Fully homomorphic encryption

Zhunzhun Chen

University of Wollongong

Follow this and additional works at: https://ro.uow.edu.au/theses1

Recommended Citation

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au
FULLY HOMOMORPHIC ENCRYPTION

A Thesis Submitted in Partial Fulfilment of
the Requirements for the Award of the Degree of

Master of Philosophy

from

UNIVERSITY OF WOLLONGONG

by

Zhunzhun CHEN

BEng, Mphil

School of Computer Science and Software Engineering
Faculty of Informatics

LaTeX, Document Class, Thesis, UoW Thesis Class
CERTIFICATION

I, Zhunzhun CHEN, declare that this thesis, submitted in partial fulfilment of the requirements for the award of Master of Philosophy, in the School of Computer Science and Software Engineering, Faculty of Informatics, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications at any other academic institution.

(Signature Required)
Zhunzhun CHEN
date LaTeX, Document Class, Thesis, UoW Thesis Class
Dedicated to

My Students
and
Anyone Who May Love \LaTeX
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>iv</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>2 Background</strong></td>
<td>8</td>
</tr>
<tr>
<td>2.1 Notation</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Lattice</td>
<td>9</td>
</tr>
<tr>
<td>2.2.1 Definition</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Lattice Computational Problems</td>
<td>11</td>
</tr>
<tr>
<td>2.3.1 Learning With Error (LWE)</td>
<td>12</td>
</tr>
<tr>
<td>2.3.2 Ring-Learning With Error (R-LWE)</td>
<td>14</td>
</tr>
<tr>
<td>2.4 Public Key Cryptography</td>
<td>15</td>
</tr>
<tr>
<td>2.5 Fully Homomorphic Encryption</td>
<td>17</td>
</tr>
<tr>
<td>2.5.1 The Definition of FHE</td>
<td>18</td>
</tr>
<tr>
<td>2.5.2 The Construction of FHE</td>
<td>19</td>
</tr>
<tr>
<td>2.5.3 The Security of FHE</td>
<td>19</td>
</tr>
<tr>
<td>2.5.4 Technique of Fully Homomorphic Encryption</td>
<td>17</td>
</tr>
<tr>
<td>2.5.5 Fully Homomorphic Encryption Schemes</td>
<td>24</td>
</tr>
<tr>
<td><strong>3 The Construction of Fully Homomorphic Encryption Scheme</strong></td>
<td>40</td>
</tr>
<tr>
<td>3.1 Our Scheme with Compression Public Key</td>
<td>40</td>
</tr>
<tr>
<td>3.1.1 SHE Scheme</td>
<td>40</td>
</tr>
<tr>
<td>3.1.2 Correctness of Somewhat Homomorphic Encryption</td>
<td>42</td>
</tr>
<tr>
<td>3.2 The Security</td>
<td>43</td>
</tr>
<tr>
<td>3.2.1 Leftover Hash Lemma</td>
<td>44</td>
</tr>
<tr>
<td>3.2.2 Semantic Security</td>
<td>44</td>
</tr>
<tr>
<td>3.3 FHE Scheme</td>
<td>45</td>
</tr>
<tr>
<td>3.3.1 The squashed Scheme</td>
<td>45</td>
</tr>
<tr>
<td>3.3.2 Bootstrapping</td>
<td>48</td>
</tr>
<tr>
<td>3.3.3 Security of The Squashed Scheme</td>
<td>50</td>
</tr>
<tr>
<td>3.4 Attacks</td>
<td>50</td>
</tr>
<tr>
<td>3.4.1 Brute Force Attack</td>
<td>50</td>
</tr>
<tr>
<td>3.4.2 Birthday Attack</td>
<td>51</td>
</tr>
</tbody>
</table>
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.3 SDA-Simultaneous Diophantine Approximation</td>
<td>51</td>
</tr>
<tr>
<td>3.4.4 Nguyen and Stern's Orthogonal Lattice</td>
<td>53</td>
</tr>
<tr>
<td>3.4.5 Coppersmith’s Method</td>
<td>54</td>
</tr>
<tr>
<td>3.4.6 BDD-Bounded Distance Decoding</td>
<td>54</td>
</tr>
<tr>
<td>3.5 Extension of The HIL Encryption to Higher Degree</td>
<td>55</td>
</tr>
<tr>
<td>3.6 Parameters and Constraints</td>
<td>56</td>
</tr>
<tr>
<td>4 Conclusion and Future Work</td>
<td>59</td>
</tr>
<tr>
<td>References</td>
<td>68</td>
</tr>
</tbody>
</table>
ABSTRACT

The notion of a fully homomorphic encryption scheme over integers with public key compression has been proposed by Coron. The main attractive feature of this scheme is the reduction of the public key size, which is obtained by encrypting the plaintext with a quadratic form in the public key elements instead of in a linear form. In this work, we adopt this technique and apply it to the hidden ideal lattice scheme to acquire a more efficient scheme based on the hidden ideal lattice. The security of our scheme is based on the bounded distance decoding over the hidden ideal lattice. Additionally, we also describe a variant of the scheme with higher degrees. The scheme shows a better level of efficiency in comparison to the original scheme.
Acknowledgements

I would have not finished this project without the support of my family who has always been there for me whenever I need them, the encouragement they give to keep me going and their love to empower me that never fails all the time. I would like to thank my supervisors Willy SUSILO and Thomas PLANTARD who have given me a chance to prove that i can do things on my own. They gave me a lot of positive perspective in life. They who taught me things far more of my understanding. I thank them for challenging me to do this project. Thank you.
Chapter 1

Introduction

Post-quantum cryptography aims to construct cryptographic algorithms which is secure against an attack by a quantum computer. There are three algorithms with their security relies on hard mathematical problems: the integer factorization problem, the discrete logarithm problem or the elliptic-curve discrete logarithm problem. Under the current electronic computer, these algorithms are hard to solve, however, they can be solved by quantum computer easily by running Shor’s algorithm \cite{60}. Symmetric cryptographic algorithms and hash functions in the public-key system are considered to be relatively secure against attacks by quantum computers \cite{2}.

Post-quantum cryptography is using quantum phenomena to achieve secrecy and detect eavesdropping. There are six categories of algorithms based on different hard mathematical problems.

- Latticed-based cryptography: the cryptographic system includes Learning with Errors, Ring-Learning with Errors (Ring-LWE), the Ring Learning with Errors Key Exchange and the Ring Learning with Errors Signature, the older NTRU or GGH encryption schemes, and the newer NTRU signature and BLISS signatures \cite{49}.  

• Multivariate cryptography: this type of cryptographic system includes the Rainbow scheme \cite{21}. Rainbow also could provide a quantum secure multivariate signature schemes which called Rainbow Signature Scheme \cite{21}.

• Hash-based cryptography: the cryptographic system includes Lamport signatures and the Merkle signature scheme. The most familiar hash based digital signatures like RSA and DSA.

• Code-based cryptography: the cryptographic system is based on error-correcting codes, the McEliece and Niederreiter encryption algorithms are two classic algorithms \cite{17}.

• Supersingular elliptic curve isogeny cryptography: the cryptographic system relies on the properties of supersingular elliptic curves. to create a Diffie-Hellman replacement with forward secrecy. Diffie-Hellman like key exchange has better performance to resist quantum computing than the Diffie-Hellman and elliptic curve Diffie-Hellman key exchange methods \cite{23}.

• Symmetric key-based cryptography: Grover’s algorithm is the best quantum attack against generic symmetric-key systems. These approach is more effective in small key size for post-quantum cryptography \cite{34}.

To prove the security of a cryptographic algorithm is equivalent to prove the mathematical problem is hard. The procedure of proves is called "security reductions". The security of given cryptographic algorithms above are reduced to the security of different known hard problems.

• Ring-LWE Signature: the security reduction of RLWE is the shortest-vector problem (SVP) in a lattice as a lower bound on the security which is NP-hard problem \cite{30}. 
• NTRU, BLISS: the security reduction of these two algorithms are the closest-vector problem (CVP) in a lattice as a lower bound on the security which is also NP-hard problem [22].

• Rainbow: multivariate quadratic equation cryptosystems called ”Unbalanced Oil and Vinegar Cryptosystems” is NP-hard problem. The Rainbow Multivariate Equation Signature Scheme is a class of multivariate quadratic equation cryptosystems. The Rainbow Multivariate Equation Signature Scheme is equivalent to NP-hard problem [8].

• Merkle signature scheme: one-way hash functions is a well known hard problem. The security reduction of Merkle Hash Tree signatures has proved to relies on one-way hash function [24].

• McEliece: the Syndrome Decoding Problem (SDP) is also known to be NP-hard problem. The security reduction of McEliece Encryption System is SDP [59].

• Supersingular elliptic curve isogeny cryptography: Unlike other cryptosystems, this system has no security reduction to a known NP-hard problem. Delfs and Galbraith indicates the difficulty of the problem is as hard as the inventors of the key exchange which relies on constructing an isogeny between two supersingular curves with the same number of points [18].

A lattice $\mathcal{L}$ in real analysis is a set of points in the n-dimensional Euclidean space $\mathbb{R}^n$ with a strong periodicity property. A basis of $\mathcal{L}$ is a set of vectors is represented by the linear combination of any element with integer coefficients uniquely. Due to the property of the cryptosystems, the ciphertext, public key, and private key must be taken from a finite space, therefore, the lattices used for cryptography is over the finite field only. The most two famous mathematical problems based on the lattices are the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP) [1].
Both are hard to solve without a good basis, the security of algorithm relies on the hard problem to find the good basis. The effective method to find the good basis (nearly orthogonal vectors) is using lattice basis reduction. If one can compute such a lattice basis, the CVP and SVP problems are easy to solve. The LLL algorithm is a quite effective to compute good basis, and so many alternative algorithms based on LLL algorithm to run faster or more efficiency [53].

The notion of fully homomorphic encryption scheme has been known to be very useful in the cloud computing environment, ciphertext retrieval and secure multiparty computation [13]. In 1978, Rivest, Adleman and Dertouzos [55] introduced the basic concept of privacy homomorphism which allows computation on encrypted data without decryption. They posed the construction of privacy homomorphism (and hence, fully homomorphic encryption) as an open research problem. A scheme is called to be fully homomorphic if it can operate on the ciphertext without the knowledge of the secret key. For any valid function $f$ and plaintext $m$, the operation on ciphertexts is equivalent to the same operation on plaintext. In such a definition, given a function $f$ and a ciphertext $c$ which encrypt a plaintext $m$, it is able to transfer $c$ into a new ciphertext $c'$ which encrypts $f(m)$. There have been many attempts to achieve this goal. Some of them can satisfy the additive homomorphism only or the multiplicative homomorphism only, and meanwhile some other schemes have been successful on enabling both operations with limited level of operations. The 'Polly Cracker' scheme can evaluate arbitrary level operation in any circuit. Nevertheless, the size of ciphertext will increase exponentially with the depth of the circuit [64]. We note that none of these scheme is fully homomorphic scheme. The first breakthrough has been provided by Gentry in his construction of the first fully homomorphic encryption in 2009 [25]. Due to the characteristic of the addition and multiplication over $\mathbb{Z}_2$, which forms a complete set of those operations, the scheme can evaluate the operation
on encrypted data in polynomial time.

Gentry's approach on achieving fully homomorphic encryption is achieved by incorporating the bootstrapping technique, which seems to be the inherent efficiency bottleneck [28]. This is the primary reason why fully homomorphic encryption scheme cannot be adopted in practice yet. The natural fully homomorphic encryption scheme has not been found so far, majority schemes are proposed by Gentry’s first idea: constructed a somewhat homomorphic encryption scheme first, then applied the squash on the decryption algorithm, finally used the bootstrapping technique to achieve the fully homomorphic encryption scheme [26].

To construct a somewhat homomorphic encryption scheme means construct a scheme with a limited number of homomorphic operations. The somewhat homomorphic of Gentry’s framework is s GGH cryptosystem which based on the ideal lattice. There are two kinds of basis, one is ‘good’ basis which can be used as the secret key, another basis is ‘bad’ for the public key [26]. The underlying lattice problem is a bounded distance decoding problem over ideal lattice. The encryption is mapping a message to a vector close to the lattice by using the bas basis witch is public key, the decryption is reducing the vector to the message buy using the good basis which is secret key.

During the evaluations, since the noise of the ciphertext is expanded over the bound especially in multiplicative operation, it occurs the failure in the decryption. Gentry used ‘homomorphic decryption’ to control the noise increasing. Encrypt the ciphertext and the corresponding public key by evaluate key, and input the result into the decryption circuit, output a new ciphertext. If the error of ciphertext is able to evaluate one more time especially in multiplication after each operation, then the ciphertext can perform unlimited times operation [28], [30]. Since the somewhat homomorphic encryption scheme can only perform limited operations with low-degree
polynomials, the next step is to squash the decryption procedure so that it can be expressed as a low-degree polynomial which is supported by the scheme. Finally the application of a bootstrapping can transform the somewhat homomorphic encryption scheme to a fully homomorphic scheme \[26\].

There are three categories of fully homomorphic encryption scheme: ideal lattice based scheme, integer based scheme and learning with error based scheme. Smart and Vercauteren \[62\] used the principle ideal lattice to construct the fully homomorphic scheme. They selected two integer to represent the lattice and maintained a smaller key size. The integer based scheme proposed by van Dijk \[20\], where its security is based on the approximate greatest common factor. Plantard, Susilo and Zhang \[50\] proposed the notion of hidden ideal lattice for the construction of fully homomorphic encryption schemes. The hidden ideal lattice scheme unifies ideal lattice scheme and integer scheme. Instead of publishing the lattice, they used vectors close to a lattice which is called the hidden ideal lattice. The security of the hidden ideal lattice scheme relies on a bounded distance decoding problem over hidden ideal lattice rather than the subset sum problem.

The implementation of the fully homomorphic encryption scheme by van Dijk et al. \[20\] shows that the public key size is too big for any practical system \[29\]. Reducing the size of the public key is the key point to make the scheme more practical, which is achieved by shortening the length or decreasing the number of public keys \[57\]. Coron proposed a technique to reduce the number of public, then shrunk the size of the public key based on the scheme over the integers \[15\].

**Our Contributions** Since Coron’s scheme is over the integers, his work can be reduced to the AGCD problem, the attacker can recover the noise or public key by lattice reduction. We choose Plantard, Susilo and Zhang’s hidden ideal lattice scheme to combine the bounded distance decoding problem (BDD) with approximate greatest
common divisor problem. Therefore, the scheme based on the ideal lattice gives a stronger security by the hardness of problems. Coron’s technique can be applied on the Planard, Susilo and Zhang’s hidden ideal lattice scheme, which improves the efficiency exponentially by smaller size of public key. The less public key we publish, the less information of the public key or noise will be leaked. In this work, we are to construct a somewhat homomorphic scheme with public key compression on hidden ideal lattice. Our approach is summarized as follows. We first generate a random polynomial vector as the ring element, then divide these vectors into two groups. Then, we choose a vector from each group and the product of two polynomial. Therefore, the original public key will be replaced by the new quadratic key. The scheme can reduce the number of public key from $\tau$ keys to $2\sqrt{\tau}$ keys. We also extend the technique into higher degrees to reduce the public key size further. The efficiency can improve for shorter time consuming and less space in public key storage.
Chapter 2

Background

In this section, we will provide the background of research on three disciplines: Lattice, Public-Key Cryptosystems, and Fully Homomorphic Encryption.

2.1 Notation

The parameters that are used in the scheme are as follows:

- $\lambda$: security parameter.
- $\rho$: the norm of random noise vector.
- $\eta$: the bit length of the norm of generating polynomial (secret vector).
- $\gamma$: the bit length of the norm of the random multiplier vector.
- $\tau$: the number of vectors in the public key in encryption algorithm.
- $\beta$: the number of vectors in the public key.
- $\zeta$: the norm of noise used in encryption.
- $n$: the dimension of the hidden lattice.
2.2. Lattice

- \( \theta \): the constant factor depending on the polynomial.

For integers \( z \) and \( d \), denote \([z]_d\) as the \( z \) mod \( d \) with in \((-d/2, d/2]\) and \([z]\) as the closest integer to \( d \). Recall the definition of the integer residue ring \( \mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z} \), the element in the residue ring is generated by modular operation which in the set \{0, 1, 2, \ldots, n-1\}. For \( x \) Mod \( n \equiv y \), \( y \) is defined as \( y \equiv x \mod n \) with the interval \((-n/2, n/2]\). \( D \) is a distribution by parameter \( \gamma \) and \( \rho \), \( D_{\gamma, \rho}(p) := \{\text{choose } q \leftarrow \mathbb{Z} \cap [0, q_0), e \leftarrow \mathbb{Z} \cap (-2\rho, 2\rho) : \text{output } x = pq + e\} \).

Denote the vector \( v \) to represent the coefficient of polynomial \( f(x) \). Let the polynomial \( f(x) \) in the form of \( \text{Vec}(f(x)) = \sum_{i=0}^{n-1} v_i x^i \), denote vector \( v = \langle v_1, \ldots, v_n \rangle \), where \( v_i \) represents the coefficient of element of \( x^i \). For two vectors \( v_1 \) and \( v_2 \), denote \( v_1 \times v_2 \) be the polynomial multiplication over the ring, \( v_1 \times v_2 = \text{Vec}(v_1(x) \times v_2(x) \mod f(x)) \).

2.2 Lattice

Based on introduction of lattices by Micciancio, Nguyen and Lenstra, this section will give the definition, proofs and properties of lattice.

2.2.1 Definition

Let \( \mathbb{R}^n \) be the \( n \)-dimensional Euclidean vector. \( x \) and \( y \) are denoted as column vectors like \( x = (x_1, \ldots, x_n)^T \) and \( y = (y_1, \ldots, y_n)^T \) in \( \mathbb{R}^n \). The Euclidean inner product is denoted by \( <x, y> = \sum_{i=1}^{n} x_i y_i \), and the corresponding norm is \( ||x|| = \sqrt{x_1^2 + \cdots + x_n^2} \).

The distance between two vectors is \( d(x, y) = ||x - y|| \). The distance between a vector \( x \in \mathbb{R}^n \) and a subset \( E \subset \mathbb{R}^n \) is defined as \( \text{dist}(x, E) = \min_{y \in E} \{d(x, y)\} \).

Definition 2.2.1. (Lattice)\([53]\) Lattice is the set of integer combinations of \( n \) linearly independent vectors \( v_1, \ldots, v_n \) in \( \mathbb{R}^n \). Denote the set of vectors \( v_1, \ldots, v_n \) as the basis
of the lattice.

\[ \mathcal{L}(v_1, \ldots, v_n) = \{ \sum_{i=1}^{n} v_i b_i : v_i \in \mathbb{Z} \text{ for } 1 \leq i \leq n \} \]

In the matrix notation, \( \mathbf{B} = [v_1, \ldots, v_n] \in \mathbb{R}^{n \times n} \) denotes as the basis for lattice \( \mathcal{L}(\mathbf{B}) = \{ \mathbf{B} \mathbf{x} : \mathbf{x} \in \mathbb{Z}^n \} \). The determinant of a lattice is \( \det(\mathcal{L}) = \sqrt{\mathbf{B} \times \mathbf{B}^T} \).

**Definition 2.2.2. (Norm)** \[53\] Let \( v = <v_1, \ldots, v_n> \in \mathbb{R}^n \) be the vector of lattice, the Euclidean norm is defined as \( \|v\| = \sqrt{\sum_{i=1}^{n} v_i^2} \). For two vectors \( v_1, v_2 \in \mathbb{R}^n \), we have \( \|v_1 + v_2\| \leq \|v_1\| + \|v_2\| \), and \( \|v_1 \times v_2\| \leq \theta \cdot \|v_1\| \cdot \|v_2\| \), where \( \theta = \sqrt{n} \).

**Definition 2.2.3. (Ideal Lattice)** \[42\] An ideal lattice \( \mathcal{L}(\text{Rot}(v,f)) \) over a polynomial ring \( \mathbb{Z}[X]/f, f \in \mathbb{Z}[X] \) is a monic irreducible polynomial of degree \( n \). \( \text{Rot}(v,f) \) is the rotation matrix where the \( i \)-th row of this matrix equals to the coefficients of \( v \times x_i^{r-1} \mod f \).

Plantard, Susilo and Zhang constructed a fully homomorphic encryption scheme by using hidden ideal lattice \[50\]. The hidden ideal lattice scheme is unified by two schemes which are ideal lattice based schemes and integer based schemes. The security of the scheme does not rely on the sparse subset sum problem (SSSP), but rather, it relies on the bounded distance decoding problem (BDD) of ideal lattices.

**Definition 2.2.4. (Hidden Ideal Lattice)** \[50\] Let \( \alpha \in \mathbb{R}^+ \) be a positive real, \( v_i \in \mathbb{Z}^n \) be \( \tau \) integer vectors such that there exists a unique (ideal) lattice \( \mathcal{L} \) and some unique vectors \( v_i \in \mathcal{L} \) respecting \( \forall 1 \leq i \leq \tau, \ dist(v_i, w_i) \leq \alpha \). Then \( \mathcal{L} \) is called an \( \alpha \)-hidden ideal lattice hidden under \( \{v_i\} \). For a vector \( v \in \mathbb{R}^n \) and a lattice \( \mathcal{L} \), the distance between the two, denoted by \( \text{dist}(v, \mathcal{L}) = \min(\|v - u\|), \forall u \in \mathcal{L} \).

**Definition 2.2.5. (BDD over Hidden Ideal Lattice)** \[50\] Let \( \gamma \in \mathbb{R}^+ \) be a positive real. Let \( \mathcal{L} \) be an \( n \) dimensional ideal lattice, and \( v \in \mathbb{Z} \) such that there exists a unique vector \( u \in \mathcal{L} \) satisfying \( \text{dist}(v, u) \leq \gamma \). The \( \gamma \)-Bounded Distance Decoding problem over ideal lattice, denoted by \( \gamma - BDDHI_n \), is to find \( u \), given a basis of \( \mathcal{L} \) and \( v \).
2.3. Lattice Computational Problems

Definition 2.2.6. (Dec BDD over Hidden Ideal Lattice) [50] Let $\gamma \in \mathbb{R}^+$ be a positive real. Let $L$ be an $n$ dimensional ideal lattice, and $v \in \mathbb{Z}$. The decisional $\gamma$-Bounded Distance Decoding problem over ideal lattice, denoted by Dec $\gamma - BDDHI_n$, is to decide if there exists a unique vector $u \in L$ satisfying $\text{dist}(v, u) \leq \gamma$. or not, given a basis of $L$ and $v$.

Definition 2.2.7. (Subset Sum Problem) [50] Let $\{c_1, c_2, \ldots, c_n\}$ be a set of positive integers. Let $c = \sum_{i=1}^{n} s_i c_i$, where $s_i \in \{0, 1\}$. Let $d \leftarrow \sum_{i=1}^{n} s_i$. The subset sum problem, denoted by $d, n$-SSP, is to find $\{s_i\}$, given $\{c_i\}$ and $c$.

2.3 Lattice Computational Problems

To prove the security of algorithms, we introduce several classic computational problems which based on lattices computation in this section. Then we review the lattice application on cryptosystem with LWE and R-LWE.

Definition 2.3.1. (Classic Lattice Problems) [41], [51]

- **Decisional Shortest Vector Problem** ($\text{GapSVP}_\gamma$): Given a basis $B$ of a full-rank $n$-dimensional lattice $\mathcal{L}$, decide if $\lambda_1(\mathcal{L}) \leq 1$ or $\lambda_1(\mathcal{L}) > \gamma(n)$.

- **Shortest Independent Vectors Problem** ($\text{SIVP}_\gamma$): Given a basis $B$ of a full-rank $n$-dimensional lattice $\mathcal{L}$, find a set of linearly independent vectors $S = \{s_1, \ldots, s_n\}$, for $s_i \in \mathcal{L}(B)$, minimizing the quantity $\|S\| = \max_{1 \leq i \leq n} \|s_i\|$.

Definition 2.3.2. (Modern Lattice Problems) [51], [53]

- **Small Integer Solutions** ($\text{SIS}_\beta$): Given a prime $q$, a random matrix $A \in \mathbb{Z}_{q}^{n \times m}$ and a real number $\beta$. Find a non-zero vector $d \in \mathbb{Z}^m$ such that $Ad = 0 \mod q$ and $\|d\| \leq \beta$. Note, finding a solution of $\text{SIS}_\beta$ can be seen as finding a short lattice point in lattice $\Lambda^\perp_q(A)$. 
Lemma 2.3.1. (Average-case to Worst-case) [44], [52] Let \( n, p \geq 1 \) be some integers and \( \chi \) be some distribution on \( \mathbb{Z}_p \). Assume that we have access to a distinguisher \( W \) that distinguishes \( A_{s,\chi} \) from \( U \) for a non-negligible fraction of all possible \( s \), then there exists an efficient algorithm \( W' \) that for all \( s \) accepts with probability exponentially close to 1 on inputs from \( U \).

Lemma 2.3.2. (Decision to Search) [52] Let \( n \geq 1 \) be some integers, \( 2 \leq p \leq \text{poly}(n) \) be a prime, and \( \chi \) be some distribution on \( \mathbb{Z}_p \). Assume that we have access to procedure \( W \) that for all \( s \) accepts with probability exponentially close to 1 on inputs from \( A_{s,\chi} \) and rejects with probability exponentially close to 1 on inputs from \( U \). Then, there exists an efficient algorithm \( W' \) that, given samples from \( A_{s,\chi} \) for some \( s \), outputs \( s \) with probability exponentially close to 1.

Lemma 2.3.3. (Discrete to Continuous) [52] Let \( n, p \geq 1 \) be some integers, let \( \phi \) be some probability density function on \( T \), and let \( \bar{\phi} \) be its discretisation to \( \mathbb{Z}_p \). Assume that we have access to an algorithm \( W \) that solves \( \text{LWE}_{p,\bar{\phi}} \). Then, there exists an efficient algorithm \( W' \) that solves \( \text{LWE}_{p,\phi} \).

2.3.1 Learning With Error (LWE)

Based on worst-case hardness assumption, the “learning with error” problem is a classical problem which is to distinguish random linear equations with small amount of noise from uniform ones. The main theory is to recover a secret key \( s \in \mathbb{Z}_q^n \) given a sequence of ”approximate” random linear equations on \( s \). If without error, the secret key \( s \) can be found by gaussian elimination in polynomial time. Introducing the small error perturbs the linear combinations into nonlinear combinations, then the gaussian elimination algorithm seems impossible to solve the problem directly.

Definition 2.3.3. (GLWE) [5] For security parameter \( \lambda \), let \( n = n(\lambda) \) is the dimension, and the polynomial \( f(x) = x^d + 1 \) with \( d \) is power of 2, fix \( q = q(\lambda) \leq 2 \) as a prime
integer, let $R = \mathbb{Z}[x]/(f(x))$ and $R_q = R/qR$, and let $\chi = \chi(\lambda)$ be a distribution over $R$. The $GLWE_{n,f,q,\chi}$ problem is to distinguish the following two distributions:

In the first distribution, one samples $(a_i, b_i)$ uniformly from $R_q^{n+1}$. In the second distribution, one first draws $s \leftarrow R_q^n$ uniformly and then samples $(a_i, b_i) \in R_q^{n+1}$ by sampling $a_i \leftarrow R_q^n$ uniformly, $e_i \leftarrow \chi$, and setting $b_i = <a_i, s> + e_i$. The $GLWE_{n,f,q,\chi}$ assumption is that the $GLWE_{n,f,q,\chi}$ problem is infeasible.

LWE is simply GLWE instantiated with $d = 1$ and RLWE is GLWE instantiated with $n = 1$. The brief description of the LWE problem is, given a size parameter $n \geq 1$ and a modulus $q \geq 2$, also given an error probability distribution $\chi$ on $\mathbb{Z}_q$. Choose a random vector $\alpha \in \mathbb{Z}_q^n$ uniformly and $e \in \mathbb{Z}_q$ according to $\chi$, then output $(\alpha, <\alpha, s> + e)$ in $\mathbb{Z}_q$. The $A_{\alpha,\chi}$ consists of independent and uniform random $\alpha$. The LWE problem can be considered as decoding from random linear codes. On the lattice view, the LWE problem is decoding the code by a random bounded distance \cite{38}. The maximum likelihood algorithm is a way to solve the LWE problem with the running time $2^{O(n \log n)}$. The best known algorithm for the LWE problem is Blum et al. algorithm, and the running time is $2^{O(n)}$ \cite{9}, \cite{32}, \cite{16}, \cite{58}. Due to several reasons, we believe the LWE problem is hard. The well known algorithm for solving the LWE problem is running in exponential time. Next, the LWE problem is based on certain assumptions regarding the worst-case hardness of standard lattice problems such as GapSVP and SIVP on lattices. The ‘dual’ problem of LWE problem is SIS problem. The hardness of the SIS problem is also based on the worst-case lattice problem such as SIVP and GapSVP \cite{13}, \cite{14}, \cite{38}.

**Theorem 2.3.4.** (Informal) \cite{52} Let $n, p$ be integer and $\alpha \in (0, 1)$ be such that $\alpha p > 2\sqrt{n}$. If there exists an efficient algorithm that solves $LWE_{p,\tilde{\Psi}}$ then there exists an efficient quantum algorithm that approximates the decision version of the shortest vector problem (GapSVP) and the shortest independent vectors problem (SIVP) to
within $\tilde{O}(n/\alpha)$ in the worst case.

### 2.3.2 Ring-Learning With Error (R-LWE)

Normal LWE problem is defined over integers. It turns out the LWE problem over some special rings is also hard. Due to the large size of key over the integers, the ring structure allows LWE-based cryptosystems to have shorter public key size, which is reduced to almost linear size from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ \cite{10}. The structure in the Ring-LWE of the NTRU cryptosystem is as follow. Choose a random vector $\alpha_1 = (a_1, a_2, \ldots, a_n)$ uniformly on the ring $\mathbb{Z}_q[x]/<x^n+1>$, the rest of $n-1$ vectors $(a_2, \ldots, a_n)$ in the form of $a_i = (a_i, a_{i+1}, \ldots, a_n, -a_1, \ldots, -a_{i-1})$ with $n$ is the power of 2. Some schemes do not require the $x^n + 1$ has to be irreducible over the rational recently \cite{10}. Fix the $s \in \mathbb{R}_q$ as the secret key, $\alpha \in \mathbb{R}_q$ chosen uniformly, and $e$ is an error chosen from the distribution over $\mathbb{R}_q$. The sample forms as $(\alpha, b = \alpha \cdot s + e) \in \mathbb{R}_q \times \mathbb{R}_q$, where each $\alpha$ is uniformly random and each inner product $\alpha \cdot s$ is perturbed by a term draw independently from the error distribution over $\mathbb{R}$ \cite{10}.

The hardness of the Ring-LWE problem is based on the worst-case lattice problem. The goal is to recover the secret $s$ from these samples. The main theory \cite{18} is

**Theorem 2.3.5.** Suppose that it is hard for polynomial-time quantum algorithms to approximate (the search version of) the shortest vector problem (SVP) in the worst case on ideal lattices in $\mathbb{R}$ to within a fixed poly($n$) factor. Then any poly($n$) number of samples drawn from the R-LWE distribution are pseudorandom to any polynomial-time attacker.
2.4 Public Key Cryptography

Public key cryptography is asymmetric cryptography. Any cryptographic system creates public key and private key in pairs. Public keys can be distributed widely, and private keys are known to the owner only. Everyone can encrypted plaintext by public key, and the ciphertext can be decrypted by owner’s private key. The strength of a public key cryptography system relies on the difficulty of generating a private key from its corresponding public key. Security depends only on keeping the private key private, and the public key may be published without compromising security[61]. Public key cryptographic algorithms are based on mathematical problems, such as integer factorization, discrete logarithm, and elliptic curve relationships.

Diffie and Hellman [19] introduced a new approach for distributing the key information over public insecure channels. Public key cryptosystem also known as asymmetric key cryptosystem. Each user has two keys, one is a public encryption key another is a private decryption key. The public key is generated by secret key with different mathematical techniques, while the private key cannot be derived from the public key. In other words, the public key is widely known and the secret key is only known to the recipients. Messages are encrypted with the recipient’s public key and can only be decrypted with the corresponding private key. The first implementation of the idea is published by Rivest, Shamir and Adleman in 1977. It is also known as RSA algorithm [56].

The LWE problem has been widely used in the public key cryptosystem. There is an example of the application of LWE with public key cryptosystem, but the efficiency needs to be improved [54].

- **Private Key**: Choose a random vector $s$ uniformly from $\mathbb{Z}_q^n$.

- **Public Key**: The public key consists of $m$ samples $(\alpha_i, b_i)_{i=1}^{m}$ from the LWE
distribution with secret $s$, modulus $q$, and error parameter $\alpha$.

- **Encryption**: For each bit of the message, do the following. Choose a random set $S$ uniformly among all $2^m$ subsets of $[n]$. The encryption is $(\sum_{i \in S} \alpha_i, \sum_{i \in S} b_i)$ if the bit is 0 and $(\sum_{i \in S} \alpha_i, \lfloor \frac{q}{2} \rfloor + \sum_{i \in S} b_i)$ if the bit is 1.

- **Decryption**: The decryption of a pair $(\alpha, b)$ is 0 if $b - <\alpha, s>$ is closer to 0 than to $\lfloor \frac{q}{2} \rfloor$ modulo $q$, and 1 otherwise.

The correctness is obviously. It requires the error of $s$ smaller than $q/4$, since each error’s standard deviation is $\alpha q$, the standard deviation of the sum is at most $\sqrt{maq} < q/\log n$. The security is based on the decision-LWE problem. With very high probability over the choice of $(\alpha_i, b_i)_{i=1}^m$, the distribution over $s$ of a random subset sum $(\sum_{i \in S} \alpha_i, \sum_{i \in S} b_i)$ is extremely close to be uniform in statistical distance. The encryptions of 0 or 1 is essentially identically distributed, the algorithm assuming decision LWE is hard $[32], [48]$

The R-LWE is more widely and efficiently in real world. There is an example of public key cryptosystem satisfying the semantically secure $[40]$. Fix the ring $R = \mathbb{Z}[x]/<x^n + 1>$ with the $n$ is a power of 2.

- **Secret Key**: Choose a random vector $s$ uniformly from $\mathbb{R}$.

- **Public Key**: Then choose a uniformly random ring element $\alpha \in \mathbb{R}_q$, and another small element $e \in \mathbb{R}$ from the error distribution. Output a pair $(a, b = a \cdot s + e)$ as the public key.

- **Encryption**: To encrypt $n$-bits binary message, choose three random elements $r, e_1, e_2 \in \mathbb{R}$ from the error distribution. Output the cyphertext in pair $(u, v) \in \mathbb{R}_q^2$, where
  \[
  u = a \cdot r + e_1 \text{ and } v = b \cdot r + e_2 + \lfloor \frac{q}{2} \rfloor \cdot z \mod q
  \]
2.5. Fully Homomorphic Encryption

2.5.1 The Definition of FHE

There are four algorithms: KeyGen, Encrypt, Decrypt and Evaluate. The first three are the basic algorithm of public key encryption system and the evaluation algorithm is the core algorithm of fully homomorphic encryption which performing the operation of ciphertext. The algorithm is inputting a group ciphertext \( c =< c_1, c_2, \ldots, c_t > \) which has been encrypted with plaintext into a circuit \( C \), each circuit \( C \) represents a function. The group of new output ciphertext can be decrypted to the corresponding plaintext.

**Definition 2.5.1.** (Correct Homomorphic Decryption) [25] For any key-pair \((pk, sk)\) generated by algorithm KeyGen, any \( t \)-input circuit, any plaintext \( m_1, \ldots, m_t \) and any ciphertext \( c_1, \ldots, c_t \) with \( c_i \leftarrow \text{Encrypt}(m_i) \), if

\[
\text{Decrypt}(sk, \text{Evaluate}(pk, C, c)) = C(m_1, \ldots, m_t)
\]

exists, the scheme \( \mathcal{E} = (\text{KeyGen, Encrypt, Decrypt, Evaluate}) \) is correct.

**Definition 2.5.2.** (Compact Homomorphic Decryption) [26] For any security parameter \( \lambda \), if there exists a polynomial \( f \), output length of the evaluate algorithm be most

\[
v - u \cdot s = (r \cdot e - s \cdot e_1 + e_2) + \lfloor q/2 \rfloor \cdot z \mod q.
\]

This application also requires the coefficient of the error item \( r \cdot e - s \cdot e_1 + e_2 \in \mathbb{R} \) less than \( q/4 \) [10].
2.5. Fully Homomorphic Encryption

\( f(\lambda) \). The scheme \( \mathcal{E} = (\text{KeyGen, Encrypt, Decrypt, Evaluate}) \) is compactness.

**Definition 2.5.3.** (Fully Homomorphic Decryption) \([26]\) A scheme \( \mathcal{E} \) is fully homomorphic if it is both compact and homomorphic for the class of all arithmetic circuit.

**Definition 2.5.4.** (Somewhat Homomorphic Decryption) \([26]\) The encryption scheme can handle circuits of depth roughly \( \log \log N - \log \log n \), which means the maximum depth of the permit circuits is greater than twice of the depth of decryption circuit.

**Definition 2.5.5.** (Fully Homomorphic Decryption) \([26]\) A scheme \( \mathcal{E} \) is fully homomorphic if it is both compact and homomorphic for the class of all arithmetic circuit.

**Definition 2.5.6.** (Leveled Fully Homomorphic Decryption) \([26]\) For any \( d \in \mathbb{Z}^+ \), the scheme \( \mathcal{E}^{(d)} \) with the same decryption circuit is compactness and the depth of circuit is \( d \). The complexity of the scheme \( \mathcal{E}^{(d)} \) is polynomial in \( \lambda, d \). A family of schemes \( \mathcal{E}^{(d)} \) is leveled homomorphic.

### 2.5.2 The Construction of FHE

The crucial point to construct a fully homomorphic encryption scheme being he scheme is able to evaluate polynomials of higher degree, in other words, the decryption procedure can be expressed as a polynomial of lower degree. Once the degree of polynomials that can be evaluated by the scheme exceeds the degree of the decryption polynomial (times two), the scheme is a fully homomorphic scheme \([26]\).

There is no natural fully homomorphic encryption scheme so far, majority schemes are constructed by Gentry’s idea. Firstly, construct a somewhat homomorphic encryption scheme which is a linear error correcting code \( C \) on the ring \( \mathcal{R} \). Linear code satisfies the additive homomorphism and error correcting code means the code with an error. Since \( C \) is an ideal of the ring, it satisfies the multiplicative homomorphism.
2.5. Fully Homomorphic Encryption

The code $C$ has two kinds of basis, one is 'good' basis which can be used as secret key, another basis is 'bad' for public key [26].

Since the error of the ciphertext will be expanded over the bound especially in multiplication, it occurs the failure in the decryption. Gentry used 'homomorphic decryption' to control the noise increasing. Encrypt the ciphertext and the corresponding public key by evaluate key, and input the result into the decryption circuit, output a new ciphertext. If the error of ciphertext is able to evaluate one more time especially in multiplication after each operation, then the ciphertext can perform unlimited times operation [28], [30]. Since the somewhat homomorphic encryption scheme can only perform limited operations with low-degree polynomials, the second step is to squash the decryption procedure so that it can be expressed as a low-degree polynomial which is supported by the scheme. Finally the application of a bootstrapping can transform the somewhat homomorphic encryption scheme to a fully homomorphic scheme [26].

2.5.3 The Security of FHE

The SHE and FHE is secure against chosen plaintext attacks. But no SHE and FHE scheme can be IND-CCA2 secure, based on the fact that the adversary is allowed to manipulate the challenged ciphertext and submit it to the decryption oracle in an IND-CCA2 attack. The IND-CCA1 has been proved to be not secure for FHE and SHE scheme [39]. Zhang provided a way to recovery the secret key by using the decryption oracle over the DGHV scheme [65] [66]. Chenal give more algorithm to allow an adversary to recover the pubic keys through decryption oracle queries [11].

2.5.4 Technique of Fully Homomorphic Encryption

The key point to construct fully homomorphic encryption is how to control the increase of the noise, there are some techniques like key switching and modulus switching.
2.5.1 Boostrapple

The fully homomorphic decryption requires the depth of decryption circuit less than the depth of the decryption circuit of evaluate algorithm. In fact, the depth of decryption of circuit is greater than the depth of the decryption circuit of evaluate algorithm. Using 'sub set sum phrase' is the way to squash the depth of decryption circuit [26].

2.5.2 Homomorphic Decryption

Homomorphic decryption can generate new ciphertext and reduce the error of ciphertext with conditions. Let $\text{Encrypt}(pk_1, m) \rightarrow c_1$ and $\text{Encrypt}(pk_2, sk_1) \rightarrow \bar{s}k_1$, the algorithm of homomorphic decryption is:

\[
\text{Recrypt}(pk_2, D, \bar{s}k_1, c_1) \\
\text{Encrypt}(pk_2, c_1) \rightarrow \bar{c}_1 \\
\text{Evaluate}(pk_2, D, \bar{s}k_1, \bar{c}_1) \rightarrow c_2
\]

$D$ is the circuit of decryption algorithm [26]. It decrypts the ciphertext after first encryption witch eliminates the error, then encrypted by new public key to get new ciphertext with new error. If the new error allows one more multiplication, in other words, the error is still within the bound after multiplication, then the goal of homomorphic decryption is achieved [26].

2.5.3 Key Switching

Key switching technique is based on the LWE of R-LWE, it can generate a new ciphertext corresponding to the different secret keys and reduce the dimension of ciphertext [5]. The new ciphertext $c_2$ is formed by a matrix $M$ multiplying the fresh ciphertext $c_1$. The row of matrix $M$ is the dimension of $c_1$ and the column of $M$ is the dimension
of $c_2$. The technique transforms $c_1$ with dimension $n_1$ to $c_2$ with dimension $n_2$ with the same modulus, the error of $c_2$ increases $<\text{BitDecomp}(c_1), c_2>$ than the error of $c_1$. The algorithm is:

$$\text{SwitchKeyGen}(s_1 \in \mathbb{R}^{n_1}, s_2 \in \mathbb{R}^{n_2}) : A \leftarrow \text{E.PublicKeyGen}(s_2, N)$$

$$B \leftarrow A + \text{Powersof2}(s_1),$$

( where $N = n_1 \cdot \lceil \log q \rceil$)

$$\text{output } \tau_{s_1 \rightarrow s_2} = B$$

$$\text{SwitchKey}(\tau_{s_1 \rightarrow s_2}, c_1) :$$

$$\text{output } c_2 = \text{BitDecomp}(c_1)^T \cdot B \in \mathbb{R}^{n_2}$$

2.5.4.4 Modulus Switching

Let the modulus be $q = x^k$, and each ciphertext with error $x$, the new error is approximately $x^2$ after multiplication. The error will reach the bound of decryption circuit after $\log k$ levels multiplicative. If the error times $1/x$ after each operation, the error will be reduced to the original value, meanwhile, the modulus decrease to $q/x$. The iteration can perform $k$ levels without bootstrapping before reaching the bound of error. The algorithm is:

$$\text{Scale}(c, q, p, r) : \text{input } s, q \text{ and } p \text{ with } (q > p > m)$$

$$\text{output } (p/q) \cdot c \text{ and } c' = c \mod r$$

2.5.4.5 Chinese Reminder Theorem

$p_1, \ldots, p_2$ are pairwise co-prime integers and $\text{CRT}_{(p_1, \ldots, p_2)}(m_1, \ldots, m_k)$ is a number in $\mathbb{Z} \cap (-\frac{\pi}{2}, \frac{\pi}{2})$, where $\pi = \prod_{i=1}^{k} p_i$. $\text{CRT}_{(p_1, \ldots, p_2)}(m_1, \ldots, m_k)$ is equivalent to $m_i \mod p_i$.
for all $i \in \{1, \ldots, k\}$. So we have

$$\text{CRT}_{(p_1, \ldots, p_2)}(m_1, \ldots, m_k) = \sum_{i=1}^{k} m_i M_i (M_i^{-1} \mod p_i) \mod \pi$$

where $M_i = \frac{\pi}{p_i}$.

The distributions of single bit with single private key is

$$D_{\gamma,\rho}(p) := \{ \text{choose } q \leftarrow \mathbb{Z} \cap [0, \frac{2^\gamma}{p}), e \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho) : \text{output } x = pq + e \}$$

The distributions of $\ell_Q$-bits with multi-private keys is

$$D_p(p_1, \ldots, p_k; Q_1, \ldots, Q_k; q_0) := \{ \text{choose } e_0 \leftarrow \mathbb{Z} \cap [0, q_0), e_i \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho) \text{ for } \forall i \in \{1, \ldots, k\} : \text{output } x = \text{CRT}_{(q_0, p_1, \ldots, p_k)}(e_0, e_1 Q_1, \ldots, e_k Q_k) \}$$

Consider the value of $x$ when $k = 1$, since $D := \{ \text{choose } q \leftarrow \mathbb{Z} \cap [0, q_0), e \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho) : \text{output } x = p_1 q + e \mod p_1 q_0 \}$, there is $x \leftarrow D_p(p_1; q_0)$.

$$x = \text{CRT}_{(q_0, p_1)}(e_0, e_1) = e_0 p_1 (p_1^{-1} \mod q_0) + e_1 q_0 (q_0^{-1} \mod p_1) \mod q_0 p_1 = e_0 p_1 \alpha + e_1 (p_1 \beta + 1) \mod q_0 p_1 = (e_0 \alpha + e_1 \beta) p_1 + e_1 \mod q_0 p_1$$

for some $\alpha$ and $\beta$. Since $e_0 \not\equiv \mod q_0$ and $\gcd(\alpha, q_0) = 1$, $(e_0 \alpha + e_1 \beta) \mod q_0$ is uniformly in $\mathbb{Z} \cap [0, q_0)$. 
2.5.4.6 Public Key Compression

The implementation of the DGHV fully homomorphic encryption scheme has a large size of public key in $\tilde{\Theta}(\lambda^{10})$. To resist lattice attack, each public key needs at least $2^{23}$ bits, the size of public key will be $2^{46}$ bits, it is too large for practical system. Coron [16] presented an efficient way to compress public key of the DGHV scheme, by using quadratic form instead of linear form when computing a ciphertext:

- **KeyGen($\lambda$):**
  
  Pick a random prime $p \in [2^{\eta - 1}, 2^{\eta})$. Let $x_0 = q_0 \cdot p$ where $q_0$ is a random square free $2^{\lambda}$-rough integer in $[0, 2^7/p)$. Generate integers $x_{i,b} \leftarrow p \cdot q_{i,b} + r_{i,b}$, where $1 \leq i \leq \beta$, $b \in \{0, 1\}$, $q_{i,b}$ is random integer in $[0, q_0)$ and $r_{i,b}$ is random integer in $(-2^\rho, 2^\rho)$. Output $sk = p$ and $pk = \langle x_0, x_{1,0}, x_{1,1}, \ldots, x_{\beta,0}, x_{\beta,1} \rangle$.

- **Encrypt(pk,m):**
  
  Input a random vector $b = (b_{i,j}) \in [0, 2^\alpha)$, of size $\tau = \beta^2$. Generate a random integer $r \in (-2^\rho, 2^\rho)$. Output a ciphertext $c \leftarrow m + 2r + 2 \sum_{1 \leq i,j \leq \beta} b_{i,j} \cdot x_{i,0} \cdot x_{j,1}$

- **Decrypt and Evaluate:**
  
  It is the same as the original scheme but with modulus $x_0$ after addition and multiplication.

The scheme can extend into higher degree [17]. Use the same way to generate elements and encrypt the plaintext as follow:

$$c^* = m + 2r + 2 \sum_{1 \leq i,j \leq \beta} b_{i,j} \ldots x_{i,0} \ldots x_{j,1} \mod x_0$$

The authors proved the scheme is semantically secure under the error-free approximate GCD assumption. They applied the leftover hash lemma on hash function family $h' : [0, 2^\alpha)^{\beta^2} \rightarrow \mathbb{Z}_{q_0}$, where $h'(b) = \sum_{1 \leq i,j \leq \beta} b_{i,j} \cdot q_{i,0} \cdot q_{j,1} \mod q_0$. 
Definition 2.5.7. (Hash Function) A family $\mathcal{H}$ of hash function $h : X \rightarrow Y$ is $\varepsilon-$pairwise independent if

$$\sum_{x \neq x'} (Pr_{h \leftarrow \mathcal{H}} [h(x)] = h'(x)) - \frac{1}{Y} \leq |X|^2 \cdot \frac{\varepsilon}{|Y|}$$

Lemma 2.5.1. (Leftover Hash Lemma) Let $\mathcal{H}$ be a family of $\varepsilon-$pairwise independent hash functions. Suppose that $h \leftarrow \mathcal{H}$ and $x \leftarrow X$ are chosen uniformly and independently. Then $(h, h(X))$ is $(\frac{1}{2} \sqrt{|Y|/|X| + \varepsilon})-$uniformly over $\mathcal{H} \times Y$

The key element $x_{i,b}$ have been proved that a certain family of quadratic hash function is close enough to be pairwise independent, so this can apply the leftover hash lemma. The significance of this method is reducing the size of public key from $	au = \tilde{O}(\lambda^3)$ down to $2\beta = \tilde{O}(\lambda^{1.5})$.

The semantic security of the scheme based on approximate-GCD assumption with error-free $x_0$. The adversary can find the exact multiple $p$ by solving the AGCD problem. The known attack had been presented on van Dijk’ paper [20].

Definition 2.5.8. (AGCD Problem) Let $c_i \in \mathbb{Z}$, $\tau$ integers such that there exist some unique integers $r_i \in \mathbb{Z}$ and a unique integer $p \in \mathbb{N}$ such that $\forall i$, $(c_i - r_i)|p$ and $\forall i$, $|r_i| \leq \gamma \leq p/2$. Then, the Approximate Greatest Common Divisor problem, denoted by $\gamma - AGCD_\tau$, is to find $p$, given $c_i$.

2.5.5 Fully Homomorphic Encryption Schemes

Many SHE and FHE schemes have been proposed after Gentry’s work. These schemes can be classified based on different hardness assumptions as in figure [11].

- The first category is based on hard problems on lattices that starts with Gentry’s work[Gen09a,Gen09b] [25], [26], [62], [27], [63], [27].
2.5. Fully Homomorphic Encryption

- The second category relies on the approximate greatest common divisor (AGCD) problem and some variants. The typical scheme is [vDGHV10] [15], [20].

- The third category is based on the learning with errors (LWE) and on the ring-learning with errors (RLWE) problems like schemes [NLV11, BGV12, GHS12b, Bra12, GSW13] [4], [5], [7], [6], [31].

![Figure 2.1: Hardness assumptions and schemes](image)

### 2.5.5.1 Gentry’s First Fully Homomorphic Encryption Scheme [25]

Gentry used the ideal lattices to construct fully homomorphic encryption scheme of PKE [26]. A ciphertext $\psi$ is in form of $v + x$ when $v$ is in the ideal lattices and $x$ is in the error distribution. The coefficient of ciphertext vectors is the elements in a polynomial ring $\mathbb{Z}[x]/f(x)$. The addition and multiplication of ciphertext satisfy the ring operation. The security of the scheme is based on a natural decisional version of the closest vector problem for ideal lattices for ideals in a ring.
Gentry’s initial somewhat homomorphic encryption scheme is based on lattice.

- **KeyGen**
  
  Input a ring \( R \) and a basis \( B_1 \) of lattice \( I \). It sets \((B^j_k, B^j_{sk}) \leftarrow \text{IdealGen}(R, B_1)\). The public key has \( R, B_1, B^j_{pk} \) and \text{Samp}, where \text{Samp} is an algorithm with sampling basis from the coset of lattice.
  
  The secret key is \( B^j_{sk} \).

- **Encrypt**
  
  Input a public key \( pk \) and a plaintext \( m \in \mathcal{P} \). It sets \( \psi' \leftarrow \text{Samp}(m, B_I, R, B^j_{pk}) \)
  
  Output a ciphertext \( \psi' \leftarrow \psi \mod B^j_{pk} \).

- **Decrypt**
  
  Input the secret key and a ciphertext \( \psi \).
  
  Output \( m \leftarrow (\psi \mod B^j_{sk}) \mod B_I \)

- **Evaluate**
  
  Input the public key \( pk \) and a circuit \( C \) composed of \text{Add}_{B_I} \) and \text{Mult}_{B_I}, \) and a set of ciphertext \( \psi \).
  
  Output new ciphertext \( \psi \).
  
  \text{Add}(pk, \psi_1, \psi_2). \) Output \( \psi_1 + \psi_2 \mod B^j_{pk} \).
  
  \text{Mult}(pk, \psi_1, \psi_2). \) Output \( \psi_1 \times \psi_2 \mod B^j_{pk} \).

This somewhat homomorphic scheme is not yet bootstrappable. The way to squash the decryption procedure is reducing the degree of the decryption polynomial. Gentry added the public key in form of sparse subset-sum problem (SSSP). The public key is augmented with a big set of vectors, so there is a very sparse subset of public key that adds up to the secret key [27]. A ciphertext of the underlying scheme can be decrypted with a low-degree polynomial. The scheme with bootstrapping is the fully
2.5. Fully Homomorphic Encryption

homomorphic scheme $E^{(d)}$ with security parameter $\lambda$ which can handle all circuit of depth $d$ is given:

- **KeyGen$_{E^{(d)}}(\lambda, d)$:**
  \[
  (sk_i, pk_i) \leftarrow \text{KeyGen}_E(\lambda) \quad \text{for} \quad i \in [0, d]
  \]
  \[
  sk_{ij} \leftarrow \text{Encrypt}_E(pk_{i-1}, sk_{ij}) \quad \text{for} \quad i \in [1, d], j \in [1, \ell]
  \]
  \[
  sk^{(d)} \leftarrow sk_0, \quad pk^{(d)} \leftarrow (< pk_i >, < sk_{ij} >)
  \]

- **Encrypt$_{E^{(d)}}(pk^{d}, m)$:**
  Input a public key $pk^{(d)}$ and a plaintext $m \in \mathcal{P}$,
  Output a ciphertext $\psi \leftarrow \text{Encrypt}_E(pk^{d}, m)$.

- **Decrypt$_{E^{(d)}}(sk^{d}, \psi)$:**
  Input a secret key $sk^{(d)}$ and a ciphertext $\psi$.
  Output $\text{Decrypt}_E(sk_0, \psi)$.

- **Evaluate$_{E^{(d)}}(pk^{(d)}, C^{(d)}, \Psi^{(d)})$**:
  Input the public key $pk^{(d)}$, a circuit $C^{(d)}$ of depth at most $d$, and a tuple of ciphertext $\Psi^{(d)}$.
  Output new tuple of ciphertext $\Psi^{(d-1)}$ until $d = 0$ and terminates.
  Set $(C^{(d-1)}_\delta, \Psi^{(d-1)}_\delta) \leftarrow \text{Augment}_{E^{(d)}}(pk^{(d)}, C^{(d)}, \Psi^{(d)})$.
  Set $(C^{(d-1)}_\delta, \Psi^{(d-1)}_\delta) \leftarrow \text{Reduce}_{E^{(d-1)}}(pk^{(d-1)}, C^{(d-1)}_\delta, \Psi^{(d-1)}_\delta)$.
  Runs $\text{Evaluate}_{E^{(d-1)}}(pk^{(d-1)}, C^{(d-1)}_\delta, \Psi^{(d-1)}_\delta)$.

Unfortunately, Gentry’s scheme has the inherent efficiency limitations. In the decryption circuit, each secret-key bit is replaced by a large ciphertext that encrypts that bit. The complexity of the scheme is extremely large, which will be defined as the bit-length of the individual ciphertexts that are used to encrypt the bits of the secret key times the complexity of the decryption\textsuperscript{28}. The bottleneck in practical is the time of per-gate evaluation.
2.5.5.2 Dijk, Gentry, Halevi and Vaikuntanathan’s Scheme Over The Integers (DGHV) \([20]\)

Dijk, Gentry, Halevi and Vaikuntanathan published a fully homomorphic encryption scheme over the integer rather than on ideal lattice \([20]\). The construction of the somewhat fully homomorphic scheme:

- **KeyGen(\(\lambda\))**: Secret Key: choose random odd \(\eta\) bits integer \(p \leftarrow (2\mathbb{Z} + 1) \cap [2\eta \cap, 2\eta)\).
  Public Key: sample uniformly \(x_i \leftarrow D_{\gamma}, \rho(p)\), for \(i = 1, 2, \ldots, \tau\). The odd integer \(x_0\) has to be the largest number and the remainder of \(x_0 \mod p\) is even.
  Output \(sk = p\) and \(pk = <x_1, x_2, \ldots, x_\tau>\).

- **Encrypt(pk,m)**: Input a random subset \(S \subset 1, \ldots, \tau\), a random integer \(r\) in \((-2^\rho', 2^\rho)\) and a plaintext \(m \in 0, 1\)
  Output a ciphertext \(c \leftarrow [m + 2r + 2 \sum_{i \in S} x_i] x_0\).

- **Decrypt(sk,c)**: Input a secret key \(sk\) and a ciphertext \(c\).
  Output \(m \leftarrow (c \mod p) \mod 2\).

- **Evaluate(pk,C,c_1,\ldots,c_\tau)**: Input \(t\) ciphertext \(c_i\) as \(t\) inputs to the binary circuit \(C_\varepsilon\), apply addition and multiplication gates of \(C_\varepsilon\) on ciphertext.
  Output the integer of operation result.

The noise expands quickly especially under multiplication. Assuming the bound of noise in the fresh ciphertext \(x_0\) is \(B\), let the degree of decryption polynomial \(f\) is \(d\). The scheme can decrypt the ciphertext correctly when \(|f| < p/2\). Due to the condition, the bound of noise has \(t^d \cdot B^d < p/2\) after \(d\) times multiplication, in other words \(d < (\log p)/(\log tB)\). The depth of decryption circuit depends on the operation levels on \(c \cdot p^{-1}\) which is at least \(2(\log p)^{2.71}\) levels. It is obviously bigger than the
2.5. Fully Homomorphic Encryption

polynomial degree d. To get fully homomorphic encryption, the bootstrapping with squashing the decryption circuit is still essential [30].

The security of the DGHV scheme is based on two problems, one is the hardness of approximate-gcd problem in somewhat homomorphic encryption and another is SSSP in bootstrapping. To against the known attack on the approximate-gcd problem like brute-forcing the remainders, continued fraction and Howgrave-Graham’s approximate gcd algorithm, the security parameter of the scheme needs at least $2\lambda$ [10], [35].

2.5.5.3 Brakerski and Vaikuntanathan’s Scheme Based on RLWE (BV11b) [6]

Brakerski and Vaikuntanathan presented a scheme based on RLWE [6]. They used two technique: re-linearization and dimension-modulus reduction to construct the scheme. Re-linearization can reduct the size of the ciphertext back down to $n + 1$. Let $s$ be the original secret key and $t$ is the new secret key. Each ciphertext is $(\alpha_{i,j}, b_{i,j})$ where

$$b_{i,j} = \langle \alpha_{i,j}, t \rangle + 2e_{i,j} + s[i] \cdots [j] \approx \langle \alpha_{i,j}, t \rangle + s[i] \cdot [j].$$

Consider the multiplication of two polynomials,

$$f_{\alpha,b}(x) \cdot f_{\alpha',b'}(x) = (b - \sum \alpha[i]x[i]) \cdot (b' - \sum \alpha'[i]x[i])$$

$$= h_0 + \sum h_i \cdot x[i] + \sum h_{i,j} \cdot x[i]x[j]$$

$$= h_0 + \sum h_i(b_i - \langle \alpha_i, t \rangle) + \sum_{i,j} h_{i,j} \cdot (b_{i,j} - \langle \alpha_{i,j}, t \rangle)$$

the result is the linear polynomial with $n + 1$ coefficients and can be decrypt by the new secret key $t$. It is a good way to multiplicative two ciphertext without expanding the size and can be decrypted under the new secret key. The somewhat fully homomorphic encryption scheme is given as:
2.5. Fully Homomorphic Encryption

- **KeyGen(λ):**
  
  Choose random $L + 1$ vectors $s_0, s_1, \ldots, s_L \leftarrow \mathbb{Z}_q^n$

  Choose random matrix $A \leftarrow \mathbb{Z}_{m \times n}$

  Choose a random vector $e \leftarrow \chi^m$

  Compute $b = As_0 + 2e$

  Output $sk = s_L$ and $pk = (A, b)$.

- **Encrypt(pk, m):**

  Choose a random vector $r \leftarrow \{0, 1\}^m$

  Set $v = A^T r \in \mathbb{Z}_q^n$

  Set $w = b^T r + m \in \mathbb{Z}_q$

  Output a ciphertext $c \leftarrow ((v, w), \ell)$.

- **Decrypt(sk, c):**

  Input a secret key $sk$ and a ciphertext $c$.

  Output $m \leftarrow (w - \langle v, s_L \rangle \mod p) \mod 2$.

- **Evaluate(pk, f, c_1, \ldots, c_\tau):**

  Addition gates: $c_{Add} = ((v_{Add}, w_{Add}, \ell) := ((\sum v_i, \sum w_i), \ell)$

  Multiplication gates: $c_{Mult} = ((v_{Mult}, w_{Mult}), \ell)$

The security of the scheme relies on the worst-case hardness of standard problem on lattices [44].

2.5.5.4 Brakerski, Gentry and Vaikuntanathan’ Scheme Based on LWE (BGV12) [5]

BGV is the most efficiency scheme so far. The scheme applies key switching and modulus switching, it is a leveled fully homomorphic encryption scheme without bootstrapping. It reduces the production of two ciphertext down to the original dimension
by key switching and reduces the noise by modulus switching on each level\cite{5}. The scheme can be based on the LWE and also the RLWE. The scheme on the RLWE has a better efficiency than on the LWE. Let ring $R = \mathbb{Z}[x]/(x^d + 1)$, where $d$ is the power of 2 and $N = \lceil (2n + 1) \log q \rceil$. The schemes is:

- **KeyGen($\lambda$):**
  
  Choose random $s' \leftarrow \chi^n$

  Let $s = (1, s)$ with $s[0] = 1$ and $s' \in \mathbb{Z}_{q}^n$ Choose random matrix $A \leftarrow \mathbb{Z}_{q}^{N \times n}$

  Choose a random vector $e \leftarrow \chi^N$

  Compute $b = A's' + 2e$

  Set $A$ is the $(n+1)$ column matrix $(b | -A')$

  Output $sk = (1, s'[1], \ldots, s'[n]) \in \mathbb{Z}_{q}^{n+1}$ and $pk = A$.

- **Encrypt($pk, m$):**

  Let $m \leftarrow (m, 0, \ldots, 0) \in \mathbb{Z}_{q}^{n+1}$

  Choose a random vector $r \leftarrow \mathbb{Z}_{2}^{N}$

  Output a ciphertext $c \leftarrow m + A^T r \in \mathbb{Z}_{q}^{n+1}$.

- **Decrypt($sk, c$):**

  Input a secret key $sk$ and a ciphertext $c$.

  Output $m \leftarrow (<v, s_L> \mod p) \mod 2$.

- **Evaluate($pk, f, c_1, \ldots, c_r$):**

  Addition gates: $c_4 = \text{Refresh}(c_3, \tau_{s_j}'' \rightarrow s_{j-1}, q_j, q_{j-1})$, where $c_3 \leftarrow c_1 + c_2 \mod q_j$

  Multiplication gates: $c_4 = \text{Refresh}(c_3, \tau_{s_j}'' \rightarrow s_{j-1}, q_j, q_{j-1})$, where $c_3$ is the linear equation of $L_{c_1,c_2}^{long}(x \otimes x)$

Assuming the error with bound $B$ and the corresponding modulus is $q_j$. The noise will increase to $2B$ by addition and approximated to be $B^2$ by multiplication. After the key
switching, the error becomes to be \( E^2 + e_{\text{switch}} \). Processing the modulus switching, the error decrease to \((q_j^{-1}/q_j) \cdot (E^2 + e_{\text{switch}}) + e_{\text{scale}}\). To decrypt the ciphertext correctly, the error should be smaller than \( B \) on each level \[3\]. The scheme can operate on the circuit of the depth \( L \). It can transfer to the fully homomorphic encryption by bootstrapping. The security of the scheme relies on the SVP problem on lattices \[41\].

### 2.5.5.5 Brakerski’s Scheme Based on LWE (Bra12) \[4\]

The scheme is also based on the LWE problem which can be extend on the RLWE. The advantages of Bra scheme are: using the same modulus which means the scheme does not need to do the modulus switching. The security can be classical reducing from the worst-case hardness of the GapSVP problem \[4\].

The technique they used to construct the scheme is vector decomposition and key switching. Vector decomposition is a way to operate the inner product.

- **BitDecomp\(_q\)(x):**
  
  For \( x \in \mathbb{Z}^n \), let \( w_i \in \{0, 1\}^n \), \( x \) can represent as \( x = \sum_{i=0}^{\lceil \log q \rceil - 1} 2^i \cdot w_i \bmod q \).

- **PowerOfTwo\(_q\)(y):**
  
  For \( y \in \mathbb{Z}^n \), output \([y, 2 \cdot y, \ldots, 2^\lceil \log q \rceil - 1 \cdot y]_q \in \mathbb{Z}_q^{n \cdot \lceil \log q \rceil - 1}\)

The somewhat homomorphic encryption scheme is:

- **KeyGen(\(\lambda\)):**
  
  Choose random \( L + 1 \) vectors \( s_0, s_1, \ldots, s_L \leftarrow \mathbb{Z}_q^n \)

  Choose random matrix \( A \leftarrow \mathbb{Z}_q^{N \times n} \)

  Choose a random vector \( \epsilon \leftarrow \chi^m \)

  Compute \( b_0 = A s_0 + 2 \epsilon \)

  Set \( P_0 \) is the \((n + 1)\) column matrix \((b_0, A')\)

  \( \tilde{s}_{i-1} = \text{BitDecomp}(1, s_{i-1}) \bigotimes \text{PowerOfTwo}(1, \tilde{s}_{i-1}) \in \{0, 1\}^{(n+1)(\lceil \log q \rceil)^2} \)
2.5. Fully Homomorphic Encryption

Compute $P_{i-1} : i \leftarrow \text{SwitchKeyGen}(\tilde{s}_{i-1}, s_i)$

Output $sk = s_L$ and $pk = P_0$, and $evk = \{P_{(i-1):i}\}_{i \in [L]}$

- Encrypt($pk, m$):
  Let $m \leftarrow (m, 0, \ldots, 0) \in R_{q}^{n+1}$
  Choose a random vector $r \leftarrow R_2^{N}$
  Output a ciphertext $c \leftarrow m + A^T r \in R_{q}^{n+1}$

- Decrypt($sk, c$):
  Input a secret key $sk$ and a ciphertext $c$.
  Output $m \leftarrow <v, s> \mod p \mod 2$.

- Evaluate($pk, f, c_1, \ldots, c_r$):
  Addition gates: $c_{\text{Add}} = \text{SwitchKey}(P_{(i-1):i}, \tilde{c}_{\text{Add}}) \in \mathbb{Z} : q^{n+1}$, where $\tilde{c}_{\text{Add}} = \text{PowerOfTwo}(c_1 + c_2) \otimes \text{PowerOfTwo}(1, 0, \ldots, 0)$
  Multiplication gates: $c_{\text{Mult}} = \text{SwitchKey}(P_{(i-1):i}, \tilde{c}_{\text{Mult}}) \in \mathbb{Z} : q^{n+1}$, where $\tilde{c}_{\text{Mult}} = \lceil \frac{2}{q} \cdot \text{PowerOfTwo}(c_1) \otimes \text{PowerOfTwo}(c_2) \rceil$

The noise increasing in this scheme is different as previously schemes. Assuming the noise bound of the ciphertext is $E$ and the fresh ciphertext has noise bound of $N \cdot B$. The noise increases to $2E + n^2 \log q^3$ after addition and $(n \cdot \log q) \cdot E + (n^2 \log q^3) \cdot B$ after multiplication. Since multiplication is defined as $(2/q) \cdot (c_1 \otimes c_2)$, each item divided by $q$, $E2/q$ can be ignored when $q$ is large enough for classical reduction from GapSVP \[4\]. To get the decryption correctly, the error needs to be less than $\lceil q/2 \rceil / 2$, therefore $q/B > (n \cdot \log q)^L$. The depth $L$ depends on the ratio of $q/B$.

2.5.5.6 Plantard, Susilo and Zhang’ s Hidden Ideal Lattice \[50\]

Plantard, Susilo and Zhang constructed a fully homomorphic encryption scheme by using hidden ideal lattice \[50\]. They used hidden ideal lattice to unify two schemes
which are ideal lattice based schemes and integer based schemes. The security of the
scheme does not relies on the sparse sub set sum problem (SSSP), but relies on the
bounded distance decoding problem (BDDH) of ideal lattice and approximate greatest
common divisor problem (AGCD) of integer.

Definition 2.5.9. (BDDP over Ideal Lattice) [50] Let $\gamma \in \mathbb{R}^+$ be a positive real. Let $L$ be an $n$ dimensional ideal lattice, and $v \in \mathbb{Z}$ such that there exists a unique vector $u \in L$ satisfying $\text{dist}(v, u) \leq \gamma$. The $\gamma$-Bounded Distance Decoding problem over ideal lattice, denoted by $\gamma$-BDD$n$, is to find $u$, given a basis of $L$ and $v$.

Definition 2.5.10. (Dec BDDP over Ideal Lattice) [50] Let $\gamma \in \mathbb{R}^+$ be a positive real. Let $L$ be an $n$ dimensional ideal lattice, and $v \in \mathbb{Z}$. The decisional $\gamma$-Bounded Distance Decoding problem over ideal lattice, denoted by Dec $\gamma$-BDDin, is to decide if there exists a unique vector $u \in L$ satisfying $\text{dist}(v, u) \leq \gamma$, or not, given a basis of $L$ and $v$.

Definition 2.5.11. (AGCD Problem) [50] Let $c_i \in \mathbb{Z}$, $\tau$ integers such that there exist some unique integers $r_i \in \mathbb{Z}$ and a unique integer $p \in \mathbb{N}$ such that $\forall i$, $(c_i - r_i)p$ and $\forall i$, $|r_i| \leq \gamma \leq p/2$. Then, the Approximate Greatest Common Divisor problem, denoted by $\gamma - \text{AGCD}_\tau$, is to find $p$, given $c_i$.

Definition 2.5.12. (Subset Sum Problem) [50] Let $\{c_1, c_2, \ldots, c_n\}$ be a set of positive integers. Let $c = \sum_{i=1}^{n} s_i c_i$, where $s_i \in \{0, 1\}$. Let $d \leftarrow \sum_{i=1}^{n} s_i$. The subset sum problem, denoted by $d, n$-SSP, is to find $\{s_i\}$, given $\{c_i\}$ and $c$.

The hidden ideal lattice homomorphic encryption scheme gives an idea, that instead of giving the lattice as public key, given some vectors close to the lattice. The lattice is only known by secret key holder. Since the ciphertext are also vectors close to the lattice with a bounded distance, the property of the homomorphism of the ciphertext still holds [38]. The somewhat scheme is:
• KeyGen(\(\lambda\)):

Choose a random irreducible polynomial of degree \(n\), \(f(x) = x^n + 1\).

Choose a random vector \(v\) in \(\{u \in \mathbb{Z}^n, 2^{n-1} < \|u\| < 2^n, \sum_{i=0}^{n-1} u_i \mod 2 = 1\}\).

Generate the random matrix \(V \leftarrow \text{Rot}(v,f)\):

\[
\begin{vmatrix}
  v_0 & v_1 & v_2 & \cdots & v_{n-1} \\
-\v_{n-1} & v_0 & v_1 & \cdots & v_{n-2} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
-v_1 & -v_2 & -v_3 & \cdots & v_0 \\
\end{vmatrix}
\]

Let \(d \leftarrow |\text{det}(V)|\) is the determinant of \(V\).

Choose random \(\tau - 1\) vectors \(g_i\) in \(\{u \in \mathbb{Z}^n, 2^{\gamma-1} < \|u\| < 2^{\gamma}\}\), and another vector \(g_\tau\) in \(\{u \in \mathbb{Z}^n, \|u\| < 2^{\gamma}, \sum_{i=0}^{n-1} u_i \mod 2 = 1\}\).

Choose random \(\tau - 1\) vectors \(r_i\) in \(\{-1, 0, 1\}^n, \|u\| \leq \rho\}\), and another vector \(r_\tau\) in \(\{-1, 0, 1\}^n, \|u\| \leq \rho, \sum_{i=0}^{n-1} u_i \mod 2 = 1\}\).

Compute \(\tau\) vectors \(\pi_i \leftarrow g_i \times v + r_i\), for \(1 \leq i \leq \tau\).

Find the integer polynomial \(w(x)\), which satisfies \(w(x) \times v(x) = d \mod f(x)\), denote \(W \leftarrow \text{Rot}(w,f)\). Output \(sk = s\{d, w\}\) and \(pk = \{\pi_i\}\).

• Encrypt(pk,m):

Choose random \(\tau - 1\) vector \(s_i\) in \(\{\sum_{j=1}^{n} s_{i,j} \mod 2 = 0, 1 \leq i \leq \tau - 1\}\), a vector \(s_\tau\) in \(\{\sum_{j=1}^{n} s_{\tau,i} \mod 2 = m\}\), and a vector \(s_{\tau+1}\) in \(\{\sum_{j=1}^{n} s_{\tau+1,j} \mod 2 = 0\}\).

Output a ciphertext \(c \leftarrow \sum_{i=1}^{n} s_i \times \pi_i + s_{\tau+1}\).

• Decrypt(sk,c):

\(c' \leftarrow \lfloor c \times w/d \rfloor\). Output \(c \leftarrow c'(1) \mod 2\).

• Evaluate(pk, \(C, \ldots, c_\tau\)):

Addition gates \((c_1, c_2)\): Output \(c \leftarrow c_1 + c_2\).

Multiplication gates \((c_1, c_2)\): Output \(c \leftarrow c_1 \times c_2\).

The semantic security of the scheme has been proved:
2.5. Fully Homomorphic Encryption

Theorem 2.5.2. [33] If an algorithm \( A \) breaks the semantic security with advantage \( \varepsilon \), then there exist an algorithm \( B \) that solves the Dec \( \alpha, \beta - BDDHi_{u, \tau} \) with advantage of \( \varepsilon/8 \). The running time of \( B \) is polynomial in the running time of \( A \).

2.5.5.7 Nuida and Kurosawa’s Batching Scheme [12]

The majority Fully homomorphic encryption schemes and somewhat homomorphic schemes can only encrypt single bit each time. The efficiency can be improved by using batch plaintext into a single bit [12]. The scheme can encrypt multiple bits into a single ciphertext by using Chinese Remainder Theorem. But it is only applied in binary space. Kofi et. modified the scheme into the non-binary space.

In DGHV scheme, the public key size in *somewhat homomorphic encryption* is \( \widetilde{O}(\lambda^{10}) \) and in *fully homomorphic encryption* is \( \widetilde{O}(\lambda^{13}) \). Coron describes public key compression for fully homomorphic encryption over integers. It reduces the public key size to \( \widetilde{O}(\lambda^{5}) \) of *somewhat homomorphic encryption* and \( \widetilde{O}(\lambda^{8}) \) of *fully homomorphic encryption*. Consider the batch fully homomorphic encryption scheme in non-binary space, the public key size is \( \widetilde{O}(\lambda^{8}) \) in both *somewhat homomorphic encryption* and *fully homomorphic encryption*. To achieve this goal, CRT (Chinese Remainder Theory) is an important technique [36]. Let \( p_1, \ldots, p_2 \) are pairwise coprime integers and \( CRT(p_1, \ldots, p_2)(m_1, \ldots, m_k) \) is a number in \( \mathbb{Z} \cap (-\frac{\pi}{2}, \frac{\pi}{2}] \), where \( \pi = \prod_{i=1}^{k} p_i \). \( CRT(p_1, \ldots, p_k)(m_1, \ldots, m_k) \) is equivalent to \( m_i \mod p_i \) for all \( i \in \{1, \ldots, k\} \). So we have

\[
CRT(p_1, \ldots, p_2)(m_1, \ldots, m_k) = \sum_{i=1}^{k} m_i M_i (M_i^{-1} \mod p_i) \mod \pi
\]

where \( M_i = \pi/p_i \). The distributions of \( \ell_Q \)-bits with multi-private keys is

\[
\mathcal{D}_\rho(p_1, \ldots, p_k; Q_1, \ldots, Q_k; q_0) := \{\text{choose } e_0 \leftarrow \mathbb{Z} \cap [0, q_0), e_i \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}.
\]
Output

\[ x = \text{CRT}_{q_0, p_1, \ldots, p_k}(e_0, e_1 Q_1, \ldots, e_k Q_k), \quad \text{for } \forall i \in \{1, \ldots, k\}. \]

Consider the value of \( x \) when \( k = 1 \), since \( D := \{ \text{choose } q \leftarrow \mathbb{Z} \cap [0, q_0), e \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho) : \text{output } x = p_1 q + e \mod p_1 q_0 \} \), there is \( x \leftarrow D_\rho(p_1; q_0) \).

\[
\begin{align*}
x &= \text{CRT}_{q_0, p_1}(e_0, e_1) \\
    &= e_0 p_1 p_1^{-1} \mod q_0 + e_1 q_0 q_0^{-1} \mod p_1 p_1 \\
    &= (e_0 \alpha + e_1 \beta) p_1 + e_1 \mod q_0 p_1
\end{align*}
\]

for some \( \alpha \) and \( \beta \). Since \( e_0 \not\equiv \mod q_0 \) and \( \gcd(\alpha, q_0) = 1 \), \( (e_0 \alpha + e_1 \beta) \mod q_0 \) is uniformly in \( \mathbb{Z} \cap [0, q_0) \). Recall Nuida and Kurosawa’s batch somewhat homomorphic encryption scheme. The plaintext space is \( \mathcal{M} = (\mathbb{Z}_{Q_1})^{h_1} \times (\mathbb{Z}_{Q_2})^{h_2} \times \cdots \times (\mathbb{Z}_{Q_k})^{h_k} \), where \( k \geq 1 \), \( h_j \geq 1 \) and \( Q_1, \ldots, Q_k \) are distinct primes. The scheme to pack \( \ell \) plaintext bits \( m_0, \ldots, m_{\ell-1} \) into a single ciphertext is the extension of the DGHV scheme which listed as following [12]:

- KeyGen (1^\lambda): Pick random prime numbers \( p_{i,j} \) as secret key, \( (i, j) \in \mathcal{I} \) and \( \mathcal{I} := \{(i, j) | i, j \in \mathbb{Z}, 1 \leq i \leq k, 1 \leq j \leq h_i\} \) and \( Q_{i'} \) are different. Choose

\[
q_0 \leftarrow [1, 2^\gamma / \prod_{(i, j) \in \mathcal{I}} p_{i,j}] \cap \text{ROUGH}(2^\lambda),
\]

which is coprime to all \( p_{i,j} \) and all \( Q_{i'} \). Choose \( e_{\xi,0} \) and \( e_{\xi,i,j} \) for \( \xi \in \{1, 2, \ldots, \tau\} \) and \( (i, j) \in \mathcal{I} \):

\[
e_{\xi,0} \leftarrow [0, q_0] \cap \mathbb{Z}, e_{\xi,i,j} \leftarrow (-2^\rho, 2^\rho) \cap \mathbb{Z}.
\]
2.5. Fully Homomorphic Encryption

Let \( x_\xi \) be the unique integer in \((-N/2, N/2]\) satisfying

\[
x_\xi \equiv e_{\xi,0} \pmod{q_0}, \quad x_\xi \equiv e_{\xi,i,j}Q_i \pmod{p_{i,j}} \text{ for } (i, j) \in \mathcal{I}
\]

Similarly, for \((i, j), (i', j') \in \mathcal{I}\), choose \( e'_{i,j,0} \) and \( e'_{i,j',i',j'} \):

\[
e'_{i,j,0} \leftarrow [0, q_0) \cap \mathbb{Z}, \quad e'_{i,j,i',j'} \leftarrow (-2^\rho, 2^\rho) \cap \mathbb{Z},
\]

and let \( x'_{i,j} \) be the unique integer in \((-N/2, N/2]\) satisfying:

\[
x'_{i,j} \equiv e'_{i,j,0} \pmod{q_0}
\]

\[
x'_{i,j} \equiv e'_{i,j,i',j'}Q_i + \delta_{(i,j),(i',j')} \pmod{p_{i',j'}},
\]

where \( \delta \) is Kronecker delta. The public key \( pk \) consists of all \( N, x_\xi \) and \( x'_{i,j} \), and the secret key \( sk \) consists all \( p_{i,j} \).

- Encrypt \((pk, m \in M)\):
  Generate a random subset \( T \subseteq \{1, 2, \ldots, \tau\} \). Output the ciphertext as

\[
c := \sum_{(i,j) \in \mathcal{I}} m_{i,j}x'_{i,j} + \sum_{\xi \in T} \text{Mod } N, \quad c \in (-N/2, N/2] \cap \mathbb{Z}.
\]

- Evaluate \((pk, f, c_1, \ldots, c_n)\):
  Given a polynomial \( f \) with integer coefficients and ciphertext \( c_1, \ldots, c_n \), output \( c^* \) is

\[
c^* := f(c_1, \ldots, c_n) \pmod{N}
\]

- Decrypt \((sk, c)\):
  Output \( m := ((c \pmod{p_{i,j}}) \pmod{Q_i})_{(i,j)} \in \mathcal{I} \).
The scheme is secured under the Error-Free Approximate-GCD assumption.

**Definition 2.5.13.** (Error-Free Approximate GCD problem) [50] For a random $\eta$-bit prime $p$, $y_0 = q_0 \ldots p$ where $q_0$ is a random integer in $[0, 2^{\gamma}/\rho)$, and polynomially many samples from $D_\rho(p, q_0)$, then output $p$.

Similarly, for batching scheme, the specific integers are $q_0$ and $p_{i,j}(i, j) \in I$. We input a vector $m \in \mathbb{Z}^\ell$ into the oracle $O_{q_0, (p_{i,j})(i, j)}(i, j) \in I$ and the output will be $X$,

$$X = CRT_{q_0, (p_{i,j})}(q_0, m_{1,1} + Q_1 \cdot r_{1,1}, \ldots, m_{k,h_i} + Q_k \cdot r_{k,h_i})$$

where $q \leftarrow [0, q_0)$ and $r_{i,j} \leftarrow (-2^\rho, 2^\rho)$ Since it is hard to distinguish between an encryption of zero and an encryption of a random message by using the public-key encryption instead of the oracle. The scheme can be proved to be semantically secure [33].
Chapter 3

The construction of Fully Homomorphic Encryption Scheme

3.1 Our Scheme with Compression Public Key

Our technique consists in working on vectors with integer coefficient $Vec(\pi_{i,j})$ of the form $Vec(\pi_{i,j}) = Vec(\pi_i) \times Vec(\pi_j)$. The number of the public key stored is $2\beta$ not $\tau$ as initial. The $\tau$ public keys in the encryption can be generated by $\tau = \beta^2$ public keys. Then our encryption is no longer choosing a linear form as the public key. We will use a quadratic form.

3.1.1 SHE Scheme

The somewhat homomorphic scheme with public key compression on hidden ideal lattice is constructed as follows. Generate random polynomial vectors as the ring element, divide into two groups. Choose a vector from each group, then multiply the two vector modular the irreducible polynomial. Therefore, the original public key will be replaced by the new quadratic key.
• KeyGen($\lambda$):

Choose a random irreducible polynomial of degree $n$, $f(x) = x^n + 1$.

Choose a random vector $v$ in \( \{ u \in \mathbb{Z}^n, 2^{n-1} < \|u\| < 2^n, \sum_{i=0}^{n-1} u_i \mod 2 = 1 \} \).

Generate the random matrix $V \leftarrow \text{Rot}(v, f)$:

\[
\text{Rot}(v, f) = \begin{vmatrix}
  v_0 & v_1 & v_2 & \ldots & v_{n-1} \\
  -v_{n-1} & v_0 & v_1 & \ldots & v_{n-2} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  -v_1 & -v_2 & -v_3 & \ldots & v_0
\end{vmatrix}
\]

Let $d \leftarrow |\det(V)|$ is the determinant of $V$. It is almost the same as the initial scheme so far, the difference is starting from the public key vector generation.

Choose two groups of random vectors, each group has $\beta$ vectors $g_i$ or $g_j$ for $1 \leq i, j \leq \beta$, the Euclidean norm of each vector is in \( \{ u \in \mathbb{Z}^n, 2^{\gamma-1} < \|u\| < 2^\gamma \} \). There is at least one vector of each group with the Euclidean norm in \( \{ u \in \mathbb{Z}^n, \|u\| < 2^\gamma, \sum_{i=0}^{n-1} u_i \mod 2 = 1 \} \). The total number of vectors is $2\beta$ which equals $2\sqrt{\tau}$.

Choose two groups of random vectors, each group has $\beta$ vectors $r_i$ or $r_j$ for $1 \leq i, j \leq \beta$, the Euclidean norm of each vector is in \( \{ u \in \mathbb{R}^n, \|u\| \leq \rho \} \). There is at least one vector of each group with the Euclidean norm in \( \{ u \in \mathbb{R}^n, \|u\| \leq \rho, \sum_{i=0}^{n-1} u_i \mod 2 = 1 \} \). The total number of vectors is $2\beta$ which equals $2\sqrt{\tau}$.

Compute $\beta$ vectors $\pi_i \leftarrow g_i \times v + r_i$ and $\beta$ vectors $\pi_j \leftarrow g_j \times v + r_j$, for $1 \leq i, j \leq \beta$.

Find the integer polynomial $w(x)$, which satisfies $w(x) \times v(x) = d \mod f(x)$, denote $W \leftarrow \text{Rot}(w, f)$.

Output $sk = \{d, w\}$ and $pk = \{\pi_i$ and $\pi_j\}$.

• Encryption:

Choose random $\tau - 1$ vector $s_{i,j}$ in \( \{ \sum_{t=1}^{n} s_{i,j,t} \mod 2 = 0, 1 \leq i, j \leq \beta - 1 \} \), a vector $s_{\tau}$ in \( \{ \sum_{t=1}^{n} s_{\tau,t} \mod 2 = m \} \) and a vector $s_{\tau+1}$ in \( \{ \sum_{t=1}^{n} s_{\tau+1,t} \mod 2 = m \} \)
3.1. Our Scheme with Compression Public Key

The Euclidean norm of all \( s \) are \( \| s \| \leq \zeta \).

Output a ciphertext \( c \leftarrow \sum_{i,j=1}^{\beta} s_{i,j} \times \pi_i \times \pi_j + s_{\tau+1} \).

- Evaluation and Decryption:
  
The decryption and evaluation are the same as the initial scheme.

3.1.2 Correctness of Somewhat Homomorphic Encryption

First, we recall the definition of \( r_{Enc} \) and \( r_{Dec} \) in Gentry’s idea \cite{26}.

Definition 3.1.1. \((r_{Enc} \text{ and } r_{Dec}) \) \cite{50} \( r \) represents the distance between a ciphertext \( \psi \) and the hidden ideal lattice \( \mathcal{L} \). \( r_{Enc} \) is the maximum possible distance for the encryption algorithm, and \( r_{Dec} \) is the minimum possible distance for the decryption algorithm.

We define the \( r_{pk} \) as the maximum distance between a public key \( \pi_{i,b} \) and the hidden ideal lattice. According to the KeyGen algorithm, we have \( r_{pk} \geq \theta \cdot \rho^2 \). The noise of a ciphertext is the production of \( s \) and \( r_{i,b} \) with the quadratic form in the encryption. The distance between a ciphertext and the hidden ideal lattice is \( r_{Enc} \leq \theta \cdot \rho^2 \cdot \zeta = \theta^2 \cdot \rho^2 \cdot \zeta \). Since \( \theta = \sqrt{n} \) is a constant of polynomial, we have \( r_{Enc} \leq n \cdot \rho^2 \cdot \zeta \). From the result in \cite{29}, it shows \( r_{Dec} \sim 2^n \). To decrypt the ciphertext correctly, we assume \( r_{Enc} < r_{Dec} \) which means \( n \cdot \rho^2 \cdot \zeta \leq 2^n \).

We will first prove the correctness of the decryption algorithm. For any ciphertext \( \psi \), we consider the ciphertext has two parts, \( \psi = a + b \), where \( a \in \mathbb{Z}^n \) and \( b \in \mathcal{L} \). Since \( a = \sum_{i,j=1}^{\beta} r_i \times r_j \times s_{i,j} + s_{\tau+1} \), we have \( \| a \| \leq n \rho^2 \zeta \). Because \( b \in \mathcal{L} \), we realize the only factor impacts on the decryption is \( a \), hence \( a = \psi \mod V = \psi - \lfloor \psi \cdot V^{-1} \rfloor \cdot V \). \( V^{-1} = W/d \) and the norm of the lattice \( \mathcal{L} \) \( d \) is odd. Since \( d \) and \( 2 \) is co-prime, so we have \( a \mod 2 = \lfloor \psi \cdot W/d \rfloor \mod 2 = \lfloor \psi \cdot w/d \rfloor \mod 2 = \psi' \mod 2 \). Therefore \( \psi' \mod 2 = a \mod 2 = \sum_{i,j=1}^{\beta} r_i \times r_j \times s_{i,j} + s_{\tau+1} \mod 2 \). Next consider \( \psi'(1) \)
3.2. The Security

\[ \text{mod } 2 = \sum_{i,j=1}^{\beta} r_i(1) \times r_j(1) \times s_{i,j}(1) + s_{\tau+1}(1) \mod 2, \]
since random vectors \( s_{i,j} \) are in \( \{ \sum_{t=1}^{n} s_{i,j,t} \mod 2 = 0, 1 \leq i \leq \tau - 1 \} \) and \( \{ \sum_{t=1}^{n} s_{\tau+1,t} \mod 2 = 0 \} \), we have \( \psi'(1) \mod 2 = r_{\beta,\beta}(1) s_{\beta,\beta}(1) \mod 2 = m \mod 2 \). The correctness of decryption has been proved.

Next, we prove the correctness of evaluation algorithm. Suppose \( \psi_1 = a_1 + b_1 \), and \( \psi_2 = a_2 + b_2 \) where \( a_1, a_2 \in \mathbb{Z}^n \), \( b_1, b_2 \in \mathcal{L} \), \( \|a_1\|, \|a_2\| \leq n\rho^2\zeta \), and \( a_t(x) = \sum_{i=1}^{\tau^2} r_{t,i}(x)s_{t,i}(x) + s_{\beta,\beta+1}(x) \mod f(x) \). Consider the Add algorithm first, \( \psi(x) \leftarrow \psi_1(x) + \psi_2(x) \mod f(x) = (a_1(x) + a_2(x)) + (b_1(x) + b_2(x)) \mod f(x) \). Since \( (b_1(x) + b_2(x)) \in \mathcal{L} \), we have \( \|Vec(a_1(x) + a_2(x))\| \leq r_{Dec} \), decryption will be \( a_1(1) + a_2(1) = m_1 + m_2 \). The add algorithm is correct.

Similarly for Multiplication algorithm, \( \psi(x) \leftarrow \psi_1(x) \times \psi_2(x) \mod f(x) = (a_1(x) a_2(x)) + (a_1(x)b_2(x)) + (a_2(x)b_1(x)) + (b_1(x)b_2(x)) \mod f(x) \). Since \( (b_1(x)b_2(x)), (a_1(x) b_2(x)) \) and \( (a_2(x)b_1(x)) \in \mathcal{L} \), we have \( \|Vec(a_1(x) \cdot a_2(x))\| \leq r_{Dec} \), decryption will be \( a_1(1) \times a_2(1) = m_1 \times m_2 \). Multiplication algorithm is correct.

3.2 The Security

In this section, we will prove our new scheme is semantically secure under the adaptation of the approximate greatest common factor (AGCD) assumption. The adversary can break the semantic security by instead of finding the vector \( \mathcal{V} \). The hash function family \( h(b) = \sum_{i=1}^{\tau} b_i \cdot \pi_i \) in the linear form is pairwise independent. By applying the leftover hash lemma, we want to prove that hash function family \( h'(b) = \sum_{i,j=1}^{\beta} b_{i,j} \cdot \pi_i \cdot \pi_j \) in the quadratic form is almost pairwise independent, which is \( \varepsilon \)-pairwise independent.
3.2. The Security

3.2.1 Leftover Hash Lemma

For a pairwise independent hash function family $\mathcal{H}$, the hash function $h : X \rightarrow Y$ holds $\Pr[h(x) = h(x')] = 1/|Y|$ for all $x \neq x'$. For our variant, the hash function family is $h' : \mathbb{Z}^{(n \times \beta) \times (n \times \beta)} \rightarrow \mathbb{Z}^n$, where $h'(b) = \sum_{i,j=1}^{\beta} b_{i,j} \cdot \pi_i \cdot \pi_j$. It is not an exactly pairwise independent, but it could be almost pairwise independent with parameter.

The following definition gives the $\varepsilon-$pairwise independent:

**Definition 3.2.1.** \cite{16}(\varepsilon-pairwise independent) A family $\mathcal{H}$ of hash function $h : X \rightarrow Y$ is $\varepsilon-$pairwise independent if

$$\sum_{x \neq x'} (Pr_{h \leftarrow \mathcal{H}}[h(x)] = h'(x)) - \frac{1}{|Y|} \leq |X|^2 \cdot \frac{\varepsilon}{|Y|}$$

The leftover hash lemma give by the prior definition.

**Lemma 3.2.1.** (Leftover Hash Lemma) \cite{16} Let $\mathcal{H}$ be a family of $\varepsilon-$pairwise independent hash functions. Suppose that $h \leftarrow \mathcal{H}$ and $x \leftarrow X$ are chosen uniformly and independently. Then $(h, h(X))$ is $(\frac{1}{2} \sqrt{|Y|/|X| + \varepsilon})-$uniformly over $\mathcal{H} \times Y$

**Lemma 3.2.2.** For an odd determinant $d$, the hash function family $\mathcal{H}$ is $\varepsilon-$ pairwise independence, with

$$\varepsilon = \frac{1}{d} + \frac{n^2 \tau}{2n^2 - 2n\beta}$$

3.2.2 Semantic Security

The semantic security of the scheme has been proved:

**Lemma 3.2.3.** If an algorithm $A$ breaks the semantic security with advantage $\varepsilon$, then there exist an algorithm $B$ that solves the Dec $\alpha$, $\beta - BDDHi_{n,\tau}$ with advantage of $3\varepsilon/32$. The running time of $B$ is polynomial in the running time of $A$. 
3.3 FHE Scheme

In this section, we follow the Gentry’s idea, use the bootstrapping technique to achieve fully homomorphic encryption scheme. Our scheme is similar as the initial scheme, the slightly difference is we using pseudo-random vector generator to construct the public key set. First, we will squash the decryption algorithm into a lower degree of the decryption polynomial. Then, we describe the bootstrappable scheme, which the post-processed ciphertext can be decrypted by modified decryption polynomial more efficiently.

3.3.1 The squashed Scheme

First, introduce four more parameters $\kappa$, $\sigma$, $\Theta$ and $\phi$ with functions of $\lambda$. More precisely, $\kappa = \eta + \gamma + 1 + \phi$, $\sigma = \lambda$, $\Theta = \tilde{O}(\lambda^3)$ and $\phi = \lceil \log_2(\sigma + 1) \rceil$. We will add a set of public key $y = \{y_1, \ldots, y_\Theta\}$ of rational numbers in $[0, 2)$ of $\kappa$ bits. There is a sparse subset $S \subset \{1, \ldots, \Theta\}$ of size $\sigma$ with $\sum_{i \in S} y_i \simeq w_i/d$. The ciphertext is expanded by computing with $y_i$. The secret key $sk$ is replaced by the binary vector of the subset $S$.

Instead of storing the whole set of $y_i$ in the public key, we will use the pseudo-random vector generator $f(se)$ with seed $se$ to generate $y_i$ for $2 \leq i \leq \Theta$. Then the public key consists of $se$ and $y_1$. The scheme will be as follows:

- **KeyGen($\lambda$):**
  
  Generated $sk^* = w_1, d$ and $pk^*$ for the somewhat scheme. Set $x_i =< x_1, \ldots, x_n >$ with $x_i \leftarrow \lceil 2^\kappa \times w_i/d \rceil$

  Choose $n$ vectors $s_i =< s_{i,1}, \ldots, s_{i,\Theta} >$ with $\Theta$-dimensional, each $s_i$ has hamming weight $\sigma$. Specifically, let $s_{i,1} = 1$ and $S = \{i, j : s_{i,j} = 1\}$

  Set $u_{i,1}$ such that $\sum_{i,j \in S} u_{i,j} = x_i$. Use $f(se)$ to generate vectors of $\Theta$-dimensional $u_i =< u_{i,1}, \ldots, u_{i,\Theta} >$ with $u_{i,j} \in [0, 2^{\kappa+1})$, for $2 \leq i \leq \Theta$. 

Set $y_{i,j} = u_{i,j}/2^\gamma$ and $y_i = \{y_{i,1}, \ldots, y_{i,\Theta}\}$, with $\gamma$ bits after binary point. 

$$[\sum_{j=1}^\Theta y_{i,j}]_2 = (w_i/d) - \Delta_d$$ for some $|\Delta_d| < 2^{-\kappa}$.

Output $sk = \{s_i\}$ and $pk = \{\pi_{i,j}, se, y_{i,1}\}$, for $i \in n$.

- **Encryption and Evaluation:** Given a ciphertext $\psi = \langle \psi_1, \ldots, \psi_n \rangle$, for each coefficient with respect to $\psi_i$ with $i \in n$, generate $z_j = \psi_j \cdot y_j$, for $j \in \{1, \ldots, \Theta\}$, and keep $\phi = \lceil \log_2(\sigma + 1) \rceil$ bits after binary point for each $z_j$.

Output $\psi$ and $z_j$

- **Decryption:**

$$\psi_i^* = [z_i \cdot s_i]$$ and $\psi^* \leftarrow \langle \psi_1^*, \ldots, \psi_n^* \rangle$, for $i \in n$.

$$\psi' = \sum \psi_i^*$$

$$\psi \leftarrow \psi'(1) \mod 2$$

### 3.3.1.1 Correctness

We first prove the correctness of the squashed algorithm by rounding off the noise. Our algorithm has assumption likes:

$$\psi_i^* = z_i \cdot s_i$$

$$= \psi_i \cdot y_i \cdot s_i$$

$$= \psi_i \cdot u_i \cdot s_i/2^\kappa$$

$$= \psi_i \cdot x_i/2^\kappa$$

$$= \psi_i \cdot \frac{2^\kappa \times w_i/d}{2^\kappa}$$

$$= \psi_i \times w_i/d$$

Hence, the algorithm is correct.

Next, we recall the definition of the permitted polynomial. When considering the noise, we will prove the scheme is correct for the set $C(\mathcal{P}_E)$ of circuit that computes
permitted polynomial.

Lemma 3.3.1. The squashed scheme is correct for the set \( C(\mathcal{P}_E) \) of the circuit that computed permitted polynomial. For every ciphertext \((\psi, z_i)\) that is generated by evaluating a permitted polynomial, it holds that \([s_i \cdot z_i]\) is within \(1/2\).

Proof. Fix a permitted polynomial \( P(x_1, \ldots, x_t) \in \mathcal{P}_E \), an arithmetic circuit \( C \) can compute \( P \), and \( t \) fresh ciphertext \( c_1, \ldots, c_t \) that encrypts the input into \( C \). Denote \( \psi = \text{Evaluate}(pk, C, c_1, \ldots, c_t) \). Meanwhile, fix the public key and the secret key with respect to security parameter \( \lambda \). For each \( i \in \mathcal{P}_E \), we have \( y_i = \langle y_{i,1}, \ldots, y_{i,\Theta} \rangle \) as the integer vectors in the public key and \( s_i = \langle s_{i,1}, \ldots, s_{i,\Theta} \rangle \) as binary vectors in the secret key. From the above assumption, we need to prove \([\psi_i \cdot w_i/d - z_i \cdot s_i]\) = \([\psi_i \cdot w_i/d - s_i \cdot (\psi_i \cdot y_i) - s_i \cdot \Delta_i]\)

\[= [\psi_i \cdot w_i/d - \psi_i \cdot (s_i \cdot y_i) - s_i \cdot \Delta_i] \]
\[= [\psi_i \cdot w_i/d - \psi_i \cdot (w_i/d - \Delta_d) - s_i \cdot \Delta_i] \]
\[= [\psi_i \cdot w_i/d - \psi_i \cdot w_i/d + \psi_i \cdot \Delta_d - s_i \cdot \Delta_i] \]
\[= [\psi_i \cdot \Delta_d - s_i \cdot \Delta_i] \]
Recall $[s_i \times y_i] = (w_i/d) - \Delta_d$ with $\Delta_d \leq 2^{-\kappa}$. We have

$$||\psi_i \cdot \Delta_d - s_i \cdot \Delta_i|| \leq ||\psi_i \cdot \Delta_d| + |s_i \cdot \Delta_i||$$

$$\leq 2^{\gamma+n} \cdot 2^{-\kappa} + \sigma \cdot \frac{1}{2(\sigma + 1)}$$

$$= 2^{\gamma+n-\kappa} + \sigma \cdot \frac{1}{2(\sigma + 1)}$$

$$= 2^{\gamma+n-\kappa} + \sigma \cdot \frac{1}{2(\sigma + 1)}$$

$$< \frac{1}{2(\sigma + 1)} + \frac{\sigma}{2(\sigma + 1)}$$

$$= 1/2$$

Therefore, the claim follows.

\[ \square \]

## 3.3.2 Bootstrapping

In this section, we will prove the squashed scheme is bootstrappable. From Gentry’s idea \cite{26}, our scheme can achieve fully homomorphic for circuit of any depth.

**Theorem 3.3.2.** \cite{26} Let $\mathcal{E}$ be the scheme above, and let $\mathcal{D}_E$ be the set of augmented (squashed) decryption circuits. Then, $\mathcal{D}_E \subset C(\mathcal{P}_E)$.

**Proof.** To prove $\mathcal{E}$ is bootstrappable, we need to show the modified decryption $m \leftarrow \sum [z_i \cdot s_i](1) \mod 2$ is a permitted polynomial size circuit. Recall $s_i = <s_{i,1}, \ldots, s_{i,\Theta}>$ for each $i \in n$ are binary number vectors and each $s_{i,j}$ is a bit, similarly, $z_i = <z_{i,1}, \ldots, z_{i,\Theta}>$ for each $i \in n$ are rational number vectors and each $z_{i,j}$ is rational number in $[0, 2)$, keeping $\phi = \lceil \log(\sigma + 1) \rceil$ bits of precision after the binary point. We also proved $\sum [s_i \cdot z_i]$ is within $1/2$, and the Hamming weight is $\sigma$ of the bits $s_i$ for each $i$. The computation algorithm of the decryption can be split into four steps:

- Step 1: For $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, \Theta\}$, set $a_{i,j} \leftarrow s_{i,j} \cdot z_{i,j}$. $a_{i,j} = z_{i,j}$ when $s_{i,j} = 1$ and $a_{i,j} = 0$ when $s_{i,j} = 0$, with $\phi = \lceil \log(\sigma + 1) \rceil$ bits of precision after
the binary point in binary representation. Set the vectors \( a_i = \langle a_i, 1, \ldots, a_i, \Theta \rangle \).

- Step 2: Set the vectors \( x_j = \sum a_j \) for each \( i \in n \).

- Step 3: For each \( i \in n \), from the \( \Theta \) rational numbers \( \{x_j\}_{j=1}^{\Theta} \), generate other \( \phi + 1 \) rational numbers \( \{y_t\}_{t=1}^{\Theta+1} \), each \( y_t \) has less than \( \phi \) bits of precision, and satisfy \( \sum x_j = \sum t y_t \mod 2 \)

- Step 4: Output \( m \leftarrow \sum_t y_t(1) \mod 2 \)

Step 1 and 2 require only constant depth, because when adding vectors, there is no expensive carry operations needed. For step 3, we will apply the grade-school addition to handle the carries, and the carries are constant-depth. The last step, we can just use the a constant depth circuit having polynomial fan-in add-gates and constant fan-in mult-gates. Therefore, the total degree of the squashed scheme depends on the decryption polynomial in the binary representation.

From [29], the bound of the noise in the evaluated ciphertext is \( r_{Eva} \leq (r_{Enc})^d \times \sqrt{m} \), where \( d \) is the degree of the polynomial and \( m \) is the number of monomials. For elementary symmetric polynomials with the degree of \( d \), the number of monomials is \( m = \prod_{i=0}^{\lfloor \log_2 d \rfloor} \binom{d}{2^i} \sim 2^{m'} \). To guarantee the ciphertext is inside the decryption radius of the secret key, we have

\[
(r_{Enc})^d \sqrt{m} = (n\rho^2 \zeta)^d \sqrt{m} \leq 2^n. \tag{3.1}
\]

The degree of the permitted polynomial is \( d \leq (\eta - m'/2)/(\log (n\rho^2 \zeta)) \). From the result in [29], we need to support the degree of the polynomial up to \( d \), then we have \( \eta \geq d \cdot \log (n\rho^2 \zeta) + \log \sqrt{m} \) to evaluate the "squashed decryption circuit" for deep enough circuits.
3.3.3 Security of The Squashed Scheme

The security of the squashed scheme is based on the sub set sum problem. To recover the secret key, the attacker has to find all the coefficient of $w_i$. Since we public the polynomial vectors instead of ideal lattice, the attacker is not able to recover $w_i$ correctly. The brute force attack on chosen ciphertext can achieve the goal. To solve the $n$ different $t$, the complexity is $(\sigma)^n$ [50].

3.4 Attacks

The SHE and FHE is secure against chosen plaintext attacks. But no SHE and FHE scheme can be IND-CCA2 secure, based on the fact that the adversary is allowed to manipulate the challenged ciphertext and submit it to the decryption oracle in an IND-CCA2 attack. The IND-CCA1 has been proved to be not secure for FHE and SHE scheme [39]. Zhang provided a way to recovery the secret key by using the decryption oracle over the DGHV scheme [65] [66]. Chenal gave more algorithm to allow an adversary to recover the pubic keys through decryption oracle queries [11]. Since the scheme based on the two cryptosystem: ideal lattice based system and integer based system. We consider the attacks on the approximate-gcd problem [35] and the BDD problem [38].

3.4.1 Brute Force Attack

The simplest attack is brute force attack on the noise in the public key. Since the ciphertext is protected by noise, this attack will guess the possible noise to recover the secret key. Since each public key $\pi_i = g_i \times v + r_i$, the $v$ can be computed by $v = \gcd(\pi_1 - r_1, \ldots, \pi_2\beta - r_2\beta)$. The scheme needs the number of possible $r_i$ more than $2^\lambda$ to against the attack. The Stehle-Zimmermann algorithm to compute the GCD’s
3.4. Attacks

told us, the time complexity of the algorithm is $\tilde{O}(\gamma + \eta)$ for the norm of the vectors of $\gamma + \eta$ bits. The attack complexity is $\rho \cdot \tilde{O}(\gamma + \eta)$. Therefore, the attack is thwarted when $\rho = \omega(\log \lambda)$.

3.4.2 Birthday Attack

To resist the birthday attack on our scheme, the running time of the attack needs to be greater than $2^{\log \rho / 2}$ [29]. So the bit length of the noise has to be at least $2\lambda$ bits to against this attack, and the possible numbers of the noise relative to a single key is at least $2^{\lambda / 2}$.

3.4.3 SDA-Simultaneous Diophantine Approximation

In this section, we start with the know attack based on AGCD problem. To solve the AGCD problem with many numbers, we can apply simultaneous Diophantine approximation (SDA) [37]. The element in our variant is polynomial vector rather than single integer, we can not apply the SDA directly. Fortunately, the coefficient of each polynomial is integer, then we can modify the SDA to find the target vector.

First, we generate a lattice $\mathcal{L}(B)$ by spanning the rows with $k + t \leq 2\beta$ public keys.

$$
B = \begin{bmatrix}
\theta \text{Mult} \cdot I_n & \text{Rot}(\pi_2) & \text{Rot}(\pi_3) & \ldots & \text{Rot}(\pi_{k+t}) \\
0 & -\text{Rot}(\pi_1) & 0 & \ldots & 0 \\
0 & 0 & -\text{Rot}(\pi_1) & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & -\text{Rot}(\pi_1)
\end{bmatrix}
$$

For $\pi_1 = r_1 + g_1$ and $\pi_i = r_i + g_i$ where $1 \leq i \leq k + t$, we have $\pi_i g_1 - \pi_1 g_i = r_i g_1 - r_1 g_i$.

Then we can apply LLL reduction algorithm to find the vector $u =< \theta \cdot \rho g_1, r_1 g_1 - r_1 g_1, r_2 g_1 - r_1 g_2, \ldots, r_{k+t} g_1 - r_1 g_{k+t}> =< \theta \text{Mult} \cdot \rho \cdot g_1, r_1 g_1(\frac{a_2}{g_1} - \frac{a_2}{g_1}), r_1 g_1(\frac{a_3}{g_1} - \frac{a_3}{g_1}), \ldots, r_1 g_1(\frac{a_{k+t}}{g_1} - \frac{a_{k+t}}{g_1}) >$. Once the attacker find the vector $u$, they can recover all
The attack problem comes to the lattice reduction problem. Based on the lattice reduction algorithm [9], the target vector $u$ cannot be found if \( \frac{\lambda_2(L)}{\|u\|} < c^{n(k+t)} \), where $c$ is a constant reached to the smallest value of 1.009 [9].

From the definition of successive minima \( \lambda_2 \leq \left( \frac{n}{n/2} \det(L) \right)^{\frac{1}{n(k+t)}} \), we have \( \lambda_2 \leq \sqrt{n(k+t)\det(L)}^{\frac{1}{n(k+t)-1}} \). If \( \sqrt{n(k+t)}\det(L)^{\frac{1}{n(k+t)-1}} < c^{n(k+t)} \), the target vector $u$ cannot be computed by lattice reduction. Briefly, to guarantee $\|u\|$ is hard to be found, we need the matrix satisfy the condition of $\det(B) < c^{n(k+t)/n(k+t)-1}\|u\|^{n(k+t)}$. Since $\det(B) = (\theta \rho)^n\|\pi_1\|^{k+t-1}$, therefore we can get $\det(B) \leq (\theta \rho)^n(\theta \cdot \|g_1\| \cdot \|v\|)^{n(k+t-1)}$. As $\|u\| > \theta \cdot \rho \cdot \|g_1\|$, the successful attack achieves when $(\theta \rho)^n(\theta \cdot \|g_1\| \cdot \|v\|)^{n(k+t-1)} \geq c^{n(k+t)/n(k+t)-1}(\theta \cdot \rho \cdot \|g_1\|)^{n(k+t)} \geq c^{n^2(k+t)^2}(\theta \cdot \rho \cdot \|g_1\|)^{n(k+t)}$. We conclude an inequation as:

\[
\eta(k + t - 1) \geq n(k + t)^2 \log_2 c + \gamma + (k + t - 1) \log_2 \rho, \tag{3.2}
\]

and

\[
\log_2 c \leq \frac{(\eta - \log_2 \rho)(k + t - 1) - \gamma}{n(k + t)^2}. \tag{3.3}
\]

To get the right hand side maximum, we need to find the maximum value of

\[
\frac{\eta - \log_2 \rho}{n(k + t)} - \frac{\eta - \log_2 \rho - \gamma}{n(k + t)^2},
\]

since $\log_2 \rho$ is quite small compared with $\eta$, which means $k + t \sim \mathcal{O}(\frac{\gamma}{\eta})$ gives the best attack.

In our parameter setting, $k + t \sim \mathcal{O}(\frac{\gamma}{\eta})$. To resist the modified SDA attack, our numbers of public key has to satisfy $\frac{k}{\gamma} > 2\beta$. The attacker can use all public keys to give themselves best advantage. For $2\beta \geq \frac{k}{\gamma}$, the time to get a $2^\eta$ approximation is $2^{\gamma/\eta}$. Therefore to thwart this attack, we need $\gamma/\eta = \omega(\log \lambda)$. The setting of $\gamma$ is
\( \gamma = \omega(\eta \log \lambda). \)

### 3.4.4 Nguyen and Stern’s Orthogonal Lattice

Using Nguyen and Stern’s orthogonal lattice \[45\] is another way to operate lattice attack. The attacker will be failure if the dimension of the lattice is larger than the ratio of the bit length of the public key and the bit length of secret key \( k + t > (\gamma + \eta)/\eta \), more precisely, the target vector will not be covered when \( k + t > (\gamma + \eta)/(\eta - \log_2 \rho) \). The time complexity is roughly \( 2^{2\gamma/\eta^2} \).

We generate the lattice spanned by the row of the following \((t + k) \times (t + k + 1)\) matrix:

\[
B = \begin{bmatrix}
\text{Rot}(\pi_1) & R_1 I_n \\
\text{Rot}(\pi_2) & R_2 I_n \\
\vdots & \ddots \\
\text{Rot}(\pi_{k+t}) & R_{k+t} I_n \\
\end{bmatrix}
\]

The row \((i)\) represents the constraint \( \text{Rot}(\pi_i) - r_i I_n = 0 \mod V \), where \( R_i \) is an upper bound on \(|r_i|\). Let the vector \( v = < v_0, v_1, \ldots, v_\beta > = \sum_{i=1}^{k+t} g_i B_i n \), for \( n \in 1, \ldots, n \) in the lattice above. We obtain

\[
v_0 - \sum_{i=1}^{k+t} \frac{v_i}{R_i} r_i = \sum_{i=1}^{k+t} g_i (\text{Rot}(\pi_i) - r_i I_n) = 0 \mod V.
\]

The vector are orthogonal to \((1, -\frac{r_1}{R_1}, \ldots, -\frac{r_{k+t}}{R_{k+t}})\) by lattice reduction algorithm, then the noise \( r_i \) can be recovered.

The determinant of the lattice is approximately to the product of the columns of \( B \), which is \( \sqrt{k + t} \| \text{Rot}(\pi_i) \| \prod_{i=1}^{k+t} R_i \approx \sqrt{k + t} \| \text{Rot}(\Pi_i) \| R_i^{k+t} \), where \( \| \text{Rot}(\Pi_i) \| \) is the upper bound of \( \| \text{Rot}(\pi_i) \| \). Therefore, we have \( R^{k+t} \sqrt{\Pi} < \sigma \rho \), where \( k + t > (\gamma + \eta)/(\eta - \log_2 \rho) \). The parameter to against the know attack will be similar as the SDA attack.
3.4.5 Coppersmith’s Method

We consider the Coppersmith’s technique \cite{14} to attack on recovering the noise of public keys. Coppersmith’s method does not only focus on the relations \( \text{Rot}(\pi_i) - r_i I_n = 0 \mod V \), but also consider the relations like \((\text{Rot}(\pi_i) - r_i I_n)^2 = 0 \mod V\), or \((\text{Rot}(\pi_i) - r_i I_n) \times (\text{Rot}(\pi_i') - r_i' I_n) = 0 \mod V\). The lattice will be generated as follow. We still set \( \Pi \) as the bound of the public key and \( R \) is the bound of noise, and let all \( \pi_{i,j} \) are roughly the same size \( \Pi \). The first row of the matrix has size \( \tilde{O}(\Pi^d) \), \( d \) is the relations of product, where \( d \leq 2\beta \). The next \( 2\beta \) rows has size \( \tilde{O}(\Pi^{d-1}R) \) on the pivots position. In general case, on the pivots position, there are \( \binom{2\beta+d-1}{d} \) rows of the size \( \tilde{O}(\Pi^{d-1}R^i) \). Remaining rows are the size of \( \tilde{O}(R^d) \).

The determinant of the lattice \( \det(B) \approx \Pi^2 \cdot (\Pi R)^{2\beta} \cdot (R^2)^{2\beta} = \Pi^{2+2\beta}R^{4\tau-2} \). The attacker will take the best advantage if \( 2\beta \leq (\gamma - \rho)/(\eta - \rho) \). To against the attack, we need to choose the number of public key \( 2\beta > (\gamma - \rho)/(\eta - \rho) \sim O(\gamma/\eta) \), which is also close to the previous attack.

3.4.6 BDD-Bounded Distance Decoding

We will use BDD problem to recover the random vectors \( u \leftarrow 1, s_1, s_1, \ldots, s_\beta \). This is the know message attack to find the shortest vector by lattice reduction. By using the ciphertext, the matrix generate as follow:

\[
B = \begin{bmatrix}
1 & 0 & 0 & 0 & \ldots & 0 & \psi \\
0 & I_n & 0 & 0 & \ldots & 0 & \text{Rot}(\pi_1 \times \pi_1) \\
0 & 0 & I_n & 0 & \ldots & 0 & \text{Rot}(\pi_1 \times \pi_2) \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \ldots & I_n & \text{Rot}(\pi_\beta \times \pi_\beta)
\end{bmatrix}
\]

As previously, \( \det(\text{Rot}(\pi_1)) \leq ||\pi_1||^n \leq (\sigma ||g_1|| ||v||)^n \), therefore, \( \det(B) \leq ||\psi|| + \tau(\theta^2 ||g_1|| ||v|| ||g_2|| ||v||)^n \), and \( ||u|| = 1 + (\tau + 1)\zeta \). From the Minkowski bounds, we relax
the condition to against the best lattice reduction, we have

$$2 \log_2 \theta + 2 \gamma + 2 \eta < \tau (n \tau + 1) \log_2 c + \tau \log_2 (1 + \tau \zeta).$$

(3.4)

The number of public key has to be $n \cdot \tau + 1$ to against the known attack.

### 3.5 Extension of The HIL Encryption to Higher Degree

In the section 4, we use a quadratic form to compute ciphertext instead of linear form. Due to reduce the number of public key, the significant benefit of this scheme is reduce the size of the public key. Followed by this idea, we modified the scheme further by higher degree $t$ of the encryption procedure, $c \leftarrow \sum_{i_1, \ldots, i_t} s_{i_1, \ldots, i_t} \times \pi_{i_1} \times \pi_{i_2} \cdots \times \pi_{i_t} + s_{\tau+1}$. The key point is to prove the hash function family $\mathcal{H}_t$ with $h : \mathbb{Z}^{(n \beta)^t} \rightarrow \mathbb{Z}^n$ is almost a pairwise independent hash function family, then it is suitable to apply a variant of leftover hash lemma. The constraint will be $\beta^t \geq \gamma + \eta + \omega(\log \lambda)$.

To get the decryption correctly, we also need $r_{Enc} = \theta^t \cdot \rho^t \cdot \zeta < 2^n$, and $r_{Enc} = \sqrt{m(\theta^t \cdot \rho^t \cdot \zeta)^t} < 2^n$ for bootstrapping. We consider the constant factor $m \cdot \theta$ as small number, so $r_{Enc}$ requires $\sqrt{m} \cdot \theta^t (\rho^t \cdot \zeta)^t < 2^n$. To against the known attack, we need $\gamma$ to satisfy the equation

$$\gamma \geq (\eta - \log_2 \rho)(t \beta - 1) - n(t \beta)^2 \log_2 c,$$

(3.5)

and

$$t \log_2 \gamma + t \gamma + t \eta < \tau (n \tau + 1) \log_2 c + \tau \log_2 (1 + \tau \zeta).$$

(3.6)

We set a convenient parameter set as: $\rho = \lambda$, $\zeta = \lambda$, $\eta = \mathcal{O}(\lambda^2 \log^k \lambda^2)$, $\gamma =$
\(O(\lambda^3 \log^k \lambda^3), t = \log \lambda \) and \( \tau = \beta^t = O(\lambda^{3/t} \log^k \lambda^{3/t}) \). Now, we store \( \beta = O(\lambda^{3/t} \log^k \lambda^{3/t}) \) integers. Hence, the public key size becomes \( O(\lambda^4 \log^k \lambda^4) \) rather than \( O(\lambda^6 \log^k \lambda^6) \) in the original scheme.

### Table 3.1: Comparisons between Quadratic and Higher Degree

<table>
<thead>
<tr>
<th>Columns</th>
<th>Quadratic</th>
<th>Higher Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numbers of PK in Each Column</td>
<td>( \beta = O(\lambda^2 \log^k \lambda^2) )</td>
<td>( \beta = O(\lambda \log^\lambda \log^k \lambda^\log \lambda) )</td>
</tr>
<tr>
<td>Total Numbers of PK</td>
<td>( 2 \cdot \beta = O(\lambda^2 \log^k \lambda^2) )</td>
<td>( t \cdot \beta = O(\log \lambda \cdot \lambda \log^\lambda \log^k \lambda^\log \lambda) )</td>
</tr>
<tr>
<td>PK Size</td>
<td>( O(\lambda^6 \log^k \lambda^6) )</td>
<td>( O(\lambda^4 \log^k \lambda^4) )</td>
</tr>
</tbody>
</table>

### 3.6 Parameters and Constraints

We choose parameters under the following constraints:

- \( \rho = \omega(\log \lambda) \) to avoid brute force attack on noise.

- \( \eta \geq \log (n^\rho^2 \zeta) \cdot \Theta(\lambda/\log \lambda) \) to support the evaluation of squashed decryption circuits.

- \( \gamma = \omega(\eta \cdot \lambda) \) to against lattice-based attacks.

- \( \beta^2 \geq \log (\gamma + \eta) + \omega(\log \lambda) \) to use the leftover hash lemma in the reduction to approximate common vector.

- \( \zeta = \omega(\log \lambda) \) for secondary noise parameter.

- \( n = \omega(\lambda \log \lambda) \) to foils lattice-based reduction [29] and \( \theta = \sqrt{n} \) [26].

We set a convenient parameter set as: \( \rho = \lambda, \zeta = \lambda, \eta = O(\lambda^2 \log^k \lambda^2), \gamma = O(\lambda^3 \log^k \lambda^3) \) and \( \tau = \beta^2 = O(\lambda^3 \log^k \lambda^3) \). The main difference is that instead of having \( \tau = O(\lambda \log^k \lambda) \) integers, we store \( \beta = O(\lambda^{1.5} \log^k \lambda^{1.5}) \) integers. Hence, the public key size becomes \( O(\lambda^{4.5} \log^k \lambda^{4.5}) \) rather than \( O(\lambda^6 \log^k \lambda^6) \) in the original scheme.
3.6. Parameters and Constraints

We will use $\lambda = 80$ as an example to compare the new scheme and the initial scheme. Under the assumption that the dimension of the lattice is the square root of the dimension of the normal lattice.

Table 3.2: The Relations between Degree of Decryption Polynomial and Number of Monomials

<table>
<thead>
<tr>
<th>Degree of Decryption Polynomial</th>
<th>Number of Monomials</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3 = 2^2 - 1$</td>
<td>9</td>
</tr>
<tr>
<td>$7 = 2^3 - 1$</td>
<td>5145</td>
</tr>
<tr>
<td>$15 = 2^4 - 1$</td>
<td>$\sim 2^{34}$</td>
</tr>
<tr>
<td>$31 = 2^5 - 1$</td>
<td>$\sim 2^{75}$</td>
</tr>
<tr>
<td>$63 = 2^6 - 1$</td>
<td>$\sim 2^{176}$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$1023 = 2^{10} - 1$</td>
<td>$\sim 2^{3180}$</td>
</tr>
</tbody>
</table>

In this case, it requires the lattice with dimension 31 (From the Table 1) to be large enough to resist the lattice reduction. To resist birthday paradox attack, the maximum norm of each noise is $\sqrt{32}$. $s$ has $\tau + 1$ blocks, to stop the brute force attack, we set maximum 5 blocks with nonzero entries besides the $\tau$-th block. So we can find the number of public key used in the encryption scheme. The total sample is at least $(\frac{\tau+1}{5})(\frac{\tau}{2})^2 > 2^{80}$, which is $\tau = 111$. We keep the same security level of the squashed secret key by $\Theta = 6$ and $\sigma = 1$. We set maximum 11 coefficient to be 1 or $-1$, for $r_{Enc}$, the maximum norm of the noise in each ciphertext is $32 \cdot \sqrt{11^{3}} \sim 2^{10}$. Here, we use the suggestion in [29], the expansion factor for production of two random vectors is much small, we can consider $\|v_1 \times v_2\| \approx \|v_1\| \cdot \|v_2\|$ for our example in the bootstrapping. The worst case occurs when $r_{Enc} = 2^{10}$. To achieve the bootstrapping, $\eta$ has to satisfy the equation [1]: $2^\eta \geq \sqrt{2^{75}(2^{10})^{31}}$, therefore, $\eta = 348$. According to the know attack, we choose $\gamma = 7090$ as the smallest value to satisfy equation [5], [6] which guarantee the scheme is secure.

In our SHE scheme, the ciphertext size is $(348 + 7090) \times 31 \sim 225Kb$, the public key space is $(348 + 7090) \times 31 \times 22 \sim 4.8Mb$. In the squashed scheme, the public key
size is \((348 + 7090) \times 31 \times 6 \sim 1.3\text{Mb}\), the secret key size is \(225 \times 31 \times 6 \sim 40.9\text{Mb}\).

The whole scheme with the public key size is \(4.8 + 1.3 + 40.9 = 47\text{Mbits}\) which is much smaller than the original scheme with 173.5\text{Mbits}.

<table>
<thead>
<tr>
<th>Table 3.3: Comparisons with Original HIL Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Columns</td>
</tr>
<tr>
<td>Security Parameter</td>
</tr>
<tr>
<td>Lattice Dimension</td>
</tr>
<tr>
<td>Public Key Size</td>
</tr>
<tr>
<td>Ciphertext Size</td>
</tr>
</tbody>
</table>
Chapter 4

Conclusion and Future Work

The theory includes the basic notions of lattices, the computational problems of lattices, lattices basis reduction, the average-case hardness to worst-case hardness reduction, and the one-way trapdoor function from lattices. Moreover, the knowledge of ideal lattice is essential. The technique of lattice provides us the fundamental mathematical knowledge to carry out the research. By analyzing the existing fully homomorphic encryption schemes: security model, construction, functionality, and proof of technique. There are three categories of fully homomorphic encryption scheme: ideal lattice based scheme, integer based scheme and LWE based scheme. The construction of schemes are based on some mathematic tools: integer ring, polynomial ring, lattice, and ideal lattice.

By summarizing the result of the current works, we realized the efficiency of the fully homomorphic encryption scheme can be improved by reducing the key size. So we focus on how to apply the technique on reducing the key size especially the public key. We also identified that not all three categories can achieve this goal by the same technique. Therefore, the research aims to apply the public key compressing on different schemes and also applied on the batching schemes.

To answer the question is: how to reducing the size of public key in somewhat
homomorphic encryption? We use the same way to construct fully homomorphic encryption schemes. First, to construct somewhat homomorphic encryption and then use the bootstrapping to achieve fully homomorphic encryption schemes. That means, we have to make sure our somewhat homomorphic scheme is correct, then we can apply squashing and bootstrapping. We construct the squashed scheme which needs to do the "post process" on the ciphertext. The scheme will use a pseudo-random generator to generate public key. The ciphertext expansion will be generated by pseudo-random generator again. Next, to prove the correctness, the decryption, the addition and the multiplication. The attack will be the same as somewhat homomorphic encryption.

To answer the second question: How to construct a more efficient scheme based on the existing scheme with new technique? We focus on applying the public key compressing on the batching schemes to improve the efficiency of the fully homomorphic encryption. We will try to batch Plantard, Susilo and Zhang’s scheme. Then the scheme can encrypt a plaintext vector rather than single bit. If the goal is achieved, then we can apply the public key compression to see if the scheme has better key size with less time consuming. The correctness and security need to be proved, the known attack should be analyzed as well.
References


References


[31] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-
References


