Institute of Advanced Engineering and Science

On Fully Homomorphic Encryption, Approximate Lattice Problem and LWE

Gu Chun-sheng*,, Wu Fang-sheng***
* School of Computer Engineering, Jiangsu Teachers University of Technology, China Changzhou 213001
** School of Computer Science and Technology, University of Science and Technology of China, China Hefei 230027

Article Info

Article history:

Received Sept 09th, 2012
Revised Sept 25th, 2012
Accepted Oct 27th, 2012

Keyword:

Fully Homomorphic Encryption
Approximate Lattice Problem
LWE
Approximate GCD
Integer Factoring

ABSTRACT

First, we introduce a new concept of approximate lattice problem (ALP), which is an extension of learning with errors (LWE) and similar to the closest vector problem in lattice. Next, we present two ALP-based public key encryption schemes. Then, we construct a fully homomorphic encryption scheme (FHE) based on approximate principal ideal lattice problem with related modulus (APIP-RM), whose security depends on the hardness of solving the APIP-RM problem. Finally, we design a new fully homomorphic encryption scheme (FHE) based on approximate lattice problem with unrelated modulus (ALP-UM), whose security relies on the hardness of solving the ALP-UM problem.

Corresponding Author:

Gu Chun-sheng,
School of Computer Engineering,
Jiangsu Teachers University of Technology,
1801 Zhongwu Main Road, Zhonglou District, Changzhou City, Jiangsu Province, China, 213001.
Email: guchunsheng@gmail.com

1. INTRODUCTION

This paper presents two new fully homomorphic encryption schemes. In the first scheme, the public key is a list of approximate multiples $\{b_i \in R\}_{i=0}^{\tau}, \tau = O(n)$ for a hidden polynomial $f \in R$, which is computed as $b_i = a_i f + 2e_i$, where a_i , e_i is the uniformly random elements over R such that $\|2e_i\|_\infty \le n$ where n is security parameter, $p = 2^{O(n)}$ an odd integer, and R a polynomial ring. The secret key is a polynomial s with 'small' coefficient such that $(f \times s) \bmod p = 0$. To encrypt a message bit x , the ciphertext is evaluated as $c = (\sum_{i \in T, T \subseteq \{0,\ldots,\tau\}} b_i + 2e + x) \bmod p$, where $\|2e\|_\infty \le n$. To obtain addition or multiplication of the messages in the ciphertexts, we simply add/multiply the ciphertexts as the addition/multiplication over R . To decrypt a ciphertext c , we compute the message bit $x = [c \times s]_p \bmod x \bmod 2$. Recall that $[z]_p$ is an integer in $(-p/2, p/2)$ throughout this paper.

In the second scheme, given a pair of matries $A \in Z_p^{n \times n}$, $T \in Z^{n \times n}$ with $\|T\|_\infty = O(1)$ and $AT = I \bmod p$, the public key is $pk = (n, p, \{b_i = (2s_i A + 2e_i) \bmod p\}_{i=0}^{\tau=O(n)})$ such that $\|s_i\|_\infty \le n/2$ and $\|e_i\|_\infty \le n/2$, where I is an identity matrix . To encrypt a message bit x , the ciphertext is $c = (\sum_{i \in T, T \subseteq \{0,\ldots,\tau\}} b_i + 2e + \vec{x}) \bmod p$, where $\|e\|_\infty \le n/2$ and $\vec{x} = \{x, 0, \ldots, 0\} \in Z^n$. To decrypt the ciphertext c , we compute the message bit $x=\left[(c,t)\right]_p \bmod 2$,where t is the first column vector of T , the first coordinate t_0 of t with $t_0 \bmod 2=1$, and (c,T_1) is the inner product of vectors. For the homomorphic operations of ciphertexts, we will give in Section 7.

1.1 Our Contribution

The main difference between our schemes and previous work is efficiency and underlying hardness assumption. The public key size is $O(n^3)$ bits, and the expansion factor of ciphertext $O(n^2)$ in our scheme, which can be improved to $O(n)$. The security of the first scheme relies on the hardness assumption of the decision version of finding an approximate principle ideal lattice problem over related modulus (APIP-RM), given a list of approximate multiples of hidden polynomial f . The security of the second scheme is based on the hardness of solving approximate lattice problem with unrelated modulus (ALP-UM).

In high level, our schemes are similar to the fully encryption scheme over the integers [1]. But the secret key in their scheme is a big odd integer, whereas the secret key in our schemes is modular lattice. Suppose the determinant p of the circulant matrix of the secret key s is a product of distinct smoothing primes, we reduce the LWE/Ring-LWE problem to its corresponding decisional ALP/APIP.

As far as we know, the approximate lattice problem does not appear among previous works, except the approximate GCD problem [1]. Our work extends AGCD to approximate lattice problem, namely, we generalize AGCD from one dimension to multiple dimensions. We think that this problem is independent of interest.

1.2 Related work

Many researchers [2-5] have worked at this open problem since Rivest, Adleman, and Dertouzos [6] studies a privacy homomorphism, Until 2009, Gentry [7] constructed the first fully homomorphic encryption using ideal lattice. In Gentry's scheme, the public key is approximately n^7 bits, the computation per gate costs $O(n^6)$ operations. Smart and Vercauteren [8] presented a fully homomorphic encryption scheme with both relatively small key $O(n^3)$ bits , ciphertext size $O(n^{1.5})$ bits and computation per gate at least $O(n^3)$ operations, which is in some sense a specialization and optimization of Gentry's scheme. Dijk, Gentry, Halevi, and Vaikuntanathan [1] constructed a fully homomorphic encryption scheme over the integers based on approximate integer GCD, and proved that its security is equivalent to the hardness of solving approximate GCD. Stehle and Steinfeld [9] improved Gentry's fully homomorphic scheme and obtained to a faster fully homomorphic scheme, with $O(n^{3.5})$ bits complexity per elementary binary addition/multiplication gate, but the hardness assumption of the scheme security in [9] is stronger than that in [1]. Gentry and Halevi [10] implemented the first fully homomorphic scheme and presented many optimizations about it.

1.3 Organization

Section 2 recalls some notations, and the definitions of lattice, learning with error and approximate lattice problem. Section 3 gives two new public key encryption schemes based on the ALP problem. Section 4 constructs a somewhat homomorphic encryption based on APIP-RM. Section 5 transforms the somewhat homomorphic encryption into fully homomorphic encryption. Section 6 analyzes the security of our scheme and discusses two possible attacks. Section 7 proposes a new fully homomorphic encryption based on ALP-UM. Section 8 concludes this paper and gives some open problems.

2. Preliminaries

2.1 Notations

Let λ be a security parameter. $k=k(\lambda)$ a power of 2, and $[k]$ a set of integers $\{0,1,...,k\}$. Let p be an integer, $R=Z[x]/(x^k+1)$, $R_p=R/pR$. For $u\in R$, $\|u\|_\infty$ denotes the infinity norm of its coefficient vector. Let $\gamma_R=k$ be the expansion factor of R , that is, $\|u\times v\|_\infty \le k\cdot\|u\|_\infty\cdot\|v\|_\infty$, where $\times$ is multiplication in R . Let $r\leftarrow_\psi S$ denote an element choosing from S by the distribution ψ . Let $A\equiv_c B$ denote computationally indistinguishing distributions by arbitrary probabilistic polynomial time algorithm.

2.2 Lattice and Learning with Error (LWE)

Given n linearly independent vectors $b_1, b_2, ..., b_m \in R^n$, the lattice is the set $L(b_1, b_2, ..., b_m) = \{\sum_{i=1}^{m} x_i b_i, x_i \in Z\}$ of all integer linear combinations of the b_i 's. We also denote by matrix B the b_i 's. In this paper, we only consider the lattice over the integers, i.e., $b_i \in Z^n$.

For the coefficient vector $\vec{u} = (u_0, u_1, ..., u_{n-1})^T$ of $u \in R$, we define its cyclic rotation $rot(\vec{u}) = (-u_{n-1}, u_0, ..., u_{n-2})^T$, and the circulant matrix $Rot(u) = (\vec{u}, rot(\vec{u}), ..., rot^{n-1}(\vec{u}))^T$. $Rot(u)$ is called the rotation basis of the ideal lattice (u). An ideal $I \subseteq R$ is principal if it only has a single generator.

Definition 2.1. (Learning With Error (LWE) [11]). Let n, p be integers related to security parameter λ, and χ a distribution over Z_p. Given a list samples (s_i, b_i) of the distribution $D_{n,p,\chi}$ over Z_p^{n+1} such that $a \leftarrow Z_p^n$, $s_i \leftarrow Z_p^n$, $e_i \leftarrow \chi$ and $b_i = <s_i, a> + e_i \bmod p$, the LWE problem $LWE_{n,p,\chi}$ is to distinguish the distribution $D_{n,p,\chi}$ from the uniform distribution over Z_p^{n+1}.

Definition 2.2. (Learning with Errors in a Ring of Integers [12]). Let k, p be integers related to security parameter λ, and χ a distribution over R_p. Given a list samples (a_i, b_i) of the distribution $D_{k,p,\chi}$ over $R_p \times R_p$ such that $a \leftarrow R_p$, $s_i \leftarrow R_p$, $e_i \leftarrow \chi$ and $b_i = s_i \times a + e_i$, the RLWE problem $RLWE_{k,p,\chi}$ is to distinguish the distribution $D_{k,p,\chi}$ from the uniform distribution over $R_p \times R_p$.

2.3 Approximate Lattice Problem

In the following, we introduce a new concept, called approximate lattice problem (ALP). Our starting point is from AGCD defined in [1]. At the same time, ALP generalizes LWE [11] as well. Indeed, ALP is to adapt from AGCD over the integers to other rings.

Definition 2.3. (Approximate-GCD over the Integers (AGCD)[1]). Given a list of approximate multiples $\{b_i = s_i a + e_i : s_i \in Z_+, e_i \in Z, |e_i| < 2^{n-1}\}$ of an odd integer a, find a.

Definition 2.4. (Approximate Lattice Problem (ALP)). Let n, m, p be integers related to security parameter λ, and χ a distribution over Z_p^m. Given a list samples b_i of the distribution $D_{n,m,p,\chi}$ over Z_p^m such that $A \leftarrow Z_p^{n\times m}$, $s_i \leftarrow Z_p^n$, $e_i \leftarrow \chi$ and $b_i = s_i A + e_i$, the ALP $ALP_{n,m,p,\chi}$ is to distinguish $D_{n,m,p,\chi}$ from the uniform distribution over Z_p^m.

Definition 2.5. (Approximate Principal Ideal Lattice Problem (APIP)). Let k, p be integers related to security parameter λ, and χ a distribution over R_p. Given a list samples b_i of the distribution $D_{k,p,\chi}$ over R_p such that $a \leftarrow R_p$, $s_i \leftarrow R_p$, $e_i \leftarrow \chi$ and $b_i = s_i \times a + e_i$, the APIP problem $APIP_{k,p,\chi}$ is to distinguish the distribution $D_{n,p,\chi}$ from the uniform distribution over R_p.

Definition 2.6. (General Approximate Lattice Problem (GALP)). Let n, k, m, p be integers related to security parameter λ, and χ a distribution over R_p^m. Given a list samples b_i of the distribution $D_{n,k,m,p,\chi}$ over R_p^m such that $A \leftarrow R_p^{n\times m}$, $s_i \leftarrow R_p^n$, $e_i \leftarrow \chi$ and $b_i = s_i A + e_i$, the GALP problem $GALP_{n,k,m,p,\chi}$ is to distinguish the distribution $D_{n,k,m,p,\chi}$ from the uniform distribution over R_p^m.

For the GALP problem, we get the concrete ALP problem if we set $k = 2$; we get the APIP problem if we set $n = 1, m = 1$.

We can also define the general approximate lattice problem over the integers without modulus.

Definition 2.7. (General Approximate Lattice Problem over the Integers (GALP-I)). Let n, k, m be integers related to security parameter λ, and χ a distribution over R^m. Given a list samples b_i of the distribution $D_{n,k,m,\chi}$ over R^m such that $A \leftarrow R^{n\times m}$, $s_i \leftarrow R^n$, $e_i \leftarrow \chi$ and $b_i = s_i A + e_i$, the GALP problem $GALP_{n,k,m,\chi}$ is to distinguish the distribution $D_{n,k,m,\chi}$ from the uniform distribution over R^m.

3. Public Key Schemes Based on ALP

In this section, we first present new trapdoor functions. Then, we construct two public key schemes based on the ALP problem by using our trapdoor functions.

3.1 Trapdoor Functions

For the ALP problem, the first trapdoor function we require is a trapdoor sampling algorithm constructed by Alwen and Peikert [13]. For an almost uniformly random matrix $A \in Z_p^{n\times m}$, the trapdoor $T \in Z_p^{m\times m}$ generated by this trapdoor algorithm can be used to solve the ALP problem. That is, given $b = sA + e$, it can be used to find s.

Lemma 3.1. (Theorem 3.1 and 3.2 [13]). There is a probabilistic polynomial-time algorithm that, on input a positive integer *n*, positive integer p, and a poly(*n*)-bounded positive integer $m \geq 8n\log p$, outputs a pair of matries $A \in Z_p^{n\times m}$, $T \in Z^{m\times m}$ such that A is statistically close to uniform over $Z_p^{n\times m}$, $AT = 0 \bmod p$, and $\|T\|_\infty = O(n\log p)$.

To construct the trapdoor algorithm based on the APIP problem, we first fix $k = k(\lambda)$ and choose a small coefficient principal ideal $t \in R$, then evaluate the orthogonal principal ideal a of t over R_p, where $p \,|\, \det(Rot(t))$ is an appropriate integer.

Lemma 3.2. Given an arbitrary $t \in R$, there is a polynomial time algorithm that generate the orthogonal principal ideal a of t over R_p with $p \,|\, \det(Rot(t))$, that is, $a \times t = 0 \bmod p$.

Proof: We construct a linear equation system according to the relationship $a \times t = 0 \bmod p$ as follows:

$$\begin{pmatrix} t_0 & -t_{k-1} & \cdots & -t_1 \\ t_1 & t_0 & \cdots & -t_2 \\ \vdots & \vdots & \vdots & \vdots \\ t_{k-1} & t_{k-2} & \cdots & t_0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} = \begin{pmatrix} qv_0 \\ qv_2 \\ \vdots \\ qv_{k-1} \end{pmatrix}, \text{ where } v_i \in Z_q.$$

Since $p \,|\, q = \det(Rot(t))$, we choose a uniformly random vector $v \in Z_q^n$ and solve the integer coefficients $a_i^{'}$ for a' modulo q by using Cramer rule. By $p \,|\, \det(Rot(t))$, we get $a = a' \bmod p$. ■

The aim we introduce the ALP problem is to construct a new fully homomorphic encryption. But in the Lemma 3.1, the entries of the trapdoor T is too large and its dimension m depends on the modulus p. So, we also apply the above method to generate the short basis for general lattice. Our construction differs from one of [13-14]. Here we first fix n, m and $n \leq m$, choose a random basis $T \in Z^{m\times m}$ with small entries, then evaluate the random orthogonal basis $A' \in Z_p^{m\times m}$ for T by applying Cramer rule such that $A'T = 0 \bmod p$, where $p \,|\, \det(T)$, and finally set A to be equal to n random different rows of A'. Whereas the algorithms in [13-14] first fix m, n, p, and then generate the matries A, T such that $AT = 0 \bmod p$ and $\|T\| \leq O(n\log p)$.

Lemma 3.3. There is a probabilistic polynomial time algorithm that, on input positive integers $n \leq m$, outputs a pair of matries $A \in Z_p^{n\times m}$, $T \in Z^{m\times m}$ such that $AT = 0 \bmod p$, $\|T\|_\infty = O(1)$, and $p \,|\, \det(T)$.
In Lemma 3.3, we assume that A is statistically close to uniform over $Z_p^{n\times m}$. Whether this assumption can be proved remains an open problem. Of course, if one can prove that the instantiation of ALP generated by A is almost uniform over Z_p^m, then it is also feasible for our use.

Since there is a dependent relationship among the columns of A (resp. a) over the modulus p in Lemma 3.1-3.3, they can not be uniform over $Z_p^{n\times m}$ (resp. R_p). So, we give a new trapdoor in the following Lemma.

Lemma 3.4. There is a probabilistic polynomial-time algorithm that, on input positive integers m, p, outputs a pair of matries $A \in Z_p^{m\times m}$, $T \in Z^{m\times m}$ such that A is statistically close to uniform over $Z_p^{m\times m}$, $AT = I \bmod p$, $\|T\|_\infty = O(1)$, and $\gcd(p, \det(T)) = 1$, where I is an identity matrix of $Z_p^{m\times m}$.

Proof: Given m,p, one first chooses at random $T\in Z^{m\times m}$ with $\|T\|_\infty=O(1)$ and $q=\det(T)$, and decides whether $\gcd(p,q)=1$. If $\gcd(p,q)=1$, it is easy to evaluate that A' and the inverse q' of q over modulus p such that $A'T=q\Box I$ and $q'\Box q=1\bmod p$. Now, we set $A=(q'\Box A')\bmod p$ and get $AT=I\bmod p$.■

It is obvious that the Lemma 3.4 works over the ring R_p as well since the principal ideal lattice is a special case of general lattice.

3.2 Public Key Scheme Based on ALP

For simplicity, we only give ALP-based public key encryption schemes in this section. The first public key encryption scheme is based on the ALP problem with related modulus p, called ALP-RM, whereas the second scheme is based on the ALP problem with unrelated modulus p, called ALP-UM.

3.2.1 Construction of PKE-1

Key Generating Algorithm (PKE-1.KeyGen):

(1) Let n,m,p be integers related to security parameter λ, and p an odd integer. By using Lemma 3.1, one generates a pair of matries $A\in Z_p^{n\times m}$, $T\in Z^{m\times m}$ such that A is statistically close to uniform over $Z_p^{n\times m}$, $AT=0\bmod p$, $\det(T)$ is an odd integer, and $\|T\|_\infty=O(n\log p)$.

(2) Let χ be a distribution over Z_p^m. Choose a list $\tau=O(\lambda)$ elements $b_i=s_iA+2e_i$ over Z_p^m such that $s_i\leftarrow Z_p^n$, $e_i\leftarrow\chi$ with $\|e_i\|_\infty\le n/2$.

(3) Output the public key $pk=(m,p,b_i,i\in[\tau])$ and the secret key $sk=(T)$.

Encryption Algorithm (PKE-1.Enc). Given the public key pk and a message $x\in\Box_2^m$, choose a random subset $S\subseteq[\tau]$ and an independent 'small' error term $e\leftarrow\chi$ with $\|e\|_\infty\le n/2$. Evaluate a ciphertext $c=\left[\sum_{i\in S}b_i+2e+x\right]_p$.

Decryption Algorithm (PKE-1.Dec). Given the secret key sk, and the ciphertext c, decipher $x=\left[\left[[cT]_p\right]_2([T]_2)^{-1}\right]_2$.

Correctness: When $p>2\left\|(x+\sum_{i\in S}2e_i)T\right\|_\infty$, Dec works correctly because

$$\begin{aligned}&\left[\left[[cT]_p\right]_2([T]_2)^{-1}\right]_2\\ &=\left[\left[\left[(x+\sum_{i\in S}s_iA+2e_i)T\right]_p\right]_2([T]_2)^{-1}\right]_2\\ &=\left[\left[\left[(x+\sum_{i\in S}2e_i)T\right]_p\right]_2([T]_2)^{-1}\right]_2\\ &=\left[\left[(x+\sum_{i\in S}2e_i)T\right]_2([T]_2)^{-1}\right]_2 \qquad .\\ &=\left[[xT]_2([T]_2)^{-1}\right]_2\\ &=\left[[x]_2[T]_2([T]_2)^{-1}\right]_2\\ &=x\end{aligned}$$

3.2.2 Construction of PKE-2

Key Generating Algorithm (PKE-2.KeyGen):

(1) Let m,p be integers related to security parameter λ, and p an odd integer. By using Lemma 3.4, one generates a pair of matries $A\in Z_p^{m\times m}$, $T\in Z^{m\times m}$ with $\|T\|_\infty=O(1)$ such that A is statistically close to uniform over $Z_p^{m\times m}$, $AT=I\bmod p$, where I is an identity matrix of $Z_p^{m\times m}$.

(2) Let χ, φ respectively be the distributions over Z^m. Choose a list $\tau=O(\lambda)$ elements

$b_i = (2s_i A + 2e_i) \bmod p$ over Z_p^m such that $s_i \leftarrow \varphi$ with $\|s_i\|_\infty \le m/2$, $e_i \leftarrow \chi$ with $\|e_i\|_\infty \le m/2$.

(3) Output the public key $pk = (m, p, b_i, i \in [\tau], \chi)$ and the secret key $sk = (T)$.

Encryption Algorithm (PKE-2.Enc). Given the public key pk and a message $x \in Z_2^m$, choose a random subset $S \subseteq [\tau]$ and an independent 'small' error term $e \leftarrow \chi$ with $\|e\|_\infty \le m/2$. Evaluate a ciphertext $c = \left[\sum_{i \in S} b_i + 2e + x\right]_p$.

Decryption Algorithm (PKE-2.Dec). Given the secret key sk , and the ciphertext c , decipher $x = \left[\left[[cT]_p\right]_2 ([T]_2)^{-1}\right]_2$.

Correctness: Dec works correctly because

$$\begin{aligned} & \left[\left[[cT]_p\right]_2 ([T]_2)^{-1}\right]_2 \\ &= \left[\left[[(sA + 2e + x)T]_p\right]_2 ([T]_2)^{-1}\right]_2 \\ &= \left[[s + (2e + x)T]_2 ([T]_2)^{-1}\right]_2 \\ &= \left[[xT]_2 ([T]_2)^{-1}\right]_2 \\ &= x \end{aligned} .$$

In the above, we use $s = 0 \bmod 2$ and $p > 2\|s + (x + 2e)T\|_\infty$ because s, e, T all have small entries.

4. Somewhat Homomorphic Encryption (SHE-1)

In Section 4, we present a somewhat homomorphic encryption based on the APIP with related modulus p, call APIP-RM. In Section 5, we transform the SHE-1 scheme into a new fully homomorphic encryption (FHE-1). In Section 6, we analyze the security of the FHE-1 and discuss two possible attacks for the FHE-1.

4.1 Construction

To construct fully homomorphic encryption, the SHE requires to evaluate an arbitrary circuit with depth $d = O(\log n)$. Moreover, the depth of its decryption circuit is less than d . Thus, we first choose special secret key to implement FHE, and then extend it to general parameters setting.

Key Generating Algorithm (SHE-1.KeyGen):

(1) Select a random polynomial $s = \sum_{i=0}^{n-1} s_i x^i$ such that $s_0 = 2^\theta + 1$ with $\theta \in [\eta] \backslash 0$, $s_i \in S, i \in [n-1] \backslash 0$ and $l = \sum_{i=0}^{n-1} w(s_i) = \omega(\log n)$, and evaluate $p = \det(Rot(s))$ such that $p \ge 2^{\eta n}$ is an odd integer, where $S = \{0, 1, 2^1, ..., 2^\eta\}$, η in general is a constant integer, $w(s_i)$ is the hamming weight of s_i .

(2) By Lemma 3.2, one compute a random f over R , $f = \sum_{i=0}^{n-1} f_i x^i$ subject to $s \times f = 0 \bmod p$.

(3) Pick $\tau = O(n)$ uniformly $b_i = (a_i \times f + 2e_i) \bmod p$ with $\|e_i\|_\infty \le n/2$.

(4) Output the public key $pk = (n, p, b_i, i \in [\tau])$ and the secret key $sk = (s)$.

Encryption Algorithm (SHE-1.Enc). Given the public key pk and a message bit $m \in \{0,1\}$, choose a random subset $T \subseteq [\tau]$ with $|T| \le n-2$ and an independent 'small' error term e with $\|2e_i\|_\infty \le n$. Evaluate a ciphertext $c = (\sum_{i \in T} b_i + 2e + m) \bmod p$.

Add Operation (SHE-1.Add). Given the public key pk, and the ciphertexts c_1, c_2, evaluate the ciphertext $c = (c_1 + c_2) \bmod p$.

Multiplication Operation (SHE-1.Mul). Given the public key pk , and the ciphertexts c_1, c_2 , evaluate the ciphertext $c = (c_1 \times c_2) \bmod p$.

Decryption Algorithm (SHE-1.Dec). Given the secret key sk, and the ciphertext c, decipher $m=([c\times s]_p)\bmod x \bmod 2$.

Remark 4.1: We can replace $pk=(n,p,b_i,i\in[\tau])$ with $pk=(n,p,b)$ such that $b=(a\times f+2e)\bmod p$ with $\|2e\|_\infty \le n$. When encrypting a message bit $m\in\{0,1\}$, we select at random $u_1,u_2\in R$ with $\|2u_i\|_\infty \le n$, and output a ciphertext $c=(b\times u_1+2u_2+m)\bmod p$.

4.2 Correctness

Lemma 4.1. The above SHE-1.Dec algorithm is correct, if the infinity norm of the error term in the ciphertext is less than $p/(4n^2\times 2^\eta)$.

Proof. Given the ciphertext c and the secret key sk, it is not difficult to verify that c has general form $c=(a\times f+2e+m)\bmod p$. To decrypt c, we evaluate

$$c_s=[c\times s]_p=[(a\times f+2e+m)\times s]_p=[2e\times s+m\times s]_p.$$

Since $\|2e\|_\infty < p/(4n^2\times 2^\eta)$, $\|c_s\|_\infty=\|(2e+m)\times s\|_\infty \le \|s\|_1\times p/(4n^2\times 2^\eta)\le p/(4n)$. By $s_0=1 \bmod 2$, we get the message bit $m=c_s \bmod x \bmod 2=(m\times s_0)\bmod 2$.■

Remark 4.2: The reason the error term is less than $p/(4n)$ is to implement the Recrypt algorithm in the fully homomorphic encryption.

Lemma 4.2. The above scheme is correct for arbitrary arithmetic circuit C with addition and multiplication gates, and circuit depth $d\le \log(n\eta-\eta-2)-\log\log n-2$.

Proof. Assume $c_j=(\sum_{i\in T_j} b_i+2e_j+m_j)\bmod p, j=1,2$ are the ciphertext generated by **Enc**. To correctly decrypt, the error term of the ciphertext output by arithmetic circuit can not be too large. The error term in addition gate is linearly rising, whereas the error term in multiplication gate is exponentially increasing. So, the multiplication operation dominates the depth of arithmetic circuit. Now, we estimate the bound of the error term in the ciphertext generated by one multiplication operation.

$$\begin{aligned} c &= c_1\times c_2 \bmod p \\ &=(\textstyle\sum_{i\in T_1} b_i+2e_1+m_1)\times(\sum_{i\in T_2} b_i+2e_2+m_2)\bmod p. \\ &=(a\times f+2e+m_1m_2)\bmod p \end{aligned}$$

where $a=((\sum_{i\in T_1} b_i+2e_1+m_1)\times(\sum_{i\in T_2} a_i)+2\sum_{i\in T_2} e_i+2e_2+m_2)\bmod p$,

$e=((\sum_{i\in T_1} e_i+e_1)\times(\sum_{i\in T_2} 2e_i+2e_2+m_2)+m_1\times(\sum_{i\in T_2} e_i+e_2)\bmod p$.

So,

$$\begin{aligned} \|2e\|_\infty &= \left\|((\textstyle\sum_{i\in T_1} 2e_i+2e_1)\times(\sum_{i\in T_2} 2e_i+2e_2+m_2)+m_1\times(\sum_{i\in T_2} 2e_i+2e_2)\right\|_\infty \\ &\le n\left\|((\textstyle\sum_{i\in T_1} 2e_i+2e_1)\right\|_\infty \left\|\textstyle\sum_{i\in T_2} 2e_i+2e_2+m_2\right\|_\infty+\left\|\textstyle\sum_{i\in T_2} 2e_i+2e_2\right\|_\infty \quad . \\ &\le n((n-2)n+n)((n-2)n+n+1)+(n-2)n+n \\ &< n^5 \end{aligned}$$

In the other hand, the error terms in the ciphertexts c_1,c_2 are at most n^2. So, the error term for one multiplication is less than $n(n^2)^2<(n^2)^{2^{1+1}-1}$. To correctly decrypt, the arithmetic circuit depth d must be satisfied inequality $(n^2)^{2^{d+1}-1}<p/(4n^2\times 2^\eta)$, namely, $d\le\log(n\eta-\eta-2)-\log\log n-2$.■

4.3 Performance

The public key size $pk=(n,p,b_i,i\in[\tau])$ is $O(n^3\eta)$, the secret key size $sk=(s)$ is $O(n\eta)$. The expansion factor of ciphertext is $O(n^2\eta)$. The running times of the Enc, Dec, Add, Mul algorithms are respectively $O(n^3\eta)$, $O(n^2\eta\log n)$, $O(n^2\eta)$, and $O(n^3\eta\log n)$.

5. Fully Homomorphic Encryption (FHE-1)

5.1 Construction of FHE-1

We design an FHE-1 from the SHE-1 by using self-loop bootstrappable technique. We give a new algorithm Recrypt, which refreshes a ‘dirty’ ciphertext c to a new ciphertext c_{new} with ‘smaller’ error term and the same plaintext of c. To do this, we require to add the ciphertexts of encrypted the secret key to the public key. So, we assume that our scheme is KDM-secure. We modify SHE-1 as follows:

FHE-1.KeyGen Algorithm:

(1) Generate $pk=(n,p,b_i,i\in[\tau])$ and $sk=(s)$ by using SHE-1.KeyGen algorithm.

(2) Select at random $n(\eta+1)$ pair elements $a_{i,j}\in R, i\in[n-1], j\in[\eta]$, and perturbed error terms $e_{i,j}\in R$ such that $\|2e_{i,j}\|_\infty\le n$, and encrypt the j-th bit $s_{i,j}$ of s_i as $\vec{s}_{i,j}=(a_{i,j}\times f+2e_{i,j}+s_{i,j}) \bmod p$. We denote $\vec{s}_i=\sum_{j=0}^{\eta}\vec{s}_{i,j}2^j$ and $\vec{s}=\sum_{i=0}^{n-1}\vec{s}_i x^i$.

(3) Output the public key $pk=(n,p,\{b_i\}_{i=0}^{\tau},\vec{s})$ and the secret key $sk=(s)$.

FHE-1.Recrypt Algorithm:

(1) Set $\vec{c}_0=c_0$, $\vec{c}_i=p-c_i$ for $i\in[n-1]\backslash 0$, and $\vec{h}_{i,j}=\left\langle \vec{c}_i\times 2^j/p\right\rangle$ for $i\in[n-1], j\in[\eta]$, keeping only $k=\log n$ bits of precision after the binary point for each $\vec{h}_{i,j}$, where $\vec{h}_{i,j}=\left\langle \vec{c}_i\times 2^j/p\right\rangle$ is satisfied to $\left|\vec{h}_{i,j}-\vec{c}_i\times 2^j/p\right|<1/(2n)$.

(2) Evaluate $\vec{h}_i=\left[\sum_{j=r}^{\eta}\vec{h}_{i,j}\times\vec{s}_{t,j}\right]_2$ for $i+t=0 \bmod n$, $i\in[n-1]$, where $r=1$ if $i=0$, otherwise $r=0$, and $\vec{g}=\left\lfloor \sum_{i=0}^{n-1}\vec{h}_i+\vec{h}_{0,0}+0.5\right\rfloor \bmod 2$.

(3) Evaluate $\vec{u}=(\sum_{i+t=0 \bmod n}\vec{c}_i\times\vec{s}_t) \bmod 2=(\sum_{i+t=0 \bmod n}\left[\vec{c}_i\right]_2\times\vec{s}_{t,0}) \bmod 2$.

(4) Output a new ciphertext $c_{new}=\vec{u}\oplus\vec{g}$.

Theorem 5.1. The FHE-1.Recrypt correctly generates a ‘fresh’ ciphertext c_{new} with the same message of c, and two homomorphic decrypted ciphertexts support one multiplication when $2n^{6n-3}\eta^{4n-1}\le p/(4n^2 2^\eta)$.

Proof: First, we have

$$\begin{aligned}&(c\times s) \bmod (x^n+1) \bmod x\\&=c_0s_0-c_1s_{n-1}-c_2s_{n-2}-\cdots-c_{n-1}s_1\\&=c_0s_0+(p-c_1)s_{n-1}+(p-c_2)s_{n-2}+\cdots+(p-c_{n-1})s_1\\&=\sum_{i+t=0 \bmod n}\vec{c}_i s_t\end{aligned} \qquad (5\text{-}1).$$

So, the decryption algorithm computes as follows

$$\begin{aligned}&\left[c\times\vec{s} \bmod (x^n+1) \bmod x\right]_p \bmod 2\\&=(\sum_{i+t=0 \bmod n}\vec{c}_i\times\vec{s}_t) \bmod 2)\oplus(\left\lfloor(\sum_{i+t=0 \bmod n}\vec{c}_i/p\times\vec{s}_t)+0.5\right\rfloor \bmod 2)\\&=\vec{u}\oplus(\left\lfloor\sum_{i+t=0 \bmod n}\vec{c}_i/p\times\sum_{j=0}^{\eta}\vec{s}_{t,j}2^j+0.5\right\rfloor \bmod 2)\\&=\vec{u}\oplus(\left\lfloor\sum_{i+t=0 \bmod n}\sum_{j=0}^{\eta}\vec{s}_{t,j}(\vec{c}_i\times 2^j)/p+0.5\right\rfloor \bmod 2) \qquad (5\text{-}2).\\&=\vec{u}\oplus(\left\lfloor\sum_{i+t=0 \bmod n}\sum_{j=0}^{\eta}\vec{h}_{i,j}\vec{s}_{t,j}+0.5\right\rfloor) \bmod 2\\&=\vec{u}\oplus(\left\lfloor\sum_{i=0}^{n-1}\vec{h}_i+\vec{h}_{0,0}+0.5\right\rfloor) \bmod 2\\&=\vec{u}\oplus\vec{g}\end{aligned}$$

So, we merely prove that FHE-1.Recrypt correctly evaluates the formula (5-2) in the form of ciphertexts. Since $\vec{u}=(\sum_{i+t=0 \bmod n}\left[\vec{c}_i\right]_2\times\vec{s}_{t,0}) \bmod 2$ and $\|2e_{t,0}\|_\infty\le n$ in $\vec{s}_{t,0}$, we evaluate the sum modulo 2 of n ciphertexts. Hence, the error term in the ciphertext $\vec{u}$ is at most n^2.

To estimate the error term of $\vec{g}$, we first determine the error term of $\vec{h}_i$. According to FHE-1.KeyGen, there is at most single 1-bit among $\vec{s}_{t,j}$, $j\in[\eta]$ except for $\vec{s}_0$ that includes two 1-bits. So, the error term in $\vec{h}_i = \left[\sum_{j=r}^{\eta} \vec{h}_{i,j} \times \vec{s}_{t,j}\right]_2$ is at most $n(\eta+1)$. What is more, there is at most $l+1$ non-zero numbers among encrypted $n+1$ rational numbers via $l = \sum_{i=0}^{n-1} w(s_i)$.

Since $\left|\vec{h}_{i,j} - \vec{c}_i \times 2^j / p\right| < 1/(2n)$, we get $\left|\vec{h}_i - \sum_{j=0}^{\eta} \vec{s}_{t,j} (\vec{c}_i \times 2^j)/p\right| < 1/(2n)$. So, $\left|\sum_{i=0}^{n-1} \vec{h}_i + \vec{h}_{0,0} - \sum_{i+t=0 \bmod n} \vec{c}_i \times \vec{s}_t / p\right| < \frac{l+1}{2n}$. According to Lemma 4.1, there is an encrypted integer $\vec{z}$ such that

$$\begin{aligned} &\left|\sum_{i=0}^{n-1} \vec{h}_i + \vec{h}_{0,0}\right| \\ &\le \left|\sum_{i=0}^{n-1} \vec{h}_i + \vec{h}_{0,0} - \sum_{i+t=0 \bmod n} \vec{c}_i \times \vec{s}_t / p\right| + \left|\sum_{i+t=0 \bmod n} \vec{c}_i \times \vec{s}_t / p\right|. \\ &\le (n-1)/2n + \vec{z} + 1/(4n) \\ &< \vec{z} + 1/2 \end{aligned}$$

So, suppose $\vec{g}' = \sum_{i=0}^{n-1} \vec{h}_i + \vec{h}_{0,0} = \vec{g}_0 . \vec{g}_{-1} ... \vec{g}_{-k}$, then $\vec{g} = (\vec{g}_0 + \vec{g}_{-1}) \bmod 2$.

By applying the symmetric polynomial technique [10], we use the polynomial with total degree $l+1$ to evaluate the sum of $n+1$ encrypted rational numbers with at most $l+1$ nonzero numbers. It is easy to verify that the number of degree $l+1$ monomials in the polynomial representing our addition of ciphertexts is equal to $\binom{l+1}{\lceil l+1/2 \rceil} \times \binom{l+1}{\lceil l+1/4 \rceil} \times ... \times \binom{l+1}{1}$, which is less than $(l+1)^{l+1}$. The error term of a degree $l+1$ monomial over ciphertexts is at most $(n\eta)^{2l+1}$. So, the error term of $\vec{g}$ is at most $(l+1)^{(l+1)} (n\eta)^{2l+1}$.

To obtain FHE, the scheme must support another homomorphic multiplication. So, the scheme needs to correctly decrypt a ciphertext with error term $((l+1)^{(l+1)} (n\eta)^{2l+1} + n^2) \le 2(l+1)^{2(l+1)} (n\eta)^{4l+2}$. Thus, $2(l+1)^{2(l+1)} (n\eta)^{4l+2} \le p/(4n^2 2^{\eta})$ by Lemma 4.1.■

5.2 General Parameters

In the FHE-1.KeyGen algorithm, we use a special form for the secret key. Indeed, one may set general parameters. Assume $s = \sum_{i=0}^{n-1} s_i x^i$ with $\|s\|_\infty = 2^\eta$ and $p = O(2^{n\eta})$ an odd integer. One selects at random a polynomial $u = \sum_{j=0}^{n-1} u_j x^j \in R$ with $w(u_j) \le 1$ and $l = \sum_{j=0}^{n-1} w(u_j) = \omega(\log n)$, and take $v = s - u$. We then encrypt u as $\vec{u}$ same as $\vec{s}$ in FHE-1.KeyGen, and output the public key $pk = (n, p, \{b_i\}_{i=0}^{\tau}, \vec{u}, v)$ and the secret key $sk = (s)$.

For the general parameters of the secret key, we will use it to generate p as a product of smoothing primes and prove the security of scheme in the following.

In addition, we can also apply the Gentry's method, which introduces the hardness assumption of the sparse subset sum problem when implementing FHE-1.

5.3 Extension to Large Message Space

For the FHE-1, one can reduce the expansion factor of ciphertext from $O(n^2\eta)$ to $O(n\eta)$ by extending plaintext message space. For a message $m \in \{0,1\}^n$, one maps it to a polynomial $m(x) = \sum_{i=0}^{n-1} m_i x^i$. FHE-1.Enc is $c = (\sum_{i \in T} b_i + 2e + m(x)) \bmod p$, and FHE-1.Dec is $m(x) = Rot((s \bmod 2)^{-1}) \times ([c \times s]_p \bmod 2)$.

In this case, we add to the public key the ciphertexts $\vec{s}_v$ of the inverse polynomial $s_v = (s \bmod 2)^{-1}$ of $s \bmod 2$ to unpack message. Moreover, when FHE-1.Recrypt refreshes a ciphertext, one gets n ciphertexts: $\sum_{i=0}^{n-1} \vec{C}_i x^i = \vec{s}_v \times ([c \times \vec{s}]_p \bmod 2)$, where each $\vec{C}_i$ is a ciphertext of one bit. So, we must pack $\sum_{i=0}^{n-1} \vec{C}_i \times x^i$ consisting of n ciphertexts into a new ciphertext $c_{new} = \sum_{i=0}^{n-1} (\vec{C}_i \times x^i) \bmod (x^n + 1)$.

One can perform homomorphic bit operations for the large message space above. To evaluate homomorphic operation over the bits, one firstly calls FHE-1.Recrypt to obtain each encrypted bit of message m, then performs homomorphic operations over each bit, and finally packs the ciphertexts of n encrypted bits into a ciphertext of n bits message by evaluating c_{new}.

6. Security of FHE-1

6.1 Security Analysis

The security of the SHE-1 follows directly from the hardness of the decisional hidden principal ideal lattice problem. The proof of the following theorem adapts the proof of Theorem 3 of [15]. We include it here for completeness.

Theorem 6.1. Suppose there is an algorithm A which breaks the semantic security of our SHE-1 with advantage ε. Then there is a distinguishing algorithm D against $APIP-RM_{k,p,\chi}$ with running in about the same time A and advantage at least $\varepsilon/2$.

Proof. We construct a distinguishing algorithm D with advantage at least $\varepsilon/2$ between the distribution $D_{n,p,\chi}$ and the uniform distribution over R_p. The algorithm D receives as input c. D picks at random $\alpha \in \{0,1\}$, sends the challenge ciphertext $2c+\alpha \bmod p$ to A, then returns 1 if A guesses the right α, and otherwise 0. We omitted the remainder of proof, which is almost identical to [15]. ■

Recall that f is an arbitrary in the $RLWE_{k,p,\chi}$ problem in Definition 2.2, whereas f in the SHE-1 is satisfied to $p\,|\det(Rot(f))$. Thus, the hardness result in this paper is only available for this special $RLWE_{k,p,\chi}$ problem.

Theorem 6.2. Suppose p is the product of distinct smoothing primes. Then there is a probabilistic polynomial time reduction from $RLWE_{k,p,\chi}$ to $APIP-RM_{k,p,\chi}$.

Proof. It is obvious that by removing a, we transform an instantiation of $RLWE_{k,p,\chi}$ into an instantiation of $APIP-RM_{k,p,\chi}$.■

Theorem 6.3. Suppose p is a product of distinct smoothing primes. Then there is a probabilistic polynomial time reduction from $RLWE_{k,p,\chi}$ to the search $RLWE_{k,p,\chi}$.

Proof. The proof of Theorem 6.3 is adapted from that of Lemma 3.6 in [16]. ■

Theorem 6.4. Suppose p is the product of distinct smoothing primes. Then there is a probabilistic polynomial time reduction from the search $RLWE_{k,p,\chi}$ to $APIP-RM_{k,p,\chi}$.

From Theorem 6.4, we know that breaking our scheme is harder than solving the $RLWE_{f,\varphi}$ problem when p is the product of distinct smoothing primes.

Theorem 6.5. Suppose the $APIP-RM_{k,p,\chi}$ problem is hard for any PPT adversary A. Then the FHE-1 is semantic security.

6.2 Known Attack

6.2.1 Attacking Generator of the Secret Key

When p is a prime, $\gcd(x^n+1,s)\neq 1 \bmod p$. Since one can factor x^n+1 modulo p and guess a principal ideal generator for the secret key s. For example, $s=x^3+2x^2+x+1=(x+8)(x^2+11x+15)\bmod 17$, where $p=\det(Rot(s))=17$, $x^n+1=(x+9)(x+15)(x+2)(x+8)\bmod 17$. So, one can enumerate the generators of all possible principal ideals of s, and find a small generator for each principal ideal. The hardness of breaking the scheme is reduced to finding a small generator of a principal ideal given two integers (p,α_i), where α_i is the *i*-th root of x^n+1 modulo p. We observe that in fact one must not find the smallest generator of a principal ideal, and only needs to solve a 'small' multiple of the smallest generator. So, we must avoid this attack to guarantee the security of our scheme. We may adopt methods as follows.

(1) The security of our scheme depends on factoring integer problem. In order to use small n, such as *n*=64, 128, we set the modulo p to be a product of two large primes. For example, one selects at random $s_i \in R, i=1,2$ with $p_i=\det(Rot(s_i))$ primes, and takes $s=s_1\times s_2 \bmod (x^n+1)$ and $p=p_1p_2$. To implement FHE, we apply the method of general parameters in Section 4.2. As far as we know, there is not an efficient

algorithm which factors x^n+1 modulo p without factoring p. This is probably independent of interest. Since all previous schemes are based on (principal) ideal lattices [7, 8, 10] or the approximate GCD [1].

(2) We may use p to be the product of $O(n)$ distinct smoothing primes. For example, one picks $\varsigma = O(n)$ small polynomials $s_i \in R, i \in [\varsigma]$, whose determinants $p_i = \det(Rot(s_i))$ of their circulant matrices are co-prime smoothing factors and $p_i = n^{O(1)}$, and takes $s = \prod_{i=0}^{\varsigma} s_i \bmod (x^n+1)$ and $p = \prod_{i=0}^{\varsigma} p_i$. For this case, we require that the lattice dimension n are large enough to ensure the above attack to be infeasible for arbitrary subset with size $\omega(\log n)$ of $[\varsigma]$. It is easy to check that the number of all possible distinct principal ideals of s is $n^{O(n)}$. To obtain FHE, we also apply the method in Section 4.2.

(3) We can set large lattice dimension, lower hamming weight of secret key, and smaller error term in original ciphertexts. For example, we take $n = 8192$, $|p| = 460$, $s = \sum_{i=0}^{n-1} s_i x^i = \sum_{i=0}^{n-1} (\sum_{j=0}^{7} s_{i,j} 2^j) x^i$ such that $\sum_{i=0}^{n-1} s_{i,j} = 1$ for $j \in [7]$. In FHE-1, we take $\|e\|_\infty = 1$. In this case, we change Recrypt as follows:

- Set $g = c/p$, keeping only $\theta = 8+4+3 = 15$ bits of precision after the binary point for each coefficient g_i of g.
- Evaluate $\vec{u}_j = g \times \vec{s}_{\square,j} 2^j$ for $j \in [7]$, where $\vec{s}_{\square,j}$ is the ciphertexts of $s_{\square,j} = \sum_{i=0}^{n-1} s_{i,j} 2^j x^i$, and $u_1 = \left\lfloor \sum_{j=0}^{7} \vec{u}_j + 0.5h \right\rfloor$.
- Evaluate $u_2 = [c]_2$, and output a new ciphertext $c_{new} = (u_1 \oplus u_2) \bmod x$.

If using large message space, one requires to transform n ciphertexts into a new ciphertext.

Since the error term size of $\vec{u}_j$ is at most $8192 \times 2 = 2^{14}$, one can sum 8 encrypted rational numbers $\vec{u}_j$ and easily verify that the error size of u_1 is at most 2^{218} by applying the method in [10]. To support one multiplication over homomorphic decrypted ciphertexts, we set $p > 2^{457}$. To quickly generate the secret key, we use the method in Section 4.2. On the other hand, the approximation factor of lattice reduction algorithm is about $(1.02)^{2*8192} \approx 2^{469}$ over average case according to [17].

6.2.2 Lattice Reduction Attack over the Ciphertexts

For a 0-bit ciphertext b in the public key, one can construct a $(n+1)\times(n+1)$ matrix as follows:

$$M = \begin{pmatrix} p & 0 & 0 & \cdots & 0 \\ b_0 & 1 & 0 & \cdots & 0 \\ b_1 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{n-1} & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

According to the Minkowski's Theorem, the lattice generated by M has non-zero vector less than $\sqrt{n+1}p^{1/(n+1)} \approx \sqrt{n}2^{\eta}$. On the other hand, by the parameter of our scheme, there is a non-zero vector s such that

$$\begin{pmatrix} \sum_{i+j=0 \bmod n} 2e_i s_j & s_0 & s_1 & \cdots & s_{n-1} \\ b_0 & 1 & 0 & \cdots & 0 \\ b_1 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{n-1} & 0 & 0 & \cdots & 1 \end{pmatrix}$$

However, it is not difficult to verify that there are exponential numbers of vectors with length $\left\|(\sum_{i+j=0 \bmod n} 2e_i s_j \quad s_0 \quad s_1 \quad \cdots \quad s_{n-1})\right\|_2$, which is maybe not the shortest non-zero vector of the above lattice. Thus, one can not get the secret key by this attack method.

7. Fully Homomorphic Encryption (FHE-2)

Since the determinant p of circulant matrix of the secret key is public in FHE-1, one can factor x^n+1 mod p and attempt to evaluate the generator polynomial of the secret key. So, to avoid this attack, we will construct ALP-UM based fully homomorphic encryption scheme (FHE-2). For efficiency, we also construct a self-loop fully homomorphic encryption scheme. In this section, we first discuss how to perform homomorphic operations over the PKE-2 scheme, next construct a new fully homomorphic encryption, analyze its security, and finally discuss issues of optimization and implementation. Notice that we use message space $x \in Z_2$, instead $x \in Z_2^m$ for PKE-2 in the following section.

7.1 Homomorphic Operations of PKE-2

It is obvious that the PKE-2 supports addition operation over the ciphertexts. So, we only discuss how to perform multiplication operation. According to the method of [18-20], they consider the multiplication operation over ciphertexts as the quadratic equation, that is, given the ciphertexts c_1, c_2 that encrypts x_1, x_2 and the secret key t: $Q_{c_1,c_2}(t) =< c_1, t > \times < c_2, t >$. If the noise of c_1, c_2 is small, then we can get $x_1 \times x_2$ by computing $\left[\left[Q_{c_1,c_2}(t)\right]_p\right]_2$. The problem is how to perform this function under ciphertexts. In [BV11, Gen11], they use the tensor product $t \otimes t$ of t to implement dimension reduction (key switching). Here, we apply another approach. Since $< c_1, t > \times < c_2, t > =<< c_1, t > \times c_2, t > =< c_2 \sum_{i=0}^{m-1} c_{1,i} t_i, t >$, we only require generate a new ciphertext by evaluating $c_2 \sum_{i=0}^{m-1} c_{1,i} t_i = (\sum_{i=0}^{m-1} c_{2,0} c_{1,i} t_i, \ldots, \sum_{i=1}^{m} c_{2,m-1} c_{1,i} t_i)$. To compute this ciphertext, we only need to call the following subroutines BitDecomp and Powersof2 introduced by [18-20].

Definition 7.1. (BitDecomp (Definition 5 [20])). Let $y \in Z_p^m$ and $N = m\lceil \log p \rceil$. We decompose y into its bit representation $y = \sum_{j \in [\lfloor \log p \rfloor]} 2^j u_j$, where all of the vectors $u_j \in Z_2^m$. Output $(u_0, u_1, \ldots, u_{\lfloor \log p \rfloor}) \in Z_2^N$.

Definition 7.2. (Powersof2 (Definition 6 [20])). Let $y \in Z_p^m$ and $N = m\lceil \log p \rceil$. We define Powersof2($y$, p) to be the vector $(y, 2 \cdot y, \ldots, 2^{\lfloor \log p \rfloor} \cdot y) \in Z_p^N$.

Lemma 7.1. (Lemma 2 [20]). For vectors c, t of equal length, we have

$$< BitDecomp(c, p), Powersof2(t, p) >=< c, t > \bmod p.$$

7.2 FHE-2 Based on ALP-UM

We now construct our self-loop FHE-2 scheme based on ALP-UM. We want to give addition algorithm, multiplication algorithm and recrypting algorithm over ciphertexts. To implement these algorithms, we need to add the ciphertexts of encrypted secret key to the public key.

In particular, we also use the method of FHE-1 for recrypting algorithm, that is, we choose the secret key with small hamming weight. Certainly, we may choose general parameters by applying the method in Section 5.2. In addition, to implement FHE-2, we also can directly use the dimension reduction (key switching) and modulus switching in [19-20].

Notice that in some sense, our scheme extends their schemes [19-20] to more general form. The public key of our scheme is the ciphertexts of their scheme. On the surface, this difference is small. In fact, this results in that the security of our scheme depends on the hardness assumption of the ALP problem. In this point, we believe that there is a relationship between the ALP and the closest vector problem (CVP). So, we generalize the LWE problem to the ALP problem, and construct a new fully homomorphic encryption based on ALP-UM.

FHE-2 constructs as follows:

FHE-2.KeyGen.

(1) Generate $pk = (m, p, b_i, i \in [\tau])$, $sk = (A, T)$ by using PKE-2.KeyGen. Without loss of generality, let t be the first column vector of T. By Lemma 3.4, assume $t = (t_0, t_1, \ldots, t_{m-1})^{\mathrm{T}} = (\sum_{j=0}^{\eta-1} 2^j t_{0,j}, \sum_{j=0}^{\eta-1} 2^j t_{1,j}, \ldots, \sum_{j=0}^{\eta-1} 2^j t_{m-1,j})^{\mathrm{T}}$ such that $t_0 = 2^\theta + 1$ with $\theta \in [\eta] \setminus 0$, $t_i \in S, i \in [m-1] \setminus 0$ and $\rho = \sum_{i=0}^{m-1} w(t_i) = \omega(\log \lambda)$, where $S = \{0, 1, 2^1, \ldots, 2^{\eta-1}\}$, η is a positive integer, $w(t_i)$ is the hamming weight of t_i.

(2) Let $N = m\lceil \log p \rceil$. Choose a list elements $b_{i,j} = 2s_{i,j}A + 2e_{i,j}$ over Z_p^m such that $s_{i,j} \leftarrow Z_p^m$, $e_{i,j} \leftarrow \chi$ with $\left\|s_{i,j}\right\|_\infty \le m/2, \left\|e_{i,j}\right\|_\infty \le m/2$, where $i \in [m-1], j \in [N-1]$.

(3) Let $B_i^{'}$, $i \in [m-1]$ be a matrix with row vectors $b_{i,j}$, $j \in [N-1]$. Evaluate

$B_i = B_i^{'} + Powersof\,2(t,p)_i$, where $Powersof\,2(t,p)_i$ is added to the *i*-th column of $B_i^{'}$.

(4) Choose a list elements $b_{i,j} = 2s_{i,j}A + 2e_{i,j}$, $i \in [m-1], j \in [\eta-1]$ over Z_p^m such that $s_{i,j} \leftarrow Z_p^m$, $e_{i,j} \leftarrow \chi$ with $\|s_{i,j}\|_\infty \le m/2, \|e_{i,j}\|_\infty \le m/2$, and evaluate $\vec{t}_{i,j} = \left[b_{i,j} + (t_{i,j},0,...,0)\right]_p$, denoted as $\vec{t} = (\sum_{j=0}^{\eta-1} 2^j \vec{t}_{0,j}, \sum_{j=0}^{\eta-1} 2^j \vec{t}_{1,j}, ..., \sum_{j=0}^{\eta-1} 2^j \vec{t}_{m,j})^{\mathrm{T}}$.

(5) Output the public key $pk = (m, p, \{b_i\}_{i=0}^{\tau}, \{B_i\}_{i=0}^{m-1}, \vec{t})$, and the secret key $sk = (t)$.

FHE-2.Enc. Given pk and a message bit $x \in □_2$, call PKE-2.Enc(*pk*, *x*).

FHE-2.Dec. Given sk, and a ciphertext c, call PKE-2.Dec(*sk*, *c*).

FHE-2.Add. Given pk and ciphertexts c_1, c_2, output $c = \left[c_1 + c_2\right]_p$.

FHE-2.Mul. Given pk and ciphertexts c_1, c_2, set $c = \left[\sum_{i=0}^{m-1} BitDecomp(c_{2,i}c_1)□B_i\right]_p$.

FHE-2.Recrypt. Given pk and ciphertext c, compute as follows:

(1) Set $\vec{c} = c/p$, keeping only $\theta = \lceil \log \rho \rceil + 3$ bits of precision after the binary point for each entry $\vec{c}_i$ of vector $\vec{c}$.

(2) Evaluate $u_1 = \left[\lfloor <\vec{c}, \vec{t}> + 0.5 \rfloor\right]_2$ and $u_2 = \left[<c, \vec{t}>\right]_2$.

(3) Output a new ciphertext $c_{new} = u_1 \oplus u_2$.

Correctness: the FHE-2.Add works correctly since

$$\left[\left[<\left[c_1 + c_2\right]_p, t>\right]_p\right]_2 = \left[\left[<c_1 + c_2, t>\right]_p\right]_2 = \left[\left[<c_1, t>\right]_p\right]_2 + \left[\left[<c_2, t>\right]_p\right]_2 = x_1 + x_2.$$

The FHE-2.Mul works correctly since

$$\begin{aligned}
&\left[\left[<c,t>\right]_p\right]_2 \\
&= \left[\left[<\left[\sum_{i=0}^{m-1} BitDecomp(c_{2,i}c_1) \times B_i\right]_p, t>\right]_p\right]_2 \\
&= \left[\left[<\sum_{i=0}^{m-1} BitDecomp(c_{2,i}c_1) \times B_i, t>\right]_p\right]_2 \\
&= \left[\left[<\sum_{i=0}^{m-1} BitDecomp(c_{2,i}c_1) \times B_i, t>\right]_p\right]_2 \\
&= \left[\left[<\sum_{i=0}^{m-1} BitDecomp(c_{2,i}c_1) \times (B_i^{'} + Powersof\,2(t,p)_i), t>\right]_p\right]_2 \\
&= \left[\left[< BitDecomp(c_{2,i}c_1) \times Powersof\,2(t,p)_i), t>\right]_p\right]_2 \\
&= \left[\left[<\sum_{i=0}^{m-1} c_{2,0}c_{1,i}t_i, ..., \sum_{i=1}^{m} c_{2,m-1}c_{1,i}t_i, t>\right]_p\right]_2 \\
&= \left[\left[<c_1, t> \times <c_2, t>\right]_p\right]_2 \\
&= x_1 \times x_2
\end{aligned}$$

In the above equality, the noise of ciphertext is less than $p/(2m\|t\|)$.

Now, we estimate the noise bound of the ciphertext after one homomorphic multiplication. Given two ciphertexts c_1, c_2, we have

$$\left[<c_1,t> \times <c_2,t>\right]_p = \left[<\left[<c_1,t>\right]_p \times c_2, t>\right]_p = \left[<<2e_1 + \bar{x}, t> \times c_2, t>\right]_p.$$

According to FHE-2.Enc, $\|<2e_1 + \bar{x}, t> \times 2e_2\| \le m^3 \|t\|$. On the other hand, to compute $<2e_1 + \bar{x}, t> \times c_2$, one requires to sum $m^2 \log p$ ciphertexts. This results in noise at most $m^3 \log p$. So, the noise bound of the ciphertext $c = c_1 \times c_2$ is at most $m^3 \log p + m^3 \|t\| \approx O(m^3 \log p)$.

Theorem 7.1. When $m^{O(\rho)} < p$, the FHE-2.Recrypt correctly generates a ‘refresh’ ciphertext c_{new} with the same message of c and smaller error term, and two homomorphic-decrypted ciphertexts support one multiplication.

Proof: This proof is similar as that of theorem 5.1.■

7.3 Security

In this section, we present the hardness assumption of the security of our scheme.

Theorem 7.2. Suppose p is the product of distinct smoothing primes. Then there is a probabilistic polynomial time reduction from the search $LWE_{n,p,\chi}$ to $ALP-UM_{n,m,p,\chi}$.

Proof: This proof is similar as that of theorem 6.2.■

Theorem 7.3. Suppose the $ALP-UM_{n,m,p,\chi}$ problem is hard for any PPT adversary A. Then the FHE-2 is semantic security.

7.4 Optimization

7.4.1 Large Message Space

In the FHE-2, one can use large message space as well. Of course, one requires to add the encrypted secret key to the public key. Namely, the public key includes the encrypted matrix $T\times([T]_2)^{-1}$ same as the encrypted vector t in FHE-2. This is because

$$\begin{aligned}
&\left[\left[c\times(T\times([T]_2)^{-1})\right]_p\right]_2\\
&=\left[\left[(x+\sum\nolimits_{i\in S}2s_iA+2e_i)\times(T\times([T]_2)^{-1})\right]_p\right]_2\\
&=\left[\left[(x\times(T\times([T]_2)^{-1})+\sum\nolimits_{i\in S}2s_i([T]_2)^{-1}+(\sum\nolimits_{i\in S}(2e_i)\times(T\times([T]_2)^{-1})\right]_p\right]_2.\\
&=\left[(x\times(T\times([T]_2)^{-1})+\sum\nolimits_{i\in S}2s_i([T]_2)^{-1}+(\sum\nolimits_{i\in S}2e_i)\times(T\times([T]_2)^{-1})\right]_2\\
&=\left[x\times(T\times([T]_2)^{-1})\right]_2\\
&=x
\end{aligned}$$

When one needs to perform bit operation, one must firstly unpack the ciphertext of encrypted *m* bits into *m* ciphertexts, each of which encrypts one bit. After operating, one can pack *m* ciphertexts into a ciphertexts by using homomorphic operation.

Hence, the expansion rate of our FHE-2 is $\log p=O(\lambda)$, which can be improved to $O(\log\lambda)$ by applying the dimension reduction [19].

7.4.2 Setting the Aggressive Public Key

Since A in PKE-2 is not public, we can set aggressively $B_i=2S_iA+\underbrace{(0,...,t,...,0)}_{t\ is\ in\ i-th\ column}\bmod p$ in FHE-2. KeyGen. So, we decrease a factor $\log p$ of the public key size.

7.4.3 Optimizing the Secret Key

For FHE-2, we can further optimize to decrease modulus p. Take $t=\sum_{i=0}^{\rho}u_i2^i$ with $u_i\in\square_2^m$ such that $\sum_{j=0}^{m-1}u_{i,j}=1$ for $i\in[\rho]$ and $\vec{u}_i$ is ciphertext vector of u_i. We modify FHE-2.Recrypt as follows:

(1) Set $\vec{c}=c/p$, keeping only $\theta=\lceil\log\rho\rceil+3$ bits of precision after the binary point for each entry $\vec{c}_i$ of vector $\vec{c}$.

(2) Evaluate $\vec{\vec{u}}_i=<2^i\vec{c},\vec{u}_i>$ for $i\in[\rho]$, $u_1=\left[\sum_{j=0}^{\rho}\vec{\vec{u}}_j+0.5h\right]_2$, and $u_2=\left[<c,\vec{u}_0>\right]_2$.

(3) Output a new ciphertext $c_{new}=u_1\oplus u_2$.

7.5 Extension to APIP-UM

To descrease the public key size in FHE-2, it is not difficult to construct APIP-UM based FHE by using the method of FHE-2. We here omit concrete details.

8. CONCLUSION

We have constructed two new fully homomorphic encryption schemes, whose securities respectively depend on the hardness assumptions of the APIP problem and the ALP problem.

This paper raises some interesting open problems. First, the securities of our schemes are based on the hardness of the decisional version of the APIP and ALP. It would be most desirable to reduce the search version to the decision version for the APIP/ALP problem. Second, the FHE-2 scheme has low efficient, can we improve its efficiency? Third, our public key has the form of the closest vector problem, whether or not we can build the relationship between the ALP problem and the CVP problem.

ACKNOWLEDGEMENTS

This work is supported by National Natural Science Foundation of China Grants (No. 61142007), Application Foundation Research of Changzhou City (No. CJ20120021), and by Foundation of Jiangsu Teachers University of Technology (No. KYY11055).

REFERENCES

[1] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. Eurocrypt 2010, LNCS 6110, pp. 24-43.

[2] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. LNCS, 2005, Volume 3378, pp. 325-341, 2005.

[3] C. Aguilar Melchor, G. Castagnos, and G. Gaborit. Lattice-based homomorphic encryption of vector spaces. In IEEE International Symposium on Information Theory, ISIT'2008, pp. 1858-1862, 2008.

[4] T. Sander, A. Young, and M. Yung. Non-interactive CryptoComputing for NC1. In 40th Annual Symposium on Foundations of Computer Science (FOCS '99), pp. 554-567, 1999.

[5] A. C. Yao. Protocols for secure computations (extended abstract). In 23rd Annual Symposium on Foundations of Computer Science (FOCS '82), pp. 160-164, 1982.

[6] R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In Foundations of Secure Computation, pp. 169-180, 1978.

[7] C. Gentry. Fully homomorphic encryption using ideal lattices. STOC 2009, pp. 169-178, 2009.

[8] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. PKC 2010, LNCS 6056, pp. 420–443.

[9] D. Stehle and R. Steinfeld. Faster Fully Homomorphic Encryption. Asiacrypt 2010, LNCS 6477, pp. 377-394.

[10] Craig Gentry and Shai Halevi. Implementing Gentry’s fully-homomorphic encryption scheme. Eurocrypt 2011, LNCS 6632, pp. 129–148.

[11] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, Journal of the ACM (JACM), 56(6), pp. 1-40, 2009.

[12] V. Lyubashevsky and C. Peikert and O. Regev. On Ideal Lattices and Learning with Errors over Rings. In Proc. of Eurocrypt, volume 6110, pp. 1–23, 2010.

[13] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In STACS, pp. 75–86, 2009.

[14] M. Ajtai. Generating Hard Instances of the Short Basis Problem. LNCS, Volume 1644, 1999, pp. 1-9.

[15] C. Gentry and S. Halevi and V. Vaikuntanathan. A Simple BGN-type Cryptosystem from LWE. In Proc. of Eurocrypt, volume 6110, pp. 506-522, 2010.

[16] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. STOC 2009, pp. 333-342, 2009.

[17] P.Q. Nguyen and D. Stehle, LLL on the average, proc. Of ANTS VII, 2006, LNCS 4076, pp. 238-256.

[18] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In CRYPTO, 2011. To appear.

[19] Zvika Brakerski and Vinod Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. ePrint Archive: Report 2011/344: http://eprint.iacr.org/2011/344.

[20] Zvika Brakerski, C. Gentry and Vinod Vaikuntanathan. Fully Homomorphic Encryption without Bootstrapping. ePrint Archive: Report 2011/279: http://eprint.iacr.org/2011/277.