A Fully Homomorphic Encryption Scheme with Better Key Size

Zhigang Chen [1,2a,3], Jian Wang [1], ZengNian Zhang [2b], Xinxia Song [2c]

1. College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, P.R. China.

2a. College of Computer and Information, Zhejiang Wanli University, Zhejiang NingBo 315100, P.R. China.

2b. Faculty of Electronic and Information Engineering, Zhejiang Wanli University, Zhejiang NingBo 315100, P.R. China.

2c. College of Junior，Zhejiang Wanli University, Zhejiang Ningbo 31510, P.R.China

3. Information Security Group, Royal Holloway, University of London,UK.

Abstract: Fully homomorphic encryption is faced with two problems now. One is candidate fully homomorphic encryption schemes are few. Another is that the efficiency of fully homomorphic encryption is a big question. In this paper, we propose a fully homomorphic encryption scheme based on LWE, which has better key size. Our main contributions are: (1) According to the binary-LWE recently, we choose secret key from binary set and modify the basic encryption scheme proposed in Linder and Peikert in 2010. We propose a fully homomorphic encryption scheme based on the new basic encryption scheme. We analyze the correctness and give the proof of the security of our scheme. The public key, evaluation keys and tensored ciphertext have better size in our scheme. (2) Estimating parameters for fully homomorphic encryption scheme is an important work. We estimate the concert parameters for our scheme. We compare these parameters between our scheme and Bra12 scheme. Our scheme have public key and private key that smaller by a factor of about $\log q$ than in Bra12 scheme. Tensored ciphertext in our scheme is smaller by a factor of about $\log^2 q$ than in Bra12 scheme. Key switching matrix in our scheme is smaller by a factor of about $\log^3 q$ than in Bra12 scheme.

.

Key words: Fully Homomorphic Encryption; public key encryption; Learning with error; concert parameters

I. INTRODUCTION

Fully homomorphic encryption (FHE) can compute arbitrary function on encrypted data without using secret key. This powerful primitive has a myriad of potential applications such as private cloud computing. Since Gentry

proposed the first FHE scheme [1], some schemes based on different hardness assumptions have been proposed [1, 2, 3, 4, 5, 6, 7] and have developed some techniques to improve efficiency [8, 9].

The efficiency of FHE has been the big question following its invention, which hinder application of FHE in practical. Specially, the size of key in FHE scheme is big. A FHE scheme based on LWE not only includes public key and private key but also includes some evaluation keys. For an L-leveled FHE scheme, there are L evaluation keys. Each evaluation key is a $(n+1)^2\lceil \log q \rceil \times (n+1)$ matrix. A public key is at least $2n\log q \times (n+1)$ matrix. Clearly, these matrixes are high dimension, which not only need a lot of space to store but also affect the efficiency of computation.

Recently, there is a variant of LWE problem called binary-LWE. It means the secret key in LWE are chosen uniformly from the binary set $\{0,1\}^n$ or $\{-1,0,1\}^n$. Both papers [10, 11] recently show that binary-LWE is hard, but those results require increasing the parameter n to approximately $n\log n$. But Bai and Galbraith in paper [12] show one can use binary-LWE with parameter $n\log(\log n)$, which is more than sufficient. This is much smaller and even the increasing dimension cannot cause any impact on the application. The paper [13] also has similar result.

The goal of this paper is to construct a FHE scheme with better key size. The style of the basic encryption scheme our scheme builds on is different from previous works [3, 4, 5]. Previous works is based on the Regev's encryption scheme in [15] to construct FHE scheme, which choose a random set uniformly and add these LWE samples according to the random set. In our basic encryption scheme, we choose LWE samples from Gaussian distribution and add Gaussian error to it, which result in that the number of LWE samples decrease from $2n\log q$ to $n+1$. The proof for our scheme uses the LWE assumption twice, which is different with prior LWE-based scheme involve a statistical arguments, but this requires larger keys. In addition, in order to achieve homomorphic property, we choose the secret key from $\{0,1\}^n$ rather than using binary decomposition for secret key as in Brakerski's scheme proposed in 2012 (Bra12)[5]. It results in that our scheme has the smaller tensored ciphertext and key switching matrix.

We note that our scheme and Bra12 scheme have the similar noise growth, but our scheme is different with Bra12 scheme. On the one hand, both FHE scheme build on the different basic encryption scheme. Our FHE scheme build on the Linder and Peikert's encryption scheme (LP10) proposed in [14], while Bra12 scheme build on the Regev's encryption scheme in [15]. On the other hand, we take the different method to reduce the noise. We do not use binary decomposition for secret key to reduce the noise but choose the secret key from $\{0,1\}^n$. Our

scheme have public key and private key that smaller by a factor of about log$q$ than in Bra12 scheme. Tensored ciphertext in our scheme is smaller by a factor of about $\log^2 q$ than in Bra12 scheme. Key switching matrix in our scheme is smaller by a factor of about $\log^3 q$ than in Bra12 scheme. It is most important that our FHE scheme is more space efficient than the FHE schemes based on LWE commonly known in the literature. Not just than Bra12 scheme. The smaller key come from the different style of the basic encryption scheme.

Estimating parameters for FHE scheme is an important work. We estimate the concert parameters for our scheme. These parameters include circuit depth L, dimension n, modulus q and Gaussian parameter r. From these parameters, we can obtain public key size, ciphertext size, the size of tensored ciphertext for multiplication and the size of key switching matrix. We compare the size of these parameters between our scheme and Bra12 scheme.

This paper is organized as follows. Section 2 defines notational conventions, introduces the LWE assumption and defines homomorphic encryption and its related terms. Section 3 describes the basic encryption scheme. Section 4 defines homomorphic addition and homomorphic multiplication so that we achieve homomorphic property for the basic encryption scheme. Section 5 describes a FHE scheme. Section 6 analyzes the noise in homomorphic addition and homomorphic multiplication, which show it is possible to achieve a leveled FHE scheme. Section 7 gives the parameters property and concert parameters.

II. PRELIMINARIES

2.1 Basic Notation

We We use $\lfloor x \rceil$ to indicate rounding x to the nearest integer, and $\lfloor x \rfloor$, $\lceil x \rceil$ (for $x \geqslant 0$) to indicate rounding down or up. When q is not a power of two, we will use $\lceil \log q \rceil$ to denote $1+\lfloor \log q \rfloor$.For an integer q, we define the set $\mathbb{Z}_q = (-q/2, q/2] \cap \mathbb{Z}$. For any $x \in \mathbb{Z}$, let $y=[x]_q$ denote the unique value $y \in (-q/2, q/2]$. $x \leftarrow \mathcal{D}$ means that x is a sample from a distribution $\mathcal{D}$. We define B-bounded distributions as ones whose magnitudes never exceed B.

For two vectors $\boldsymbol{v}$, $\boldsymbol{u}$ of dimension n, its inner product $<\boldsymbol{v}, \boldsymbol{u}>$ is defined as $<\boldsymbol{v}, \boldsymbol{u}>= \boldsymbol{v}^{\mathrm{T}} \bullet \boldsymbol{u}$. The tensor product of two vectors $\boldsymbol{v}$, $\boldsymbol{u}$ of dimension n, denoted $\boldsymbol{v} \otimes \boldsymbol{u}$, is the n^2 dimensional vector containing all elements of the form $\boldsymbol{v}[i]\boldsymbol{u}[j]$. Note that $<\boldsymbol{v} \otimes \boldsymbol{u}, \boldsymbol{x} \otimes \boldsymbol{y}> = <\boldsymbol{v}, \boldsymbol{x}> \bullet <\boldsymbol{u}, \boldsymbol{y}>$.

2.2 Learning with Error (LWE)

The learning with errors (LWE) problem was introduced by Regev [15] as a generalization of the well-known

"learning parity with noise" problem, to larger moduli. This problem was later generalized as the ring learning with errors (RLWE) problem by Lyubaskevsky, Peikert and Regev [16].

The LWE problem is parameterized by a dimension $n \geqslant 1$ and integer modulus $q \geqslant 2$, as well as a probability distribution χ over $\mathbb{Z}$ or $\mathbb{Z}_q$. For a vector $\boldsymbol{s} \in \mathbb{Z}_q^n$, the LWE distribution $\mathcal{A}_{\boldsymbol{s},\chi}$ is obtained by choosing a vector $\boldsymbol{a}$ from $\mathbb{Z}_q^n$ uniformly at random and a noise term $e \leftarrow \chi$, and outputting $(\boldsymbol{a}, b = <\boldsymbol{a}, \boldsymbol{s}> + e \bmod q) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. The search-LWE problem is, given an arbitrary number of independent samples $(\boldsymbol{a}_i, b_i) \leftarrow \mathcal{A}_{\boldsymbol{s},\chi}$, to find $\boldsymbol{s}$. We are primarily interested in the decision-LWE (DLWE) problem for cryptographic applications. The decision-LWE problem is to distinguish with some non-negligible advantage between the two cases. One case is any desired number of independent samples $(\boldsymbol{a}_i, b_i) \leftarrow \mathcal{A}_{\boldsymbol{s},\chi}$. Another case is the same number of independent samples drawn from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

There are two kinds of reductions such as quantum reduction [15] and classical reduction [10, 17] from worst-case lattice problems to the LWE problem. In addition, if the vector $\boldsymbol{s}$ is sampled from the distribution χ, then the LWE problem is still hard.

For a lattice Λ and a positive real $r > 0$, we denote $D_{\Lambda,r}$ as the discrete Gaussian distribution over Λ and parameter r, which is the probability distribution that assigns mass proportional to $\exp(-\pi \|\boldsymbol{x}\|^2 / s^2)$ to each point $\boldsymbol{x} \in \Lambda$. For $\Lambda = \mathbb{Z}^n$, the discrete Gaussian $D_{\mathbb{Z}^n,r}$ is simply the product distribution of n independent copies of $D_{\mathbb{Z},r}$. We will need two tail bounds on discrete Gaussians that come from paper [18, 19].

Lemma 2.1 . Let $c \geq 1$ and $C = c \cdot \exp(\frac{1-c^2}{2}) < 1$. Then for any real r > 0 and any integer $n \geq 1$, we have

$$\Pr\left[\left\|D_{\mathbb{Z}^n,r}\right\| \geq c \cdot \frac{1}{\sqrt{2\pi}} \cdot r\sqrt{n}\right] \leq C^n.$$

Lemma 2.2 . For any real $r > 0$, $T > 0$, and $\boldsymbol{x} \in \mathbb{R}^n$, we have $\Pr\left[\left|< \boldsymbol{x}, D_{\mathbb{Z}^n,r} >\right| \geq T \cdot r\|\boldsymbol{x}\|\right] < 2\exp(-\pi \cdot T^2)$.

2.3 Leveled Homomorphic Encryption

A homomorphic encryption scheme HE=(Keygen, Enc, Dec, Eval) includes a quadruple of PPT algorithms. For the definition of full homomorphic encryption, readers can refer to these papers [1, 5].

At present, there are two types of fully homomorphic encryption schemes. One is leveled fully homomorphic encryption schemes, in which the parameters of a scheme depend on the depth of the circuits that the scheme can

evaluate. In this case any circuit with a polynomial depth can be evaluated. The other is pure fully homomorphic encryption schemes, which can be built from a leveled fully homomorphic encryption scheme with the assumption of circular security. A pure fully homomorphic encryption scheme can evaluate the circuit whose depth is not limited. The following definitions are taken from [5].

Definition 2 (*L*-homomorphism). A scheme HE is *L*-homomorphic, for $L=L(\lambda)$, if for any depth *L* arithmetic circuit *f* (over GF(2)) and any set of inputs $m_1,\ldots,m_l$, it holds that

$\Pr[\mathrm{HE.Dec}_{sk}(\mathrm{HE.Eval}_{evk}(f,c_1,\ldots,c_l)) \neq f(m_1,\ldots,m_l)] = \mathrm{negl}(\lambda)$,

where $(pk, evk, sk) \leftarrow \mathrm{HE.Keygen}(1^\lambda)$ and $c_i \leftarrow \mathrm{HE.Enc}_{pk}(m_i)$.

Definition 3 (compactness, full homomorphism and leveled full homomorphism). A homomorphic scheme is compact if its decryption circuit is independent of the evaluated function. A compact scheme is fully homomorphic if it is *L*-homomorphic for any polynomial *L*. The scheme is leveled fully homomorphic if it takes 1^L as additional input in key generation.

III. THE BASIC ENCRYPTION SCHEME

We take the encryption scheme proposed by Lindner and Peikert [14] as building blocks. Their scheme is an abstract system described by Daniele and Oded [20]. We instantiate their scheme in here and do a little change. The secret key was chosen from a Gaussian distribution $D_{\mathbb{Z}^n,r}$ in original Lindner and Peikert's encryption scheme. However, we choose the secret key from the set $\{0,1\}^n$ in this basic encryption scheme in order to improve the efficiency of fully homomorphic scheme we describe late. The security of this scheme is still hard under the assumption of binary-LWE that means the secret vectors are chosen uniformly from the set $\{0,1\}^n$ or $\{-1,0,1\}^n$. Recently, both paper [10, 11] give reductions that ensure the hardness of binary-LWE.

An integer modulus $q \geq 2$, integer dimension $n = n' \cdot \log(\log n')$ where n' is the dimension of LWE problem in paper [15], and a Gaussian distribution $D_{\mathbb{Z}^n,r}$ denoted as χ^n , which relate to the underlying binary-LWE problem. In order for the smallest public keys, a uniformly random public matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times n}$ can be generated by a trusted source, and is used by all parties in the system. If the trusted source is not got in the system, **A** may be generated in the step of key generation and as part of public key. The basic encryption scheme is described as follows.

E.SecretKeygen(1^n): Choose uniformly $\boldsymbol{s}' \leftarrow \{0,1\}^n$. Output $\mathbf{sk} = \boldsymbol{s} \leftarrow (1, \boldsymbol{s}')$.

E.PublicKeygen(**A,** $\boldsymbol{s}$): Choose $\boldsymbol{e}_1 \leftarrow \chi^n$, and let $\boldsymbol{p} = \boldsymbol{e}_1 - \mathbf{A} \cdot \boldsymbol{s}' \in \mathbb{Z}_q^n$. Set the public key $\mathbf{pk} = \boldsymbol{p}$.

E.Enc(**A, pk**, m): To encrypt a message $m \in \{0,1\}$, sample $\boldsymbol{e}_2 \leftarrow \chi^n$, $\boldsymbol{e}_3 \leftarrow \chi^n$, and $e_4 \leftarrow \chi$, and output $\boldsymbol{c} \leftarrow (\boldsymbol{p}^{\mathrm{t}} \cdot \boldsymbol{e}_2 + e_4 + \left\lfloor \frac{q}{2} \right\rfloor \cdot m, \mathbf{A}^{\mathrm{t}} \cdot \boldsymbol{e}_2 + \boldsymbol{e}_3) \in \mathbb{Z}_q^{n+1}$.

E.Dec(**sk**, $\boldsymbol{c}$): Output $m \leftarrow \lfloor \frac{2}{q} [< \boldsymbol{c}, \boldsymbol{s} >]_q \rceil \bmod 2$.

To illustrate the correctness of this basic encryption, we analyze the noise magnitude at encryption and decryption.

Lemma 3.1 (encryption noise). Let q, n, $\mathbf{A}$, $|\chi| \le B$ be parameters in above encryption scheme. The secret key $\boldsymbol{s}$ and public key $\boldsymbol{p}$ are generated from **E.SecretKeygen**(1^n) and **E.PublicKeygen**(**A,** $\boldsymbol{s}$). Set $\boldsymbol{c} \leftarrow$**E.Enc**(**A, pk**, m). Then for some e with $|e| \leqslant nB^2 + nB + B$, it holds that

$$< \boldsymbol{c}, \boldsymbol{s} > = \left\lfloor \frac{q}{2} \right\rfloor \cdot m + e \quad (mod\ q).$$

Proof. By definition

$$< \boldsymbol{c}, \boldsymbol{s} > = \boldsymbol{p}^{\mathrm{t}} \cdot \boldsymbol{e}_2 + e_4 + \left\lfloor \frac{q}{2} \right\rfloor \cdot m + (\mathbf{A}^{\mathrm{t}} \cdot \boldsymbol{e}_2 + \boldsymbol{e}_3)^{\mathrm{t}} \cdot \boldsymbol{s}' \quad (mod\ q)$$

$$= \boldsymbol{e}_2^{\mathrm{t}} \cdot (\boldsymbol{e}_1 - \mathbf{A} \cdot \boldsymbol{s}') + e_4 + \left\lfloor \frac{q}{2} \right\rfloor \cdot m + \boldsymbol{e}_2^{\mathrm{t}} \cdot \mathbf{A} \cdot \boldsymbol{s}' + \boldsymbol{e}_3^{\mathrm{t}} \cdot \boldsymbol{s}' \quad (mod\ q)$$

$$= \left\lfloor \frac{q}{2} \right\rfloor \cdot m + \boldsymbol{e}_2^{\mathrm{t}} \cdot \boldsymbol{e}_1 + \boldsymbol{e}_3^{\mathrm{t}} \cdot \boldsymbol{s}' + e_4 \quad (mod\ q).$$

Since $|\chi| \le B$, we have $| \boldsymbol{e}_2^{\mathrm{t}} \cdot \boldsymbol{e}_1 + \boldsymbol{e}_3^{\mathrm{t}} \cdot \boldsymbol{s}' + e_4 | \leqslant nB^2 + nB + B$ and the lemma follows.

The correctness of decryption is decided by the noise magnitude in ciphertexts. The bound of noise magnitude is $\left\lfloor \frac{q}{4} \right\rfloor$, which is as same as Regev's encryption scheme in [15].

Lemma 3.2 (decryption noise). Let $\boldsymbol{c} \in \mathbb{Z}_q^{n+1}$ and $\boldsymbol{s} \in \{0,1\}^n$ be two vectors such that

$$< \boldsymbol{c}, \boldsymbol{s} > = \left\lfloor \frac{q}{2} \right\rfloor \cdot m + e \quad (mod\ q),$$

where $m \in \{0,1\}$. If $|e| < \left\lfloor \frac{q}{4} \right\rfloor$, then we have $m \leftarrow$ **E.Dec**($\boldsymbol{s}$, $\boldsymbol{c}$).

This proof is as same as the proof in Regev's encryption scheme and is omitted.

Lemma 3.3 (security). The above encryption scheme is CPA-security, assuming the hardness of decision-LWE with parameters n, q, χ for: (i) secret sample from binary secret, and (ii) secret sample from a Gaussian distribution.

Proof. For any plaintext bit m encrypted by the encryption scheme, the adversary's view consists of ($\mathbf{A}$, $\boldsymbol{p}$, $\boldsymbol{c}$), where $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$ is uniformly random, $\boldsymbol{p} \leftarrow$**E.PublicKeygen**, and $\boldsymbol{c} \leftarrow$**E.Enc**(**A,** $\boldsymbol{p}$, m)**.** It is sufficient to show that the triples ($\mathbf{A}$, $\boldsymbol{p}$, $\boldsymbol{c}$) in the IND-CPA attack is computationally indistinguishable from uniformly random ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}$*), where $\boldsymbol{p}^* \in \mathbb{Z}_q^n$ and $\boldsymbol{c}^* \in \mathbb{Z}_q^{n+1}$. First, we show it is computationally indistinguishable between ($\mathbf{A}$, $\boldsymbol{p}$) and ($\mathbf{A}$, $\boldsymbol{p}$*). Since $\boldsymbol{p} = \boldsymbol{e}_1 - \mathbf{A} \cdot \boldsymbol{s}'$, where $\boldsymbol{s}'$ is chosen uniformly from $\{0,1\}^n$ and $\boldsymbol{e}_1$ is drawn from a Gaussian distribution χ^n, ($\mathbf{A}$, $\boldsymbol{p}$) is computationally indistinguishable from uniformly random ($\mathbf{A}$, $\boldsymbol{p}$*) under the assumption (i) in the lemma statement. We say the adversary's view ($\mathbf{A}$, $\boldsymbol{p}$, $\boldsymbol{c}$) is computationally indistinguishable from uniformly random ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}$). Since $\boldsymbol{c}$ is computationally indistinguishable from $\boldsymbol{c}'$, where $\boldsymbol{c}' \leftarrow$**E.Enc**(**A,** $\boldsymbol{p}$*, m)**,** we can replace $\boldsymbol{c}$ with $\boldsymbol{c}'$ in the triples ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}$). We have ($\mathbf{A}$, $\boldsymbol{p}$, $\boldsymbol{c}$) is computationally indistinguishable from ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}'$). Second, we show it is computationally indistinguishable between ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}'$) and uniformly random ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}$*). Let $\mathrm{A}' = (\mathbf{A}, \boldsymbol{p}^*)$. Since $\boldsymbol{c}'$ $= ((\mathrm{A}')^{\mathrm{t}} \cdot \boldsymbol{e}_2 + \begin{bmatrix} \boldsymbol{e}_3 \\ e_4 \end{bmatrix}) + \begin{bmatrix} 0 \\ \lfloor q/2 \rfloor \cdot m \end{bmatrix}$, where A' is uniform and $\boldsymbol{e}_2$, $\boldsymbol{e}_3$, e_4 are drawn from a Gaussian distribution, we have ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}'$) is computationally indistinguishable from ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}$*) under assumption (ii) in the lemma statement. Therefore, it is computationally indistinguishable between the adversary's view ($\mathbf{A}$, $\boldsymbol{p}$, $\boldsymbol{c}$) and uniformly random ($\mathbf{A}$, $\boldsymbol{p}$*, $\boldsymbol{c}$*).

IV. HOMOMORPHIC OPERATION

Suppose $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ under the secret key $\boldsymbol{s}$ encrypt m_1 and m_2 in that $< \boldsymbol{c}_i, \boldsymbol{s} > = \left\lfloor \frac{q}{2} \right\rfloor \cdot m_i + e_i \ (mod\ q) = \left\lfloor \frac{q}{2} \right\rfloor \cdot m_i + e_i +$ $k_i q$ for small e_i. If the ciphertext $\boldsymbol{c}$ resulted from addition or multiplication of two ciphertext $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ can hold $< \boldsymbol{c}, \boldsymbol{s} > = \left\lfloor \frac{q}{2} \right\rfloor \cdot (m_1 + m_2) + e \ (mod\ q)$ or $< \boldsymbol{c}, \boldsymbol{s} > = \left\lfloor \frac{q}{2} \right\rfloor \cdot (m_1 \cdot m_2) + e \ (mod\ q)$ for small e, we say that addition or

multiplicative homomorphism could be achieved. So this structure like $\left\lfloor \frac{q}{2} \right\rfloor \cdot m_i + e_i$ is most important in homomorphic operation, we call $\left\lfloor \frac{q}{2} \right\rfloor \cdot m_i + e_i$ as invariant structure. For the basic encryption scheme described above, homomorphic addition can be achieved directly, but homomorphic multiplication cannot be achieved directly. We need to construct the ciphertext of homomorphic multiplication to satisfy invariant structure.

4.1 Homomorphic Addition

By definition

$$< \boldsymbol{c}_1+\boldsymbol{c}_2 , \boldsymbol{s} > = < \boldsymbol{c}_1 , \boldsymbol{s} > + < \boldsymbol{c}_2 , \boldsymbol{s} > = \left\lfloor \frac{q}{2} \right\rfloor \bullet (m_1 + m_2) + e_1+e_2 \ (\mathrm{mod}\ q).$$

The noise magnitude e_1+e_2 increase a little as previous fully homomorphic encryption scheme. If the noise magnitude is small, namely, $| e_1+e_2 | < \left\lfloor \frac{q}{2} \right\rfloor / 2$, the ciphertext $\boldsymbol{c}_1+\boldsymbol{c}_2$ can be decrypted correctly. It means the sum of ciphertext encrypts the sum of the message.

4.2 Homomorphic Multiplication

Since the basic encryption scheme itself does not has the property of homomorphic multiplication, we require to define the representation of the ciphertext resulted from homomorphic multiplication so as to achieve he property of homomorphic multiplication. We define the ciphertext for multiplication as $\lfloor \frac{2}{q} \cdot (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2) \rceil$ like definition in paper [5], which can be decrypted using a tensored secret key $\boldsymbol{s} \otimes \boldsymbol{s}$. The reasons of this definition are as follows.

Let an error $\boldsymbol{r} = \lfloor \frac{2}{q} \cdot (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2) \rceil - \frac{2}{q} \bullet (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2)$. By definition

$$< \lfloor \frac{2}{q} \cdot (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2) \rceil , \boldsymbol{s} \otimes \boldsymbol{s} > = \ < \frac{2}{q} \bullet (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2), \ \boldsymbol{s} \otimes \boldsymbol{s} > + < \boldsymbol{r} , \boldsymbol{s} \otimes \boldsymbol{s} >$$

$$= \left\lfloor \frac{q}{2} \right\rfloor \bullet m_1 m_2 + m_1 e_2 + m_2 \ e_1 + 2(e_1 k_2 + k_1 e_2) + q \bullet (m_1 k_2 + k_1 m_2 + 2 k_1 k_2) - [q]_2 \bullet (m_1 k_2 + k_1 m_2) + \frac{[q]_2}{q} \bullet (m_1 e_2 - m_2$$

$$e_1 - \left\lfloor \frac{q}{2} \right\rfloor \bullet (m_1 m_2)) + \frac{2}{q} \bullet e_1 e_2 + < \boldsymbol{r} , \boldsymbol{s} \otimes \boldsymbol{s} > \qquad (1)$$

$$= \left\lfloor \frac{q}{2} \right\rfloor \bullet m_1 m_2 + e_1^{mult} + e_2^{mult} ,$$

where $e_1^{mult} = m_1e_2 + m_2\ e_1 + 2(e_1k_2 + k_1e_2) + q\bullet(m_1k_2 + k_1m_2 + 2k_1k_2) - [q]_2\bullet(m_1k_2 + k_1m_2) + \frac{[q]_2}{q}\bullet(m_1e_2 - m_2\ e_1 - \left\lfloor \frac{q}{2} \right\rfloor \bullet(m_1m_2)) + \frac{2}{q}\bullet e_1e_2$ and $e_2^{mult} = |< \boldsymbol{r}, \boldsymbol{s} \otimes \boldsymbol{s} >|$.

The invariant structure appears in the above equation (1). If $|e_1^{mult} + e_2^{mult}| < \left\lfloor \frac{q}{2} \right\rfloor / 2$, the tensored ciphertext for multiplication $\frac{2}{q}\bullet(\boldsymbol{c}_1 \otimes \boldsymbol{c}_2)$ can be decrypted correctly. It means multiplicative homomorphism is achieved by the above definition for multiplication.

4.3 Key Switching

Even though the tensored ciphertext for multiplication enable us to achieve the property of homomorphic multiplication, there is a problem that the dimension of the ciphertext increases from $n+1$ to $(n+1)^2$ after a homomorphic multiplication. We use the key switching technique to solve this problem. Key switching consists of two procedures, namely **SwitchKeyGen** ($\boldsymbol{s}_1$, $\boldsymbol{s}_2$,n_1 , n_2 , q) and **SwitchKey**(τ , $\boldsymbol{c}_1$, n_1 , n_2 , q). The goal of Key switching is to transform a ciphertext $\boldsymbol{c}_1$ under a secret key $\boldsymbol{s}_1$ to a new ciphertext $\boldsymbol{c}_2$ under a secret key $\boldsymbol{s}_2$, in which $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ encrypt the same message. If the dimension of $\boldsymbol{c}_2$ and $\boldsymbol{s}_2$ is lower than the dimension of $\boldsymbol{c}_1$ and $\boldsymbol{s}_1$, the dimension of the key and ciphertext vectors is reduced by key switching.

SwitchKeyGen($\boldsymbol{s}_1 \in \mathbb{Z}_q^{n_1}$， $\boldsymbol{s}_2 \in \mathbb{Z}_q^{n_2}$)：

（1） Run **A**←**E.PublicKeygen**($\boldsymbol{s}_2$) for $N = n_1 \bullet \lceil \log q \rceil$， namely **A**=[$\boldsymbol{b}$ | −A′].

（2） Set **B**← [(Powerof2($\boldsymbol{s}_1$) + $\boldsymbol{b}$) | −A′]， which means to add the Powerof2($\boldsymbol{s}_1$) $\in \mathbb{Z}_q^N$ to −A′ 's first column and add $\boldsymbol{b}$ to −A′ 's second column. Output $\boldsymbol{\tau}_{s_1 \to s_2}$ =**B**.

SwitchKey($\boldsymbol{\tau}_{s_1 \to s_2}$, $\boldsymbol{c}_1$)： Output $\boldsymbol{c}_2$ = BitDecomp($\boldsymbol{c}_1$)$^{\mathrm{T}}\bullet$**B** $\in \mathbb{Z}_q^{n_2}$.

Key switching is essentially the product of a high dimension vector and a high dimension matrix. Next, we describe the correctness of key switching, namely the decryption of the new ciphertext can preserve correctness. The proof is based on the definition (see [4]).

Lemma 4.1 Let $\boldsymbol{s}_1$, $\boldsymbol{s}_2$, q， **A**， **B** = $\boldsymbol{\tau}_{s_1 \to s_2}$ be parameters as described in **SwitchKeyGen,** and have **A**$\bullet\boldsymbol{s}_2 = \boldsymbol{e}_2 \in \mathbb{Z}_q^N$.

Let $\boldsymbol{c}_1 \in \mathbb{Z}_q^N$ and $\boldsymbol{c}_2$ ← **SwitchKey**($\boldsymbol{\tau}_{s_1 \to s_2}$, $\boldsymbol{c}_1$) . Then, < $\boldsymbol{c}_2$, $\boldsymbol{s}_2$> = < BitDecomp($\boldsymbol{c}_1$) , $\boldsymbol{e}_2$> + < $\boldsymbol{c}_1$, $\boldsymbol{s}_1$> (mod q).

V. A HOMOMORPHIC ENCYPTION SCHEME.

5.1 A Leveled Homomorphic Encryption Scheme

We construct a leveled homomorphic encryption scheme based on the basic encryption scheme described in section 3 and homomorphic property described in section 4. For a leveled homomorphic encryption scheme, different level has different secret key in circuit. Homomorphic operations are just to be performed from level L to 1. The first level is level L, and the last level is level 0. The level 0 is only used to switch key. After each homomorphic operation, we need to transform the result to enter the next level of circuit. Before each homomorphic operation, it requires that the two ciphertext have the same secret key (namely, the same level). Otherwise, we need transform the higher level ciphertext between the two ciphertext to the same level with another lower level ciphertext. The function of **FHE.RefreshNextLevel** is to do it. We note the operation of key switching is just used for tensored ciphertext. Thus the ciphertext of normal dimension need to tensor with a trivial ciphertext (1,0,…,0) before using key switching.

FHE.Setup(λ , L): Input the security parameter λ and the circuit level L, output the noise distribution χ , and the dimension n. Note that χ and n are as same as in the above basic encryption scheme. If there is a trusted source in the system, all parties in the system would the trusted source to generate a uniformly random public matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$. If not, **A** may be generated in the step of key generation and as part of public key.

FHE.KeyGen(n, L)：For $i = L$ down to 0, do the following:：

(1) Run $\boldsymbol{s}_i \leftarrow$**E.SecretKeygen**($1^n$) . Let **sk**=$\{\boldsymbol{s}_i\}$.

(2) When $i = L$ do this step. Run $\boldsymbol{p}_L \leftarrow$**E.PublicKeygen**(**A,** $\boldsymbol{s}_L$). Let $\mathbf{pk}_1 = \{ \boldsymbol{p}_L \}$.

(3) Set $\boldsymbol{s}_i' \leftarrow \boldsymbol{s}_i \otimes \boldsymbol{s}_i \in (0,1)^{(n+1)^2}$.（Omit this step when i=0.）

(4) Run $\boldsymbol{\tau}_{\boldsymbol{s}_i \to \boldsymbol{s}_{i-1}} \leftarrow$ **SwitchKeyGen**($\boldsymbol{s}_i'$, $\boldsymbol{s}_{i\text{-}1}$).（Omit this step when i=0.）Let $\mathbf{pk}_2 = \{ \boldsymbol{\tau}_{\boldsymbol{s}_i \to \boldsymbol{s}_{i-1}} \}$.

Then output **sk**=$\{\boldsymbol{s}_i\}$ and **pk**=（$\mathbf{pk}_1$， $\mathbf{pk}_2$）.

FHE.Enc($\mathbf{pk}_1$, m)：Take a message $m \in \{0,1\}$. Run **E.Enc**($\boldsymbol{p}_L$, m).

FHE.Dec(**sk**, $\boldsymbol{c}_i$)：Assume that $\boldsymbol{c}_i$ is a ciphertext under the secret key $\boldsymbol{s}_i$. Run **E.Dec**(**sk**, $\boldsymbol{c}_i$).

FHE.Add($\mathbf{pk}_2$, $\boldsymbol{c}_1$, $\boldsymbol{c}_2$)：Do the following steps.

(1) If ciphertexts $\boldsymbol{c}_1$, $\boldsymbol{c}_2$ has the same secret key $\boldsymbol{s}_i$, first compute $\boldsymbol{c}_3 \leftarrow \boldsymbol{c}_1 + \boldsymbol{c}_2$. In order to provide an output that

corresponds to the next level key s_{i-1} and not s_i, we call **FHE.RefreshNextLevel** to do it. Output $c_{add} \leftarrow$ **FHE.RefreshNextLevel**$(i, c_3, \tau_{s_i \to s_{i-1}}) \in \mathbb{Z}_q^{n+1}$.

(2) If ciphertexts c_1, c_2 has the different secret key, we input the higher level ciphertext between c_1 and c_2 to **FHE.RefreshNextLevel**. We can repeat to call **FHE.RefreshNextLevel** till the output from **FHE.RefreshNextLevel** has the same secret key with the lower level ciphertext between c_1 and c_2 . Then go to step (1).

FHE.Mult($\mathbf{pk}_2$, c_1, c_2)：Do the following steps.

(1) If ciphertexts c_1, c_2 has the same secret key s_i, first compute $c_3 \leftarrow \lfloor \frac{2}{q} \cdot (c_1 \otimes c_2) \rceil$ that corresponds to s_i' . Then output $c_{mult} \leftarrow$ **SwitchKey**($\tau_{s_i \to s_{i-1}}$, c_3).

(2) If ciphertexts c_1, c_2 has the different secret key, what we do as same as the step (2) in **FHE.Add**($\mathbf{pk}_2$, c_1, c_2).

FHE.RefreshNextLevel(i, c, $\tau_{s_i \to s_{i-1}}$): First compute $c' = c \otimes (1,0,\ldots,0)$, then output **SwitchKey**($\tau_{s_i \to s_{i-1}}$,c').

VI. NOISE ANALYSIS

Suppose ciphertext c_i under the secret key s is a fresh ciphertext, namely, $c_i \leftarrow$**E.Enc**(**A, pk**, m_i). By lemma 3.1, we have $< c_i, s > = \left\lfloor \frac{q}{2} \right\rfloor \cdot m_i + e_i \ (mod\ q)$, where $| e_i | \leqslant E = nB^2 + nB + B$. Next we analyze the noise magnitude in ciphertext after one addition or one multiplication.

6.1 Analysis for Addition

By definition

$$< c_1 + c_2, s > = < c_1, s > + < c_2, s > \pmod q$$

$$= \left\lfloor \frac{q}{2} \right\rfloor \cdot [m_1 + m_2]_2 + 2 \cdot \left\lfloor \frac{q}{2} \right\rfloor \cdot \lfloor (m_1 + m_2)/2 \rfloor + e_1 + e_2 \ (\mathrm{mod}\ q) .$$

Then we get $|e^{add}| = | 2 \cdot \left\lfloor \frac{q}{2} \right\rfloor \cdot \lfloor (m_1 + m_2)/2 \rfloor + e_1 + e_2 | \leqslant 2nB^2 + 2nB + 2B + 1$.

6.2 Analysis for Multiplication

Let c_{mult} be the output of **FHE.Mult**($\mathbf{pk}_2$, c_1, c_2) under the secret key s'. According to the result in section 4.2 and Lema 4.1, we have

$$< \boldsymbol{c}_{\text{mult}}, \boldsymbol{s}' > = < \lfloor \frac{2}{q} \cdot (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2) \rceil, \boldsymbol{s} \otimes \boldsymbol{s} > + < \text{BitDecomp}(\lfloor \frac{2}{q} \cdot (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2) \rceil), \boldsymbol{e} >$$

$$= \left\lfloor \frac{q}{2} \right\rfloor \cdot m_1 m_2 + e_1^{mult} + e_2^{mult} + e_3 \ (\text{mod } q),$$

where $e_1^{mult} = m_1 e_2 + m_2 e_1 + 2(e_1 k_2 + k_1 e_2) - [q]_2 \bullet (m_1 k_2 + k_1 m_2) + \frac{[q]_2}{q} \bullet (m_1 e_2 - m_2 e_1 - \left\lfloor \frac{q}{2} \right\rfloor \cdot (m_1 m_2)) + \frac{2}{q} \cdot e_1 e_2$, e_2^{mult} $= | < \boldsymbol{r}, \boldsymbol{s} \otimes \boldsymbol{s} > |$ and $e_3 = < \text{BitDecomp}(\lfloor \frac{2}{q} \cdot (\boldsymbol{c}_1 \otimes \boldsymbol{c}_2) \rceil), \boldsymbol{e} >$.

We first analyze the bound of e_1^{mult}. The magnitude of e_1^{mult} mainly depends on the term $2(e_1 k_2 + k_1 e_2)$, so we check the bound of the absolute value of k_1 (the same bound also holds for k_2):

$$| k_1 | = | < \boldsymbol{c}_1, \boldsymbol{s} > - \left\lfloor \frac{q}{2} \right\rfloor \cdot m_1 - e_1 | / q \leqslant | < \boldsymbol{c}_1, \boldsymbol{s} > | / q + 1 \leqslant (\| \boldsymbol{c}_1 \|_\infty / q) \cdot \| \boldsymbol{s} \|_1 + 1 \leqslant n + 1.$$

We have $| e_1^{mult} | \leqslant 2E(n+3)+2$. Furthermore, we have $| e_2^{mult} | = | < \boldsymbol{r}, \boldsymbol{s} \otimes \boldsymbol{s} > | \leq \| \boldsymbol{r} \|_\infty \cdot | \boldsymbol{s} \otimes \boldsymbol{s} | \leqslant (1/2) \cdot (n+1)^2$ and $\left| e_3 \right| \leq (n+1)^2 \lceil \log q \rceil \cdot B$.

Putting these together, we get the bound of noise magnitude after one multiplication for fresh ciphertext such as

$$| e_1^{mult} + e_2^{mult} + e_3 | \leqslant 2E \cdot (n+3) + (n+1)^2 \cdot (\lceil \log q \rceil \cdot B + \frac{1}{2}).$$

After we evaluate a circuit of depth L, the upper bound on the noise magnitude in resulting ciphertext is $c_1^L \cdot E + L \cdot c_1^{L-1} \cdot c_2$, where c_1=2(n+3), $c_2 = (n+1)^2 \cdot (\lceil \log q \rceil \cdot B + \frac{1}{2})$. As long as the parameters of this scheme satisfy $c_1^L \cdot E + L \cdot c_1^{L-1} \cdot c_2 < \lfloor q/4 \rfloor$, we can evaluate homomorphic operation with L multiplication. For security, the best known algorithm for LWE runs in time approximately $2^{n/\log(q/B)}$. Therefore we choose B to be polynomial in $n = \lambda$ and $q = 2^{n^\varepsilon}$ for every $\varepsilon < 1$, we can get $L \approx \log q \approx n^\varepsilon$. It means we could evaluate a circuit of polynomial depth. In short, we would have a leveled fully homomorphic encryption scheme. Thus we can obtain the following theorem.

Theorem 6.1. For every $L > 0$, there exists $\varepsilon < 1$ and a L-leveled fully homomorphic encryption scheme under LWE assumption, where $q/B \leq 2^{n^\varepsilon}$.

VII. PARAMETERS SEETING

In this section, we estimate the concert parameters for our scheme. These parameters include circuit depth L,

dimension n, modulus q and Gaussian parameter r. From these parameters, we can obtain public key size, ciphertext size, the size of tensored ciphertext for multiplication and the size of key switching matrix. Since Bra12 scheme is also a leveled fully homomorphic encryption scheme without modulus switching and homomorphic property of Bra12 scheme is similar with our scheme, we compare these parameters between our scheme and Bra12 scheme.

7.1 Parameters Property

We first list some properties of our scheme and Bra12 scheme in table 1. All sizes are in bits. We note the number of LWE sample in Bra12 scheme we take $N=2n\log q$. The key switching matrix is a kind of public key, which is actually evaluation key for homomorphic operation. There are L key switching matrixes in L-leveled fully homomorphic encryption scheme, which need a lot of space to store.

Table 1. Some properties of our scheme and Bra12 scheme

	Public key	**Public key**	**Private key**
Our scheme	$n\log q$	$(n+1)^2\log q$	$n+1$
Bra12 scheme	$2n\log^2 q$	$2n(n+1)\log^3 q$	$(n+1)\log q$

	Ciphertext	**Tensored ciphertext**	**Key switching**
Our scheme	$(n+1)\log q$	$(n+1)^2\log q$	$(n+1)^3$
Bra12 scheme	$(n+1)\log q$	$((n+1)\lceil \log q \rceil)^2\log q$	$(n+1)^3\lceil \log q \rceil^2\log q$

From above the table 1, the advantage in our scheme is obvious for the parameters size beyond ciphertext. Specially, the public key size (include key switching matrix) improve a factor $\log^3 q$ at most.

7.2 Concert parameters

In this section, we consider the concert value of parameters. We first choose circuit depth L and dimension n. For leveled homomorphic encryption scheme, it has to decide the circuit depth L before performance computation. The dimension n is security parameter. We can choose n according to the requirement of security. Then we choose modulus q and Gaussian parameter r. Last, we can estimate the values such as public key, private key, ciphertext, tensored ciphertext and key switching matrix.

As an example parameters, we consider the case that n=128 and L=0,1,5. For a security level λ bits, we need $n > \log(q/r)\cdot(\lambda+110)/7.2$ following the analysis in paper [14], which make us obtain the function about security level and dimension. Next we state how to choose modulus q and Gaussian parameter r. Recall that the message decrypts correctly in the basic encryption scheme in section 3 if $|<\boldsymbol{e}_l, \boldsymbol{e}_r>| < \left\lfloor \frac{q}{4} \right\rfloor$, where $\boldsymbol{e}_l = \begin{bmatrix} \boldsymbol{e}_2 \\ \boldsymbol{e}_3 \\ \boldsymbol{e}_4 \end{bmatrix}$ and $\boldsymbol{e}_r = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{s}' \\ 1 \end{bmatrix}$. Each entries of $\boldsymbol{e}_l$ and $\boldsymbol{e}_r$ are independent and have distribution $D_{\mathbb{Z},r}$. By lemma 2.2, we have $\Pr\left[\left|<\boldsymbol{e}_l, D_{\mathbb{Z}^n,r}>\right| \geq T\cdot r\|\boldsymbol{e}_l\|\right] < 2\exp(-\pi\cdot T^2)$. Let $T\cdot r\|\boldsymbol{e}_l\| = \left\lfloor \frac{q}{4} \right\rfloor$, then we get $T = \lfloor q/4 \rfloor / (r\|\boldsymbol{e}_l\|)$. By lemma 2.1, there is a $c \geq 1$ such that $\|\boldsymbol{e}_l\| \leq c\cdot\frac{1}{\sqrt{2\pi}}\cdot r\sqrt{2n}$. We thus have $r^2 \geq \sqrt{2\pi}\cdot\lfloor q/4 \rfloor / (c\cdot T\sqrt{2n})$. We denote δ as the decryption error probability per bit in the basic encryption. We typically use δ=0.01, namely, keeping the error probability below 1%. By lemma 2.2, we have δ=2exp($-\pi\cdot T^2$), then $T = \sqrt{\ln(2/\delta)}/\sqrt{\pi}$. Therefore, we obtain

$$r^2 \geq \sqrt{2\pi}\cdot\lfloor q/4 \rfloor / (c\cdot\sqrt{2n\cdot\ln(2/\delta)}). \qquad (2)$$

From inequation (2), we can choose the concert values for modulus q and Gaussian parameter r if we know c. The parameter c in inequation (2) can be determined as following step. By lemma 2.1, let $\Pr[\|\boldsymbol{e}_l\| \geq c\cdot\frac{1}{\sqrt{2\pi}}\cdot r\sqrt{2n}\,] \leqslant C^n = 2^{-40}$, namely, the probability of choosing a unexpected vector $\boldsymbol{e}_l$ is at most 2^{-40}. Since $C = c\cdot\exp(\frac{1-c^2}{2})$, we can solve c from $(c\cdot\exp(\frac{1-c^2}{2}))^n = 2^{-40}$. From inequation (2), we should choose the modulus q to be just large enough so that Gaussian parameter $r \geq 6$, which result in that the discrete Gaussian $D_{\mathbb{Z}^n,r}$ approximates the continuous Gaussian D_r.

In addition, circuit depth also affects the value of modulus q. Recall the noise analysis in section 6.2. In order to decrypt correctly, for a circuit depth L, modulus q needs to satisfy

$$c_1^L\cdot E + L\cdot c_1^{L-1}\cdot c_2 < \lfloor q/4 \rfloor. \qquad (3)$$

From the inequation (2) and (3), we choose the maximum value as the value of modulus q. In table 2, we provide the sizes of modulus q for dimensions n=1024, 2048, 4096 and levels L= 0, 1, 5 in our scheme and Bra12 scheme.

Table 2. the sizes of modulus q in our scheme and Bra12 scheme.

(a) Bra12 scheme

n	1024	2048	4096
L=0	23	24	25
L=1	34	36	38
L=5	79	85	91

(b) Our scheme

n	1024	2048	4096
L=0	20	21	23
L=1	39	42	44
L=5	111	117	124

VIII. CONCLUSION

The goal of this paper is to construct a FHE scheme with better key size. The smaller key come from the different style of the basic encryption scheme and we choose secret key from binary set. We analyze the correctness and give the proof of the security of our scheme. At last, We estimate the concert parameters for our scheme. We compare these parameters between our scheme and Bra12 scheme. Our scheme have public key and private key that smaller by a factor of about $\log q$ than in Bra12 scheme. Tensored ciphertext in our scheme is smaller by a factor of about $\log^2 q$ than in Bra12 scheme. Key switching matrix in our scheme is smaller by a factor of about $\log^3 q$ than in Bra12 scheme. It is most important that our FHE scheme is more space efficient than the FHE schemes based on LWE commonly known in the literature.

Acknowledgements

The first author would like to thank for the Fund of Jiangsu Innovation Program for Graduate Education (No.CXLX12_0162), the Fundamental Research Funds for the Central Universities, and Ningbo Natural Science Foundation (No.2012A610067) and the Chinese National Scholarship fund, and also appreciate the benefit to this work from projects in science and technique of Ningbo municipal. The third author would like to thank for Ningbo Natural Science Foundation (No.2013A610071).

REFERENCES

[1] Craig Gentry. Fully Homomorphic Encryption Using Ideal Lattices [M]. Proceedings of the 41st annual ACM symposium on Theory of computing. Bethesda, MD, USA; ACM. 2009: 169-178.

[2] Marten van Dijk, Craig Gentry, Shai Halevi, Vinod Vaikuntanathan. Fully Homomorphic Encryption over the Integers [M]//GILBERT H. Advances in Cryptology – Eurocrypt 2010. Springer Berlin / Heidelberg. 2010: 24-43.

[3] Z. Brakerski, V. Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) Lwe [M]//OSTROVSKY R. 2011 Ieee 52nd Annual Symposium on Foundations of Computer Science. Los Alamitos; IEEE Computer Society. 2011: 97-106.

[4] Zvika Brakerski, Craig Gentry, Vinod Vaikuntanathan. (Leveled) Fully Homomorphic Encryption without Bootstrapping [M]. Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. Cambridge, Massachusetts; ACM. 2012: 309-325.

[5] Zvika Brakerski. Fully Homomorphic Encryption without Modulus Switching from Classical Gapsvp [M]//SAFAVI-NAINI R, CANETTI R. Advances in Cryptology – Crypto 2012. Springer Berlin Heidelberg. 2012: 868-886.

[6] Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan. On-the-Fly Multiparty Computation on the Cloud Via Multikey Fully Homomorphic Encryption [M]. Proceedings of the 44th symposium on Theory of Computing. New York, New York, USA; ACM. 2012: 1219-1234.

[7] Craig Gentry, Amit Sahai, Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based [M]//CANETTI R, GARAY J. Advances in Cryptology – Crypto 2013. Springer Berlin Heidelberg. 2013: 75-92.

[8] Craig Gentry, Shai Halevi, Nigel Smart. Fully Homomorphic Encryption with Polylog Overhead [M]//POINTCHEVAL D, JOHANSSON T. Advances in Cryptology– Eurocrypt 2012. Springer Berlin / Heidelberg. 2012: 465-482.

[9] Zvika Brakerski, Craig Gentry, Shai Halevi. Packed Ciphertexts in Lwe-Based Homomorphic Encryption [M]//KUROSAWA K, HANAOKA G. Public-Key Cryptography – Pkc 2013. Springer Berlin Heidelberg. 2013: 1-13.

[10] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, Damien Stehl, #233. Classical Hardness of Learning with Errors [M]. Proceedings of the 45th annual ACM symposium on Symposium on theory of computing. Palo Alto, California, USA; ACM. 2013: 575-584.

[11] Daniele Micciancio, Chris Peikert. Hardness of Sis and Lwe with Small Parameters [M]//CANETTI R, GARAY J. Advances in Cryptology – Crypto 2013. Springer Berlin Heidelberg. 2013: 21-39.

[12] Shi Bai, Steven D. Galbraith. Lattice Decoding Attacks on Binary Lwe [J]. IACR Cryptology ePrint Archive, 2013, 2013(839.

[13] MartinR Albrecht, Jean-Charles Faugère, Robert Fitzpatrick, Ludovic Perret. Lazy Modulus Switching for the Bkw Algorithm on Lwe [M]//KRAWCZYK H. Public-Key Cryptography – Pkc 2014. Springer Berlin Heidelberg. 2014: 429-445.

[14] Richard Lindner, Chris Peikert. Better Key Sizes (and Attacks) for Lwe-Based Encryption [M]//KIAYIAS A. Topics in Cryptology – Ct-Rsa 2011. Springer Berlin Heidelberg. 2011: 319-339.

[15] Oded Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography [M]. Proceedings of the thirty-seventh annual ACM symposium on Theory of computing. Baltimore, MD, USA; ACM. 2005: 84-93.

[16] Vadim Lyubashevsky, Chris Peikert, Oded Regev. On Ideal Lattices and Learning with Errors over Rings [M]//GILBERT H. Advances in Cryptology – Eurocrypt 2010. Springer Berlin Heidelberg. 2010: 1-23.

[17] Chris Peikert. Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem: Extended Abstract [M]. Proceedings of the 41st annual ACM symposium on Theory of computing. Bethesda, MD, USA; ACM. 2009: 333-342.

[18] W. Banaszczyk. New Bounds in Some Transference Theorems in the Geometry of Numbers [J]. Math Ann, 1993, 296(1): 625-635.

[19] W. Banaszczyk. Inequalities for Convex Bodies and Polar Reciprocal Lattices Inr N [J]. Discrete Comput Geom, 1995, 13(1): 217-231.

[20] Daniele Micciancio, Oded Regev. Lattice-Based Cryptography [M]//BERNSTEIN D, BUCHMANN J, DAHMEN E. Post-Quantum Cryptography. Springer Berlin Heidelberg. 2009: 147-191.