Design of New Linearly Homomorphic Signatures on Lattice

Rakyong Choi *  Kwangjo Kim *

Abstract: This paper introduces two designs to enhance the Boneh and Freemans linearly homomorphic signature over binary fields, to overcome the limitations to implement homomorphic signatures to the real world scenario due to the heavy calculation and under multiple signers setting for a message.

Based on our concurrent work on classification on lattice-based trapdoor functions in SCIS 2017, we modify some algorithms from the original signature. We design the linearly homomorphic ring signature by adopting Wang and Sun's sampling algorithm $GenSamplePre()$ instead of the original sampling algorithm $SamplePre()$ by Gentry et al. Also, we adopt the mixing and vanishing technique of trapdoors by Boyen to design more efficient linearly homomorphic signature scheme with short signatures.

Keywords: ring signature, homomorphic signature, lattices, trapdoor function, sampling algorithm

1 Introduction

1.1 Background and Motivation

As the infrastructure of cloud systems increases, one of uprising security challenges is how the cloud server computes a function of encrypted messages without decryption. Despite of numerous studies on fully homomorphic encryption on lattices [1–3] to enable the server to calculate any function of encrypted message without decryption, there is another security issue in cloud system as how the cloud server gives authenticity for the function of encrypted message.

For authenticity of cloud systems, a signature is a well-known cryptographic primitive. But, the cloud server should have the power to generate the proper signature for a computation of messages without permission from the signer of each message as well. If the signature satisfies this condition, we say that the signature has the homomorphic property. Especially, a signature is called linearly homomorphic when it supports constructing the proper signature for the linear combination of messages [4, 5] and fully homomorphic when it supports constructing the proper signature for any function of messages [6, 7].

But, there are limitations to implement homomorphic signatures to the real world scenario due to the heavy calculation and under multiple signers setting for a message.

1.2 Our Contribution

In the work of Choi and Kim in SCIS 2017 [8], they summarize the characteristics of lattice-based trapdoor functions and their preimage sampling algorithms using a trapdoor. Based on their paper, we modify the algorithms from lattice-based linearly homomorphic signature over binary fields by Boneh and Freeman [4].

We first consider the linearly homomorphic ring signature over binary fields by adopting Wang and Sun's preimage sampling algorithm $GenSamplePre()$ [9]. We let each member in the ring take their own public key and secret key by trapdoor generation function $TrapGen()$ during the setup phase in the first design. Then, we concatenate the public key of each member to make the common public key. Then, in the signing phase, we modify the preimage sampling algorithm from well-known $SamplePre()$ to $GenSamplePre()$.

This design can be used in the real world scenario since some information on cloud system is signed by an organization instead of an individual and there should be at least two people to authenticate the message where each person has his/her secret key. In this situation, we need multiple signers with different secret keys for a single message and the corresponding signature is valid only if all signers are trustworthy.

Also, we adopt the mixing and vanishing technique of trapdoors introduced by Boyen [10] to design the linearly homomorphic signature scheme over binary fields with short signatures so that we have more practical linearly homomorphic scheme.

1.3 Related Work

In 2011, Boneh and Freeman [4] published their seminal work on linearly homomorphic signature over binary fields based on lattices with new lattice-based hard problems called $k$-SIS problem. Boneh and Freeman [5] also suggested that some bounded homomorphic signature can be constructed using ideal lattices from Gentry’s fully homomorphic encryption [1].

After Boneh and Freeman’s work, lattices have become a main tool to make linearly and fully homomorphic signatures. Zhang et al. [11] introduced the notion
of a homomorphic aggregate signature which doesn’t need to have the same secret key to combine multiple messages. Then, they suggested a linearly homomorphic aggregate signature using the random basis generation algorithm $\text{RandBasis}()$ by Cash et al. [12] to generate multiple secret keys.

Jing [13] separately suggested an efficient homomorphic aggregate signature with linear homomorphism as they concatenate a public key of each signer and use the extending trapdoor basis algorithm $\text{ExtBasis}()$ by Cash et al. [12]. Both Zhang et al. [11] and Jing’s [13] contributions are making multi-key linearly homomorphic signatures.

Choi and Kim [14] used the same technique suggested by Jing but pre-shared the message to multiple signers of a message to get the linearly homomorphic multisignature. This work suggested the first construction of multi-key multi-party linearly homomorphic signatures to the best of our knowledge.

Besides the linearly homomorphic signatures, Gurbunov et al. [6] suggested the first fully homomorphic signature scheme with a homomorphic trapdoor function but there is only one secret key. Recently, Fiore et al. [7] suggested a fully homomorphic signature scheme with multi-key setting, i.e., there are multiple secret keys.

### 1.4 Outline of the Paper

Section 2 gives a notation and a background on a lattice and lattice-based cryptography from the definition of lattices and hard problems on lattices to lattice-based algorithms for trapdoor generation and sampling. Then, formal definition and security requirement of linearly homomorphic signatures with detailed construction is given in Section 3.

We give the design of new linearly homomorphic signatures in Section 4 and we give a concluding remark with future work in Section 5.

### 2 Preliminaries

#### 2.1 Notation

We denote vectors as small bold letters (e.g., $\mathbf{x}$, $\mathbf{y}$) and matrices as big bold letters (e.g., $\mathbf{A}$, $\mathbf{B}$).

Let $\mathbb{R}$ and $\mathbb{Z}$ express the set of real numbers and the set of integers, respectively and small alphabet letters express real numbers (e.g., $a$, $b$, $c$).

For any integer $q \geq 2$, $\mathbb{Z}_q$ denotes the ring of integers modulo $q$ and $\mathbb{Z}_q^{n \times m}$ denotes the set of $n \times m$ matrices with entries in $\mathbb{Z}_q$. When $\mathbf{A} \in \mathbb{Z}_q^{n \times m_1}$, $\mathbf{B} \in \mathbb{Z}_q^{n \times m_2}$, we write the concatenation of $\mathbf{A}$ and $\mathbf{B}$ as $[\mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n \times (m_1 + m_2)}$.

Let $f : \mathbb{Z} \rightarrow \mathbb{R}^+$ be a function $f$ on $a$ and $b$. We say a function $f : \mathbb{Z} \rightarrow \mathbb{R}^+$ is negligible when $f = O(n^{-c})$ for all $c > 0$ and denoted by $\text{negl}(n)$. A function $g(m) = \lceil m \rceil$ is the ceiling function from $\mathbb{R}$ to $\mathbb{Z}$ such that $g(m)$ is the smallest integer which is greater than or equal to $m$.

$||\mathbf{x}||$ represents the Euclidean norm of $\mathbf{x}$ and $||\mathbf{B}||$ represents the maximum of Euclidean norms of the columns of $\mathbf{B}$. For instance, when $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$, $||\mathbf{B}|| = \max_i ||\mathbf{b}_i||$. Then, we denote $\tilde{\mathbf{B}} = (\mathbf{b}_1 | \mathbf{b}_2 | \cdots | \mathbf{b}_m)$ for the Gram-Schmidt orthogonalization of columns of $\mathbf{B}$ and denote $||\tilde{\mathbf{B}}|| = \max_i ||\mathbf{b}_i||$ for Gram-Schmidt norm of $\mathbf{B}$.

#### 2.2 Lattices and Algorithm for Trapdoor Basis Delegation

Briefly, lattices are a fascinating tool in modern cryptography and a lattice $\Lambda$ can be defined as a discrete subgroup of $\mathbb{R}^m$ with its basis $\mathbb{S}$. A basis $\mathbb{S}$ of $\Lambda$ is a set of linearly independent vectors $\mathbb{S} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m\}$ which spans the lattice $\Lambda$ and $\mathbb{S} = (\mathbf{b}_1 | \mathbf{b}_2 | \cdots | \mathbf{b}_m)$ is a basis matrix of lattice $\Lambda$.

Integer lattices are defined as a subgroup of $\mathbb{Z}^m$ instead of $\mathbb{R}^m$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times m}$, we can denote lattices as a set $\Lambda_q^\mathbf{A} = \{\mathbf{e} \in \mathbb{Z}_q^m | \mathbf{A} \cdot \mathbf{e} = \mathbf{u} \mod q\}$ and as a set $\Lambda_q^\mathbf{e} = \{\mathbf{e} \in \mathbb{Z}_q^m | \mathbf{A} \cdot \mathbf{e} = 0 \mod q\}$ when $\mathbf{u} = \mathbf{0}$.

Lattice-based cryptography has a lot of advantages that their security is based on the average-case hardness problems like Short Integer Solution (SIS) problem and Learning With Errors (LWE) problem, which remain secure against quantum computing attacks and can be reduced to the worst-case hardness problem in lattices like Shortest Vector Problem (SVP) and Closest Vector Problem (CVP). Among them, SIS problem is defined as below.

**Definition 1.** (SIS problem) Given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with $m \geq n \log q$ and its corresponding lattice $\Lambda_q^\mathbf{A} = \{\mathbf{e} \in \mathbb{Z}_q^m | \mathbf{A} \cdot \mathbf{e} = 0\}$, it is hard to find a small vector $\mathbf{e} \in \Lambda_q^\mathbf{A}$, such that $||\mathbf{e}|| \leq \beta$ for some $\beta \geq \sqrt{n \log q}$ and $\mathbf{A} \cdot \mathbf{e} = 0 \mod q$, whose coefficients are either $-1$, $0$, or $1$.

If we have the short “trapdoor” basis, all hard problems in lattice become solvable efficiently. Alwen and Peikert [15] introduced the trapdoor generation algorithm $\text{TrapGen}(n, m, q)$ which generates a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with its “trapdoor” matrix $\mathbf{T} \in \mathbb{Z}^{m \times m}$ satisfying the following functionality:

$\text{TrapGen}(n, m, q)$:

For the security parameter $n, m = [6n \log q]$ and an integer $q$, this algorithm outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and its trapdoor $\mathbf{T}$ such that $\mathbf{T}$ is a basis of $\Lambda_q^\mathbf{A}$ with low Gram-Schmidt norm $||\mathbf{T}|| \leq 30\sqrt{n \log q}$.

Without loss of generality, we assume that a matrix $\mathbf{A}$ extracted from $\text{TrapGen}(n, m, q)$ has a full rank. In our construction, a matrix $\mathbf{A}$ and its trapdoor $\mathbf{T}$ are used as a public key and a secret key, respectively.

Cash et al. [12] introduced the technique to randomly generate the basis from the matrix and to extend the basis to higher dimension in the concept of bonsai trees using the following algorithms.
RandBasis(T, s) :
For the trapdoor matrix $T$ of $A \in \mathbb{Z}_q^{n \times m}$ and parameter $s \geq \|T\| \cdot \omega(\sqrt{\log n})$, this algorithm outputs a basis $T'$ for $\Lambda^+_L(A)$ with $\|T'\| \leq s \cdot \sqrt{m}$.

ExtBasis(T, B) :
For the trapdoor matrix $T$ of $A \in \mathbb{Z}_q^{n \times m}$ and the matrix $B = A||A' \in \mathbb{Z}_q^{n \times (m+n')}$, this algorithm outputs a basis $S$ for $\Lambda^+_L(B)$ with $\|S\| = \|T\|$ in polynomial time, i.e., Gram-Schmidt norm of $S$ is equal to that of $T$.

The extending trapdoor basis algorithm ExtBasis(T, B) can be implemented to get a short basis of the higher-dimensional lattice from the lower-dimensional lattice.

2.3 Discrete Gaussian Distribution and Sampling Algorithm

For any subset $L \subset \mathbb{Z}^m$, a Gaussian function on $\mathbb{R}^m$ with center $c$ and parameter $\gamma$ can be defined as

$$\rho_{\gamma, c}(x) = \frac{1}{\sqrt{2\pi\|c\|^2}} e^{-\frac{(x-c)^T(x-c)}{2\|c\|^2}}$$

For the simplicity, we denote $\rho_\gamma(x)$ and $D_{L, \gamma, c}(x)$ when center $c = 0$.

Gentry et al. [16] proved that this distribution can be sampled efficiently for $\gamma \geq \|T\| \cdot \omega(\sqrt{\log n})$ where $T$ is a trapdoor matrix of an $n$-dimensional lattice $\Lambda$ as follows:

SamplePre(A, T, $\gamma$, u) :
For the matrix $A \in \mathbb{Z}_q^{n \times m}$, its trapdoor matrix $T \in \mathbb{Z}_q^{n \times m}$, a real number $\gamma > 0$, and a vector $u \in \mathbb{Z}^m$, this algorithm outputs a sample $\sigma$ from a distribution that is statistically close to $D_{\Lambda^+_L(A), \gamma}$.

The smoothing parameter $\eta_\epsilon(\Lambda)$ of $\Lambda$ enables every coset of $\Lambda$ to get roughly equal mass in the following Lemmas 1 and 2.

Lemma 1. [16] Let $q$ be a prime and $n, m$ be integers with $m > 2n \log q$. Let $f$ be some $\omega(\sqrt{\log n})$ function. Then, there is a negligible function $\epsilon(m)$ such that for all but at most $q^{-n}$ fraction of matrix $A \in \mathbb{Z}_q^{n \times m}$, we have $\eta_\epsilon(m)(\Lambda^+_L(A)) < f(m)$.

Lemma 2. [4] Let $\Lambda \subset \mathbb{R}^n$ be a lattice. Suppose $\rho \geq \eta_\epsilon(\Lambda)$ for some negligible $\epsilon$. Then, we have

$$\Pr \left[ 0 \leq \|v\| \leq 2\rho \sqrt{\frac{n}{2\pi}} : \forall v \leftarrow D_{\Lambda^+_L(A), \gamma} \right] \geq 1 - \text{negl}(n).$$

Lemma 1 declares that a sample vector from SamplePre(A, T, $\gamma$, u) with proper parameters can be extracted uniformly and Lemma 2 determines the upper bound on the length $\|v\|$ of a sample vector $v$ from the Gaussian distribution $D_{\Lambda^+_L(A), \gamma}$.

Wang and Sun [9] suggested a new preimage sampling algorithm GenSamplePre($A_R, A_S, T_S, v, \gamma$) to construct a ring trapdoor function and a ring signature on lattice. They use the idea of the lattice basis delegation technique by Cash et al. [12].

Let $k, k_1, k_2, k_3, k_4$ be positive integers as $k = k_1 + k_2 + k_3 + k_4$. We write $A_S = [A_{S_1} | A_{S_2} | A_{S_3} | A_{S_4}] \in \mathbb{Z}_q^{n \times km}$ where $A_{S_i} \in \mathbb{Z}_q^{n \times (k_i + k_3)m}$ for each $i$ and $A_R = [A_{S_1}, A_{S_3}] \in \mathbb{Z}_q^{n \times (k_1 + k_3)m}$ with its trapdoor $T_R$. Then, one can sample a preimage from a vector $y$ as below:

GenSamplePre($A_S, A_R, T_R, \gamma, y$) :
1. Sample $e_{S_1} \in \mathbb{Z}_q^{n \times 2m}$ and $e_{S_2} \in \mathbb{Z}_q^{n \times km}$.
2. Let $z = y - A_{S_2} e_{S_2} - A_{S_3} e_{S_3}$ and sample $e_R = [e_{S_1} | e_{S_2}] \in \mathbb{Z}_q^{n \times (k_3 + k_4)m}$ from SamplePre($A_R, T_R, \gamma, z$).
3. Output $e = [e_{S_1} | e_{S_2} | e_{S_3} | e_{S_4}]$.

2.4 Lattice Mixing and Vanishing Technique

Boyen [10] proposed the general framework to encode all bits at once by lattice trapdoor mixing and vanishing techniques.

He introduced a new trapdoor generation algorithm TwoSideGen(1$^A$) by slightly modifying Cash et al.'s extending trapdoor basis algorithm ExtBasis(T, B), as below:

TwoSideGen(1$^A$) :
For a security parameter $\lambda$, this algorithm outputs two random matrix $A \in \mathbb{Z}_q^{n \times m}$ and $R \in \mathbb{Z}_q^{n \times m}$ where $A$ is uniform and $R$ is from some distribution $R$. Then, for some $B \in \mathbb{Z}_q^{n \times m}$, $F = [A \mid AR + B] \in \mathbb{Z}_q^{n \times 2m}$ and $q$ defines the public parameters of a two-sided function.

$(F, q)$ is a trapdoor function that samples the preimage with a trapdoor for either $A$ or $B$.

The characteristic of using a two-sided function is that we use the firm preimage trapdoor $T_A$ that can always sample the preimage in the real scheme, whereas we use the fickle preimage trapdoor $T_B$ for a matrix $B$ which sometimes vanishes depending on a given message. With TwoSideGen(1$^A$) algorithm, Boyen constructed a signature $\mathcal{BS}$ to get shorter signatures as below:

B.KeyGen(1$^A$) :
Given a security parameter $\lambda$ and corresponding public parameters $n = n(\lambda), m = m(\lambda), q = q(\lambda)$,

1. Use $\text{TrapGen}(n, m, q)$ to extract a matrix $A_0$ and its trapdoor $T_{A_0}$.
2. Choose $t+1$ random matrix $R_0, R_1, \cdots, R_t \in \mathbb{Z}_q^{n \times m}$ from discrete Gaussian distribution $D_{\mathbb{Z}_q^{nm}}(x)$. 

3
3. Choose $t$ uniformly random integers $h_1, h_2, \ldots, h_t \in \mathbb{Z}_q$ and fix $h_0 = 1 \in \mathbb{Z}_q$.

4. Output a tuple $(A_0, \{C_i = A_i R_{i}, h_i B_0 \}_{i=0}^{t})$ as a public key $pk$ and $T_{A_0}$ as a secret key $sk$.

B.Sign$(sk, v)$:
Given a secret key $sk$ and a message $v \in \{0\} \times \{0,1\}^t$,

1. Define $C_v = \sum_{i=0}^{t} (-1)^{v[i]} C_i$ where $v[i]$ is the $i$-th value of the message $v$ and let a message-dependent matrix $F_v = [A_0 | C_v] \in \mathbb{Z}_q^{n \times 2m}$.

2. Get the extending trapdoor basis $T_F$ of $F_v$ using ExtBasis$(T_{A_0}, F_v)$.

3. Sample a non-zero random vector $d \in \Lambda^\perp(F_v) \subset \mathbb{Z}_q^{2m}$ using SamplePre$(F_v, T_F, v, \gamma)$ and output a signature $\sigma_v = d$.

B.Verify$(pk, v, \sigma)$:
Given a public key $pk$, a message $v$, and a signature $\sigma_v$,

1. Check that $v$ is in $\{0\} \times \{0,1\}^t$ and $\sigma_v$ is a small non-zero vector, i.e., $0 < \|\sigma_v\| \leq \sqrt{2m}$.

2. Check that $\sigma_v$ satisfies that

\[
\left[A_0 | \sum_{i=0}^{t} (-1)^{v[i]} C_i \right] d = 0 \mod q
\]

3. If both are correct, accept the signature. Otherwise, reject.

3 Linearly Homomorphic Signature
We restate the formal definition and security requirements of linearly homomorphic signature over binary fields from Boneh and Freeman's work [4]. Then, we illustrate detailed construction.

3.1 Definition and Security Requirements
Boneh and Freeman [4] defined the linearly homomorphic signature over binary fields $\mathcal{LHS}$ as below:

Definition 2. (Linearly homomorphic signature). A linearly homomorphic signature over binary fields $\mathcal{LHS}$ over $\mathbb{F}_2$ is a tuple of PPT algorithms $\mathcal{LHS} = (\text{Setup}, \text{Sign}, \text{Combine}, \text{Verify})$ with the following functionality:

Setup$(n, \text{params})$:
Given the security parameter $n$ and other public parameters $\text{params}$, this algorithm outputs a public key $pk$ and a secret key $sk$.

Sign$(sk, id, v)$:
Given a secret key $sk$, a tag $id$ and a vector $v$, this algorithm outputs a signature $\sigma$.

Combine$(pk, id, \{(\alpha_i, \sigma_i)\}_{i=1}^{l})$:
Given a public key $pk$, a tag $id$ and pairs $\{(\alpha_i, \sigma_i)\}_{i=1}^{l}$ where $\alpha_i \in \mathbb{F}_2 = \{0,1\}$ and $\sigma_i$ is the signature of a vector $v_i$ for each $i$, this algorithm outputs a signature $\sigma$ for a vector $\sum_{i=1}^{l} \alpha_i v_i$.

Verify$(pk, id, y, \sigma)$:
Given a public key $pk$, a tag $id$, a vector $y$ and a signature $\sigma$, this algorithm outputs either 0 (reject) or 1 (accept).

To check the correctness, for each $(pk, sk)$, we must have

1. For all tags $id$ and all vectors $y$, the verification algorithm $\text{Verify}(pk, id, y, \sigma)$ outputs 1 for all valid signatures $\sigma \leftarrow \text{Sign}(sk, id, y)$.
2. Whenever we operate a linear combination of some vectors $\{v_i\}_{i=1}^{l}$, we can output the valid signature for that linear combination.

The security requirements of linearly homomorphic signature are unforgeability and weakly context hiding property as below:

Definition 3. (Unforgeability). A linearly homomorphic signature is unforgeable if the advantage of any PPT adversary $\mathcal{A}$, in the following security game is negligible in the security parameter $n$.

Setup:
The challenger $\mathcal{C}$ sets $(pk, sk) \leftarrow \text{Setup}(n, \text{params})$, then sends the public key $pk$ to $\mathcal{A}$.

Queries:
Proceeding adaptively, $\mathcal{A}$ specifies a sequence of $k$-dimensional subspaces $V_i$ with basis vectors $\{v^{(i)}_j\}_{j=1}^{k}$.

For each $i$, $\mathcal{C}$ chooses a tag $id_i \leftarrow \{0,1\}^n$ uniformly and gives $id_i$ with $j$ signatures $\sigma_{ij} \leftarrow \text{Sign}(sk, id_i, v_{j}^{(i)})$ for $j = 1, 2, \ldots, k$.

Output:
$\mathcal{A}$ outputs a tag $id^* \in \{0,1\}^n$, a non-zero vector $y^*$, and a signature $\sigma^*$.

$\mathcal{A}$ wins the game if the signature $\sigma$ is valid and either (1) $id^* \neq id_i$ for all $i$, or (2) $id^* = id_i$ for some $i$ but $y^* \notin V_i$.

Definition 4. (Weakly context hiding). A linearly homomorphic signature is weakly context hiding if the advantage of any PPT adversary $\mathcal{A}$, in the following security game is negligible in the security parameter $n$.

Setup:
The challenger $\mathcal{C}$ sets $(pk, sk) \leftarrow \text{Setup}(n, \text{params})$ and sends both public key $pk$ and secret key $sk$ to $\mathcal{A}$.
Challenge:
\( \mathcal{A} \) outputs two \( k \)-dimensional vector spaces \( V_0, V_1 \) with basis vectors \( \{v_i^{(0)}\}_{i=1}^k \) and \( \{v_i^{(1)}\}_{i=1}^k \), respectively and linear functions on both \( \{v_i^{(0)}\}_{i=1}^k \) and \( \{v_i^{(1)}\}_{i=1}^k \) which satisfies
\[
f_j \left( v_1^{(0)}, v_2^{(0)}, \ldots, v_k^{(0)} \right) = f_j \left( v_1^{(1)}, v_2^{(1)}, \ldots, v_k^{(1)} \right)
\]
for all \( j = 1, 2, \ldots, s \).
\( \mathcal{C} \) chooses \( b \in \{0, 1\} \) and a tag \( id \in \{0, 1\}^n \) and signs the vector space \( V_b \) with that tag id.
Then, \( \mathcal{C} \) uses Combine\((pk, id, (\{\alpha_i, \sigma_i\})_{i=1}^k)\) algorithm to derive signatures \( \sigma_j \) of the function \( f_j \left( v_1^{(b)}, v_2^{(b)}, \ldots, v_k^{(b)} \right) \) for all \( j = 1, 2, \ldots, s \).
\( \mathcal{A} \) gets signatures \( \sigma_j \). The function can be out adaptively after choosing \( V_0 \) and \( V_1 \).

Output:
\( \mathcal{A} \) outputs a bit \( b' \).
\( \mathcal{A} \) wins the game if \( b = b' \).

3.2 Construction by Boneh and Freeman

We let the public parameters \( \text{params} = (N, k, L, m, q, \gamma) \) where \( N = n \) is the dimension of vectors to be signed, \( k \) is the dimension of the subspace to be signed (\( k < N \)), \( L \) is the maximum number of signatures in linear combinations, \( m(n, L) > n \) is an integer, \( q(n, L) \) is an odd prime, and \( \gamma(n, L) \) is a real number.

With those parameters, Boneh and Freeman [4] presented the first linearly homomorphic signature over binary fields with a tuple of PPT algorithms \( \mathcal{LHS} = (\text{Setup}, \text{Sign}, \text{Combine}, \text{Verify}) \) with the following functionality:

\text{Setup}(n, \text{params}) :
Given a security parameter \( n \) and public parameters \( \text{params} = (N, k, L, m, q, \gamma) \),
1. \( (A, T) \leftarrow \text{TrapGen}(n, m, 2q) \) where a matrix \( A \in \mathbb{Z}_{2q}^{n \times m} \) and its trapdoor basis \( T \) of \( \Lambda_2^n(A) \) satisfies that \( \|T\| \leq 30\sqrt{n \log 2q} \).
2. Let \( H : \{0, 1\}^* \rightarrow \mathbb{Z}_{2q}^{n \times m} \) be a hash function, viewed as a random oracle.
3. Output the public key \( pk = (A, H) \) and the secret key \( sk = (A, H, T) \).

\text{Sign}(sk, id, v) :
Given a secret key \( sk = (A, H, T) \), a tag \( id \in \{0, 1\}^n \) and a vector \( v \in \mathbb{F}_2^m \),
1. Set \( B \leftarrow A||H(id)\in\mathbb{Z}_{2q}^{n \times 2m} \).
2. Let \( S \leftarrow \text{ExtBasis}(T, B) \) be a basis for \( \Lambda_2^n(B) \) with \( \|S\| = \|T\| \).
3. Output \( \sigma \leftarrow \text{SamplePre}(B, S, \gamma, q \cdot v) \).

\text{Combine}(pk, id, ((\alpha_i, \sigma_i))_{i=1}^k) :
Given a public key \( pk = (A, H) \), a tag \( id \in \{0, 1\}^n \) and pairs \( ((\alpha_i, \sigma_i))_{i=1}^k \) where \( \alpha_i \in \mathbb{F}_2 \) and \( \sigma \) is a signature of the \( i \)-th vector \( v_i \), output \( \sigma \leftarrow \Sigma_{i=1}^k \alpha_i \sigma_i \in \mathbb{Z}_{2^m} \).

\text{Verify}(pk, id, y, \sigma) :
Given a public key \( pk = (A, H) \), a tag \( id \in \{0, 1\}^n \), a vector \( y \in \mathbb{F}_2^m \) and a signature \( \sigma \in \mathbb{Z}_{2^m} \),
1. Set \( B \leftarrow [A | H(id)] \in \mathbb{Z}_{2q}^{n \times 2m} \).
2. If \( \|\sigma\| \leq L \cdot \gamma \sqrt{2m} \) and \( B \cdot \sigma = q \cdot y \mod 2q \), output 1 (accept). Otherwise, output 0 (reject).

Lemma 3. Let \( \mathcal{LHS} \) be the linearly homomorphic signature over \( \mathbb{F}_2 \) as above. Suppose \( q \) be a prime, \( n, m \) be integers with \( m > 2n \log q \) and \( \gamma > 30\sqrt{n \log 2q} \cdot \omega(\sqrt{\log n}) \). Then \( \|\sigma\| \leq L \cdot \gamma \sqrt{2m} \) and \( B \cdot \sigma = q \cdot y \mod 2q \) for all valid signatures \( \sigma \leftarrow \text{Combine}(pk, id, ((\alpha_i, \sigma_i))_{i=1}^k) \).

Moreover, Lemmas 4 and 5 from Boneh and Freeman’s work show that this signature is unforgeable in the random oracle model and it holds the weakly context hiding property [4].

Lemma 4. Let \( \mathcal{LHS} \) be the linearly homomorphic signature over \( \mathbb{F}_2 \) as above. Suppose that \( m = [6n \log 2q] \) and \( \gamma = 30\sqrt{n \log 2q} \cdot \log n \). Then \( \mathcal{LHS} \) is unforgeable in the random oracle model assuming that \( k \)-SIS\(_{2m, \beta, \gamma} \) problem is infeasible.

Lemma 5. Let \( \mathcal{LHS} \) be the linearly homomorphic signature over \( \mathbb{F}_2 \) as above. Suppose that \( k < \frac{n \log 2q}{2 \log \log n} \).
Then \( \mathcal{LHS} \) is weakly context hiding.

4 Design of New Linearly Homomorphic Signatures

In this section, we define the linearly homomorphic ring signature and its security requirements and design the linearly homomorphic ring signature using a new preimage sampling algorithm \( \text{GenSamplePre}(A_R, A_s, T_s, v, \gamma) \) by Wang and Sun [9]. We also construct the signature scheme with short signature using lattice mixing and vanishing technique by Boyen [10].

We set the public parameters \( \text{params} = (N, k, L, m, q, \gamma) \) same as the signature by Boneh and Freeman in Section 3.2.

4.1 Linearly Homomorphic Ring Signature

In a ring signature, a signer chooses any subset of all possible signers including himself/herself to form a ring, without getting their permission [17]. Thus, ring signature provides the anonymity of the signer since the signature of the message only convinces that one member in the ring signed the message without revealing
a signer’s identity. We define the linearly homomorphic ring signature using a new preimage sampling algorithm GenSamplePre($A_R, A_S, T_S, v, \gamma$) by Wang and Sun [9] as below:

**Definition 5.** (linearly homomorphic ring signature). A linearly homomorphic ring signature $LHRS$ is a tuple of PPT algorithms $LHRS = (R.\text{Setup}, R.\text{Sign}, R.\text{Combine}, R.\text{Verify})$ with the following functionality:

- **R.\text{Setup}(n, \text{params})**: Given the security parameter $n$ and public parameters $\text{params}$, this algorithm outputs a public key $pk$ and a secret key $sk$.

- **R.\text{Sign}(pk, sk, id, R, v)**: Given a key pair $(pk, sk)$ of a signer where $pk \in R$, a tag $id$, a public key $R$ of the ring, and a vector $v$, this algorithm outputs a signature $\sigma$ of the vector $v$ under $sk$.

- **R.\text{Combine}(R, id, \{(\alpha_i, \sigma_i)\}_{i=1}^l)**: Given a public key $R$ of the ring, a tag $id$, and pairs $\{(\alpha_i, \sigma_i)\}_{i=1}^l$ where $\alpha_i \in \mathbb{F}_2 = \{0,1\}$ and $\sigma_i$ is the signature of a vector $v_i$ for each $i$, this algorithm outputs a signature $\sigma$ for a vector $\Sigma_{i=1}^l \alpha_i v_i$.

- **R.\text{Verify}(R, id, y, \sigma)**: Given a public key $R$ of the ring, a tag $id$, a vector $y$, and a signature $\sigma$, this algorithm outputs either 0 (reject) or 1 (accept).

To check the correctness, for each $(pk, sk)$, we must have

a. For all key pairs $(pk_i, sk_i)$ where $pk_i \in R$, tags id, and all vectors $y$, the verification algorithm $\text{Verify}(R, id, y, \sigma)$ outputs 1 for all valid signatures $\sigma \leftarrow \text{Sign}(pk_i, sk_i, id, R, y)$.

b. Whenever we operate a linear combination of some vectors $\{v_i\}_{i=1}^l$, we can output the valid signature for that linear combination.

The security requirements of linearly homomorphic signature are unforgeability and weakly context hiding property for linearly homomorphic ring signatures as well as anonymity like other ring signature. Here, we only give the formal definition of the anonymity since the others are analogous to the one for linearly homomorphic signatures.

**Definition 6.** (anonymity). A linearly homomorphic ring signature is anonymous if the advantage of any PPT adversary $A$, in the following security game is negligible in the security parameter $n$.

**Setup**:

The challenger $C$ obtains $(pk_i, sk_i) \leftarrow R.\text{Setup}(n, \text{params})$ $r$ times where $r$ is the size of the ring $R$, then sends public keys $\{pk_i\}_{i=1}^r$ to $A$.

**Queries**:

$A$ specifies the pair $(i, R, v)$ where $i$ is a signer index, $R$ is a public key of the ring $R$, and $v$ is a vector to be signed. Then, the challenger $C$ chooses a tag $id_i \leftarrow \{0,1\}^n$ uniformly and gives $id_i$ with a signature $\sigma_i \leftarrow R.\text{Sign}(pk, sk, id_i, R, v)$.

**Challenge**:

$A$ requests a challenge by sending $(i_0, i_1, R^*, v^*)$ to $C$, where $i_0$ and $i_1$ are signer indices, $R^*$ is a public key of the ring $R^*$ which contains $pk_{i_0}$ and $pk_{i_1}$, and $v^*$ is a vector to be signed. $A$ non-zero vector $y^*$, and a signature $\sigma^*$. Then, $C$ chooses a bit $b \leftarrow \{0,1\}$ and a tag $id^* \leftarrow \{0,1\}^n$ and sends a challenge signature $\sigma_b \leftarrow R.\text{Sign}(pk_{i_b}, sk_{i_b}, id^*, R^*, v^*)$ to the adversary $A$.

**Output**:

$A$ outputs a bit $b'$.

$A$ wins the game if $b = b'$.

With these definition and security requirements, we design the linearly homomorphic ring signature on lattice as below:

**R.\text{Setup}(n, g, \text{params})**: Given a security parameter $n$, a number of all possible signers $g$, and public parameters $\text{params} = (N, k, L, m, q, \gamma)$, do the following:

1. Run $\text{TrapGen}(n, m, 2q)$ to generate a matrix $\{A_i\}_{i=1}^g \in \mathbb{Z}_{2q}^{n \times m}$ and its corresponding trapdoor basis $\{T_i\}_{i=1}^g$ of $A_i$ such that $\|T_i\| \leq 30\sqrt{n \log 2q}$.

2. Let $H : \{0,1\}^* \rightarrow \mathbb{Z}_{2q}^{n \times m}$ be a hash function, viewed as a random oracle and choose the ring $R$.

3. Output the public key $pk_i = (A_i, H)$ and the secret key $sk_i = T_i$ for each signer $i$ of the ring and $R$ is a subset of public keys of all possible signers including $pk_i$ to form a ring $R$.

**R.\text{Sign}(pk, sk, id, R, v)**: For a key pair $(pk, sk_i) = (A_i, T_i)$ of a signer where $pk_i \in R$ when the size of the ring is $r$, a tag $id \in \{0,1\}^n$, and a vector $v \in \mathbb{F}_2^n$, do the following:

1. Set a matrix $A_R = [A_1 \mid A_2 \mid \cdots \mid A_r \mid H(id)] \in \mathbb{Z}_{2q}^{n \times (r+1)m}$.

2. Output a signature $\sigma \leftarrow \text{GenSamplePre}(A_R, A_S, T_S, q, \gamma)$.

**R.\text{Combine}(R, id, \{(\alpha_j, \sigma_j)\}_{j=1}^l)**: Given a public key $R$ of the ring of size $r$, a hash function $H$, a tag $id \in \{0,1\}^n$, and set of signatures $\{(\alpha_i, \sigma_i)\}_{i=1}^l$, output $\sigma = \sum_{j=1}^l \alpha_i \sigma_i \in \mathbb{Z}^{(r+1)m}$. 

6
R. Verify \((R, H, id, y, \sigma)\) :
Given a public key \(R\) of the ring with the size \(r\), a hash function \(H\), a tag \(id \in \{0, 1\}^n\), a vector \(y \in \mathbb{F}_2^r\), and a signature \(\sigma \in \mathbb{Z}^{r+1m}\), do the following:

1. Set a matrix \(A_R = [A_1 | A_2 | \cdots | A_r | H(id)] \in \mathbb{Z}_{2^q}^{n \times (r+1)m}\).
2. If \(\|\sigma\| \leq L \cdot \gamma \sqrt{(r+1)m}\) and \(A_R \cdot \sigma = q \cdot y \mod 2q\), output 1 (accept). Otherwise, output 0 (reject).

We believe that the above construction holds the linearly homomorphic property for signatures from the same ring \(R\).

4.2 Linearly Homomorphic Signature with Short Signatures

We design a new linearly homomorphic signature with short signatures as a tuple of PPT algorithms \(\mathcal{S}\mathcal{L}\mathcal{H} = (\text{S.Setup, S.Sign, S.Combine, S.Verify})\) by adopting lattice mixing and vanishing technique by Boyen [10] as below:

\textbf{S.Setup}(n, params) :
Given a security parameter \(n\) and public parameters \(\text{params} = (N, k, L, m, q, \gamma)\), do the following:

1. Use \(\text{TrapGen}(n, m, 2q)\) to extract a trapdoor \(T_{A_0}\) such that \(\|T_{A_0}\| \leq 30q/m \log 2q\).
2. Choose \(n+1\) random matrix \(R_0, R_1, \cdots, R_n \in \mathbb{Z}^{m \times m}\) from discrete Gaussian distribution \(\mathcal{D}_{\mathbb{Z}^m, \gamma}(x)\).
3. Choose \(n\) uniformly random integers \(h_1, h_2, \cdots, h_n \in \mathbb{Z}_{2q}\) and fix \(h_0 = 1 \in \mathbb{Z}_{2q}\).
4. Let \(H : \{0, 1\}^n \rightarrow \mathbb{Z}_{2q}^{n \times m}\) be a hash function, viewed as a random oracle.
5. Calculate \(C_i = A_i R_i + h_i B_0\) for each \(i\) and output a public key \(pk = (A_0, \{C_i\}_{i=0}^n, H)\) and a secret key \(sk = T_{A_0}\).

\textbf{S.Sign}(sk, id, v) :
Given a secret key \(sk = T_{A_0}\), a tag \(id \in \{0, 1\}^n\) and a vector \(v \in \{0, 1\}^n\), do the following:

1. Define \(C_v = C_0 + \sum_{i=1}^n (-1)^i v[i] C_i\) where \(v[i]\) is the \(i\)-th value of the message \(v\) and set \(F_v = [A_0 | C_v | H(id)] \in \mathbb{Z}^{n \times 3m}\) as a message-dependent matrix.
2. Get the extending trapdoor basis \(T_F\) of \(F_v\) using \(\text{ExtBasis}(T_{A_0}, F_v)\).
3. Sample a non-zero random vector \(d \in \mathbb{Z}^{3m}\) using \(\text{SamplePre}(F_v, T_F, q, \gamma)\) and output a signature \(\sigma_v = d\) of the vector \(v\).

\textbf{S.Combine}(pk, id, \{(\alpha_j, \sigma_j)\}_{j=1}^k) :
Given a public key \(pk = (A_0, \{C_i\}_{i=0}^n, H)\), a tag \(id \in \{0, 1\}^n\), and pairs \(\{(\alpha_j, \sigma_j)\}_{j=1}^k\) where \(\alpha_i \in \{0, 1\}\) and \(\sigma_j\) is a signature of the \(j\)-th vector \(v_j\), output \(\sigma = \sum_{j=1}^k \alpha_j \sigma_j \in \mathbb{Z}^{3m}\).

\textbf{S.Verify}(pk, id, y, \sigma) :
Given a public key \(pk = (A, H)\), a tag \(id \in \{0, 1\}^n\), a vector \(y \in \{0, 1\}^n\), and a signature \(\sigma \in \mathbb{Z}^{3m}\), do the following:

1. Check that \(y\) is in \(\{0, 1\}^n\) and \(\sigma_v\) is a small non-zero vector, i.e., \(0 < \|\sigma_v\| \leq L \cdot \gamma \sqrt{3m}\).
2. Set \(C_y = C_0 + \sum_{i=1}^n (-1)^i v[i] C_i\) and check that \(\sigma_v\) satisfies that \([A_0 | C_y | H(id)] d = q \cdot y \mod 2q\).
3. If both are correct, accept the signature. Otherwise, reject.

We expect that the above construction extracts the shorter signatures than the signature from Boneh and Freeman’s signature scheme.

5 Concluding Remark

We have two different methods to enhance lattice-based linearly homomorphic signature scheme over binary fields by modifying linearly homomorphic signature by Boneh and Freeman [4]. We design the linearly homomorphic ring signature and linearly homomorphic signature with short signatures.

But, we haven’t proved the validity of the suggested schemes or their security analysis yet. Thus, first of all, we have to give the concrete proof of designed signatures so that these signatures can be safely applicable to the real world scenario.

Aside from this, it will be also challenging to design new fully homomorphic signatures by adding new functionalities to the existing homomorphic trapdoor functions like lattice mixing and vanishing technique used in this paper or by changing the hard problems on lattice from SIS problem to Ring-SIS, LWE or LWR problems.

Acknowledgement

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. NRF-2015R1A2A2A01000812).

References


