Compact Adaptively Secure Identity-Based Encryption from Middle-Product Learning with Errors

Jingjing Fan[1][0000−0002−3203−311X], Xingye Lu*[2][0000−0002−3595−044X], Man Ho Au[2][0000−0003−2068−9530], and Siu Ming Yiu[1][0000−0002−3975−8500]

[1] University of Hong Kong {jjfan, smyiu}@cs.hku.hk
[2] The Hong Kong Polytechnic University {xing-ye.lu,mhaau}@polyu.edu.hk

Abstract. Identity-Based Encryption (IBE) is a cryptographic primitive where any string, such as an email address, can serve as a public key. With the advent of quantum computing, post-quantum secure IBE constructions have become critical for ensuring long-term data security. The state-of-the-art construction based on MPLWE introduced by Fan et al. significantly advanced the field by achieving adaptive security under standard assumptions, however the size of the master public key (MPK) grows linearly with the identity length, posing scalability challenges for real-world applications. In this work, we build on Fan et al.'s construction by employing a fully homomorphic trapdoor function to optimize the number of polynomials required for generating secret keys. This approach significantly reduces the MPK size from $O(\ell)$ polynomial vectors to $O(\ell^{1/d})$, where d is a constant. Despite this compactness, our scheme retains the same secret key and ciphertext sizes as Fan et al.'s construction and introduces no additional security assumptions.

Keywords: Lattice-Based Cryptography · Middle-Product LWE · Compact Identity-Based Encryption

1 Introduction

Identity-Based Encryption (IBE), introduced by Shamir in 1984 [17], is a public-key encryption paradigm where any arbitrary string, such as a mobile phone number or an email address, can function as a public key. This eliminates the need for a Public Key Infrastructure (PKI), which is a major bottleneck in traditional public-key cryptosystems. However, practical IBE schemes were not realized until 2001, when Boneh and Franklin [5], and independently Cocks [7], proposed the first IBE constructions.

The importance of IBE in its practical applications lies in simplifying key management. By leveraging identities directly as public keys, IBE reduces the complexity of certificate management in scenarios such as secure email, access control, and device authentication. As communication systems become increasingly

*Corresponding author

interconnected and complex, IBE provides a scalable and streamlined solution to security challenges.

With the advent of quantum computing, the security assumptions that underpin traditional cryptographic primitives are threatened. Quantum algorithms, such as Shors algorithm, can efficiently solve the integer factorization and discrete logarithm problems, undermining the foundations of widely used IBE schemes. This highlights the urgent need for post-quantum secure IBE constructions, particularly those based on quantum-resistant problems such as lattice-based cryptography.

Lattice-based IBE schemes began with Gentry et al.'s construction in 2008 [9], which was selectively secure in the random oracle model. This was followed by Agrawal et al. in 2010 [3], who achieved adaptive security in the standard model but at the cost of a large master public key (MPK). Subsequent research has focused on reducing the MPK size while maintaining security and efficiency. In standard lattice setting, Yamada's 2016 scheme [20], Zhang et al. [22], and Yamada [21] achieved logarithmic MPK size with respect to the identity length ℓ, and the MPK size further shortened by Abla [1] to $\omega(\frac{\log(\ell)}{\log\log\log(\ell)})$. For ring-based constructions, Katsumata and Yamada [10] first achieved a MPK size (number of ring vectors in MPK) sublinear to ℓ, and later Abla et al. [2] proposed a scheme with an MPK size that is independent of the identity length.

Despite these advancements, lattice-based IBE schemes still face challenges in balancing efficiency and security, particularly when adapting to real-world constraints. Recent constructions Middle-product LWE (MPLWE) have shown promise, offering robustness and efficiency. However, earlier MPLWE-based IBE schemes either relied on the random oracle model or only achieved selective security in the standard model. The first adaptively secure MPLWE-based IBE in the standard model [8] built on these efforts but inherited the inefficiency of a linearly growing MPK size.

In this paper, we address the challenge of constructing an adaptively secure IBE scheme in the standard model based on MPLWE with a shorter MPK size. By improving the abort-hash function from prior work [20], we encode identities more efficiently, achieving a sublinear growth of MPK size relative to identity length. Our proposed scheme not only reduces the MPK size but also maintains comparable secret key and ciphertext size to the state-of-the-art construction [8]. This result marks a significant step toward practical, quantum-secure IBE schemes, balancing compactness, efficiency, and adaptive security in the standard model.

1.1 Our Contribution

Our main construction This paper introduces a compact, adaptively secure Identity-Based Encryption (IBE) scheme based on the Middle-Product Learning with Errors (MPLWE) problem. Our scheme significantly reduces the master public key size to $O(n\ell^{1/c}\log n)$, while maintaining $O(n\log n)$ size for both ciphertexts and private keys. As shown in Table 1, this is a marked improvement

Table 1. Comparison of Lattice-Based Adaptively Secure IBEs in the Standard Model.

LWE type	Scheme	$\lvert\mathsf{mpk}\rvert$	$\lvert C\rvert, \lvert\mathsf{sk}_{\mathsf{id}}\rvert$
Standard LWE	CHKP10 [6]	$\tilde{O}(n^2\ell)$	$\tilde{O}(n\ell)$
	ABB10 [3]	$\tilde{O}(n^2\ell)$	$\tilde{O}(n)$
	Yam16: Type 1 [20]	$\tilde{O}(n^2\ell^{\frac{1}{c}})$	$\tilde{O}(n)$
	Yam16: Type 2 [20]	$\tilde{O}(n^2\ell^{\frac{1}{c}})$	$\tilde{O}(n)$
	ZCZ16 [22]	$\tilde{O}(n^2\log\ell)$	$\tilde{O}(n)$
	Yam17: Type 1 [21]	$\tilde{O}(n^2\log^3\ell)$	$\tilde{O}(n)$
	Yam17: Type 2 [21]	$\tilde{O}(n^2\log^2\ell)$	$\tilde{O}(n)$
	Abla24 [1]	$\tilde{O}(n^2\frac{\log\ell}{\log\log\ell})$	$\tilde{O}(n)$
RLWE	KY16: Type 1 [10]	$\tilde{O}(n\ell^{\frac{1}{c}})$	$\tilde{O}(n)$
	KY16: Type 2 [10]	$\tilde{O}(n\ell^{\frac{1}{c}})$	$\tilde{O}(n)$
	ALWW21: Type 1 [2]	$\tilde{O}(n\log\ell)$	$\tilde{O}(n)$
	ALWW21: Type 2 [2]	$\tilde{O}(n)$	$\tilde{O}(n)$
MPLWE	FLA23 [8]	$\tilde{O}(n\ell)$	$\tilde{O}(n)$
	Our IBE	$\tilde{O}(n\ell^{\frac{1}{c}})$	$\tilde{O}(n)$

c is a flexible constant (recommended to be 2 or 3 in [10]). $\tilde{O}(f(n)) = O(f(n)\log^k n)$ for some constant k.

over existing MPLWE-based IBE schemes. This reduction is achieved through a tuple representation of identity, leveraging the specific structure of isomorphic sets to enable a more compact encoding. Furthermore, taking a broader view, our scheme, compared to standard LWE-based constructions, offers reduced public key bandwidth, while avoiding the dependency on the specific algebraic structure of polynomial rings present in RLWE assumptions, thus striking a balance between the two.

Comparison with Existing IBEs Table 1 demonstrates a comparison between our IBE scheme and existing lattice-based adaptively secure IBEs. For our analysis, we set the identity length as $\ell = O(n)$. Specifically, compare with [8], our IBE scheme is the most compact adaptively secure IBE scheme from the MPLWE. The size of the public parameters, ciphertexts, and secret keys are $\tilde{O}(n\ell^{\frac{1}{c}}), \tilde{O}(n)$, and $\tilde{O}(n)$, respectively. Among the RLWE-based and standard LWE-based IBE schemes, [2] has the smallest public key size. Despite the efficiency, the security of the schemes based on RLWE crucially depend on the choice of the underlying ring. Based on MPLWE, our scheme is at least as secure as the schemes in [10], while enjoying a broader selection of number fields. Compared with the schemes from standard lattice, i.e. schemes in [6, 3, 20, 22, 21] our scheme has shorter master public key size when assuming the length of the identity ℓ to be $O(n)$.

1.2 Overview of Our Construction and Security Proof

Our work builds on the KY16 framework for constructing compact IBE schemes from lattice problems [10]. While KY16 uses RLWE, we instead employ

MPLWE - a variant offering stronger security guarantees for broader number fields. Our scheme preserves KY16's identity compression methodology but introduces key adaptations for MPLWE compatibility. Notably, KY16's security reduction requires invertible ring elements Fy(id), which becomes problematic in MPLWE settings as inverses rarely exist. Our critical modification replaces ring elements with scalars for Fy(id), ensuring invertibility while maintaining functionality. This migration to MPLWE constitutes our main contribution, overcoming fundamental algebraic constraints through novel parameterization and structural adjustments. The resulting scheme retains KY16's compactness while enhancing security foundations and efficiency through optimized parameter selection.

In detail, the security proof prepares the master public keys as follows: $\overrightarrow{b_0} = \overrightarrow{S_0}\overrightarrow{a} + y_0\overrightarrow{g}, \overrightarrow{b_{i,j}} = \overrightarrow{S_{i,j}}\overrightarrow{a} + y_{i,j}\overrightarrow{g}$ for $(i,j) \in [2] \times [\kappa]$. We use $R^{<d}[x]$ to denote polynomials in the ring R with a degree less than d. Let $\mathbf{d} = (d_1, \cdots, d_n)^T \in \mathbb{Z}^n$ be an integer vector with positive coefficients. We use $R^{<\mathbf{d}}[x]$ to denote the polynomial vector, whose i-th entry is in $R^{<d_i}[x]$ for $i \in [n]$. In the above equation, $\overrightarrow{a} \in (\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$, short $\overrightarrow{S_0}, \overrightarrow{S_{i,j}} = [\overrightarrow{s_1}, \overrightarrow{s_2}, \cdots, \overrightarrow{s_{t'}}]^T$, where $\overrightarrow{s_j} \in (\mathbb{Z}_q^d[x])^t \times \mathbb{Z}_q^{\gamma\tau}$ for $j \in [t']$, and $\overrightarrow{b_0}, \overrightarrow{b_{i,j}} \in (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ are uniformly random polynomial vectors, $y_0, y_{i,j} \in \mathbb{Z}_q$ are uniformly random scalars and $\overrightarrow{g} \in (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ is an analogue of the matrix $\mathbf{G}$ in [13]. We can write $h(\mathsf{id}) = \overrightarrow{S_{\mathsf{id}}}\overrightarrow{a} + F_{\mathbf{y}}(\mathsf{id})\overrightarrow{g}$, where $F_{\mathbf{y}}(\mathsf{id}) = y_0 + \sum_{(i,j)\in\mathfrak{F}(\mathsf{id})} y_{1,i}y_{2,j}$.

We partition identities by following method: non-zero $F_{\mathbf{y}}(\mathsf{id})$ enable trapdoor-based key extraction via $\overrightarrow{g}$, while $F_{\mathbf{y}}(\mathsf{id}^*) = 0$ for the challenge identity id^* allows for MPLWE challenge embedding. The security proof requires non-negligible probability that all queried identities satisfy $F_{\mathbf{y}}(\mathsf{id}^*) = 0$ for the challenge identity id^*. This introduces a parameterization constraint: Since $y_{i,j}$ scales with query count, and $\mathbf{S}_{i,j}$ scales accordingly, both require super-polynomial modulus q. To maintain polynomial q , we must bound Q - a restriction absent in KY16's RLWE-based scheme where $R_q = \mathbb{Z}_q^n$ naturally supports exponential identity space. To deal with this subtlety, we have to restrict the number of queries Q to n^c for some constant c.

2 Preliminaries

2.1 Notations

In this paper, the security parameter is denoted by n, and the growth of functions will be classified using the big-O notation. Also, we write $f(n) = \tilde{O}(g(n))$ if $f(n) = O(g(n) \cdot \log^c n)$ for some constant c. $\mathsf{poly}(n)$ is used to denote a random function $f(n) = O(n^c)$ for some constant c. A function $f(n)$ is negligible, denoted by $\mathsf{negl}(n)$, if for every polynomial g, $\exists N \in \mathbb{N}$ s.t. $f(n) < \frac{1}{g(n)}$, $\forall n > N$. We say that a probability is overwhelming if it is $1 - \mathsf{negl}(n)$. In this paper, we will work on the polynomials in the ring $R = \mathbb{Z}[x]/(x^N+1)$ or $R_q = R/qR = \mathbb{Z}_q[x]/(x^N+1)$ for some positive integer N.

2.2 Identity-Based Encryption

Syntax An identity-based encryption(IBE) scheme consists of four algorithms, $\mathsf{KeyGen}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec}$, which runs as follows: $\mathsf{KeyGen}(1^n) \to (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter 1^n output the master public key mpk and the master secret key msk. $\mathsf{Extract}(\mathsf{id}, \mathsf{mpk}, \mathsf{msk}) \to \mathsf{sk}_{\mathsf{id}}$: On input the identity id, the master public key mpk and the master secret key msk, output the secret key for the identity $\mathsf{sk}_{\mathsf{id}}$. $\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, m) \to c$: On input the master public key mpk, the identity id and the message m, output the ciphertext c. $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, c) \to m$: On input the ciphertext c and the secret key for the identity $\mathsf{sk}_{\mathsf{id}}$, output the message m.
An IBE scheme $\mathsf{IBE} = (\mathsf{KeyGen}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ possesses correctness if for all the identities id and message m, $\Pr[\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, \mathsf{Enc}\ (\mathsf{mpk}, \mathsf{id}, m)) = m] = 1 - \mathsf{negl}(n)$, where the probability is taken over the randomness of KeyGen, $\mathsf{Extract}$, Enc and Dec.
IND-CPA Security The IND-CPA security for an IBE is defined as follows: **Setup**: The challenger $\mathcal{C}$ runs $\mathsf{KeyGen}(1^n) \to (\mathsf{mpk}, \mathsf{msk})$ and sends mpk to the adversary $\mathcal{A}$. **Key Extraction Query**: The adversary $\mathcal{A}$ sends an identity id to challenger $\mathcal{C}$. $\mathcal{C}$ runs the extraction algorithm $\mathsf{Extract}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id}) \to \mathsf{sk}_{\mathsf{id}}$ and sends $\mathsf{sk}_{\mathsf{id}}$ back to $\mathcal{A}$. **Challenge**: $\mathcal{A}$ chooses two messages m_0, m_1 and an identity id^* to be challenged. $\mathcal{A}$ sends $(\mathsf{id}^*, m_0, m_1)$ to $\mathcal{C}$. If id^* has been queried, $\mathcal{C}$ aborts and sends $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ picks $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and sends $c \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}^*, m_{\mathsf{coin}})$ to $\mathcal{A}$. **Key Extraction Query**: $\mathcal{A}$ can make further key extraction queries on identity $\mathsf{id} \neq \mathsf{id}^*$. **Output**: $\mathcal{A}$ outputs $\mathsf{coin}^* \in \{0,1\}$.
$\mathcal{A}$ wins the game if $\mathsf{coin}^* = \mathsf{coin}$. The advantage of $\mathcal{A}$ winning the IND-CPA game is defined to be $Adv_{\mathcal{A}} = \Pr[\mathcal{A} \text{ wins IND-CPA game}] - \frac{1}{2}$.

Definition 1 (IND-CPA). *An IBE scheme is IND-CPA secure if, for any polynomial-time adversary $\mathcal{A}$, it has $Adv_{\mathcal{A}}^{IBE} = \mathsf{negl}(n)$.*

2.3 Lattices and Gaussian Distributions

An n-dimensional lattice in m-dimensional Euclidean space is an additive discrete set defined as follows: $\Lambda(\mathbf{b_1}, \ldots, \mathbf{b_n}) = \{\sum_{i=1}^{n} x_i \mathbf{b_i} | x_i \in \mathbb{Z}\}$, where $\mathbf{b_1}, \ldots, \mathbf{b_n} \in \mathbb{Z}^m$ are linearly independent column vectors. Λ is full-rank if and only if $m = n$. $\Lambda^* := \{\mathbf{y} \in \mathrm{span}_{\mathbb{R}}(\Lambda) | \langle \mathbf{y}, \mathbf{x} \rangle \in \mathbb{Z}, \forall \mathbf{x} \in \Lambda\}$ is defined to be the dual lattice of Λ, denoted as $\Lambda^{\perp}$. For positive integers n, q and any matrix $\mathbf{A} \in \mathbb{Z}^{n \times m}$, let $\Lambda^{\perp} := \{\mathbf{z} \in \mathbb{Z}^m | \mathbf{Az} = \mathbf{0} \mod q\}$. For $\mathbf{u} \in \mathbb{Z}_q^n$ s.t. $\exists \mathbf{t} \in \mathbb{Z}_q^m$ satisfying $\mathbf{At} = \mathbf{u}$, let $\Lambda_{\mathbf{u}}^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m | \mathbf{Az} = \mathbf{u} \mod q\}$, $\Lambda_{\mathbf{u}}^{\perp}(\mathbf{A}) = \Lambda^{\perp}(\mathbf{A}) + \mathbf{t}$. Next, we will recap some well-known definitions concerning Gaussian distribution.
Continuous Gaussian Distribution: For a positive semi-definite matrix $\Sigma \in \mathbb{R}^{n \times n}$, the continuous Gaussian distribution D_{Σ} is the probability distribution over $\mathbb{R}^n$ whose density is proportional to $\rho_{\Sigma}(\mathbf{x}) = \exp(-\pi \mathbf{x}^T \Sigma^{-1} \mathbf{x})$.
Discrete Gaussian Distribution: Given a countable set $S \subset \mathbb{R}^n$ and $\sigma > 0$, the discrete Gaussian distribution $D_{S,\sigma,\mathbf{c}}$ is the probability distribution over S whose density is proportional to $\rho_{\sigma,\mathbf{c}(x)} := \exp(\frac{-\pi \cdot \|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2})$. That is, for $x \in S$,

$D_{S,\sigma,\mathbf{c}} := \frac{\rho_{\sigma,\mathbf{c}}(x)}{\rho_{\sigma,\mathbf{c}}(S)}$. If $\mathbf{c} = 0$, we can omit $\mathbf{c}$ and write $D_{S,\sigma}$ instead.
Next, we will illustrate some inequalities that will be used in our proof.
For any n-dimensional lattice Λ and real $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\lambda)$ is defined to be the smallest real $s > 0$ such that $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \epsilon$ [14].
Gaussian Tail Inequality: For any $\epsilon > 0$, any $\sigma \geq \eta_\epsilon(\mathbb{Z})$, and any $t > 0$, we have $\Pr_{x \leftarrow D_{\mathbb{Z},\sigma,c}}[|x - c| \geq t \cdot \sigma] \leq 2e^{-\pi t^2} \cdot \frac{1+\epsilon}{1-\epsilon}$. In particular, for $\epsilon \in (0, 1/2)$ and $t \geq \omega(\sqrt{\log n})$, the probability that $|x - c| \geq t \cdot \sigma$ is negligible in n [9].

2.4 Polynomials, Matrices and Middle Product of Polynomials

First, we will recap some notations and operations for common vectors and matrices. In this paper, a vector refers to a column vector. For a matrix $\mathbf{A}$, $A_{i,j}$ denotes its (i,j)-th entry. For a vector $\mathbf{v} \in \mathbb{R}^n$, its Euclidean and sup norm is denoted by $\|\mathbf{v}\|$ and $\|\mathbf{v}\|_\infty$, respectively. We define the largest singular value of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ as $\sigma_1(\mathbf{A}) := max_{\|\mathbf{u}\|=1}\|\mathbf{A}\mathbf{u}\|$. Also, we may define $\sigma_1(\mathbf{A})$ and the tensor product and introduce the noise rerandomization algorithm.

Lemma 1. *For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, we have $\sigma_1(\mathbf{A}) \leq \sqrt{mn}\max_{i,j}|A_{i,j}|$*

Definition 2 (Tensor Product). *For $m_1, m_2, n_1, n_2 > 0$, the tensor product $\otimes : \mathbb{R}^{m_1 \times n_1} \times \mathbb{R}^{m_2 \times n_2} \mapsto \mathbb{R}^{m_1 m_2 \times n_1 n_2}$ is defined to be the map: $\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} A_{1,1}\mathbf{B} & \dots & A_{1,n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ A_{m,1}\mathbf{B} & \dots & A_{m,n}\mathbf{B} \end{bmatrix}$, for matrix $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$ and matrix $\mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$.*

Next, we will define the polynomials to be used in this paper. For any $d > 0$ and any set $S \subset R$, we will use $S^{<d}[x]$ to denote the set of polynomials in $R[x]$ of degree $< d$ whose coefficients are in S. For any distribution χ defined over R, we set $\chi^{<d}[x]$ to be the set of polynomials in $R^{<d}[x]$, with coefficients sampled independently according to χ.
Let $a = \sum_{i=0}^{d-1} \mathtt{a}_i x^i$ be a polynomial in $R^{<d}[x]$. We define the coefficient vector of a to be $\mathbf{a} = (\mathtt{a}_0, \mathtt{a}_1, \dots, \mathtt{a}_{d-1})$ and for $0 \leq i \leq d-1$, $\mathtt{a}_i$ denotes the coefficient of x^i in a. Also, we set $\overline{\mathbf{a}} = (\mathtt{a}_{d-1}, \dots, \mathtt{a}_1, \mathtt{a}_0)$. Naturally, we have $\overline{\overline{\mathbf{a}}} = \mathbf{a}$. Let $\mathbf{d} \in \mathbb{Z}^n = (d_1, \cdots, d_n)^T$ be an integer vector with positive coefficients. The polynomial vector, which is a column vector, is defined to be $\overrightarrow{a} := (a_1, \cdots, a_n)^T \in R^{<\mathbf{d}}[x]$, where a_i is a polynomial in $R[x]^{<d_i}$ for $i = 1, \cdots, n$.
Let $\mathbf{d} = (d_1, \cdots, d_n)^T \in \mathbb{Z}^n$ be an integer vector with positive coefficients and $t > 0 \in \mathbb{N}$, we define the polynomial matrix $\overrightarrow{A} := [\overrightarrow{a_1}, \cdots, \overrightarrow{a_t}]^T \in R^{t \times <\mathbf{d}}[x]$, where $\overrightarrow{a_i}$ is a polynomial vector in $R[x]^{<\mathbf{d}}[x]$ for $i = 1, \cdots, t$. We recap the inner product between polynomial vectors, multiplication between polynomial matrix and polynomial vector as in [8].

Definition 3 ([8], Definition 2). *Let $\mathbf{d}, \mathbf{k} \in \mathbb{Z}^n$ be integer vectors with positive coefficients. Let $\overrightarrow{a} = (a_1, \dots, a_n) \in R^{\mathbf{d}}[x]$, $\overrightarrow{b} = (b_1, \dots, b_n) \in R^{\mathbf{k}}[x]$. The inner product of $\overrightarrow{a}$ and $\overrightarrow{b}$ is defined to be $\overrightarrow{a} \cdot \overrightarrow{b} := \sum_{i=1}^n a_i b_i$.*

Definition 4 ([8],Definition 2). *Let $\mathbf{d}, \mathbf{k} \in \mathbb{Z}^n$ be integer vectors with positive coefficients. Let $\overrightarrow{A} = [\overrightarrow{a_1}, \ldots, \overrightarrow{a_t}]^T \in R^{t \times <\mathbf{d}}[x]$ where $\overrightarrow{a_1}, \ldots, \overrightarrow{a_t} \in R^{\mathbf{d}}[x]$. Let $\overrightarrow{b} \in R^{\mathbf{k}}[x]$. The multiplication between polynomial matrix and polynomial vector is defined to be $\overrightarrow{A} \cdot \overrightarrow{b} := [\overrightarrow{a_1} \cdot \overrightarrow{b}, \ldots, \overrightarrow{a_t} \cdot \overrightarrow{b}]^T$.*

For simplicity, in calculation and analysis, we tend to transform polynomial vectors and matrices into scalar vectors and matrices. Next, we will introduce the way we transform vectors into Toepliz matrices. Specifically, we make remarks on the special polynomial vectors and matrices that will be used in this paper.

Remark 1. In this paper, without further notation, we set gadget polynomial vector to be $\overrightarrow{g} = (g_1, g_2, \ldots, g_{\gamma\tau}) \in (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ s.t. $g_j = 2^u x^{dv}$, for $j = v\tau + u + 1$, where $u \in \{0, \cdots, \tau - 1\}, v \in \{0, \cdots, \gamma - 1\}$ and its corresponding Topeliz matrix as $\mathbf{G} = \mathbf{I}_{\gamma d} \otimes [1 \cdots 2^{\tau-1}] \in \mathbb{Z}_q^{\gamma d \times \gamma d\tau}$. Here $\mathbf{I}_{\gamma d}$ is the identity matrix with size $\gamma d \times \gamma d$.

Definition 5 ([12], Definition 6). *Let R be a ring and $d, k > 0$ be positive integers. For any polynomial $a \in R^{<k}[x]$ of degree less than k, let $\mathsf{T}^{k,d}(a)$ denote the matrix in $R^{(k+d-1)\times d}$ whose ith column, for $i = 1, \cdots, d$ is given by the coefficients of $x^{i-1} \cdot a$, listed from the lowest to the highest degree.*

Lemma 2 ([13]). *For a gadget polynomial vector $\overrightarrow{g} = (g_1, g_2, \ldots, g_{\gamma\tau}) \in (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ s.t. $g_j = 2^{\mathbf{t}} x^{dv}$, for $j = v\tau + \mathbf{t} + 1$, where $\mathbf{t} \in \{0, \cdots, \tau - 1\}, v \in \{0, \cdots, \gamma - 1\}$. There exists a deterministic efficiently computable function $\overrightarrow{g_b}^{-1}$ that takes as input a polynomial vector $\overrightarrow{u} \in (\mathbb{Z}_q^{n+2d-2}[x])^{\gamma\tau}$ outputs a polynomial matrix $\overrightarrow{R}$ s.t. $\overrightarrow{g_b}^T \overrightarrow{R} = \overrightarrow{u}^T$ and the coefficients of the polynomials in $\overrightarrow{R}$ are in the interval $[-b, b]$.*

Proof. Set $\overrightarrow{u} = (u_1, u_2, \ldots, u_{\gamma\tau})$. The problem can be transformed to find polynomial vectors $\overrightarrow{r_1}, \ldots, \overrightarrow{r_{\gamma\tau}}$ s.t. $\overrightarrow{g} \cdot \overrightarrow{r_i} = u_i$. And the Topeliz matrix corresponding to $\overrightarrow{g}$ is $\mathbf{G} = \mathbf{I}_{\gamma d} \otimes [1 \cdots 2^{\tau-1}] \in \mathbb{Z}_q^{\gamma d \times \gamma d\tau}$ by Remark 1. Without loss of generality, we will take $\overrightarrow{r_i}$ and u_i as an example. Set $\mathbf{u_i} = \mathsf{T}^{n+d-1,d}(u_i)$. By [13] and [11], we are able to find $\mathbf{e}$ s.t. $\mathbf{Ge} = \mathbf{u_i}$. Then we write $\mathbf{e}$ as $(\mathsf{T}^{d,1}(r_{i,1})^T | \mathsf{T}^{d,1}(r_{i,2})^T | \ldots | \mathsf{T}^{d,1}(r_{i,\gamma\tau})^T)^T$ where $\deg r_{i,j} < d$ for $1 \le j \le \gamma\tau$. Set $\sigma = \omega(d\tau\gamma\sqrt{\gamma})$, the distribution of $(r_{i,j})_{j=1}^{\gamma\tau}$ is exactly $(D_{\mathbb{Z}^d,\sigma}[x])^{\gamma\tau} | \sum_{j=1}^{\gamma\tau} g_j \cdot r_{i,j} = u_i$. So we can find a deterministic computable function $\overrightarrow{g_b}^{-1}$ that on input a polynomial vector $u \in (\mathbb{Z}_q^{n+2d-2})^{\gamma\tau}$ outputs a polynomial matrix $\overrightarrow{R}$ s.t. $\overrightarrow{g}^T \overrightarrow{R} = \overrightarrow{u}^T$. Moreover, we require the coefficients of each polynomial in $\overrightarrow{R}$ in the interval $[-b, b]$.

After introducing the definitions of polynomial vectors and matrices and operations on them, we will introduce the middle-product operation. For our purpose, we focus on the polynomials with $\mathsf{poly}(n)$-bounded expansion factors. One such class [12] is the family of all $f = x^m + h$ where $\deg(h) \le m/2$ and $\|h\|_\infty \le \mathsf{poly}(n)$.

Definition 6 (Middle-Product [16], Definition 3.1). *Let d_a, d_b, d, k be integers such that $d_a + d_b - 1 = d + 2k$. The middle product $\odot_d : R^{<d_a}[x] \times R^{<d_b}[x] \to R^{<d}[x]$ is defined to be the map: $(a, b) \mapsto a \odot b = \lfloor \frac{(a \cdot b) \mod x^{k+d}}{x^k} \rfloor = \sum_{k \le i+j \le k+d-1} (\mathsf{a}_i \mathsf{b}_j) x^{i+j-k}$, where $\mathbf{a} = (\mathsf{a}_0, \cdots \mathsf{a}_{d_a-1})$ and $\mathbf{b} = (\mathsf{b}_0, \cdots \mathsf{b}_{d_b-1})$ are the coefficient vectors of a and b, respectively.*

From Definition 6, we can get that the middle product is commutative trivially, i.e., $a \odot_d b = b \odot_d a$ for all polynomials a, b. Besides, the middle product also satisfies a "quasi-associative" property defined below in Lemma 3.

Lemma 3 ([16]). *Let $d, k, n > 0$. For all $r \in R^{<k+1}[x], a \in R^{<n}[x], s \in R^{<n+d+k-1}[x]$, we have $r \odot_d (a \odot_{d+k} s) = (r \cdot a) \odot_d s$.*

Next, we exhibit the transformation of the middle-product between polynomials into the product of scalar matrices.

Lemma 4 ([15], Lemma 3.2). *Let $d, k > 0$. Let $r \in R^{<k+1}[x]$ and $a \in R^{<k+d}[x]$ and $b = r \odot_d a$. Then $\overline{\mathbf{b}} = (\mathsf{T}^{k+1,d}(r))^T \cdot \bar{\mathbf{a}}$. In other words, we have $\mathbf{b} = \overline{(\mathsf{T}^{k+1,d}(r))^T \cdot \bar{\mathbf{a}}}$.*

Finally, we come to the definition of MPLWE.

Definition 7 (MP-LWE [11], Definition 9). *Let $n > 0$, $q > 2$, $m > 0$, $\mathbf{d} \in [\frac{n}{2}]^t$, $\alpha \in (0, 1)$ and let $\chi = D_{\alpha \cdot q}$ be a distribution over $\mathbb{R}_q$. For $s \in \mathbb{Z}_q^{<n-1}[x]$, we define the distribution $\mathrm{MP}_{q,n,\mathbf{d},\chi}(s)$ over $\Pi_{i=1}^t U(\mathbb{Z}_q^{n-d_i} \times \mathbb{R}_q^{<d_i}[x])$ as follows. 1. For each $i \in [t]$, sample $a_i \xleftarrow{\$} \mathbb{Z}_q^{n-d_i}[x]$ and sample $e_i \leftarrow \chi^{<d_i}[x]$ 2. Output $(a_i, b_i := a_i \odot_{d_i} s + e_i)$ The MP-LWE problem consists of distinguishing between arbitrarily many samples from $\mathrm{MP}_{q,n,\mathbf{d},\chi}(s)$ (denoted as $\mathrm{MPLWE}_{q,n,\mathbf{d},\chi}(s)$) and the same number of samples from $\Pi_{i=1}^t U(\mathbb{Z}_q^{n-d_i} \times \mathbb{R}_q^{<d_i}[x])$ with non-negligible probability over the choice of $s \xleftarrow{\$} \mathbb{Z}_q^{<n-1}[x]$.*

Theorem 1 (Hardness of MPLWE [11], Theorem 2). *Let $n > 0$, $q \ge 2$, $t > 0$, $\mathbf{d} \in [\frac{n}{2}]^t$, and $\alpha \in (0, 1)$. For $S > 0$, let $\mathcal{F}(S, \mathbf{d}, n)$ be the set of polynomials in $\mathbb{Z}[x]$ that are monic, have constant coefficient coprime with q, have degree m in $\cap_{i=1}^t [d_i, n - d_i]$ and satisfy $\mathrm{EF}(f) < S$. Then there exists a ppt reduction from $\mathrm{PLWE}_{D_{\alpha \cdot q}}^{(f)}$ for any $f \in \mathcal{F}(S, \mathbf{d}, n)$ to $\mathrm{MPLWE}_{q,n,\mathbf{d},D_{\alpha' \cdot q}}$ with $\alpha' = \alpha \cdot \sqrt{\frac{n}{2}} \cdot S$. Here $\mathrm{EF}(f) := \max_{g \in \mathbb{Z}^{<2m-1}[x]} \frac{\|g \mod f\|_\infty}{\|g\|_\infty}$.*

It has been proven that solving $\mathrm{PLWE}_{q,\chi}^{(f)}(s)$ is as hard as solving the Shortest Vector Problem (SVP) over the ideal lattice in $\mathbb{Z}_q[x]/f$ [18].

3 Trapdoor Generators and Related Operations

In this section, we will introduce the trapdoor generator and the sampling algorithms used in this paper. We will use the trapdoor generation and preimage sampling algorithm proposed in [8].

Theorem 2 ([13], Definition 5.2, Theorem 4). *Let* $\mathbf{G} := \mathbf{I}_k \otimes \left[1\ 2 \dots 2^{\tau-1}\right] \in \mathbb{Z}_q^{k\times k\tau}$. *Then given a matrix* $\mathbf{A} \in \mathbb{Z}^{k\times(m+k\tau)}$, *a matrix* $\mathbf{R} \in \mathbb{Z}^{m\times k\tau}$ *is a* ***G-trapdoor*** *for* $\mathbf{A}$ *if* $\mathbf{A}\left[\mathbf{R}^T|\mathbf{I}_{k\tau}\right]^T = \mathbf{G}$. *And there exists an efficient algorithm* $\mathcal{C}(\mathbf{A}, \mathbf{R}, \mathbf{u})$ *that operates as follows: On input matrices* $\mathbf{A}, \mathbf{R}$ *and a vector* $\mathbf{u}$, *sample from* $D_{\Lambda_u^\perp(\mathbf{A}),\sigma}$, *as long as* $\sigma \geq \omega(\sqrt{\log k})\sqrt{7(\sigma_1(\mathbf{R})^2+1)}$. *Moreover, the running time of* $\mathcal{C}$ *is the time to compute* $\mathbf{Rx}$ *for* $\mathbf{x} \in \mathbb{Z}^{k\tau}$ *plus* $\tilde{O}(m+k\tau)$.

Theorem 3 (TrapGen [11], Theorem 5). *Let* $q = \mathsf{poly}(n)$ *be a prime,* $n \in \mathbb{Z}$, $d \leq n$, $dt/n = \Omega(\log n)$, $\sigma = c\ell \cdot \omega(n\log^{3/2} n)$ *for some constant* c, $\gamma = \lceil \log_2 q \rceil$ *and* $\gamma = \frac{n+2d-2}{d}$ *be an integer. Then there exists a PPT algorithm* TrapGen *that generates a polynomial vector* $\overrightarrow{a} = (a_1, a_2, \cdots, a_t, a_{t+1}, \cdots, a_{t+\gamma\tau}) \in (\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ *together with a trapdoor* td *able to be stored in* $O(n\tau t)$ *space, where* $\overrightarrow{a} \approx_s U((\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau})$

Theorem 4 (SamplePre$(\overrightarrow{a}, \mathsf{td}, u, \sigma)$ [11], Theorem 5). *Let* $q = \mathsf{poly}(n)$ *be a prime,* $n \in \mathbb{Z}$, $d \leq n$, $dt/n = \Omega(\log n)$, $\sigma = \mathfrak{d} n^{c(\mathfrak{d}-1)} \cdot \omega(n^2 \log^2 n)$ *for some constant* c, $\gamma = \frac{n+2d-2}{d}$ *be an integer,* $\tau = \lceil \log_2 q \rceil$ *and* $t' = t + \gamma\tau$. *Then there exists a ppt algorithm* SamplePre *that on input polynomial vector* $\overrightarrow{a} = (a_1, a_2, \cdots, a_t, a_{t+1}, \cdots, a_{t+\gamma\tau})$ *with a trapdoor* td *and a syndrome* $u \in \mathbb{Z}_q^{n+2d-2}[x]$, *outputs* $\overrightarrow{r} = (r_i)_{i=1}^{t+\gamma\tau}$ *satisfying* $\sum_{i=1}^{t+\gamma\tau} a_i \cdot r_i = u$ *in* $\tilde{O}(nt)$ *time. And the output distribution of* $(r_i)_{i=1}^{t'}$ *is exactly the conditional distribution* $(D_{\mathbb{Z}^{2d-1},\sigma}[x])^t \times (D_{\mathbb{Z}^d,\sigma}[x])^{\gamma\tau} | \overrightarrow{a} \cdot \overrightarrow{r} = u$.

After illustrating the algorithms, we would like to adjust the SampleLeft algorithm and the SampleRight algorithm in [8] to fit in our new setting.
Algorithm SampleLeft$(\overrightarrow{a}, \mathsf{td}, u, \sigma)$**.** Inputs: a polynomial vector $\overrightarrow{a}$ in $(\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$, a trapdoor td, a polynomial u, a parameter $\sigma = \ell n^{c(\mathfrak{d}-1)} \cdot \omega(n^2\log^2 n)$. Outputs: $(r_i)_{i=1}^{t+2\gamma\tau} \approx_s (D_{\mathbb{Z}^{2d-1},\sigma}[x])^t \times (D_{\mathbb{Z}^d,\sigma}[x])^{2\gamma\tau} | \sum_{i=1}^{t+\gamma\tau} a_i \cdot r_i + \sum_{i=1}^{\gamma\tau} h_i \cdot r_{i+t'} = u$. The correctness of this algorithm is shown in [8] Lemma 6.
Algorithm SampleRight $(\overrightarrow{g}, y, \overrightarrow{a}, \overrightarrow{S}, u, \sigma)$**.** Inputs: a polynomial vector $\overrightarrow{a}$ in $(\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$, a polynomial vector $\overrightarrow{g}$ acting as a trapdoor together with a non-zero scalar y, a polynomial u, a polynomial matrix $\overrightarrow{S}$ a parameter $\sigma = \ell n^{c(\mathfrak{d}-1)} \cdot \omega(n^2\log^2 n)$. Outputs: $(r_i)_{i=1}^{t+2\gamma\tau} \approx_s (D_{\mathbb{Z}^{2d-1},\sigma}[x])^t \times (D_{\mathbb{Z}^d,\sigma}[x])^{2\gamma\tau} | \sum_{i=1}^{t+\gamma\tau} a_i \cdot r_i + \sum_{i=1}^{\gamma\tau} h_i \cdot r_{i+t'} = u$. The correctness of this algorithm will be shown in Lemma 5

Lemma 5 (SampleRight$(\overrightarrow{g}, y, \overrightarrow{a}, \overrightarrow{S}, u, \sigma)$ [8], Lemma 7). *Let* $q = \mathsf{poly}(n)$ *be a prime,* $3d \leq n$, $dt/n = \Omega(\log n)$, $\sigma = \ell n^{c(\mathfrak{d}-1)} \cdot \omega(n^2\log^2 n)$ *for some constant* c, $\gamma = \frac{n+2d-2}{d}$ *is an integer,* $\tau = \lceil \log_2 q \rceil$ *and* $t' = t+\gamma\tau$. *Then there exists a ppt algorithm* SampleRight *that on input polynomial vector* $\overrightarrow{g} \in (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$, *a non-zero integer* $y \in \mathbb{Z}^*$, *polynomial vector* $\overrightarrow{a} = (a_1, a_2, \cdots, a_{t+\gamma\tau})$, *and polynomial matrix* $\overrightarrow{S} = (\overrightarrow{s_1}, \overrightarrow{s_2}, \cdots, \overrightarrow{s_{t'}})^T$ *where* $\overrightarrow{s_j} \in (\mathbb{Z}_q^d)^t \times (\mathbb{Z}_q)^{\gamma\tau}$ *for* $1 \leq j \leq \gamma\tau$ *and a syndrome* $u \in \mathbb{Z}_q^{n+2d-2}[x]$, *outputs polynomial vector* $\overrightarrow{r} = (r_i)_{i=1}^{t+2\gamma\tau}$ *satisfying* $\sum_{i=1}^{t+\gamma\tau} a_i \cdot r_i + \sum_{i=1}^{\gamma\tau} h_i \cdot r_{t+i+\gamma\tau} = u$, *where* $h_i = \overrightarrow{s_i} \cdot \overrightarrow{a} + y\overrightarrow{g}$ *for* $i \in [\gamma\tau]$

in $\tilde{O}(nt)$ time. And the output distribution of (r_i) is exactly the conditional distribution $(D_{\mathbb{Z}^{2d-1},\sigma}[x])^t \times (D_{\mathbb{Z}^d,\sigma}[x])^{2\gamma\tau} | \sum_{i=1}^{t+\gamma\tau} a_i \cdot r_i + \sum_{i=1}^{\gamma\tau} h_i \cdot r_{i+t+\gamma\tau} = u'$.

proof The proof is exactly the same as that of Lemma 7 in [8] except for the value of σ. Let $\mathbf{S}$ be the matrix corresponding to $\overrightarrow{S}$. We can get that $\sigma_1(\tilde{\mathbf{S}}) = \sqrt{\gamma d\tau((2d-1)t + d\gamma\tau)} + \ell\omega(\gamma d\tau((2d-1)t + d\gamma\tau))\delta^{\mathfrak{d}-1}$. So this algorithm is correct by setting $\sigma = \ell n^{c(\mathfrak{d}-1)} \cdot \omega(n^2 \log^2 n)$. □

4 Our Middle-Product IBE

*Injective Map*Let $\mathfrak{d}$ and ℓ be some integers, and $\kappa = \lceil \ell^{\frac{1}{\mathfrak{d}}} \rceil$. For any element in $[1, \ell]$, there exists an injective map that maps the element to an element in $[1, \kappa]^{\mathfrak{d}}$. Hence, there exists an injective map that maps any element in $\{0,1\}^\ell$ to a subset of $\{[1, \kappa]^{\mathfrak{d}}\}$. The latter injective map can be achieved by naturally extending the previous one. In this way, we can define an effectively computable injective function $\mathfrak{F}$ that maps a bit-string $\mathsf{id} \in \mathcal{ID} = \{0,1\}^\ell$ to $\mathfrak{F}(\mathsf{id}) \subset \{[1, \kappa]^{\mathfrak{d}}\}$. *Homomorphic Computation*Let $\mathfrak{d} \in \mathbb{N}$ be a constant. Let $\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}} \in \mathbb{Z}_q^{<n+d-1}$ be polynomial vectors. We introduce a function $\mathsf{PubEval}_{\mathfrak{d}} : (\mathbb{Z}_q^{<n+d-1})^{\mathfrak{d}} \mapsto \mathbb{Z}_q^{<n+d-1}$ as follows:$\mathsf{PubEval}_{\mathfrak{d}}(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}}) = \overrightarrow{h_1}$, if $\mathfrak{d} = 1$; $\mathsf{PubEval}_{\mathfrak{d}}(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}}) = (\overrightarrow{h_1}^T \cdot \overrightarrow{g_b}^{-1}(\mathsf{PubEval}(\overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}})^T))^T$ if $\mathfrak{d} \geq 2$.

Remark 2. In this paper, we define a series of matrices (denoted by $\overrightarrow{S}$, $\overrightarrow{S'}$, $\overrightarrow{S''}$, $\overrightarrow{S_{i,j}}$, ...) which enjoy the following properties: Without loss of generality, here we will denote the polynomial matrix by $\overrightarrow{S}$. 1. $\overrightarrow{S} = [\overrightarrow{s_1}, \overrightarrow{s_2}, \cdots, \overrightarrow{s_{t'}}]^T$, where $\overrightarrow{s_j} \leftarrow (\Psi^d[x])^t \times \Psi^{\gamma\tau}$ for $\Psi := U(\{-1, 0, 1\})$, $j \in [t']$. 2. We use the scalar matrix $\tilde{\mathbf{S}}$ to denote the Toepliz matrix corresponding to $\overrightarrow{S}$. Here $\tilde{\mathbf{S}}$ is defined as follows:

$$\tilde{\mathbf{S}} = \left[\begin{array}{ccc} \mathsf{T}^{d,d}(s_{1,1}) & \ldots & \mathsf{T}^{d,d}(s_{1,\gamma\tau}) \\ \vdots & \vdots & \vdots \\ \mathsf{T}^{d,d}(s_{t,1}) & \ldots & \mathsf{T}^{d,d}(s_{t,\gamma\tau}) \\ \hline \mathsf{T}^{1,d}(s_{t+1,t+1}) & \ldots & \mathsf{T}^{1,d}(s_{t+1,\gamma\tau}) \\ \vdots & \vdots & \vdots \\ \mathsf{T}^{1,d}(s_{t',t+1}) & \ldots & \mathsf{T}^{1,d}(s_{t',\gamma\tau}) \end{array}\right].$$

Lemma 6 ([10], Lemma 2, Lemma 6). *Let polynomial vector $\overrightarrow{a} = (a_1, a_2, \cdots, a_{t+\gamma\tau})$, and polynomial matrices $\overrightarrow{S_1}, \ldots \overrightarrow{S_{\mathfrak{d}}}$, where for $0 \leq i \leq \mathfrak{d}$, $\overrightarrow{S_i}$ is generated following Remark 2. Let $y_1, y_2, \cdots, y_{\mathfrak{d}} \in \mathbb{Z}_q$. Let $\overrightarrow{h_i} = \overrightarrow{S_i} \cdot \overrightarrow{a} + y_i \cdot \overrightarrow{g}$. Further, let $\tilde{\mathbf{S}}_\mathbf{i}$ be the Toepliz matrix corresponding to $\overrightarrow{S_i}$ generated following Remark 2, we can get that $\sigma_1(\tilde{\mathbf{S}}_\mathbf{i}) \leq \sqrt{\gamma d\tau((2d-1)t + d\gamma\tau)}$ and assume $|y_i| \leq \delta$ for $i \in \mathfrak{d}$, and $\delta > \sqrt{\gamma d\tau((2d-1)t + d\gamma\tau)}$. Then there exists an efficient algorithm $\mathsf{TrapEval}_{\mathfrak{d}}$ that on input $\overrightarrow{S_1}, \ldots, \overrightarrow{S_{\mathfrak{d}}}$, $y_1, \ldots, y_{\mathfrak{d}}$ outputs $\overrightarrow{S'}$ s.t.*

$$\mathsf{PubEval}_{\mathfrak{d}}(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}}) = \overrightarrow{S'} \cdot \overrightarrow{a} + y_1 y_2 \cdots y_{\mathfrak{d}} \overrightarrow{g}. \quad (1)$$

Let $\tilde{\mathbf{S}}'$ be the Toepliz matrix corresponding to $\overrightarrow{S'}$.
Moreover, we have $\sigma_1(\tilde{\mathbf{S}}') \leq \sqrt{\gamma d\tau((2d-1)t + d\gamma\tau)}\mathfrak{d}\delta^{\mathfrak{d}-1}$

proof We will prove it by mathematical induction. **Case 1** When $\mathfrak{d} = 1$, Eq. 1 holds trivially. **Case 2** When $\mathfrak{d} \geq 2$, we may assume that $\mathsf{PubEval}_{\mathfrak{n}}(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{n}}}) = \overrightarrow{S''} \cdot \overrightarrow{a} + y_1 y_2 \cdots y_{\mathfrak{n}} \overrightarrow{g}$, holds for all $\mathfrak{n} \leq \mathfrak{d}$. We have $\mathsf{PubEval}_{\mathfrak{d}}(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}})^T = (\overrightarrow{S_1}^T \overrightarrow{g_b}^{-1}(\mathsf{PubEval}(\overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}})^T) + y_1 \overrightarrow{S''}^T)^T \overrightarrow{a} + y_1 y_2 \cdots y_{\mathfrak{d}} \overrightarrow{g}$. Set $\overrightarrow{S'} = \overrightarrow{g_b}^{-1}(\mathsf{PubEval}(\overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}}))\ \overrightarrow{S_1} + y_1 \overrightarrow{S''}$, we can get that Eq. 1 holds. We may set $\overrightarrow{g_b}^{-1}(\mathsf{PubEval}(\overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathfrak{d}}})^T) = R$, and $\tilde{\mathbf{R}}$ be its corresponding Toepliz matrix. From Lemma2 and 6 in [10], $\sigma_1(\tilde{\mathbf{S}}') \leq \sigma_1(\tilde{\mathbf{S}}_1)\delta^{\mathfrak{d}-1} + \sigma_1(\tilde{\mathbf{S}}_1)\sigma_1(\tilde{\mathbf{R}})(\delta^{\mathfrak{d}-1}-1)/(\delta-1)$. In our setting, we have $\sigma_1(\tilde{\mathbf{R}}) = b\sigma_1(\tilde{\mathbf{S}}_1)$, we have $\sigma_1(\tilde{\mathbf{S}}') = \omega(\gamma d\tau((2d-1)t + d\gamma\tau))\delta^{\mathfrak{d}-1}$. □

4.1 Our IBE Scheme

Overview of Our Construction In our scheme, let $\mathfrak{d} \in \mathbb{N}$ be a flexible constant. Set the identity space to be $\mathcal{ID} = \{0,1\}^{\ell}$ for some constant $\ell \in \mathbb{N}$. There exists an efficiently computable injective mapping $\mathfrak{F}$ that maps an identity $\mathsf{id} \in \mathcal{ID}$ to a subset of $\{[1,\kappa]^{\mathfrak{d}}\}$ where $\kappa = \lceil \ell^{\frac{1}{\mathfrak{d}}} \rceil$. Then we define a hash function $h(\cdot)$ using homomorphic computation defined in section 4. $\overrightarrow{h_{\mathsf{id}}} = h(\mathsf{id}) = \overrightarrow{h_0} + \sum_{(j_1,j_2,\ldots,j_{\mathfrak{d}}) \in \mathfrak{F}(\mathsf{id})} \mathsf{PubEval}(\overrightarrow{h_{1,j_1}}, \ldots, \overrightarrow{h_{\mathfrak{d},j_{\mathfrak{d}}}})$. The master public key contains a polynomial u, a polynomial vector $\overrightarrow{a} \in (\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ together with its trapdoor td generated by $\mathsf{TrapGen}$ algorithm introduced in Theorem 3 and polynomial vectors $\{\overrightarrow{h_0}, \{\overrightarrow{h_{i,j}}_{(i,j)\in[\mathfrak{d},\kappa]}\}\}$. Set $t' = t + \gamma\tau$ and $t'' = t' + \gamma\tau$, the secret key for the identity is a polynomial vector $\overrightarrow{r}$ s.t. $[\overrightarrow{a}^T | \overrightarrow{h_{\mathsf{id}}}^T]^T \cdot \overrightarrow{r} = u$. The polynomial vector $\overrightarrow{r}$ can be sampled by $\mathsf{SampleLeft}$ algorithm defined in [8] Lemma 6. using trapdoor td. Let $q = q(n)$ be prime, $\tau := \lceil log_2 q \rceil$, $n, d, k \in \mathbb{N}$ satisfying $\gamma = \frac{n+2d-2}{d} \in \mathbb{N}$ and $2d + k \leq n$. Let $t > 0$, $t' = t + \gamma\tau$, $t'' = t' + \gamma\tau$. Set $\chi := \lfloor D_{\alpha \cdot q} \rceil, \chi_1 := \lfloor D_{\alpha_1 \cdot q} \rceil$ be the distributions over $\mathbb{Z}$ where $\epsilon \leftarrow D_{\alpha \cdot q}$, or $\epsilon_1 \leftarrow D_{\alpha_1 \cdot q}$ is sampled and then rounded to the nearest integer respectively. Let $0 < s \leq d/2$ be an integer. Let $\sigma = c\ell \cdot \omega(n \log^{3/2} n)$ for some constant c. We define an IBE scheme with message space $\mathcal{M} = \{0,1\}^{<k+1}[x]$.

Key Generation $\mathsf{KeyGen}(1^n)$: Generate $\overrightarrow{a} = (a_1, \ldots, a_n) \in (\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ and its trapdoor td using $\mathsf{TrapGen}$. Sample $\overrightarrow{h_0} \xleftarrow{\$} \mathbb{Z}_q^{<n+d-1}$, $\overrightarrow{h_{i,j}} \xleftarrow{\$} \mathbb{Z}_q^{<n+d-1}$ for $(i,j) \in [\mathfrak{d}] \times [\kappa]$. Sample $u \xleftarrow{\$} \mathbb{Z}_q^{n+2d-2}$. Define $h(\mathsf{id}) = \overrightarrow{h_0} + \sum_{(j_1,j_2,\ldots,j_{\mathfrak{d}}) \in \mathfrak{F}(\mathsf{id})} \mathsf{PubEval}(\overrightarrow{h_{1,j_1}}, \ldots, \overrightarrow{h_{\mathfrak{d},j_{\mathfrak{d}}}})$. Output $\mathsf{mpk} = \{\overrightarrow{a}, \overrightarrow{h_0}, \{\overrightarrow{h_{i,j}}_{(i,j)\in[\mathfrak{d},\kappa]}\}, u\}$, $\mathsf{msk} = \mathsf{td}$.

Extraction $\mathsf{Extract}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$: Let $\overrightarrow{h_{\mathsf{id}}} = h(\mathsf{id})$. Use $\mathsf{SampleLeft}$ ($\overrightarrow{a}$, $\overrightarrow{h_{\mathsf{id}}}$, td, u, σ) to generate $(r_1, \ldots, r_{t''})$ s.t. $\sum_{i=0}^{t'} a_i\ r_i + \sum_{i=1}^{\gamma\tau} h_{\mathsf{id},i}\ r_{i+t'} = u$. Output $\mathsf{sk}_{\mathsf{id}} = (r_1, \cdots, r_{t''})$. Note that $[\overrightarrow{a}^T | \overrightarrow{h_{\mathsf{id}}}^T]\mathsf{sk}_{\mathsf{id}} = u$.

Encryption $\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, m)$: Compute $\overrightarrow{h_{\mathsf{id}}} = h(\mathsf{id})$. Sample $p \xleftarrow{\$} \mathbb{Z}_q^{<n+2d+k-1}[x]$. For $1 \leq i \leq t$, sample $e_i \leftarrow \chi^{2d+k}[x]$, and compute $c_i = a_i \odot_{2d+k} p + e_i$. For $t+1 \leq i \leq t'$, sample $e_i \leftarrow \chi^{d+k+1}[x]$, and compute $c_i = a_i \odot_{d+k+1} p + e_i$. For $1 \leq$

$i \leq \gamma\tau$, sample $e_{i+t'} \leftarrow \chi_1^{d+k+1}[x]$, compute $c_{i+t'} = h_{\mathsf{id},i} \odot_{d+k+1} p + e_{i+t'}$. Sample $e_0 \leftarrow \chi^{k+1}$, and compute $c_0 = m\lfloor \frac{q}{2} \rfloor + u \odot_{k+1} p + e_0$. Output $c = (c_0, c_1, \cdots, c_{t''})$.
Decryption $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, c) : w = c_0 - \sum_{i=1}^{t''} c_i \odot_{k+1} r_i \mod q$. Output $\lfloor \frac{w}{q/2} \rfloor$.

Lemma 7. *Our IBE scheme satisfies* $1-\mathsf{negl}(n)$ *correctness when* $\alpha = (n^{2c(\mathfrak{d}-1)+4} l^2 \mathfrak{d}^2 \cdot \omega(\log^{\frac{9}{2}} n) + 1)^{-1}$ *and* $\alpha_1 = (n^{c(\mathfrak{d}-1)+\frac{5}{2}} \ell \mathfrak{d} \cdot \omega(\log^3 n) + 1)^{-1}$.

proof We aim to show that $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, m)) = m$ with probability $1 - \mathsf{negl}(n)$ over the randomness of KeyGen and Enc. Consider a random key pair $(\mathsf{mpk}, \mathsf{msk})$ and a random identity $\mathsf{id} = \{0,1\}^\ell$. Let $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{Extract}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$ and compute ciphertext $c = (c_0, (c_i)_{1 \leq i \leq t''}) \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, m)$. By Lemma 3 (the quasi-associative law for middle products), we have $c_0 = m + (\sum_{i=0}^{t'} a_i r_i + \sum_{i=1}^{\gamma\tau} h_{\mathsf{id},i} r_{i+t'}) \odot_{k+1} p + e' = m + \sum_{i=1}^{t} r_i \odot_{k+1} (a_i \odot_{2d+k} p) + \sum_{i=t+1}^{t'} r_i \odot_{k+1} (a_i \odot_{d+k+1} p) + \sum_{i=1}^{\gamma\tau} r_{i+t'} \odot_{k+1} (h_{\mathsf{id},i} \odot_{d+k+1} p) + e'$, which leads to the following equation $c_0 - \sum_{i=1}^{t''} c_i \odot_{k+1} r_i = m + (e_0 - \sum_{i=1}^{t''} r_i \odot_{k+1} e_i)$. So if $\|m + (e_0 - \sum_{i=1}^{t''} r_i \odot_{k+1} e_i)\|_\infty < q/2$, $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, c)$ will output the message m correctly. Then, we will calculate the bound of the coefficients in $\sum_{i=1}^{t''} r_i \odot_{k+1} e_i$ to make the correctness criterion of our scheme satisfied. The coefficient of x^l in $r_i \odot_{k+1} e_i$ equals $\sum_{\omega \in [0, deg(r_i)] \cap [l+k-deg(e_i), z+k]} (\mathbf{y}_i)_\omega (\mathbf{e}_i)_{l+k-\omega}$. Applying the Gaussian tail inequality and a union bound, we can get the following bounds on $\|\mathbf{y}_i\|_\infty$ and $\|\mathbf{e}_i\|_\infty$: $\Pr[\|\mathbf{y}_i\|_\infty > \omega(\sqrt{\log n})\sigma)] = \mathsf{negl}(n)$ and $\Pr[\|\mathbf{e}_i\|_\infty > \omega(\sqrt{\log n})\alpha_1 q)] = \mathsf{negl}(n)$ as $\alpha_1 > \alpha$. Therefore, again by union bound, except with $\mathsf{negl}(n)$ probability $\|e_0 - \sum_{i=1}^{t''} r_i \odot_{k+1} e_i\|_\infty < K\omega(\sqrt{\log n})\sigma)\omega(\sqrt{\log n})\alpha_1 q) + \omega(\sqrt{\log n})\alpha_1 q)$ for $K := (2d-1)t + 2\gamma\tau d \geq \sum_{i=1}^{t''}(deg(r_i)+1)$. Setting $\alpha_1 < (4\sigma K \cdot \omega(\log n) + 1)^{-1} = (n^{c(\mathfrak{d}-1)+3} \ell \cdot \omega(\log^4 n) + 1)^{-1}$, the above is less than $q/4$ and thus the scheme is $(1 - \mathsf{negl}(n))$-correct. In this case, $\alpha = (n^{2c(\mathfrak{d}-1)+6} \ell^2 \cdot \omega(\log^{\frac{13}{2}} n) + 1)^{-1}$. □

5 Security Proof

In this section, we are to prove the security of our IBE scheme. The main idea of our proof is the partition technique following [19, 4, 3, 20, 21, 8].

Lemma 8 ([20], Lemma 8 Claim 1). *Let* q *be a prime and* $0 \leq Q \leq \frac{n^{\bar{c}}}{2} - 1$. *Define the hash function* $F_{\mathrm{Wat}} : \{F_{\mathbf{y}} : (\mathbb{Z}_q)^\ell \mapsto \mathbb{Z}_q\}$ *as :* $F_{\mathbf{y}}(\mathsf{id}) = y_0 + \sum_{(j_1, j_2, \ldots, j_{\mathfrak{d}}) \in \mathfrak{F}(\mathsf{id})} y_{1,j_1} \cdots y_{\mathfrak{d},j_{\mathfrak{d}}}$. *The hash function* F_{Wat} *is* $(Q, \alpha_{\min}, \alpha_{\max})$ *abort-resistant, with* $\alpha_{\min} = \frac{1}{\kappa+1}(\frac{1}{\mathfrak{d} n^{\bar{c}}})^{\mathfrak{d}}(1 - \frac{Q}{n^{\bar{c}}})$, $\alpha_{\max} = \frac{1}{\kappa+1}(\frac{1}{\mathfrak{d} n^{\bar{c}}})^{\mathfrak{d}}$.

For identity queries $\{\mathsf{id}_1, \cdots, \mathsf{id}_Q\}$ and challenge identity id^*, we further define the probability that for all key extraction queries, $F_{\mathbf{y}}(\mathsf{id}_i) \neq 0$ and for the challenge identity id^*, $F_{\mathbf{y}}(\mathsf{id}^*) = 0$ to be $\Upsilon(\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q)$. With artificial abort, according to [10], $\Upsilon(\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q)$ is non-negligible for all possible combinations of $(\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q)$. In our setting $\overrightarrow{g}$ is equipped with a trapdoor, which is publicly known [13]. Then we transform the polynomial vectors $\overrightarrow{a}$, $\overrightarrow{g}$

and $\overrightarrow{S_{\mathsf{id}}}$ to matrices $\mathbf{A}$, $\mathbf{G}$ and $\mathbf{S}$ respectively. The goal of the key extraction query is to find $\mathbf{e}$, the vector representation of $\mathsf{sk}_{\mathsf{id}}$, satisfying $[\mathbf{A}|\mathbf{SA}+F_{\mathbf{y}}(\mathsf{id})\mathbf{G}]\mathbf{e}=\mathbf{u}$. Here $\mathbf{u}$ is the vector representation of the polynomial u predefined in mpk. As $[\mathbf{A}|\mathbf{SA}+F_{\mathbf{y}}(\mathsf{id})\mathbf{G}]\left[-\mathbf{S}^T|\mathbf{I}\right]^T = F_{\mathbf{y}}(\mathsf{id})\mathbf{G}$, when $F_{\mathbf{y}}(\mathsf{id})\neq 0$, $\mathsf{SampleRight}$ algorithm works and the key extraction queries can be answered. When it comes to the challenge identity id^*, $F_{\mathbf{y}}(\mathsf{id}^*)=0$ and $\overrightarrow{h_{\mathsf{id}}}=\overrightarrow{S_{\mathsf{id}}}\cdot\overrightarrow{a}$. Set $(c_i)_{i=1}^{t'}$ and $c_0-\lfloor\frac{q}{2}\rfloor m_{\mathsf{coin}}$ to be the MPLWE samples, where m_{coin} $\mathsf{coin}\xleftarrow{\$}\{0,1\}$ is the challenge message. Then the rest of ciphertext, i.e. $(c_i)_{i=t'+1}^{t''}$, will be generated using $(c_i)_{i=1}^{t'}$ and the rerandomization algorithm ReRand from [10]. At this final stage, the ciphertext contains no information about coin under the MPLWE assumption. Our IBE is IND-CPA secure with the parameter setting given below.

Parameter Settings The parameters used in our IBE are listed below.

$$d=\Theta(n) \qquad t=\Omega(\log n)$$
$$q=n^{2c(\mathfrak{d}-1)+c'+7}l^2\cdot\omega(\log^{\frac{13}{2}}) \qquad \sigma=\ell n^{c(\mathfrak{d}-1)}\cdot\omega(n^2\log^2 n)$$
$$\alpha=(\ell^2(n^{2c(\mathfrak{d}-1)+6}\cdot\omega(\log^{\frac{13}{2}}n)+1)^{-1} \quad \alpha_1=(\ell n^{c(\mathfrak{d}-1)+3}\cdot\omega(\log^4 n)+1)^{-1}$$

Theorem 5. *Assume that* $\sigma=\ell\mathfrak{d}n^{c(\mathfrak{d}-1)+\frac{3}{2}}\cdot\omega(\log n)$ *for some constant* c, $dt/n=\Omega(\log n)$, $q=\mathsf{poly}(n)$ *is a prime number,* $q=\Omega(\alpha^{-1}n^{1+c})$ *and* $q=\sigma\cdot\omega(\log^{1/2}n)$, $\alpha_1<(4\sigma K\cdot\omega(\log n)+1)^{-1}$, *and* $\alpha=\alpha_1/(2\sqrt{t'}\ell\mathfrak{d}n^{c(\mathfrak{d}-1)+3/2}\cdot\omega(\log n))$ *where* $K:=(2d-1)t+2\gamma\tau d$, *the IBE system is IND-CPA adaptively secure assuming* $\mathrm{MPLWE}_{q,n+2d+k,\mathbf{d},D_{\alpha\cdot q}}$ *is hard with degree vector* $\mathbf{d}=(d_i)_{i=1}^{t'+1}$ *where* $d_i=2d+k$, *for* $1\leq i\leq t$; $d_i=d+k+1$, *for* $t+1\leq i\leq t'$; $d_i=k+1$, *for* $i=t''+1$.

Proof: Let $\mathcal{A}$ be a PPT adversary to break our scheme. Let $\epsilon=\epsilon(n)$ $\mathcal{A}$'s advantage. Let $Q=Q(n)$ be the upper bound of the number of key extraction queries $\mathcal{A}$ makes. Here, without the loss of generality, we may assume that $\mathcal{A}$ always makes Q queries. Then we define constant $\tilde{c}$ s.t. $Q\leq\frac{n^{\tilde{c}}}{2}-1$. We can see that such $\tilde{c}$ always exists. Our proof consists of a sequence of games. Among them, the first game is identical to the IND-CPA adaptive game from Section 2.2. And in the last game in the sequence, $\mathcal{A}$ has a negligible advantage to win. In Game 1, we employ the artificial abort to ensure that the queried identities $(\mathsf{id}_1,\ldots,\mathsf{id}_Q)$ and the challenge id id^* selected by the adversary $\mathcal{A}$ satisfy our requirement. From Game 2 to Game 4, the way mpk is generated is changed gradually. After the alternation of mpk, $\mathcal{C}$ is able to answer all the key extraction queries with the new embedded trapdoor, while for the challenge identity id^*, the trapdoor does not exit. From Game 5 to Game 7, the generation method of the ciphertext is gradually changed. In Game 7, the $\mathcal{A}$ has no advantage to win the game. The MPLWE assumption is used to prove that Game 6 and Game 7 are indistinguishable. In each game, two values $\mathsf{coin}',\widetilde{\mathsf{coin}}\in\{0,1\}$ are defined. While we set $\mathsf{coin}'=\widetilde{\mathsf{coin}}$ in the first game, these values might be different in the later guess when artificial abort occurs. Finally, we define X_i to be the event that $\mathsf{coin}'=\mathsf{coin}$ at the end of event Game i.

Game 0. This is the original game, identical to the IND-CPA adaptive game in Section 2.2, between a PPT adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. In the challenge

phase, the ciphertext is set to be $c = (c_0, (c_i)_{i \le t''}) \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}^*, m_{\mathsf{coin}})$. Here mpk is generated in the setup phase and id^* is the challenge identity. At the end of the game, $\mathcal{A}$ outputs a guess $\widetilde{\mathsf{coin}}$ for coin. Finally, $\mathcal{C}$ sets $\mathsf{coin}' = \widetilde{\mathsf{coin}}$. By definition, we have $|\Pr[X_0] - \frac{1}{2}| = |\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}|$.

Game 1. Game 1 is identical to Game 0 except that we add an artificial abort at the beginning of the game. The challenger $\mathcal{C}$ picks $\mathbf{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[\mathfrak{d}]\times[\kappa]})$ where $y_0 \xleftarrow{\$} [-(\kappa+1)(\mathfrak{d}n^{\tilde{c}})+1, 0]$ and $y_{i,j} \xleftarrow{\$} [1, \mathfrak{d}n^{\tilde{c}}]$ for $(i,j) \in [\mathfrak{d}]\times[\kappa]$. Then define the function $F_{\mathbf{y}} : \mathcal{ID} \mapsto \mathbb{Z}_q$ as follows: $F_{\mathbf{y}}(\mathsf{id}) = y_0 + \sum_{(j_1,j_2,\ldots,j_{\mathfrak{d}})\in\mathfrak{F}(\mathsf{id})} y_{1,j_1} \cdots y_{\mathfrak{d},j_{\mathfrak{d}}}$. Recall that by Lemma 8, F_{Wat} is a $(Q, \alpha_{\min}, \alpha_{\max})$ abort-resistant hash function. Then $\mathcal{C}$ checks whether the following condition holds:

$$F_{\mathbf{y}}(\mathsf{id}^*) = 0 \wedge F_{\mathbf{y}}(\mathsf{id}_1) \neq 0 \wedge F_{\mathbf{y}}(\mathsf{id}_2) \neq 0 \wedge \cdots \wedge F_{\mathbf{y}}(\mathsf{id}_Q) \neq 0 \tag{2}$$

where id^* is the challenge identity and $\mathsf{id}_1, \cdots \mathsf{id}_Q$ are the identities on which $\mathcal{A}$ has made key extraction queries. And set $\Upsilon(\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q) = \Pr_{\mathbf{y}}[F_{\mathbf{y}}(\mathsf{id}^*) = 0 \wedge F_{\mathbf{y}}(\mathsf{id}_1) \neq 0 \wedge F_{\mathbf{y}}(\mathsf{id}_2) \neq 0 \wedge \cdots \wedge F_{\mathbf{y}}(\mathsf{id}_Q) \neq 0]$. Here we call $F_{\mathbf{y}}(\mathsf{id})$ an abort-resistant hash function. Then $\mathcal{C}$ does as follows. (I) *Abort Check*: If (2) does not hold, $\mathcal{C}$ sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$. In this case, we say that the challenger aborts. If condition (2) holds, the challenger $\mathcal{C}$ sets $\mathsf{coin}' = \widetilde{\mathsf{coin}}$. (II) *Artificial Abort*: As in [3], with probability $\Upsilon(\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q)$, $\mathcal{C}$ ignores $\mathcal{A}$'s output, and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$, otherwise, $\mathcal{C}$ sets $\mathsf{coin}' = \widetilde{\mathsf{coin}}$. For a $(Q+1)$ tuple of identities $\mathfrak{I} = (\mathsf{id}^*, \mathsf{id}_1, \ldots, \mathsf{id}_Q)$, let $\epsilon(\mathfrak{I})$ be the probability that an abort (either real or artificial) does not happen when extraction queries are made. Then we set $\epsilon_{\min}$ and $\epsilon_{\max}$ to be scalars s.t. $\epsilon(\mathfrak{I}) \in [\epsilon_{\min}, \epsilon_{\max}]$ for all $\mathfrak{I}$. We have the following lemma. The probability that Game 0 and Game 1 give a different output is bounded by [3] Lemma 28. Here, F_{Wat} is a$(Q, \alpha_{\min}, \alpha_{\max})$ abort-resistant hash function. If no abort occurs, $\epsilon_{\min} = \alpha_{min}$ and $\epsilon_{\max} = \alpha_{\max}$. But then $\epsilon_{\max} - \epsilon_{\min} = \frac{1}{(\kappa+1)}(\frac{1}{\mathfrak{d}n^{\tilde{c}}})^{\mathfrak{d}} \frac{Q}{n^{\tilde{c}}}$, which is non-negligible. To obtain a good lower bound on $|\Pr[X_1 - \frac{1}{2}]|$, we opt for the Waters approach [19]. We introduce the artificial abort and define $\gamma(\cdot)$ as in [19]. With the artificial abort, $(\epsilon_{\max} - \epsilon_{\min}) \le \alpha_{\min}$. So we can get $|\Pr[X_1] - \frac{1}{2}| \ge \frac{1}{2}\alpha_{\min}|\Pr[X_0 - \frac{1}{2}]| \ge \frac{1}{2(\kappa+1)}(\frac{1}{\mathfrak{d}n^{\tilde{c}}})^{\mathfrak{d}}(1 - \frac{Q}{n^{\tilde{c}}})|\Pr[X_0 - \frac{1}{2}]|$.

Game 2. Game 2 is identical to Game 1 except that we change the way that $\mathcal{C}$ generates the polynomial vectors $\overrightarrow{h_0}, \{\overrightarrow{h}_{i,j}\}_{(i,j)\in[\mathfrak{d}]\times[\kappa]}$. Instead of selecting them uniformly at random from $(\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$, $\mathcal{C}$ first generates a polynomial vector $\overrightarrow{g} = (g_1, g_2, \cdots, g_{\gamma\tau}) \in (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$ s.t. $g_j = 2^u x^{dv}$, for $j = v\tau + u + 1$, where $u \in \{0, \cdots, \tau-1\}, v \in \{0, \cdots, \gamma-1\}$. After that the challenger $\mathcal{C}$ picks $\mathbf{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[\mathfrak{d}]\times[\kappa]})$ where $y_0 \xleftarrow{\$} [-(\kappa+1)(\mathfrak{d}n^{\tilde{c}})+1, 0]$ and $y_{i,j} \xleftarrow{\$} [1, \mathfrak{d}n^{\tilde{c}}]$ for $(i,j) \in [\mathfrak{d}]\times[\kappa]$. Then $\mathcal{C}$ generates polynomial matrices $\overrightarrow{S_0}, \{\overrightarrow{S}_{i,j}\}_{(i,j)\in[\mathfrak{d}]\times[\kappa]}$ according to Remark 2. Next, set $\overrightarrow{h_0} = \overrightarrow{S_0} \cdot \overrightarrow{a} + y_0 \overrightarrow{g}$, and $\overrightarrow{h_{i,j}} = \overrightarrow{S_{i,j}} \cdot \overrightarrow{a} + y_{i,j} \overrightarrow{g}$. Then by [8] Theorem 5, the distribution of h_0 and $h_{i,j}$ are indistinguishable in Game 2 and Game 1, i.e. $\overrightarrow{h_0} \approx_c U((\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau})$ and $\overrightarrow{h_{i,j}} \approx_c U((\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau})$ for $(i,j) \in [\mathfrak{d}]\times[\kappa]$. Therefore, Game 2 is computa-

tionally indistinguishable from Game 1, i.e. $|\Pr[X_2] - \Pr[X_1]| = \mathsf{negl}(n)$.

Game 3. In this game, $\mathcal{C}$ changes the time the abortion is made. In game 2, $\mathcal{C}$ aborts at the end of the game if the condition (2) does not hold. Here, the challenger aborts as soon as the abort condition is satisfied. Since this is only a conceptual change, we have $\Pr[X_3] = \Pr[X_2]$.

Game 4. In this game, instead of generating $\overrightarrow{a}$ together with a trapdoor td according to KeyGen, $\mathcal{C}$ picks $\overrightarrow{a} \xleftarrow{\$} (\mathbb{Z}_q^{<n}[x])^t \times (\mathbb{Z}_q^{<n+d-1}[x])^{\gamma\tau}$. By Theorem 2, this only differs negligibly with game 3. Then $\mathcal{C}$ sets $\overrightarrow{S_{\mathsf{id}}} = \overrightarrow{S_0} + \sum_{(j_1,j_2,\ldots,j_{\mathfrak{d}})\in\mathfrak{F}(\mathsf{id})} \mathsf{TrapEval}(\overrightarrow{S_{1,j_1}}, \ldots, \overrightarrow{S_{\mathfrak{d},j_{\mathfrak{d}}}}, y_{1,j_1}, \ldots, y_{\mathfrak{d},j_{\mathfrak{d}}})$. It holds that $\overrightarrow{h_{\mathsf{id}}} = \overrightarrow{S_{\mathsf{id}}} \cdot \overrightarrow{a} + F_{\mathbf{y}}(\mathsf{id})\overrightarrow{g}$. When $\mathcal{A}$ makes a key extraction query. If the abortion does not happen, it runs $\mathsf{SampleRight}(\overrightarrow{g}, \mathsf{td}, F_{\mathbf{y}}(\mathsf{id}), \overrightarrow{a}, \overrightarrow{S_{\mathsf{id}}}, u, \sigma) \to \overrightarrow{r}$ and returns $\overrightarrow{r}$ to $\mathcal{A}$. Compared with Game 3, where the key $\overrightarrow{r}$ is sampled using $\mathsf{SampleLeft}(\overrightarrow{a}, \overrightarrow{h_{\mathsf{id}}}, \mathsf{td}, u, \sigma) \to \overrightarrow{r}$. By [8] Lemma 6., Lemma 5 and the choice of σ, the difference in the output distributions of $\mathsf{SampleRight}$ and $\mathsf{SampleLeft}$ are $\mathsf{negl}(n)$-close. Therefore, we have $|\Pr[X_4] - \Pr[X_3]| = \mathsf{negl}(n)$.

Game 5. In this game, $\mathcal{C}$ changes the way the ciphertext is generated. If $F_{\mathbf{y}}(\mathsf{id}^*) = 0$ (i.e. if it does not abort), to create the challenge ciphertext, $\mathcal{C}$ chooses $p \xleftarrow{\$} \mathbb{Z}_q^{<n+2d+k-1}[x]$, $\chi \leftarrow \lfloor D_{\alpha\cdot q}\rceil$, $\chi_1 \leftarrow \lfloor D_{\alpha_1\cdot q}\rceil$, and noise $e_0, e_1, e_2, \cdots, e_{t''}$, where $e_i \leftarrow \chi^{k+1}$ for $i = 0$, $e_i \leftarrow \chi^{2d+k}[x]$ for $1 \le i \le t$, $e_i \leftarrow \chi^{d+k+1}[x]$ for $t+1 \le i \le t'$, $e_i \leftarrow \chi_1^{d+k+1}[x]$ for $t'+1 \le i \le t''$. Next, for $1 \le i \le t$ set $c_i = a_i \odot_{2d+k} p + e_i$. For $t+1 \le i \le t'$ set $c_i = a_i \odot_{d+k+1} p + e_i$. For $t = 0$, set $c_0 = \lfloor \frac{q}{2} \rfloor m_{\mathsf{coin}} + u \odot_{k+1} p + e_i$. For $t'+1 \le i \le t''$, WLOG, c_ι, where $t'+1 \le \iota \le t''$, will be taken as an example. Since $c_\iota = h_{\mathsf{id},\iota} \odot_{d+k+1} p + e_\iota$, and $h_{\mathsf{id}} = \overrightarrow{S_{\mathsf{id}}} \cdot \overrightarrow{a}$ as $F_{\mathbf{y}}(\mathsf{id}^*) = 0$, the following equations hold: $c_\iota = h_{\mathsf{id},\iota} \odot_{d+k+1} p + e_\iota = (\sum_{i=1}^{t'} s_i \cdot a_i) \odot_{d+k+1} p + e_\iota = \sum_{i=1}^{t} (s_i \cdot a_i) \odot_{d+k+1} p + \sum_{i=t+1}^{t'} (s_i \cdot a_i) \odot_{d+k+1} p + e_\iota = \sum_{i=1}^{t} s_i \odot_{d+k+1} (a_i \odot_{2d+k} p) + \sum_{i=t+1}^{t'} s_i (a_i \odot_{d+k+1} p) + e_\iota$, where $(s_1, s_2, \cdots, s_{t'})$ are the entries on the ι-th row of the polynomial matrix $\overrightarrow{S_{\mathsf{id}}}$. The fourth equality holds by Lemma 3, and $s_i \in \mathbb{Z}_q$ for $t+1 \le i \le t'$. Next, $\mathcal{C}$ sets $\phi_i = a_i \odot_{2d+k} p$, for $1 \le i \le t$, $\phi_i = a_i \odot_{d+k+1} p$, for $t+1 \le i \le t'$, and uses $\boldsymbol{\phi}_i$ to represent its coefficient vector. And the coefficient vector of e_ι is denoted by $\mathbf{e}_\iota$. Then $\mathcal{C}$ sets $\overline{\mathbf{c}_\iota} = \sum_{i=1}^{t} (\mathsf{T}^{d,d+k+1}(s_i))^T \overline{\boldsymbol{\phi}_i} + \sum_{i=t+1}^{t'} s_i \overline{\boldsymbol{\phi}_i} + 2\overline{\mathbf{e}_\iota}$. By Lemma 4, we can see that $\mathbf{c}_\iota = \overline{\overline{\mathbf{c}_\iota}}$ is the coefficient vector of c_ι. So Game 4 and Game 5 are identical and $\Pr[X_5] = \Pr[X_4]$.

Game 6. In this game, we slightly change the way the ciphertext is generated. If the game does not abort, $\mathcal{C}$ first chooses $p \xleftarrow{\$} \mathbb{Z}_q^{<n+2d+k-1}[x]$, $\chi \leftarrow \lfloor D_{\alpha\cdot q}\rceil$, $\chi_1 \leftarrow \lfloor D_{\alpha_1\cdot q}\rceil$, and noise $e_0, e_1, e_2, \cdots, e_{t''}$, where $e_i \leftarrow \chi^{k+1}$ for $i = 0$, $e_i \leftarrow \chi^{2d+k}[x]$ for $1 \le i \le t$, $e_i \leftarrow \chi^{d+k+1}[x]$ for $t+1 \le i \le t'$. Next, for $1 \le i \le t$ set $c_i = a_i \odot_{2d+k} p + e_i$. For $t+1 \le i \le t'$ set $c_i = a_i \odot_{d+k+1} p + e_i$. For $t = 0$, set $c_0 = \lfloor \frac{q}{2} \rfloor m_{\mathsf{coin}} + u \odot_{k+1} p + e_i$. For $t'+1 \le i \le t''$, WLOG, we will take c_ι, where $t'+1 \le \iota \le t''$, as an example. First, since $c_i = a_i \odot_{2d+k} p + e_i$ for $1 \le i \le t$ and $c_i = a_i \odot_{d+k+1} p + e_i$, for $t+1 \le i \le t'$. And the coefficients vectors are denoted as $\mathbf{c}_i$. Then, $\mathcal{C}$ sets $\overline{\mathbf{c}_\iota} = \sum_{i=1}^{t} \mathsf{ReRand}((\mathsf{T}^{d,d+k+1}(s_i))^T, \overline{\mathbf{c}_i}, \alpha q, \frac{\alpha_1}{2\sqrt{t'\alpha}})$

$+ \sum_{i=t+1}^{t'} \mathsf{ReRand}(s_i I_{d+k+1}, \overline{\mathbf{c}_i}, \alpha q, \frac{\alpha_1}{2\sqrt{t'}\alpha})$. Set c_ι to be the polynomial whose coefficient vector is $\mathbf{c}_\iota = \overline{\mathbf{c}_\iota}$. Since $\alpha_1 = 2\sqrt{t'}\ell n^{c(\mathfrak{d}-1)} \cdot \omega(n^2 \log^2 n)\alpha$, $\frac{\alpha_1}{2\sqrt{t'}\alpha} > \ell n^{c(\mathfrak{d}-1)} \cdot \omega(n^2 \log^2 n) \geq \sigma_1(\mathsf{T}^{d,d+k+1}(s_i))$, from [10] Lemma 1, we can see that c_ι in this game is distributed statistically close to c_ι in Game 5. So we have $|\Pr[X_6] - \Pr[X_5]| = \mathsf{negl}(n)$.

Game 7. In Game 7, changes are made to the generation of ciphertext. If the game does not abort, $\mathcal{C}$ sets $c_1, c_2, \cdots, c_t \xleftarrow{\$} \mathbb{Z}_q^{2d+k}$, and $c_{t+1}, \cdots, c_{t'} \xleftarrow{\$} \mathbb{Z}_q^{d+k+1}$ and $\mathfrak{c} \xleftarrow{\$} \mathbb{Z}_q^{k+1}$, $c_0 = \mathfrak{c} + \lfloor \frac{q}{2} \rfloor m_{\mathsf{coin}}$. Then $\mathcal{C}$ generates the rest of the ciphertexts as in Game 6. We can see that $|\Pr[X_7] - \frac{1}{2}| = \mathsf{negl}(n)$ as it contains no information about the coin. By Lemma 9, we can see that $|\Pr[X_7] - \Pr[X_6]| = \mathsf{negl}(n)$, assuming $\mathrm{MPLWE}_{q,n+2d+k,\mathbf{d},D_{\alpha\cdot q}}$ is hard with degree vector $\mathbf{d} = (d_i)_{i=1}^{t'+1}$ where $d_i = 2d+k$, for $1 \leq i \leq t$; $d_i = d+k+1$, for $t+1 \leq i \leq t'$; $d_i = k+1$, for $i = t''+1$.

Analysis From the previous games, we can get that when $|\Pr[X_7] - \frac{1}{2}| = \mathsf{negl}(n)$, $|\Pr[X_0] - \frac{1}{2}| = \mathsf{negl}(n)$. The previous equation holds trivially as $\mathcal{A}$ has no information about the message m. □

Lemma 9 (Lemma11, [8]). *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that $|\Pr[X_7] - \Pr[X_6]| \leq Adv_{\mathcal{B}}^{\mathrm{MPLWE}_{q,n+2d+k,\mathbf{d},D_{\alpha\cdot q}}}$. In particular, under the $\mathrm{MPLWE}_{q,n+2d+k,\mathbf{d},D_{\alpha\cdot q}}$ assumption, we have $|\Pr[X_7] - \Pr[X_6]| = \mathsf{negl}(n)$.*

Proof Suppose $\mathcal{A}$ has non-negligible advantage in distinguishing Game 6 and 7. We use $\mathcal{A}$ to construct an MPLWE algorithm, denoted $\mathcal{B}$. Recall from Theorem 1 that an MPLWE problem instance is provided as a sampling oracle $\mathcal{O}$ that can be either truly random $\mathcal{O}_\$$ or noisy pseudo-random $\mathcal{O}_s$ for some secret $\mathbb{Z}_q^{<n+2d+k-1}[x]$. The simulator $\mathcal{B}$ uses the adversary $\mathcal{A}$ to distinguish between the two, and proceeds as follows: **Instance.** $\mathcal{B}$ requests from $\mathcal{O}$ and receives fresh pairs $(a_1^*, w_1), \cdots, (a_{t'}^*, w_{t'}), (u^*, v^*)$, where $a_i^* \in \mathbb{Z}_q^{<n}$, $w_i^* \in \mathbb{Z}_q^{<2d+n}$ for $1 \leq i \leq t$, $a_i^* \in \mathbb{Z}_q^{<n+d-1}$, $w_i^* \in \mathbb{Z}_q^{<d+k+1}$ for $t+1 \leq i \leq t'$, and $u^* \in \mathbb{Z}_q^{n+2d-2}$, $v^* \in \mathbb{Z}_q^{<k+1}$. **Setup** To construct master public key mpk, as follows: 1. Set $a_i = a_i^*$ for $i = 1, ...t'$ 2. $\mathcal{B}$ picks $\mathbf{y}$ as in Game 1. 3. Generates polynomial vectors $\overrightarrow{h_1}, \cdots, \overrightarrow{h_\mathfrak{d}}$ by picking polynomial matrices $\overrightarrow{S_1}, \overrightarrow{S_2}, \cdots, \overrightarrow{S_\mathfrak{d}}$ and generating polynomial vector $\overrightarrow{b}$ with trapdoor td as in Game 6. 4. Set $u = u^*$. $\mathcal{C}$ Then returns mpk to $\mathcal{A}$. $\mathcal{B}$ also picks a random bit $\mathsf{coin} \leftarrow \{0,1\}$ and keeps it secret. **Queries** When $\mathcal{A}$ makes a key extraction for id, $\mathcal{B}$ first calculates $F_{\mathbf{y}}(\mathsf{id})$. It aborts and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$ if $F_{\mathbf{y}}(\mathsf{id}) = 0$. Otherwise, $\mathcal{B}$ generates the private key as in Game 4. **Challenge** When $\mathcal{A}$ makes a key extraction query for the challenge identity id^*, $\mathcal{B}$ first computes $F_{\mathbf{y}}(\mathsf{id})$ and aborts and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$ if $F_{\mathbf{y}}(\mathsf{id}) \neq 0$. Otherwise, it proceeds as follows: If $\mathsf{coin} = 0$, it computes $\overrightarrow{S}_{\mathsf{id}^*} = [\overrightarrow{s_1}, \overrightarrow{s_2}, \cdots, \overrightarrow{s_{t'}}]^T$ where $\overrightarrow{s_j} \leftarrow (\chi^d[x])^t \times 0^{\gamma\tau}$ for $j = 1, \cdots, t$, $\overrightarrow{s_j} \leftarrow (\chi^d[x])^t \times \chi^{\gamma\tau}$ for $j = t+1, \cdots, t'$. And we can write $\overrightarrow{s_j}$ as $(s_{j,1}, \cdots, s_{j,t'})$. Set $\chi := \lfloor D_{\alpha\cdot q} \rceil, \chi_1 := \lfloor D_{\alpha_1\cdot q} \rceil$. Let $e_i \leftarrow \chi^{k+1}$ for $i = 0$, $e_i \leftarrow \chi^{2d+k}[x]$ for $1 \leq i \leq t$, $e_i \leftarrow \chi^{d+k+1}[x]$ for $t+1 \leq i \leq t'$, $e_i \leftarrow \chi_1^{d+k+1}[x]$ for $t'+1 \leq i \leq t'+t$, $e_i \leftarrow \chi_1^{d+k+1}[x]$ for $t'+t+1 \leq i \leq 2t'$. $c_0 = m + v^*$, $c_i = w_i^*$ for $1 \leq i \leq t'$.

For $t'+1 \leq i \leq 2t'$, generate c_i as follows: For $t'+1 \leq i \leq t'+t$, WLOG, we will take c_ι, where $t'+1 \leq \iota \leq t'+t$, as an example. First, we have $c_i = a_i \odot_{2d+k} p + e_i$, for $1 \leq i \leq t$. We will use $\mathbf{c}_i$ to represent the coefficient vectors. Then, we set $\overline{\mathbf{c}_\iota} = \sum_{i=1}^{t} \mathsf{ReRand}((\mathsf{T}^{d,d+k+1}(s_i))^T, \overline{\mathbf{c}_i}, \alpha q, \frac{\alpha_1}{2\sqrt{t}\alpha q})$. And set c_ι to be the polynomial whose coefficient vector is $\mathbf{c}_\iota$. From [10] Lemma 1, we can see that c_ι in this game is distributed statistically close to c_ι in Game 5. For $t'+t+1 \leq i \leq 2t'$, WLOG, we will take c_ι, where $t'+t+1 \leq \iota \leq 2t'$, as an example. First, since $c_i = a_i \odot_{2d+k} p + e_i$ for $1 \leq i \leq t$, $c_i = a_i \odot_{d+k+1} p + e_i$, for $t+1 \leq i \leq t'$. Use $\mathbf{c}_i$ to represent the coefficient vectors. Then, we set $\overline{\mathbf{c}_\iota} = \sum_{i=1}^{t} \mathsf{ReRand}((\mathsf{T}^{d,d+k+1}(s_i))^T, \overline{\mathbf{c}_i}, \alpha q, \frac{\alpha_1}{2\sqrt{t'}\alpha}) + \sum_{i=t+1}^{t'} \mathsf{ReRand}(s_i I_{d+k+1}, \overline{\mathbf{c}_i}, \alpha q, \frac{\alpha_1}{2\sqrt{t'}\alpha})$. And set c_ι to be the polynomial whose coefficient vector is $\mathbf{c}_\iota$. Finally, it returns $(c_1, (c_i)_{i=1}^{2t'})$ to $\mathcal{A}$. **Guess** At last, $\mathcal{A}$ outputs its guess $\widetilde{\mathsf{coin}}$ if the abort condition has not been satisfied. Then $\mathcal{B}$ sets $\mathsf{coin}' = \widetilde{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin} = \mathsf{coin}'$ and 0 otherwise. **Analysis** We can see that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in Game 6 if $\mathcal{O} = \mathcal{O}_s$ and Game 7 if $\mathcal{O} = \mathcal{O}_\$$. Note that both games only differ in the generation of the challenge ciphertext in the case of $\mathsf{coin} = 0$. Furthermore, we can see that the generation of mpk, the queries, and the challenge phase are perfect for the case of both $\mathsf{coin} = 1$. Therefore, $|\Pr[X_6] - \Pr[X_7]| = Adv_{\mathcal{B}}^{\mathrm{dMPLWE}_{q,n+2d+k,\mathbf{d},D_{\alpha \cdot q}}} = \mathsf{negl}(n)$ as desired. □

6 Conclusion

In this paper, we propose a new compact, adaptively secure IBE from MPLWE. Our scheme inherits the typical advantage of MPLWE. Namely, it has comparable efficiency to its RLWE counterparts while offering stronger security. Its master public key size is smaller than the state-to-art MPLWE-based construction.

7 Acknowledgement

This work is partially supported by the Research Grant Council of Hong Kong (Project No.: 17201421, C1029-22GF, R1012-21), Innovation and Technology Fund of the Innovation and Technology Commission of Hong Kong (Project No.: ITS/176/23FP), The Hong Kong Polytechnic University (Project No.: P0046340, P0048350, P0046317).

References

1. Abla, P.: Identity-based encryption from LWE with more compact master public key. In: Oswald, E. (ed.) Topics in Cryptology - CT-RSA 2024 - Cryptographers' Track at the RSA Conference 2024, San Francisco, CA, USA, May 6-9, 2024, Proceedings. Lecture Notes in Computer Science, vol. 14643,

pp. 319–353. Springer (2024). https://doi.org/10.1007/978-3-031-58868-6_13, https://doi.org/10.1007/978-3-031-58868-6_13
2. Abla, P., Liu, F., Wang, H., Wang, Z.: Ring-based identity based encryption - asymptotically shorter MPK and tighter security. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13044, pp. 157–187. Springer (2021). https://doi.org/10.1007/978-3-030-90456-2_6, https://doi.org/10.1007/978-3-030-90456-2_6
3. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6110, pp. 553–572. Springer (2010). https://doi.org/10.1007/978-3-642-13190-5_28, https://doi.org/10.1007/978-3-642-13190-5_28
4. Bellare, M., Ristenpart, T.: Simulation without the artificial abort: Simplified proof and improved concrete security for waters' IBE scheme. In: Joux, A. (ed.) Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5479, pp. 407–424. Springer (2009). https://doi.org/10.1007/978-3-642-01001-9_24, https://doi.org/10.1007/978-3-642-01001-9_24
5. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings. Lecture Notes in Computer Science, vol. 2139, pp. 213–229. Springer (2001). https://doi.org/10.1007/3-540-44647-8_13, https://doi.org/10.1007/3-540-44647-8_13
6. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6110, pp. 523–552. Springer (2010). https://doi.org/10.1007/978-3-642-13190-5_27, https://doi.org/10.1007/978-3-642-13190-5_27
7. Cocks, C.C.: An identity based encryption scheme based on quadratic residues. In: Honary, B. (ed.) Cryptography and Coding, 8th IMA International Conference, Cirencester, UK, December 17-19, 2001, Proceedings. Lecture Notes in Computer Science, vol. 2260, pp. 360–363. Springer (2001). https://doi.org/10.1007/3-540-45325-3_32, https://doi.org/10.1007/3-540-45325-3_32
8. Fan, J., Lu, X., Au, M.H.: Adaptively secure identity-based encryption from middle-product learning with errors. In: Simpson, L., Baee, M.A.R. (eds.) Information Security and Privacy - 28th Australasian Conference, ACISP 2023, Brisbane, QLD, Australia, July 5-7, 2023, Proceedings. Lecture Notes in Computer Science, vol. 13915, pp. 320–340. Springer (2023). https://doi.org/10.1007/978-3-031-35486-1_15, https://doi.org/10.1007/978-3-031-35486-1_15
9. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008.

pp. 197–206. ACM (2008). https://doi.org/10.1145/1374376.1374407, https://doi.org/10.1145/1374376.1374407
10. Katsumata, S., Yamada, S.: Partitioning via non-linear polynomial functions: More compact ibes from ideal lattices and bilinear maps. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10032, pp. 682–712 (2016). https://doi.org/10.1007/978-3-662-53890-6_23, https://doi.org/10.1007/978-3-662-53890-6_23
11. Lombardi, A., Vaikuntanathan, V., Vuong, T.D.: Lattice trapdoors and IBE from middle-product LWE. In: Hofheinz, D., Rosen, A. (eds.) Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11891, pp. 24–54. Springer (2019). https://doi.org/10.1007/978-3-030-36030-6_2, https://doi.org/10.1007/978-3-030-36030-6_2
12. Lyubashevsky, V.: Digital signatures based on the hardness of ideal lattice problems in all rings. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10032, pp. 196–214 (2016). https://doi.org/10.1007/978-3-662-53890-6_7, https://doi.org/10.1007/978-3-662-53890-6_7
13. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7237, pp. 700–718. Springer (2012). https://doi.org/10.1007/978-3-642-29011-4_41, https://doi.org/10.1007/978-3-642-29011-4_41
14. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: 45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings. pp. 372–381. IEEE Computer Society (2004). https://doi.org/10.1109/FOCS.2004.72, https://doi.org/10.1109/FOCS.2004.72
15. Rosca, M., Sakzad, A., Stehlé, D., Steinfeld, R.: Middle-product learning with errors. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III. Lecture Notes in Computer Science, vol. 10403, pp. 283–297. Springer (2017). https://doi.org/10.1007/978-3-319-63697-9_10, https://doi.org/10.1007/978-3-319-63697-9_10
16. Rosca, M., Stehlé, D., Wallet, A.: On the ring-lwe and polynomial-lwe problems. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I. Lecture Notes in Computer Science, vol. 10820, pp. 146–173. Springer (2018). https://doi.org/10.1007/978-3-319-78381-9_6, https://doi.org/10.1007/978-3-319-78381-9_6
17. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings. Lecture Notes in Com-

puter Science, vol. 196, pp. 47–53. Springer (1984). https://doi.org/10.1007/3-540-39568-7_5, https://doi.org/10.1007/3-540-39568-7_5
18. Stehlé, D., Steinfeld, R., Tanaka, K., Xagawa, K.: Efficient public key encryption based on ideal lattices. In: Matsui, M. (ed.) Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5912, pp. 617–635. Springer (2009). https://doi.org/10.1007/978-3-642-10366-7_36, https://doi.org/10.1007/978-3-642-10366-7_36
19. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3494, pp. 114–127. Springer (2005). https://doi.org/10.1007/11426639_7, https://doi.org/10.1007/11426639_7
20. Yamada, S.: Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In: Fischlin, M., Coron, J. (eds.) Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9666, pp. 32–62. Springer (2016). https://doi.org/10.1007/978-3-662-49896-5_2, https://doi.org/10.1007/978-3-662-49896-5_2
21. Yamada, S.: Asymptotically compact adaptively secure lattice ibes and verifiable random functions via generalized partitioning techniques. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III. Lecture Notes in Computer Science, vol. 10403, pp. 161–193. Springer (2017). https://doi.org/10.1007/978-3-319-63697-9_6, https://doi.org/10.1007/978-3-319-63697-9_6
22. Zhang, J., Chen, Y., Zhang, Z.: Programmable hash functions from lattices: Short signatures and ibes with small key sizes. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III. Lecture Notes in Computer Science, vol. 9816, pp. 303–332. Springer (2016). https://doi.org/10.1007/978-3-662-53015-3_11, https://doi.org/10.1007/978-3-662-53015-3_11