SCIS 2017 2017 Symposium on
Cryptography and Information Security
Naha, Japan, Jan. 24 - 27, 2017
The Institute of Electronics,
Information and Communication Engineers

A Classification of Lattice-based Trapdoor Functions

Rakyong Choi * Kwangjo Kim *

Abstract: A trapdoor function is a one-way function with trapdoor, which is indispensable for getting a preimage of the function. In lattice-based cryptography, trapdoor function plays an important role in constructing the secure cryptographic schemes like identity-based encryption, homomorphic encryption, or homomorphic signature. There are three categories of trapdoor functions as standard trapdoor, lossy trapdoor, and homomorphic trapdoor functions. Lossy trapdoor function is a trapdoor function that behaves in two ways as a standard trapdoor function or by losing information from the input. Homomorphic trapdoor function can evaluate the computation of trapdoor function results.

In this paper, we survey all the public literature on lattices studying lattice-based trapdoor functions and their preimage sampling algorithms to the best of our knowledge. Then, we classify these trapdoor functions into three categories depending on their cryptographic features and suggest their feasibility to design cryptographic primitives.

Keywords: lattice-based trapdoor function, (preimage) sampling algorithm, lossy trapdoor function, homomorphic trapdoor function

1 Introduction

1.1 Background and Motivation

After Ajtai's seminal work [1] on lattices, lattices have become one of the essential tools in cryptology. Lattice-based cryptography is one candidate of post quantum cryptography, which remains secure against the upcoming quantum computer. It has the advantage of the security based on the worst-case hardness assumptions instead of average-case hardness assumptions. With these advantages, lattices have been applied to various areas in cryptology recently such as identity-based encryption [2], fully homomorphic encryption [3–5], multilinear maps [6], and homomorphic signatures [7–9].

A trapdoor function samples a preimage of a given output with a trapdoor but remains as a one-way function without its trapdoor. To construct cryptographic primitives on lattices and prove their security rigorously, we should find the proper lattice-based trapdoor function. In order to choose the proper trapdoor function, we study the literature on lattice-based cryptography. Then, we categorize the types of trapdoor functions depending on their cryptographic features and suggest their feasibility to design cryptographic primitives.

1.2 Outline of the Paper

In this work, we provide the formal definition of three types of trapdoor functions and their applications. Section 2 describes a notation in this paper and some background on lattices including hard lattice problems and discrete Gaussian distribution. In Section 3, we give a formal definition of trapdoor functions and construction of Gentry *et al.*'s lattice-based trapdoor function. Then, we introduce the various trapdoor functions with their applications.

The description and applications of lossy trapdoor functions and homomorphic trapdoor functions are discussed in Sections 4 and 5, respectively. Finally, we give concluding remarks and possible directions for future work in Section 6.

2 Preliminaries

2.1 Notation

We denote vectors with small bold letters (*e.g.*, $\mathbf{x}$, $\mathbf{y}$) and matrices with large bold letters (*e.g.*, $\mathbf{A}$, $\mathbf{B}$). We denote $\mathbb{R}$ and $\mathbb{Z}$ to express the set of real numbers and the set of integers, respectively, and small non-bold letters to 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 $[\mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n\times(m_1+m_2)}$ and a transpose of $\mathbf{A}$ as $\mathbf{A}^T$.

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

2.2 Hard Problems on Lattices

A lattice Λ can be defined as a discrete subgroup of $\mathbb{R}^m$ with its basis $\mathcal{B}$. A basis $\mathcal{B}$ of Λ is a set of linearly

* School of Computing, KAIST. 291, Daehak-ro, Yuseong-gu, Daejeon, South Korea 34141. {thepride, kkj}@kaist.ac.kr

independent vectors $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_m\}$ which spans the lattice Λ and $\mathbf{B} = (\mathbf{b}_1|\mathbf{b}_2|\cdots|\mathbf{b}_m)$ is called a basis matrix of the lattice Λ.

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

Shortest Vector Problem (SVP) is to find the nonzero lattice vector $\mathbf{v}$ which is the closest to the origin, in a lattice Λ, and Closest Vector Problem (CVP) is to find the lattice vector $\mathbf{v}$ in a lattice Λ which is the closest to the vector $\mathbf{w}$. Both SVP and CVP are considered as a worst-case hardness problem in lattice. On the other hand, there are two popular average-case hardness problems in lattice which can be reduced to the aforementioned worst-case hardness problem in lattice.

One is called Short Integer Solution (SIS) problem, which can be used to construct one-way functions or signature schemes and the other is Learning With Errors (LWE) problem whose decisional version is used for guaranteeing the security of encryption schemes like identity based encryption and fully homomorphic encryption schemes. There are also ring variant of these problems. The formal definition of SIS and LWE problems are given 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 $\Lambda_q^{\perp}(\mathbf{A}) = \{\mathbf{e} \in \mathbb{Z}^m | \mathbf{A}\cdot\mathbf{e} = \mathbf{0} \bmod q, \}$, it is hard to find a small vector $\mathbf{e} \in \Lambda_q^{\perp}(\mathbf{A})$, such that $\|\mathbf{e}\| \leq \beta$ for some $\beta \geq \sqrt{n\log q}$ and $\mathbf{A}\cdot\mathbf{e} = 0 \pmod q$, whose coefficients are $-1, 0,$ or 1.

Definition 2. (LWE problem) For a positive integer m, n with $m > n$, integer $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution $\mathcal{D}$ on the interval $[0, q)^m$, it is hard to distinguish between uniformly chosen $(\mathbf{A}, \mathbf{y}) \leftarrow \mathbb{Z}_q^{n\times m} \times [0, q)^m$ and the sampling $(\mathbf{A}, \mathbf{A}^T\mathbf{s}+\mathbf{e} \mod q)$ where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and $\mathbf{e} \leftarrow \mathcal{D}$.

2.3 Discrete Gaussian Distribution

Given L as any subset of $\mathbb{Z}^m$, a Gaussian function on $\mathbb{R}^m$ with center $\mathbf{c}$ and a parameter γ can be defined as $\rho_{\gamma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\|\mathbf{x}\text{-}\mathbf{c}\|^2/\gamma^2)$ for any vector $\mathbf{c} \in \mathbb{R}^m$ and any positive parameter $\gamma > 0$,

For a subset $L \subset \mathbb{Z}^m$, we can define *discrete Gaussian distribution*, which is the m-dimensional Gaussian distribution whose support is restricted to the subset L and its density function is defined as

$$\mathcal{D}_{L,\gamma,\mathbf{c}}(\mathbf{x}) = \frac{\rho_{\gamma,\mathbf{c}}(\mathbf{x})}{\sum_{\mathbf{y}\in L}\rho_{\gamma,\mathbf{c}}(\mathbf{y})}$$

and for the sake of simplicity, we denote $\rho_\gamma(\mathbf{x})$ and $\mathcal{D}_{L,\gamma}(\mathbf{x})$ when a center $\mathbf{c} = \mathbf{0}$.

3 Trapdoor Functions

We give a formal definition and security requirements to make trapdoor functions and the concrete construction of lattice-based trapdoor functions by Gentry *et al.* [10]. Then, we discuss several trapdoor functions on lattices and their applications in cryptographic protocols.

3.1 Lattices and Trapdoor Functions

Gentry *et al.* [10] proposed the notion of preimage sampleable (trapdoor) functions whose preimage can be efficiently sampled using the trapdoor as below:

TrapGen(n) :
Output a description $\mathbf{A}$ for a function $f_A : D_n \to R_n$ and its trapdoor information $\mathbf{T}$. In the context of lattice-based cryptography, a description $\mathbf{A}$ is a matrix for a lattice $\Lambda_{\mathbf{A}}$ and a trapdoor information $\mathbf{T}$ is a short basis for $\Lambda_{\mathbf{A}}$.

SampleDom(n) :
Sample an $\mathbf{e}$ from a distribution over the domain D_n where the output distribution $f_{\mathbf{A}}(\mathbf{e}) \leftarrow R_n$ is uniform.

SamplePre(A, T, y) :
Sample a preimage $\mathbf{x} \in D_n$ such that $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{y} \in R_n$.

This function behaves as a one-way function without trapdoor. they also defined the collision-resistance of the trapdoor functions as the one for hash functions.

Alwen and Peikert [11] proposed the lattice-based trapdoor generation algorithm **L.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:

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

Without loss of generality, we assume that a matrix $\mathbf{A} \leftarrow$ **L.TrapGen**(n, m, q) has a full rank. Then, the distribution of the output of **L.TrapGen**(n, m, q) can be sampled efficiently for $\gamma \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$ where $\mathbf{T}$ is a trapdoor matrix of an n-dimensional lattice Λ as below:

L.SamplePre(A, T, γ, b) :
This is a preimage sampling algorithm for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, its trapdoor matrix $\mathbf{T} \in \mathbb{Z}_q^{m\times m}$, a real number $\gamma > 0$, and a vector $\mathbf{b} \in \mathbb{Z}^n$. This algorithm outputs a sample $\mathbf{s}$ from a distribution that is statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\gamma}$.

The above two algorithms $\mathbf{L.TrapGen}(n, m, q)$ and $\mathbf{L.SamplePre}(\mathbf{A}, \mathbf{T}, \gamma, \mathbf{b})$ construct a collision-resistant trapdoor function assuming SIS problem in lattices with discrete Gaussian distribution and can be essential tools to make lattice-based signature schemes.

3.2 Application of Trapdoor Functions

Following Gentry *et al.*'s work [10], there are variants of techniques to use trapdoor functions.

In 2009, Peikert [12] introduced the notion of chosen-output security for trapdoor functions to construct a secure cryptosystem against a chosen-ciphertext attack. To get this property, a new deterministic polynomial-time algorithm $\mathbf{Ver}(\mathbf{A}, \mathbf{s}, \mathbf{b})$ called a preimage verifier was introduced. Then, we say a trapdoor function is *chosen-output secure* if it satisfies the following:

1. (Completeness) For any $\mathbf{s} \in D_n$, $\mathbf{Ver}(\mathbf{A}, \mathbf{s}, \mathbf{b})$ accepts with overwhelming probability when $f_{\mathbf{A}}(\mathbf{s}) = \mathbf{b}$.
2. (Uniqueness) For any $\mathbf{b} \in R_n$, there is at most one value of $\mathbf{s}$ where $f_{\mathbf{A}}(\mathbf{s}) = \mathbf{b}$ is accepted.
3. (Correctness) For any $\mathbf{b} \in R_n$, b has a valid preimage $\mathbf{s}$. *i.e.* $\mathbf{s} \leftarrow \mathbf{SamplePre}(\mathbf{A}, \mathbf{T}, \gamma, \mathbf{b})$.

Cash *et al.* [13] introduce the technique to extend the basis to higher dimension in the concept of bonsai trees using the following algorithm.

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

The above algorithm indicates the controlled growth of the bonsai tree and the hierarchy of the lattices has a well-quasi-ordering where any short basis of a parent lattice can be easily extended to a short basis of any higher-dimensional child lattice.

They showed that this technique can be applicable to a hash-then-sign signature without random oracles and a hierarchical identity-based encryption scheme.

Boyen [14] proposed the general framework to encode all bits at once by lattice trapdoor mixing and vanishing techniques.

In that signature scheme, they introduced a new generation algorithm $\mathbf{TwoSideGen}(1^{\lambda})$ by slightly modifying Cash *et al.*'s extending basis algorithm.

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

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

The benefit of using a two-sided function is that we use the "firm" preimage trapdoor $\mathbf{T_A}$ that can always sample the preimage in the real scheme, whereas we use the "fickle" preimage trapdoor $\mathbf{T_B}$ for a matrix $\mathbf{B}$ which sometimes "vanishes" depending on a given message.

In the signature scheme, they generated $l+1$ random matrices $\mathbf{C}_0, \mathbf{C}_1, \cdots, \mathbf{C}_l$ for a message $\mathbf{m}$ with length l and "mixed" them as $\mathbf{C_m} = \sum_{i=0}^{l} (-1)^{\mathbf{m}_i} C_i$ where $\mathbf{m}_i$ is the i-th value of the message $\mathbf{m}$. Then, they built practical and fully secure signatures and identity based encryption in the standard model by achieving this encoding framework.

Brakerski and Kalai [2] introduced the formal definition of a *ring trapdoor function* and gave a generic construction of ring signature scheme from a ring trapdoor function. We present a ring homomorphic function for lattices which satisfies the following:

1. **Sampling a function**:
   We can sample a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and the function $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{Ax}$.
2. **Sampling a trapdoor**:
   We can generate a pair of a random matrix $\mathbf{A}$ and its trapdoor $\mathbf{T}$ using the trapdoor generation algorithm $\mathbf{L.TrapGen}(n, m, q)$.
3. **Trapdoor property**:
   Given $\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_t \in \mathbb{Z}_q^{n \times m}$, a trapdoor $\mathbf{S}_i$ for some $\mathbf{A}_i$, and a vector $\mathbf{v} \in \mathbb{Z}_q^n$, we can sample $\mathbf{x}_j$ from a distribution $\mathcal{X}$ for all $j \neq i$ and use $\mathbf{S}_i$ to sample $\mathbf{x}_i$ from $\mathcal{X}$ such that $\mathbf{A}_i \mathbf{x}_i = \mathbf{v} - \sum_{j \neq i} \mathbf{A}_j \mathbf{x}_j$.
4. **Ring one-wayness**:
   The above trapdoor function is a one-way function assuming the hardness of SIS problem.

Wang and Sun [15] also designed a ring signature using a different sampling algorithm for ring trapdoor functions.

They introduced a new preimage sampling algorithm $\mathbf{GenSamplePre}(\mathbf{A}_S, \mathbf{A}_R, \mathbf{T}_R, \gamma, \mathbf{y})$ using the lattice basis delegation technique.

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 $\mathbf{A}_S = [\mathbf{A}_{S_1} \mid \mathbf{A}_{S_2} \mid \mathbf{A}_{S_3} \mid \mathbf{A}_{S_4}] \in \mathbb{Z}_q^{n \times km}$ where $\mathbf{A}_{S_i} \in \mathbb{Z}_q^{n \times k_i m}$ for each i and $\mathbf{A}_R = [\mathbf{A}_{S_1} \mid \mathbf{A}_{S_3}] \in \mathbb{Z}_q^{n \times (k_1 + k_3)m}$ and its trapdoor $\mathbf{T}_R$. Then, one can sample a preimage from a vector $\mathbf{y}$ as below:

GenSamplePre($\mathbf{A}_S, \mathbf{A}_R, \mathbf{T}_R, \gamma, \mathbf{y}$) :

a. Sample $\mathbf{e}_{S_2} \in \mathbb{Z}_q^{n \times k_2 m}$ and $\mathbf{e}_{S_4} \in \mathbb{Z}_q^{n \times k_4 m}$.
b. Let $\mathbf{z} = y - \mathbf{A}_{S_2} \mathbf{e}_{s_2} - \mathbf{A}_{S_4} \mathbf{e}_{s_4}$ and sample $\mathbf{e}_R = [\mathbf{e}_{S_1} \mid \mathbf{e}_{S_3}] \in \mathbb{Z}_q^{n \times (k_1 + k_3)m}$ from $\mathbf{L.SamplePre}\ (\mathbf{A}_R, \mathbf{T}_R, \gamma, \mathbf{z})$.
c. Output $\mathbf{e} = [\mathbf{e}_{S_1} \mid \mathbf{e}_{S_2} \mathbf{e}_{S_3} \mid \mathbf{e}_{S_4}]$.

Compared to Brakerski and Kalai's work [2], Wang and Sun's work gives ring signature scheme in the random oracle model.

Micciancio and Peikert [16] gave a new concept of trapdoors called a *"gadget" trapdoor* and suggested simpler and more efficient algorithms for inverting LWE, sampling SIS preimages, and delegating basis securely.

They defined a *primitive matrix* $\mathbf{G} \in \mathbb{Z}_q^{n \times w}$ satisfying $\mathbf{G} \cdot \mathbb{Z}^w = \mathbb{Z}_q^n$ and constructed a lattice with $\mathbf{G}$. Then, given $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{G} \in \mathbb{Z}_q^{n \times w}$, they introduced a new notion of ***G**-trapdoor* $\mathbf{R} \in \mathbb{Z}_q^{(m-w) \times w}$ and *tag (label)* of the trapdoor $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$ where $\mathbf{A}\begin{bmatrix}\mathbf{R}\\ \mathbf{I}\end{bmatrix} = \mathbf{HG}$ and $\mathbf{H}$ is invertible.

There are efficient algorithms for generating trapdoors, inverting LWE, sampling SIS preimages, and delegating basis securely. Among them, we describe the new trapdoor generation algorithm by Micciancio and Peikert [16] as below:

MP.TrapGen($\mathbf{A}^*, \mathbf{H}$) :
Given a matrix $\mathbf{A}^* \in \mathbb{Z}_q^{n \times m^*}$ and an invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$, we can generate a matrix $\mathbf{A} = [\mathbf{A}^* \mid \mathbf{HG} - \mathbf{A}^*\mathbf{R}] \in \mathbb{Z}_q^{n \times m}$ for a matrix $\mathbf{R} \in \mathbb{Z}_q^{m^* \times w}$.

Since the size of a $\mathbf{G}$-trapdoor grows linearly in the lattice dimension m, the basis delegation algorithm becomes much more efficient than previous work by Cash *et al.* [13] and derive faster implementation on lattice-based cryptography such as threshold protocols for lattice-based signatures or hierarchical identity-based encryption [17].

4 Lossy Trapdoor Function

In this section, we give a formal definition and security requirements of lossy trapdoor functions and discuss several applications of lossy trapdoor functions from the literature.

4.1 Definition and Security Requirements

In 2011, Peikert and Waters [18] proposed a general cryptographic primitive called *lossy trapdoor functions* and its construction based on Diffie-Hellman problem and LWE problem.

As a security parameter, $n(\lambda) = \text{poly}(\lambda)$ represents the input length of the trapdoor function, $k(\lambda) \leq n(\lambda)$ represents the lossiness of the collection, and $r(\lambda) = n(\lambda) - k(\lambda)$ represents the residual leakage.

Then, (n, k)-lossy trapdoor function is a tuple of poly-time algorithms $(S_{ltdf}, F_{ltdf}, F_{ltdf}^{-1})$ with the following functionality:

S_{ltdf} : $S_{inj}(\cdot) = S_{ltdf}(\cdot, 1)$ generates a description $\mathbf{A}$ of a function $f_{\mathbf{A}}$ and its trapdoor information $\mathbf{T}$ like the normal trapdoor function. But, $S_{lossy}(\cdot) = S_{ltdf}(\cdot, 0)$ outputs a description $\mathbf{A}$ of a function $f_{\mathbf{A}}$ but $\perp$ for trapdoor information.

F_{ltdf} : $F_{ltdf}(\mathbf{A}, \cdot)$ computes an injective function $f_{\mathbf{A}}(\cdot)$ over the domain $\{0,1\}^n$.

F_{ltdf}^{-1} : $F_{ltdf}^{-1}(\mathbf{T}, \cdot)$ computes $f_{\mathbf{A}}^{-1}(\cdot)$. If a value $y \in \{0,1\}^n$ doesn't have a preimage, then the behavior of $F_{ltdf}^{-1}(\mathbf{T}, \cdot)$ is not clear and thus, we need to check the correctness.

For the security of lossy trapdoor functions, the distribution of the output of $S_{inj}(\cdot) = S_{ltdf}(\cdot, 1)$ and the output of $S_{lossy}(\cdot) = S_{ltdf}(\cdot, 0)$ should be hard to distinguish.

In lattice-based constructions, they use the relaxed term called "almost-always" lossy trapdoor functions such that there is only a negligible probability that $f_{\mathbf{A}}(\cdot)$ is not injective or $F_{ltdf}^{-1}(\mathbf{T}, \cdot)$ incorrectly computes $f_{\mathbf{A}}^{-1}(\cdot)$ for some input.

If we extend this concept to a tree-like construction for more than two functions by denoting each leaf as a function, then we call it an *all-but-one (ABO) trapdoor function* if only one leaf expresses a lossy function.

Lossy trapdoor functions can be applied to many cryptographic primitives like pseudorandom generators, collision-resistant hash functions, and oblivious transfer protocols.

4.2 Application of Lossy Trapdoor Functions

After Peikert and Waters [18] had introduced the concept of lossy trapdoor functions, Bellare *et al.* [19] and Qin *et al.* [20] extended the notion as identity-based lossy trapdoor functions and leakage-resilient lossy trapdoor functions, respectively, without using lattices as a main tool.

Meanwhile, Xie *et al.* [21] introduced the notion of inner-product trapdoor and inner-product lossy trapdoor functions to construct an inner-product encryption scheme from lattices.

An inner-product trapdoor function consists of four poly-time algorithms (**IP.Param**, **IP.TrapGen**, **IP.Eval**, **IP.Inv**) with the following functionality:

IP.Param(n, l) :
Given a security parameter n and a predicate length parameter l, it computes a public parameter $params = (\mathbf{A}, \mathbf{B}, \{\mathbf{A}_{ij}\})$ and a master secret key $msk = \mathbf{T}_{\mathbf{A}}$ using **L.TrapGen**(n, m, q).

IP.TrapGen$(params, \mathbf{a}, msk)$:
Given a predicate $\mathbf{a} \in \mathbb{Z}_q$, it outputs an inversion key $sk_{\mathbf{a}} = \mathbf{S}_{\mathbf{a}}$ where $\mathbf{S}_{\mathbf{a}}$ is a basis for a matrix $\mathbf{U}_{\mathbf{a}} = [\mathbf{A} \mid \Sigma_i \Sigma_j \mathbf{a}_{ij} \mathbf{A}_{ij}]$ by **ExtBasis**$(\mathbf{T}_{\mathbf{A}}, \mathbf{U}_{\mathbf{a}})$.

IP.Eval$(params, \mathbf{b}, \mathbf{m})$:
Given an attribute $\mathbf{b} \in \Sigma$ and an input value $\mathbf{m}$, it outputs an output value $C_{\mathbf{b}}$ by some matrix multiplication.

IP.Inv$(params, sk_{\mathbf{a}}, \{C_{\mathbf{b}}, \mathbf{b}\})$:
Given an function value $\{C_{\mathbf{b}}, \mathbf{b}\}$, this algorithm outputs an inverse value $\mathbf{m}$.

Alwen *et al.* [22] introduced new lossy trapdoor functions and ABO trapdoor functions based on Learning With Rounding (LWR) problem, instead of standard LWE problem, as below:

Definition 3. (LWR problem) For a security parameter λ and corresponding integers $n = n(\lambda), m = m(\lambda), q = q(\lambda), p = p(\lambda)$, LWR problem states that for a given $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and $\mathbf{u} \leftarrow \mathbb{Z}_q^m$, $\left(\mathbf{A}, \lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p\right)$ and $\left(\mathbf{A}, \lfloor \mathbf{u} \rfloor_p\right)$ are computationally indistinguishable.

They define the $l(\lambda)$-*entropic lossy trapdoor functions* and $l(\lambda)$-*entropic ABO trapdoor functions* by modifying the definition of the *lossiness* as losing the entropy and give a construction based on LWR problem.

Injective functions :
For any (pk, sk) from $\mathbf{Gen}(1^\lambda, \text{injective})$ we get $F^{-1}(sk, F(pk, \mathbf{s})) = \mathbf{s}$ for all $\mathbf{s}$ and $f_{pk}(\cdot)$ is a injective function.

Lossy functions : For $pk \leftarrow \mathbf{Gen}(1^\lambda, \text{lossy})$, $f_{pk}(\cdot)$ loses the entropy up to $l(\lambda)$. This parameter $l(\lambda)$ is the residual leakage of this lossy trapdoor function.

Both $l(\lambda)$-entropic lossy trapdoor functions and $l(\lambda)$-entropic ABO trapdoor functions can be used to replace lossy trapdoor functions by Peikert and Waters [18] in various applications.

5 Homomorphic Trapdoor Function

In this section, we give a formal definition and security requirements of homomorphic trapdoor functions and the concrete construction based on lattice by Gorbunov *et al.* [9]. Then, we discuss several applications of lossy trapdoor functions from the literature.

5.1 Definition and Security Requirements

Gorbunov *et al.* [9] introduced the new concept of *homomorphic trapdoor functions* to construct the levelled fully homomorphic signature scheme.

Conceptually, a homomorphic trapdoor function is a trapdoor function that can take a tuple of values $\{u_i, x_i, v_i = f_{pk,x_i}(u_i)\}_{i \in [N]}$ and compute an input u^* and an output $v^* = f_{pk,g(x_1,x_2,\cdots,x_N)}(u^*)$. Formally, a homomorphic trapdoor function consists of five polytime algorithms ($\mathbf{H.Gen}$, f, f^{-1}, $\mathbf{H.Eval}^{in}$, $\mathbf{H.Eval}^{out}$) with the following functionality:

$\mathbf{H.Gen}(1^\lambda)$:
A security parameter λ defines the index space $\mathcal{X}$, input space $\mathcal{U}$, output space $\mathcal{V}$, and input distribution $\mathcal{D}_\mathcal{U}$ where one can efficiently sample uniformly at random from $\mathcal{V}$.

$f_{pk,x}$: An deterministic algorithm to get the function value.

$f_{sk,x}^{-1}$: A probabilistic algorithm to get the inverse

$\mathbf{H.Eval}^{in}$ **and** $\mathbf{H.Eval}^{out}$:
From a function g and a tuple of $\{u_i, x_i, v_i\}$, it outputs u^* and v^*.

The security requirements of homomorphic trapdoor functions (*HTDF security*) are the one-wayness of the function without trapdoors and somewhat claw-freeness which holds when it is negligible to find $u, u' \in \mathcal{U}$ and $x \neq x' \in \mathcal{X}$ such that $f_{pk,x}(u) = f_{pk,x'}(u')$. A collision resistance property is not necessary for the security of homomorphic trapdoor functions and the existential unforgeability of fully homomorphic signatures.

5.2 Application of Homomorphic Trapdoor Functions

As an extension of Gorbunov *et al.*'s trapdoor function [9], Alperin-Sheriff [23] define the notion of puncturable homomorphic trapdoor functions to get shorter signature scheme, by combining homomorphic trapdoor functions with lattice mixing and vanishing technique [14].

Wang *et al.* [24] suggested the identity-based homomorphic trapdoor functions with a better security notion by achieving collision resistance. Then, they constructed a strongly unforgeable identity-based fully homomorphic signature scheme.

Recently, Fiore *et al.* [25] showed a multi-key homomorphic signature scheme with the homomorphic trapdoor functions.

6 Conclusion and Future Work

We investigate previous lattice-based trapdoor functions from Gentry *et al.*'s trapdoor function [10] with an efficient sampling algorithm to Gorbunov *et al.*'s recent work [9] on homomorphic trapdoor functions to construct levelled fully homomorphic signature scheme.

We summarize the classification of lattice-based trapdoor functions as standard trapdoor, lossy trapdoor, and homomorphic trapdoor functions with their features and cryptographic applications in Appendix A.

As future work, we plan to define ring homomorphic trapdoor function and its security requirements to construct a ring homomorphic signature scheme.

Acknowledgement

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. NRF-2015R1A2A2A01006812).

References

[1] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pp. 99–108, ACM, 1996.

[2] Z. Brakerski and Y. T. Kalai, "A framework for efficient signatures, ring signatures and identity based encryption in the standard model," *IACR Cryptology ePrint Archive 2010/86*, 2010.

[3] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the Forty-First Annual ACM on Symposium on Theory of Computing*, pp. 169–178, ACM, 2009.

[4] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) lwe," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.

[5] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Advances in Cryptology–CRYPTO 2013*, pp. 75–92, Springer, 2013.

[6] S. Garg, C. Gentry, and S. Halevi, "Candidate multilinear maps from ideal lattices," in *Advances in Cryptology–EUROCRYPT 2013*, vol. 7881, pp. 1–17, Springer, 2013.

[7] D. Boneh and D. M. Freeman, "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures," in *Public Key Cryptography–PKC 2011*, pp. 1–16, Springer, 2011.

[8] D. Boneh and D. M. Freeman, "Homomorphic signatures for polynomial functions," in *Advances in Cryptology–EUROCRYPT 2011*, pp. 149–168, Springer, 2011.

[9] S. Gorbunov, V. Vaikuntanathan, and D. Wichs, "Leveled fully homomorphic signatures from standard lattices," in *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pp. 469–477, ACM, 2015.

[10] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proceedings of the Fortieth Annual ACM on Symposium on Theory of Computing*, pp. 197–206, ACM, 2008.

[11] J. Alwen and C. Peikert, "Generating shorter bases for hard random lattices," *Theory of Computing Systems*, vol. 48, no. 3, pp. 535–553, 2011.

[12] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pp. 333–342, ACM, 2009.

[13] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai trees, or how to delegate a lattice basis," *Journal of Cryptology*, vol. 25, no. 4, pp. 601–639, 2012.

[14] X. Boyen, "Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more," in *Public Key Cryptography–PKC 2010*, pp. 499–517, Springer, 2010.

[15] J. Wang and B. Sun, "Ring signature schemes from lattice basis delegation," in *International Conference on Information and Communications Security*, pp. 15–28, Springer, 2011.

[16] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 700–718, Springer, 2012.

[17] R. Bendlin, S. Krehbiel, and C. Peikert, "How to share a lattice trapdoor: Threshold protocols for signatures and (h) ibe," in *Applied Cryptography and Network Security*, pp. 218–236, Springer, 2013.

[18] C. Peikert and B. Waters, "Lossy trapdoor functions and their applications," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1803–1844, 2011.

[19] M. Bellare, E. Kiltz, C. Peikert, and B. Waters, "Identity-based (lossy) trapdoor functions and applications," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 228–245, Springer, 2012.

[20] B. Qin, S. Liu, K. Chen, and M. Charlemagne, "Leakage-resilient lossy trapdoor functions and public-key encryption," in *Proceedings of the First ACM workshop on Asia Public-Key Cryptography*, pp. 3–12, ACM, 2013.

[21] X. Xie, R. Xue, and R. Zhang, "Inner-product lossy trapdoor functions and applications," in *International Conference on Applied Cryptography and Network Security*, pp. 188–205, Springer, 2012.

[22] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs, "Learning with rounding, revisited," in *Advances in Cryptology–CRYPTO 2013*, pp. 57–74, Springer, 2013.

[23] J. Alperin-Sheriff, "Short signatures with short public keys from homomorphic trapdoor functions," in *Public Key Cryptography–PKC 2015*, pp. 236–255, Springer, 2015.

[24] F. Wang, K. Wang, B. Li, and Y. Gao, "Leveled strongly-unforgeable identity-based fully homomorphic signatures," in *International Information Security Conference*, pp. 42–60, Springer, 2015.

[25] D. Fiore, A. Mitrokotsa, L. Nizzardo, and E. Pagnin, "Multi-key homomorphic authenticators," in *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Part II*, pp. 499–530, Springer, 2016.

Appendix A. Classification of Lattice-based Trapdoor Functions

Scheme	Used Type	Features and Applications
GPV08 [10]	Normal	• Propose the definition of trapdoor function and collision-resistance property. • Lattice-based trapdoor functions based on SIS and LWE problem with the preimage sampling algorithm • Applicable to any lattice-based signature schemes
Pei09 [12]	Normal	• Introduce the notion of chosen-output security for trapdoor functions • Applicable to constructing a secure cryptosystem against chosen-ciphertext attack
Boy10 [14]	Normal	• Introduce a two-sided function which behaves differently for "firm" trapdoor and "fickle" trapdoor • Applicable to a more practical signature scheme by mixing the lattice trapdoors
AP11 [11]	Normal	• Propose the improved lattice-based trapdoor generation algorithm
CHKP12 [13]	Normal	• Explain how to delegate lattice trapdoors using the extending basis technique • Applicable to constructing a signature scheme without random oracles and hierarchical ID-based encryption
BK10 [2]	Normal	• Define ring trapdoor functions and give a concrete construction based on lattice • Applicable to constructing a ring signature
WS11 [15]	Normal	• Introduce a new sampling algorithm to construct ring trapdoor function • Applicable to constructing a ring signature scheme in the random oracle model or to enhancing the security of a ring signature scheme in the standard model
MP12 [16]	Normal	• Introduce a new notion of $\mathbf{G}$-trapdoor and the tag of the trapdoor to get the simpler, more efficient, and smaller trapdoor • Propose new algorithms for inverting LWE, sampling SIS preimages, and securely delegating basis, with $\mathbf{G}$-trapdoor • Applicable to getting a more efficient lattice-based cryptographic protocol
PW11 [18]	Lossy	• Propose general cryptographic primitives called lossy trapdoor functions and all-but-one (ABO) trapdoor functions • Introduce the concrete construction of both lossy trapdoor functions and ABO trapdoor functions based on LWE problem
XXZ12 [21]	Lossy	• Introduce the notion of inner-product trapdoor functions and inner-product lossy trapdoor functions to construct inner product encryption
AKPW13 [22]	Lossy	• Lossy/ABO trapdoor functions based on LWR problem •Both $l(\lambda)$-entropic lossy trapdoor functions and $l(\lambda)$-entropic ABO trapdoor functions can be used to replace lossy trapdoor functions in PW11 [18]
GVW15 [9]	Homomorphic	• Define the notion of homomorphic trapdoor function and its security requirements as HTDF security • Applicable to lattice-based levelled fully homomorphic signature scheme
Alp15 [23]	Homomorphic	• Introduce puncturable homomorphic trapdoor function to construct a short signature scheme by combining homomorphic trapdoor functions and lattice mixing and vanishing technique
WWLG15 [24]	Homomorphic	• Construct an identity-based homomorphic trapdoor function
FMLP16 [25]	Homomorphic	• Define the multi-key authenticator and construct multi-key homomorphic signature on lattices using homomorphic trapdoor functions

$\star$ Note that each scheme is identified with the authors' initials (*e.g.* GPV) and the publication year (*e.g.* 08).