Research Article

LWR-Based Fully Homomorphic Encryption, Revisited

Fucai Luo ^{1,2,3}, Fuqun Wang,^{4,5} Kunpeng Wang,^{1,2,3} Jie Li,^{1,2,3} and Kefei Chen^{4,5}

¹School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

²State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

³Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, Beijing, China

⁴College of Science, Hangzhou Normal University, Hangzhou, China

⁵Westone Cryptologic Research Center, Beijing, China

Correspondence should be addressed to Fucai Luo; luofucai@iie.ac.cn

Received 29 August 2017; Revised 3 January 2018; Accepted 18 January 2018; Published 23 April 2018

Academic Editor: Amir Anees

Copyright © 2018 Fucai Luo et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Very recently, Costache and Smart proposed a fully homomorphic encryption (FHE) scheme based on the Learning with Rounding (LWR) problem, which removes the noise (typically, Gaussian noise) sampling needed in the previous lattices-based FHEs. But their scheme did not work, since the noise of homomorphic multiplication is complicated and large, which leads to failure of decryption. More specifically, they chose LWR instances as a public key and the private key therein as a secret key and then used the tensor product to implement homomorphic multiplication, which resulted in a tangly modulus problem. Recall that there are two moduli in the LWR instances, and then the moduli will tangle together due to the tensor product. Inspired by their work, we built the first workable LWR-based FHE scheme eliminating the tangly modulus problem by cleverly adopting the celebrated *approximate eigenvector* method proposed by Gentry et al. at Crypto 2013. Roughly speaking, we use a specific matrix multiplication to perform the homomorphic multiplication, hence no tangly modulus problem. Furthermore, we also extend the LWR-based FHE scheme to the multikey setting using the tricks used to construct LWE-based multikey FHE by Mukherjee and Wichs at Eurocrypt 2016. Our LWR-based multikey FHE construction provides an alternative to the existing multikey FHEs and can also be applied to multiparty computation with higher efficiency.

1. Introduction

Fully homomorphic encryption (FHE) is a cryptographic primitive that allows performing arbitrarily complex and efficiently computable evaluations over encrypted data without decrypting them. But the problem of how to construct a FHE scheme had been bothering cryptologists since it was initially introduced by Rivest et al. [1]. Until 2009, this conundrum was compromised due to Gentry's first plausible candidate FHE construction based on ideal lattices [2]. Since then, a series of works [3–8] have been presented with much progress mainly in security assumptions and efficiencies.

Gentry's seminal work [2] showed for the first time that FHE can be based on cryptographic assumptions and put forward a remarkable “bootstrapping” theorem to achieve full homomorphism (which needs a “circular security” assumption). However, his scheme relies on relatively stronger

cryptographic assumptions on ideal lattices (ideal lattices are a special breed that we know relatively little about) and can only evaluate “low degree” polynomials homomorphically without “bootstrapping.”

The LWE-based FHEs [5, 9–13] enjoy higher efficiency and stronger security compared to the previous schemes [2–4, 7] (following a similar framework to Gentry's work), due to the simple algebraic structure of the well-studied LWE [14] and classical (quantum) reduction from some apparently intractable lattice problems (e.g., GapSVP) to LWE [14, 15]. The first LWE-based FHE was proposed by Brakerski and Vaikuntanathan [5] (henceforth, BV11b), who used a novel *relinearization* technique to construct a “somewhat homomorphic” encryption scheme based on LWE problem and introduced a novel *dimension-modulus reduction* technique, without resorting to the “squashing paradigm” used in the previous schemes [2–4, 7]. The subsequent improved

works mainly refer to Brakerski et al.'s [9] BGV12 and Brakerski's [10] Bra12. In BGV12, Brakerski, Gentry, and Vaikuntanathan used the *dimension reduction* and *modulus reduction* (which are originated from BV11b) iteratively and gradually, to construct a "leveled" FHE scheme (capable of evaluating arbitrary polynomial-depth circuits). Bra12 used the *dimension reduction* without the *modulus reduction*, to build a better leveled FHE scheme, which is superior to the previous best known in simplicity, noise management, and security. This is mainly because of their noise which only grows linearly ($B \rightarrow B \cdot \text{poly}(n)$) with every homomorphic multiplication, while in all previous works, the noise grows quadratically ($B \rightarrow B^2 \cdot \text{poly}(n)$) without *modulus reduction*.

In Crypto 2013, Gentry et al. [16] (henceforth, GSW13) presented a LWE-based FHE scheme of GSW style (which was improved by Alperin-Sheriff and Peikert [17], henceforth, AP14), using two novel techniques of so-called *approximate eigenvector* and *flatten*, where the ciphertext is a matrix rather than vector. For the most part, its homomorphic addition and multiplication are just matrix addition and multiplication, which avoids the *key switching*, *modulus switching*, and the "evaluation" key used in previous schemes (e.g., BV11b, BGV12). It is important to note that, besides the fact that the scheme does not need an "evaluation" key, it has an interesting property of asymmetric noise growth because of its specific GSW style. Based on GSW13, a sequence of schemes was proposed, including bootstrapping schemes [11, 17], multikey schemes [6, 12, 13], and some other related schemes [18, 19] (these schemes mainly leverage the homomorphic operations of GSW13).

Motivations. The above-mentioned LWE-based FHEs and related schemes suffered the complex and time-consuming Gaussian or sub-Gaussian noise sampling, due to the fact that the corresponding LWE problem needs a noise (error) vector sampled from a distribution, typically (discrete) Gaussian or sub-Gaussian distribution [5, 9, 10, 12, 16, 17]. In particular, some schemes (e.g., [6, 18]) based on the LWE problem have to sample Gaussian noise in the encryption process, which seriously weakens the schemes' efficiencies. Moreover, it has been recently shown (e.g., [20, 21]) that the Gaussian sampling will create lots of potential side-channel vulnerabilities that result in complete leakage of the secret key. Although it is possible to design good implementations which protect against side-channel attacks, these implementations are often very complex.

As a matter of course, this raises a question: *can we cast away the Gaussian noise sampling in building a FHE scheme while maintaining the same (almost) security level as those based on LWE problem?* Indeed, this is valuable theoretically and practically and even pedagogically.

Very recently, Costache and Smart [22] showed a FHE scheme based on the ring-LWR problem (or RLWR, a variant of Learning with Rounding (LWR) problem). Their scheme removes the Gaussian noise needed in the previous LWE-based FHEs and results in slightly smaller ciphertexts. Roughly speaking, they focused their attention on BGV12 and used the techniques of *relinearization* and *modulus switching*

TABLE 1: Comparison with LWE-based FHEs: GSW13 and AP14.

FHE	Modulus q	Security loss	Gaussian noise
GSW13 and AP14	$(O(n \log q))^{L+O(1)}$	$O(1)$	Yes
Our LWR-based FHE	$(O(n \log q))^{L+O(1)}$	$O(qn \log q)$	No

(1) We consider a leveled FHE scheme where the depth of circuits is polynomial L . The homomorphic evaluation capability, efficiency, and security of the FHE scheme mainly depend on modulus q , under the same dimension n . Note that the modulus q of AP14 is relatively smaller than that of GSW13, for the sub-Gaussian in AP14 results in a tighter noise growth than the Gaussian. Here, we ignore this little difference. (2) Here, the security loss is caused by the reduction between the security of the FHE scheme and the LWE problem. Since the security of our FHE scheme is directly based on the LWR problem, the security loss involves the reduction loss incurred by the reduction between LWE and LWR (see Theorem 7 in Section 2).

to build a RLWR-based FHE scheme. However, the LWR (the definition of LWR will be presented in Section 2.2), mainly leveraged by a scaled rounding function [23] including two different moduli q and p , makes the tensor product $\otimes$ used by them to implement homomorphic multiplication intractable. In more detail, they chose RLWR instances as a public key and the private key therein as a secret key, and the ciphertext was computed as a vector $\mathbf{c} = (v, w) \in R_q \times R_p$ decrypting to message $\mu \in R_t$, where R_q, R_p, R_t are quotient rings. Then, they used the tensor product $\otimes$ to implement homomorphic multiplication, which results in a product of two different elements belonging to different rings (i.e., two moduli are tangled together). This makes their analysis of the multiplication noise complicated and obscure. In fact, this "tangly modulus" problem brings a large multiplication noise to their decryption equation (in terms of ciphertext after homomorphic multiplication) and thus leads to failure of the decryption. Therefore, how to construct a FHE scheme based on LWR problem, we think, is still an open problem, while we focus our attention on this problem.

Our Results. In this paper, we propose a workable LWR-based FHE scheme eliminating the tangly modulus problem by cleverly adopting the celebrated *approximate eigenvector* method in GSW13. Roughly speaking, we use a specific matrix multiplication to perform the homomorphic multiplication, which avoids the tangly modulus problem, where the specific matrix multiplication involves a variant of *gadget* matrix (which will be described in Section 2.3). The efficiency of our scheme is almost comparable to that of GSW13 and AP14 without counting the cost of Gaussian noise sampling, for the size of modulus q is almost the same as theirs which can be seen from Table 1 (our modulus q is larger; see Section 4.2). Indeed, it is mainly our larger security loss (up to a polynomial factor) that results in the larger modulus q (up to a polynomial factor). Our scheme can be seen as an alternative to the GSW13 and AP14.

Furthermore, we also extend the LWR-based FHE scheme to multikey setting using the tricks used to construct LWE-based multikey FHE by Mukherjee and Wichs [12] at Eurocrypt 2016. Again, we leverage the specific GSW

style to avoid the tangly modulus problem. Interestingly, it seems that the method of [12] has been tailored to be employed in constructing our LWR-based multikey FHE scheme, since it helps to avoid the tangly modulus problem when expanding the valid ciphertexts needed in multikey setting (see Section 5). Compared to Mukherjee and Wichs's scheme, our scheme can also be applied to multiparty computation and is more efficient; again, this is largely for the reason that there is no Gaussian noise sampling. This can be also verified by Table 1 and by the fact that the LWE-based multikey FHE [12] is an extension of API4.

What is more, our LWR-based FHE and its extended multikey FHE support bootstrapping procedures; this is due to the fact that the encryption and decryption processes in our schemes are very similar to those in API4, except that there are two different moduli in our scheme (which will not tangle together because of our specific structure). We believe that it is straightforward to convert our LWR-based scheme into a RLWR-based one.

Our Techniques. In our LWR-based FHE, we choose m pairs of LWR instances as a public key and the private key therein as a secret key. The public key is assembled as a matrix

$$\mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{b} \end{bmatrix} \in \mathbb{Z}_q^{n \times m} [\times] \mathbb{Z}_p^m \quad (1)$$

such that $\mathbf{s} \cdot \mathbf{A} = \mathbf{e} \bmod p$,

where $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{b} \in \mathbb{Z}_p^m$, and the secret key is $\mathbf{s} = (-p/q \cdot \mathbf{s}', 1)$, where $\mathbf{s}' \in \{0, 1\}^n$. Since there are two different moduli in a matrix, we use the symbol $[\times]$ (see Section 2) to differentiate from the conventional symbol $\times$.

For a message $\mu \in \{0, 1\}$, the ciphertext is computed as

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{R} + \mu \mathbf{G} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N, \quad (2)$$

where $\mathbf{G} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$ is a variant of **gadget matrix** (see Section 2.3). To perform homomorphic multiplication (addition is very natural), do the following: given two ciphertexts $\mathbf{C}_1, \mathbf{C}_2$ decrypting to message $\mu_1, \mu_2 \in \{0, 1\}$, we have

$$\mathbf{C}_{\text{mult}} \triangleq \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N, \quad (3)$$

since matrix $\mathbf{G}^{-1}(\mathbf{C}_2) \in \{0, 1\}^{N \times N}$ does not affect the structure of matrix $\mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)$ (see **Operations** in Section 2). Hence, we assert that the structures of ciphertexts remain unchanged after L levels of multiplication, so they can be properly decrypted. Indeed, our homomorphic multiplication will not cause the tangly modulus problem existent in Costache and Smart's solution [22], because these two different modular operations (in terms of moduli q, p) can be partitioned by our matrix operation but cannot be by the tensor product used by them. We defer the details to Section 4.

For the secret key, $\mathbf{s} = (-p/q \cdot \mathbf{s}', 1)$, where p, q are public and $\mathbf{s}'$ is privately and uniformly chosen from $\{0, 1\}^n$ instead of $\mathbb{Z}_q$ for initial noise and efficiency. This is reasonable and secure relying on Theorem 7 in Section 2.2 and the hardness

of LWE problem with binary secret (which was proved at least as hard as the original LWE problem by Brakerski et al. [24]).

Organization. In Section 2, we give some preliminaries including the LWE problem, LWR problem, and a variant of gadget matrix. We describe a basic LWR-based encryption scheme and give its full security proof in Section 3. Section 4 goes into the core of our LWR-based FHE scheme. In Section 5, we extend our main construction to the LWR-based multikey FHE. Finally, we conclude the paper in Section 6.

2. Preliminaries

As a preliminary matter, we give some explanations for some notations and operations throughout the paper.

Notations. For an integer q , we define the set $\mathbb{Z}_q \triangleq (-q/2, q/2] \cap \mathbb{Z}$, and all logarithms on q are to base 2. All arithmetics are performed over $\mathbb{Z}$ or $\mathbb{Q}$ when division is used, and for ease of use, we let $[n] \triangleq \{1, \dots, n\}$. We denote vectors in bold lowercase (e.g., $\mathbf{x}$) and matrices in bold uppercase (e.g., $\mathbf{A}$). Let $\mathbf{A}^t$ (resp., $\mathbf{x}^t$) be the transpose of $\mathbf{A}$ (resp., $\mathbf{x}$). For any $x \in \mathbb{Q}$, we denote by $\lfloor x \rfloor$, $\lceil x \rceil$, and $\lceil x \rceil$ the rounding of x down, up, or to the nearest integer; these notations also apply to vector and matrix. We say that a function $\text{negl}(n)$ is negligible if there are not any polynomial fractions smaller than the $\text{negl}(n)$ for sufficiently large n . All vectors are treated as rows in the paper.

Probability. We let $x \leftarrow_r \mathcal{D}$ denote that x is sampled uniformly at random from a distribution $\mathcal{D}$ and $x \leftarrow_r \mathcal{S}$ (e.g., $\mathbb{Z}_q$) denote that x is uniform over a set $\mathcal{S}$.

Operations. For an n -dimensional vector $\mathbf{x} = \{x_1, \dots, x_n\}$, we denote its norm by $\|\mathbf{x}\| \triangleq \max\{|x_i|\}_{i \in [n]}$, where $|x_i|$ refers to its magnitude. The multiplication between two vectors $\mathbf{x}, \mathbf{y}$ over $\mathbb{Z}_q$ is denoted by $\langle \mathbf{x}, \mathbf{y} \rangle_q$ (i.e., $\langle \mathbf{x}, \mathbf{y} \rangle \bmod q$). What we want to particularly explain beforehand is the multiplication between two matrices.

Given two different moduli q, p , and any integer n, m , for $(n+1) \times m$ matrix $\mathbf{A}$, by abuse of notation, we let $\mathbf{A} \in \mathbb{Z}_q^{n \times m} [\times] \mathbb{Z}_p^m$ indicate that all row vectors of $\mathbf{A}$ are over $\mathbb{Z}_q$ but the last one (i.e., $(n+1)$ -th) is over $\mathbb{Z}_p^m$, and we state that we should be careful about, for example, multiplying a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m} [\times] \mathbb{Z}_p^m$ by a matrix $\mathbf{R} \in \{0, 1\}^{m \times m}$ (or a vector $\mathbf{r} \in \{0, 1\}^m$). More precisely, we should multiply every row vector of $\mathbf{A}$ by the matrix $\mathbf{R} \in \{0, 1\}^{m \times m}$, that is, multiply each row vector of $\mathbf{A}$ by each column vector of $\mathbf{R} \in \{0, 1\}^{m \times m}$, and then take a modular operation (take the product modulo a modulus). Note that the last row vector of $\mathbf{A}$ is over $\mathbb{Z}_p^m$, and thus the multiplications between this row vector and the matrix $\mathbf{R} \in \{0, 1\}^{m \times m}$ are evaluated over $\mathbb{Z}_p^m$ (take the products modulo the modulus p), while all the other multiplications are evaluated over $\mathbb{Z}_q^m$ (take the products modulo the modulus q). (These two modular operations can be partitioned due to the matrix structure which, roughly speaking, avoids the tangly

modulus problem existent in [22].) This principle also applies to addition between two matrices over $\mathbb{Z}_q^{n \times m}[\times] \mathbb{Z}_p^m$. Indeed, this is an important difference between LWE and LWR when performing addition and multiplication operations on matrices.

In our security proof, we will use the following variant of the standard leftover hash lemma [25].

Lemma 1 (see [5]). *Let λ, n, q, p , and $m \geq n \log q + \log p + 2\lambda$ be integers, and let $\mathbf{y} \leftarrow_r \mathbb{Z}_q^n \times \mathbb{Z}_p$. For matrix $\mathbf{A} \leftarrow_r \mathbb{Z}_q^{n \times m}[\times] \mathbb{Z}_p^m$, let $\mathbf{r} \leftarrow_r \{0, 1\}^m$; one has*

$$\Delta((\mathbf{A}, \mathbf{A} \cdot \mathbf{r}), (\mathbf{A}, \mathbf{y})) \leq \frac{1}{2} \sqrt{\frac{q^n \cdot p}{2^m}} < 2^{-\lambda}, \quad (4)$$

where $\Delta(X, Y)$ denotes the statistical distance between two distributions X, Y .

2.1. Learning with Errors (LWE). The well-known LWE problem has been enjoying fame for its applicability in constructions of lattice-based schemes conjectured to be secure in quantum setting, ever since Regev introduced it and showed a quantum reduction [14] from some worst-case hardness of the standard lattice problems to LWE problem (followed by classical reductions [15, 24]). It is defined as follows.

Definition 2 (see [9]). For positive integers $n, q = q(n), m \geq O(n \log q)$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$, and an error distribution χ over $\mathbb{Z}$ (typically, a (discrete) Gaussian distribution with standard deviation αq for $\alpha = o(1)$). Let $\mathcal{A}_{n,q,m,\chi}$ be the distribution over $\mathbb{Z}_q^{n+1}$ obtained by choosing m independent samples $\mathbf{a} \leftarrow_r \mathbb{Z}_q^n$ and an error term $e \leftarrow_r \chi$ and outputting m pairs $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle_q + e)$. Then, the (average-case) decision problem of the LWE, denoted by $\text{DLWE}_{n,q,m,\chi}$, is to distinguish, given arbitrarily many independent samples, the uniform distribution over $\mathbb{Z}_q^{n+1}$ from $\mathcal{A}_{n,q,m,\chi}$, for a fixed $\mathbf{s} \in \mathbb{Z}_q^n$. The search problem of $\text{LWE}_{n,q,m,\chi}$ is aim to find secret $\mathbf{s}$ given m independent samples from $\text{LWE}_{n,q,m,\chi}(\mathbf{s})$ (for $\mathbf{s} \in \mathbb{Z}_q^n$). The $\text{LWE}_{n,q,m,\chi}$ assumption is that the $\text{LWE}_{n,q,m,\chi}$ problem is infeasible.

Definition 3 (B -bounded distributions [16]). A distribution ensemble $\{\chi_n\}_{n \in \mathbb{N}}$, supported over the integers, is called B -bounded if

$$\Pr_{e \leftarrow_r \chi_n} [|e| > B] = \text{negl}(n). \quad (5)$$

2.2. Learning with Rounding (LWR). Learning with Rounding (LWR) problem, which was used to construct lossy trapdoor functions, reusable computational extractors, and deterministic encryption, was firstly proposed by Banerjee et al. [23] for improving the efficiency of pseudorandom generator (PRG) based on the LWE problem. Indeed, the LWR problem can be seen as a deterministic alternative to LWE problem, except that the noise in LWR is deterministic which derandomizes the Gaussian noise in LWE, and the noise in LWR resulted from the scale rounding function being

smaller than that in LWE. Specifically, the noise in LWE is B -bounded ($B > 2\sqrt{n}$ for security [14]), whereas it has magnitude of less than $1/2$ in LWR. We recall the scaled rounding function $\lceil \cdot \rceil_p$ [23] which is defined as follows: for $p < q$,

$$\begin{aligned} \lceil \cdot \rceil_p : \mathbb{Z}_q &\longrightarrow \mathbb{Z}_p \\ a &\longmapsto \left\lceil \frac{p}{q} \cdot a \right\rceil. \end{aligned} \quad (6)$$

Similar to the LWE problem, we get the following analogous definition for the LWR problem.

Definition 4 (see [23]). For integers $n, q > p$, and m , sample uniformly at random an n -dimensional vector $\mathbf{s}$ from $\mathbb{Z}_q^n$, and then the LWR distribution is defined as $\text{LWR}_{n,q,p}(\mathbf{s}) \triangleq \{(\mathbf{a}, \lceil \langle \mathbf{a}, \mathbf{s} \rangle_q \rceil_p) \in \mathbb{Z}_q^n \times \mathbb{Z}_p \mid \mathbf{a} \leftarrow_r \mathbb{Z}_q^n\}$, where the pair $(\mathbf{a}, \lceil \langle \mathbf{a}, \mathbf{s} \rangle_q \rceil_p)$ is LWR sample (instance), and let $\text{LWR}_{n,m,q,p}(\mathbf{s})$ be the distribution comprised of m independent samples from $\text{LWR}_{n,q,p}(\mathbf{s})$. Then, the search $\text{LWR}_{n,m,q,p}$ problem is defined as finding secret $\mathbf{s}$ given m independent samples from $\text{LWR}_{n,q,p}(\mathbf{s})$, while the decision $\text{DLWR}_{n,m,q,p}$ problem is to distinguish m independent samples (with nonnegligible advantage) chosen from the distribution $\text{LWR}_{n,m,q,p}(\mathbf{s})$, for a fixed $\mathbf{s} \leftarrow_r \mathbb{Z}_q^n$, from m samples chosen from uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_p$. The $\text{LWR}_{n,m,q,p}$ assumption is that the $\text{DLWR}_{n,m,q,p}$ problem is infeasible.

As to the hardness of the LWR problem, Banerjee et al. [23] presented an efficient reduction from the LWE problem to the LWR problem for modulus q of superpolynomial size, followed by Alwen et al. [26], who gave a reduction that allowed for a polynomial modulus q , but restricted the number of samples and failed to apply to all values of the modulus q . In 2016, Bogdanov et al. [27] generalized the theorem of [26] by eliminating the theoretic restriction on the modulus q , but the number of samples was required to be less than $O(q/Bp)$ (weaker than that of [26]), while Alperin-Sheriff and Apon [28] showed a dimension-preserving reduction from LWE to LWR with a polynomial-sized modulus, which immediately implies improvements in parameters (i.e., security and efficiency) for all known applications of polymodulus LWR.

Note that Brakerski et al. [24] proved that the bin-LWE (implies that the secret key is uniformly chosen from $\{0, 1\}^n$) is at least as hard as the original LWE problem (up to logarithmic loss). Hence, we can uniformly choose the secret $\mathbf{s}$ in LWR from $\{0, 1\}^n$ under the hardness of bin-LWE and the following theorem (Theorem 5). Next, we recall the main theorem in [27] (it is sufficient for our schemes, though the number of samples was required to be less than $O(q/Bp)$). Then, by combining this theorem with Lemma 6 (it is a simple fact; here we present it as a lemma), we get our crucial Theorem 7 on which the proposed schemes' security is based. Note that Theorem 7 concerns the search problem of bin-LWE, which is harder than the decision problem of bin-LWE.

Theorem 5 (see [27]). For every $\varepsilon > 0$, $n, m, q \geq 2mpB$, and if there is an algorithm $\mathcal{A}$ such that

$$\left| \Pr_{\mathbf{A}, \mathbf{s}} [\mathcal{A}(\mathbf{A}, \lceil \mathbf{A}\mathbf{s} \rceil_p) = 1] - \Pr_{\mathbf{A}, \mathbf{u}} [\mathcal{A}(\mathbf{A}, \lceil \mathbf{u} \rceil_p) = 1] \right| \geq \varepsilon, \quad (7)$$

where $\mathbf{A} \leftarrow_r \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow_r \{0, 1\}^n$, and $\mathbf{u} \leftarrow_r \mathbb{Z}_q^m$, then there exists an efficient algorithm $\mathcal{B}$ that runs in time polynomial in n, m , the number of divisors of q , and the running time of $\mathcal{A}$ such that

$$\Pr_{\mathbf{A}, \mathbf{s}} [\mathcal{B}(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e}) = \mathbf{s}] \geq \left(\frac{\varepsilon}{4qm} - \frac{2^n}{p^m} \right)^2 \cdot \frac{1}{(1 + 2Bp/q)^m} \quad (8)$$

for noise distribution $\mathbf{e}$ that is B -bounded and balanced in each coordinate.

Lemma 6. If $p \mid q$, then the distribution $\lceil \mathbf{u} \rceil_p$ is uniform over $\mathbb{Z}_p^m$ for $\mathbf{u} \leftarrow_r \mathbb{Z}_q^m$ (i.e., the distribution $\lceil \mathbf{u} \rceil_p$ is equivalent to $\mathbb{Z}_p^m$).

Theorem 7. For every $\varepsilon > 0$, $n, m, q \geq 2mpB$, $p \mid q$, and if there is an algorithm $\mathcal{A}$ such that

$$\left| \Pr_{\mathbf{A}, \mathbf{s}} [\mathcal{A}(\mathbf{A}, \lceil \mathbf{A}\mathbf{s} \rceil_p) = 1] - \Pr_{\mathbf{A}, \mathbf{v}} [\mathcal{A}(\mathbf{A}, \mathbf{v}) = 1] \right| \geq \varepsilon, \quad (9)$$

where $\mathbf{A} \leftarrow_r \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow_r \{0, 1\}^n$, and $\mathbf{v} \leftarrow_r \mathbb{Z}_p^m$, then there exists an efficient algorithm $\mathcal{B}$ that runs in time polynomial in n, m , the number of divisors of q , and the running time of $\mathcal{A}$ such that

$$\Pr_{\mathbf{A}, \mathbf{s}} [\mathcal{B}(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e}) = \mathbf{s}] \geq \left(\frac{\varepsilon}{4qm} - \frac{2^n}{p^m} \right)^2 \cdot \frac{1}{(1 + 2Bp/q)^m} \quad (10)$$

for noise distribution $\mathbf{e}$ that is B -bounded and balanced in each coordinate.

Proof of Sketch. Note that there is no restriction on the condition $p \mid q$ in Theorem 5. In other words, Theorem 5 holds even under the additional condition $p \mid q$. Therefore, Theorem 7 is obvious by combining Lemma 6 which implies that the vector $\lceil \mathbf{u} \rceil_p$ is equivalent to $\mathbf{v}$. $\square$

We remark that the term $\Pr_{\mathbf{A}, \mathbf{s}} [\mathcal{A}(\mathbf{A}, \lceil \mathbf{A}\mathbf{s} \rceil_p) = 1] - \Pr_{\mathbf{A}, \mathbf{v}} [\mathcal{A}(\mathbf{A}, \mathbf{v}) = 1]$ in Theorem 7 can be interpreted as the decision DLWR $_{n,m,q,p}$ problem, for the fixed $\mathbf{s} \leftarrow_r \{0, 1\}^n$.

2.3. Variant of Gadget Matrix. The gadget matrix, which was proposed by [29], was used to build a LWE-based FHE scheme of GSW style by AP14, which extended the “flatten” skill in GSW13. Here, we construct a variant of gadget matrix which is specific to our LWR-based FHE scheme.

Since there are two moduli q, p in the LWR problem, a variant of gadget matrix $\mathbf{G}$ with invariant function is as follows:

$$\mathbf{G} := \begin{bmatrix} \mathbf{g} & & \\ & \ddots & \\ & & \mathbf{g} \\ & & & \mathbf{g}_{n+1} \end{bmatrix} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N, \quad (11)$$

where $\mathbf{g} := (1, 2, \dots, 2^{\lceil \log q \rceil - 1}) \in \mathbb{Z}_q^{\lceil \log q \rceil}$, $\mathbf{g}_{n+1} := (1, 2, \dots, 2^{\lceil \log p \rceil - 1}) \in \mathbb{Z}_p^{\lceil \log p \rceil}$, and $N = n \lceil \log q \rceil + \lceil \log p \rceil$. We also define the deterministic inversion function $\mathbf{G}^{-1} : \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N \rightarrow \{0, 1\}^{N \times N}$, which is equal to bit decomposition that decomposes x into its bit representation over $\mathbb{Z}_q$ or $\mathbb{Z}_p$ and has the following property: for any matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$, it holds that $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A}$. Note that $\mathbf{G}^{-1}$ is an expensively randomized inversion function in AP14, although it leads to a tighter noise analysis for its homomorphic operations.

3. Building Block

As a warmup, in this section, we present a basic encryption scheme construction based on LWR which is adapted from the ring-LWR-based FHE scheme [22]. Then, based on the basic encryption scheme, we give our LWR-based FHE scheme in the next section. We remark that the secret key is uniformly chosen from $\{0, 1\}^n$ to keep the noise growing slower and speed up encryption and decryption processes, under the same (almost) security level as GSW13 and AP14. This can be guaranteed by Theorem 7 in Section 2.2 and the hardness of bin-LWE problem.

3.1. The Basic LWR-Based Encryption Scheme. The basic encryption scheme based on LWR is constructed as follows.

- (i) *Setup*(1^λ): choose parameters $n = n(\lambda)$, $\ell = \ell(\lambda)$, moduli $q = q(\lambda)$, and $p = p(\lambda)$ satisfying $p \mid q$ (this condition is needed for the security proof). Let $\text{params} = (n, \ell, q, p)$.
- (ii) *KeyGen*(params): choose uniformly at random a secret key $\mathbf{s}' \leftarrow_r \{0, 1\}^n$. Choose uniformly at random ℓ vectors $\mathbf{a}_i \leftarrow_r \mathbb{Z}_q^n$, and then compute

$$b_i = \left\lceil \langle \mathbf{a}_i, \mathbf{s}' \rangle_q \right\rceil_p \in \mathbb{Z}_p \quad (12)$$

for $i \in [\ell]$, which results in ℓ pairs of LWR samples $\{(\mathbf{a}_1, b_1), \dots, (\mathbf{a}_\ell, b_\ell) \in \mathbb{Z}_q^n \times \mathbb{Z}_p\}$. Assemble a matrix $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times \ell}$ by the ℓ column vectors $\mathbf{a}_i^t$ and set $\mathbf{b} = (b_1, \dots, b_\ell)$, and then let $\mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{b} \end{bmatrix} \in \mathbb{Z}_q^{n \times \ell} [\times] \mathbb{Z}_p^\ell$. The final public key is $\text{pk} := \mathbf{A}$, and the secret key is $\text{sk} : \mathbf{s} = (-p/q \cdot \mathbf{s}', 1)$. For later use, we write $e_i = b_i - (p/q) \langle \mathbf{a}_i, \mathbf{s}' \rangle_q \in [-1/2, 1/2]$ for $i \in [\ell]$. Then, according to (15), it holds that $\mathbf{s} \cdot \mathbf{A} = \mathbf{e} \bmod p$, where the noise $\mathbf{e} = (e_1, \dots, e_\ell) \in [-1/2, 1/2]^\ell$.

- (iii) *Enc*($\text{params}, \text{pk}, \mu$): to encrypt a message $\mu \in \{0, 1\}$, choose a random vector $\mathbf{r} \leftarrow_r \{0, 1\}^\ell$, and then output

$$\mathbf{c} = \mathbf{A} \cdot \mathbf{r}^t + \left(\mathbf{0}, \mu \cdot \left\lceil \frac{p}{2} \right\rceil \right)^t \in \mathbb{Z}_q^n \times \mathbb{Z}_p. \quad (13)$$

- (iv) *Dec*($\text{params}, \text{sk}, \mathbf{c}$): given the ciphertext $\mathbf{c} \in \mathbb{Z}_q^n \times \mathbb{Z}_p$, output

$$\mu = \left\lceil \frac{1}{\lceil p/2 \rceil} \cdot \langle \mathbf{s}, \mathbf{c} \rangle \right\rceil \pmod{2}. \quad (14)$$

Correctness. As long as the noise $|e| \triangleq |\langle \mathbf{e}, \mathbf{r} \rangle| = |\sum_{i=1}^{\ell} r_i \cdot e_i| \leq 1/2 \cdot \ell < 1/2 \cdot \lceil p/2 \rceil$, where $r_i \in \{0, 1\}$ is the entry of the vector $\mathbf{r}$, the above Dec algorithm can rightly recover the message μ . We prove this by the following steps: For every column vector $(\mathbf{a}_i, b_i)^t$ of the matrix $\mathbf{A}$, it holds that

$$\begin{aligned} \mathbf{s} \cdot (\mathbf{a}_i, b_i)^t &= -\frac{p}{q} \langle \mathbf{s}', \mathbf{a}_i \rangle + b_i \\ &= -\frac{p}{q} \cdot \left(\langle \mathbf{s}', \mathbf{a}_i \rangle_q + kq \right) + b_i \\ &= -\frac{p}{q} \langle \mathbf{s}', \mathbf{a}_i \rangle_q + b_i - kp = e_i - kp. \end{aligned} \quad (15)$$

Then, we have $\mathbf{s} \cdot \mathbf{A} = \mathbf{e} \bmod p$, and it follows that

$$\begin{aligned} &\left\lceil \frac{1}{\lceil p/2 \rceil} \cdot \langle \mathbf{s}, \mathbf{c} \rangle \right\rceil \\ &= \left\lceil \frac{1}{\lceil p/2 \rceil} \cdot \left(\langle \mathbf{e}, \mathbf{r} \rangle + \mu \cdot \left\lceil \frac{p}{2} \right\rceil - kp \right) \right\rceil = \mu \bmod 2, \end{aligned} \quad (16)$$

iff $|\langle \mathbf{e}, \mathbf{r} \rangle| \leq 1/2 \cdot \ell < 1/2 \cdot \lceil p/2 \rceil$.

3.2. Security. We argue that the above basic encryption scheme is IND-CPA secure under Theorem 7. In fact, the structure of our basic encryption scheme is identical to that of [14, 16, 17], and so is the strategy of security proof. In our paper, Theorem 7 guarantees the hardness of $\text{LWR}_{n,m,q,p}$ assumption (where $\mathbf{s} \leftarrow_r \{0, 1\}^n$) assuming the bin-LWE problem; hence, we can base the security of our basic encryption scheme on the LWR problem. For completeness, we give a full security proof below.

Theorem 8. *The above basic LWR-based encryption scheme is IND-CPA secure under the $\text{LWR}_{n,m,q,p}$ assumption.*

Proof. We prove the theorem via a series of hybrid experiments. The analysis of probability is omitted here, since it is natural and very similar to that of [4, 14].

- (i) Hybrid 0: this is the real system (we set $\ell = m$, where $m = O(n \log q + \log p) \geq n \log q + \log p + 2\lambda$).
- (ii) Hybrid 1: this is the same as Hybrid 0 except that we use m pairs $\{(\mathbf{a}_i, b_i)\}_{i=1,\dots,m}$ chosen uniformly from $\mathbb{Z}_q^n \times \mathbb{Z}_p$, instead of being chosen from LWR samples in Hybrid 0, to assemble public key $\mathbf{A}$. The ciphertext $\mathbf{c}$ is thus generated by encrypting μ using the public key $\mathbf{A}$ as per the encryption procedure in Hybrid 0.
- (iii) Hybrid 2: this is the same as Hybrid 1 except that the ciphertext $\mathbf{c} \in \mathbb{Z}_q^n \times \mathbb{Z}_p$ is chosen from $\mathbb{Z}_q^n \times \mathbb{Z}_p$ uniformly and independently of the public key.

Firstly, we claim that Hybrid 1 is indistinguishable from Hybrid 0, under the $\text{LWR}_{n,m,q,p}$ assumption. This is shown by a simple reduction: if any hypothetical adversary $\mathcal{A}$ can distinguish these two hybrids, then we can construct an algorithm $\mathcal{B}$ to break the $\text{LWR}_{n,m,q,p}$ assumption. Specifically, $\mathcal{B}$ simply collects its input samples (challenged samples) into

the public key $\mathbf{A}$ and encrypts a message μ using the public key $\mathbf{A}$ to get a ciphertext $\mathbf{c}$, and then it invokes $\mathcal{A}$ on $(\mathbf{A}, \mathbf{c})$, outputting the same accept/reject decision. It is clear that $\mathcal{B}$ perfectly simulates Hybrid 0 or Hybrid 1 depending on whether its input samples are LWR samples or uniform over $\mathbb{Z}_q^n \times \mathbb{Z}_p$; that is, if the input samples are LWR samples, then the ciphertext $\mathbf{c}$ is generated as per Hybrid 0; otherwise, the ciphertext $\mathbf{c}$ is generated as per Hybrid 1. Therefore, $\mathcal{B}$ and $\mathcal{A}$ have equal distinguishing advantages. Because $\mathcal{B}$'s advantage must be negligible by hypothesis, so is $\mathcal{A}$'s.

In the second place, we claim that Hybrid 1 is statistically indistinguishable from Hybrid 2; that is, even a computationally unbounded adversary has only negligible advantage in distinguishing them. This is easy to be justified by Lemma 1 in Section 2. In more detail, by Lemma 1, the term $\mathbf{A} \cdot \mathbf{r}$ in Hybrid 1 is statistically indistinguishable from any element $\mathbf{y} \in \mathbb{Z}_q^n \times \mathbb{Z}_p$. Hence, the ciphertext in Hybrid 1 is statistically indistinguishable from that of Hybrid 2, for adding any fixed vector $(\mathbf{0}, \mu \cdot \lceil p/2 \rceil)^t$ to $\mathbf{A} \cdot \mathbf{r}$ preserves its uniform distribution.

To sum up, we conclude that Hybrid 0 is indistinguishable from Hybrid 2, and the ciphertext in Hybrid 2 is independent of message μ (regardless of $\mu = 0$ or $\mu = 1$), which completes the proof. $\square$

4. LWR-Based FHE Scheme

4.1. Our Construction. Our LWR-based FHE scheme is constructed as follows.

- (i) *Setup*($1^\lambda, 1^L$): choose dimension parameter $n = n(\lambda, L)$ and moduli $q = q(\lambda, L)$ and $p = p(\lambda, L)$ satisfying $p \mid q$. Also, choose parameter $m = m(\lambda, L) = O(n \log q + \log p)$. Let $\text{params} = (n, q, p, m)$. Let $k_1 = \lceil \log q \rceil$, $k_2 = \lceil \log p \rceil$, and $N = n \cdot k_1 + k_2$.
- (ii) *KeyGen*(params): choose uniformly at random a secret key $\mathbf{s}' \leftarrow_r \{0, 1\}^n$. As per KeyGen algorithm in Section 3, generate a matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{A} \\ \mathbf{b} \end{bmatrix} \in \mathbb{Z}_q^{n \times m} [\times] \mathbb{Z}_p^m. \quad (17)$$

The final public key is $pk := \mathbf{A}$, and the secret key is $sk : \mathbf{s} = (-p/q \cdot \mathbf{s}', 1)$. Note that it holds that $\mathbf{s} \cdot \mathbf{A} = \mathbf{e} \bmod p$, where the noise $\mathbf{e} = (e_1, \dots, e_m) \in [-1/2, 1/2]^m$.

- (iii) *Enc*(params, pk, μ): to encrypt a message $\mu \in \{0, 1\}$, choose a random matrix $\mathbf{R} \leftarrow_r \{0, 1\}^{m \times N}$. Output the ciphertext

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{R} + \mu \mathbf{G} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N, \quad (18)$$

where the gadget matrix $\mathbf{G}$ is brought from Section 2.3. Since we should perform the above matrix multiplication as per the specific matrix operations described in Section 2, we give Algorithm 1 that makes the encryption algorithm more readable.

Input: Parameters, a public-key matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m} [\times] \mathbb{Z}_p^m$ and a message $\mu \in \{0, 1\}$.

Output: A ciphertext matrix $\mathbf{C} = \mathbf{A} \cdot \mathbf{R} + \mu \mathbf{G} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$.

- (1) Generate a variant gadget matrix $\mathbf{G}$ according to the structure of gadget matrix given in Section 2.3.
- (2) Choose uniformly at random a matrix $\mathbf{R} \leftarrow_r \{0, 1\}^{m \times N}$.
- (3) Compute $\mathbf{C} = \mathbf{A} \cdot \mathbf{R} + \mu \mathbf{G} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$ as per our specific matrix operations described in Section 2.

ALGORITHM 1: Encryption algorithm.

Input: Parameters, a secret key $sk = \mathbf{s}$ and a ciphertext matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$.

Output: A binary value $\mu \in \{0, 1\}$.

- (1) Choose a vector $\mathbf{v} := (0, \dots, 0, \lceil p/2 \rceil) \in \mathbb{Z}_p^{n+1}$, and compute a vector $\mathbf{r} \in \{0, 1\}^N$, such that $\mathbf{G} \cdot \mathbf{r}^t = \mathbf{v}$, i.e., $\mathbf{r}^t = \mathbf{G}^{-1}(\mathbf{v}^t)$.
- (2) Compute

$$v = \mathbf{s} \cdot \mathbf{C} \cdot \mathbf{r}^t = \mathbf{e} \cdot \mathbf{R} \cdot \mathbf{r}^t + \mu \cdot \left\lfloor \frac{p}{2} \right\rfloor.$$

- (3) Output $\mu = \lceil 1/\lceil p/2 \rceil \cdot v \rceil \pmod{2}$.

ALGORITHM 2: Decryption algorithm.

Input: Two ciphertext matrices $\mathbf{C}_1, \mathbf{C}_2$ decrypting to messages μ_1, μ_2 , respectively.

Output: A derived ciphertext matrix $\mathbf{C}_{\text{mult}}$ decrypting to message $\mu_1 \mu_2$.

- (1) Compute a matrix $\mathbf{X} \in \{0, 1\}^{N \times N}$, such that $\mathbf{G} \cdot \mathbf{X} = \mathbf{C}_2$, i.e., $\mathbf{X} = \mathbf{G}^{-1}(\mathbf{C}_2)$.
- (2) Output

$$\mathbf{C}_{\text{mult}} = \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N.$$

ALGORITHM 3: Homomorphic multiplication.

- (iv) $\text{Dec}(\text{params}, sk, \mathbf{C})$: given the ciphertext $\mathbf{C} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$, choose a vector $\mathbf{v} := (0, \dots, 0, \lceil p/2 \rceil) \in \mathbb{Z}_p^{n+1}$ and compute

$$v = \mathbf{s} \cdot \mathbf{C} \cdot \mathbf{G}^{-1}(\mathbf{v}^t) = \mathbf{e} \cdot \mathbf{R} \cdot \mathbf{G}^{-1}(\mathbf{v}^t) + \mu \cdot \left\lfloor \frac{p}{2} \right\rfloor. \quad (19)$$

Output $\mu = \lceil 1/\lceil p/2 \rceil \cdot v \rceil \pmod{2}$. We also give Algorithm 2 that makes the decryption algorithm more readable.

- (v) $\text{Add}(\mathbf{C}_1, \mathbf{C}_2)$: given two ciphertext matrices $\mathbf{C}_1, \mathbf{C}_2$ decrypting to messages μ_1 and μ_2 , respectively, output

$$\mathbf{C}_{\text{Add}} \triangleq \mathbf{C}_1 + \mathbf{C}_2 \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N. \quad (20)$$

- (vi) $\text{Mult}(\mathbf{C}_1, \mathbf{C}_2)$: for two ciphertext matrices $\mathbf{C}_1, \mathbf{C}_2$ decrypting to messages μ_1 and μ_2 , respectively, the multiplication is defined as

$$\mathbf{C}_{\text{mult}} \triangleq \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N, \quad (21)$$

where $\mathbf{G}^{-1}$ is brought from Section 2.3. We give Algorithm 3 only for homomorphic multiplication, because homomorphic addition is trivial, and homomorphic NAND depends on homomorphic multiplication.

- (vii) $\text{NAND}(\mathbf{C}_1, \mathbf{C}_2)$: for two ciphertext matrices $\mathbf{C}_1, \mathbf{C}_2$ decrypting to messages μ_1 and μ_2 , respectively, the NAND operation is defined as

$$\mathbf{C}_{\text{NAND}} \triangleq \mathbf{G} - \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N. \quad (22)$$

Note that any Boolean circuit can be converted to use only NAND operation.

- (viii) $\text{Eval}(f, (\mathbf{C}_1, \mathbf{C}_2, \dots, \mathbf{C}_t))$: evaluate the Boolean function $f: \{0, 1\}^t \rightarrow \{0, 1\}$ on t ciphertexts $\mathbf{C}_1, \mathbf{C}_2, \dots, \mathbf{C}_t$ by the NAND operation. Output a ciphertext $\mathbf{C}_f$.

The IND-CPA security of the above scheme follows immediately from the security of the basic encryption scheme in Section 3. The ciphertext matrix is just $\mu \mathbf{G}$ plus a matrix of N encryptions of 0 under the key $\mathbf{s}$ by the basic encryption in Section 3, which is pseudorandom by Theorem 8 and hence hides $\mu \mathbf{G}$.

4.2. Correctness. In this subsection, we analyze the scheme's correctness and the noise growth of each homomorphic operation. Moreover, we also give a more detailed analysis for homomorphic multiplication to illustrate that our proposed construction is practicable.

Correctness. The correctness is obvious according to the correctness of the basic encryption scheme in Section 3, iff the

magnitude of noise $\mathbf{e}^* \triangleq \mathbf{e} \cdot \mathbf{R} \cdot \mathbf{G}^{-1}(\mathbf{v}^t)$ is less than $1/2 \cdot \lceil p/2 \rceil$. In fact, according to the Dec, we have

$$\mathbf{s} \cdot \mathbf{C} = \mu \mathbf{s} \cdot \mathbf{G} + \mathbf{e} \cdot \mathbf{R} \bmod p, \quad (23)$$

and then

$$\mathbf{s} \cdot \mathbf{C} \cdot \mathbf{G}^{-1}(\mathbf{v}^t) = \mu \cdot \left\lfloor \frac{p}{2} \right\rfloor + \mathbf{e} \cdot \mathbf{R} \cdot \mathbf{G}^{-1}(\mathbf{v}^t) \bmod p, \quad (24)$$

where $\|\mathbf{e}^*\| \triangleq \|\mathbf{e} \cdot \mathbf{R} \cdot \mathbf{G}^{-1}(\mathbf{v}^t)\| < (1/2)m \cdot \lceil \log p \rceil$. Hence, the correctness holds iff $(1/2)m \cdot \lceil \log p \rceil < (1/2) \cdot \lceil p/2 \rceil$.

Since the homomorphic addition is obvious, we mainly analyze homomorphic multiplication and NAND operation.

Homomorphic Multiplication. To multiply two ciphertext matrices $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$ designated for messages $\mu_1, \mu_2 \in \{0, 1\}$, we have

$$\begin{aligned} \mathbf{s} \cdot \text{Mult}(\mathbf{C}_1, \mathbf{C}_2) &= \mathbf{s} \cdot \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) \\ &= (\mathbf{e}_1 + \mu_1 \mathbf{s} \cdot \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{C}_2) \\ &= \mathbf{e}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1 (\mathbf{e}_2 + \mu_2 \mathbf{s} \cdot \mathbf{G}) \quad (25) \\ &= (\mathbf{e}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1 \mathbf{e}_2) + \mu_1 \mu_2 \mathbf{s} \\ &\quad \cdot \mathbf{G}, \end{aligned}$$

where $\mathbf{G}^{-1}(\mathbf{C}_2) \in \{0, 1\}^{N \times N}$ and $\mathbf{e}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1 \mathbf{e}_2$ is the total noise which is of magnitude

$$\|\mathbf{e}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1 \mathbf{e}_2\| \leq \frac{1}{2}m(N+1), \quad (26)$$

due to $\|\mathbf{e}_i\|_{i \in [2]} \leq (1/2)m$ by the correctness of our basic encryption scheme. Therefore, it is clear that the noise growth factor is $N+1$, and after L levels of homomorphic multiplication, the noise grows from initial magnitude of $(1/2)m$ to $(1/2)m(N+1)^L$. By comparison, in GSW13, the noise at L level is near $m'B(N'+1)^L$, where $m' = O(n \log q)$, $N' = (n+1) \cdot \lceil \log q \rceil$, $B \geq \omega(\log n) \cdot \sqrt{n}$.

Homomorphic NAND. To perform NAND operation on two ciphertext matrices $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$ designated for messages $\mu_1, \mu_2 \in \{0, 1\}$, we have

$$\begin{aligned} \mathbf{s} \cdot \text{NAND}(\mathbf{C}_1, \mathbf{C}_2) &= \mathbf{s} \cdot (\mathbf{G} - \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)) \\ &= (\mathbf{e}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1 \mathbf{e}_2) \quad (27) \\ &\quad + (1 - \mu_1 \mu_2) \mathbf{s} \cdot \mathbf{G}, \end{aligned}$$

where $\mathbf{e}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1 \mathbf{e}_2$ is the noise of homomorphic NAND, which is identical to that of homomorphic multiplication. Therefore, the noise is also increased by a factor of at most $N+1$.

Analysis for Homomorphic Multiplication. Given two ciphertext matrices $\mathbf{C} = \mathbf{A} \cdot \mathbf{R} + \mu \mathbf{G}$, $\mathbf{C}' = \mathbf{A} \cdot \mathbf{R}' + \mu' \mathbf{G} \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$

designated for messages $\mu, \mu' \in \{0, 1\}$, we let $\mathbf{C} = \begin{bmatrix} \mathbf{C}_q \\ \mathbf{C}_p \end{bmatrix}$, $\mathbf{C}' = \begin{bmatrix} \mathbf{C}'_q \\ \mathbf{C}'_p \end{bmatrix} \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$. Recall that

$$\begin{aligned} \mathbf{G} &= \begin{bmatrix} 1, \dots, 2^{\lceil \log q \rceil - 1} & & \\ & \ddots & \\ & & 1, \dots, 2^{\lceil \log q \rceil - 1} \\ & & & 1, \dots, 2^{\lceil \log p \rceil - 1} \end{bmatrix} \quad (28) \\ &\in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N. \end{aligned}$$

Compute $\mathbf{G}^{-1}(\mathbf{C}')$ and let $\mathbf{X} = \mathbf{G}^{-1}(\mathbf{C}')$, that is, $\mathbf{G} \cdot \mathbf{X} = \mathbf{C}'$, and then we have $\mathbf{X} = \begin{bmatrix} \mathbf{X}_q \\ \mathbf{X}_p \end{bmatrix} \in \{0, 1\}^{N \times N}$, where $\mathbf{X}_q \in \{0, 1\}^{(N - \lceil \log p \rceil) \times N}$ and $\mathbf{X}_p \in \{0, 1\}^{\lceil \log p \rceil \times N}$. Then, we have

$$\begin{aligned} \text{Mult}(\mathbf{C}, \mathbf{C}') &= \mathbf{C} \cdot \mathbf{G}^{-1}(\mathbf{C}') = (\mathbf{A} \cdot \mathbf{R} + \mu \mathbf{G}) \cdot \begin{bmatrix} \mathbf{X}_q \\ \mathbf{X}_p \end{bmatrix} \\ &= \mathbf{A} \cdot \mathbf{R} \cdot \begin{bmatrix} \mathbf{X}_q \\ \mathbf{X}_p \end{bmatrix} + \mu \mathbf{C}' \quad (29) \\ &= \mathbf{A} \cdot \mathbf{R} \cdot \begin{bmatrix} \mathbf{X}_q \\ \mathbf{X}_p \end{bmatrix} + \mathbf{A} \cdot \mu \mathbf{R}' + \mu \mu' \mathbf{G} \\ &= \mathbf{A} \cdot (\mathbf{R} \cdot \mathbf{X} + \mu \mathbf{R}') + \mu \mu' \mathbf{G}, \end{aligned}$$

where, as per the specific matrix multiplication described in Section 2, we can compute $\mathbf{A} \cdot \mathbf{R} \cdot \mathbf{X} \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$. Since $\mathbf{s} \cdot \mathbf{A} = \mathbf{e} \bmod p$, $\mathbf{X} = \begin{bmatrix} \mathbf{X}_q \\ \mathbf{X}_p \end{bmatrix}$ does not affect the correctness of $\mathbf{s} \cdot (\mathbf{A} \cdot \mathbf{R} \cdot \mathbf{X}) = \mathbf{e} \cdot \mathbf{R} \cdot \mathbf{X} \bmod p$ (recall that $\mathbf{s} = (-p/q \cdot \mathbf{s}', 1)$). Therefore, (25) holds.

4.3. Parameters and Comparisons. Compared to GSW13 and API4, we conclude, from the above analysis, that our noise growth factor is also $N+1$ (identical to theirs) and the size of parameter q is almost the same as theirs. Hence, we claim that the efficiency of our scheme is almost comparable to theirs without considering the cost of the Gaussian noise sampling, since our encryption and decryption processes are identical to theirs. Concretely, we assume the depth of NAND operation is L , and after L levels of homomorphic NAND operation, the noise is near $(1/2)m(N+1)^L$, and it holds that $(1/2)m(N+1)^L < 1/2 \cdot \lceil p/2 \rceil$ for decryption. Then, combining with the requirements of $q \geq 2mpB$ in Theorem 7 in Section 2, and $m = O(n \log q + \log p)$, $N = n \lceil \log q \rceil + \lceil \log p \rceil$ in our scheme, we can set $q = (O(n \log q))^{L+O(1)}$, $p = (O(n \log p))^{L+O(1)}$ satisfying $q \geq 2mpB$. Compared to the parameter q of $q = (O(n \log q))^{L+O(1)}$ (since $q \geq 8m'B(N'+1)^L$) in GSW13 and API4, our parameter q is larger (up to a polynomial factor); this is caused by the condition $q \geq 2mpB$ needed for achieving the same (almost) security level as GSW13 and API4. Therefore, in consideration of the cost (e.g., running time, memory) of Gaussian or sub-Gaussian

noise sampling in GSW13 and AP14, our scheme has some advantages over them and thus can be seen as an alternative to them. Actually, with the development of reduction between LWE and LWR, the moduli can be reduced.

Bootstrapping involves homomorphically evaluating the decryption function. Because our decryption process is identical to AP14, bootstrapping works on our proposed scheme as well. In more detail, similar to AP14, we can also directly evaluate the decryption function in an elementary and efficient arithmetic form, using just basic facts about cyclic groups, so as to achieve bootstrapping. In our scheme, the bootstrapping method also just results in polynomial noise, which allows the security to be based on the LWR problem with inverse-polynomial noise rates. Since the reduction from LWE to LWR incurs a polynomial loss ($O(qn \log q)$) according to Theorem 7 in Section 2, the security can be based on LWE problem with inverse-polynomial noise rates and hence on worst-case lattice problems (e.g., GapSVP) with polynomial approximation factors.

5. LWR-Based Multikey FHE Scheme

FHE has been studied extensively also due to its versatility in cryptography and industrial application, such as the problem of outsourcing computation to a third trust party (a remote server) without compromising its privacy [5] and secure multiparty computation problem [12, 30]. In Eurocrypt 2016, Mukherjee and Wichs [12] extended AP14 to multikey case and presented a multikey FHE scheme for implementing two round multiparty computations, which enabled performing homomorphic computation over encrypted data under different keys and satisfied threshold decryption. Their solution, however, relies on the LWE problem, the efficiency of which is retarded by the sub-Gaussian noise sampling. But the tricks in [12] inspire us to construct a multikey FHE scheme based on LWR. In fact, our construction also satisfies threshold decryption; we will describe it in Section 5.3. Here, we just present some necessary tools and our main construction, and we refer interested readers to [12] for the definitions of multikey FHE and multiparty computation, the security definition of multiparty computation protocol, and some other related concepts.

5.1. Overview of the Techniques. In this subsection, we will describe how to construct a LWR-based multikey FHE scheme (adapted from [12]); this helps readers understand our construction more easily. For simplicity, we consider a case of two keys, for it is natural to generalize to the case of any polynomial number of keys. According to (23) in Section 4.2, given two ciphertexts C_1 and C_2 decrypting to messages μ_1 and μ_2 , we can perform homomorphic operations on these two ciphertexts, iff they satisfy the equation $\mathbf{s} \cdot \mathbf{C} = \mu_i \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_i$ for small noise $\mathbf{e}_i$, where $i \in [2]$. Hence, for the “combined secret key” $\widehat{\mathbf{s}} = (\mathbf{s}_1, \mathbf{s}_2)$, where $\mathbf{s}_1 = (-p/q \cdot \mathbf{s}'_1, 1)$ and $\mathbf{s}_2 = (-p/q \cdot \mathbf{s}'_2, 1)$ which correspond to two different users, if we can construct two expanded ciphertexts $\widehat{C}_1$ and $\widehat{C}_2$, such that $\widehat{\mathbf{s}} \cdot \widehat{C}_1 \approx \mu_1 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}$ and $\widehat{\mathbf{s}} \cdot \widehat{C}_2 \approx \mu_2 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}$ (henceforth, $\approx$ is used for omitting the noise term $\mathbf{e}$), where $\widehat{\mathbf{G}}$ is the expansion of $\mathbf{G}$,

then we can perform homomorphic operations on these two ciphertexts $\widehat{C}_1$ and $\widehat{C}_2$.

Exactly as Mukherjee and Wichs [12] showed, we can create the expanded ciphertext as follows:

$$\widehat{C}_1 := \begin{bmatrix} C_1 & X_1 \\ \mathbf{0} & C_1 \end{bmatrix}, \quad (30)$$

such that

$$\widehat{\mathbf{s}} \cdot \widehat{C}_1 \approx \mu_1 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}, \quad (31)$$

iff

$$\mathbf{s}_1 \cdot X_1 + \mathbf{s}_2 \cdot C_1 \approx \mu_1 \mathbf{s}_2 \cdot \mathbf{G}, \quad (32)$$

where $\widehat{\mathbf{G}} := \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} \end{bmatrix}$. Similarly, we can also generate the ciphertext $\widehat{C}_2$. Here, we have made a big step forward; next, we proceed to analyze how to construct this X without revealing the secret key $\mathbf{s}_2$.

Assume we are given a user-specific public key $\mathbf{A}_1 = \begin{bmatrix} \overline{\mathbf{A}} \\ \mathbf{b}_1 \end{bmatrix}$, a secret key $\mathbf{s}_1 = (-p/q \cdot \mathbf{s}'_1, 1)$, and a ciphertext

$$C_1 = \mathbf{A}_1 \cdot \mathbf{R}_1 + \mu \mathbf{G} = \begin{bmatrix} \overline{\mathbf{A}} \cdot \mathbf{R}_1 \\ \mathbf{b}_1 \cdot \mathbf{R}_1 \end{bmatrix} + \mu \mathbf{G}, \quad (33)$$

which is generated as per Enc algorithm in Section 4. Assume we are also given another user-specific public key $\mathbf{A}_2 = \begin{bmatrix} \overline{\mathbf{A}} \\ \mathbf{b}_2 \end{bmatrix}$ and secret key $\mathbf{s}_2 = (-p/q \cdot \mathbf{s}'_2, 1)$ (assume $\overline{\mathbf{A}}$ is the public shared parameter), such that $\mathbf{s}_2 \cdot \mathbf{A}_2 = \mathbf{e} \bmod p$; then, we have

$$\begin{aligned} \mathbf{s}_2 \cdot C_1 &= -\frac{p}{q} \cdot \mathbf{s}'_2 \cdot \overline{\mathbf{A}} \cdot \mathbf{R}_1 + \mathbf{b}_1 \cdot \mathbf{R}_1 + \mu_1 \mathbf{s}_2 \cdot \mathbf{G} \\ &\approx (\mathbf{b}_1 - \mathbf{b}_2) \cdot \mathbf{R}_1 + \mu_1 \mathbf{s}_2 \cdot \mathbf{G}. \end{aligned} \quad (34)$$

Subtracting (34) from (32), we get $\mathbf{s}_1 \cdot X_1 \approx (\mathbf{b}_2 - \mathbf{b}_1) \cdot \mathbf{R}_1$ over $\mathbb{Z}_p$. We proceed to use an important tool brought from [12] to construct such matrix X .

Linear Combination. For a matrix $\mathbf{R} \in \{0, 1\}^{m \times N}$ with mN entries $\{\mathbf{R}[i, j]\}$ for $i \in [m]$, $j \in [N]$, let $\mathbf{C}^{(i, j)} \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$ be LWR-based FHE encryption of $\mathbf{R}[i, j]$ under a secret key $\mathbf{s} = (-p/q \cdot \mathbf{s}', 1)$, where $\mathbf{s}' \leftarrow_r \{0, 1\}^n$. Let $\mathbf{b} \in \mathbb{Z}_p^m$ be a vector, where $\mathbf{b}[i]$ is its i th entry. Then, there is a polynomial time deterministic algorithm

$$\mathbf{C}^{\text{lc}} = \text{LComb}((\mathbf{C}^{(1,1)}, \dots, \mathbf{C}^{(m,N)}), \mathbf{b}), \quad (35)$$

which outputs $\mathbf{C}^{\text{lc}} \in \mathbb{Z}_q^{n \times N}[\times] \mathbb{Z}_p^N$ such that $\mathbf{s} \cdot \mathbf{C}^{\text{lc}} = \mathbf{b} \cdot \mathbf{R} + \mathbf{e}$, where $\|\mathbf{e}\|_\infty \leq (1/2)(mN)^2$.

The algorithm $\text{LComb}((\mathbf{C}^{(1,1)}, \dots, \mathbf{C}^{(m,N)}), \mathbf{b})$ is implemented as follows:

- (1) For each $i \in [m]$, $j \in [N]$, define a matrix $\mathbf{Z}_{i,j} \in \mathbb{Z}^{m \times N}_p$ as follows:

$$\mathbf{Z}_{i,j}[a, b] = \begin{cases} \mathbf{b}[i] & \text{when } a = n, b = j, \\ 0 & \text{otherwise.} \end{cases} \quad (36)$$

In other words, $\mathbf{Z}_{i,j}$ will be 0 everywhere except the n th row and j th column where it has the value $\mathbf{b}[i]$.

- (2) Output $\mathbf{C}^{\text{lc}} \in \mathbb{Z}_q^{n \times N} [\times] \mathbb{Z}_p^N$, where $\mathbf{C}^{\text{lc}} = \sum_{i=1, j=1}^{m, N} \mathbf{C}^{(i,j)} \cdot \mathbf{G}^{-1}(\mathbf{Z}_{i,j})$.

Then, we analyze its correctness as follows:

$$\begin{aligned} \mathbf{s} \cdot \mathbf{C}^{\text{lc}} &= \mathbf{s} \cdot \sum_{i,j}^{m, N} \mathbf{C}^{(i,j)} \cdot \mathbf{G}^{-1}(\mathbf{Z}_{i,j}) \\ &= \sum_{i,j}^{m, N} (\mathbf{R}[i, j] \cdot \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{i,j}') \cdot \mathbf{G}^{-1}(\mathbf{Z}_{i,j}) \\ &= \sum_{i,j}^{m, N} (\mathbf{R}[i, j] \cdot \mathbf{s} \cdot \mathbf{Z}_{i,j} + \mathbf{e}_{i,j}') \\ &= \mathbf{s} \cdot \sum_{i,j}^{m, N} \mathbf{R}[i, j] \cdot \mathbf{Z}_{i,j} + \sum_{i,j}^{m, N} \mathbf{e}_{i,j}' \\ &= \left(-\frac{p}{q} \cdot \mathbf{s}', 1 \right) \cdot \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \cdot \mathbf{R} \end{bmatrix} + \mathbf{e}'' = \mathbf{b} \cdot \mathbf{R} + \mathbf{e}'', \end{aligned} \quad (37)$$

where $\mathbf{e}_{i,j}$ is the noise contained in $\mathbf{C}^{(i,j)}$ with magnitude of $\|\mathbf{e}_{i,j}\|_{\infty} \leq (1/2)m$ (according to Section 3), and then $\mathbf{e}_{i,j}' = \mathbf{e}_{i,j} \cdot \mathbf{G}^{-1}(\mathbf{Z}_{i,j})$ has magnitude $\|\mathbf{e}_{i,j}'\|_{\infty} \leq (1/2)mN$, and finally $\mathbf{e}'' = \sum_{i,j}^{m, N} \mathbf{e}_{i,j}'$ has magnitude $\|\mathbf{e}''\|_{\infty} \leq (1/2)(mN)^2$.

5.2. Construction of Two-Key FHE Scheme from LWR. For ease of description, here we just give a construction of two-key FHE scheme based on LWR, for everything extends naturally to any polynomial number of keys.

Construction. Now, we describe the two-key FHE construction.

- (i) *TFHE.Setup*($1^\lambda, 1^L$): run the *Setup* and *KeyGen* algorithms in Section 4 to generate the parameters:

$$\text{params} := (n, q, p, m, \overline{\mathbf{A}}). \quad (38)$$

- (ii) *TFHE.KeyGen*(*params*): run the *KeyGen* algorithms in Section 4 to get

$$\begin{aligned} sk : \mathbf{s}_i &= \left(-\frac{p}{q} \cdot \mathbf{s}_i', 1 \right), \\ pk : \mathbf{A}_i &= \begin{bmatrix} \overline{\mathbf{A}} \\ \mathbf{b}_i \end{bmatrix} \in \mathbb{Z}_q^{n \times m} [\times] \mathbb{Z}_p^m, \end{aligned} \quad (39)$$

for $i \in [2]$.

- (iii) *TFHE.Enc*(*params*, pk_i, μ_i): for $i \in [2]$, run the *Enc* algorithm in Section 4 to get a ciphertext $\mathbf{C}_i$ and the corresponding random matrix $\mathbf{R}_i \in \{0, 1\}^{m \times N}$. For all entries $\{\mathbf{R}_i[i, j]\}_{i \in [m], j \in [N]}$ of the uniform matrix

$\mathbf{R}_i \in \{0, 1\}^{m \times N}$, we use the same secret key $\mathbf{s}_i$ to encrypt them, which leads to a helper set $\mathcal{H}_i = \{\mathbf{C}_i^{(i,j)}\}_{i \in [m], j \in [N]}$. Output the ciphertext $\mathbf{C}_i$ and the helper set $\mathcal{H}_i$.

- (iv) *TFHE.Expand*(*params*, (pk_1, pk_2), $\mathbf{C}_i, \mathcal{H}_i$): given $\mathbf{C}_i$ and $\mathcal{H}_i$, when $i = 1$, run the linear combination algorithm to generate a matrix $\mathbf{X}_1 = \text{LComb}((\mathbf{C}_1^{(1,1)}, \dots, \mathbf{C}_1^{(m,N)}), \mathbf{b}_2 - \mathbf{b}_1)$, such that

$$\mathbf{s}_1 \cdot \mathbf{X}_1 = (\mathbf{b}_2 - \mathbf{b}_1) \cdot \mathbf{R}_1 + \mathbf{e}_{\mathbf{X}_1}. \quad (40)$$

Finally, output an expanded ciphertext:

$$\widehat{\mathbf{C}}_1 := \begin{bmatrix} \mathbf{C}_1 & \mathbf{X}_1 \\ \mathbf{0} & \mathbf{C}_1 \end{bmatrix}, \quad (41)$$

where ciphertext $\mathbf{C}_1$ corresponds to the message μ_1 . Similarly, when $i = 2$, output an expanded ciphertext $\widehat{\mathbf{C}}_2$.

- (v) *TFHE.Eval*(*params*, $f, (\widehat{\mathbf{C}}_1, \widehat{\mathbf{C}}_2)$): on inputting two expanded ciphertexts, run *Eval* algorithm in Section 4, albeit with expanded dimensions $n' = 2n + 2$ and $N' = 2N$, and the gadget matrix $\widehat{\mathbf{G}} := \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} \end{bmatrix}$. Output the ciphertext $\widehat{\mathbf{C}}_f$.
- (vi) *TFHE.Dec*(*params*, (sk_1, sk_2), $\widehat{\mathbf{C}}_f$): on inputting a ciphertext $\widehat{\mathbf{C}}_f$ and two secret keys sk_1 and sk_2 , parse $sk_1 = \mathbf{s}_1$ and $sk_2 = \mathbf{s}_2$, and then obtain a new secret key $\widehat{\mathbf{s}} = (\mathbf{s}_1, \mathbf{s}_2)$. Finally, run the *Dec* algorithm in Section 4 to get a plaintext $f(\mu_1, \mu_2)$.

Correctness. It is sufficient to verify that $\widehat{\mathbf{s}} \cdot \widehat{\mathbf{C}}_1 \approx \mu_1 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}$ and $\widehat{\mathbf{s}} \cdot \widehat{\mathbf{C}}_2 \approx \mu_2 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}$. Adding (34) and (40) up, that is,

$$\mathbf{s}_2 \cdot \mathbf{C}_1 \approx (\mathbf{b}_1 - \mathbf{b}_2) \cdot \mathbf{R}_1 + \mu_1 \mathbf{s}_2 \cdot \mathbf{G} \quad (42)$$

plus

$$\mathbf{s}_1 \cdot \mathbf{X}_1 = (\mathbf{b}_2 - \mathbf{b}_1) \cdot \mathbf{R}_1 + \mathbf{e}_{\mathbf{X}_1}, \quad (43)$$

results in

$$\begin{aligned} \widehat{\mathbf{s}} \cdot \widehat{\mathbf{C}}_1 &= (\mathbf{s}_1 \cdot \mathbf{C}_1, \mathbf{s}_1 \cdot \mathbf{X}_1 + \mathbf{s}_2 \cdot \mathbf{C}_1) \\ &\approx (\mu_1 \mathbf{s}_1 \cdot \mathbf{G}, \mu_1 \mathbf{s}_2 \cdot \mathbf{G}) = \mu_1 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}. \end{aligned} \quad (44)$$

This also applies to verify $\widehat{\mathbf{s}} \cdot \widehat{\mathbf{C}}_2 \approx \mu_2 \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}}$.

Parameters and Security. Though our LWR-based multikey FHE scheme does not need the sub-Gaussian noise sampling compared to Mukherjee and Wichs's [12], the structure of our construction is identical to theirs. Indeed, we just cleverly use the scaled rounding function instead of their sub-Gaussian noise, to hide the message, and this is done thanks to our specific structure which helps us circumvent the tangly modulus problem that exists in [22]. Since the LWE-based multikey FHE [12] is an extension of AP14, while the

parameter q in our LWR-based FHE is almost identical to that of AP14 according to Section 4.2, hence the LWR-based multikey FHE which is the extension of LWR-based FHE has almost the same parameter q as that of [12].

As to the security, we remark that the main difference between multikey setting in this section and single key setting in Section 4 is the number of ciphertexts. In multikey setting, there is an additional helper set $\mathcal{H}$ which involves mN ciphertexts, but the mN ciphertexts are encryptions of entries of the random matrix $\mathbf{R} \in \{0, 1\}^{m \times N}$, which are independent of other ciphertexts. In other words, the helper set $\mathcal{H}$ does not affect scheme's security; therefore, our LWR-based multikey FHE scheme has the same security level as the LWR-based FHE scheme in Section 4.

5.3. Threshold Decryption for Two-Key FHE Scheme. In [12], the main idea of designing a multiparty computation protocol is to leverage the property of “threshold decryption” of multikey FHE and rely on the “smudging lemma [31].” In the secure multiparty computation distributed protocol, the parties run a secure distributed protocol using their own secret key to decrypt the output ciphertext and finally recover the plaintext y . Specifically, each party will obtain partial information by operating their own secret key on the common ciphertext $\widehat{\mathbf{C}}$, and then they broadcast the information they get added by some medium-sized smudging noise from a uniform distribution (adding the noise is needed to “smudge out” any information about the noise contained in ciphertext $\widehat{\mathbf{C}}$ and is needed for security proof); finally, each party can sum up all the other partial information and get the final decryption y . We can achieve the construction of multiparty computation protocol as well, for our LWR-based multikey FHE scheme also satisfies the threshold decryption. Compared to [12], our threshold decryption also needs the smudging noise, but the whole computation overhead is lower in that our LWR-based multikey FHE scheme does not need the sub-Gaussian noise sampling. The threshold decryption follows. For simplicity, we again cover the case of two-key FHE.

(i) $TFHE.PartDec(\widehat{\mathbf{C}}, (pk_1, pk_2), (sk_1, sk_2))$: given the output (common) ciphertext $\widehat{\mathbf{C}}$ under two public keys pk_1, pk_2 and the secret keys $sk_1 = \mathbf{s}_1, sk_2 = \mathbf{s}_2$, do the following:

- (1) Parse the ciphertext $\widehat{\mathbf{C}}$ consisting of two submatrices $\widehat{\mathbf{C}}^{(i)}$ such that $\widehat{\mathbf{C}} = \begin{bmatrix} \widehat{\mathbf{C}}^{(1)} \\ \widehat{\mathbf{C}}^{(2)} \end{bmatrix}$.
- (2) Define a vector $\widehat{\mathbf{v}} \in \mathbb{Z}_p^{2n+2}$ as $\widehat{\mathbf{v}} = (0, \dots, 0, \lceil p/2 \rceil)$.
- (3) For $i \in [2]$, compute $\phi_i = \mathbf{s}_i \widehat{\mathbf{C}}^{(i)} \cdot \widehat{\mathbf{G}}^{-1}(\widehat{\mathbf{v}}^t) \in \mathbb{Z}_p$, and then compute $\tau_i = \phi_i + e_i^* \in \mathbb{Z}_p$, where $e_i^* \leftarrow_r [-B_{\text{smdg}}^{\text{dec}}, B_{\text{smdg}}^{\text{dec}}]$ is random “smudging noise.” We can set $B_{\text{smdg}}^{\text{dec}} = 2^{L\lambda \log \lambda}$. Note that the parameter $B_{\text{smdg}}^{\text{dec}} = B_\chi \cdot 2^{L\lambda \log \lambda}$ in [12], where $B_\chi > \omega(\log n) \cdot \sqrt{n}$.

(ii) $TFHE.FinDec(\tau_1, \tau_2)$: given τ_1, τ_2 , compute the sum $\tau = \sum_{i=1}^2 \tau_i$. Finally, output $\mu' = \lfloor (1/\lceil p/2 \rceil) \cdot \tau \rfloor$.

Correctness. Given the common ciphertext $\widehat{\mathbf{C}}$ designated for the message $\mu \in \{0, 1\}$, parse the parties' keys as $\widehat{\mathbf{s}} = (\mathbf{s}_1, \mathbf{s}_2)$. By the correctness of LWR-based two-key FHE scheme, it holds that

$$\widehat{\mathbf{s}} \cdot \widehat{\mathbf{C}} = \sum_{i=1}^2 \mathbf{s}_i \cdot \widehat{\mathbf{C}}^{(i)} = \mu \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}} + \widehat{\mathbf{e}} \bmod p. \quad (45)$$

Then, we have

$$\begin{aligned} \tau &= \sum_{i=1}^2 \tau_i = \sum_{i=1}^2 \phi_i + \sum_{i=1}^2 e_i^* = \sum_{i=1}^2 \mathbf{s}_i \cdot \mathbf{C}^{(i)} \cdot \widehat{\mathbf{G}}^{-1}(\widehat{\mathbf{v}}^t) + e^* \\ &= (\mu \widehat{\mathbf{s}} \cdot \widehat{\mathbf{G}} + \widehat{\mathbf{e}}) \cdot \widehat{\mathbf{G}}^{-1}(\widehat{\mathbf{v}}^t) + e^* \\ &= \mu \cdot \left\lfloor \frac{p}{2} \right\rfloor + e' + e^* \bmod p, \end{aligned} \quad (46)$$

where $|e' + e^*| < 1/2 \cdot \lceil p/2 \rceil$ holds under the appropriate parameters; for example, we can set $p = 2^{\omega(L\lambda \log \lambda)}$. Therefore, the correctness holds.

6. Conclusions

We present the first workable LWR-based FHE scheme. Our FHE scheme erases the expensive Gaussian noise sampling and thus can be seen as an alternative to the existing LWE-based FHEs. Furthermore, based on our main construction, we give the first LWR-based multikey FHE scheme, which is an alternative to the existing multikey FHEs that can also be applied to multiparty computation. However, neither our LWR-based multikey FHE scheme nor Mukherjee and Wichs's can be used to construct multiparty computation protocol without the smudging lemma (which means that the corresponding modulus has to be set exponentially in security parameter). Therefore, it remains an open problem to construct multiparty computation protocol via multikey FHEs without the smudging lemma.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is supported in part by the National Basic Research Program of China (973 Project, no. 2014CB340603), the National Key Research and Development Program of China (2017YFB0802000), the National Natural Science Foundation of China (nos. 61672030, 61272040, and U1705264), the Research Foundation of Hangzhou Normal University (no. 2017QDL002), and the Scientific Research Fund of Zhejiang Provincial Education Department (no. Y201737292).

References

- [1] R. L. Rivest, L. Adleman, and M. L. Dertouzos, “On data banks and privacy homomorphisms,” *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

- [2] C. Gentry, *A Fully Homomorphic Encryption Scheme*, Stanford University, 2009.
- [3] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, vol. 6110 of *Lecture Notes in Computer Science*, pp. 24–43, Springer, Berlin, Germany, 2010.
- [4] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Annual Cryptology Conference*, vol. 6841 of *Lecture Notes in Computer Science*, pp. 505–524, Springer, Heidelberg, Berlin, Germany, 2011.
- [5] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pp. 97–106, IEEE Computer Society, 2011.
- [6] M. Clear and C. McGoldrick, "Multi-identity and multi-key leveled fhe from learning with errors," in *Proceedings of the Annual Cryptology Conference*, pp. 630–656, Springer, Heidelberg, Berlin, Germany, 2015.
- [7] N. P. Smart and F. Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes," in *Public Key Cryptography*, vol. 6056 of *Lecture Notes in Computer Science*, pp. 420–443, Springer, Berlin, Germany, 2010.
- [8] L. Ducas and D. Micciancio, "FHEw: Bootstrapping homomorphic encryption in less than a second," *EUROCRYPT*, vol. 9056, pp. 617–640, 2015.
- [9] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," in *Proceedings of the 3rd Conference on Innovations in Theoretical Computer Science, ITCS 2012*, pp. 309–325, USA, January 2012.
- [10] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical GapSVP," in *CRYPTO*, vol. 7417, pp. 868–886, Springer, 2012.
- [11] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds," in *Proceedings of the 22nd International Conference on the Theory and Application of Cryptology and Information Security, Lecture Notes in Computer Science*, pp. 3–33, Springer, Hanoi, Vietnam, 2016.
- [12] P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key fhe," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, vol. 9666 of *Lecture Notes in Computer Science*, pp. 735–763, Springer, Berlin, Germany, 2016.
- [13] C. Peikert and S. Shiehian, "Multi-key fhe from lwe, revisited," in *Theory of Cryptography Conference*, vol. 9986 of *Lecture Notes in Computer Science*, pp. 217–238, Springer, Berlin, Germany, 2016.
- [14] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM (JACM)*, vol. 56, no. 6, Article ID 1568324, 34 pages, 2009.
- [15] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC '09*, pp. 333–342, ACM, USA, June 2009.
- [16] 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*, vol. 8042, pp. 75–92, Springer, 2013.
- [17] J. Alperin-Sheriff and C. Peikert, "Faster bootstrapping with polynomial error," in *Proceedings of the International Cryptology Conference*, pp. 297–314, Springer, Berlin, Germany, 2014.
- [18] Z. Brakerski, D. Cash, R. Tsabary, and H. Wee, "Targeted homomorphic attribute-based encryption," in *Proceedings of the Theory of Cryptography Conference*, pp. 330–360, Springer, Berlin, Germany, 2016.
- [19] S. Gorbunov, V. Vaikuntanathan, and D. Wichs, "Leveled fully homomorphic signatures from standard lattices," in *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, pp. 469–477, Portland, Oregon, USA, June 2015.
- [20] P. Pessl, "Analyzing the shuffling side-channel countermeasure for lattice-based signatures," in *Proceedings of the 17th International Conference on Cryptology in India*, vol. 10095, pp. 153–170, Springer, Kolkata, India, 2016.
- [21] L. G. Bruinderink, A. Hülsing, T. Lange, and Y. Yarom, "Flush, gauss, and reload—a cache attack on the bliss lattice-based signature scheme," in *Proceedings of the International Conference on Cryptographic Hardware and Embedded Systems*, pp. 323–345, Springer, 2016.
- [22] A. Costache and N. P. Smart, "Homomorphic encryption without gaussian noise," *IACR Cryptology ePrint Archive*, vol. 163, 2017.
- [23] A. Banerjee, C. Peikert, and A. Rosen, "Pseudorandom functions and lattices," *Advances in Cryptology-EUROCRYPT 2012*, pp. 719–737, 2012.
- [24] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé, "Classical hardness of learning with errors (extended abstract)," in *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, pp. 575–584, ACM, New York, NY, USA, 2013.
- [25] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby, "A pseudorandom generator from any one-way function," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1364–1396, 1999.
- [26] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs, "Learning with rounding, revisited: New reduction, properties and applications," in *Advances in Cryptology-CRYPTO 2013*, pp. 57–74, Springer, 2013.
- [27] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen, "On the hardness of learning with rounding over small modulus," in *Proceedings of the Theory of Cryptography Conference*, vol. 9562, pp. 209–224, Springer, Berlin, Germany, 2016.
- [28] J. Alperin-Sheriff and D. Apon, "Dimension-preserving reductions from lwe to lwr," *IACR Cryptology ePrint Archive*, vol. 2016, no. 589, 2016.
- [29] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *EuroCrypt*, vol. 7237 of *Lecture Notes in Computer Science*, pp. 700–718, Springer, Heidelberg, Berlin, Germany, 2012.
- [30] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, pp. 1219–1234, ACM, New York, NY, USA, 2012.
- [31] G. Asharov, A. Jain, A. López-Alt, E. Tromer, V. Vaikuntanathan, and D. Wichs, "Multiparty computation with low communication, computation and interaction via threshold FHE," in *Proceedings of the 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology - EUROCRYPT 2012*, pp. 483–501, Cambridge, UK, April 15–19, 2012.