On Lattice based Cryptographic Algorithms

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

DOCTOR OF PHILOSOPHY

By
Uddipana Dowerah

Supervisor:
Dr. Srinivasan Krishnaswamy

Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India
November, 2020

To

My Parents

Mrs. Aroti Gohain Dowerah and

Mr. Robindranath Dowerah

Certificate

This is to certify that the thesis entitled "**On Lattice based Cryptographic Algorithms**", submitted by **Uddipana Dowerah** (136102006), a research scholar in the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati, for the award of the degree of **Doctor of Philosophy**, is a record of an original research work carried out by her under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the institute and in my opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Date:

Place: Guwahati

Dr. Srinivasan Krishnaswamy
Dept. of Electronics and Electrical Engg.,
Indian Institute of Technology Guwahati,
Guwahati - 781039, Assam, India.

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Uddipana Dowerah

Roll No.: 136102006

Date:

Place: Guwahati

Acknowledgements

At the outset, I would like to express my sincere gratitude towards my supervisor Dr. Srinivsan Krishnaswamy without whom this dissertation would be far from complete. I am forever grateful to him for sharing his invaluable knowledge and ideas with infinite patience throughout the entire duration of this work. His continued guidance, feedback and encouragement throughout this journey has made this work a great learning experience. I would also like to thank him for going through the tedious task of reviewing and correcting all my manuscripts. It has been a great honour and privilege to work under his guidance.

I would like to thank my doctoral committee members Dr. Indrani Kar, Dr. Tony Jacob, Dr. Vinay Wagh and Dr. Brijesh Rai (former member) for their invaluable suggestions in shaping this work. I am deeply thankful to my former advisor Dr. Debasattam Pal (Dept. of EE, IIT Bombay) for his valuable guidance during the initial days of my research. I am extremely thankful to Prof. Chitralekha Mahanta (Dean, Academic affairs), Prof. Rohit Sinha (HoD, Dept. of EEE), Dr. Gaurav Trivedi, Dr. Smarajit Das and all the faculty members of the department of Electronics and Electrical Engineering, IIT Guwahati. I wish to extend my special thanks to Dr. Jan Pidanic, Dr. Zdeněk Němec and the Faculty of Electrical Engineering and Informatics, University of Pardubice, Czech Republic for hosting me in their institute for three months.

I wish to thank the technical and office staff members of the department, Mr. Mukut Baruah, Late Mr. Uday Shankar Uzir, Mr. Dasarath Das, Mr. Sundeep Borah, Mr. Sidananda Sonowal, Syed Samimul Mazid, Mr. Sanjib Das, Mrs. Chayanika Borah Majumdar and Ms. Khurshida Yasmin for their help throughout my work. I also appreciate the support of my fellow collegues from the Control and Instrumentation laboratory.

I convey my heartfelt gratitute to my dearest friends Ganji, Gargi baa, Jitu, Kamakshi, Nilu, Vivek, Trusna, DJ, Suman Sir, Sundaram Sir, Charu Sir, Karnika Di, Gautam, Sumi and my constant companions from the past thirteen years, Bandita, Jayshree and Nayantara for making this journey a wonderful experience. I will always cherish the times I have spent with these wonderful people. Special thanks to Subrata, Abhijit and Kasturi for being the best collegues.

Last but not the least, I would like to thank my parents, my sisters Sandipana and Rituporna, and dear Pratanu for being my constant pillars of support. Words are never enough to express my gratitude for all they have done for me.

Date: Uddipana Dowerah

Abstract

In this thesis, we propose a few algorithms in the area of lattice-based cryptography. Lattice-based cryptography is the construction of cryptographic algorithms the security of which, can be based on the conjectured hardness of lattice problems. Some of the important features of lattice-based cryptography are simple and efficient constructions, resistance to attacks by quantum algorithms, strong security proofs based on the worst-case hardness of lattice problems, etc.

First, we propose a Fully Homomorphic Encryption (FHE) scheme using multivariate polynomial evaluations. The scheme is designed in the framework of LWE (Learning with Errors) based schemes and its security depends on the hardness of the LWE problem. In this thesis, we have tried to utilize the intrinsic homomorphism in polynomial rings in order to perform homomorphic multiplication. Unlike other LWE based schemes, the size of the ciphertext does not grow with multiplication. Further, the noise associated with the ciphertexts increases only linearly.

We then show that the multiplication technique used in the FHE scheme can be extended to previous LWE-based schemes. By doing this, we can avoid the process of relinearization and the associated change of key after homomorphic multiplication. The evaluation key for the proposed multiplication technique is a third order tensor. In order to recover the secret key from the evaluation key, a system of non-linear equations must be solved.

Finally, we introduce a decision problem called the Hidden Subspace Membership problem and prove its hardness with respect to the LWE problem.

সাৰাংশ

(Abstract in Assamese language)

এই গৱেষণা গ্ৰন্থখনত 'লেতিছ-ভিত্তিক ক্ৰিপ্তোগ্ৰাফি' (lattice-based cryptography)ৰ অন্তৰ্গত কেইটিমান algorithmৰ প্ৰস্তাৱ কৰা হৈছে | যিবিলাক cryptographic algorithmৰ নিৰাপত্তা অৰ্থাৎ security লেতিছ-ভিত্তিক problemৰ আনুমানিক কঠিনতা (conjectured hardness)ৰ ওপৰত নিৰ্ভৰ কৰে, সেইবিলাক algorithmৰ গঠন প্ৰণালীকে লেতিছ-ভিত্তিক ক্ৰিপ্তোগ্ৰাফি বোলে | লেতিছ-ভিত্তিক ক্ৰিপ্তোগ্ৰাফিৰ কিছুমান উল্লেখযোগ্য বৈশিষ্ট্য হৈছে - সহজ তথা দক্ষ নিৰ্মাণশৈলী, quantum-algorithm attack ৰোধ কৰিব পৰা ক্ষমতা ইত্যাদি |

পোনপ্ৰথমে এটি নতুন fully homomorphic encryption প্ৰণালীৰ প্ৰস্তাৱ কৰা হৈছে | এই প্ৰণালীখন প্ৰস্তুত কৰিবৰ বাবে multivariate polynomial evaluationৰ ব্যৱহাৰ কৰা হৈছে | উক্ত প্ৰণালীখন Learning with Errors (LWE) ভিত্তিক প্ৰণালীসমূহৰ পদ্ধতিত গঠন কৰা হৈছে আৰু ইয়াৰ নিৰাপত্তা LWE problemৰ কঠিনতাৰ ওপৰত নিৰ্ভৰশীল | Homomorphic operationsৰ বাবে এটি polynomial ভিত্তিক পূৰণ পদ্ধতিৰ ব্যৱহাৰ কৰা হৈছে | এই পদ্ধতিটোৰ বিশেষ আকৰ্ষণ হ'ল যে ই ciphertextৰ আকাৰ (size) একেই ৰখাত সহায় কৰে | তদুপৰি, homomorphic পূৰণৰ পিছত ciphertextৰ লগত জড়িত noiseও কেৱল ৰৈখিক (linearly) ভাৱেহে বাঢ়ে |

ইয়াৰ পিছত আমি দেখুৱাইছো যে ওপৰত উল্লেখিত পূৰণ পদ্ধতিটো পূৰ্বৰ LWE ভিত্তিক প্ৰণালীসমূহতো ব্যৱহাৰ কৰিব পৰা যায় | এই পদ্ধতিটো ব্যৱহাৰ কৰি relinearization অথবা key switchingৰ দৰে ব্যয়বহুল প্ৰযুক্তিসমূহ সম্পূৰ্ণৰূপে পৰিহাৰ কৰিব পাৰি | উক্ত পূৰণ পদ্ধতিটোৰ evaluation key হৈছে এটি তৃতীয় orderৰ tensor | আমি দেখুৱাইছো যে উক্ত tensorৰ পৰা গোপন অৰ্থাৎ secret key টো গণনা কৰি উলিয়াবলৈ `বহু চলকত অৰৈখিক সমীকৰণ প্ৰণালী' (system of multivariate non-linear equations)ৰ সমাধান কৰিব লাগিব |

গৱেষণা গ্ৰন্থখনৰ অন্তিম পৰ্যায়ত Hidden Subspace Membership (HSM) নামৰ এটি decision problemৰ সূচনা কৰা হৈছে আৰু LWEৰ সম্পৰ্কে ইয়াৰ কঠিনতাৰ প্ৰমাণ দিয়া হৈছে |

Contents

List of Symbols

λ	: Security parameter
$\mathbb{R}$	: Set of real numbers
$\mathbb{R}^+$	: Set of positive real numbers
$\mathbb{Q}$	: Set of rational numbers
$\mathbb{Z}$	: Set of integers
$\mathbb{Z}_{\geq 0}$	: Set of non-negative integers
$\mathbb{Z}^+$	: Set of positive integers
$\mathbb{N}$	: Set of natural numbers
$GF(2)$	: Galois field of order 2
$\mathbb{Z}_q$ or $\mathbb{F}_q$	: Finite field of prime order q whose elements are represented by the integers in the interval $\left(-\frac{q}{2}, \frac{q}{2}\right]$
$a \ mod \ q$	: Modular reduction of any $a \in \mathbb{Q}$ into the interval $\left(-\frac{q}{2}, \frac{q}{2}\right] \cap \mathbb{Z}$
$\lvert x \rvert$	: Absolute value of x for some $x \in \mathbb{R}$
$x \xleftarrow{\$} S$	: x is sampled uniformly at random from a set S
$x \xleftarrow{\$} \mathcal{D}$	: x is sampled from a distribution $\mathcal{D}$
$\lfloor x \rfloor$	: Floor function of x, it gives the greatest integer less than or equal to x
$\lceil x \rceil$	: Ceiling function of x, it gives the smallest integer greater than or equal to x

$\lceil x \rceil$	: Rounding of x to the nearest integer
log	: logarithm to the base 2
$\boldsymbol{v} \in \mathbb{F}_q^n$	: a row vector of order n over $\mathbb{F}_q$
$\|\boldsymbol{v}\|_1$	: ℓ_1 norm of a vector $\boldsymbol{v}$
$\|\boldsymbol{v}\|_\infty$	: ℓ_∞ norm of a vector $\boldsymbol{v}$
$\|\boldsymbol{v}\|$	: Euclidean or ℓ_2 norm of a vector $\boldsymbol{v}$
$\langle \boldsymbol{v}_1, \boldsymbol{v}_2 \rangle$	: Inner product of two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$
$(\boldsymbol{v}_1, \boldsymbol{v}_2)$	: Concatenation of vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ given by $[\boldsymbol{v}_1 \parallel \boldsymbol{v}_2]$
$\boldsymbol{v}^T$	: Transpose of a vector $\boldsymbol{v}$
$\boldsymbol{v}_1 \odot \boldsymbol{v}_2$	: component-wise product of two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$
$\mathbb{Z}_q[x_1, \ldots, x_n]$	: Ring of polynomials in indeterminates $x_1, \ldots, x_n$ with coefficients from $\mathbb{Z}_q$
$\langle f_1, \ldots, f_t \rangle$	: Ideal generated by the polynomials $f_1, \ldots, f_t \in \mathbb{Z}_q[x_1, \ldots, x_n]$
$\boldsymbol{A}(i, :)$	: i^{th} row of matrix $\boldsymbol{A}$
$\boldsymbol{A}(:, i)$	: i^{th} column of matrix $\boldsymbol{A}$
$span(\boldsymbol{A})$	: The row span of a matrix $\boldsymbol{A}$
$Ker(\boldsymbol{A})$	: Null space of a matrix $\boldsymbol{A}$
$G_n(\mathbb{Z}_q^\ell)$	: Grassmannian of n dimensional subspaces of $\mathbb{Z}_q^\ell$
$\omega(\cdot)$, $\mathcal{O}(\cdot)$, $\Omega(\cdot)$	: Asymptotic notations
$\tilde{\mathcal{O}}(\cdot)$, $\tilde{\Omega}(\cdot)$	: Asymptotic notations with logarithmic factors suppressed

List of Abbreviations

CVP	: Closest Vector Problem
DLWE	: Decisional Learning with Errors
FHE	: Fully Homomorphic Encryption
HSM	: Hidden Subspace Membership
IND-CPA	: Indistinguishability under Chosen Plaintext Attack
IND-CCA1	: Indistinguishability under Chosen Ciphertext Attack
IND-CCA2	: Indistinguishability under Adaptive Chosen Ciphertext Attack
LLL	: Lenstra, Lenstra and Lovász
LPN	: Learning Parity with Noise
LWE	: Learning With Errors
PPT	: Probabilistic Polynomial Time
RLWE	: Ring Learning with Errors
SIMD	: Single Instruction Multiple Data
SIVP	: Shortest Integer Vector Problem
SVP	: Shortest Vector Problem

List of Publications

Journal

1. **Under review:** Uddipana Dowerah and Srinivasan Krishnaswamy, "Fully Homomorphic Encryption based on Multivariate Polynomial Evaluation", paper submitted to *IET Information Security*.

2. Uddipana Dowerah and Srinivasan Krishnaswamy, "A New Multiplication Technique for LWE Based Fully Homomorphic Encryption," in *IEEE Letters of the Computer Society*, vol. 3, no. 2, pp. 62-65, 1 July-Dec. 2020, doi: 10.1109/LOCS.2020.3021706.

Conference

1. Uddipana Dowerah and Srinivasan Krishnaswamy, "A Somewhat Homomorphic Encryption Scheme based on Multivariate Polynomial Evaluation," *29th International Conference Radioelektronika (RADIOELEKTRONIKA)*, pp. 1-6, IEEE, 16-18 April, 2019, Czech Republic.

2. Uddipana Dowerah and Srinivasan Krishnaswamy, "A New Symmetric Key Homomorphic Encryption Scheme", *23rd International Symposium on Mathematical Theory of Networks and Systems (MTNS),* 16-20 July, 2018, Hongkong.

Chapter 1

Introduction

The conjectured hardness of problems associated with lattices can be used to develop secure cryptographic algorithms. Cryptosystems [DH76, RSA78, RAD78, ElG84] based on number-theoretic assumptions like integer factorization or discrete logarithm were rendered insecure with the development of quantum algorithms [Sho99]. No such algorithms are known for solving lattice-based problems. Therefore, cryptographic constructions whose hardness depends on these problems are promising candidates for post-quantum cryptography. Also, such constructions are simple as they involve only linear operations and are highly parallelizable. Further, it is possible to have encryption schemes whose security depends on the worst-case hardness of lattice problems [Ajt96].

Some well known computational problems based on lattices are the Shortest Vector Problem (SVP), the Closest Vector Problem (CVP) and the Shortest Integer Vector Problem (SIVP). The most basic of these problems is the shortest vector problem. Given an arbitrary basis for a lattice $\mathcal{L}$, the shortest vector problem is to find the shortest non-zero vector in $\mathcal{L}$. However, for cryptographic purposes, one generally considers the approximated versions of these problems specified by an approximation factor γ. In the approximate

shortest vector problem denoted as SVP_γ, the challenge is to output a vector whose length is γ times the length of the shortest non-zero vector in the lattice. These problems are known to be NP-hard for small approximation factors, viz., $\gamma \leq \mathcal{O}(1)$ [Ajt98, DKS98, HR07, Kho05, Kho09, Mic01]. These lattice problems can be solved using lattice basis reduction algorithms. The best known such algorithm that runs in polynomial time is the LLL algorithm by Lenstra, Lenstra and Lovász [LLL82]. However, the LLL algorithm and its variants [Sch87, AKS01] achieve only slightly sub-exponential approximation factors. On the other hand, the algorithms that achieve polynomial approximation factors require exponential run time [Kan83, MV13, ADRSD15]. Therefore, there are no known polynomial time algorithms that approximate lattice problems to within polynomial approximation factors [MR09].

The first cryptographic construction based on the worst-case hardness of lattice problems was proposed in [Ajt96]. Subsequently, Ajtai and Dwork gave the first lattice-based public key encryption scheme with a security proof based on the worst-case hardness assumptions of lattice problems [AD97]. Another significant construction based on lattices is the NTRU cryptosystem proposed in [HPS98]. It is the first cryptographic construction to use polynomial rings and is practically efficient but lacks a supporting proof of security. Regev, in his seminal work [Reg05] introduced the well-known Learning with Errors (LWE) problem and gave the first public key encryption scheme based on its hardness. Ever since, LWE has been extensively used to design some of the best known cryptographic constructions based on lattices.

The LWE problem is an extension of the Learning Parity with Noise (LPN) problem. It can also be seen as the problem of decoding random linear codes. The average-case hardness of the LWE problem can be reduced to the worst-case hardness of the above mentioned lattice prob-

lems [Reg05, Reg09, BLP+13]. The concrete hardness of LWE has been extensively studied in [LP11, ACF+15, APS15]. Among other cryptographic primitives [Reg09, ACPS09, GPV08, LP11, ADPS16, Geo11], LWE is mainly used in the construction of fully homomorphic encryption schemes [BV11, Bra12, GSW13, HAO16, BV14a].

In this thesis, we present the following results in the area of lattice-based cryptography.

1. We present a fully homomorphic encryption scheme using multivariate polynomials. The scheme is designed in the framework of LWE-based schemes and its security depends on the hardness of the LWE problem. For homomorphic multiplication, we use a polynomial-based technique that does not increase the size of the ciphertexts.

2. We show that the multiplication technique in the proposed FHE scheme can be extended to previous LWE-based schemes.

3. We introduce a decision problem called the Hidden Subspace Membership (HSM) problem and give evidence of its hardness with respect to the hardness of the LWE problem.

1.1 Fully Homomorphic Encryption

Fully homomorphic encryption enables computation of arbitrary mathematical functions on encrypted data without decryption. As a result, data can be outsourced to a cloud service for storage and computation without compromising its privacy. The computations to be performed are specified in terms of either boolean or arithmetic circuits.

Homomorphic encryption can be classified mainly into three types – Partially Homomorphic Encryption, Somewhat Homomorphic Encryption and

Fully Homomorphic Encryption. If an encryption scheme can evaluate circuits with only one type of gates, i.e., either addition or multiplication, then it is called *partially homomorphic*. An encryption scheme is called *somewhat homomorphic* if it can evaluate circuits of only limited complexity. An encryption scheme that can evaluate circuits of arbitrary complexity is said to be *fully homomorphic*.

The notion of fully homomorphic encryption was introduced in [RAD78]. This notion was called *privacy homomorphism* back then. Thirty years later, the first construction of fully homomorphic encryption was proposed in [Gen09]. Even before the construction of [Gen09], several encryption schemes with partial and somewhat homomorphic capabilities were proposed. This includes the RSA encryption algorithm proposed in [RSA78] which is homomorphic with respect to multiplication. Other such constructions include the Goldwasser-Micali cryptosystem [GM82], the ElGamal cryptosystem [ElG84], Paillier [Pai99] and Benaloh [Ben87] cryptosystems, cryptosystems by Sander-Young-Yung [SYY99], Ishai-Paskin [IP07], Boneh-Goh-Nissim [BGN05] etc.

Apart from the conventional schemes, attempts were made to construct homomorphic encryption schemes using different algebraic structures. Homomorphic encryption using lattices and linear codes were proposed in [MCG08, MGH10, PW11, AS08, AAPS11]. A family of schemes called Polly Cracker [FK93, BCE$^+$94] uses multivariate polynomial algebra. These schemes are naturally homomorphic with respect to addition and multiplication. However, the Polly Cracker based schemes are vulnerable to attacks using Gröbner basis construction algorithms as well as attacks using linear algebra [BCE$^+$94, LdVMPT09]. Noisy variants of these cryptosystems were proposed in [AFFP11, Her12, AFF$^+$16] to overcome these vulnerabilities. However, in all variants of Polly Cracker as well as multiplicatively homomorphic

schemes based on lattices and linear codes [MGH10, AS08, AAPS11], the size of the ciphertext expands exponentially after multiplication.

Following the initial construction of a Fully Homomorphic Encryption (FHE) [Gen09] scheme, similar schemes were proposed in [VDGHV10, SV10, BV11, GH11, GHS12b, GHS12a, CMNT11, CCK+13]. These schemes follow a similar design policy and are collectively called the first generation of FHE. In these constructions, a somewhat homomorphic scheme is converted to a fully homomorphic one using *bootstrapping*. Bootstrapping is the process of 'refreshing' a ciphertext when the noise after homomorphic operations grows too large. The refreshed ciphertext is an encryption of the same message with reduced noise. However, the complexity of bootstrapping is very high and as an alternative, a second generation of schemes were proposed [BV14a, BGV14, Bra12, FV12] based on the hardness of the LWE problem and its ring variant Ring Learning with Errors (RLWE). In these schemes, homomorphic multiplication blows up the size of the ciphertext and the noise associated with it. Various new techniques like *relinearization* and *modulus switching* were proposed [BV14a, BGV14] to deal with these blow-ups. Using these techniques, a (leveled) fully homomorphic encryption scheme can be obtained from a somewhat homomorphic one without bootstrapping (a leveled FHE scheme can evaluate circuits of 'fixed' arbitrary complexity). These schemes require an evaluation key in order to perform homomorphic multiplication.

Further, a third generation of schemes [GSW13, DM15, CGGI16a], starting with GSW (Gentry-Sahai-Waters) [GSW13], were proposed without the expensive relinearization procedure. It is based on the *approximate eigenvector method* and the ciphertexts are matrices in this case. It removes the need for an evaluation key to obtain a leveled FHE scheme and homomorphic

addition and multiplication are roughly matrix addition and multiplication respectively. Further, it was observed in [BV14b] that the noise during homomorphic multiplication in GSW grows in an asymmetric fashion and this can be used to achieve bootstrapping with weaker hardness assumptions. The GSW scheme has a large performance overhead which can be improved significantly by relying the hardness on the RLWE problem. Ring-LWE variants of the GSW scheme were developed in [DM15, CGGI16a].

The GSW scheme [GSW13] together with BGV (Brakerski-Gentry-Vaikuntanathan) [BGV14] and B/FV (Brakerski/Fan-Vercauteren) [Bra12, FV12] are standard constructions of homomorphic encryption [ACC+19]. Many subsequent work deals with improving the performance of these schemes for efficient implementation of homomorphic encryption. Various optimizations like SIMD (Single Instruction Multiple Data) style *batching* [SV14, GHS12b] and faster bootstrapping techniques were proposed to improve the efficiency of the second generation schemes [ASP13, GHS12a, HS15, HS18, CJP20]. Further improvements of second generation schemes include the CKKS scheme [CKKS17] that introduces homomorphic encryption for approximate numbers, and various software and hardware implementations of the B/FV scheme [PRR17, SEA20, BEHZ16, HPS19, RTJ+19, TRV20] based on the RLWE problem. Optimizations for GSW and its variants mainly focus on achieving faster bootstrapping techniques [DM15, CGGI16a, CGGI17]. For instance, a single bootstrapping can be executed in less than a second in [DM15] which was further sped up to less than a 0.1 seconds in [CGGI16a]. All of these improvements and optimizations have led to the development of various FHE libraries [HS14, PRR17, SEA20, DM15, CGGI16b] and compilers [Cro17, CDS15, CMTM18, VS18, vEPIL19, DKS+20, VJH21].

In this thesis, we propose a multi-bit leveled fully homomorphic encryp-

tion scheme based on multivariate polynomial evaluations. The idea is to represent an encryption of zero by noisy evaluations of a polynomial, sampled uniformly at random from a secret ideal. For homomorphic operations, we use a polynomial-based multiplication technique that inherits multiplicative homomorphism from the corresponding property of polynomial rings. The security of the scheme depends on the hardness of the LWE problem. The proposed scheme is first presented in a manner similar to the LWE variants of BGV [BGV14] and B/FV [Bra12]. However, unlike these schemes, homomorphic multiplication does not increase the size of the ciphertexts in the proposed scheme. Therefore, there is no need for relinearization. The noise associated with the ciphertexts increases only linearly with each homomorphic operation. Hence, a leveled FHE scheme can be obtained without modulus switching. The per-gate computation of the scheme is $\tilde{\mathcal{O}}(\lambda^3 \cdot L^2)$ where λ denotes the security parameter and L denotes the multiplicative depth of the circuit. In its current form, the proposed scheme performs significantly better than other LWE based schemes.

Further, we show that the polynomial-based multiplication technique can be extended to other LWE-based schemes [BV14a, BGV14, Bra12]. This removes the need for relinearization or key switching after the multiplication process. It also removes the need for modulus switching since the increase in noise is linear after every multiplication,

1.2 Hidden Subspace Membership

In this thesis, we introduce an extension of the decisional learning with errors problem called the Hidden Subspace Membership (HSM) problem. This problem is also a generalization of the LWE problem with multiple secrets. The HSM problem is to distinguish elements of a subspace perturbed with noise

from those sampled uniformly at random from the vector space. In other words, given a subspace $\mathcal{S}$ of a vector space $\mathcal{V}$ and a noise distribution $\mathcal{N}$ on $\mathcal{V}$, the HSM problem is to distinguish a vector in $\mathcal{S}+\mathcal{N}$ from a vector in the uniform distribution on $\mathcal{V}$. If $\mathcal{S}$ is an n-dimensional subspace of the vector space $\mathcal{V} = \mathbb{Z}_q^\ell$ for some $n, q, \ell \in \mathbb{N}$, then the Learning with Errors problem with parameters n and q is a specific case of the HSM problem. We show that various instances of the HSM problem is as hard as the Learning with Errors problem.

1.3 Outline of the Thesis

The remainder of the thesis is organized as follows. Chapter 2 contains the necessary mathematical and cryptographic preliminaries to be used in the rest of the thesis. In Chapter 3, we describe the construction of the (leveled) fully homomorphic encryption scheme based on the hardness of the Learning with Errors problem. We first describe a symmetric key variant of the scheme for simplicity and then convert it into a public key scheme. In Chapter 4, the multiplication technique from Chapter 3 is extended to previous LWE-based schemes [BV14a, BGV14, Bra12]. In Chapter 5, we describe the Hidden Subspace Membership problem and prove its hardness with respect to the Learning with Errors problem. Finally, in Chapter 6, we give a summary of the work done in this thesis and give possible research directions for future work.

1.4 Notations

The following notations are used in this thesis. λ denotes the security parameter. We use $\mathbb{R}$ to denote the set of real numbers, $\mathbb{Q}$ the set of rational numbers,

$\mathbb{Z}$ the set of integers and $\mathbb{N}$ the set of natural numbers. A finite field of order q for some prime $q \in \mathbb{N}$ is denoted by $\mathbb{Z}_q$ or $\mathbb{F}_q$ and its elements are represented by the integers in the interval $(-\frac{q}{2}, \frac{q}{2}]$. For some $a \in \mathbb{Z}$, we use $(a \ mod \ q)$ to denote the modular reduction of a by q into the interval $(-q/2, q/2] \cap \mathbb{Z}$. We use $\mathbb{Z}_q[x_1, \ldots, x_n]$ to denote the ring of polynomials in $x_1, \ldots, x_n$ with coefficients in $\mathbb{Z}_q$. Given polynomials $f_1, \ldots, f_t \in \mathbb{Z}_q[x_1, \ldots, x_n]$, $\langle f_1, \ldots, f_t \rangle$ denotes the ideal generated by $f_1, \ldots, f_t$. For a polynomial ring $\mathcal{R}$ and an ideal $\mathcal{I}$, $\mathcal{R}_{\leq r}$ and $\mathcal{I}_{\leq r}$ denote the set of polynomials in $\mathcal{R}$ and $\mathcal{I}$ respectively of degree at most r for some $r \in \mathbb{N}$. Given a set S, $x \xleftarrow{\$} S$ means that x is sampled uniformly at random from S. Similarly, $x \xleftarrow{\$} \mathcal{D}$ means that x is sampled from a distribution $\mathcal{D}$. For a real number x, $\lfloor x \rfloor$, $\lceil x \rceil$ and $\lfloor x \rceil$ denote the rounding of x down, up or to the nearest integer. For some $x \in \mathbb{R}$, $|x|$ denotes the absolute value of x. Scalars are denoted using plain letters, vectors using bold lowercase letters and matrices using bold uppercase letters. The i^{th} row or column of a matrix $\boldsymbol{A}$ is denoted using $\boldsymbol{A}(i, :)$ or $\boldsymbol{A}(:, i)$ respectively. Tensors are denoted by uppercase bold script letters $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}, \ldots$ etc. A vector $\boldsymbol{v}$ is usually a row vector unless stated otherwise. We use $(\boldsymbol{v}, \boldsymbol{w})$ to denote the concatenation of vectors $\boldsymbol{v}$ and $\boldsymbol{w}$ given by $[\boldsymbol{v} \parallel \boldsymbol{w}]$. The one norm of a vector $\boldsymbol{v}$ is denoted by $\|\boldsymbol{v}\|_1$, the Euclidean or ℓ_2 norm by $\|\boldsymbol{v}\|$ and the infinity norm by $\|\boldsymbol{v}\|_\infty$. The inner product of two vectors $\boldsymbol{v}, \boldsymbol{w}$ is denoted using $\langle \boldsymbol{v}, \boldsymbol{w} \rangle := \boldsymbol{v}\boldsymbol{w}^T$. The notation $\boldsymbol{v} \odot \boldsymbol{w}$ denotes the component-wise product of $\boldsymbol{v}$ and $\boldsymbol{w}$. We use 'log' to denote logarithm to the base 2. We use $\mathcal{O}(\cdot), \Omega(\cdot), \omega(\cdot)$ etc. to denote standard asymptotic notations. Further, $\tilde{\mathcal{O}}(\cdot), \tilde{\Omega}(\cdot)$ means that the logarithmic factors are suppressed in the main parameter.

Chapter 2

Preliminaries

This chapter contains an account of the mathematical and cryptographic background to be used in the remainder of the thesis. First, we recall some basic concepts from Abstract Algebra and Linear Algebra. Then, we discuss some basic definitions and computational problems based on lattices. Further, we give an overview of the main concepts related to fully homomorphic encryption.

2.1 Groups, Rings and Fields

Definition 2.1.1. (Group [LN97]). *A group $(\mathcal{G}, *)$ is a set $\mathcal{G}$ equipped with a binary operation $*$ such that it satisfies the following properties:*

- *$*$ is associative; i.e., $a * (b * c) = (a * b) * c$ for all $a, b, c \in \mathcal{G}$.*
- *There exists an element $e \in \mathcal{G}$ called the identity element such that $a * e = e * a = a$ for all $a \in \mathcal{G}$.*
- *For all $a \in \mathcal{G}$, there exists an inverse element a^{-1} such that $a * a^{-1} = a^{-1} * a = e$*

Additionally, if a group satisfies a fourth property where $a * b = b * a$ for all $a, b \in \mathcal{G}$, then it is called an *Abelian* or a *commutative* group. An example of a group is the set of integers together with the operation of addition denoted as $(\mathbb{Z}, +)$.

Definition 2.1.2. (Ring [LN97]). *A ring $\mathcal{R}(+, \cdot)$ is a set $\mathcal{R}$, equipped with two binary operations, '$+$' and '$\cdot$' that satisfies the following set of axioms.*

1. *$\mathcal{R}$ is a commutative group with respect to '$+$'.*
2. *'$\cdot$' is associative i.e., for all $a, b, c \in \mathcal{R}$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.*
3. *The distributive laws hold; i.e., for all $a, b, c \in \mathcal{R}$, we have $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(b + c) \cdot a = b \cdot a + c \cdot a$.*

Examples of rings include the set of integers $\mathbb{Z}$, the set of polynomials with integer coefficients $\mathbb{Z}[x]$, the set of all 2×2 matrices over the real numbers $\mathbb{R}$ etc. Further, we can have the following different types of rings.

1. A ring is called *commutative* if '$\cdot$' is commutative, i.e., for all $a, b \in \mathcal{R}$, $a \cdot b = b \cdot a$.
2. A *ring with identity* is a ring that has a multiplicative identity, i.e., for all $a \in \mathcal{R}$, there is an element $e \in \mathcal{R}$ such that $a \cdot e = e \cdot a = a$.
3. A ring is an *integral domain* if it is a commutative ring with identity $e \neq 0$ such that if $ab = 0$, then either $a = 0$ or $b = 0$.
4. If the non-zero elements of a ring forms a group under '$\cdot$', then it is called a *division ring*.
5. A commutative division ring is called a *field*.

Definition 2.1.3. (Subring). *A subset of a ring $\mathcal{R}$ that is closed under the operations of $+$ and $\cdot$ and forms a ring under these operations is called a subring of $\mathcal{R}$.*

Definition 2.1.4. (Ideal). *A subring $\mathcal{I}$ of a ring $\mathcal{R}$ is called a left (right) ideal of $\mathcal{R}$ if for all $r \in \mathcal{R}$ and $a \in \mathcal{I}$, $ra \in \mathcal{I}$ ($ar \in \mathcal{I}$).*

An ideal is called two-sided if it is both a left and a right ideal.

Let $\mathcal{R}$ be a commutative ring with identity. Then, the ideal generated by a single element $a \in \mathcal{R}$, given by $\mathcal{I} := (a) = \{ra \mid r \in \mathcal{R}\}$, is called the principal ideal generated by a.

Given a ring $\mathcal{R}$ and a (two-sided) ideal $\mathcal{I}$, an equivalence relation $\sim$ on $\mathcal{R}$ can be defined as follows:

$$\{a \sim b \text{ if and only if } a - b \in \mathcal{I}\}$$

The equivalence class or the residue class of an element $a \in \mathcal{R}$ is denoted by $[a] = a + \mathcal{I}$. The set of residue classes of $\mathcal{R}$ modulo $\mathcal{I}$ is denoted by $\mathcal{R}/\mathcal{I}$. It forms a ring called the residue class ring or the quotient ring with respect to the following operations.

$$(a + \mathcal{I}) + (b + \mathcal{I}) = (a + b) + \mathcal{I}$$
$$(a + \mathcal{I})(b + \mathcal{I}) = ab + \mathcal{I}$$

Definition 2.1.5. (Field). *A field is an algebraic structure given by a set $\mathbb{F}$ together with the operations of addition and multiplication such that the following properties are satisfied.*

1. *$\mathbb{F}$ is a commutative group with respect to addition.*

2. *If $\mathbb{F}^*$ denotes the non-zero elements of $\mathbb{F}$, then $\mathbb{F}^*$ is a commutative group*

with respect to multiplication.

3. *The multiplication operation is distributive over addition, i.e.,* $\forall\ a, b, c \in \mathbb{F}$, $a \cdot (b + c) = a \cdot b + a \cdot c$

The set of real numbers together with the operations of addition and multiplication denoted by $\mathbb{R}(+, \cdot)$ forms a field. The cardinality or the number of elements in this field is infinite. A field with finite number of elements is called a finite field.

Theorem 2.1.1. ([LN97]). *The ring of residue classes of the integers modulo the principal ideal generated by a prime q denoted by $\mathbb{Z}/(q)$ is a field.*

Definition 2.1.6. (Galois Field [LN97]). *For a prime q, let $\mathbb{F}_q$ be the set of integers $\{0, 1, \ldots, q-1\}$ and let $\phi : \mathbb{Z}/(q) \to \mathbb{F}_q$ be the mapping defined by $\phi([a]) = a$ for $a = 0, 1, \ldots, q-1$. Then $\mathbb{F}_q$ is a finite field with the operations of addition and multiplication modulo q, called the Galois field of order q.*

2.2 Multivariate Polynomials over Finite Fields

In this section, we briefly discuss a few definitions regarding multivariate polynomials that are relevant to this work.

Definition 2.2.1. (Monomial [CLO92]). *A monomial in n variables $x_1, \ldots, x_n$ is a product of the form $x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdots x_n^{\alpha_n}$ where $\alpha_i \in \mathbb{Z}_{\geq 0}$ for all i. The total degree of this monomial is $\sum_{i=1}^{n} \alpha_i$.*

If $\alpha = (\alpha_1, \ldots, \alpha_n)$ denotes a tuple of non-negative integers, then for simplicity, we can write $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$. We are now ready to define a multivariate polynomial in $x_1, \ldots, x_n$.

Definition 2.2.2. (Multivariate Polynomial). *Let $\mathbb{F}_q$ be a finite field of order q. Then, a polynomial f in $x_1, \ldots, x_n$ over $\mathbb{F}_q$ is a finite linear combination of monomials with coefficients in $\mathbb{F}_q$ given by*

$$f = \sum_{\alpha} a_\alpha x^\alpha, \quad a_\alpha \in \mathbb{F}_q$$

where the sum is over a finite number of n-tuples $\alpha = (\alpha_1, \ldots, \alpha_n)$. a_α is called the coefficient of the monomial x^α and for $a_\alpha \neq 0$, $a_\alpha x^\alpha$ is called a term of f and the maximum $|\alpha|$ for which $a_\alpha \neq 0$ is called the total degree of f.

The set of all polynomials in $x_1, \ldots, x_n$ with coefficients in $\mathbb{F}_q$ is denoted by $\mathbb{F}_q[x_1, \ldots, x_n]$ and forms a multivariate polynomial ring over $\mathbb{F}_q$. An ideal of $\mathbb{F}_q[x_1, \ldots, x_n]$ is the ideal generated by a finite set of polynomials $f_1, \ldots, f_s \in \mathbb{F}_q[x_1, \ldots, x_n]$ denoted as $\langle f_1, \ldots, f_s \rangle$.

Definition 2.2.3 (Ideal generated by $f_1, \ldots, f_s$ [CLO92]). *The ideal generated by a finite set of polynomials $f_1, \ldots, f_s \in \mathbb{F}_q[x_1, \ldots, x_n]$ is defined as*

$$\langle f_1, \ldots, f_s \rangle = \left\{ \sum_{i=1}^{s} h_i f_i \mid h_1, \ldots, h_s \in \mathbb{F}_q[x_1, \ldots, x_n] \right\}$$

In order to arrange the terms of a polynomial in a descending (or ascending) order, an ordering on its monomials must be defined. A monomial ordering can be defined as follows.

Definition 2.2.4. (Monomial Ordering [CLO92]). *Given the polynomial ring $\mathcal{R} := \mathbb{F}_q[x_1, \ldots, x_n]$, a monomial ordering is any relation $>$ on the set of monomials x^α, $\alpha \in \mathbb{Z}^n_{\geq 0}$ that satisfies the following properties:*

- *$>$ is a total ordering on $\mathbb{Z}^n_{\geq 0}$.*
- *If $\alpha > \beta$ and $\gamma \in \mathbb{Z}^n_{\geq 0}$, then $\alpha + \gamma > \beta + \gamma$.*

- $>$ *is a well ordering on* $\mathbb{Z}_{\geq 0}^n$*, i.e., every non-empty subset of* $\mathbb{Z}_{\geq 0}^n$ *has a smallest element under* $>$*.*

Following are the examples of two extensively used monomial orderings.

Definition 2.2.5. (Lexicographic Ordering [CLO92]). *If* $\alpha := (\alpha_1, \ldots, \alpha_n)$ *and* $\beta := (\beta_1, \ldots, \beta_n) \in \mathbb{Z}_{\geq 0}^n$*, then* $\alpha >_{lex} \beta$ *when the leftmost non-zero entry of the vector* $\alpha - \beta \in \mathbb{Z}^n$ *is positive.*

Definition 2.2.6. (Degree Reverse Lexicographic Ordering [CLO92]). *For* $\alpha = (\alpha_1, \ldots, \alpha_n)$ *and* $\beta = (\beta_1, \ldots, \beta_n) \in \mathbb{Z}_{\geq 0}^n$*,* $\alpha >_{degrevlex} \beta$ *if*

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i \text{ or } |\alpha| = |\beta|$$

and the rightmost nonzero entry of the vector $\alpha - \beta \in \mathbb{Z}^n$ *is negative.*

With respect to a monomial ordering, the leading monomial and leading term of a polynomial can be defined as follows.

Definition 2.2.7. (Leading Monomial, Leading Term [CLO92]). *For a non-zero polynomial* $f = \sum_\alpha a_\alpha x^\alpha \in \mathbb{F}_q[x_1, \ldots, x_n]$ *and a monomial ordering* $>$*, the leading coefficient and leading monomial of* f *are* $\mathrm{LC}(f) = a_{max(\alpha)}$ *and* $\mathrm{LM}(f) = x^{max(\alpha)}$ *respectively where the maximum is taken with respect to the monomial order* $>$*. The leading term of* f *is given by* $\mathrm{LT}(f) = \mathrm{LC}(f) \cdot \mathrm{LM}(f)$*.*

Given a polynomial $f \in \mathbb{F}_q[x_1, \ldots, x_n]$ and a set of polynomials $f_1, \ldots, f_k$ in $\mathbb{F}_q[x_1, \ldots, x_n]$, division of f by $f_1, \ldots, f_k$ means that f can be expressed as

$$f = a_1 f_1 + \cdots + a_k f_k + r \tag{2.1}$$

where the quotients $a_1, \ldots, a_k$ and the remainder r are elements of $\mathbb{F}_q[x_1, \ldots, x_n]$ and either $r = 0$ or is a linear combination of monomials such

that none of it is divisible by the leading terms of the divisors with respect to a given monomial ordering (Theorem 3 [CLO92]).

2.3 Tensors

Let $\phi_{\boldsymbol{\mathcal{T}}} : \mathcal{V}_1 \times \mathcal{V}_2 \times \cdots \times \mathcal{V}_n \to \mathcal{W}$ be a multilinear map, where $\mathcal{V}_1, \ldots, \mathcal{V}_n$ and $\mathcal{W}$ are finite-dimensional vector spaces. Given fixed bases for the vector spaces, $\phi_{\boldsymbol{\mathcal{T}}}$ can be represented by a multidimensional array $\boldsymbol{\mathcal{T}}$ called a tensor. The order of a tensor is the number of indices required to represent a component of the array.

Definition 2.3.1. (Slices). *Slices in a tensor are two-dimensional sections generated by fixing all indices except two. In a third order tensor, matrices generated by keeping the last index fixed are called frontal slices. Therefore a third order tensor* $\boldsymbol{\mathcal{T}}^{I_1 \times I_2 \times I_3}$ *is an* I_3 *array of* $I_1 \times I_2$ *matrices.*

Definition 2.3.2. (Bilinear Map). *A Bilinear map is a function* $\phi : \mathcal{V}_1 \times \mathcal{V}_2 \to \mathcal{V}_3$ *that takes two elements from two vector spaces* $\mathcal{V}_1$ *and* $\mathcal{V}_2$ *and maps it to an element of a third vector space* $\mathcal{V}_3$ *such that it is linear in each of its elements; i.e., for a fixed* $\boldsymbol{v}_1 \in \mathcal{V}_1$, $\boldsymbol{v}_1 \mapsto \phi(\boldsymbol{v}_1, \boldsymbol{v}_2)$ *is a linear function from* $\mathcal{V}_1$ *to* $\mathcal{V}_3$ *and for a fixed* $\boldsymbol{v}_2 \in \mathcal{V}_2$, $\boldsymbol{v}_2 \mapsto \phi(\boldsymbol{v}_1, \boldsymbol{v}_2)$ *is a linear function from* $\mathcal{V}_2$ *to* $\mathcal{V}_3$.

An order-3 tensor $\boldsymbol{\mathcal{T}}$ can be used to represent a bilinear map $\phi : \mathcal{V} \times \mathcal{V} \to \mathcal{V}$, on the vector space $\mathcal{V}$ over $\mathbb{F}$. If $dim(\mathcal{V}) = n$ and $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ denotes a basis for $\mathcal{V}$, then

$$\phi(\boldsymbol{b}_i, \boldsymbol{b}_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \boldsymbol{T}_{ijk} \cdot \boldsymbol{b}_k \tag{2.2}$$

For a fixed k, $\boldsymbol{T}_k := (\boldsymbol{T}_{ijk})_{1 \le i,j \le n}$ represents a unique matrix $\boldsymbol{T}_k$ of order

$n \times n$. For $1 \leq k \leq n$, $\boldsymbol{T}_k$ forms the frontal slices of the tensor $\boldsymbol{\mathcal{T}}$. The bilinear map ϕ then acts on two arbitrary vectors $\boldsymbol{v}_1 = (v_{1,1}, v_{2,1}, \ldots, v_{n,1}), \boldsymbol{v}_2 = (v_{1,2}, v_{2,2}, \ldots, v_{n,2}) \in \mathcal{V}$ as follows:

$$\begin{aligned} \phi(\boldsymbol{v}_1, \boldsymbol{v}_2) &= \sum_{i,j=1}^{n} v_{i,1}\phi(\boldsymbol{b}_i, \boldsymbol{b}_j)v_{j,2} \\ &= \begin{bmatrix} \boldsymbol{v}_1\boldsymbol{T}_1\boldsymbol{v}_2^T & \ldots & \boldsymbol{v}_1\boldsymbol{T}_n\boldsymbol{v}_2^T \end{bmatrix} \end{aligned} \tag{2.3}$$

Definition 2.3.3. (The n-mode Product). *The n-mode product defines multiplication of a tensor by a matrix. In general, the elementwise n-mode product of a tensor $\boldsymbol{\mathcal{T}} \in \mathbb{F}^{I_1 \times \cdots \times I_{n-1} \times I_n \times I_{n+1} \times \cdots \times I_N}$ and a matrix $\boldsymbol{M} \in \mathbb{F}^{J \times I_n}$ is defined as:*

$$(\boldsymbol{\mathcal{T}} \times \boldsymbol{M})_{i_1 i_2 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} \boldsymbol{\mathcal{T}}_{i_1 i_2 \cdots i_{n-1} i_n i_{n+1} \cdots i_N} \boldsymbol{M}_{j, i_n} \tag{2.4}$$

The resultant tensor is of the order of $(I_1 \times I_2 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N)$.

2.4 Lattices

Let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ be a set of n linearly independent vectors in $\mathbb{R}^n$ and $\mathbf{B} \in \mathbb{R}^{n \times n}$ be the matrix with $\mathbf{B}(i, :) = \boldsymbol{b}_i$ for $1 \leq i \leq n$. Then, the n-dimensional lattice generated by $\mathbf{B}$ can be defined as

$$\mathcal{L}(\mathbf{B}) = \{\boldsymbol{v} \cdot \mathbf{B} \mid \boldsymbol{v} \in \mathbb{Z}^n\} \tag{2.5}$$

Definition 2.4.1. (Lattice). *A lattice $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^n$ given by the set of all integer linear combinations of $k \leq n$ linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k \in \mathbb{R}^n$.*

The vectors $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k)$ is called a basis for $\mathcal{L}$ and k is called the dimension

of $\mathcal{L}$. If $k = n$, then $\mathcal{L}$ is called a full rank lattice.

Definition 2.4.2. (Minimum Distance). *The minimum distance of a lattice $\mathcal{L}$ is the length of the shortest non-zero lattice vector given by*

$$\lambda_1(\mathcal{L}) := \min_{\mathbf{0} \neq \boldsymbol{v} \in \mathcal{L}} \|\boldsymbol{v}\|$$

where $\|\boldsymbol{v}\|$ denotes the Euclidean norm or ℓ_2 norm of $\boldsymbol{v}$.

Definition 2.4.3. (Successive Minima). *Let $\mathcal{L}$ be a lattice of dimension n. The i^{th} successive minimum denoted by $\lambda_i(\mathcal{L})$ is the smallest radius r such that there exists i linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_i \in \mathcal{L}$ of norm at most r. i.e.,*

$$\|\boldsymbol{b}_1\|, \ldots, \|\boldsymbol{b}_i\| \leq \lambda_i(\mathcal{L}) \text{ for all } i$$

Lattices used in this thesis are q-ary lattices. A q-ary lattice can be defined as follows:

Definition 2.4.4. (q-ary Lattices). *For some $q \in \mathbb{N}$, a lattice $\mathcal{L}$ is called a q-ary lattice if it satisfies $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$. A q-ary lattice can be thought of as a subgroup of $\mathbb{Z}_q^n$. A vector $\boldsymbol{v}$ is in the lattice $\mathcal{L}$ if $\boldsymbol{v}$ mod $q \in \mathcal{L}$. Given a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times M}$, the following are two M dimensional q-ary lattices.*

$$\mathcal{L}_q(\boldsymbol{A}) = \left\{ \boldsymbol{v} \in \mathbb{Z}_q^M \mid \boldsymbol{v} = \boldsymbol{w} \cdot \boldsymbol{A} \ mod\ q \text{ for some } \boldsymbol{w} \in \mathbb{Z}_q^n \right\}$$
$$\mathcal{L}_q^{\perp}(\boldsymbol{A}) = \left\{ \boldsymbol{v} \in \mathbb{Z}_q^M \mid \boldsymbol{v} \cdot \boldsymbol{A}^T = \mathbf{0} \ mod\ q \right\}$$

The first lattice corresponds to the linear code generated by the rows of $\boldsymbol{A}$ and the second lattice corresponds to a linear code whose parity check matrix is given by $\boldsymbol{A}$. The dual of a lattice $\mathcal{L}$ is the set of vectors $\boldsymbol{v} \in span(\mathcal{L})$ such

that $\langle \boldsymbol{v}, \boldsymbol{w} \rangle \in \mathbb{Z}$ for all $\boldsymbol{w} \in \mathcal{L}$. By definition, the above two lattices are dual to each other [MR09].

2.4.1 Computational Problems based on Lattices

We now discuss some of the well-known computational problems based on lattices. The security of lattice-based cryptographic constructions depends on the hardness of these problems. One of the most extensively studied problem is the Shortest Vector Problem (SVP).

Definition 2.4.5. (Shortest Vector Problem (SVP)). *Given a basis* $\mathbf{B}$ *of a lattice* $\mathcal{L}$*, find the shortest non-zero vector in* $\mathcal{L}(\mathbf{B})$.

For cryptographic purposes, one generally considers the approximated versions of these problems specified by an approximation factor $\gamma = \gamma(n)$.

Definition 2.4.6. (Approximate Shortest Vector Problem ($\mathbf{SVP}_\gamma$)). *Given a basis* $\mathbf{B}$ *of an* n*-dimensional lattice* $\mathcal{L}$*, find a non-zero vector* $\boldsymbol{v} \in \mathcal{L}(\mathbf{B})$ *such that* $\|\boldsymbol{v}\| \leq \gamma(n) \cdot \lambda_1(\mathcal{L})$.

The decisional variant of the approximate shortest vector problem called GapSVP_γ can be defined as follows:

Definition 2.4.7. (Decisional Approximate Shortest Vector Problem ($\mathbf{GapSVP}_\gamma$)). *Given a basis* $\mathbf{B}$ *of an* n*-dimensional lattice* $\mathcal{L}$ *and a positive integer* d*, the* $GapSVP_\gamma$ *problem is to distinguish between the cases* $\lambda_1(\mathcal{L}(\mathbf{B})) < d$ *and* $\lambda_1(\mathcal{L}(\mathbf{B})) > \gamma(n) \cdot d$.

Another computational lattice problem which is a close variant of the shortest vector problem is the Shortest Independent Vectors Problem (SIVP) and its approximation variant can be defined as follows:

Definition 2.4.8. (Approximate Shortest Independent Vectors Problem (SIVP$_\gamma$)). *Given a basis* $\mathbf{B}$ *of an* n*-dimensional lattice* $\mathcal{L}$*, find a set of* n *linearly independent lattice vectors* $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n$ *such that* $\|\boldsymbol{s}_i\| \leq \gamma(n) \cdot \lambda_n(\mathcal{L})$ *for all* i*.*

These problems are known to be NP-hard for small approximation factors (for factors above $\sqrt{n/\log n}$, approximating lattice problems are no longer NP-hard [LLS90, GG00]). There exists either polynomial time algorithms [LLL82, Sch87] that achieve exponential approximation factors or algorithms that obtain polynomial approximation factors but require exponential time [Kan83, MV13]. The best known polynomial time algorithms are successors of the lattice reduction algorithm LLL [LLL82] and obtains exponential approximation factors like $\gamma = 2^{\mathcal{O}(n \log\log n/\log n)}$ [AKS01].

2.5 Statistical Distributions and Gaussian Measures

The encryption algorithms used in this work are probabilistic in nature. Therefore, distinguishing between the encryptions of two messages amounts to distinguishing between two probability distributions. In this context, it is important to define the following.

Definition 2.5.1 (Statistical Distance). *Let* X *and* Y *be two random variables taking values in a set* S*. Then, the statistical distance between* X *and* Y *is defined as*

$$\Delta(X, Y) = \frac{1}{2} \sum_{x \in S} |Pr[X = x] - Pr[Y = x]|$$

Two probability distributions can be considered sufficiently close if it is difficult to distinguish between their samples. This brings us to the question

of computational indistinguishability. But before that we need to define what we mean by a negligible function.

Definition 2.5.2 (**Negligible Function**). *A function* $negl(x) : \mathbb{N} \to \mathbb{R}$ *is called negligible if, for every* $c \in \mathbb{N}$, *there exists an integer* n_c *such that* $|negl(x)| < \frac{1}{x^c}$ *for all* $x > n_c$. *We write* $negl(\cdot)$ *to denote an arbitrary negligible function.*

Definition 2.5.3 (**Computational Indistinguishability**). *Two distribution ensembles* $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ *are said to be computationally indistinguishable if for all Probabilistic Polynomial Time (PPT) algorihms* $\mathcal{A}$, *there exists a negligible function, negl, such that*

$$Adv_{\mathcal{A}}^{X,Y}(\lambda) = \left| \Pr_{x\leftarrow X_\lambda}[\mathcal{A}(x) = 1] - \Pr_{x\leftarrow Y_\lambda}[\mathcal{A}(x) = 1] \right| \leq negl(\lambda)$$

where $Adv_{\mathcal{A}}^{X,Y}(\lambda)$ *is called the advantage of* $\mathcal{A}$.

2.5.1 Discrete Gaussians

Cryptographic protocols based on lattices use Gaussian-like probability distributions over lattices called *discrete Gaussians*. The following overview of Gaussian measures is adopted from the works of [Reg05],[GPV08].

The normal distribution with mean 0 and variance σ^2 is the distribution on $\mathbb{R}$ given by the density function $\frac{1}{\sigma\sqrt{2\pi}}exp(\frac{-x^2}{2\sigma^2})$. The sum of two normal variables with mean 0 and variances σ_1^2 and σ_2^2 is a normal variable with mean 0 and variance $\sigma_1^2 + \sigma_2^2$. For any $s > 0$, the Gaussian function on $\mathbb{R}^n$ centered at 0 is defined as:

$$\rho_s(\boldsymbol{x}) = exp(-\pi \|\boldsymbol{x}/s\|^2)$$

Definition 2.5.4 (**Discrete Gaussian**). *For a lattice $\mathcal{L}$ and any $s > 0$, the discrete Gaussian probability distribution over $\mathcal{L}$ can be defined as*

$$\forall \boldsymbol{x} \in \mathcal{L}, \quad D_{\mathcal{L},s} = \frac{\rho_s(\boldsymbol{x})}{\rho_s(\mathcal{L})}$$

Let $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ be the group of reals $[0, 1)$ with mod 1 addition. For $\alpha \in \mathbb{R}^+$, Ψ_α is defined to be the distribution on $\mathbb{T}$ obtained by sampling from a normal variable with mean 0 and standard deviation $\frac{\alpha}{\sqrt{2\pi}}$ and reducing the result modulo 1. For a probability distribution ϕ over $\mathbb{T}$ and an integer $q \in \mathbb{Z}^+$, its discretization $\bar{\phi}$ is the discrete probability distribution over $\mathbb{Z}_q$ obtained by sampling from ϕ, multiplying by q and rounding to the nearest integer modulo q.

Further, in lattice-based encryption algorithms, one often encounters distributions where, with high probability, the value taken by a random variable lies in a given range.

Definition 2.5.5. (**B-bounded distribution**). *A distribution $\mathcal{X}$ over the set of integers is said to be B-bounded (denoted as $|\mathcal{X}| \leq B$) if*

$$Pr\left[|x| > B \mid x \xleftarrow{\$} \mathcal{X}\right] = negl(\lambda) \tag{2.6}$$

2.6 Cryptographic Encryption Schemes

An encryption scheme can be of two types – symmetric key encryption and public key encryption. A symmetric key encryption is a type of encryption that uses a single key for both encryption and decryption. The parties communicating via symmetric key encryption must exchange the key beforehand. On the other hand, a public key encryption scheme uses separate keys for encryption and decryption. The sender encrypts a message using the receiver's

public encryption key which can be decrypted by the receiver using the corresponding *private* decryption key.

A public key (symmetric key) encryption scheme $\mathcal{PKE} = \{\mathsf{PK.Keygen}, \mathsf{PK.Enc}, \mathsf{PK.Dec}\}$ consists of three PPT algorithms ($\mathcal{SKE} = \{\mathsf{SK.KeyGen}, \mathsf{SK.Enc}, \mathsf{SK.Dec}\}$ for a symmetric key scheme) for key generation, encryption and decryption respectively.

- **Key Generation**(1^λ). The key generation algorithm, $\mathsf{PK.Keygen}$, takes the security parameter λ and outputs a public encryption key pk and a secret decryption key sk, i.e., $(pk, sk) \leftarrow \mathsf{PK.Keygen}(1^\lambda)$ (In case of a symmetric key scheme, the algorithm $\mathsf{SK.KeyGen}(1^\lambda)$ outputs a single secret key sk).

- **Encryption**(m, pk). The encryption algorithm takes a message m and the public key pk and outputs a ciphertext $\boldsymbol{c} \leftarrow \mathsf{PK.Enc}(m, pk)$ (The algorithm $\mathsf{SK.Enc}(m, sk)$ outputs an encryption of a message m using the key sk).

- **Decryption**$(\boldsymbol{c}, sk)$. The decryption algorithm outputs the message m after taking the ciphertext $\boldsymbol{c}$ and the secret key sk as inputs, i.e., $m \leftarrow \mathsf{PK.Dec}(\boldsymbol{c}, sk)$. (The algorithm $\mathsf{SK.Dec}(\boldsymbol{c}, sk)$ decrypts a ciphertext $\boldsymbol{c}$ to its corresponding message m using sk).

The scheme $\mathcal{PKE}$ is said to be correct if for any key pair $(pk, sk) \leftarrow \mathsf{PK.KeyGen}$ and any message m

$$Pr[\mathsf{PK.Dec}(\mathsf{PK.Enc}(m, pk), sk) = m] = 1 - negl(\lambda)$$

(The correctness of $\mathcal{SKE}$ can be similarly defined).

2.7 Security

The security of public key encryption schemes can be classified on the basis of various possible goals and attack models. Two standard goals for a public key encryption scheme are semantic security and ciphertext indistinguishability. The notion of semantic security was proposed in [GM82]. A cryptosystem is said to be semantically secure if only negligible information about a plaintext can be extracted from its ciphertext which can also be computed without the ciphertext. In case of ciphertext indistinguishability, given the encryption of one of two messages, an adversary cannot distinguish between the two messages. Security in terms of indistinguishability can be commonly classified in terms of the following notions.

1. *Indistinguishability under Chosen Plaintext Attack (IND-CPA)*: An encryption scheme is said to be IND-CPA secure if an adversary, given polynomially bounded number of encryptions of its choice, cannot distinguish between the encryptions of two messages with probability greater than $\frac{1}{2} + negl(\lambda)$.

2. *Indistinguishability under Chosen Ciphertext Attack (IND-CCA1)*: An encryption scheme is said to be indistinguishable under chosen ciphertext attack if an adversary, given access to a decryption oracle (in addition to the public key) which decrypts ciphertexts of its choice, cannot distinguish between the encryptions of two messages with probability greater than $\frac{1}{2} + negl(\lambda)$. The adversary can query the decryption oracle until a challenge ciphertext is received.

3. *Indistinguishability under Adaptive Chosen Ciphertext Attack (IND-CCA2)*: Indistinguishability under adaptive chosen ciphertext attack

can be defined similarly to IND-CCA1 except that in addition to its abilities under IND-CCA1, the adversary can now query the decryption oracle even after a challenge ciphertext is received. However, it cannot ask for the decryption of the challenge ciphertext.

In this thesis, similar to other lattice-based cryptographic algorithms, we primarily deal with IND-CPA security. The IND-CPA security of a public key encryption scheme can be formally defined in terms of the game shown in Figure 2.1. This game can be described in terms of the following steps.

- The Challenger generates a key pair $(pk, sk) \leftarrow \mathsf{PK.KeyGen}(1^\lambda)$ and outputs pk.
- A PPT adversary $\mathcal{A}$ selects polynomially many messages and receives their encryptions (using the public key pk).
- $\mathcal{A}$ then outputs two equal length messages (m_0, m_1).
- The Challenger selects a bit $\beta \in \{0, 1\}$ uniformly at random and outputs the encryption of m_β given by $\mathsf{PK.Enc}(m_\beta, pk)$.
- The adversary outputs a guess for the value of β.

The adversary $\mathcal{A}$ wins the game if it can guess the value of β with a non-negligible advantage. The advantage of $\mathcal{A}$ is defined as

$$Adv_{\mathcal{A}}^{\text{IND-CPA}}(\lambda) := \left| Pr[\ \text{IND-CPA}_{\mathsf{PK}}^{\mathcal{A}}(\lambda) = 1] - \frac{1}{2} \right|$$

The IND-CPA security of a symmetric encryption scheme can be similarly defined except that in a public key encryption scheme, the adversary can encrypt messages itself using the public encryption key but in case of a symmetric key scheme it has no means to see the ciphertexts. Therefore

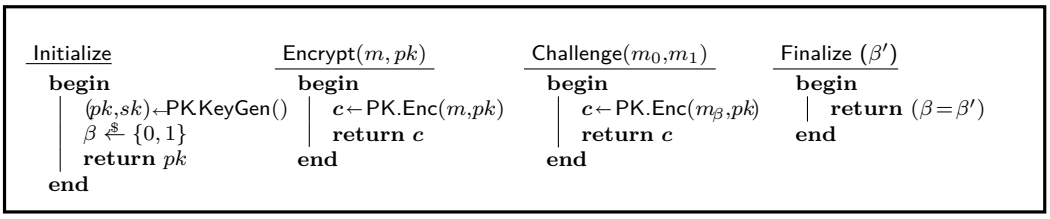

Fig 2.1: IND-CPA Game of $\mathcal{PKE}$

the adversary is provided with an encryption oracle in this case. A detailed analysis on the security of a symmetric key encryption scheme can be found in [BDJR97].

2.8 Homomorphic Encryption

A homomorphic encryption scheme is a scheme which enables the user to evaluate encryptions of functions on a set of plaintexts from their respective ciphertexts without explicit decryption. Given a function ϕ which acts on a set of k plaintexts, an encryption scheme is said to be homomorphic with respect to ϕ if $\mathsf{Enc}(\phi(m_1, m_2, \ldots, m_k))$ can be efficiently calculated from the encryptions of $m_1, m_2, \ldots, m_k$ without decryption. We now formally define homomorphic encryption and its related terminology. The following definitions are adopted from [Gen09, BGV14, BV14a].

A public key homomorphic encryption scheme consists of the following four PPT algorithms:

- **Key Generation**(1^λ). Takes the security parameter λ and generates the public encryption key pk, the secret decryption key sk and the public evaluation key evk. i.e., $(sk, pk, evk) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$.

- **Encryption**(m, pk). Takes the public key pk and a message $m \in \{0, 1\}$ and outputs the corresponding ciphertext $\boldsymbol{c} \leftarrow \mathsf{HE.Enc}(m, pk)$.

- **Homomorphic Evaluation**$(evk, \phi, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_t)$. Takes the public evaluation key evk, a function $\phi : \{0,1\}^t \rightarrow \{0,1\}$ and ciphertexts $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t$ and outputs a new ciphertext $\boldsymbol{c}_{eval} \leftarrow \mathsf{HE.Eval}(evk, \phi, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_t)$.

  The function ϕ represents an arithmetic circuit over $GF(2)$ with addition and multiplication gates.

- **Decryption**$(\boldsymbol{c}, sk)$. Takes the secret key sk and decrypts a ciphertext $\boldsymbol{c}$ to its corresponding message $m \leftarrow \mathsf{HE.Dec}(\boldsymbol{c}, sk)$.

(In a symmetric key homomorphic encryption scheme the key generation function generates the secret key which is used for both encryption and decryption along with the public evaluation key).

The IND-CPA security of a (public key) homomorphic encryption scheme is similar to that of any other encryption scheme except that the adversary has access to both the public encryption key and the public evaluation key. (In case of a symmetric key homomorphic scheme, the adversary has access to the public evaluation key).

Definition 2.8.1. (IND-CPA Security). *A public key homomorphic encryption scheme is said to be secure if an adversary $\mathcal{A}$, having access to public encryption key pk, public evaluation key evk and an encryption of a message m_β given by $\mathsf{HE.Enc}(m_\beta, pk)$ for a $\beta \in \{0,1\}$ chosen uniformly at random, cannot guess the value of β with probability more than $\frac{1}{2} + negl(\lambda)$.*

The correctness of a homomorphic encryption scheme relies on the correct decryption of an evaluated ciphertext and can be defined as follows.

Definition 2.8.2. (Correctnes). *Let Φ be a set of functions and let $(pk, sk, evk) \leftarrow \mathsf{HE.KeyGen}(1^\lambda)$, $\boldsymbol{c}_i \leftarrow \mathsf{HE.Enc}(m_i, pk)$ for all i and $\boldsymbol{c}_{eval} \leftarrow \mathsf{HE.Eval}(evk, \phi, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_t)$, where $\phi \in \Phi$. A homomorphic encryption*

scheme correctly evaluates the set of functions Φ if for all $\phi \in \Phi$ and for all $m_1, \ldots, m_t \in \{0,1\}$, it holds that

$$Pr\left[\mathsf{HE.Dec}(\boldsymbol{c}_{eval}, sk) = \phi(m_1, \ldots, m_t)\right] = 1 - negl(\lambda)$$

If the above equation holds for any depth L arithmetic circuit over $GF(2)$ and for all $m_1, \ldots, m_t \in \{0,1\}$, then such a scheme is said to be L-homomorphic.

Definition 2.8.3. (Compactness). *A homomorphic encryption scheme is said to be compact if there exists a fixed polynomial bound $\mathrm{b} = \mathrm{b}(\lambda)$ such that the size of the ciphertext output from HE.Eval is at most b bits, independent of the function ϕ.*

A fully homomorphic encryption scheme can be defined as follows.

Definition 2.8.4. (Fully Homomorphic Encryption). *If Φ denotes the set of all efficiently computable functions, then an encryption scheme is called fully homomorphic if it is compact and homomorphic for the set of functions Φ.*

In this thesis, we focus on the construction of a *leveled* fully homomorphic encryption scheme. A leveled FHE scheme can be defined as follows.

Definition 2.8.5. (Leveled Fully Homomorphic Encryption). *A homomorphic encryption scheme is called leveled fully homomorphic if it takes an additional input $L \in \mathbb{N}$ and compactly evaluates all functions of depth at most L such that the length of the ciphertexts is bounded by $\mathrm{b}(\lambda)$, independent of L.*

A leveled FHE scheme should not be confused with an L-homomorphic scheme. An L-homomorphic scheme is a somewhat homomorphic scheme

that *correctly* evaluates any depth L circuit whereas a leveled FHE scheme takes L as an additional input and *compactly* evaluates all circuits of depth at most L.

Gentry's bootstrapping theorem shows how to convert an L-homomorphic scheme to a fully homomorphic one. Before stating the theorem, we define a bootstrappable encryption scheme.

Definition 2.8.6. (Bootstrappable Scheme). *An L-homomorphic scheme is said to be bootstrappable if the depth of its decryption circuit is less than L. The computational complexity of its algorithms is polynomial in the security parameter λ.*

An encryption scheme is said to be circular secure if it is secure against an adversary that has access to the encryptions of the bits of the secret key. We now state Gentry's bootstrapping theorem.

Theorem 2.8.1. (Bootstrapping [Gen09]). *If there exists a bootstrappable L-homomorphic scheme, then there exists a leveled fully homomorphic encryption scheme.*

Further, if the scheme is circular secure, then there exists a fully homomorphic encryption scheme.

2.9 Learning with Errors

Learning with Errors is the problem of solving a system of noisy linear equations over $\mathbb{Z}_q$. It can be defined as follows.

Definition 2.9.1. (Learning With Errors). *Let $\mathcal{X}$ be a probability distribution on $\mathbb{Z}$ and $\boldsymbol{s}$ be a secret vector chosen uniformly at random from $\mathbb{Z}_q^n$ for some $n, q \in \mathbb{N}$. Let $\mathcal{A}_{\boldsymbol{s},\mathcal{X}}$ be the distribution that generates a pair*

$(\boldsymbol{a}, b = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ *obtained by choosing a vector* $\boldsymbol{a} \xleftarrow{\$} \mathbb{Z}_q^n$ *and an error* $e \xleftarrow{\$} \mathcal{X}$.

Given polynomially many samples from $\mathcal{A}_{\boldsymbol{s},\mathcal{X}}$, *the learning with errors problem denoted by* $\mathrm{LWE}_{n,q,\mathcal{X}}$ *is to output the vector* $\boldsymbol{s} \in \mathbb{Z}_q^n$ *with overwhelming probability.*

The decisional variant of the problem denoted by $\mathrm{DLWE}_{n,q,\mathcal{X}}$ *is to distinguish the distribution* $\mathcal{A}_{\boldsymbol{s},\mathcal{X}}$ *from the uniform distribution over* $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

The decisional LWE problem has been shown to be at least as hard as the LWE search problem [Reg09].

In terms of the game playing framework adopted from [BDJR97], the LWE problem (search and decision) can be defined in terms of the games shown in Figure 2.2 [AFFP11] and Figure 2.3. The advantage of a Probabilistic Polynomial Time (PPT) adversary $\mathcal{A}$ in solving the $\mathrm{LWE}_{n,q,\mathcal{X}}$ problem is given by

$$Adv^{\mathrm{LWE}}_{\mathcal{A},n,q,\mathcal{X}}(\lambda) := Pr\left[\mathrm{LWE}^{\mathcal{A}}_{n,q,\mathcal{X}}(\lambda) = 1\right] \tag{2.7}$$

A PPT adversary $\mathcal{A}$ solves the $\mathrm{DLWE}_{n,q,\mathcal{X}}$ problem with an advantage

$$Adv^{\mathrm{DLWE}}_{\mathcal{A},n,q,\mathcal{X}}(\lambda) := \left| Pr\left[\mathrm{DLWE}^{\mathcal{A}}_{n,q,\mathcal{X}}(\lambda) = 1\right] - \frac{1}{2} \right| \tag{2.8}$$

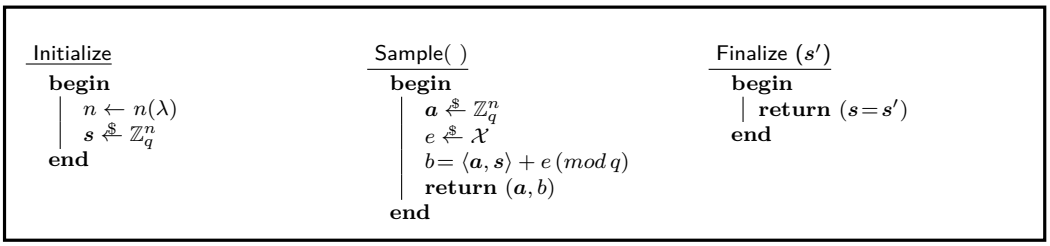

Fig 2.2: $\mathrm{LWE}_{n,q,\mathcal{X}}$ Game

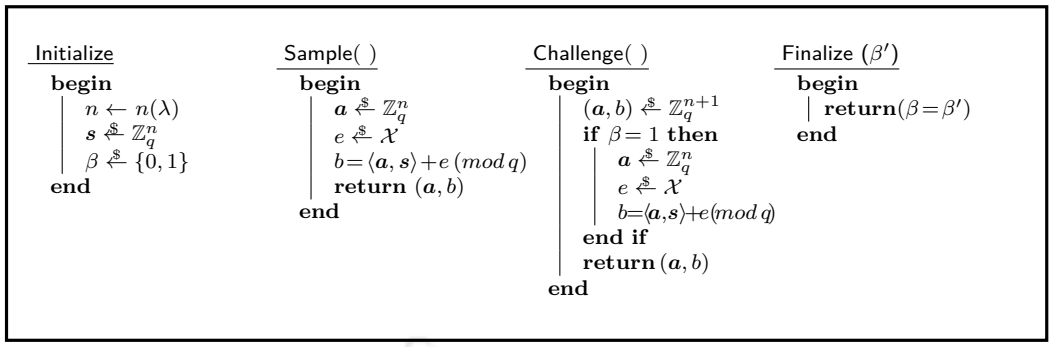

Fig 2.3: $\text{DLWE}_{n,q,\mathcal{X}}$ Game

2.9.1 Hardness of LWE

Given that $\mathcal{X}$ in LWE is a discretized Gaussian distribution with standard deviation $\sigma \geq 2\sqrt{n}$, indistinguishable from a B-bounded distribution, there exists a quantum reduction of LWE to approximating the decisional Shortest Vector Problem (GapSVP_γ) with approximation factor $\gamma = (q/B) \cdot \tilde{\mathcal{O}}(n)$ [Reg05, Reg09]. Further, [Pei09] and [BLP+13] gave classical reductions of LWE from worst-case GapSVP_γ for an exponential modulus and a polynomial modulus respectively. The best known algorithms for GapSVP_γ require $2^{\tilde{\Omega}(n/\log\gamma)}$ time [Sch87, MV13].

The hardness of LWE can be summarized in terms of the following theorem derived from the works of [Reg09, Pei09, BLP+13]. We state the results in terms of the bound B as stated in [Bra12, GSW13].

Theorem 2.9.1. ([Reg09, Pei09, BLP+13]). *Let $q = q(n)$ be prime and let $B \geq \omega(\log n) \cdot \sqrt{n}$. Then, there exists an efficiently sampleable B-bounded distribution $\mathcal{X}$ such that if there is an efficient algorithm that solves the average-case $DLWE_{n,q,\mathcal{X}}$ problem, then*

- *There exists an efficient quantum algorithm for solving $GapSVP_{\tilde{\mathcal{O}}(n \cdot q/B)}$ on any n-dimensional lattice.*

- *If $q \geq 2^{n/2}$, then there exists an efficient classical algorithm that solves the $GapSVP_{\tilde{\mathcal{O}}(n \cdot q/B)}$ problem on any n-dimensional lattice.*
- *There exists an efficient classical algorithm for solving a worst-case lattice problem (e.g., GapSVP) on any lattice of dimension $\sqrt{n}$.*

2.10 Summary

In this chapter, we have stated some basic concepts from Abstract Algebra and Linear Algebra. We have discussed some well-known computational problems based on lattices including the Learning with Errors problem and its hardness. Further, definitions and terminologies related to homomorphic encryption have also been discussed.

Chapter 3

Fully Homomorphic Encryption based on Multivariate Polynomial Evaluations

In this chapter, we propose a multi-bit leveled fully homomorphic encryption scheme based on multivariate polynomial evaluations. In the proposed scheme, multiple plaintext bits are encrypted in a single ciphertext. Homomorphic operations can be performed in an SIMD (Single Instruction Multiple Data) style which means homomorphic addition and multiplication can be performed simultaneously on multiple plaintext bits. Some of the schemes that explore this property are [SV14, BGH13, PVW08, GHS12b]. We introduce the scheme as a symmetric key scheme and then convert it into a public key scheme. The security of the scheme depends on the hardness of the LWE problem.

The proposed scheme is based on the evaluation of multivariate polynomials of a secret ideal $\mathcal{I}$. A ciphertext is obtained by evaluating a random polynomial in $\mathcal{I}$ on a set of distinct points and adding scaled plaintext bits to a number of these evaluations corrupted with noise. Multiplication in the scheme is performed by evaluating a bilinear map on the ciphertexts. This

map is represented by a 3-way tensor and is given as the public evaluation key for multiplication. The aim here is to use the multiplicative property of the polynomial ring to perform homomorphic multiplication. Unlike other LWE-based schemes, homomorphic multiplication does not increase the size of the ciphertexts. Therefore, there is no need for relinearization. The noise associated with the ciphertexts increases only linearly with each homomorphic operation. Hence, a leveled FHE scheme can be obtained without modulus switching. The per gate computation of the scheme is $\mathcal{O}(n^3 \cdot L^2)$ where L denotes the multiplicative depth of the circuit.

3.1 The Proposed Scheme

In this section, we discuss the construction of the leveled FHE scheme. For simplicity, we present a private key variant of this scheme. Subsequently, we explain its conversion to a public key scheme.

Let $\mathcal{R}$ be the polynomial ring $\mathcal{R} := \mathbb{Z}_q[x_1, \ldots, x_v]/\langle x_1^q - x_1, \ldots, x_v^q - x_v\rangle$ where $q = q(\lambda)$ is prime. Let us consider an ideal $\mathcal{I}$ of the ring $\mathcal{R}$. For some $r < q \in \mathbb{N}$, $\mathcal{R}_{\leq r}$ is a vector space of dimension $dim(\mathcal{V}) := \binom{v+r}{r}$ over $\mathbb{Z}_q$. Let $\mathcal{I}_{\leq r}$ denote the set of polynomials in $\mathcal{I}$ with degree $\leq r$. Then, $\mathcal{I}_{\leq r}$ is a subspace of $\mathcal{R}_{\leq r}$. Let n be the dimension of $\mathcal{I}_{\leq r}$. It is very easy to see that a polynomial $f \in \mathcal{R}_{\leq r}$ evaluated at all points of $\mathbb{Z}_q^v$ generates a vector in $\mathbb{Z}_q^{q^v}$. The set of such vectors obtained by evaluating all polynomials in $\mathcal{R}_{\leq r}$ constitutes a $dim(\mathcal{V})$-dimensional subspace of $\mathbb{Z}_q^{q^v}$. Similarly, evaluating polynomials in $\mathcal{I}_{\leq r}$ gives us an n-dimensional subspace of $\mathbb{Z}_q^{q^v}$. We therefore have the following lemma

Lemma 3.1.1. *Let $\mathcal{R} := \mathbb{Z}_q[x_1, \ldots, x_v]/\langle x_1^q - x_1, \ldots, x_v^q - x_v\rangle$. For $n, \ell \in \mathbb{N}$ with $n < \ell$, we can find ℓ distinct points $\{\boldsymbol{z}_i \in \mathbb{Z}_q^v\}_{1 \leq i \leq \ell}$ such that evaluating*

polynomials in $\mathcal{I}_{\leq r} \subseteq \mathcal{R}_{\leq r}$ *with* $dim(\mathcal{I}_{\leq r}) := n$ *at* $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell)$ *spans an* n*-dimensional subspace of the vector space* $\mathbb{Z}_q^\ell$.

We choose a set of ℓ evaluation points $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell\} \in \mathbb{Z}_q^v$ which satisfy the following conditions:

1. Every vector in $\mathbb{Z}_q^\ell$ can be got by evaluating a polynomial in $\mathcal{R}_{\leq r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell)$.
2. Every vector in $\mathbb{Z}_q^n$ can be got by evaluating a polynomial in $\mathcal{I}_{\leq r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)$.

This ensures that the vector space $\mathcal{S}_{\mathcal{I}_{\leq r}}$ is the row span of a matrix $\boldsymbol{B}$ which is of the form $[\boldsymbol{B}_1|\boldsymbol{B}_2]$ where $\boldsymbol{B}_1 \in \mathbb{Z}_q^{n \times n}$ has rank n.

Therefore, we can find a basis $\{\tilde{\boldsymbol{s}}_{n+1}, \tilde{\boldsymbol{s}}_{n+2}, \ldots, \tilde{\boldsymbol{s}}_\ell\} \in \mathbb{Z}_q^\ell$ for $(\mathcal{S}_{\mathcal{I}_{\leq r}})^\perp$ which has the following form:

$$\left.\begin{aligned} \tilde{\boldsymbol{s}}_{n+1} &= \begin{bmatrix} s_{n+1}^1 & \ldots & s_{n+1}^n & 1 & 0 & \ldots & 0 \end{bmatrix} \\ \tilde{\boldsymbol{s}}_{n+2} &= \begin{bmatrix} s_{n+2}^1 & \ldots & s_{n+2}^n & 0 & 1 & \ldots & 0 \end{bmatrix} \\ &\vdots \\ \tilde{\boldsymbol{s}}_{\ell} &= \begin{bmatrix} s_{\ell}^1 & \ldots & s_{\ell}^n & 0 & 0 & \ldots & 1 \end{bmatrix} \end{aligned}\right\} \tag{3.1}$$

Thus, $\mathcal{S}_{\mathcal{I}_{\leq r}}$ is the null space of the matrix $\begin{bmatrix} \boldsymbol{S} & \boldsymbol{I}_{\ell-n} \end{bmatrix}^T$ where $\boldsymbol{S}(i,j) = s_{n+i}^j$ for $1 \leq i \leq \ell - n$ and $1 \leq j \leq n$. In this work, we assume that the choice of $\mathcal{I}$ and the points $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_\ell$ is such that the rows of the matrix $\boldsymbol{S}$ are linearly independent.

3.1.1 The Basic Encryption Scheme

We describe the basic encryption scheme in terms of the following algorithms. (The homomorphic operations are described separately in Section 3.1.3.) The

plaintext space is $\{0,1\}^{\ell-n}$ and the ciphertexts are vectors in $\mathbb{Z}_q^\ell$. Further, we restrict ourselves to the case where $\mathcal{I}$ is a principal ideal.

- $\mathsf{Setup}(1^\lambda, 1^L)$: The Setup algorithm takes as input the security parameter λ and a parameter L and outputs $n = n(\lambda, L)$, $\ell = \ell(\lambda, L)$, modulus $q = q(\lambda, L)$ and noise distribution $\mathcal{X} = \mathcal{X}(\lambda, L)$ such that $|\mathcal{X}| \leq B$. Here L denotes the depth of the circuit that can be homomorphically evaluated. Let $\pi = (n, \ell, q, \mathcal{X})$.

- $\mathsf{KeyGen}(\pi)$: Choose two positive integers v and r' such that $n = \binom{v+r'}{r'}$. Choose integers r and r_g such that $r - r_g = r'$. Choose ℓ points, $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell$, from $\mathbb{Z}_q^v$ such that the aforementioned conditions are satisfied. Choose a random polynomial $g(x_1, \ldots, x_v)$ with degree r_g in v variables such that $g(\boldsymbol{z}_1), \ldots, g(\boldsymbol{z}_\ell)$ are all non zero. This polynomial acts as the generator of the ideal $\mathcal{I}$. Generate a basis $\mathcal{B}_{\mathcal{I}_{\leq r}} = (gh_1, \ldots, gh_n)$ for the subspace $\mathcal{I}_{\leq r}$ by considering linearly independent polynomials $h_1, \ldots, h_n$ having degree less than or equal to r'. A basis for $\mathcal{S}_{\mathcal{I}_{\leq r}}$ is obtained by evaluating the polynomials in $\mathcal{B}_{\mathcal{I}_{\leq r}}$ at the points $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell)$. Construct a basis of $(\mathcal{S}_{\mathcal{I}_{\leq r}})^\perp$ (as given in Equation 3.1) which can be written in terms of the matrix $\begin{bmatrix} \boldsymbol{S} & \boldsymbol{I}_{\ell-n} \end{bmatrix}$ where

$$\boldsymbol{S} = \begin{bmatrix} s_{n+1}^1 & s_{n+1}^2 & \cdots & s_{n+1}^n \\ s_{n+2}^1 & s_{n+2}^2 & \cdots & s_{n+2}^n \\ \vdots & \vdots & \ddots & \vdots \\ s_\ell^1 & s_\ell^2 & \ldots & s_\ell^n \end{bmatrix} \tag{3.2}$$

Choose a matrix $\boldsymbol{R} \in \mathbb{Z}_q^{\ell \times \ell}$ such that $\boldsymbol{R}$ is of the form

$$\boldsymbol{R} := \left[\begin{array}{c|c} \boldsymbol{R}_1^{n \times n} & 0^{n \times (\ell-n)} \\ \hline \boldsymbol{R}_2^{(\ell-n) \times n} & \boldsymbol{I}_{\ell-n} \end{array}\right] \tag{3.3}$$

where $\boldsymbol{R}_1 \in \mathbb{Z}_q^{n\times n}$ is a randomly chosen full rank matrix and the entries of $\boldsymbol{R}_2$ are chosen uniformly at random from $\mathbb{Z}_q$. $\boldsymbol{I}_{\ell-n}$ denotes the identity matrix of size $\ell - n$. The secret key is $sk = (\boldsymbol{S}, \boldsymbol{R}_1, \boldsymbol{R}_2)$. The public parameters are n, ℓ and q.

- Encrypt$(\pi, sk, \boldsymbol{m})$: To encrypt a message $\boldsymbol{m} \in \{0,1\}^{\ell-n}$, sample a vector $\boldsymbol{y}$ uniformly at random from $\mathbb{Z}_q^n$ and a vector $\boldsymbol{e} = (\boldsymbol{0}, e_{n+1}, \ldots, e_\ell) \in \mathbb{Z}_q^\ell$, where $\boldsymbol{0}$ denotes the zero vector of order n and each e_j is chosen independently from the distribution $\mathcal{X}$ for $n+1 \leq j \leq \ell$. Let $\boldsymbol{p}$ be the vector given by $\boldsymbol{p} = (\boldsymbol{0}, \boldsymbol{m}) = (\boldsymbol{0}, m_{n+1}, \ldots, m_\ell) \in \mathbb{Z}_q^\ell$. If $\boldsymbol{S}_{\text{enc}} := \begin{bmatrix} \boldsymbol{I}_n & -\boldsymbol{S}^T \end{bmatrix} \in \mathbb{Z}_q^{n\times\ell}$, then the ciphertext can be computed as:

$$\boldsymbol{c} = \left(\boldsymbol{p} \cdot \left\lfloor \tfrac{q}{2} \right\rfloor + \boldsymbol{y} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}\right) \boldsymbol{R} \; mod \; q \in \mathbb{Z}_q^\ell \tag{3.4}$$

Note that, $\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}$ essentially represents the evaluation of a polynomial $f \in \mathcal{I}_{\leq r}$ at the points $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell)$. This is because $f(\boldsymbol{z}_1), \ldots, f(\boldsymbol{z}_n)$ can take any value $\boldsymbol{y} \in \mathbb{Z}_q^n$ and $f(\boldsymbol{z}_j)$ for $n+1 \leq j \leq \ell$ can be expressed as the following linear combination of $f(\boldsymbol{z}_1), \ldots, f(\boldsymbol{z}_n)$.

$$f(\boldsymbol{z}_j) = -\sum_{i=1}^{n} s_j^i \cdot f(\boldsymbol{z}_i) \; (mod \; q) \tag{3.5}$$

The above equation is a consequence of the fact that $(f(\boldsymbol{z}_1), \ldots, f(\boldsymbol{z}_\ell)) \in \mathcal{S}_{\mathcal{I}_{\leq r}}$ and $Ker\left([\boldsymbol{S} \; \boldsymbol{I}_{\ell-n}]^T\right) = \mathcal{S}_{\mathcal{I}_{\leq r}}$.

- Decrypt$(\pi, sk, \boldsymbol{c})$: Let $\boldsymbol{S}_{\text{dec}} = \boldsymbol{R}^{-1}\begin{bmatrix} \boldsymbol{S} & \boldsymbol{I}_{\ell-n} \end{bmatrix}^T \in \mathbb{Z}_q^{\ell\times(\ell-n)}$. Given the ciphertext $\boldsymbol{c}$ and the secret key sk, $\boldsymbol{m}$ can be recovered as

$$\boldsymbol{m} = \left\lfloor \frac{1}{\lfloor q/2 \rfloor} \left(\boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}} \, mod \, q\right) \right\rceil \; mod \; 2 \tag{3.6}$$

Correctness of Decryption.

For correct decryption, the noise in the decryption process must be small. The decryption process involves computing the inner product of the ciphertext with each column of the matrix $\boldsymbol{S}_{\text{dec}}$ and reducing it modulo q. For each of these products, the decryption function outputs 0 for the corresponding entry if the magnitude of the inner product is $< q/4$ and 1 otherwise. In the following lemma, we analyze the magnitude of the noise in decryption.

Lemma 3.1.2. *Let $q, n, \ell, |\mathcal{X}| \leq B$ be as described in the scheme. Let $\boldsymbol{c} = \left(\boldsymbol{p} \cdot \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}\right) \boldsymbol{R}$ mod q be the encryption of a message $\boldsymbol{m} \in \{0,1\}^{\ell-n}$ under the key $sk = (\boldsymbol{S}, \boldsymbol{R}_1, \boldsymbol{R}_2)$. Then, for some $\boldsymbol{e} = (\boldsymbol{0}, \tilde{\boldsymbol{e}}) \in (\boldsymbol{0}, \mathcal{X}^{\ell-n})$ with $\|\boldsymbol{e}\|_\infty \leq B$, it holds that $\boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}} = \boldsymbol{m} \cdot \left\lfloor \frac{q}{2} \right\rfloor + \tilde{\boldsymbol{e}}$ $(mod\ q)$. Further, if $B < \lfloor q/2 \rfloor /2$, then $\boldsymbol{m} \leftarrow \mathsf{Decrypt}(sk, \boldsymbol{c})$.*

Proof. If $\tilde{\boldsymbol{e}} = (e_{n+1}, \ldots, e_\ell) \in \mathbb{Z}_q^{\ell-n}$ where e_j denotes the j^{th} non-zero entry of $\boldsymbol{e}$ for $n+1 \leq j \leq \ell$, then

$$\begin{aligned} \boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}} &= \left(\boldsymbol{p} \cdot \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}\right) \boldsymbol{R} \cdot \boldsymbol{S}_{\text{dec}} \ (mod\ q) \\ &= \boldsymbol{m} \cdot \left\lfloor \frac{q}{2} \right\rfloor + \tilde{\boldsymbol{e}} \ (mod\ q) \end{aligned} \tag{3.7}$$

If $B < \lfloor q/2 \rfloor /2$, then for $m_j = 0$ for any $n+1 \leq j \leq \ell$, $\boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}}(j) = e_j \ (mod\ q)$ where $|e|_j \leq B < q/4$. Hence, the decryption function outputs 0. Similarly, for $m_j = 1$, $\boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}}(j) = \left\lfloor \frac{q}{2} \right\rfloor + e_j \ (mod\ q)$ whose magnitude is $> q/4$ and hence, the decryption function outputs 1. Therefore, if $\|\boldsymbol{e}\|_\infty \leq B$ and $B < \lfloor q/2 \rfloor /2$, then $\boldsymbol{m} \leftarrow \mathsf{Decrypt}(sk, \boldsymbol{c})$. □

3.1.2 Security

In this section, we analyze the IND-CPA security of the basic encryption scheme based on the LWE problem. The security of this scheme follows from Lemma 6.2 of [PW11]. The lemma is restated as follows

Lemma 3.1.3. *Let $h, \ell = poly(n)$. Choose $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{h\times n}$, $\hat{\boldsymbol{S}} \leftarrow \mathbb{Z}_q^{(\ell-n)\times n}$ uniformly at random and $\boldsymbol{E} \leftarrow \mathcal{X}^{h\times(\ell-n)}$. If $\boldsymbol{B} = \boldsymbol{A}\hat{\boldsymbol{S}}^T + \boldsymbol{E}$, then $(\boldsymbol{A}, \boldsymbol{B})$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{h\times\ell}$ under the assumption that $LWE_{n,q,\mathcal{X}}$ is hard.*

To prove the security of the proposed scheme we consider $h = 1$.

Lemma 3.1.4. *Under the LWE assumption, given two distinct message vectors $\boldsymbol{m}_1, \boldsymbol{m}_2 \in \{0,1\}^{\ell-n}$, if $\ell - n$ is $\mathcal{O}(1)$ there exists no efficient algorithm that can distinguish between the distributions of the encryptions of $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$. Moreover, there exists no efficient algorithm that can distinguish the uniform distribution on the set of encryptions of any given message from the uniform distribution on $\mathbb{Z}_q^\ell$*

Proof. Note that the matrix $\boldsymbol{R}$ in the encryption process of the proposed scheme is the product of the following two matrices

$$\boldsymbol{R}' = \begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \text{ and } \boldsymbol{R}'' = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{R}_2 & \boldsymbol{I} \end{bmatrix} \tag{3.8}$$

Therefore, the encryption of a message $\boldsymbol{m}$ is given as

$$\boldsymbol{c} = \left(\boldsymbol{p} \cdot \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}\right) \boldsymbol{R}'\boldsymbol{R}'' \tag{3.9}$$

where $\boldsymbol{p} = (\boldsymbol{0}, \boldsymbol{m})$ and $\boldsymbol{y}$ is randomly chosen from a uniform distribution in $\mathbb{Z}_q^n$. Now, if we substitute $\boldsymbol{y}\boldsymbol{R}_1 = \boldsymbol{v}$, we get $\boldsymbol{c}' = (\boldsymbol{p}\lfloor\frac{q}{2}\rfloor + \boldsymbol{y} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e})\boldsymbol{R}' =$

$(\boldsymbol{p}\lfloor\frac{q}{2}\rfloor+(\boldsymbol{v},-\boldsymbol{v}\cdot\boldsymbol{R}_1^{-1}\boldsymbol{S}^T)+\boldsymbol{e})$. Here, the distribution on $\boldsymbol{v}$ is uniform in $\mathbb{Z}_q^n$ and the distribution on $\boldsymbol{R}_1^{-1}\boldsymbol{S}^T$ is uniform in the set of full column rank matrices in $\mathbb{Z}_q^{n\times(\ell-n)}$. Observe that the distribution on $((\boldsymbol{v},-\boldsymbol{v}\cdot\boldsymbol{R}_1^{-1}\boldsymbol{S}^T)+\boldsymbol{e})$ is similar to the one considered in Lemma 3.1.3 (for the case $h=1$) except that there the distribution on $\hat{\boldsymbol{S}}$ is random in $\mathbb{Z}_q^{(\ell-n)\times n}$ (while here the distribution of $\boldsymbol{S}\boldsymbol{R}_1^{-T}$ is restricted to full row rank matrices in $\mathbb{Z}_q^{(\ell-n)\times n}$).

When $\ell-n$ is $\mathcal{O}(1)$, the cardinality of the set of full rank matrices in $\mathbb{Z}_q^{(\ell-n)\times n}$ is a significant fraction $\left(\frac{1}{\mathcal{O}(1)}\right)$ of the cardinality of the set of all matrices in $\mathbb{Z}_q^{(\ell-n)\times n}$. Therefore, if there exists an algorithm that can efficiently distinguish between the distribution of $((\boldsymbol{v},-\boldsymbol{v}\cdot\boldsymbol{R}_1^{-1}\boldsymbol{S}^T)+\boldsymbol{e})=(\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}+\boldsymbol{e})\boldsymbol{R}'$ from the uniform distribution on $\mathbb{Z}_q^\ell$ for a non-negligible fraction of choices of $\boldsymbol{S}$ and $\boldsymbol{R}_1$, it can also distinguish the distribution on $(\boldsymbol{A},\boldsymbol{B})$ in Lemma 3.1.3 from the uniform one for a non-negligible fraction of $\hat{\boldsymbol{S}}$ (Here, non-negligible means $\left(\frac{1}{\mathcal{O}(n^c)}\right)$ for some constant c).

Now, suppose there exist a non-zero message $\boldsymbol{m}$ and an algorithm W that can distinguish the corresponding distribution of $\boldsymbol{c}'=(\boldsymbol{p}\lfloor\frac{q}{2}\rfloor+\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}+\boldsymbol{e})\boldsymbol{R}'$ from the distribution of $(\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}+\boldsymbol{e})\boldsymbol{R}'$ (for a non-negligible fraction of choices of $\boldsymbol{S}_{\text{enc}}$ and $\boldsymbol{R}'$) then such an algorithm can be used to create an algorithm W' that distinguishes between the distribution on $(\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}+\boldsymbol{e}_i)\boldsymbol{R}'$ and the uniform one. Let $p_W(\boldsymbol{m})$ be the probability that W returns 1 when the input is sampled from the distribution on $\boldsymbol{c}'=((0,\boldsymbol{m})+\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}+\boldsymbol{e})\boldsymbol{R}'$ and let $p_W(\boldsymbol{0})$ be the probability that W returns 1 when the input is sampled from the distribution on $(\boldsymbol{y}\cdot\boldsymbol{S}_{\text{enc}}+\boldsymbol{e})\boldsymbol{R}'$. Suppose $|p_W(\boldsymbol{0})-p_W(\boldsymbol{m})|>\epsilon$ for some significant value ϵ. Then, either $|p_W(\boldsymbol{0})-p_W(\boldsymbol{U})|$ or $|p_W(\boldsymbol{m})-p_W(\boldsymbol{U})|$ must be greater than $\frac{\epsilon}{2}$, where $p_W(\boldsymbol{U})$ is the probability that W returns 1 when the input is sampled from the uniform distribution. If $|p_W(\boldsymbol{0})-p_W(\boldsymbol{U})|>\frac{\epsilon}{2}$ then W' is identical to W. Otherwise, W' calls the algorithm W after altering its

input by adding $(\mathbf{0}, \boldsymbol{m})$ to it. (These arguments are similar to the arguments in the proof of Lemma 5.4 in [Reg09].) Let $\boldsymbol{c_m}$ denote the encryption of a message $\boldsymbol{m}$ under the secret key $\boldsymbol{S}, \boldsymbol{R}_1, \boldsymbol{R}_2$. The above arguments prove that the probability of distinguishing the distribution of $\boldsymbol{c_m}(\boldsymbol{R''})^{-1}$ from that of $\boldsymbol{c_0}(\boldsymbol{R''})^{-1}$ for any efficient algorithm is negligible under the LWE assumption (The probability is taken over the choices of $\boldsymbol{S}$ and $\boldsymbol{R_1}$ and the randomness involved in the encryption process).

Since $\boldsymbol{R}_2$ is chosen randomly from a uniform distribution, the above arguments imply that, under the LWE assumption, there exists no efficient algorithm that can distinguish between encryptions of a non zero message $\boldsymbol{m}$ from the encryptions of the $\mathbf{0}$ vector. Now, for an algorithm $\hat{W}$, let $p_{\hat{W}}(\boldsymbol{m})$ denote the probability of $\hat{W}$ returning 1 when the input is sampled from the distribution on the encryptions of $\boldsymbol{m}$. Clearly, for two distinct message vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$

$$|p_{\hat{W}}(\boldsymbol{m}_1) - p_{\hat{W}}(\boldsymbol{m}_2)| \leq |p_{\hat{W}}(\boldsymbol{m}_1) - p_{\hat{W}}(\mathbf{0})| + |p_{\hat{W}}(\boldsymbol{m}_2) - p_{\hat{W}}(\mathbf{0})| \tag{3.10}$$

Since, under the LWE assumption, both the terms on the right hand side of the above equation are negligible for all efficient algorithms, the value of the term on the left hand side is also negligible. Further, since there exists no efficient algorithm W such that $|p_W(\mathbf{0}) - p_W(\boldsymbol{U})|$ is non-negligible, there exists no efficient algorithm $\hat{W}$ such that $|p_{\hat{W}}(\mathbf{0}) - p_{\hat{W}}(\boldsymbol{U})|$ is non-negligible. For any message $\boldsymbol{m}$ and algorithm $\hat{W}$

$$|p_{\hat{W}}(\boldsymbol{m}) - p_{\hat{W}}(U)| \leq |p_{\hat{W}}(\boldsymbol{m}) - p_{\hat{W}}(\mathbf{0})| + |p_{\hat{W}}(\mathbf{0}) - p_{\hat{W}}(U)|. \tag{3.11}$$

Therefore, under the LWE assumption, there exists no efficient algorithm $\hat{W}$ such that $|p_{\hat{W}}(\boldsymbol{m}) - p_{\hat{W}}(U)|$ is non-negligible. In other words, there exists no efficient algorithm that can distinguish the distribution on the encryptions of

a message $\boldsymbol{m}$ from the uniform distribution on $\mathbb{Z}_q^\ell$. □

3.1.3 Homomorphic Properties

The proposed scheme can be used to homomorphically evaluate a function $\phi : \{0,1\}^{\tau(\ell-n)} \rightarrow \{0,1\}^{\ell-n}$ on ciphertexts $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_\tau$ such that $\phi(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_\tau)$ yields a ciphertext $\boldsymbol{c}_\phi$. In the proposed scheme, ϕ represents an arithmetic circuit over $GF(2)$ with addition and multiplication gates. We now show how to perform homomorphic addition and multiplication of two ciphertexts in the proposed scheme.

Addition.

Addition is performed by simply adding the ciphertexts. For some $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathbb{Z}_q^n$, if $\boldsymbol{c}_1 = \left(\boldsymbol{p}_1 \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y}_1 \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}_1\right)\boldsymbol{R} \ mod\ q$ and $\boldsymbol{c}_2 = \left(\boldsymbol{p}_2 \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y}_2 \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}_2\right)\boldsymbol{R} \ mod\ q$ denote the respective encryptions of $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$, then compute

$$\begin{aligned} \boldsymbol{c}_{add} &= \boldsymbol{c}_1 + \boldsymbol{c}_2 \ \ (mod\ q) \\ &= \left((\boldsymbol{p}_1 + \boldsymbol{p}_2)\left\lfloor \frac{q}{2} \right\rfloor + (\boldsymbol{y}_1 + \boldsymbol{y}_2) \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}_1 + \boldsymbol{e}_2\right)\boldsymbol{R}\ (mod\ q) \end{aligned} \tag{3.12}$$

where $\boldsymbol{e}_i = (\boldsymbol{0}, \tilde{\boldsymbol{e}}_i)$ for some $\tilde{\boldsymbol{e}}_i \leftarrow \mathcal{X}^{\ell-n}$ for $i \in \{1,2\}$. If $\tilde{\boldsymbol{e}}_{add} := \tilde{\boldsymbol{e}}_1 + \tilde{\boldsymbol{e}}_2$, then

$$\begin{aligned} \boldsymbol{c}_{add} \cdot \boldsymbol{S}_{\text{dec}} &= (\boldsymbol{m}_1 + \boldsymbol{m}_2)\left\lfloor \frac{q}{2} \right\rfloor + \tilde{\boldsymbol{e}}_{add}\ (mod\ q) \\ &= (\boldsymbol{m}_1 \oplus \boldsymbol{m}_2)\left\lfloor \frac{q}{2} \right\rfloor - \frac{1}{2}[\boldsymbol{m}_1 + \boldsymbol{m}_2 - (\boldsymbol{m}_1 \oplus \boldsymbol{m}_2)] + \tilde{\boldsymbol{e}}_{add}\ (mod\ q) \end{aligned} \tag{3.13}$$

If $\boldsymbol{e}_{add} := -\frac{1}{2}[\boldsymbol{m}_1 + \boldsymbol{m}_2 - (\boldsymbol{m}_1 \oplus \boldsymbol{m}_2)] + \tilde{\boldsymbol{e}}_{add}$ and $\|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \leq B$, then the magnitude of the noise after addition can be computed as

$$\|\boldsymbol{e}_{add}\|_\infty = \left\| -\frac{1}{2}[\boldsymbol{m}_1 + \boldsymbol{m}_2 - (\boldsymbol{m}_1 \oplus \boldsymbol{m}_2)] + (\tilde{\boldsymbol{e}}_1 + \tilde{\boldsymbol{e}}_2) \right\|_\infty \leq 1 + 2B \tag{3.14}$$

Multiplication.

Given two ciphertexts $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ that encrypts the messages $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$, homomorphic multiplication is performed by using a bilinear map on $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. This map is represented by a 3-way tensor $\boldsymbol{\mathcal{M}}$ which is provided as the public evaluation key for multiplication. We now proceed to construct $\boldsymbol{\mathcal{M}}$. All operations are performed over $\mathbb{Q}$ unless stated otherwise. For some $x \in \mathbb{Q}$, $y = x \ (mod\ q)$ denotes the unique value in the interval $(-q/2, q/2]$.

Homomorphic multiplication in this scheme uses the fact that given two polynomials $f_1, f_2 \in \mathcal{I}_{\leq r}$ and an evaluation point $\boldsymbol{z} \in \mathbb{Z}_q^v$,

$$f_1(\boldsymbol{z}) \cdot f_2(\boldsymbol{z}) = (f_1 \cdot f_2)(\boldsymbol{z}) \tag{3.15}$$

Although $f_1 f_2 \in \mathcal{I}$ it need not be an element of the subspace $\mathcal{I}_{\leq r}$. Instead, it is an element of the space $\mathcal{I}_{\leq 2r}$. Let n' denote the dimension of the subspace $\mathcal{I}_{\leq 2r}$ and $t = n' + \ell - n$. We choose $(n' - n)$ additional points $\boldsymbol{z}_{\ell+1}, \ldots, \boldsymbol{z}_t$ in $\mathbb{Z}_q^v$ such that every vector in $\mathbb{Z}_q^{n'}$ can be obtained by evaluating a polynomial in $\mathcal{I}_{\leq 2r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n, \boldsymbol{z}_{\ell+1}, \ldots, \boldsymbol{z}_t)$. Evaluating polynomials in $\mathcal{I}_{\leq 2r}$ on the points $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_t$ yields an n' dimensional subspace of $\mathbb{Z}_q^t$

Let f_{mult} be a polynomial in $\mathcal{I}_{\leq r}$ and $(f_{mult}(\boldsymbol{z}_1), \ldots, f_{mult}(\boldsymbol{z}_\ell)) = \boldsymbol{y}_{mult} \cdot \boldsymbol{S}_{\text{enc}}$ for some $\boldsymbol{y}_{mult} \in \mathbb{Z}_q^n$. Then, given encryptions of $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ viz. $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, our aim is to get a ciphertext of the form

$$\boldsymbol{c}_{mult} = \left((\boldsymbol{p}_1 \odot \boldsymbol{p}_2) \cdot \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y}_{mult} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}_{mult} \right) \boldsymbol{R} \ mod\ q \in \mathbb{Z}_q^\ell \tag{3.16}$$

For $i := \{1, 2\}$, ciphertexts $\boldsymbol{c}_i$ that encrypt messages $\boldsymbol{m}_i$ are given as

$$\boldsymbol{c}_i = \begin{bmatrix} f_i(\boldsymbol{z}_1) \\ \vdots \\ f_i(\boldsymbol{z}_n) \\ m_{i,n+1}\left\lfloor \frac{q}{2} \right\rfloor + f_i(\boldsymbol{z}_{n+1}) + e_{i,n+1} \\ \vdots \\ m_{i,\ell}\left\lfloor \frac{q}{2} \right\rfloor + f_i(\boldsymbol{z}_\ell) + e_{i,\ell} \end{bmatrix}^T \boldsymbol{R} \ (mod\ q) \tag{3.17}$$

where f_is are polynomials that are randomly sampled from $\mathcal{I}_{\leq r}$. The process of homomorphically multiplying $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ is done through the following steps.

1. For some $K_{i,j} \in \mathbb{Z}, i \in \{1, 2\}$, transform the ciphertext $\boldsymbol{c}_i$ to a vector of the form,

$$\tilde{\boldsymbol{c}}_i = \begin{bmatrix} f_i(\boldsymbol{z}_1) \\ \vdots \\ f_i(\boldsymbol{z}_n) \\ m_{i,n+1}\left\lfloor \frac{q}{2} \right\rfloor + e'_{i,n+1} + qK_{i,n+1} \\ \vdots \\ m_{i,\ell}\left\lfloor \frac{q}{2} \right\rfloor + e'_{i,\ell} + qK_{i,\ell} \end{bmatrix}^T \in \mathbb{Q}^\ell \tag{3.18}$$

   Note that the noise terms $e_{i,j}$ have transformed to $e'_{i,j}$. This is because we deliberately introduce some noise in this step. The reason for the same is explained later in the chapter.

2. For some $K_{i,j} \in \mathbb{Z}$ for $i \in \{1, 2\}$ and $\ell + 1 \leq j \leq t$, compute values that are equivalent mod q to evaluations of f_i at the additional points, $\boldsymbol{z}_{\ell+1}, \ldots, \boldsymbol{z}_t$ and append these entries to $\tilde{\boldsymbol{c}}_i$ to generate vector $\boldsymbol{c}'_i \in \mathbb{Q}^t$

where

$$\boldsymbol{c}'_i = \begin{bmatrix} f_i(\boldsymbol{z}_1) \\ \vdots \\ f_i(\boldsymbol{z}_n) \\ m_{i,n+1}\left\lfloor \frac{q}{2} \right\rfloor + e'_{i,n+1} + qK_{i,n+1} \\ \vdots \\ m_{i,\ell}\left\lfloor \frac{q}{2} \right\rfloor + e'_{i,\ell} + qK_{i,\ell} \\ f_i(\boldsymbol{z}_{\ell+1}) + qK_{i,\ell+1} \\ \vdots \\ f_i(\boldsymbol{z}_t) + qK_{i,t} \end{bmatrix}^T \in \mathbb{Q}^t \tag{3.19}$$

3. Take component-wise product of $\boldsymbol{c}'_1$ and $\boldsymbol{c}'_2$ and multiply the entries containing the message with $2/q$ to get

$$\boldsymbol{c}'_{mult} = \begin{bmatrix} f_1 f_2(\boldsymbol{z}_1) \\ \vdots \\ f_1 f_2(\boldsymbol{z}_n) \\ \frac{2}{q}\left(m_{1,n+1}\left\lfloor \frac{q}{2} \right\rfloor + e'_{1,n+1} + qK_{1,n+1}\right)\left(m_{2,n+1}\left\lfloor \frac{q}{2} \right\rfloor + e'_{2,n+1} + qK_{2,n+1}\right) \\ \vdots \\ \frac{2}{q}\left(m_{1,\ell}\left\lfloor \frac{q}{2} \right\rfloor + e'_{1,\ell} + qK_{1,\ell}\right)\left(m_{2,\ell}\left\lfloor \frac{q}{2} \right\rfloor + e'_{2,\ell} + qK_{2,\ell}\right) \\ f_1 f_2(\boldsymbol{z}_{\ell+1}) + qK_{\ell+1} \\ \vdots \\ f_1 f_2(\boldsymbol{z}_t) + qK_t \end{bmatrix}^T \tag{3.20}$$

where $K_j \in \mathbb{Z}$ for $\ell + 1 \leq j \leq t$.

4. Add integers equivalent to $f_1 f_2(\boldsymbol{z}_j)$ *mod* q to the j^{th} entries of $\boldsymbol{c}'_{mult}$ for $n+1 \leq j \leq \ell$. Let the resultant vector be $\boldsymbol{c}''_{mult} \in \mathbb{Q}^t$ such that for $n+1 \leq j \leq \ell$

$$\boldsymbol{c}''_{mult}(j) = \frac{2}{q}\left(m_{1,j}\left\lfloor \frac{q}{2} \right\rfloor + e'_{1,j} + qK_{1,j}\right)\left(m_{2,j}\left\lfloor \frac{q}{2} \right\rfloor + e'_{2,j} + qK_{2,j}\right) + f_1 f_2(\boldsymbol{z}_j) + qK_j \tag{3.21}$$

where $K_j \in \mathbb{Z}$ for $n+1 \leq j \leq \ell$. The remaining entries of $\boldsymbol{c}''_{mult}$ are the same as that of $\boldsymbol{c}'_{mult}$.

5. Transform the vector $\boldsymbol{c}''_{mult} \in \mathbb{Q}^t$ to a valid ciphertext $\boldsymbol{c}_{mult}$ of size ℓ over $\mathbb{Z}_q$.

We now explain in detail how each of the above steps is performed.

Step 1: The first step is to transform the ciphertexts $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ to the vectors $\tilde{\boldsymbol{c}}_1$ and $\tilde{\boldsymbol{c}}_2$ given in Equation (4.13). For $i := \{1, 2\}$, let $\boldsymbol{D}_i$s be matrices given by

$$\boldsymbol{D}_i = \boldsymbol{R}^{-1} \cdot \left[\begin{array}{c|c} \boldsymbol{I}_n & \boldsymbol{S}^T \\ \hline \boldsymbol{0} & \boldsymbol{I}_{\ell-n} \end{array}\right] + \left[\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{\epsilon}_i \\ \hline \boldsymbol{0} & \boldsymbol{0} \end{array}\right] \tag{3.22}$$

$$= \left[\begin{array}{c|c} {\boldsymbol{R}_1}^{-1} & {\boldsymbol{R}_1}^{-1}\boldsymbol{S}^T + \boldsymbol{\epsilon}_i \\ \hline -\boldsymbol{R}_2{\boldsymbol{R}_1}^{-1} & -\boldsymbol{R}_2{\boldsymbol{R}_1}^{-1}\boldsymbol{S}^T + \boldsymbol{I}_{\ell-n} \end{array}\right] \in \mathbb{Q}^{\ell \times \ell} \tag{3.23}$$

where $\boldsymbol{\epsilon}_i$s are matrices in $\mathbb{Q}^{n\times(\ell-n)}$ such that the one norm of each of their columns is less that B/q. Observe that, for $i := \{1, 2\}$ and $n+1 \leq j \leq \ell$, $\langle \boldsymbol{c}_i\boldsymbol{R}^{-1}, \tilde{\boldsymbol{s}}_j \rangle = m_{i,j}\left\lfloor \frac{q}{2} \right\rfloor + e_{i,j} + qK_{i,j}$, where $K_{1,j}, K_{2,j} \in \mathbb{Z}$ and the $\tilde{\boldsymbol{s}}_j$s are as given in Equation (3.1). Therefore, for $i := \{1, 2\}$, $\tilde{\boldsymbol{c}}_i = \boldsymbol{c}_i\boldsymbol{D}_i$ and the error terms $e'_{i,j}$ are given by $e'_{i,j} = e_{i,j} + \langle \boldsymbol{c}_i, (\boldsymbol{\epsilon}_i(:, j), \boldsymbol{0}) \rangle$ where $\boldsymbol{0} \in \mathbb{Q}^{\ell-n}$ is the all zero vector. Both terms in the right hand side of this equation are bounded by B.

Step 2: Evaluation of polynomials in $\mathcal{I}_{\leq r}$ at $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_t$ constitutes an n-dimensional subspace of $\mathbb{Z}_q^t$. Therefore, for $i \in \{1, 2\}$ and $\ell+1 \leq j \leq t$, there exists $\boldsymbol{\alpha}_j := (\alpha_j^1, \ldots, \alpha_j^n) \in \mathbb{Z}_q^n$ such that

$$f_i(\boldsymbol{z}_j) := \sum_{k=1}^{n} \alpha_j^k \cdot f_i(\boldsymbol{z}_k) \ (mod\ q) \tag{3.24}$$

Consequently, $\boldsymbol{c}'_i = \tilde{\boldsymbol{c}}_i \cdot \boldsymbol{A} \in \mathbb{Q}^t$ (this multiplication is performed by considering the elements of $\boldsymbol{A}$ to be in $\mathbb{Q}$) where $\boldsymbol{A}$ is given by

$$\boldsymbol{A} := \left[\begin{array}{c|c|ccc} & & \alpha_{\ell+1}^1 & \dots & \alpha_t^1 \\ \boldsymbol{I}_n & \boldsymbol{0} & \vdots & \ddots & \vdots \\ & & \alpha_{\ell+1}^n & \dots & \alpha_t^n \\ \hline \boldsymbol{0} & \boldsymbol{I}_{\ell-n} & \multicolumn{3}{c}{\boldsymbol{0}} \end{array}\right] \in \mathbb{Z}_q^{\ell \times t} \tag{3.25}$$

Step 3: $\boldsymbol{c}'_{mult} \in \mathbb{Q}^t$ in Equation (3.20) is obtained from $\boldsymbol{c}'_1$ and $\boldsymbol{c}'_2$ by taking their element-wise product and then multiplying the entries from $n+1$ to ℓ by $2/q$. This operation can be done by evaluating a tensor $\boldsymbol{\mathcal{U}} \in \mathbb{Q}^{t \times t \times t}$ on $\boldsymbol{c}'_1$ and $\boldsymbol{c}'_2$. For $1 \leq i \leq t$, let $\boldsymbol{U}_i$ denote the i-th frontal slice of $\boldsymbol{\mathcal{U}}$. For $1 \leq i \leq n$ and $\ell+1 \leq i \leq t$, the corresponding matrix $\boldsymbol{U}_i$, has 1 in the (i,i)-th position and 0 everywhere else. For $n+1 \leq i \leq \ell$, $\boldsymbol{U}_i(i,i) = 2/q$ and $\boldsymbol{U}_i(i,j) = 0$ when $i \neq j$. For example, if $n = 2, \ell = 4$ and $t = 5$, then the frontal slices of $\boldsymbol{\mathcal{U}}$ are given by the following matrices.

$$\boldsymbol{U}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{U}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.26}$$

$$\boldsymbol{U}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{2}{q} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{U}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{2}{q} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{U}_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.27}$$

Then, $\boldsymbol{c}'_{mult}$ can be computed using $\boldsymbol{\mathcal{U}}$ as

$$\boldsymbol{c}'_{mult} = \begin{bmatrix} \boldsymbol{c}'_1 \boldsymbol{U}_1 \boldsymbol{c}'^{T}_2 & \dots & \boldsymbol{c}'_1 \boldsymbol{U}_t \boldsymbol{c}'^{T}_2 \end{bmatrix} \in \mathbb{Q}^t \tag{3.28}$$

Given $\tilde{\boldsymbol{c}}_1$ and $\tilde{\boldsymbol{c}}_2$ from Step 1, $\boldsymbol{c}'_{mult}$ is obtained by evaluating the following tensor $\boldsymbol{\mathcal{T}} \in \mathbb{Q}^{\ell \times \ell \times t}$ on $\tilde{\boldsymbol{c}}_1$ and $\tilde{\boldsymbol{c}}_2$.

$$\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{U}} \times_1 \boldsymbol{A} \times_2 \boldsymbol{A} \in \mathbb{Q}^{\ell \times \ell \times t} \tag{3.29}$$

For example, if $n = 2$ and $\ell = 4$, then the frontal slices of $\boldsymbol{\mathcal{T}}$ for $1 \leq i \leq \ell$ can be represented by the following matrices.

$$\boldsymbol{T}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \boldsymbol{T}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.30}$$

$$\boldsymbol{T}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{2}{q} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \boldsymbol{T}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{2}{q} \end{bmatrix} \tag{3.31}$$

For any $t > \ell$, $\boldsymbol{T}_i$ for $\ell + 1 \leq i \leq t$ is a matrix of the form given below.

$$\boldsymbol{T}_i := \left[\begin{array}{cccc|c} (\alpha_i^1)^2 & \alpha_i^1 \alpha_i^2 & \dots & \alpha_i^1 \alpha_i^n & \\ \vdots & \vdots & \ddots & \vdots & 0^{n \times (\ell - n)} \\ \alpha_i^1 \alpha_i^n & \alpha_i^2 \alpha_i^n & \dots & (\alpha_i^n)^2 & \\ \hline \multicolumn{4}{c|}{0^{(\ell-n) \times n}} & 0^{(\ell-n) \times (\ell-n)} \end{array}\right] \in \mathbb{Q}^{\ell \times \ell} \tag{3.32}$$

Therefore, given $\tilde{\boldsymbol{c}}_1$ and $\tilde{\boldsymbol{c}}_2$ from Step 1, we can compute the vector $\boldsymbol{c}'_{mult}$ as:

$$\boldsymbol{c}'_{mult} := \tilde{\boldsymbol{c}}_1 \boldsymbol{\mathcal{T}} \tilde{\boldsymbol{c}}_2^T = \begin{bmatrix} \tilde{\boldsymbol{c}}_1 \boldsymbol{T}_1 \tilde{\boldsymbol{c}}_2^T & \cdots & \tilde{\boldsymbol{c}}_1 \boldsymbol{T}_t \tilde{\boldsymbol{c}}_2^T \end{bmatrix} \in \mathbb{Q}^t \tag{3.33}$$

Step 4: The evaluations of polynomials in $\mathcal{I}_{\leq 2r}$ on $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_t$ constitute an n'-dimensional subspace of $\mathbb{Z}_q^t$. Because of the way in which the points $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n, \boldsymbol{z}_{\ell+1}, \ldots, \boldsymbol{z}_t)$ are chosen, the evaluations of $f_1 f_2 \in \mathcal{I}_{\leq 2r}$ at the points $(\boldsymbol{z}_{n+1}, \ldots, \boldsymbol{z}_\ell)$ can be written as a linear combination of its evaluations at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n, \boldsymbol{z}_{\ell+1}, \ldots, \boldsymbol{z}_t)$. Therefore, for some $(\beta_j^1, \ldots, \beta_j^n, \beta_j^{\ell+1}, \ldots, \beta_j^t) \in \mathbb{Z}_q^{n'}$, $n+1 \leq j \leq \ell$,

$$f_1 f_2(\boldsymbol{z}_j) := \sum_{i=1}^{n} \beta_j^i \cdot f_1 f_2(\boldsymbol{z}_i) + \sum_{i=\ell+1}^{t} \beta_j^i \cdot f_1 f_2(\boldsymbol{z}_i) \ (mod\ q) \tag{3.34}$$

Let $\boldsymbol{B}_1$ be a matrix of size $n \times (\ell - n)$ such that $\boldsymbol{B}_1(i,j) = \beta_j^i$ for $1 \leq i \leq n$ and $n+1 \leq j \leq \ell$. Similarly, let $\boldsymbol{B}_2$ be a matrix of size $(t-\ell) \times (\ell - n)$ such that $\boldsymbol{B}_2(i,j) = \beta_j^i$ for $n+1 \leq j \leq \ell$ and $\ell+1 \leq i \leq t$. Consider the following block matrix

$$\boldsymbol{B} = \left[\begin{array}{c|c|c} \boldsymbol{I}_n & \boldsymbol{B}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{I}_{\ell-n} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{B}_2 & \boldsymbol{I}_{t-\ell} \end{array}\right] \in \mathbb{Z}_q^{t \times t} \tag{3.35}$$

$\boldsymbol{c}''_{mult}$ is obtained by multiplying $\boldsymbol{c}'_{mult}$ with the matrix $\boldsymbol{B}$ i.e. $\boldsymbol{c}''_{mult} = \boldsymbol{c}'_{mult} \cdot \boldsymbol{B}$ (considering the elements of $\boldsymbol{B}$ to be in $\mathbb{Q}$).

Step 5: In order to transform the vector $\boldsymbol{c}''_{mult}$ generated in the above step to a valid ciphertext $\boldsymbol{c}_{mult}$ we first define a map from $\mathcal{I}_{\leq 2r}$ to $\mathcal{I}_{\leq r}$ as follows.

Let LM($\mathcal{I}$) be the set of leading monomials of elements of $\mathcal{I}$. Let LM$(\mathcal{I})_{r+1} := \{\mu_1, \mu_2, \ldots, \mu_M\}$ be the set of all monomials of degree $r+1$

in LM($\mathcal{I}$). For each monomial μ_i, choose a polynomial $g_i \in \mathcal{I}$ such that the leading term of g_i is μ_i for $1 \leq i \leq M$. Let $\mathcal{G} := \{g_1, \ldots, g_M\}$ be the set of these polynomials. Given $f \in \mathcal{I}_{\leq 2r}$, serially divide f by the set of polynomials $\mathcal{G}$ using the degree reverse lexicographic order (At each step divide the remainder obtained in the previous step by the next g_i). Let $f_{\mathcal{G}}$ be the final remainder obtained. Note that the map from f to $f_{\mathcal{G}}$ is a linear one.

The above linear map from $\mathcal{I}_{\leq 2r}$ to $\mathcal{I}_{\leq r}$ naturally gives rise to the following linear map $\mathscr{L}$ from the evaluations of polynomials in $\mathcal{I}_{\leq 2r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_t)$ to the evaluations of polynomials in $\mathcal{I}_{\leq r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell)$.

$$\mathscr{L}\left(f(\boldsymbol{z}_1), f(\boldsymbol{z}_2), \ldots, f(\boldsymbol{z}_t)\right) = \left(f_{\mathcal{G}}(\boldsymbol{z}_1), f_{\mathcal{G}}(\boldsymbol{z}_2), \ldots, f_{\mathcal{G}}(\boldsymbol{z}_\ell)\right)$$

Let $(f^1, \ldots, f^{n'})$ be a basis for $\mathcal{I}_{\leq 2r}$ and let $(f^1_{\mathcal{G}}, \ldots, f^{n'}_{\mathcal{G}})$ be the respective remainders after serially dividing by the elements of $\mathcal{G}$. Let $\boldsymbol{F}_1 \in \mathbb{Z}_q^{n' \times t}$ and $\boldsymbol{F}_2 \in \mathbb{Z}_q^{n' \times \ell}$ be the following matrices,

$$\boldsymbol{F}_1 := \begin{bmatrix} f^1(\boldsymbol{z}_1) & \ldots & f^1(\boldsymbol{z}_\ell) & \ldots & f^1(\boldsymbol{z}_t) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ f^{n'}(\boldsymbol{z}_1) & \ldots & f^{n'}(\boldsymbol{z}_\ell) & \ldots & f^{n'}(\boldsymbol{z}_t) \end{bmatrix}, \tag{3.36}$$

$$\boldsymbol{F}_2 := \begin{bmatrix} f^1_{\mathcal{G}}(\boldsymbol{z}_1) & \ldots & f^1_{\mathcal{G}}(\boldsymbol{z}_\ell) \\ \vdots & \ddots & \vdots \\ f^{n'}_{\mathcal{G}}(\boldsymbol{z}_1) & \ldots & f^{n'}_{\mathcal{G}}(\boldsymbol{z}_\ell) \end{bmatrix} \tag{3.37}$$

Every $\boldsymbol{Q}$ that satisfies the equation $\boldsymbol{F}_1 \cdot \boldsymbol{Q} = \boldsymbol{F}_2 \ (mod\ q)$, defines a map $\mathscr{L}_Q$ from $\mathbb{Z}_q^t$ to $\mathbb{Z}_q^\ell$ which when restricted to the evaluations of $\mathcal{I}_{\leq 2r}$ results in $\mathscr{L}$.

In particular, there exist solutions $\boldsymbol{Q}$ that have the following structure.

$$\boldsymbol{Q} = \left[\begin{array}{c|c|c} \boldsymbol{Q}_1^{n\times n} & 0^{n\times(\ell-n)} & \boldsymbol{Q}_2^{n\times(t-\ell)} \\ \hline \boldsymbol{Q}_3^{(\ell-n)\times n} & \boldsymbol{I}_{\ell-n} & \boldsymbol{Q}_4^{(\ell-n)\times(t-\ell)} \end{array}\right]^T \in \mathbb{Z}_q^{t\times\ell} \tag{3.38}$$

Now, when $\boldsymbol{c}''_{mult}$ is multiplied with the matrix $\boldsymbol{Q}$ (considering the elements of $\boldsymbol{Q}$ to be in $\mathbb{Q}$) we get the following vector $\tilde{\boldsymbol{c}}_{mult}$

$$\tilde{\boldsymbol{c}}_{mult} = \begin{bmatrix} f_{mult}(\boldsymbol{z}_1) + qK'_1 \\ \vdots \\ f_{mult}(\boldsymbol{z}_n) + qK'_n \\ \frac{2}{q}\left(m_{1,n+1}\left\lfloor\frac{q}{2}\right\rfloor + e'_{1,n+1} + qK_{1,n+1}\right)\left(m_{2,n+1}\left\lfloor\frac{q}{2}\right\rfloor + e'_{2,n+1} + qK_{2,n+1}\right) + f_{mult}(\boldsymbol{z}_{n+1}) + qK'_{n+1} \\ \vdots \\ \frac{2}{q}\left(m_{1,\ell}\left\lfloor\frac{q}{2}\right\rfloor + e'_{1,\ell} + qK_{1,\ell}\right)\left(m_{2,\ell}\left\lfloor\frac{q}{2}\right\rfloor + e'_{2,\ell} + qK_{2,\ell}\right) + f_{mult}(\boldsymbol{z}_\ell) + qK'_\ell \end{bmatrix}^T \tag{3.39}$$

for some $K'_j \in \mathbb{Z}$ for $1 \leq j \leq \ell$. Here f_{mult} denotes the remainder obtained after serially dividing $f_1 f_2$ by the elements of $\mathcal{G}$. Multiplying $\tilde{\boldsymbol{c}}_{mult}$ by $\boldsymbol{R}$ gives us a vector in $\mathbb{Q}^\ell$ which when rounded to the closest integer vector is equivalent *mod* q to a vector which gives $\boldsymbol{m}_1 \odot \boldsymbol{m}_2$ on decryption i.e,

$$\boldsymbol{c}_{mult} = \lfloor \tilde{\boldsymbol{c}}_{mult} \cdot \boldsymbol{R} \rceil \ \ mod\ q \in \mathbb{Z}_q^\ell \tag{3.40}$$

The sequence of steps that transform the pair $\boldsymbol{c}_1, \boldsymbol{c}_2$ to $\tilde{\boldsymbol{c}}_{mult}\boldsymbol{R}$ constitute a bilinear map from $\mathbb{Q}^\ell \times \mathbb{Q}^\ell$ to $\mathbb{Q}^\ell$. This map, denoted by $\mathcal{B}_{\boldsymbol{\mathcal{M}}}$, can be represented by a 3-way tensor $\boldsymbol{\mathcal{M}}$ where $\boldsymbol{\mathcal{M}}$ is given by

$$\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{T}} \times_1 \boldsymbol{D}_1 \times_2 \boldsymbol{D}_2 \times_3 (\boldsymbol{R}^T \boldsymbol{Q}^T \boldsymbol{B}^T) \in \mathbb{Q}^{\ell\times\ell\times\ell} \tag{3.41}$$

The tensor $\boldsymbol{\mathcal{M}}$ is the evaluation key for multiplication. If $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_\ell$ denote

the frontal slices of $\boldsymbol{\mathcal{M}}$, then using $\boldsymbol{\mathcal{M}}$, $\boldsymbol{c}_{mult}$ can be computed as:

$$\begin{aligned}\boldsymbol{c}_{mult} &= \lfloor \mathcal{B}_{\boldsymbol{\mathcal{M}}}(\boldsymbol{c}_1, \boldsymbol{c}_2) \rfloor \ mod\, q \\ &= \left[\left\lfloor \boldsymbol{c}_1 \boldsymbol{M}_1 \boldsymbol{c}_2^T \right\rfloor \quad \cdots \quad \left\lfloor \boldsymbol{c}_1 \boldsymbol{M}_\ell \boldsymbol{c}_2^T \right\rfloor \right] mod\, q \in \mathbb{Z}_q^\ell \end{aligned} \tag{3.42}$$

Correctness

If $f_{mult}(\boldsymbol{Z}) = \boldsymbol{y}_{mult} \cdot \boldsymbol{S}_{\text{enc}}$ for some $\boldsymbol{y}_{mult} \in \mathbb{Z}_q^n$ and $\boldsymbol{e}_{mult} = (0, \ldots, 0, e_{mult,n+1}, \ldots, e_{mult,\ell})$ denotes the noise vector, then the resultant ciphertext after the multiplication process can be written as

$$\boldsymbol{c}_{mult} = \left((\boldsymbol{p}_1 \odot \boldsymbol{p}_2) \left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{y}_{mult} \cdot \boldsymbol{S}_{\text{enc}} + \boldsymbol{e}_{mult} \right) \cdot \boldsymbol{R} \ mod\ q \in \mathbb{Z}_q^\ell \tag{3.43}$$

Then $\boldsymbol{c}_{mult} \cdot \boldsymbol{S}_{\text{dec}} = (\boldsymbol{m}_1 \odot \boldsymbol{m}_2) \left\lfloor \frac{q}{2} \right\rfloor + \tilde{\boldsymbol{e}}_{mult} \ (mod\ q)$ where $\tilde{\boldsymbol{e}}_{mult} = (e_{mult,n+1}, \ldots, e_{mult,\ell})$. The decryption process outputs $\boldsymbol{m}_1 \odot \boldsymbol{m}_2$ if $\|\boldsymbol{e}_{mult}\|_\infty < \lfloor q/2 \rfloor /2$.

3.1.4 Security with Multiplicative Homomorphism

The entries of the tensor $\boldsymbol{\mathcal{M}} \in \mathbb{Q}^{\ell \times \ell \times \ell}$ are polynomials in the entries of the matrices $\boldsymbol{S}, \boldsymbol{R}_1, \boldsymbol{R}_2, \boldsymbol{Q}$, the $\boldsymbol{\epsilon}_i$s, the α_i^js and β_i^js. Equating these polynomials with a given instance of the evaluation key results in a system of $\mathcal{O}(\ell^3)$ equations in $\mathcal{O}(\ell^2)$ variables. The α_i^js, β_i^js and the entries of the matrix $\boldsymbol{Q}$ depend on the extra $t - \ell$ points that are chosen independent of the secret key. The entries of $\boldsymbol{Q}$ also depend on the polynomials chosen for the quotienting operation. Therefore, in order to retrieve the secret key from the evaluation key one would have to solve a system of polynomial equations. The problem of Polynomial System Solving (PoSSo) is known to be NP-hard in general. Most multivariate public key schemes rely on the hardness of

solving this problem. For detailed analysis of this problem, one may refer to [Laz83, KS99, FJ03, CP02, AFF+16, BFP09]. Further, the system of equations in this case is underdetermined (as a system of equations over $\mathbb{Q}$).

Observe that if the $\boldsymbol{\epsilon}_i$s are zero then the multiplication process maps two elements of $\mathcal{S}_{\mathcal{I}_{\leq r}}$ to another element of $\mathcal{S}_{\mathcal{I}_{\leq r}}$ (as vectors in $\mathbb{Z}_q^\ell$). Thus, $\mathcal{S}_{\mathcal{I}_{\leq r}}$ is an invariant subspace of this process. This fact could potentially be used to extract the secret key from the evaluation key. (Although, to the best of the authors' knowledge there are no efficient algorithms to extract such subspaces for all ℓ). The $\boldsymbol{\epsilon}_i$s ensure that this invariance is removed.

3.1.5 Noise in Multiplication

Let us now analyze the noise in the decryption of $\boldsymbol{c}_{mult}$. Observe that, for $n+1 \leq j \leq \ell$, the noise in $\boldsymbol{c}_{mult}(j)$ is same as that in $\tilde{\boldsymbol{c}}_{mult}(j)$. If $e_{mult,j}$ denotes the noise in the j^{th} entry of $\boldsymbol{c}_{mult}$ then using Equation (3.39), we get

$$e_{mult,j} = \frac{q-1}{q}(m_{1,j}e'_{2,j} + m_{2,j}e'_{1,j}) + (2e'_{1,j} - m_{1,j})K_{2,j} + (2e'_{2,j} - m_{2,j})K_{1,j} - \frac{m_{1,j}m_{2,j}}{2q} + \frac{2}{q}e'_{1,j}e'_{2,j} \tag{3.44}$$

The most significant term in $e_{mult,j}$ is $(2e'_{1,j} - m_{1,j})K_{2,j} + (2e'_{2,j} - m_{2,j})K_{1,j}$ where the $K_{i,j}$s are generated due to the multiplication of $\boldsymbol{c}_i$s with the matrix $\boldsymbol{D}_i$ in Step 1. Observe that the j^{th} entry of $\boldsymbol{c}_i\boldsymbol{D}_i$ is equal to $\langle \boldsymbol{c}_i\boldsymbol{R}^{-1}, \tilde{\boldsymbol{s}}_j \rangle + \langle \boldsymbol{c}_i, (\boldsymbol{\epsilon}_i(:,j), \boldsymbol{0}) \rangle$ where $\boldsymbol{0}$ is the zero vector in $\mathbb{Q}^{\ell-n}$ and $\langle \boldsymbol{c}_i\boldsymbol{R}^{-1}, \tilde{\boldsymbol{s}}_j \rangle = m_{i,j}\left\lfloor \frac{q}{2} \right\rfloor + e_{i,j} + qK_{i,j}$. The magnitude of $K_{i,j}$ is bounded by the one norm of $\boldsymbol{R}^{-1}\tilde{\boldsymbol{s}}_j$ which is $\mathcal{O}(nq)$ since $\ell = \mathcal{O}(n)$. One could choose the secret key in such a way that the one norms of the $\boldsymbol{R}^{-1}\tilde{\boldsymbol{s}}_j$s are small. Alternatively, one could use a slightly modified version of the vector decomposition techniques given in [BGV14] to limit the values of the $|K_{i,j}|$s. Firstly, for a suitable value of u (which is $\mathcal{O}(1)$), we choose the entries of the $\boldsymbol{\epsilon}_i$s (all of which are less than 1) such that their

binary expressions have less than u bits i.e. these entries can be written as $\sum_{k=1}^{u} b_k 2^{-k}$ for some b_ks in $\{0,1\}$. This technique consists of the following two functions

- $\mathsf{BitDecomp}_{q,u}(\boldsymbol{v})$: Given $\boldsymbol{v} \in \mathbb{Q}^{\ell}$, let $\boldsymbol{x}_i \in \{0,1\}^{\ell}$ be such that $\boldsymbol{v} = \sum_{i=-u}^{\lfloor \log q \rfloor} 2^i \cdot \boldsymbol{x}_i \ (mod\ q)$. Output the vector

$$\left(\boldsymbol{x}_{-u}, \ldots, \boldsymbol{x}_0, \ldots, \boldsymbol{x}_{\lfloor \log q \rfloor}\right) \in \{0,1\}^{\ell(u+\lceil \log q \rceil)}$$

- $\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{w})$: Given $\boldsymbol{w} \in \mathbb{Z}^{\ell}$, output the vector

$$\left(2^{-u} \cdot \boldsymbol{w}, \ldots, 2^{-1} \cdot \boldsymbol{w}, \boldsymbol{w}, 2 \cdot \boldsymbol{w}, \ldots, 2^{\lfloor \log q \rfloor} \cdot \boldsymbol{w}\right) \ mod\ q \in \mathbb{Z}_q^{\ell(u+\lceil \log q \rceil)}$$

It can be easily verified that

$$\langle \boldsymbol{v}, \boldsymbol{w} \rangle = \left\langle \mathsf{BitDecomp}_{q,u}(\boldsymbol{v}), \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{w}) \right\rangle \ mod\ q$$

Let $\widetilde{\boldsymbol{D}}_i$s be the matrices got by applying $\mathsf{BitDecomp}_{q,u}$ on the columns of $\boldsymbol{D}_i$s. Now, instead of multiplying the $\boldsymbol{c}_i$s with the $\boldsymbol{D}_i$s, if we multiply the $\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i)$s with the respective $\widetilde{\boldsymbol{D}}_i$s, the corresponding $K_{i,j}$s are given as

$$K_{i,j} = \frac{1}{q}\left\langle \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i), \mathsf{BitDecomp}_{q,u}(\boldsymbol{D}_i(:,j)) \right\rangle - m_{i,j} \left\lfloor \frac{q}{2} \right\rfloor - e'_{i,j} \quad (3.45)$$

Therefore, their magnitudes can be computed as

$$\begin{aligned}
|K_{i,j}| &= \frac{1}{q} \cdot \left| \left\langle \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i), \mathsf{BitDecomp}_{q,u}(\boldsymbol{D}_i(:,j)) \right\rangle - m_{i,j} \left\lfloor \frac{q}{2} \right\rfloor - e'_{i,j} \right| \\
&\leq \frac{\left| \left\langle \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i), \widetilde{\boldsymbol{D}}_i(:,j) \right\rangle \right|}{q} + 1 \\
&\leq \frac{1}{2} \cdot \left\| \widetilde{\boldsymbol{D}}_i(:,j) \right\|_1 + 1 \\
&\leq \frac{1}{2} \cdot (\ell(u + \lceil \log q) \rceil)) + 1
\end{aligned}$$

$$= \mathcal{O}(n \log q) \text{ (Since } \ell \text{ is } \mathcal{O}(n) \text{ and } u \text{ is } \mathcal{O}(1)) \tag{3.46}$$

To accommodate these changes in the evaluation key, the tensor $\boldsymbol{\mathcal{M}}$ can be modified as:

$$\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{T}} \times_1 \widetilde{\boldsymbol{D}} \times_2 \widetilde{\boldsymbol{D}} \times_3 (\boldsymbol{R}^T \boldsymbol{Q}^T \boldsymbol{B}^T) \in \mathbb{Q}^{\ell(u+\lceil \log q \rceil) \times \ell(u+\lceil \log q \rceil) \times \ell} \tag{3.47}$$

Then, given $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, $\boldsymbol{c}_{mult}$ can be computed using the corresponding bilinear map $\mathcal{B}_{\boldsymbol{\mathcal{M}}} : \mathbb{Q}^{\ell(u+\lceil \log q \rceil)} \times \mathbb{Q}^{\ell(u+\lceil \log q \rceil)} \to \mathbb{Q}^{\ell}$ as

$$\boldsymbol{c}_{mult} = \lfloor \mathcal{B}_{\boldsymbol{\mathcal{M}}} (\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_1), \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_2)) \rfloor \ mod\ q \in \mathbb{Z}_q^{\ell} \tag{3.48}$$

Noise Magnitude

If the above mentioned vector decomposition techniques are used then $|K_{i,j}| \leq \mathcal{O}(n \log q)$ for $i \in \{1, 2\}$. Since, $\left|e'_{i,j}\right| \leq 2B$ for $i \in \{1, 2\}$ and $B < \lfloor q/2 \rfloor / 2 \leq q/4$, the magnitude of the error after multiplication is as follows:

$$\|\boldsymbol{e}_{mult}\|_{\infty} \leq 4B + 2(4B+1) \cdot \mathcal{O}(n \log q) + \frac{8B^2+1}{q} = \mathcal{O}(n \log q) \cdot B \tag{3.49}$$

Theorem 3.1.5. *The proposed scheme with parameters $n, q, L, \mathcal{X}$ with $|\mathcal{X}| \leq B$ can evaluate circuits of depth L when $q/B \geq (\mathcal{O}(n \log q))^L$.*

Proof. From Lemma 3.1.2, noise in a fresh encryption is at most B. After one level of multiplication, it increases to $\mathcal{O}(n \log q) \cdot B$. If e^i_{mult} denotes the noise at level i, then $|e^i_{mult}| = \mathcal{O}(n \log q) \cdot \left|e^{i-1}_{mult}\right|$. Therefore, $\left|e^L_{mult}\right| = (\mathcal{O}(n \log q))^L \cdot B$. For correctness of decryption, we need $\left|e^L_{mult}\right| \leq q/4$. Hence, $q/B \geq (\mathcal{O}(n \log q))^L$. □

3.1.6 Bootstrapping

Bootstrapping is the process of 'refreshing' a ciphertext by re-encrypting it under a different key and then homomorphically evaluating the decryption circuit on the inner encryption. The refreshed ciphertext is an encryption of the same message with reduced noise. Bootstrapping is no longer necessary to achieve a leveled FHE scheme. However, bootstrapping with the notion of "circular security" (secure against an adversary that has access to the encryptions of the secret key bits) is the only way to achieve "pure" FHE from a somewhat homomorphic scheme that can evaluate circuits of arbitrary depth. The bootstrapping theorem (Theorem 2.8.1) states that if the decryption circuit complexity of an L-homomorphic scheme is less than L, then there exists a leveled fully homomorphic encryption scheme. Moreover, if the scheme is circular secure, then there exists a fully homomorphic encryption scheme.

To apply the bootstrapping theorem, the decryption cicuit complexity must be bounded. Since, the decryption in the proposed scheme is similar to that in previous LWE-based schemes, we can use similar techniques as [BV14a] to bound its complexity. Therefore, the depth of the decryption circuit is of the order of $\mathcal{O}(\log n + \log\log q)$. Using the bootstrapping theorem, we get the following lemma.

Lemma 3.1.6. *The proposed scheme with parameters $n, q, L, \mathcal{X}$ with $|\mathcal{X}| \leq B$ and $q/B \geq (n \log q)^{\mathcal{O}(\log n + \log\log q)}$ is bootstrappable based on the $LWE_{n,q,\mathcal{X}}$ assumption.*

Moreover, if the scheme is circular secure, then there exists a fully homomorphic encryption scheme.

3.1.7 Parameters and Performance

Similar to [GSW13], we choose $n = \mathcal{O}(\lambda)$ to be a fixed parameter and $\ell = \mathcal{O}(n)$ to be slightly bigger than n such that $\ell - n = \mathcal{O}(1)$. The proposed scheme can evaluate a circuit of depth L as long as $q/B \geq (\mathcal{O}(n \log q))^L$. Therefore, we can choose q to be of bit size $\mathcal{O}(L \log n)$ similar to [BGV14, Bra12]. Gentry's bootstrapping theorem [Gen09] states that if the decryption circuit complexity of an L-homomorphic scheme is less than L, then there exists a leveled fully homomorphic encryption scheme. Decryption circuit complexity in the proposed scheme can be bounded by $\mathcal{O}(\log n + \log \log q)$ using similar techniques as in [BV14a]. For $L = \mathcal{O}(\log n)$, q/B in the proposed scheme is quasi-polynomial in n and its security is based on the hardness of LWE for quasi-polynomial factors given by $\gamma = n^{\mathcal{O}(\log n)}$ (since $\gamma = (q/B) \cdot \tilde{\mathcal{O}}(n)$).

The cost of multiplying two ciphertexts is of the order of $\mathcal{O}(\ell^3 \log^2 q) = \tilde{\mathcal{O}}(n^3 \cdot L^2)$ while that of adding two ciphertexts is $\mathcal{O}(\ell)$. Therefore, the per gate computation of the leveled FHE scheme is $\tilde{\mathcal{O}}(n^3 \cdot L^2)$.

We give a comparison of the per gate computation of the proposed scheme with previous schemes (when the underlying hardness assumption is LWE) in Table 3.1.

Scheme	Per gate computation
[BGV14, Bra12]	$\tilde{\mathcal{O}}(n^3 L^5)$
[GSW13]	$\tilde{\mathcal{O}}\left((nL)^{2.37}\right)$
Proposed scheme	$\tilde{\mathcal{O}}(n^3 L^2)$

Table 3.1: Per gate computation overhead of LWE-based schemes

3.2 Private key to Public key Conversion

The proposed scheme can be converted to a public key scheme as follows. For some $\epsilon > 0$, let $\boldsymbol{C}_0$ be a list of $d = (1+\epsilon)(\ell \log q)$ encryptions of the zero vector under the private key scheme explained earlier in the chapter. Let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\ell-n}$ be the standard basis for $\mathbb{Z}_q^{\ell-n}$, i.e., $\boldsymbol{b}_1 = (1, 0, 0, \ldots, 0), \boldsymbol{b}_2 = (0, 1, 0, \ldots, 0)$ and so on. Let $\boldsymbol{c}_{\boldsymbol{b}_1}, \ldots, \boldsymbol{c}_{\boldsymbol{b}_{\ell-n}}$ be the encryptions of these vectors using the private key scheme. Construct a matrix $\boldsymbol{C}_{pk} \in \mathbb{Z}_q^{(\ell-n)\times\ell}$ by assigning $\boldsymbol{C}_{pk}(i, :) = \boldsymbol{c}_{\boldsymbol{b}_i}$ for $1 \leq i \leq \ell - n$. Then, the public key is given by $pk = (\boldsymbol{C}_0, \boldsymbol{C}_{pk})$ and the secret key is same as that of the private key scheme. The public key scheme can be described in terms of the following algorithms.

- PK.KeyGen(1^λ): It takes the security parameter λ and outputs the public encryption key $pk = (\boldsymbol{C}_0, \boldsymbol{C}_{pk})$ and the secret decryption key $sk = \boldsymbol{S}_{\text{dec}}$ where $\boldsymbol{S}_{\text{dec}} = \boldsymbol{R}^{-1} \cdot \begin{bmatrix} \boldsymbol{S} & \boldsymbol{I}_{\ell-n} \end{bmatrix}^T$.

- PK.Encrypt$(pk, \boldsymbol{m})$: To encrypt a message $\boldsymbol{m} \in \{0,1\}^{\ell-n}$, select a random subset S of $\boldsymbol{C}_0$ and compute the ciphertext as

$$\boldsymbol{c} = \boldsymbol{m} \cdot \boldsymbol{C}_{pk} + \sum_{\boldsymbol{c}_i \in S} \boldsymbol{c}_i \ (mod\ q) \tag{3.50}$$

- PK.Decrypt$(sk, \boldsymbol{c})$: Decryption is performed by computing

$$\boldsymbol{m} = \left\lfloor \frac{1}{\lfloor q/2 \rfloor} (\boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}} \, mod\, q) \right\rceil \ mod\ 2 \tag{3.51}$$

If $\tilde{\boldsymbol{e}}_{pk} = (\boldsymbol{0}, \boldsymbol{e}_{pk})$ denotes the noise associated with the ciphertext $\boldsymbol{c}$, then

$$\boldsymbol{c} \cdot \boldsymbol{S}_{\text{dec}} = \left(\boldsymbol{m} \cdot \boldsymbol{C}_{pk} + \sum_{\boldsymbol{c}_i \in S} \boldsymbol{c}_i \right) \cdot \boldsymbol{S}_{\text{dec}} = \boldsymbol{m} \cdot \left\lfloor \frac{q}{2} \right\rceil + \boldsymbol{e}_{pk} \ (mod\, q) \tag{3.52}$$

Observe that $\|\boldsymbol{e}_{pk}\|_\infty \leq (wt(\boldsymbol{m}) + |S|) \cdot B$ where $wt(\boldsymbol{m})$ denotes the weight of the vector $\boldsymbol{m}$ and $|S|$ denotes the cardinality of the set S. Therefore, the decryption function outputs $\boldsymbol{m}$ when $(wt(\boldsymbol{m}) + |S|) \cdot B < q/4$. The security of the scheme follows from the security of the private key scheme and Claim 5.3 in [Reg09] (a special case of the leftover hash lemma). This claim is restated as follows

Lemma 3.2.1. [Claim 5.3 [Reg09]] *Let $S = \{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_d\}$ be some subset of $\mathbb{Z}_q^\ell$ for some $d \in \mathbb{N}$. Then, for a uniform choice of S, given a hash function $h_{\boldsymbol{A}} : \{0,1\}^d \to \mathbb{Z}_q^\ell$ defined as $h_{\boldsymbol{A}}(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{A}^T \bmod q$ where $\boldsymbol{A}(:, j) = \boldsymbol{g}_j$ for $1 \leq j \leq d$, the expectation of the statistical distance of the distribution on $\boldsymbol{x}\boldsymbol{A}^T \bmod q$ from uniform has an upper bound of $\sqrt{q^\ell/2^d}$. Further, the probability of this statistical distance being greater than $\sqrt[4]{q^\ell/2^d}$ is upper bounded by $\sqrt[4]{q^\ell/2^d}$.*

Now, for some $\epsilon > 0$, if $d = (1+\epsilon)\ell \log q$, then $\sqrt{q^\ell/2^d}$ and $\sqrt[4]{q^\ell/2^d}$ are negligible in ℓ.

Lemma 3.2.2. *For parameters n, ℓ, d and $\mathcal{X}$, if there exists an efficient algorithm that can distinguish between encryptions of any two distinct messages $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ under the above described public key scheme, then there exists a message $\boldsymbol{m}$ and an algorithm that can distinguish between encryptions of $\boldsymbol{m}$ under the private key scheme described in Section 3.1 and the uniform distribution on $\mathbb{Z}_q^\ell$*

Proof. For $0 \leq i \leq d + \ell - n$, let $\boldsymbol{P}_i$ be the matrix got by taking the first $d+i$ rows of pk ($\boldsymbol{P}_0 = \boldsymbol{C}_0$). Let $\mathcal{D}_i$ denote the set of vectors got by taking the sum of subsets of the rows of $\boldsymbol{P}_i$ i.e., $\mathcal{D}_i := \{\boldsymbol{x}\boldsymbol{P}_i : \boldsymbol{x} \in \{0,1\}^{d+i}\}$. Note that, for $j \leq i$, $\mathcal{D}_j \subset \mathcal{D}_i$. We claim that, for $0 \leq i \leq d$, there exists no efficient algorithm that can distinguish between vectors sampled from a uniform distribution on $\mathcal{D}_i$ and vectors sampled from a uniform distribution on $\mathbb{Z}_q^\ell$. We prove this claim using induction.

For a set S and an algorithm W, let $p_W(S)$ be the probability that W returns 1 when the input is sampled from a uniform distribution on S. As a consequence of Lemma 3.2.1, for any algorithm W that takes as input elements of $\mathbb{Z}_q^\ell$,

$$p_W(\mathcal{D}_0) - p_W(\mathbb{Z}_q^\ell) \leq 2^{-\omega(n)} \tag{3.53}$$

Assume that the claim is true for $i \leq k$. The $k+1$-th row of $\boldsymbol{P}_{k+1}$ is a random encryption of the message $\boldsymbol{b}_{k+1}$. Let $\mathcal{D}'_k$ be the following set of vectors,

$$\mathcal{D}'_k := \{\boldsymbol{P}_{k+1}(:, k+1) + \boldsymbol{v} \mid \boldsymbol{v} \in \mathcal{D}_k\} \tag{3.54}$$

Clearly $\mathcal{D}_{k+1} := \mathcal{D}_k \cup \mathcal{D}'_k$. Therefore, for any algorithm W

$$p_W(\mathcal{D}_{k+1}) = \frac{1}{2}p_W(\mathcal{D}_k) + \frac{1}{2}p_W(\mathcal{D}'_k) \tag{3.55}$$

Therefore,

$$p_W(\mathcal{D}_{k+1}) - p_W(\mathbb{Z}_q^\ell) = \frac{1}{2}((p_W(\mathcal{D}_k) - p_W(\mathbb{Z}_q^\ell)) + (p_W(\mathcal{D}'_k) - p_W(\mathbb{Z}_q^\ell))) \tag{3.56}$$

The term $(p_W(\mathcal{D}_k) - p_W(\mathbb{Z}_q^\ell))$ is negligible by the induction assumption. Therefore, if the LHS term in Equation 3.56 is non-negligible, then $(p_W(\mathcal{D}'_k) - p_W(\mathbb{Z}_q^\ell))$ must be non-negligible. Consider the distribution of vectors $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$ where $\boldsymbol{v}_2$ is sampled from a uniform distribution on $\mathcal{D}_k$. This distribution is uniform if $\boldsymbol{v}_1$ is sampled from the uniform distribution on $\mathbb{Z}_q^\ell$. On the other hand, if $\boldsymbol{v}_1$ is sampled uniformly at random from the set of encryptions of $\boldsymbol{b}_{k+1}$ then the distribution of $\boldsymbol{v}$ is uniform in $\mathcal{D}'_k$. Thus, if $(p_W(\mathcal{D}'_k) - p_W(\mathbb{Z}_q^\ell))$ is non-negligible, then the algorithm W can be used to distinguish between vectors that are sampled from the uniform distribution on the encryptions of $\boldsymbol{b}_{k+1}$ (under the private key scheme) and vectors that are sampled from the

uniform distribution on $\mathbb{Z}_q^\ell$. □

3.3 Summary

In this chapter, a leveled fully homomorphic encryption scheme which achieves additive and multiplicative homomorphism without key switching has been discussed. The security of the scheme depends on the hardness of the LWE problem. This scheme can be used to encrypt message vectors and the addition and multiplication is done bit wise. The computation cost per multiplication is $\tilde{\mathcal{O}}(n^3 \cdot L^2)$.

Chapter 4

Homomorphic Multiplication of LWE-based Schemes without Relinearization

In this chapter, we demonstrate that the multiplication technique used in Chapter 3 can be extended to other LWE based schemes such as the ones proposed in [BV14a, BGV14, Bra12]. With the proposed multiplication technique, the size of the ciphertext does not grow after multiplication. Further, the noise associated with it grows only linearly. Therefore, homomorphic multiplication can be performed without relinearization and modulus switching.

4.1 Homomorphic Multiplication in LWE-based schemes

In the FHE schemes proposed in [BV14a, BGV14, Bra12], homomorphic multiplication is performed by taking the tensor product of two ciphertexts. The evaluated ciphertext can be decrypted by a tensored secret key. The size of

the ciphertext as well as the noise associated with it grows quadratically with each multiplication.

In order to reduce the ciphertext size after multiplication, the process of relinearization is used [BV14a]. In these schemes, the secret key and the ciphertext are vectors of size n and $n+1$ respectively. The tensored ciphertext (of size $((n+1)\lceil \log q \rceil)^2$) is converted to a ciphertext of size $(n+1)$ by multiplying with a matrix of size $((n+1)\lceil \log q \rceil)^2 \times (n+1)$. This matrix is provided as the evaluation key for multiplication. Decryption is performed using a new shorter secret key. Therefore, this process involves a change of secret key after every multiplication. The evaluation key contains encryptions of the tensored secret key under the new key. In order to perform more than one multiplication, a chain of such keys must be provided. Therefore, to evaluate a circuit of depth L, the evaluation key must consist of L relinearization matrices. [BGV14].

To deal with the increase in noise due to multiplication, a noise management technique called modulus switching is used [BGV14]. It scales down the ciphertext after every multiplication such that the associated noise reduces by the same factor. As a result, the noise magnitude remains the same after every multiplication but the modulus keeps decreasing. It consists of choosing a decreasing chain of moduli and after every multiplication, the ciphertext with respect to a modulus q_1 is switched to a smaller modulus q_2 in the chain. A new tensoring technique without modulus switching was proposed in [Bra12]. Using this technique the associated noise grows only linearly. However, all of these schemes go through the process of relinearization and the associated change of key.

4.2 Regev's Cryptosystem

The building block of all LWE based FHE schemes is the cryptosystem of [Reg05]. We briefly discuss this scheme as described in [Bra12]. Let n be the security parameter and $q = q(n)$ be prime. Let $\mathcal{X}$ be a probability distribution on $\mathbb{Z}_q$ such that $|\mathcal{X}| \leq B$.

- $\mathsf{Regev.SecretKeyGen}(1^n)$: Sample $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$. Output $sk = \boldsymbol{s}$.
- $\mathsf{Regev.PublicKeyGen}(sk)$: Let $N \geq (n+1)\log q$. Sample a matrix $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{N \times n}$, a vector $\boldsymbol{e} \xleftarrow{\$} \mathcal{X}^N$ and compute $\boldsymbol{b} = \boldsymbol{s}\boldsymbol{A}^T + \boldsymbol{e} \ (mod\ q) \in \mathbb{Z}_q^N$. Let $\boldsymbol{P}$ be the matrix defined as $\boldsymbol{P} := [-\boldsymbol{A} \parallel \boldsymbol{b}^T] \in \mathbb{Z}_q^{N \times (n+1)}$. Output $pk = \boldsymbol{P}$.
- $\mathsf{Regev.Encrypt}(m, pk)$: Let $m \in \{0,1\}$ be the message to be encrypted using pk. Let $\boldsymbol{m} = (0, \ldots, 0, m) \in \{0,1\}^{n+1}$. Sample $\boldsymbol{r} \xleftarrow{\$} \{0,1\}^N$ and compute

$$\boldsymbol{c} = \boldsymbol{m}\left\lfloor \frac{q}{2} \right\rfloor + \boldsymbol{r} \cdot \boldsymbol{P} \ (mod\ q) \in \mathbb{Z}_q^{n+1} \tag{4.1}$$

- $\mathsf{Regev.Decrypt}(sk, \boldsymbol{c})$: Using sk, the message m can be recovered as

$$m = \left\lfloor \frac{2}{q} \left(\langle \boldsymbol{c}, (\boldsymbol{s}, 1) \rangle \ mod\ q \right) \right\rceil \ mod\ 2 \tag{4.2}$$

The correctness of the scheme is captured in the following lemma.

Lemma 4.2.1. *Let $q, n, N, |\mathcal{X}| \leq B$ be as described in the scheme. Let $sk \leftarrow$ Regev.SecretKeyGen(1^n), $pk \leftarrow$ Regev.PublicKeyGen(sk) and $\boldsymbol{c} \leftarrow$ Regev.Encrypt(m, pk). Then, for some e with $|e| \leq NB$, it holds that*

$\langle \boldsymbol{c}, (\boldsymbol{s}, 1) \rangle = \boldsymbol{m} \left\lfloor \frac{q}{2} \right\rfloor + e \pmod q$. *Further, if* $|e| \leq NB < \lfloor q/2 \rfloor /2$, *then* $\mathsf{Regev.Decrypt}(sk, \boldsymbol{c}) = m$.

4.3 The Proposed Multiplication Technique

Consider an LWE based encryption scheme with a ciphertext of size $n+1$. In this section, we show that ciphertexts of such a scheme can be seen as scaled evaluations of a polynomial from an ideal on a set of $n+1$ points. This property is used to homomorphically multiply two ciphertexts. Corresponding to every secret key, one can find a suitable ideal, set of points and scaling factor.

Given n, one can find integers v and r' such that $\binom{v+r'}{r'} = n$. A trivial example of this is $v = n-1$ and $r' = 1$. Consider the polynomial ring $\mathcal{R} := \mathbb{Z}_q[x_1, \ldots, x_v] / \langle x_1^q - x_1, \ldots, x_v^q - x_v \rangle$ and an integer $r > r'$. The elements of $\mathcal{R}_{\leq r}$, constitute a vector space of dimension $\binom{v+r}{r}$. Consider the ideal $\mathcal{J}$ generated by a polynomial $g \in \mathcal{R}$ having degree $r - r'$. The elements of $\mathcal{J}_{\leq r}$ form an n-dimensional subspace of $\mathcal{R}_{\leq r}$.

Choose $n+1$ distinct points $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{n+1})$ in $\mathbb{Z}_q^\ell$ such that evaluating polynomials in $\mathcal{J}_{\leq r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)$ spans the n-dimensional space $\mathbb{Z}_q^n$. In other words, every vector in $\mathbb{Z}_q^n$ can be obtained by evaluating a polynomial in $\mathcal{J}_{\leq r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)$. Then, evaluating polynomials in $\mathcal{J}_{\leq r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{n+1})$ spans an n-dimensional subspace of $\mathbb{Z}_q^{n+1}$, denoted by $\mathcal{S}_{\mathcal{J}_{\leq r}}$. Therefore, there exist $\{\alpha_1, \ldots, \alpha_n\} \in \mathbb{Z}_q$ such that for all polynomials $f \in \mathcal{J}_{\leq r}$ the following equation is satisfied.

$$f(\boldsymbol{z}_{n+1}) = \alpha_1 f(\boldsymbol{z}_1) + \cdots + \alpha_n f(\boldsymbol{z}_n) \tag{4.3}$$

For $\boldsymbol{s}' = (s_1, \ldots, s_n, 1) \in \mathbb{Z}_q^{n+1}$, let $\lambda_{s,i} = s_i^{-1} \alpha_i$ for $1 \leq i \leq n$. Let $\boldsymbol{\Lambda_s}$ be the following matrix.

$$\boldsymbol{\Lambda_s} = \begin{bmatrix} \lambda_{s,1} & & & \\ & \ddots & & \\ & & \lambda_{s,n} & \\ & & & -1 \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times(n+1)} \tag{4.4}$$

Let $f(\boldsymbol{Z})$ denote the vector $(f(\boldsymbol{z}_1), \ldots, f(\boldsymbol{z}_{n+1}))$. Then,

$$\langle f(\boldsymbol{Z}) \cdot \boldsymbol{\Lambda_s}, \boldsymbol{s}' \rangle = 0 \ mod \ q \tag{4.5}$$

Thus, given $(\boldsymbol{s}, 1) \in \mathbb{Z}_q^{n+1}$, we can find an ideal $\mathcal{J}$ such that scaled evaluations of polynomials in $\mathcal{J}_{\leq r}$ generate the n-dimensional space $(\boldsymbol{s}, 1)^{\perp}$. For any $\boldsymbol{A} \in \mathbb{Z}_q^{N\times n}$, each row of the matrix $\boldsymbol{A'} := \left[-\boldsymbol{A} \parallel \boldsymbol{A}\boldsymbol{s}^T\right]$, can be written as $\boldsymbol{A'}(i,:) = f^i(\boldsymbol{Z}) \cdot \boldsymbol{\Lambda_s}$ for some $f^i \in \mathcal{J}_{\leq d}$. Further, for any $\boldsymbol{r} \in \{0,1\}^N$, there exists $f \in \mathcal{J}_{\leq r}$ such that $\boldsymbol{r} \cdot \boldsymbol{A'} = f(\boldsymbol{Z}) \cdot \boldsymbol{\Lambda_s}$. Thus, a ciphertext in Regev's cryptosystem corresponding to the secret key $\boldsymbol{s}$ can be written as

$$\boldsymbol{c} = \boldsymbol{m} \left\lfloor \frac{q}{2} \right\rfloor + f(\boldsymbol{Z}) \cdot \boldsymbol{\Lambda_s} + \boldsymbol{e'} \ (mod \ q) \tag{4.6}$$

where $\boldsymbol{e'} := (0, \ldots, 0, \boldsymbol{r}\boldsymbol{e}^T) \in \mathbb{Z}_q^{n+1}$. This is similar to a ciphertext of the scheme proposed in Chapter 3 except that in this case, the ciphertext is a vector got by adding noisy 'scaled' evaluations of a polynomial in the ideal $\mathcal{J}$ with the scaled plaintext vector $\boldsymbol{m}$.

From Equation 3.15, we know that for $f_1, f_2 \in \mathcal{J}_{\leq r}$ and some $\boldsymbol{z} \in \mathbb{Z}_q^{\ell}$, $f_1(\boldsymbol{z}) \cdot f_2(\boldsymbol{z}) = f_1 f_2(\boldsymbol{z})$. Although the polynomial $f_1 f_2$ lies in the ideal $\mathcal{J}$, it does not necessarily lie in the subspace $\mathcal{J}_{\leq r}$. Consequently, the vector $f_1 f_2(\boldsymbol{Z})$ need not lie in $\mathcal{S}_{\mathcal{J}_{\leq r}}$. Instead, $f_1 f_2$ lies in $\mathcal{J}_{\leq 2r}$ which is a vector subspace of $\mathcal{R}_{\leq 2r}$. Let the dimension of this subspace be n'.

We now choose $(n' - n)$ additional points $(\boldsymbol{z}_{n+2}, \ldots, \boldsymbol{z}_{n'+1})$ in $\mathbb{Z}_q^{\ell}$ such that

evaluating polynomials in $\mathcal{J}_{\leq 2r}$ at $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n, \boldsymbol{z}_{n+2}, \ldots, \boldsymbol{z}_{n'+1})$ spans the space $\mathbb{Z}_q^{n'}$. Let the set of points $(\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{n'+1})$ be denoted by $\boldsymbol{Z}'$. The evaluations of polynomials in $\mathcal{J}_{\leq 2r}$ on $\boldsymbol{Z}'$ constitute an n' dimensional subspace of $\mathbb{Z}_q^{n'+1}$. Let this subspace be denoted by $\mathcal{S}_{\mathcal{J}_{\leq 2r}}$. Because of the way in which the set $\boldsymbol{Z}'$ is chosen, there exist constants $\beta_1, \beta_2, \ldots, \beta_n, \beta_{n+2}, \ldots, \beta_{n'+1}$ such that for any $f \in \mathcal{J}_{\leq 2r}$,

$$f(\boldsymbol{z}_{n+1}) = \sum_{i=1}^{n} \beta_i \cdot f(\boldsymbol{z}_i) + \sum_{i=n+2}^{n'+1} \beta_i \cdot f(\boldsymbol{z}_i). \tag{4.7}$$

Further, there exist constants γ_i^j for $n+2 \leq i \leq n'+1$ and $1 \leq j \leq n$ such that for all $f \in \mathcal{J}_{\leq r}$,

$$f(\boldsymbol{z}_i) = \sum_{j=1}^{n} \gamma_i^j \lambda_{s,j} \cdot f(\boldsymbol{z}_j) \text{ for } n+2 \leq i \leq n'+1. \tag{4.8}$$

A linear map from $\mathcal{S}_{\mathcal{J}_{\leq 2r}}$ to $\mathcal{S}_{\mathcal{J}_{\leq r}}$ can be represented by a matrix in $\mathbb{Z}_q^{(n+1)\times(n'+1)}$. This follows from the discussion in Step 5 of the multiplication procedure in Chapter 3. Therefore, the choice of points $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{n'+1}$ enables us to find a similar map where the corresponding matrix has the following form.

$$\boldsymbol{L} = \left[\begin{array}{c|c} \boldsymbol{L}_1^{n\times n} & \boldsymbol{L}_3^{n\times 1} \\ \hline \boldsymbol{0}^{1\times n} & 1 \\ \hline \boldsymbol{L}_2^{(n'-n)\times n} & \boldsymbol{L}_4^{(n'-n)\times 1} \end{array}\right] \in \mathbb{Z}_q^{(n'+1)\times(n+1)} \tag{4.9}$$

This matrix is analogous to the matrix $\boldsymbol{Q}$ in Equation 3.38.

For $i = \{1, 2\}$, a ciphertext $\boldsymbol{c}_i$ can be written as:

$$\boldsymbol{c}_i = \begin{bmatrix} \lambda_{s,1} f_i(\boldsymbol{z}_1) \\ \vdots \\ \lambda_{s,n} f_i(\boldsymbol{z}_n) \\ m_i \left\lfloor \frac{q}{2} \right\rfloor - f_i(\boldsymbol{z}_{n+1}) + \boldsymbol{r}_i \boldsymbol{e}_i^T \end{bmatrix}^T \quad (mod\ q) \tag{4.10}$$

Given encryptions, $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, of two binary data bits m_1 and m_2, our aim is to calculate an encryption of $m_1 m_2$ without revealing either the secret key or the plaintext bits. Firstly, an encryption of $m_1 m_2$ can be calculated using the multiplication procedure in Chapter 3 with a few modifications. We discuss this procedure in the following steps. Note that these steps are done considering the entries of the concerned vectors to be rational numbers. The resultant vector is converted back to $\mathbb{Z}_q$ at the end.

1. For $i = \{1, 2\}$, transform $\boldsymbol{c}_i$ to $\boldsymbol{c}_i'$ given by

$$\boldsymbol{c}_i' = \begin{bmatrix} \lambda_{s,1} f_i(\boldsymbol{z}_1) \\ \vdots \\ \lambda_{s,n} f_i(\boldsymbol{z}_n) \\ m_i \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_i + qK_i \end{bmatrix}^T \tag{4.11}$$

   Note that the noise term $\boldsymbol{r}_i \boldsymbol{e}_i^T$ changes to $\tilde{e}_i$. This is because additional noise has been added for similar reasons discussed in Chapter 3.

2. Compute and append evaluations of f_1 and f_2 at the additional points $\boldsymbol{z}_{n+2}, \ldots, \boldsymbol{z}_{n'+1}$ to $\boldsymbol{c}_1'$ and $\boldsymbol{c}_2'$. Let the resultant vectors be denoted as $\boldsymbol{c}_1^{\text{ext}}$ and $\boldsymbol{c}_2^{\text{ext}}$ respectively.

3. Take component-wise product of $\boldsymbol{c}_1^{\text{ext}}$ and $\boldsymbol{c}_2^{\text{ext}}$ to get

$$\boldsymbol{c}_{\text{mult}} = \begin{bmatrix} f_1 f_2(\boldsymbol{z}_1) \\ \vdots \\ f_1 f_2(\boldsymbol{z}_n) \\ \frac{2}{q}\left(m_1 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_1 + qK_1\right)\left(m_2 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_2 + qK_2\right) \\ f_1 f_2(\boldsymbol{z}_{n+2}) + qK_{n+2} \\ \vdots \\ f_1 f_2(\boldsymbol{z}_{n'+1}) + qK_{n'+1} \end{bmatrix}^T \tag{4.12}$$

4. Add an integer equivalent to $f_1 f_2(\boldsymbol{z}_{n+1})$ *mod* q to the $(n+1)^{th}$ entry of $\boldsymbol{c}_{\text{mult}}$.

5. The vector obtained in Step 4 is of order $n'+1$ over $\mathbb{Q}$. Transform this to a vector of order $n+1$ over $\mathbb{Q}$. Let the resultant vector be $\tilde{\boldsymbol{c}}$.

6. Transform $\tilde{\boldsymbol{c}}$ to a valid ciphertext of order $n+1$ over $\mathbb{Z}_q$.

Step 1: Convert the ciphertexts to vectors of the form

$$\boldsymbol{c}_i' = \begin{bmatrix} \lambda_{s,1} f_i(\boldsymbol{z}_1) \\ \vdots \\ \lambda_{s,n} f_i(\boldsymbol{z}_n) \\ m_i \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_i + qK_i \end{bmatrix}^T \text{ for } i = 1, 2 \tag{4.13}$$

where $K_i \in \mathbb{Z}$ for $i \in \{1, 2\}$. This is done by multiplying the ciphertexts with the matrix

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{I}_n & (\boldsymbol{s} + \boldsymbol{\epsilon}_i)^T \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{4.14}$$

i.e., $\boldsymbol{c}'_i = \boldsymbol{c}_i \cdot \boldsymbol{S}$. The $\boldsymbol{\epsilon}_i$s are added deliberately for similar reasons explained in Section 3.1.4. In this case, these are randomly chosen vectors in $\mathbb{Q}^n$ whose one norms are bounded by $\frac{KB}{q}$. The integer K depends on the desired number of levels of multiplication. The resulting noise term is given by $\tilde{e}_i = \boldsymbol{r}_i \boldsymbol{e}_i^T + \langle \boldsymbol{c}_i, (\boldsymbol{\epsilon}_i, 0) \rangle$. Therefore, $\tilde{e}_i$ is bounded by $(N + K)B$. (Recall that in the proposed FHE scheme, the addition of these noise terms yields a total error of magnitude at most $2B$).

Step 2: Calculate values that are equivalent mod q to $f_1(\boldsymbol{z}_j)$ and $f_2(\boldsymbol{z}_j)$, for $n+2 \leq j \leq n'+1$ using Equation 4.8 and append these entries to $\boldsymbol{c}'_1$ and $\boldsymbol{c}'_2$ to generate vectors $\boldsymbol{c}_1^{\text{ext}}$ and $\boldsymbol{c}_2^{\text{ext}}$. These values can be written as $f_i(\boldsymbol{z}_j) + qK_{i,j}$ for $i \in \{1, 2\}$ and $n + 2 \leq j \leq n' + 1$ as shown in Equation 4.16 for some $K_{i,j} \in \mathbb{Z}$. This operation can be performed as $\boldsymbol{c}_i^{\text{ext}} = \boldsymbol{c}'_i \cdot \boldsymbol{U}$, for $i \in \{1, 2\}$ where,

$$\boldsymbol{U} = \left[\begin{array}{c|ccc} & \gamma_{n+2}^1 & \cdots & \gamma_{n'+1}^1 \\ \boldsymbol{I}_{n+1} & \vdots & \ddots & \vdots \\ & \gamma_{n+2}^n & \cdots & \gamma_{n'+1}^n \\ & 0 & \cdots & 0 \end{array}\right] \quad \text{and} \tag{4.15}$$

$$\boldsymbol{c}_i^{\text{ext}} = \begin{bmatrix} \lambda_{s,1} f_i(\boldsymbol{z}_1) \\ \vdots \\ \lambda_{s,n} f_i(\boldsymbol{z}_n) \\ m_i \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_i + qK_i \\ f_i(\boldsymbol{z}_{n+2}) + qK_{i,n+2} \\ \vdots \\ f_i(\boldsymbol{z}_{n'+1}) + qK_{i,n'+1} \end{bmatrix}^T \quad \text{for } i = 1, 2 \tag{4.16}$$

Step 3: Take an element wise product of $\boldsymbol{c}_1^{\text{ext}}$ and $\boldsymbol{c}_2^{\text{ext}}$. Multiply the first

n entries by integers that are equivalent *mod* q to the inverse of the square of the corresponding scaling factors i.e. for $1 \leq i \leq n$ multiply the i^{th} entry by an integer which is equivalent *mod* q to the inverse of $\lambda_{s,i}^2$ in $\mathbb{Z}_q$. Let the resultant vector be denoted by $\boldsymbol{c}_{\text{mult}}$ where

$$\boldsymbol{c}_{\text{mult}} = \begin{bmatrix} f_1 f_2(\boldsymbol{z}_1) \\ \vdots \\ f_1 f_2(\boldsymbol{z}_n) \\ \frac{2}{q}\left(m_1 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_1 + qK_1\right)\left(m_2 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_2 + qK_2\right) \\ f_1 f_2(\boldsymbol{z}_{n+2}) + qK_{n+2} \\ \vdots \\ f_1 f_2(\boldsymbol{z}_{n'+1}) + qK_{n'+1} \end{bmatrix}^T \tag{4.17}$$

Step 4: Add an integer which is equivalent to $f_1 f_2(\boldsymbol{z}_{n+1})$ *mod* q to the $(n+1)^{th}$ entry of $\boldsymbol{c}_{\text{mult}}$. This is done by multiplying $\boldsymbol{c}_{\text{mult}}$ with the following matrix (as a consequence of Equation 4.7).

$$\boldsymbol{B} = \left[\begin{array}{c|c|c} & \beta_1 & \\ \boldsymbol{I}_n & \vdots & \boldsymbol{0} \\ & \beta_n & \\ \hline \boldsymbol{0} & 1 & \boldsymbol{0} \\ \hline & \beta_{n+2} & \\ \boldsymbol{0} & \vdots & \boldsymbol{I}_{n'-n} \\ & \beta_{n'+1} & \end{array}\right] \tag{4.18}$$

Therefore,

$$\boldsymbol{c}_{\text{mult}} \cdot \boldsymbol{B} = \begin{bmatrix} f_1 f_2(\boldsymbol{z}_1) \\ \vdots \\ f_1 f_2(\boldsymbol{z}_n) \\ \frac{2}{q}\left(m_1 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_1 + qK_1\right)\left(m_2 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_2 + qK_2\right) + f_1 f_2(\boldsymbol{z}_{n+1}) + qK_{n+1} \\ f_1 f_2(\boldsymbol{z}_{n+2}) + qK_{n+2} \\ \vdots \\ f_1 f_2(\boldsymbol{z}_{n'+1}) + qK_{n'+1} \end{bmatrix}^T \tag{4.19}$$

Step 5: The vector obtained in Step 4 is then multiplied with the matrix $\boldsymbol{L\Lambda_s}$ where $\boldsymbol{L}$ is the matrix described in Equation 4.9 (the multiplication is done considering the elements of $\boldsymbol{L}$ to be rational numbers and not elements of $\mathbb{Z}_q$). Let the resultant vector be denoted by $\tilde{\boldsymbol{c}}$. Then, for some $K'_j \in \mathbb{Z}$, $1 \leq j \leq n+1$

$$\begin{aligned} \tilde{\boldsymbol{c}} &= \boldsymbol{c}_{\text{mult}} \cdot \boldsymbol{BL\Lambda_s} \\ &= \begin{bmatrix} \lambda_{s,1} \cdot f_{mult}(\boldsymbol{z}_1) + qK'_1 \\ \vdots \\ \lambda_{s,n} \cdot f_{mult}(\boldsymbol{z}_n) + qK'_n \\ \frac{2}{q}\left(m_1 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_1 + qK_1\right)\left(m_2 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_2 + qK_2\right) - f_{mult}(\boldsymbol{z}_{n+1}) + qK'_{n+1} \end{bmatrix}^T \end{aligned} \tag{4.20}$$

where f_{mult} is the polynomial obtained after the quotienting operation described in Chapter 3.

Step 6: Calculate the vector $\boldsymbol{c}'_{\text{prod}} = \lfloor \tilde{\boldsymbol{c}} \rceil \ mod\ q \in \mathbb{Z}_q^{n+1}$. Then,

$$\boldsymbol{c}'_{\text{prod}} = \begin{bmatrix} \lambda_{s,1} \cdot f_{mult}(\boldsymbol{z}_1) \\ \vdots \\ \lambda_{s,n} \cdot f_{mult}(\boldsymbol{z}_n) \\ \frac{2}{q}\left(m_1 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_1 + qK_1\right)\left(m_2 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_2 + qK_2\right) - f_{mult}(\boldsymbol{z}_{n+1}) \end{bmatrix}^T \tag{4.21}$$

We shall soon see that the vector $\boldsymbol{c}'_{\text{prod}}$ is an encryption of $m_1 m_2$ provided the constants K_1 and K_2 in Step 1 are sufficiently small. But before that we describe the process of generating an evaluation key for homomorphic multiplication. Steps 1 to 5 constitute a bilinear map, $\mathcal{B}_{\boldsymbol{\mathcal{M}}} : \mathbb{Q}^{n+1} \times \mathbb{Q}^{n+1} \rightarrow \mathbb{Q}^{n+1}$ which can be represented by a 3-way tensor $\boldsymbol{\mathcal{M}} \in \mathbb{Q}^{(n+1)\times(n+1)\times(n+1)}$ i.e.

$$\mathcal{B}_{\boldsymbol{\mathcal{M}}}(\boldsymbol{c}_1, \boldsymbol{c}_2) = \tilde{\mathbf{c}} = \left[\boldsymbol{c}_1 \boldsymbol{M}_1 \boldsymbol{c}_2^T \ \ldots \ \boldsymbol{c}_1 \boldsymbol{M}_{n+1} \boldsymbol{c}_2^T\right] \tag{4.22}$$

where $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_{n+1}$ are the frontal slices of the Tensor $\boldsymbol{\mathcal{M}}$. The tensor $\boldsymbol{\mathcal{M}}$ is given by

$$\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{N}} \times_1 (\boldsymbol{SU}) \times_2 (\boldsymbol{SU}) \times_3 \boldsymbol{BL\Lambda_s} \tag{4.23}$$

where the tensor $\boldsymbol{\mathcal{N}} \in \mathbb{Q}^{(n'+1)\times(n'+1)\times(n'+1)}$ represents the multiplication that happens in Step 3. It is obtained by assigning $\boldsymbol{\mathcal{N}}(i_1, i_1, i_1) = (\lambda_{s,i_1}^{-1})^2 \ \forall i_1 \neq n+1$, $\boldsymbol{\mathcal{N}}(n+1, n+1, n+1) = \frac{2}{q}$ and $\boldsymbol{\mathcal{N}}(i_1, i_2, i_3) = 0$ everywhere else. However, the $(n+1)^{th}$ frontal slice of $\boldsymbol{\mathcal{M}}$ completely reveals the secret key. It can be easily verified that the $(n+1)^{th}$ row and the $(n+1)^{th}$ column of $\boldsymbol{M}_{n+1}$ contain the entries of the secret key multiplied by $\frac{2}{q}$. As a result, the Tensor $\boldsymbol{\mathcal{M}}$ cannot be made public. In order to hide the secret key, consider three matrices

$$\boldsymbol{T}_i = \begin{bmatrix} \boldsymbol{I}_n & \mathbf{0}^T \\ \boldsymbol{t}_i & 1 \end{bmatrix} \in \mathbb{Q}^{(n+1)\times(n+1)} \text{ for } i = 1, 2, 3 \tag{4.24}$$

The $\boldsymbol{t}_i$s in these matrices are vectors with integer entries that are randomly chosen from $\boldsymbol{s}^{\perp}$. Calculate the tensor

$$\boldsymbol{\mathcal{M}}' = \boldsymbol{\mathcal{M}} \times_1 \boldsymbol{T}_1 \times_2 \boldsymbol{T}_2 \times_3 \boldsymbol{T}_3$$

$$= \boldsymbol{\mathcal{N}} \times_1 (\boldsymbol{T}_1 \boldsymbol{S} \boldsymbol{U}) \times_2 (\boldsymbol{T}_2 \boldsymbol{S} \boldsymbol{U}) \times_3 \boldsymbol{B} \boldsymbol{L} \boldsymbol{\Lambda}_s \boldsymbol{T}_3 \tag{4.25}$$

The Tensor $\boldsymbol{\mathcal{M}}'$ acts as the evaluation key. The secret key does not explicitly appear in $\boldsymbol{\mathcal{M}}'$. Observe that, $\boldsymbol{\mathcal{M}}'$ acting on a pair $\boldsymbol{c}_1, \boldsymbol{c}_2$ is equivalent to $\boldsymbol{\mathcal{M}}$ acting on the pair $\boldsymbol{c}_1\boldsymbol{T}_1, \boldsymbol{c}_2\boldsymbol{T}_2$ and the result then being multiplied by $\boldsymbol{T}_3$ i.e. $\mathcal{B}_{\boldsymbol{\mathcal{M}}'}(\boldsymbol{c}_1, \boldsymbol{c}_2) = \mathcal{B}_{\boldsymbol{\mathcal{M}}}(\boldsymbol{c}_1\boldsymbol{T}_1, \boldsymbol{c}_2\boldsymbol{T}_2) \cdot \boldsymbol{T}_3$. Note that the $\boldsymbol{T}_i$s map encryptions of a data bit m to vectors that are equivalent $mod\ q$ to other encryptions of m. Thus, if $\lfloor \mathcal{B}_{\boldsymbol{\mathcal{M}}}(\boldsymbol{c}_1, \boldsymbol{c}_2) \rfloor\ mod\ q$ is an encryption of $m_1 m_2$, then so is $\lfloor \mathcal{B}_{\boldsymbol{\mathcal{M}}'}(\boldsymbol{c}_1, \boldsymbol{c}_2) \rfloor\ mod\ q$. Let $\boldsymbol{c}_{\text{prod}} := \lfloor \mathcal{B}_{\boldsymbol{\mathcal{M}}'}(\boldsymbol{c}_1, \boldsymbol{c}_2) \rfloor\ mod\ q$.

Note that while $\boldsymbol{S}$ and the $\boldsymbol{T}_i$s completely depend on the secret key, the matrices $\boldsymbol{U}$ and $\boldsymbol{B}$ depend on the choice of the points $\boldsymbol{z}_{n+2}, \ldots, \boldsymbol{z}_{n'+1}$ which are randomly chosen. Thus, the secret key can be seen as being 'masked' by these entries.

4.3.1 Security

The elements of $\boldsymbol{\mathcal{M}}'$ are polynomials in the entries of the matrices $\boldsymbol{L}$, $\boldsymbol{U}$, $\boldsymbol{S}$, $\boldsymbol{B}$ and the $\boldsymbol{T}_i$s. Further, $\langle \boldsymbol{L}, \boldsymbol{s}' \rangle = 0$ and $\langle \boldsymbol{t}_i, \boldsymbol{s} \rangle = 0$. Therefore, given $\boldsymbol{\mathcal{M}}'$, we have a system of $\mathcal{O}(n^3)$ polynomial equations in $(2n'(n+1) + 4n - n^2 - 2)$ variables. The degree of these polynomials range from 2 to 5. Solving a system of polynomial equations (PoSSo) is known to be NP complete in general. Further, this system of equations is under-determined.

As discussed in Section 3.1.4 of Chapter 3, the multiplication operation defines an 'almost' bilinear map from $\mathbb{Z}_q^{n+1} \times \mathbb{Z}_q^{n+1}$ to $\mathbb{Z}_q^{n+1}$ (The rounding operations induce some non-linearity). In the absence of the $\boldsymbol{\epsilon}_i$s, the noiseless encryptions of 0 constitute an invariant subspace of this map. One could generate invariant subspaces of the multiplication process by randomly choosing a vector and repeatedly 'multiplying' the vector with itself using the evaluation

key. Further, the direct sum of such subspaces will also be invariant. This process could be used to generate subspaces of co-dimension 1. Each such subspace will correspond to a secret key candidate. This candidate can then be easily verified using other encryptions of zero. Further, the probability of choosing a vector in $(\boldsymbol{s}')^{\perp}$ is $\frac{1}{q}$. Therefore, it would take an average of only q random choices to get a vector in $(\boldsymbol{s}')^{\perp}$. Although, it hasn't been proven that this process converges in polynomial time, it does pose a potential vulnerability. This is countered by introducing the $\boldsymbol{\epsilon}_i$s. The $\boldsymbol{\epsilon}_i$s insert noise in the multiplication process. As a result, when two vectors in $(\boldsymbol{s}')^{\perp}$ are multiplied, the result is no longer in $(\boldsymbol{s}')^{\perp}$. Thus, the invariance is lost.

4.3.2 Correctness of Multiplication and Noise Analysis

The vector $\mathcal{B}_{\mathcal{M}'}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ can be seen as a sum of two vectors $\boldsymbol{v}$ and $\boldsymbol{v}'$ where $\boldsymbol{v}$ is a vector with integer entries which is equivalent *mod* q to a vector in $(\boldsymbol{s}')^{\perp}$ and $\boldsymbol{v}'$ is a vector of the form $(0, 0, \ldots, 0, \eta)$ where η is given as follows.

$$\begin{aligned}\eta &= \frac{2}{q}\left(m_1\left\lfloor\frac{q}{2}\right\rfloor + \tilde{e}_1 + qK_1\right)\left(m_2\left\lfloor\frac{q}{2}\right\rfloor + \tilde{e}_2 + qK_2\right) \\ &= m_1m_2\left\lfloor\frac{q}{2}\right\rfloor - \frac{m_1m_2}{2q} + \frac{q-1}{q}(m_1\tilde{e}_2 + m_2\tilde{e}_1) + (2\tilde{e}_1 - m_1)K_2 \\ &\quad + (2\tilde{e}_2 - m_2)K_1 + \frac{2}{q}\tilde{e}_1\tilde{e}_2 + q(m_1K_2 + m_2K_1 + 2K_1K_2))\end{aligned}$$

Therefore,

$$\langle \lfloor \mathcal{B}_{\mathcal{M}'}(c_1, c_2) \rfloor, \boldsymbol{s}' \rangle \ \ mod\ q = \lfloor \eta \rfloor \ mod\ q \tag{4.26}$$

Thus, if the magnitude of the term $e_{\text{mult}} = \frac{q-1}{q}(m_1\tilde{e}_2 + m_2\tilde{e}_1) + (2\tilde{e}_1 - m_1)K_2 + (2\tilde{e}_2 - m_2)K_1 + \frac{2}{q}\tilde{e}_1\tilde{e}_2 - \frac{m_1m_2}{2q}$ is less than $\left\lfloor\frac{q}{4}\right\rfloor$, the ciphertext $\boldsymbol{c}_{\text{prod}}$ yields m_1m_2 on decryption.

Observe that the most significant term in e_{mult} is $(2\tilde{e}_1 - m_1)K_2 + (2\tilde{e}_2 - m_2)K_1$. Now, the terms K_1 and K_2 are generated due to the multiplication

of $\boldsymbol{s}'$ with the $\boldsymbol{c}_i$s in Step 1 of the multiplication procedure. (These values are not affected by the introduction of the $\boldsymbol{T}_i$s). The values of $|K_1|, |K_2|$ are bounded by the one norm of $\boldsymbol{s}'$ ($\boldsymbol{s}'$ is seen as an integer vector for calculating one norm). Clearly, by appropriately choosing the secret key, the magnitude of e_{mult} can be controlled. However, if $\boldsymbol{s}'$ is arbitrarily chosen then its one norm can be $\mathcal{O}(nq)$. Note that the $\tilde{e}_i$s are bounded by $(N+K)B$ which is $\mathcal{O}(n \log q)B$. Thus, e_{mult} is bounded by $\mathcal{O}(n^2 q \log q)B$. Alternatively, we can use the modified version of the vector decomposition technique described in Chapter 3. We restrict the choice of $\boldsymbol{\epsilon}_i$s to vectors with positive entries of the form $\sum_{j=1}^{u} \frac{1}{2^j}\boldsymbol{x}_{-j}$ where $\boldsymbol{x}_{-j} \in \{0,1\}^n$, for a suitable choice of u (This will depend on the value of K and the permissible size of the evaluation key).

In this case, the function $\mathsf{BitDecomp}_{q,u}(\boldsymbol{v})$ takes a vector $\boldsymbol{v}$ of size $n+1$ over $\mathbb{Q}$ and outputs the vector

$$\left(\boldsymbol{x}_{-u}, \ldots, \boldsymbol{x}_0, \ldots, \boldsymbol{x}_{\lfloor \log q \rfloor}\right) \in \{0,1\}^{(n+1)(u+\lceil \log q \rceil)}$$

for $\boldsymbol{x}_i \in \{0,1\}^{n+1}$ such that $\sum_{i=-u}^{\lfloor \log q \rfloor} 2^i \cdot \boldsymbol{x}_i$ most closely approximates $\boldsymbol{v} \ \ mod\ q$. Similarly, the function $\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{w})$ takes a vector $\boldsymbol{w} \in \mathbb{Z}^{(n+1)}$ and outputs the vector

$$\left(2^{-u}\boldsymbol{w}, \ldots, 2^{-1}\boldsymbol{w}, \boldsymbol{w}, 2 \cdot \boldsymbol{w}, \ldots, 2^{\lfloor \log q \rfloor} \cdot \boldsymbol{w}\right) \ mod\ q$$

Note that, if $\boldsymbol{v} \equiv \sum_{i=-u}^{\lfloor \log q \rfloor} 2^i \boldsymbol{x}_i \, mod\, q$ for some $\left(\boldsymbol{x}_{-u}, \ldots, \boldsymbol{x}_0, \ldots, \boldsymbol{x}_{\lfloor \log q \rfloor}\right) \in \{0,1\}^{(n+1)(u+\lceil \log q \rceil)}$, then $\langle \boldsymbol{v}, \boldsymbol{w} \rangle \equiv \left\langle \mathsf{BitDecomp}_{q,u}(\boldsymbol{v}), \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{w}) \right\rangle \, mod\, q$. Further, the magnitude of $\left\langle \mathsf{BitDecomp}_{q,u}(\boldsymbol{v}), \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{w}) \right\rangle$ is upper bounded by $\frac{(n+1)q}{2}(u + \lceil \log q \rceil)$. (This is because each of the $(n+1)(u+\lceil \log q \rceil)$ entries of $\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{w})$ are bounded by $\left\lfloor \frac{q}{2} \right\rfloor$).

This technique can be incorporated in the multiplication procedure as follows. Convert each of the ciphertext vectors to vectors in $\mathbb{Q}^{(n+1)(u+\lceil \log q \rceil)}$

through the action of the function $\mathsf{PowersOfTwo}_{q,u}$ i.e. evaluate $\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i)$ for $i \in [1,2]$. Generate matrices $\boldsymbol{\mathcal{K}}_1, \boldsymbol{\mathcal{K}}_2 \in \mathbb{Q}^{(n'+1)\times(n+1)(u+\lceil \log q \rceil)}$ by the action of the $\mathsf{BitDecomp}_{q,u}$ on the columns of the matrices $\boldsymbol{T}_1\boldsymbol{SU}$ and $\boldsymbol{T}_2\boldsymbol{SU}$ respectively. Generate the Tensor

$$\boldsymbol{\mathcal{M}}_{evk} = \boldsymbol{\mathcal{N}} \times_1 \boldsymbol{\mathcal{K}}_1 \times_2 \boldsymbol{\mathcal{K}}_2 \times_3 \boldsymbol{BL\Lambda}_s\boldsymbol{T}_3 \tag{4.27}$$

where $\boldsymbol{\mathcal{M}}_{evk} \in \mathbb{Q}^{(n+1)(u+\lceil \log q \rceil)\times(n+1)(u+\lceil \log q \rceil)\times(n+1)}$. Evaluate $\mathcal{B}_{\boldsymbol{\mathcal{M}}_{evk}}(\mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_1), \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_2))$. Note that, if $\tilde{\boldsymbol{c}}_i = \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i)$ for $i \in \{1,2\}$, then the $(n+1)^{th}$ entries of $\tilde{\boldsymbol{c}}_i\boldsymbol{\mathcal{K}}_i$s are equal to the respective $\left\langle \mathsf{PowersOfTwo}_{q,u}(\boldsymbol{c}_i), \mathsf{BitDecomp}_{q,u}(\boldsymbol{s}') \right\rangle$s. These are of the form $m_i \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_i + q\tilde{K}_i$ and the magnitude of these terms are $\mathcal{O}(nq\,(u+\log q))$. Consequently, the magnitude of the $\tilde{K}_i$s are $\mathcal{O}(n\,(u+\log q))$. Further,

$$\begin{aligned}
&\langle \lfloor \mathcal{B}_{\boldsymbol{\mathcal{M}}_{evk}}\left(\mathsf{PowersOfTwo}_q(c_1), \mathsf{PowersOfTwo}_q(c_2)\right) \rfloor, \boldsymbol{s}' \rangle \; mod\, q \\
&= \left\lfloor \frac{2}{q}(m_1 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_1 + q\tilde{K}_1)(m_2 \left\lfloor \frac{q}{2} \right\rfloor + \tilde{e}_2 + q\tilde{K}_2) \right\rfloor \; mod\, q \\
&= \left(m_1 m_2 \left\lfloor \frac{q}{2} \right\rfloor + e'_{\text{mult}} \right) \; mod\, q
\end{aligned} \tag{4.28}$$

where $e'_{\text{mult}} = \lfloor (\frac{q-1}{q}(m_1\tilde{e}_2 + m_2\tilde{e}_1) + (2\tilde{e}_1 - m_1)\tilde{K}_2 + (2\tilde{e}_2 - m_2)\tilde{K}_1 + \frac{2}{q}\tilde{e}_1\tilde{e}_2 - \frac{m_1 m_2}{2q}) \rfloor$. Since the magnitude of the $\tilde{K}_i$s are $\mathcal{O}(n(u+\log q)$, the magnitude of e'_{mult} is $\mathcal{O}(n\log q)^2 B$. (Since u is $\mathcal{O}(1)$).

Observe that, the initial noise in a ciphertext is $E_0 = NB = \mathcal{O}(n\log q)B$. If E_i denotes the noise after the i^{th} multiplication, then $E_i = \mathcal{O}(n\log q) \cdot E_{i-1}$. Therefore, after L levels of multiplication, the noise is $(\mathcal{O}(n\log q))^{L+1}\, B$. Since, $B < \lfloor q/2 \rfloor /2$, the scheme can evaluate circuits of depth L if $q/B \geq (\mathcal{O}(n\log q))^{L+1}$.

4.3.3 Parameters and Performance

The parameters can be chosen similarly to previous LWE-based schemes [BGV14, Bra12]. For $q/B \geq (\mathcal{O}(n \log q))^{L+1}$, the scheme can evaluate circuits of depth L. We can choose $q \approx 2^{\mathcal{O}(L \log n)}$ similar to [BGV14, Bra12].

The cost of multiplying two ciphertexts using the tensor $\boldsymbol{\mathcal{M}}_{evk}$ is of the order of $\mathcal{O}(n^3 \log^2 q)$. Therefore, the per gate computation of the scheme is $\mathcal{O}(n^3 \log^2 q) = \tilde{\mathcal{O}}(n^3 \cdot L^2)$.

4.4 Summary

A new multiplication technique for LWE based fully homomorphic encryption has been proposed. This technique avoids the process of relinearization. The evaluation key is a third-order tensor and a ciphertext obtained by evaluating this tensor has the same size as the original ciphertexts. The increase in noise is linear and per gate computation of the scheme is $\tilde{\mathcal{O}}(n^3 \cdot L^2)$.

Chapter 5

Hidden Subspace Membership

In this chapter we attempt to generalize the LWE problem. We introduce a decision problem called the Hidden Subspace Membership (HSM) problem of which the LWE problem is a particular case. We then examine the hardness of a few instances of the HSM problem.

5.1 The Hidden Subspace Membership Problem

For a given noise distribution $\mathcal{N}$, the HSM problem is to distinguish between the distribution of elements of a subspace of a vector space $\mathcal{V}$ that have been corrupted with noise sampled from $\mathcal{N}$ and the uniform distribution over $\mathcal{V}$.

If $\mathcal{S}$ denotes an n-dimensional subspace of the vector space $\mathbb{Z}_q^\ell$ for some $\ell, n \in \mathbb{N}$ and $\mathcal{N}$ denotes a noise distribution on $\mathbb{Z}_q^\ell$, then the input to the HSM problem are samples of the form $(\boldsymbol{v}_i + \boldsymbol{e}_i) \in \mathbb{Z}_q^\ell$ where $\boldsymbol{v}_i \xleftarrow{\$} \mathcal{S}$ and $\boldsymbol{e}_i \xleftarrow{\$} \mathcal{N}$. Given a vector $\boldsymbol{v}$, in the HSM problem, one has to decide whether $\boldsymbol{v} \in \mathcal{S} + \mathcal{N}$ or is sampled uniformly at random from $\mathbb{Z}_q^\ell$. We denote the set of all n-dimensional subspaces of $\mathbb{Z}_q^\ell$ by $G_n(\mathbb{Z}_q^\ell)$. Formally, the HSM problem

can be defined as follows.

Definition 5.1.1. (Hidden Subspace Membership). *For positive integers $n \geq 1, \ell \geq n, q \in \mathbb{N}$ where q is prime, let $\mathcal{S}$ be sampled from a distribution $\mathcal{Y}$ over $G_n(\mathbb{Z}_q^\ell)$ and $\mathcal{N}$ be a noise distribution on $\mathcal{V}$. Then, for a noise distribtion $\mathcal{N}$, the $\mathrm{HSM}_{\ell,n,q,\mathcal{Y},\mathcal{N}}$ can be defined in terms of the game shown in Figure 5.1. A PPT adversary $\mathcal{A}$ wins the game if it can guess the value of $\beta \in \{0,1\}$ with a non-negligible advantage. The advantage of $\mathcal{A}$ in solving the $\mathrm{HSM}_{\ell,n,q,\mathcal{Y},\mathcal{N}}$ problem is given by*

$$Adv^{\mathrm{HSM}}_{\mathcal{A},\ell,n,q,\mathcal{Y},\mathcal{N}}(\lambda) := \left| Pr[\mathrm{HSM}^{\mathcal{A}}_{\ell,n,q,\mathcal{Y},\mathcal{N}}(\lambda) = 1] - \frac{1}{2} \right| \tag{5.1}$$

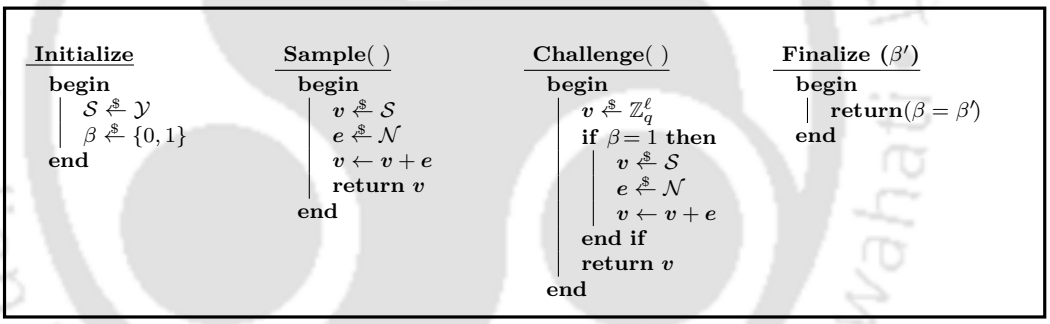

Fig 5.1: $\mathrm{HSM}_{\ell,n,q,\mathcal{Y},\mathcal{N}}$ Game

Observe that, the noise-free variant of the problem is extremely easy to solve. We can find n-linearly independent samples of $\mathcal{S}$ that can be used to construct a basis for $\mathcal{S}$ which in turn can be used to construct a basis for its perpendicular space $\mathcal{S}^\perp$. Then, one can check whether a given vector lies in $\mathcal{S}$ by checking if it lies in the kernel of $\mathcal{S}^\perp$.

Consider the LWE problem with parameters $n, q, \mathcal{X}$ denoted by $\mathrm{LWE}_{n,q,\mathcal{X}}$. Given an $\mathrm{LWE}_{n,q,\mathcal{X}}$ sample $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $\langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i = b_i \ (mod\ q)$,

we can write

$$\begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix} \begin{bmatrix} -\boldsymbol{s}^T \\ 1 \end{bmatrix} \approx_{\mathcal{X}} 0 \ (mod\ q) \tag{5.2}$$

This is a noisy equation with the noise being sampled from the distribution $\mathcal{X}$ on $\mathbb{Z}_q$. Observe that, $(\boldsymbol{a}_i, b_i)$ is a noisy element of the n-dimensional subspace $(-\boldsymbol{s}, 1)^{\perp} \subseteq \mathbb{Z}_q^{n+1}$. Clearly, $\text{LWE}_{n,q,\mathcal{X}}$ is a specific case of the HSM problem namely, $\text{HSM}_{n+1,n,q,\mathcal{Y},\mathcal{N}}$ where $\mathcal{N} = (0^n, \mathcal{X})$ and $\mathcal{Y}$ is the distribution of subspaces of the form $(-\boldsymbol{s}, 1)^{\perp}$ where $\boldsymbol{s}$ is randomly sampled from a uniform distribution over $\mathbb{Z}_q^n$.

Let $\mathcal{S}$ be an n-dimensional subspace of the vector space $\mathbb{Z}_q^{\ell}$. There exist atleast n indices such that there are no linear relations between the corresponding entries of vectors in $\mathcal{S}$. In other words, there exist integers $1 \leq i_1, i_2, \ldots, i_n \leq \ell$ such that for all $(\alpha_1, \ldots, \alpha_n) \in \mathbb{Z}_q^n$ and $\boldsymbol{v} \in \mathcal{S}$, $\sum_{j=1}^{n} \alpha_j \boldsymbol{v}(i_j) \neq 0$. Without loss of generality, we assume that these n elements are the first n elements of a vector $\boldsymbol{v} \in \mathcal{S}$ and every other element can be written as a linear combination of these n elements. In other words, we restrict ourselves to subspaces of the form $(span(-\boldsymbol{S}, \boldsymbol{I}))^{\perp}$.

5.2 Hardness of HSM

In this section we prove that the HSM problem is difficult, for various noise distributions, under the assumption that $\text{LWE}_{n,q,\mathcal{X}}$ is hard.

We start by considering Lemma 6.2 of [PW11] which proves the hardness of a special case of the HSM problem.

Lemma 5.2.1. *Let $h, \ell = poly(n)$. Choose $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{h \times n}$, $\boldsymbol{S} \leftarrow \mathbb{Z}_q^{(\ell-n) \times n}$ uniformly at random and $\boldsymbol{E} \leftarrow \mathcal{X}^{h \times (\ell-n)}$. If $\boldsymbol{B} = \boldsymbol{A}\boldsymbol{S}^T + \boldsymbol{E}$, then the distribution*

of $(\boldsymbol{A}, \boldsymbol{B})$ *is computationally indistinguishable from the uniform distribution over* $\mathbb{Z}_q^{h\times\ell}$ *under the assumption that* $LWE_{n,q,\mathcal{X}}$ *is hard.*

Observe that the rows of the $(\boldsymbol{A}, \boldsymbol{B})$ are samples of the $\text{HSM}_{\ell,n,q,\mathcal{Y},\mathcal{N}}$ problem when the subspace is of the form $(span(-\boldsymbol{S}, \boldsymbol{I}))^{\perp}$ and $\mathcal{N} := (0^n, \mathcal{X}^{\ell-n})$.

Consider a distribution $\mathcal{X}$. For a non zero vector $\boldsymbol{u} = (u_1, \ldots, u_\nu) \in \mathbb{F}_q^\nu$, let $\mathcal{X}_{\boldsymbol{u}}$ be the distribution of $x = \sum_{i=1}^{\nu} x_i u_i$ where the x_is are randomly sampled from the distribution $\mathcal{X}$. For some $\boldsymbol{s} \in \mathbb{Z}_q^n$, chosen uniformly at random, we define $\mathcal{L}_{\boldsymbol{s},\mathcal{X}_{\boldsymbol{u}}}$ to be the distribution of $(\boldsymbol{a}, b = \langle \boldsymbol{a}, \boldsymbol{s}\rangle + e)$ where $\boldsymbol{a} \xleftarrow{\$} \mathbb{Z}_q^n$ and $e \xleftarrow{\$} \mathcal{X}_{\boldsymbol{u}}$.

Lemma 5.2.2. *Given* $\boldsymbol{u} \in \mathbb{F}_q^\nu$*, if there is an efficient algorithm* $\mathcal{A}$ *that can distinguish the distribution* $\mathcal{L}_{\boldsymbol{s},\mathcal{X}_{\boldsymbol{u}}}$ *from the uniform distribution over* $\mathbb{Z}_q^n \times \mathbb{Z}_q$*, then there exists an efficient algorithm* $\mathcal{B}$ *that can solve the* $DLWE_{n,q,\mathcal{X}}$ *problem.*

Proof. Given an unknown distribution $\mathcal{D}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ where $\mathcal{D}$ is either the $\text{LWE}_{n,q,\mathcal{X}}$ distribution or the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$, $\mathcal{B}$ takes ν samples $(\boldsymbol{a}_i, b_i)_{i=1}^{\nu}$ from $\mathcal{D}$ and calls $\mathcal{A}$ with $(\boldsymbol{a} = \sum_{i=1}^{\nu} \boldsymbol{a}_i u_i', b = \sum_{i=1}^{\nu} b_i u_i')$ for some $u_i' \in \mathbb{Z}_q, 1 \le i \le \nu$. If $\mathcal{D}$ is the $\text{LWE}_{n,q,\mathcal{X}}$ distribution for some $\boldsymbol{s} \in \mathbb{Z}_q^n$, then $\mathcal{B}$ is distributed as $\mathcal{L}_{\boldsymbol{s},\mathcal{X}_{\boldsymbol{u}}}$ and if $\mathcal{D}$ is the uniform distribution, then $\mathcal{B}$ is distributed uniformly over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. Hence, $\mathcal{B}$ can efficiently solve the $\text{DLWE}_{n,q,\mathcal{X}}$ problem if $\mathcal{A}$ efficiently distinguishes the distribution $\mathcal{L}_{\boldsymbol{s},\mathcal{X}_{\boldsymbol{u}}}$ from uniform. □

We now use the above result to prove a generalization of Lemma 5.2.1.

Lemma 5.2.3. *Let* $h, \ell = poly(n)$*. Choose* $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{h\times n}, \boldsymbol{S} \leftarrow \mathbb{Z}_q^{(\ell-n)\times n}$ *uniformly at random and let* $\boldsymbol{E} \leftarrow \mathcal{X}^{h\times r}$*, for some* $r \ge \ell - n$*. Let* $\boldsymbol{U} \in \mathbb{F}_q^{r\times(\ell-n)}$ *have full column rank. If* $\boldsymbol{B} = \boldsymbol{A}\boldsymbol{S}^T + \boldsymbol{E}\boldsymbol{U}$*, then the distribution* $(\boldsymbol{A}, \boldsymbol{B})$ *is*

indistinguishable from the uniform distribution over $\mathbb{Z}_q^{h\times\ell}$ *under the* $LWE_{n,q,\mathcal{X}_{\boldsymbol{u}}}$ *assumption.*

Proof. This lemma can be proved using induction. Let $\mathcal{H}_0,\ldots,\mathcal{H}_{\ell-n}$ denote a set of distributions over $\mathbb{Z}_q^{h\times\ell}$. Here, for each $j\in[\ell-n]$, $\mathcal{H}_j$ is the distribution of matrix pairs $(\boldsymbol{A},\boldsymbol{B}_j)$ where the matrix $\boldsymbol{A}$ is chosen randomly from a uniform distribution over $\mathbb{Z}_q^{h\times n}$ and the first j columns of $\boldsymbol{B}_j$ are equal to the first j columns of $\boldsymbol{AS}^T+\boldsymbol{EU}$ where $\boldsymbol{E}$ is randomly sampled from $\mathcal{X}^{h\times(\ell-n)}$. The remaining columns of $\boldsymbol{B}_j$ are chosen randomly from a uniform distribution. Note that, since $\boldsymbol{A}$ is chosen from a uniform distribution, $\mathcal{H}_0$ is the uniform distribution over $\mathbb{Z}_q^{h\times\ell}$. Assume that the distributions $\mathcal{H}_0,\mathcal{H}_1,\ldots,\mathcal{H}_{j-1}$ are indistinguishable from the uniform distribution over $\mathbb{Z}_q^{h\times\ell}$. We will now show that $\mathcal{H}_j$ is indistinguishable from the uniform distribution based on the LWE assumption.

The ith column of a sample of $\mathcal{H}_j$ is given by $\boldsymbol{b}_{ji}=\boldsymbol{A}\boldsymbol{s}_i^T+\boldsymbol{E}\boldsymbol{u}_i$ for $1\le i\le j$ where $\boldsymbol{s}_i$ and $\boldsymbol{u}_i$ denote the i^{th} column of $\boldsymbol{S}$ and $\boldsymbol{U}$ respectively.

Consider the subspace spanned by the vectors $\left(\boldsymbol{E}\boldsymbol{u}_i\in\mathbb{F}_q^h\right)_{i=1}^{j}$. Let this space be denoted by $\mathcal{V}_j$. Since $\boldsymbol{u}_1,\ldots,\boldsymbol{u}_j$ are linearly independent, there exists a $\boldsymbol{u}\in\mathcal{V}_j$ such that $\boldsymbol{E}\boldsymbol{u}$ is statistically independent of $\boldsymbol{E}\boldsymbol{u}_1,\ldots,\boldsymbol{E}\boldsymbol{u}_{j-1}$. Such a $\boldsymbol{E}\boldsymbol{u}$ will be linearly independent of $\boldsymbol{E}\boldsymbol{u}_1,\ldots,\boldsymbol{E}\boldsymbol{u}_{j-1}$. Thus, $(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{j-1},\boldsymbol{u})$ forms a basis for $\mathcal{V}_j$. Therefore, for some $(k_1,\ldots,k_j)\in\mathbb{F}_q^j$, we have

$$\boldsymbol{u}_j=\sum_{i=1}^{j-1}k_i\boldsymbol{u}_i+k_j\boldsymbol{u} \tag{5.3}$$

Further, there exists an $\tilde{\boldsymbol{s}} \in \mathbb{F}_q^n$ such that

$$\boldsymbol{A}\boldsymbol{s}_j^T = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{A}\boldsymbol{s}_1^T & \dots & \boldsymbol{A}\boldsymbol{s}_{j-1}^T \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{s}}^T \\ k_1 \\ \vdots \\ k_{j-1} \end{bmatrix} \tag{5.4}$$

Therefore, from equations (5.3), (5.4), the j^{th} column of $\boldsymbol{B}$ can be written as

$$\boldsymbol{A}\boldsymbol{s}_j^T + \boldsymbol{E}\boldsymbol{u}_j = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{A}\boldsymbol{s}_1^T + \boldsymbol{E}\boldsymbol{u}_1 & \dots & \boldsymbol{A}\boldsymbol{s}_{j-1}^T + \boldsymbol{E}\boldsymbol{u}_{j-1} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{s}}^T \\ k_1 \\ \vdots \\ k_{j-1} \end{bmatrix} + k_j \boldsymbol{E}\boldsymbol{u} \tag{5.5}$$

The columns of the matrix $\begin{bmatrix} \boldsymbol{A}\boldsymbol{s}_1^T + \boldsymbol{E}\boldsymbol{u}_1 & \dots & \boldsymbol{A}\boldsymbol{s}_{j-1}^T + \boldsymbol{E}\boldsymbol{u}_{j-1} \end{bmatrix}$ are the first $j-1$ columns of a sample of the distribution $\mathcal{H}_{j-1}$.Therefore, by hypothesis, the distribution of this matrix is indistinguishable from the uniform distribution. Since $\boldsymbol{s}_j$ and $\boldsymbol{u}_j$ are chosen from uniform distributions, the distribution of the vector $\bar{\boldsymbol{s}} = \begin{bmatrix} \tilde{\boldsymbol{s}} & k_1 & \dots & k_{j-1} \end{bmatrix}^T$ is uniform. From Equation 5.5 it can be observed that the first j columns of a sample of the distribution $\mathcal{H}_j$ is just like a sample of the distribution $\mathcal{L}_{\bar{\boldsymbol{s}},\mathcal{X}_{\boldsymbol{u}}}$, except that the matrix $\begin{bmatrix} \boldsymbol{A}\boldsymbol{s}_1^T + \boldsymbol{E}\boldsymbol{u}_1 & \dots & \boldsymbol{A}\boldsymbol{s}_{j-1}^T + \boldsymbol{E}\boldsymbol{u}_{j-1} \end{bmatrix}$ is not sampled from a uniform distribution but from the first $j-1$ columns of a sample of $\mathcal{H}_{j-1}$. Therefore, an oracle that can distinguish a sample of $\mathcal{H}_j$ from a sample of $\mathcal{L}_{\bar{\boldsymbol{s}},\mathcal{X}_{\boldsymbol{u}}}$ can in turn be used to distinguish a sample of $\mathcal{H}_{j-1}$ from the uniform distribution. However, by hypothesis, this is not possible. Thus, the first j columns of a sample of $\mathcal{H}_j$ are indistinguishable from a sample of $\mathcal{L}_{\bar{\boldsymbol{s}},\mathcal{X}_{\boldsymbol{u}}}$.

Consider an oracle $\mathcal{W}$ which accepts or rejects samples from an unknown distribution $\mathcal{D}$ over $\mathbb{Z}_q^{h\times\ell}$. Let $p_{\mathcal{H}_j}(\mathcal{W})$ and $p_U(\mathcal{W})$ be the probabilities that $\mathcal{W}$ accepts a sample from the distribution $\mathcal{H}_j$ and the uniform distribution respectively. Let $\mathcal{L}_j$ be the distribution of samples got by appending $\ell-n+j$ vectors in $\mathbb{F}_q^h$ that are randomly chosen from a uniform distribution to a sample of $\mathcal{L}_{\bar{\boldsymbol{s}},\mathcal{X}_{\boldsymbol{u}}}$. Let $p_{\mathcal{L}_j}(W)$ be the probability that $\mathcal{W}$ accepts a sample of $\mathcal{L}_j$. Now,

$$|p_{\mathcal{H}_j}(\mathcal{W}) - p_U(\mathcal{W})| \leq |p_{\mathcal{H}_j}(\mathcal{W}) - p_{\mathcal{L}_j}(W)| - |p_{\mathcal{L}_j}(W) - p_U(\mathcal{W})| \tag{5.6}$$

Since $\mathcal{H}_j$ is indistinguishable from $\mathcal{L}_j$ and $\mathcal{L}_j$ is indistinguishable from the uniform distribution, both the terms in the right hand side of the above inequality are negligible. Therefore, the distribution $\mathcal{H}_j$ is indistinguishable from the uniform distribution. □

Given a matrix $\boldsymbol{R} \in \mathbb{Z}_q^{(\ell-n)\times n}$ and the distribution $\mathcal{X}^{\ell-n}$, Let $\mathcal{X}^{\ell-n}\boldsymbol{R}$ denote the distribution got by multiplying vectors sampled from $\mathcal{X}^{\ell-n}$ with $\boldsymbol{R}$. Let $\mathcal{N}_{\boldsymbol{R},\mathcal{X}}$ be the distribution of vectors of the form $(\boldsymbol{v}\boldsymbol{R}, \boldsymbol{v})$, where $\boldsymbol{v}$ is sampled from the distribution $\mathcal{X}^{\ell-n}$.

Lemma 5.2.4. *Let $\boldsymbol{S}$ be randomly sampled from a uniform distribution over $\mathbb{Z}_q^{(\ell-n)\times n}$ and $\boldsymbol{R}$ be chosen such that $(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{S}^T)$ is invertible. Let $\boldsymbol{v}$ and $\boldsymbol{e}$ be randomly sampled from the uniform distribution over $\mathbb{Z}_q^n$ and $\mathcal{N}_{\boldsymbol{R},\mathcal{X}}$ respectively. If there is an efficient algorithm that can distinguish the distribution of $\boldsymbol{v}\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \boldsymbol{e}$ from the uniform distribution over $\mathbb{Z}_q^\ell$ then there exists an efficient algorithm for solving the $DLWE_{n,q,\mathcal{X}}$ problem.*

Proof. Observe that

$$\boldsymbol{v}\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \boldsymbol{e} = \left(\boldsymbol{v} + \boldsymbol{e}_1\boldsymbol{R}, \boldsymbol{v}\boldsymbol{S}^T + \boldsymbol{e}_1\right) \tag{5.7}$$

where $\boldsymbol{e}_1$ is randomly sampled from the distribution $\mathcal{X}^{\ell-n}$. If $\boldsymbol{v}'_1 := \boldsymbol{v} + \boldsymbol{e}_1\boldsymbol{R}$ and $\boldsymbol{v}'_2 := \boldsymbol{v}'_1\boldsymbol{S}^T$, then equation (5.7) can be written as

$$\begin{aligned} \boldsymbol{v}\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \boldsymbol{e} &= \left(\boldsymbol{v}'_1, \boldsymbol{v}'_2 + \boldsymbol{e}_1(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{S}^T)\right) \\ &= \boldsymbol{v}'_1\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \begin{bmatrix}\boldsymbol{0} & \boldsymbol{e}_1(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{S}^T)\end{bmatrix} \end{aligned} \tag{5.8}$$

Since $\boldsymbol{v}$ is sampled from a uniform distribution, the distribution of $\boldsymbol{v}'_1$ is also uniform. Therefore, from Lemma 5.2.3 and the fact that $(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{S}^T)$ is invertible, we can conclude that there exists an efficient algorithm for solving the $\text{DLWE}_{n,q,\mathcal{X}}$ problem if there is an efficient algorithm that can distinguish the distribution of $\boldsymbol{v}\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \boldsymbol{e}$ from the uniform distribution over $\mathbb{Z}_q^\ell$. □

If $\mathcal{Y}$ denotes the uniform distribution over n dimensional subspaces of $\mathbb{Z}_q^\ell$ which are of the form $(span(-\boldsymbol{S}, \boldsymbol{I}))^\perp$, then the above lemma proves that the $\text{HSM}_{\ell,n,q,\mathcal{Y},\mathcal{N}_{\boldsymbol{R},\mathcal{X}}}$ is difficult to solve assuming the hardness of the $\text{DLWE}_{n,q,\mathcal{X}}$ problem.

Let us now consider the case where the noise is sampled from the distribution $\mathcal{X}^\ell$. Thus, every entry of a sample vector is corrupted with noise chosen independently from $\mathcal{X}$.

Lemma 5.2.5. *Let $\boldsymbol{S}$ be randomly sampled from a uniform distribution over $\mathbb{Z}_q^{(\ell-n)\times n}$. Let $\boldsymbol{v}$ and $\boldsymbol{e}$ be randomly sampled from the uniform distribution over $\mathbb{Z}_q^n$ and $\mathcal{X}^\ell$ respectively. If there is an efficient algorithm that can distinguish the distribution of $\boldsymbol{v}\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \boldsymbol{e}$ from the uniform distribution over $\mathbb{Z}_q^\ell$ then there exists an efficient algorithm for solving the $\text{DLWE}_{n,q,\mathcal{X}}$ problem.*

Proof. Observe that

$$\boldsymbol{v}\begin{bmatrix}\boldsymbol{I} & \boldsymbol{S}^T\end{bmatrix} + \boldsymbol{e} = \left(\boldsymbol{v} + \boldsymbol{e}_1, \boldsymbol{v}\boldsymbol{S}^T + \boldsymbol{e}_2\right) \tag{5.9}$$

where $\boldsymbol{e}_1$ is randomly sampled from the distribution $\mathcal{X}^n$ and $\boldsymbol{e}_2$ is randomly sampled from the distribution $\mathcal{X}^{\ell-n}$. If $\boldsymbol{v}_1' := \boldsymbol{v} + \boldsymbol{e}_1$ and $\boldsymbol{v}_2' := \boldsymbol{v}_1' \boldsymbol{S}^T$, then equation (5.9) can be written as

$$\begin{aligned} \boldsymbol{v}\begin{bmatrix} \boldsymbol{I} & \boldsymbol{S}^T \end{bmatrix} + \boldsymbol{e} &= \left(\boldsymbol{v}_1', \boldsymbol{v}_2' + \boldsymbol{e}_2 - \boldsymbol{e}_1\boldsymbol{S}^T\right) \\ &= \boldsymbol{v}_1'\begin{bmatrix} \boldsymbol{I} & \boldsymbol{S}^T \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} & \begin{bmatrix} \boldsymbol{e}_2 & \boldsymbol{e}_1 \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{I} \\ -\boldsymbol{S}^T \end{bmatrix} \end{bmatrix} \end{aligned} \tag{5.10}$$

Since $\boldsymbol{v}$ is sampled from a uniform distribution, the distribution of $\boldsymbol{v}_1'$ is also uniform. Therefore, from Lemma 5.2.3 and the fact that $\begin{bmatrix} \boldsymbol{I} \\ -\boldsymbol{S}^T \end{bmatrix}$ has full column rank, we can conclude that there exists an efficient algorithm for solving the $\mathrm{DLWE}_{n,q,\mathcal{X}}$ problem if there is an efficient algorithm that can distinguish the distribution of $\boldsymbol{v}\begin{bmatrix} \boldsymbol{I} & \boldsymbol{S}^T \end{bmatrix} + \boldsymbol{e}$ from the uniform distribution over $\mathbb{Z}_q^\ell$. □

Thus, the hardness of the $\mathrm{DLWE}_{n,q,\mathcal{X}}$ problem implies the hardness of the $\mathrm{HSM}_{\ell,n,q,\mathcal{Y},\mathcal{X}^\ell}$ problem.

5.3 Summary

In this chapter, we have described the Hidden Subspace Membership problem and have proved the hardness of a few of its instances under the assumption that the LWE problem is hard to solve.

Chapter 6

Conclusion

In this chapter, we give a summary of the results presented in this thesis and discuss possible research directions for future work.

6.1 Summary of the Results

In this thesis, we have presented a number of theoretical results in the area of lattice-based cryptography. The following is a summary of the results.

Chapter 3 deals with the construction of a fully homomorphic encryption scheme using multivariate polynomials. We have shown that such a scheme can be constructed in the framework of LWE-based schemes that can encrypt multiple plaintext bits in a single ciphertext. The security of the scheme depends on the hardness of the LWE problem. Homomorphic multiplication is performed by evaluating a bilinear map on the ciphertexts represented by a third-order tensor. This method does not increase the size of the ciphertext after multiplication and the increase in noise is only linear. The tensor is given as the public evaluation key for multiplication. We have shown that it is difficult to recover the secret key from this tensor based on the hardness of solving a system of non-linear polynomial equations.

In Chapter 4, we have shown that relinearization and modulus switching can be avoided by using the multiplication method proposed in Chapter 3 to homomorphically multiply ciphertexts in previous LWE-based schemes. We obtain a per gate computation overhead of $\tilde{\mathcal{O}}(n^3L^2)$.

In Chapter 5, we have introduced a decision problem called the Hidden Subspace Membership (HSM) problem. We have then shown that the HSM problem is hard for various noise distributions under the assumption that the LWE problem is hard.

6.2 Scope for Future Work

The work presented in this thesis can further be extended in the following directions:

- The main issue against the practical application of fully homomorphic encryption schemes is their efficiency. The proposed (leveled) FHE scheme is no different. Switching the underlying hardness assumption from LWE to ring-LWE has shown significant improvements in the performance of previous LWE-based schemes. Therefore, one may consider the possibility of constructing an RLWE variant of the proposed scheme.

- Although RLWE variants perform better than their LWE counterparts, the hardness of the RLWE problem is based on the hardness of the shortest vector problem over *ideal lattices*. To the best of our knowledge, it hasn't been proved that this problem is as hard as the corresponding problem over regular lattices. Therefore, another possible direction for future work is to improve the performance of the proposed scheme with respect to per gate time complexity so that it theoretically approaches that of RLWE based schemes.

- In a leveled FHE scheme, the size of the evaluation key depends on the depth of the evaluated circuit. The only existing way to remove this dependency is to use Gentry's bootstrapping technique. Thus, the problem of converting the proposed leveled FHE scheme to a 'pure' FHE scheme using bootstrapping can be explored.

- Several hardware accelerators have been proposed to speed up the performance of existing homomorphic schemes. It would be interesting to see if such hardware accelerators can be developed to improve the performance of the proposed scheme.

Bibliography

[AAPS11] Frederik Armknecht, Daniel Augot, Ludovic Perret, and Ahmad-Reza Sadeghi. On constructing homomorphic encryption schemes from coding theory. In Chen L., editor, *Cryptography and Coding – IMACC 2011. Lecture Notes in Computer Science*, volume 7089, pages 23–40. Springer, 2011.

[ACC+19] Martin R Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin E Lauter, et al. Homomorphic Encryption Standard. *IACR Cryptology ePrint Archive*, 2019:939, 2019. URL: `https://eprint.iacr.org/2019/939`.

[ACF+15] Martin R Albrecht, Carlos Cid, Jean-Charles Faugere, Robert Fitzpatrick, and Ludovic Perret. On the complexity of the bkw algorithm on lwe. *Designs, Codes and Cryptography*, 74(2):325–354, 2015.

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Halevi S., editor, *CRYPTO 2009. Lecture Notes in Computer Science*, volume 5677, pages 595–618. Springer, 2009.

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proc. of STOC'97*, pages 284–293. ACM, 1997.

[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange—a new hope. In *25th USENIX Security Symposium (SEC'16)*, pages 327–343, USA, 2016.

[ADRSD15] Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the shortest vector problem in 2^n time using discrete Gaussian sampling. In *Proc. of STOC'15*, pages 733–742. ACM, 2015.

[AFF+16] Martin R Albrecht, Jean-Charles Faugère, Pooya Farshim, Gottfried Herold, and Ludovic Perret. Polly cracker, revisited. *Designs, Codes and Cryptography*, 79(2):261–302, 2016.

[AFFP11] Martin R Albrecht, Pooya Farshim, Jean-Charles Faugere, and Ludovic Perret. Polly Cracker, revisited. In Wang X. Lee D.H., editor, *ASIACRYPT 2011. Lecture Notes in Computer Science*, volume 7073, pages 179–196. Springer, 2011.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proc. of STOC'96*, pages 99–108. ACM, 1996.

[Ajt98] Miklós Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions. In *Proc. of STOC'98*, pages 10–19. ACM, 1998.

[AKS01] Miklós Ajtai, Ravi Kumar, and Dandapani Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. of STOC'01*, pages 601–610. ACM, 2001.

[APS15] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of Learning with Errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

[AS08] Frederik Armknecht and Ahmad-Reza Sadeghi. A new approach for algebraically homomorphic encryption. *IACR Cryptology ePrint Archive*, 2008:422, 2008. URL: `https://eprint.iacr.org/2008/422.pdf`.

[ASP13] Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasilinear time. In *Advances in Cryptology – CRYPTO 2013. Lecture Notes in Computer Science, vol 8042, pp. 1–20. Springer, Berlin*, 2013.

[BCE+94] Boo Barkee, Deh Cac Can, Julia Ecks, Theo Moriarty, and RF Ree. Why you cannot even hope to use gröbner bases in public key cryptography: an open letter to a scientist who failed and a challenge to those who have not yet failed. *Journal of Symbolic Computation*, 18(6):497–501, 1994.

[BDJR97] Mihir Bellare, Anand Desai, Eron Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *Proc.of FOCS'97*, pages 394–403. IEEE Computer Society, 1997.

[BEHZ16] Jean-Claude Bajard, Julien Eynard, M Anwar Hasan, and Vincent Zucca. A full RNS variant of FV like somewhat homomor-

phic encryption schemes. In *Selected Areas in Cryptography – SAC 2016. Lecture Notes in Computer Science, vol 10532, pp. 423–442. Springer, Cham*, 2016.

[Ben87] Josh Daniel Cohen Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, 1987.

[BFP09] Luk Bettale, Jean-Charles Faugere, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *Journal of Mathematical Cryptology*, 3(3):177–197, 2009.

[BGH13] Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in LWE-based homomorphic encryption. In Hanaoka G. Kurosawa K., editor, *Public Key Crypytography – PKC 2013. Lecture Notes in Computer Science*, volume 7778, pages 1–13. Springer, 2013.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Kilian J., editor, *Theory of Cryptography. TCC 2005. Lecture Notes in Computer Science*, volume 3378, pages 325–341. Springer, 2005.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) Fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):13, 2014.

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of Learning with Errors. In *Proc. of STOC'13*, pages 575–584. ACM, 2013.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In Canetti R. Safavi-Naini R., editor, *CRYPTO 2012. Lecture Notes in Computer Science*, volume 7417, pages 868–886. Springer, 2012.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Rogaway P., editor, *CRYPTO 2011. Lecture Notes in Computer Science*, pages 505–524. Springer, 2011.

[BV14a] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on Computing*, 43(2):831–871, 2014.

[BV14b] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In *Innovations in Theoretical Computer Science, ITCS'14*, pages 1–12, Princeton, NJ, USA, 2014. ACM.

[CCK+13] Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrede Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch fully homomorphic encryption over the integers. In Nguyen P.Q. T., editor, *EUROCRYPT 2013. Lecture Notes in Computer Science*, volume 7881, pages 315–335. Springer, 2013.

[CDS15] Sergiu Carpov, Paul Dubrulle, and Renaud Sirdey. Armadillo: a compilation chain for privacy preserving applications. In *Proceedings of the 3rd International Workshop on Security in Cloud Computing*, pages 13–19, New York, USA, 2015. ACM.

[CGGI16a] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In Takagi T. Cheon J., editor, *ASIACRYPT 2016. Lecture Notes in Computer Science*, volume 10031, pages 3–33. Springer, 2016.

[CGGI16b] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast fully homomorphic encryption library, August 2016. URL: `https://tfhe.github.io/tfhe/`.

[CGGI17] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In *Advances in Cryptology – ASIACRYPT 2017. Lecture Notes in Computer Science, vol 10624, pp. 377–408. Springer, Cham.*, 2017.

[CJP20] Ilaria Chillotti, Marc Joye, and Pascal Paillier. Programmable bootstrapping enables efficient homomorphic inference of deep neural networks. Technical report, 2020.

[CKKS17] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology – ASIACRYPT 2017. Lecture Notes in Computer Science, vol 10624, pp. 409–437. Springer, Cham.*, 2017.

[CLO92] David Cox, John Little, and Donal O'shea. *Ideals, varieties, and algorithms*, volume 3. Springer, 1992.

[CMNT11] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Fully homomorphic encryption over the inte-

gers with shorter public keys. In Rogaway P., editor, *CRYPTO 2011. Lecture Notes in Computer Science*, volume 6841, pages 487–504. Springer, 2011.

[CMTM18] Eduardo Chielle, Oleg Mazonka, Nektarios Georgios Tsoutsos, and Michail Maniatakos. E3: A framework for compiling c++ programs with encrypted operands. *IACR Cryptology ePrint Archive*, 2018:1013, 2018.

[CP02] Nicolas T Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In Zheng Y., editor, *ASIACRYPT 2002. Lecture Notes in Computer Science*, volume 2501, pages 267–287. Springer, 2002.

[Cro17] Eric Crockett. *Simply safe lattice cryptography.* PhD thesis, Georgia Institute of Technology, 2017.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[DKS98] Irit Dinur, Guy Kindler, and Shmuel Safra. Approximating-cvp to within almost-polynomial factors is NP-hard. In *Proc. of FOCS'98*, pages 99–109. IEEE Computer Society, 1998.

[DKS+20] Roshan Dathathri, Blagovesta Kostova, Olli Saarikivi, Wei Dai, Kim Laine, and Madan Musuvathi. Eva: An encrypted vector arithmetic language and compiler for efficient homomorphic computation. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 546–561, New York, USA, 2020. ACM.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In Fischlin M. Oswald E., editor, *EUROCRYPT 2015. Lecture Notes in Computer Science*, volume 9056, pages 617–640. Springer, 2015.

[ElG84] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In Chaum D. Blakley G.R., editor, *CRYPTO 1984. Lecture Notes in Computer Science*, volume 196, pages 10–18. Springer, 1984.

[FJ03] Jean-Charles Faugere and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases. In *CRYPTO 2003. Lecture Notes in Computer Science*, volume 2729, pages 44–60. Springer, 2003.

[FK93] Michael Fellows and Neal Koblitz. Combinatorial cryptosystems galore. *Contemporary Mathematics*, 168(2):51–61, 1993.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.

[Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[Geo11] Adela Georgescu. A lwe-based secret sharing scheme. *IJCA Special Issue on Network Security and Cryptography, NSC*, (3):27–29, 2011.

[GG00] Oded Goldreich and Shafi Goldwasser. On the limits of non-approximability of lattice problems. *Journal of Computer and System Sciences*, 60(3):540–563, 2000.

[GH11] Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. In Paterson K.G., editor, *EUROCRYPT 2011. Lecture Notes in Computer Science*, volume 6632, pages 129–148. Springer, 2011.

[GHS12a] Craig Gentry, Shai Halevi, and Nigel P Smart. Better bootstrapping in fully homomorphic encryption. In Manulis M. Fischlin M., Buchmann J., editor, *Public Key Cryptography – PKC 2012. Lecture Notes in Computer Science*, volume 7293, pages 1–16. Springer, 2012.

[GHS12b] Craig Gentry, Shai Halevi, and Nigel P Smart. Fully homomorphic encryption with polylog overhead. In Johansson T. Pointcheval D., editor, *EUROCRYPT 2012. EUROCRYPT 2012. Lecture Notes in Computer Science*, volume 7237, pages 465–482. Springer, 2012.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proc. of STOC'82*, pages 365–377. ACM, 1982.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proc. of STOC'08*, pages 197–206. ACM, 2008.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Garay J.A. Canetti R., editor, *CRYPTO 2013. Lecture Notes in Computer Science*, volume 8042, pages 75–92. Springer, 2013.

[HAO16] Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. Packing messages and optimizing bootstrapping in GSW-FHE. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 99(1):73–82, 2016.

[Her12] Gottfried Herold. Polly cracker, revisited, revisited. In Manulis M. Fischlin M., Buchmann J., editor, *Public Key Cryptography – PKC 2012. Lecture Notes in Computer Science*, volume 7293, pages 17–33. Springer, 2012.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. NTRU: A ring-based public key cryptosystem. In Buhler J.P.Buhler J.P., editor, *Algorithmic Number Theory. ANTS 1998. Lecture Notes in Computer Science*, volume 1423, pages 267–288. Springer, 1998.

[HPS19] Shai Halevi, Yuriy Polyakov, and Victor Shoup. An improved RNS variant of the BFV homomorphic encryption scheme. In *Topics in Cryptology – CT-RSA 2019. Lecture Notes in Computer Science, vol 11405, pp. 83–105. Springer, Cham.*, 2019.

[HR07] Ishay Haviv and Oded Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proc. of STOC'07*, pages 469–477. ACM, 2007.

[HS14] Shai Halevi and Victor Shoup. Algorithms in HElib. In *Advances in Cryptology – CRYPTO 2014. CRYPTO 2014. Lecture Notes in Computer Science, vol 8616, pp. 554–571. Springer, Berlin*, 2014.

[HS15] Shai Halevi and Victor Shoup. Bootstrapping for HElib. In *Advances in Cryptology – EUROCRYPT 2015. Lecture Notes in Computer Science, vol 9056, pp. 641–670. Springer, Berlin*, 2015.

[HS18] Shai Halevi and Victor Shoup. Faster homomorphic linear transformations in HElib. In *Advances in Cryptology – CRYPTO 2018. Lecture Notes in Computer Science, vol 10991, pp. 93–120. Springer, Cham.*, 2018.

[IP07] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Vadhan S.P., editor, *Theory of Cryptography. TCC 2007. Lecture Notes in Computer Science*, volume 4392, pages 575–594. Springer, 2007.

[Kan83] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. of STOC'83*, pages 193–206. ACM, 1983.

[Kho05] Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *Journal of the ACM (JACM)*, 52(5):789–808, 2005.

[Kho09] Subhash Khot. Inapproximability results for computational problems on lattices. In Vallée B. Nguyen P., editor, *The LLL Algorithm. Information Security and Cryptography*, pages 453–473. Springer, 2009. URL: `https://doi.org/10.1007/978-3-642-02295-1_14`.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In Wiener M., editor,

CRYPTO 1999. Lecture Notes in Computer Science, volume 1666, pages 19–30. Springer, 1999.

[Laz83] Daniel Lazard. Gröbner bases, gaussian elimination and resolution of systems of algebraic equations. In *EUROCAL'83, European Computer Algebra Conference*, pages 146–156, London, England, 1983.

[LdVMPT09] Françoise Levy-dit Vehel, Maria Grazia Marinari, Ludovic Perret, and Carlo Traverso. A survey on polly cracker systems. In *Gröbner Bases, Coding, and Cryptography*, pages 285–305. Springer, 2009. URL: `https://doi.org/10.1007/978-3-540-93806-4_16`.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[LLS90] Jeffrey C Lagarias, Hendrik W Lenstra, and Claus-Peter Schnorr. Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4):333–348, 1990.

[LN97] Rudolf Lidl and Harald Niederreiter. *Finite fields*, volume 20. Cambridge University Press, 1997.

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Kiayias A., editor, *Topics in Cryptology – CT-RSA 2011. Lecture Notes in Computer Science*, volume 6558, pages 319–339. Springer, 2011.

[MCG08] Carlos Aguilar Melchor, Guilhem Castagnos, and Philippe Gaborit. Lattice-based homomorphic encryption of vector spaces. In *IEEE International Symposium on Information Theory*, pages 1858–1862, Toronto, 2008.

[MGH10] Carlos Aguilar Melchor, Philippe Gaborit, and Javier Herranz. Additively homomorphic encryption with d-operand multiplications. In Tal Rabin, editor, *CRYPTO 2010. Lecture Notes in Computer Science*, volume 6223, pages 138–154. Springer, 2010.

[Mic01] Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM journal on Computing*, 30(6):2008–2035, 2001.

[MR09] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009.

[MV13] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. *SIAM Journal on Computing*, 42(3):1364–1391, 2013.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Stern J., editor, *EUROCRYPT 1999. Lecture Notes in Computer Science*, volume 1592, pages 223–238. Springer, 1999.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proc. of STOC'09*, pages 333–342. ACM, 2009.

[PRR17] Yuriy Polyakov, Kurt Rohloff, and Gerard W Ryan. PALISADE Lattice Cryptography Library User Manual. *Cybersecurity Research Center, New Jersey Institute of Technology (NJIT), Tech. Rep*, 15, 2017.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In Wagner D., editor, *CRYPTO 2008. Lecture Notes in Computer Science*, volume 5157, pages 554–571. Springer, 2008.

[PW11] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM Journal on Computing*, 40(6):1803–1844, 2011.

[RAD78] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proc. of STOC'05*, pages 84–93. ACM, 2005.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[RSA78] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[RTJ+19] Sujoy Sinha Roy, Furkan Turan, Kimmo Jarvinen, Frederik Vercauteren, and Ingrid Verbauwhede. Fpga-based high-performance parallel architecture for homomorphic computing on encrypted data. In *2019 IEEE International symposium on high performance computer architecture (HPCA)*, pages 387–398. IEEE, 2019.

[Sch87] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2-3):201–224, 1987.

[SEA20] Microsoft SEAL (release 3.6). `https://github.com/Microsoft/SEAL`, November 2020. Microsoft Research, Redmond, WA.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332, 1999.

[SV10] Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Pointcheval D. Nguyen P.Q., editor, *Public Key Cryptography – PKC 2010. Lecture Notes in Computer Science*, volume 6056, pages 420–443. Springer, 2010.

[SV14] Nigel P Smart and Frederik Vercauteren. Fully homomorphic simd operations. *Designs, Codes and Cryptography*, 71(1):57–81, 2014.

[SYY99] Tomas Sander, Adam Young, and Moti Yung. Non-interactive cryptocomputing for NC^1. In *Proc. of FOCS'99*, pages 554–566. IEEE Computer Society, 1999.

[TRV20] Furkan Turan, Sujoy Sinha Roy, and Ingrid Verbauwhede. Heaws: An accelerator for homomorphic encryption on the amazon aws fpga. *IEEE Transactions on Computers*, 69(8):1185–1196, 2020.

[VDGHV10] Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Gilbert H., editor, *EUROCRYPT 2010. Lecture Notes in Computer Science*, volume 6110., pages 24–43. Springer, 2010.

[vEPIL19] Tim van Elsloo, Giorgio Patrini, and Hamish Ivey-Law. SEALion: A framework for neural network inference on encrypted data. *arXiv preprint arXiv:1904.12840*, 2019. URL: `https://arxiv.org/abs/1904.12840`.

[VJH21] Alexander Viand, Patrick Jattke, and Anwar Hithnawi. SoK: Fully homomorphic encryption compilers. *arXiv preprint arXiv:2101.07078*, 2021. URL: `https://arxiv.org/abs/2101.07078v1`.

[VS18] Alexander Viand and Hossein Shafagh. Marble: Making fully homomorphic encryption accessible to all. In *Proceedings of the*

6th Workshop on Encrypted Computing & Applied Homomorphic Cryptography, pages 49–60. ACM, 2018.