since the number of the hash queries in $\mathcal{S}_A$ is h, the probability that the guess of $\mathcal{S}_A$ in Step 4 is $1/h$. Moreover, if $\mathcal{A}_1$ succeeds in forging the signature for $\mathcal{S}$, then both Step 6 of $\mathcal{S}_A$ and Step 5 of $\mathcal{S}_B$ does not abort at the same time. Therefore, we have
$\mathrm{Adv}^{\mathcal{S}}_{\mathcal{A}_1}(n) \leq h \cdot \mathrm{Adv}^{\mathrm{SIS}_{n,q^r,\gamma}}_{\mathcal{S}_A}(n) + \mathrm{Adv}^{\mathrm{SIS}_{n,q,\delta_2}}_{\mathcal{S}_B}(n) + \mathrm{negl}(n)$.

Secondly, by using the Type 2 adversary $\mathcal{A}_2$, we construct the solver $\mathcal{S}'_A$ for the problem $\mathrm{SIS}_{n,q^r,\gamma}$ with an input $A \xleftarrow{\cup} \mathbb{Z}_{q^r}^{n\times m}$ as follows.

1) Setup':
   By using $\mathsf{Setup}(1^n, k)$, $\mathcal{S}_A$ can honestly generate $pk' := (A, B, x_{k+1}, H_0, H_1, H_2)$ and $sk' := (T_B)$ except T_A. Then $\mathcal{S}'_A$ passes pk' to $\mathcal{A}_2$.

   For sign queries by $\mathcal{A}_2$, $\mathcal{S}'_A$ answers the signatures generated by the following Step 2 and 3:
2) $\mathcal{A}_2$ asks $\mathcal{S}'_A$ for signing the data set $(x_1^{(\ell)}, \ldots, x_k^{(\ell)})$.
3) $\mathcal{S}'_A$ tags the data set with $\tau^{(\ell)} \xleftarrow{\cup} \{0,1\}^n$, and
   Sign':

for $i \in \{1, \cdots, k+1\}$:
$\sigma_{A,i}^{(\ell)} \leftarrow D_{\mathbb{Z}^m, s_1} \in \mathbb{Z}^m$,
$H_1[\tau^{(\ell)} \| i] := \alpha_i^{(\ell)} \leftarrow A\sigma_{A,i}^{(\ell)} \bmod q^r \in \mathbb{Z}_{q^r}^n$.
$\mathcal{S}'_A$ can generate $\sigma_{B,i}^{(\ell)} \in \mathbb{Z}^m$ by using Sign with T_B,
$\sigma_i^{(\ell)} = (\sigma_{A,i}^{(\ell)}, \sigma_{B,i}^{(\ell)}) \in \mathbb{Z}^{2m}$.
$\mathcal{S}'_A$ answers $\tau^{(\ell)}, (\sigma_1^{(\ell)}, \cdots, \sigma_{k+1}^{(\ell)})$ to $\mathcal{A}_2$.

4) $\mathcal{A}_2$ outputs $(\tau^{(\ell')}, \tilde{m}, \tilde{\sigma}, \tilde{f})$.
5) $\sigma' \leftarrow \mathsf{Eval}\left(pk', \tau^{(\ell')}, (\sigma_1^{(\ell')}, \cdots, \sigma_{k+1}^{(\ell')}), \tilde{f}\right)$.
6) Let the $m \times m$ matrix V such that $\mathsf{vec}(V) = \tilde{\sigma} - \sigma' (\neq 0)$, and let the $m \times n$ matrix $W := (BV)^\top$. If $W \equiv O \pmod q$, then $\mathrm{abort}_{\mathcal{S}'_A}$. Otherwise, there exists the nonzero column vector $\mathbf{w}$ in W. $\mathcal{S}'_A$ can output the vector $\mathbf{w} \in \mathbb{Z}^m$.

The output $\mathbf{w}$ satisfies $A\mathbf{w} \equiv 0 \pmod{q^r}$ and $\|\mathbf{w}\| < \delta_1\sqrt{m} = \gamma$ since $\|W\| < \delta_1$. Therefore, $\mathcal{S}'_A$ outputs a solution for $\mathrm{SIS}_{n,q^r,\gamma}$ with the input A.

Next, we construct the solver $\mathcal{S}'_B$ for the problem $\mathrm{SIS}_{n,q,\delta_2}$ with an input $B \xleftarrow{\cup} \mathbb{Z}_q^{n \times m}$ by using $\mathcal{A}_2$ as follows.

1) Setup':
By using $\mathsf{Setup}(1^n, k)$, $\mathcal{S}'_B$ can honestly generate $pk' := (A, B, x_{k+1}, H_0, H_1, H_2)$ and $sk' := (T_A)$ except T_B. Then $\mathcal{S}'_B$ passes pk' to $\mathcal{A}_2$.

For sign queries by $\mathcal{A}_2$, $\mathcal{S}'_B$ answers the signatures generated by the following Step 2 and 3:

2) $\mathcal{A}_2$ asks $\mathcal{S}'_B$ for signing the data set $(x_1^{(\ell)}, \ldots, x_k^{(\ell)})$.
3) $\mathcal{S}'_B$ tags the data set with $\tau^{(\ell)} \xleftarrow{\cup} \{0,1\}^n$, and Sign':
for $i \in \{1, \cdots, k+1\}$:
$\mathcal{S}'_B$ can generate $\sigma_{A,i}^{(\ell)} \in \mathbb{Z}^m$ by using Sign with T_A,
$\sigma_{B,i}^{(\ell)} \leftarrow \mathsf{HenselLift}(q, r, \beta_i, B, \sigma' \leftarrow D_{\mathbb{Z}^m, s_2} \in \mathbb{Z}^m)$
$H_2[\tau^{(\ell)} \| i] := \beta_i^{(\ell)} \leftarrow B\sigma_{B,i}^{(\ell)} \bmod q \in \mathbb{Z}_q^n$,
$\sigma_i^{(\ell)} = (\sigma_{A,i}^{(\ell)}, \sigma_{B,i}^{(\ell)}) \in \mathbb{Z}^{2m}$.
$\mathcal{S}'_B$ answers $\tau^{(\ell)}, (\sigma_1^{(\ell)}, \cdots, \sigma_{k+1}^{(\ell)})$ to $\mathcal{A}_2$.

4) $\mathcal{A}_2$ outputs $(\tilde{\tau}, \tilde{m}, \tilde{\sigma}, \tilde{f})$.
5) Let the $m \times m$ matrix V such that $\mathsf{vec}(V) = \tilde{\sigma}$. If $BV \not\equiv O \pmod q$, then $\mathrm{abort}_{\mathcal{S}'_B}$. Otherwise, there exists the nonzero column vector $\mathbf{v}$ in V. $\mathcal{S}'_B$ can output $\mathbf{v} \in \mathbb{Z}^m$.

The output $\mathbf{v}$ satisfies $B\mathbf{v} \equiv 0 \pmod q$ and $\|\mathbf{v} \bmod q\| < \delta_2$. Therefore, $\mathcal{S}'_B$ solves $\mathrm{SIS}_{n,q,\delta_2}$ with the input B.

By using the above solvers $\mathcal{S}'_A$ and $\mathcal{S}'_B$, we will evaluate the probability $\mathrm{Adv}_{\mathcal{A}_2}^{\mathcal{S}}(n)$. If $\mathcal{S}'_A$ or $\mathcal{S}'_B$ chooses the same tag for the different queries in Step 3, then the simulation fails. But the failure probability is $\mathrm{negl}(n)$. Moreover, if $\mathcal{A}_2$ succeeds in forging the signature for $\mathcal{S}$, then the abortion of either Step 6 in $\mathcal{S}'_A$ or Step 5 in $\mathcal{S}'_B$ does not occur. Therefore, we have

$$\mathrm{Adv}_{\mathcal{A}_2}^{\mathcal{S}}(n) \le \mathrm{Adv}_{\mathcal{S}'_A}^{\mathrm{SIS}_{n,q^r,\gamma}}(n) + \mathrm{Adv}_{\mathcal{S}'_B}^{\mathrm{SIS}_{n,q,\delta_2}}(n) + \mathrm{negl}(n).$$

□

Lemma 2 implies the following corollary about the security of our homomorphic signature scheme in section III.

Corollary 1 (Unforgeability)**.** If the $\mathrm{SIS}_{n,q,\gamma}$ with parameters $n, m, q \in \mathbb{N}$, $\gamma \in \mathbb{R}$ is infeasible, then the homomorphic signature scheme $\mathcal{S}$ described in section III is unforgeable in the random oracle model.

V. Parameters

We consider parameters such that our scheme satisfies its correctness and unforgeability. First, we show δ_1, δ_2 such that satisfy the correctness conditions in Lemma 1. Next, under these δ_1, δ_2 we set up the parameters k,p,r,q,m to make the SIS problems infeasible. Followings are detail.

According to the parameter settings in section II-C, we can choose $s_1 = \Theta(pnr \log q \log n)$, because $\mathsf{TrapGen}(n, q^r, m)$ outputs (A, T_A) such that $\|T_A\| = O(nr \log q)$ and $\mathsf{SamplePre}(p\Lambda_{q^r}^{\perp}(A), pT_A, \mathbf{t}, s_1)$ requires $s_1 \ge \|pT_A\| \cdot \omega(\sqrt{\log n})$. Similarly, we can choose $s_2 = \Theta(pn \log q \log n)$ for $\mathsf{TrapGen}(n, q, m) \to (B, T_B)$ and $\mathsf{SamplePre}(p\Lambda_q^{\perp}(B), pT_B, \mathbf{t}, s_2)$. Therefore we have

$$\delta_1 = \Theta\left(k^2 p^2 q r m^{1/2} n \log n \log q\right),$$
$$\delta_2 = \Theta\left(k^2 p^3 r m n^2 \log^2 n \log^2 q\right)$$

so that our scheme is correct according to Lemma 1. Moreover, we can choose

$$k = \Theta(1), p = \Theta(1), r = 2, q = \Theta(n^4), m = \Theta(n \log n).$$

Then we have $\gamma = \delta_1\sqrt{m} = \Theta(n^{6.5} \log^3 n)$ in Lemma 2 and $q^r = \Theta(n^8)$. Since $\gamma < q^r$, the $\mathrm{SIS}_{n,q^r,\gamma}$ problem is infeasible according to the section II-B. Also, we have $\delta_2 = \Theta(n^3 \log^5 n)$ and $q = \Theta(n^4)$. Since $\delta_2 < q$, the $\mathrm{SIS}_{n,q,\delta_2}$ problem is also infeasible. Consequently, corollary 1 implies that our homomorphic signature scheme $\mathcal{S}$ shown in section III is unforgeable.

Acknowledgment

This work was supported by JST CREST Grant Number JPMJCR1503, Japan. We thank the anonymous reviewers for helpful comments.

References

[1] G. Traverso, D. Demirel, and J. Buchmann, *Homomorphic Signature Schemes*. Cham: Springer International Publishing, 2016, pp. 11–21.
[2] D. Boneh and D. M. Freeman, "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures," in *International Workshop on Public Key Cryptography*. Springer, 2011, pp. 1–16.
[3] ——, "Homomorphic signatures for polynomial functions," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2011, pp. 149–168.
[4] S. Gorbunov, V. Vaikuntanathan, and D. Wichs, "Leveled fully homomorphic signatures from standard lattices," in *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, ser. STOC '15. New York, NY, USA: ACM, 2015, pp. 469–477.
[5] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. ACM, 1996, pp. 99–108.
[6] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, ser. STOC '08. New York, NY, USA: ACM, 2008, pp. 197–206.
[7] A. J. Laub, *Matrix analysis for scientists and engineers*. Siam, 2005.
[8] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2012, pp. 700–718.