Securely Instantiating Cryptographic Schemes Based on the Learning with Errors Assumption

Vom Fachbereich Informatik der

Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des Grades

Doktor rerum naturalium (Dr. rer. nat.)

von

Dipl.-Math. Florian Göpfert

geboren in Würzburg.

Referenten: Prof. Dr. Johannes Buchmann
Prof. Dr. Jintai Ding

Tag der Einreichung: 11.08.2016
Tag der mündlichen Prüfung: 22.09.2016

Hochschulkennziffer: D 17

Darmstadt 2016
August 11, 2016

Abstract

Since its proposal by Regev in 2005 [Reg05], the Learning With Errors (LWE) problem was used as the underlying problem for a great variety of schemes. Its applications are many-fold, reaching from basic and highly practical primitives like key exchange [ADPS16], public-key encryption [LP11, LPR10], and signature schemes [ABBD15, DDLL13] to very advanced solutions like fully homomorphic encryption [BV14, BGV11], group signatures [LLLS13], and identity based encryption [ABV+11].

One of the underlying reasons for this fertility is the flexibility with that LWE can be instantiated. Unfortunately, this comes at a cost: It makes selecting parameters for cryptographic applications complicated. When selecting parameters for a new LWE-based primitive, a researcher has to take the influence of *several parameters* on the efficiency of the scheme *and* the runtime of *a variety of attacks* into consideration. In fact, the missing trust in the concrete hardness of LWE is one of the main problems to overcome to bring LWE-based schemes to practice.

This thesis aims at closing the gap between the theoretical knowledge of the hardness of LWE, and the concrete problem of selecting parameters for an LWE-based scheme. To this end, we analyze the existing methods to estimate the hardness of LWE, and introduce new estimation techniques where necessary. Afterwards, we show how to transfer this knowledge into instantiations that are at the same time secure and efficient. We show this process on three examples:

- A highly optimized public-key encryption scheme for embedded devices that is based on a variant of Ring-LWE.
- A practical signature scheme that served as the foundation of one of the best lattice-based signature schemes based on standard lattices.
- An advanced public-key encryption scheme that enjoys the unique property of natural double hardness based on LWE instances similar to those used for fully homomorphic encryption.

Zusammenfassung

Einer der Grundpfeiler unserer modernen digitalen Gesellschaft sind asymmetrische Verschlüsselungs- und Signaturverfahren. Asymmetrische Verfahren zeichnen sich dadurch aus, dass es nicht einen geheimen Schlüssel gibt, sondern ein Schlüsselpaar bestehend aus einem geheimen und einem öffentlichen Schlüssel. Der öffentliche Schlüssel erlaubt etwa das Verschlüsseln einer Nachricht oder das verifizieren einer Signatur, zum Entschlüsseln der Nachricht oder erzeugen der Signatur ist hingegen der geheime Schlüssel notwendig.

Allen diesen Verfahren gemein ist dass ihre Sicherheit auf einem numerisch schwer zu lösenden Problem beruht. In nahezu allen heute in der Praxis benutzen Verfahren ist dieses Problem entweder das Faktorisieren großer Zahlen, oder das diskrete Logarithmus Problem in verschiedenen Gruppen. Die auf diesen Problemen beruhenden Verfahren sind effizient und werden für sicher gehalten, haben jedoch ein großes Problem: Wie Peter Shor 1994 zeigte, können sie von Quantencomputern in polynomieller Zeit gelöst werden. In einer Zukunft, in der Quantencomputer existieren (was zufolge vieler Experten bereits in 20 Jahren der Fall sein könnte), sind auf sie also zum sichern vertraulicher Kommunikation ungeeignet.

Diese Arbeit befasst sich mit der Schwere des Learning With Errors (LWE) Problems. Auf LWE basierende Verfahren gehören zu den vielversprechendsten Kandidaten, um Faktorisierungs- und diskrete Logarithmus basierte Verfahren abzulösen. Sie sind sehr effizient und können nach dem aktuellen Stand der Forschung nicht von Quantencomputern angegriffen werden. Die Verfahren sind üblicherweise mithilfe eines Sicherheitsbeweises an LWE gebunden. Dieser besagt dass man eine bestimmte LWE Instanz lösen muss, um das Verfahren zu brechen.

Über die theoretische Schwere von LWE ist bereits viel bekannt. Wie Regev bereits 2005 zeigte, sind zufällige Instanzen von LWE (asymptotisch) mindestens so schwer wie die schwersten Instanzen verschiedener, etablierter Gitterprobleme. Leider sagen weder Regev's Resultat über die asymptotische Schwere von LWE, noch der Sicherheitsbeweis etwas darüber aus, wie schwer diese LWE Instanz ist. Folglich ist es von erheblicher Bedeutung für Theorie und Praxis, die Schwere konkreter LWE Instanzen zu untersuchen, und aus diesen Erkenntnissen korrekte Parameter für die LWE-basierten Verfahren abzuleiten.

Folglich ist diese Arbeit in zwei Abschnitte unterteilt. Der Erste präsentiert neue

theoretische und experimentelle Ergebnisse über die Schwere von LWE. Es besteht aus neuen Resultaten über die Schwere von LWE Instanzen unter Einschränkungen, die in der Praxis auftreten (Kapitel 3), eine experimentelle Untersuchung der Parallelisierbarkeit eines der vielversprechendsten LWE-Lösers (Kapitel 4), ein neuer Algorithmus zum Lösen von LWE (Kapitel 5), und eine neue Methode zum experimentellen Vergleich verschiedener LWE-Löser (Kapitel 6). Der zweite Abschnitt zeigt anhand dreier Beispiele, wie sich die Erkenntnisse über die Schwere von LWE nutzen lassen, um Parameter für kryptographische Verfahren zu wählen. Die Verfahren sind ein Signaturverfahren (Kapitel 7), ein Verschlüsselungsverfahren (Kapitel 8), und ein Verschlüsselungsverfahren mit erweiterten Sicherheitseigenschaften (Kapitel 9).

Contents

Publications

Publication Used in This Thesis

1. Johannes Buchmann, Christian Bischof, Özgür Dagdelen, Robert Fitzpatrick, Florian Göpfert and Artur Mariano: *Nearest Planes in Practice.* BalkanCryptSec 2014. Chapter 4. [BBD+14]

2. Florian Göpfert, Rachel Player, Thomas Wunderer: *On the Hardness of LWE with Binary Error: Revisiting the Hybrid Lattice-Reduction and Meet-in-the-Middle Attack.* Africacrypt 2016. Chapter 5. [BGPW16]

3. Johannes Buchmann, Niklas Büscher, Florian Göpfert, Juliane Krämer, Daniele Micciancio, Christine van Vredendaal, Michael Walter: *Creating Cryptographic Challenges Using Multi-Party Computation: The LWE Challenge.* PKC@AsiaCCS. Chapter 6. [BBG+16]

4. Özgür Dagdelen, Rachid El Bansarkhani, Florian Göpfert, Tim Güneysu, Tobias Oder, Thomas Pöppelmann, Ana Helena Sanchez, Peter Schwabe: *High-speed signatures from standard lattices.* Latincrypt 2014. Chapter 7. [DBG+14]

5. Johannes Buchmann, Florian Göpfert, Tim Güneysu, Tobias Oder, Thomas Pöppelmann: *High-Performance and Lightweight Lattice-Based Public-Key Encryption.* IoT@AsiaCCS. Chapter 8. [BGG+16]

6. Özgür Dagdelen, Sebastian Gajek, Florian Göpfert: *Learning with Errors in the Exponent.* ICISC 2015. Chapter 9. [DGG15]

Other Publications

1. Martin R. Albrecht, Robert Fitzpatrick, Florian Göpfert: *On the Efficacy of Solving LWE by Reduction to Unique-SVP.* ICISC 2013. [AFG13]

2. Christian Bischof, Johannes Buchmann, Özgür Dagdelen, Robert Fitzpatrick, Florian Göpfert, Artur Mariano, Bo-Yin Yang: *Tuning GaussSieve for Speed.* Latincrypt 2014. [FBB+14]

3. Johannes Buchmann, Daniel Cabarcas, Florian Göpfert, Andreas Hülsing, Patrick Weiden: *Discrete Ziggurat: A Time-Memory Trade-off for Sampling from a Gaussian Distribution over the Integers.* SAC 2013. [BCG+13]

4. Daniel Cabarcas, Denise Demirel, Jean Lancrenon, Florian Göpfert, Thomas Wunderer: *An Unconditionally Hiding and Long-Term Binding Post-Quantum Commitment Scheme.*

5. Daniel Cabarcas, Florian Göpfert, Patrick Weiden: *Provably secure LWE encryption with smallish uniform noise and secret.* AsiaPKC@AsiaCCS 2014. [CGW14]

1 Introduction

Public-key cryptography is ubiquitous in our modern IT infrastructure. Nearly all of today's digital security solutions are either based on the problem of factoring big integers, or solving the discrete logarithm problem in certain groups. While those schemes are widely used and believed to be secure now, they come with an expiration date: In 1994, Peter Shor [Sho97] showed in a ground-breaking work that quantum computers can solve the factoring and the discrete logarithm problem in polynomial time, rendering all schemes based on those problems insecure. While no large-scale quantum computer exist now, experts agree that they are likely to exist in a not too distant future. Investigating alternatives that can resist attacks on quantum computers is therefore not only of theoretical, but also of practical interest.

At this point in time, there are five main research areas aiming to replace today's public-key cryptography: Hash-based, code-based, isogeny-based, lattice-based, and multivariate cryptography. This work is located in the field of lattice-based cryptography. This field gained a lot of interest in the last decade, since it combines many desirable features: It comes with good (quasilinear) asymptotic key sizes, good concrete runtimes and key sizes, allows worst-case secure instantiations, and several advanced cryptographic primitives that where believed to be impossible before. Virtually all modern lattice-based primitives are based on one of two problems: The Learning With Errors (LWE) problem, or the Short Integer Solution (SIS) problem.

While SIS is the older of the two problems, LWE can be used in more applications. In a nutshell, SIS allows "minicrypt" (i.e., signature schemes and hash functions), while LWE can also be used to build "cryptomania" applications (i.e., public key encryption and more). However, LWE also serves as the security foundation of recent signature schemes [ABBD15, ABB$^+$16]. Consequently, this thesis focuses on the hardness of LWE. A short excursion about the hardness of SIS is given in Chapter 7. Since the introduction of LWE by Regev [Reg05], a big effort was made by the community to investigate its full potential. Roughly speaking, the literature can be split into three lines of work:

First, many interesting results show that it is possible to build many different cryptographic applications based on LWE. This started with the public-key encryption scheme presented in Regev'a original work, and is still a growing field. Nowadays, there is hardly any big cryptographic conference without any new or improved

construction based on LWE.

The second line of work started with Regev's worst-case to average-case quantum reduction from LWE to well-established lattice problems (namely the Short Independent Vector Problem, SIVP, and the decisional Shortest Vector Problem, GapSVP). Since then, several researchers gave new and improved worst-case hardness results for LWE. Just to mention two of them, is is known today that LWE is worst-case hard under a classical reduction [BLP+13], and with uniform error [MP13]. Those works establish the theoretical and asymptotic hardness of LWE.

This thesis is part of the third line of work, dealing with the concrete hardness of LWE. Over the last decade, a variety of attacks on LWE have been proposed, and many of them are of independent interest. From a constructive point of view, this work is important since the runtime of the best known attack tells us the concrete hardness of LWE (in other words: the hardness of concrete LWE instances). This knowledge is crucial to instantiate the schemes based on LWE correctly. Unfortunately, transferring knowledge of the complexity of LWE attacks into concrete parameter proposals for a given scheme is not trivial, and one of the obstacles that prevent LWE-based schemes from a broader propagation.

In this thesis, we address this topic by showing ways to securely instantiate cryptographic primitives that are based on LWE. This includes new runtime estimations of existing attacks (Chapter 3 and 4), a new attack on an important subproblem of LWE (Chapter 5), a new way to monitor the current state of practical LWE solvers (Chapter 6), and several examples of correct instantiations of cryptographic primitives (Chapter 7, 8, and 9.

The goal of our parameter selection is to construct schemes that are not only secure, but also efficient. In this case, efficiency is an umbrella term that comprises memory consumption (i.e., the sizes of keys, signatures/ciphertexts, and of the source code), and speed of the algorithms (i.e., the key generation, encryption/signing, and decryption/verification). Depending on the scenario, one or more of these aspects may be more important than the others. We explain our efficiency goal in the individual chapters.

1.1. Organisation

This thesis evaluates the concrete hardness of the Learning with Errors (LWE) problem and shows how to select parameters for LWE-based schemes to make them at the same time secure and efficient. To this end, we show how to overcome problems that appear while solving LWE and thereby obtain methods to estimate the hardness of LWE instances. Furthermore, we show that LWE with binary error can be used to obtain very efficient schemes. Finally, we use the results to instantiate LWE-based schemes.

Chapter 2 introduces the necessary background and notation. This includes back-

ground about lattices and the most important arithmetic lattice problems: LWE, the Shortest Vector Problem (SVP), the Closest Vector Problem (CVP), and relaxed variants. We focus on LWE and conclude the chapter with an overview of the best LWE solvers, including known ways to estimate their runtime on a given LWE instance.

The scientific contributions of this thesis are split in two parts. Part I presents new theoretic and practical hardness results for the LWE problem. In a nutshell, we investigate how a limited number of samples influences the hardness of LWE (Chapter 3), investigate influences of parallel computing (Chapter 4), analyze an important variant of LWE (Chapter 5), and present first results of a new challenge on LWE (Chapter 6).

Despite the fact that the estimations presented in Chapter 2 are well-accepted in the community, they ignore an important issue that influences the runtime significantly: there are often not enough LWE samples to run the attack in the optimal dimension. In Section 3.1, we show that restricting the number of samples influences the performance of the attacks significantly. This is used to create concrete values for the hardness of LWE with few samples in Section 3.2.

Furthermore, a short glimpse in the history of established cryptographic problems (like factoring or discrete logarithms) shows that it is important to evaluate which algorithms can benefit from parallel computing. For example, records on RSA factoring challenges achieved with parallel computing contributed a lot to our understanding of the security of RSA [KAF+10]. Chapter 4 addresses this issue by presenting a parallel version of the decoding attack [LP11] and discussing the influence of further improvements in this direction.

In his original paper [Reg05], Regev proposed LWE with an arbitrary error distribution. However, his famous worst to average-case reduction was only applicable for Gaussian distributed error. On the other hand, replacing a Gaussian with a simpler distribution (like the uniform distribution on $\{0, 1\}$) leads to much more efficient schemes, (i.e., schemes with smaller memory footprint and faster run times [BGG+16]). Chapter 5 deals with the hardness assessment of LWE with binary error. First, Section 5.1 presents a new attack on binary LWE. The rest of the chapter compares this attack with existing approaches and shows that it outperforms all other attacks on certain binary LWE instances that are important in practice.

In the last chapter of Part I, we focus on the experimental performance of LWE solvers. To this end, we have set up an LWE challenge web page[1]. This challenge solves two main purposes: on the one hand, it allows cryptanalytic experts to prove the efficiency of their attacks by breaking instances provided by the challenge. On the other hand, it collects information about the successful attacks and presents them in a easily accessible way. Chapter 6 shows that the instances provided rep-

[1] `https://www.latticechallenge.org/lwe_challenge/challenge.php`

resent instances that are important in practice (Section 6.1), explains the set-up of the web page (Section 6.2), and gives results extracted from the first successful submissions (Section 6.3).

Part II focuses on constructing and instantiating concrete schemes. More precisely, we show how to select parameters for a signature scheme on standard lattices (Chapter 7), an encryption scheme on a special instance of LWE (Chapter 8), and an encryption scheme with additional security guarantees (Chapter 9). The hardness estimations used for the instantiation are based on Part I as well as on other hardness results.

The first scheme is the signature scheme by Bai/Galbraith [BG14b], that is introduced in Section 7.1. In Section 7.2, we revisit the parameters proposed in the original work [BG14b] and show that the security of this instantiation was massively overestimated. We address this issue in Section 7.3 by introducing a new parameter set that leads to an instantiation that is at the same time secure and allows fast encryption and decryption [DBG+14].

The second scheme is a highly efficient variant of the encryption scheme by Lindner and Peikert [LP11]. Section 8.1 introduces a framework that covers both, the original scheme and our high speed variant. Our instantiation is based on ring LWE with binary errors. Since the hardness of LWE depends massively on the size of the errors, instantiating LWE with binary errors looks much easier at first glance. However, it is well known that the hardness of LWE does not depend on the absolute error size $\|\mathbf{e}\|$, but on the relative error size $\|\mathbf{e}\|/q$. This means that decreasing the modulus q increases the hardness of LWE. In practice, however, correctness requirements for the scheme typically lead to a lower bound for the modulus. As we show in Section 8.2, using binary errors decrease this bound and allow to decrypt correctly with a significantly smaller modulus. Finally, Section 8.3 presents a parameter set that leads to a secure, correct and highly efficient encryption scheme. The main goal of the instantiation is to minimize the memory footprint, and an implementation by Buchman et al [BGG+16] shows that it also leads to reasonably fast runtimes.

In Chapter 9, we presents Learning With Errors in the Exponent (LWEE), a generalization of LWE that combines the hardness of learning with errors with the hardness of the Representation Problem (RP), a well-studied number theoretical problem. The necessary background on RP is given in Section 9.1, and the new assumption is introduced in Section 9.2. Afterwards, in Section 9.3, we present a new encryption scheme that is based on LWEE and propose concrete parameters. The security of these concrete instantiations is presented in Section 9.4.

2 Background

This chapter introduces the necessary background for the rest of the thesis. We start with some general properties of lattices (Section 2.1), introduce the computational problems (Section 2.2), and finally give an overview of existing LWE solvers (Section 2.3).

2.1. Lattices

For a positive integer n, we define $[n] = \{1, 2, \ldots, n\}$. We denote vectors by bold lower-case letters and matrices by bold upper-case letters. For an integer $c \in \mathbb{Z}$, let $[c]_{2^d}$ be the unique integer in the set $(-2^{d-1}, 2^{d-1}]$ such that $c \equiv [c]_{2^d} \mod 2^d$ which is basically extraction of the least significant bits. For $c \in \mathbb{Z}$, let $\lfloor c \rceil_d = (c - [c]_{2^d})/2^d$ drop the d least significant bits. Both operators can also be applied to vectors by applying it coordinate-wise to its coefficients.

2.1.1. Lattices and Bases

A lattice Λ is a discrete subgroup of the euclidian space $\mathbb{R}^m$. Lattices can be defined as the integer span by linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, where n is called the dimension of the lattice. For the rest of this work we restrict our studies to lattices in $\mathbb{R}^m$ whose dimension is maximal, e.g., m, which are called full-ranked lattices. Note that a basis $\mathbf{B}$ for a lattice Λ is not unique. Indeed, any unimodular transformation of $\mathbf{B}$ results in a different basis for the same lattice Λ. The lattice $\Lambda(\mathbf{B})$ is defined by all integer combinations of elements of $\mathbf{B}$, i.e.,

$$\Lambda(\mathbf{B}) = \left\{ \mathbf{x} \in \mathbb{R}^m \mid \exists \alpha_1, \ldots, \alpha_n \in \mathbb{Z} : \mathbf{x} = \sum_{i=1}^{n} \alpha_i \mathbf{b}_i \right\} .$$

A very important invariant for a lattice is its determinant. The determinant of a full-ranked lattice $\Lambda(\mathbf{A})$ is defined as

$$\det(\Lambda(\mathbf{A})) = |\det(\mathbf{A})|.$$

It is well known that the determinant of a lattice is well-defined (i.e. does not depend on the particular basis) and the definition can be generalized for lattices that are not full-ranked. In this case, the determinant is given by $\det(\Lambda(\mathbf{A})) = \sqrt{\det(\mathbf{A}^T\mathbf{A})}$.

Other important invariants of a lattice are the successive minima $\lambda_i((\Lambda))$. The ith successive minimum $\lambda_i(\Lambda)$ is defined as the smallest radius r such that there exist i linearly independent lattice vectors of norm at most r. Throughout this thesis, the successive minima will always refer to the Euclidian norm. The successive minima differ from the determinant in one fundamental aspect: While there are efficient ways to calculate $\det(\Lambda)$ given an arbitrary lattice basis, calculating $\lambda_i(\Lambda(\mathbf{B}))$ for a given basis $\mathbf{B}$ is typically hard (unless $\mathbf{B}$ is a high quality basis, see Section 2.1.2). However, there is an established way to estimate $\lambda_1(\Lambda)$, given $\det(\Lambda)$. It is called the Gaussian heuristic and estimates the length of the shortest lattice vector in an n-dimensional lattice Λ via

$$\lambda_1(\Lambda) \approx \frac{\Gamma(1+n/2)^{1/n}}{\sqrt{\pi}} \det(\Lambda)^{1/n}.$$

In this thesis, we are particularly interested in modular integer lattices. These are also the lattices one considers when solving LWE instances. A modular (or q-ary) lattice, for a given $q \in \mathbb{N}$, is a full-ranked lattice Λ such that $q\mathbb{Z}^m \subseteq \Lambda \subseteq \mathbb{Z}^m$. Such modular lattices are often given by a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ via

$$\Lambda_q(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m \mid \exists \mathbf{y} \in \mathbb{Z}^n : \mathbf{x} = \mathbf{A}\mathbf{y} \mod q\} \tag{2.1}$$

or

$$\Lambda_q^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = 0 \mod q\} \tag{2.2}$$

It is important to note that given a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$, it is easy to find a basis of $\Lambda_q(\mathbf{A})$ (see e.g., [AFG13]). With high probability, the determinant of a q-ary lattice is given by $\det(\Lambda_q(\mathbf{A})) = q^{m-n}$ if $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$.

An other important concept from linear algebra is the Gram-Schmidt basis. For a set of column vectors $\mathbf{B} \in \mathbb{Z}^{m\times n}$, we write $\pi_{\mathrm{span}(\mathbf{B})}(\mathbf{t})$ for the projection of the vector $\mathbf{t}$ onto the span of the vectors of $\mathbf{B}$, i.e., $\pi_{\mathrm{span}(\mathbf{B})}(\mathbf{t}) = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T \cdot \mathbf{t}$. The Gram-Schmidt orthogonalization $\widetilde{\mathbf{B}} = \{\widetilde{\mathbf{b}}_1, \ldots, \widetilde{\mathbf{b}}_n\}$ of a basis $\mathbf{B}$ is defined through $\widetilde{\mathbf{b}}_i = \mathbf{b}_i - \pi_{\mathrm{span}(\mathbf{b}_1,\ldots,\mathbf{b}_{i-1})}(\mathbf{b}_i)$ for $i \in [n]$. Note that the Gram-Schmidt basis is typically not a basis of the lattice, but nevertheless important for many lattice algorithms (see Section 2.3.3).

2.1.2. Basis Quality and Basis Reduction

While every at least two-dimensional lattice is generated by infinitely many bases, some are clearly more favorable than others. The most common measure of the

quality of a basis is the Hermite factor. We say that a basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ of an m-dimensional lattice Λ with determinant $\det(\Lambda)$ has Hermite factor δ if

$$\|\mathbf{b}_1\| \approx \delta^m \det(\Lambda)^{1/m}.$$

At first sight, this does not look like a meaningful definition, since it only depends on the first vector of the basis. However, we can use this information to get information about the other basis vectors. The well-established way to do this is the Geometric Series Assumption (GSA). It predicts the length of the ith Gram-Schmidt basis vector to be

$$\|\widetilde{\mathbf{b}}_i\| \approx \alpha^{i-1} \|\mathbf{b}_1\| \tag{2.3}$$

for a positive constant $\alpha < 1$. Despite the fact that it is possible to construct bases that do at all not follow the GSA, it proved to provide a good approximation for bases appearing in practice [LP11]. Together with Equation 2.4, and the fact that $\prod_{i=1}^{m} \|\widetilde{\mathbf{b}}_i\| = \det(\Lambda)$, this leads to

$$\|\widetilde{\mathbf{b}}_i\| \approx \delta^{-2(i-1)+m} \det(\Lambda)^{1/m}. \tag{2.4}$$

Throughout this thesis, we refer to a good basis as a basis with a small Hermite factor. The GSA implies that with increasing basis quality, the lengths of the first Gram-Schmidt vectors decrease, while the lengths of the last ones increase. This can be seen as a sign that in a good basis, the basis vectors are "more orthogonal" to each other than in a bad basis. Current estimations claim that $\delta = 1.01$ can be reached in practice, while $\delta = 1.007$ is out of reach for a foreseeable future.

It is considered hard to give more precise estimations for the effort necessary to achieve a given Hermite delta. The reason for this is that there is a significant gap between what behavior we can prove for basis reduction (for example, what upper bound we can prove for the vectors returned by a basis reduction algorithm), and the observed average behavior (for example, the average lengths of the returned vectors). This gap already shows up for the basic LLL algorithms: According to Gama et al. [GN08], LLL provably serves as SVP oracle in dimension two only. In practice, however, it can be used as SVP oracle up to dimension 35.

This behavior also appears in more advanced basis reduction algorithms like BKZ. When estimating the hardness of LWE instances, this causes big problems: the provable results can only be used to derive *upper* bounds on the effort necessary to break a certain LWE instance. However, we are interested in lower bounds, since this carries over to lower bounds on the security of our schemes.

To overcome this obstacle, researcher developed different techniques. Most of them consist of theoretical results about the asymptotic complexity of basis reduction algorithms, combined with experimental results about basis reduction on lattices in small and moderate dimensions. In the end, they typically merge to a lower bound on the number of operations necessary to achieve a certain hermite delta. In

this work, we use the well-established equation introduced by Richard Lindner and Chris Peikert [LP11]. They estimate the number of operations necessary to achieve a given hermite delta δ by

$$\mathrm{ops}_{\mathrm{BKZ}}(\delta) = 2^{1.8/\log_2(\delta)-110} \cdot 2.3 \cdot 10^9. \tag{2.5}$$

2.2. Computational Problems

2.2.1. Short Vector Problems (SVP)

The probably oldest and most-studied class of lattice problems are short vector problem. For all those problems, the task is to find one or more short lattice vectors, given a bases of a lattice Λ. The most basic problem is the Shortest Vector Problem, which asks to find a shortest non-zero lattice vector, given a basis of a "random" lattice. However, this problem plays a minor role for most modern applications.

More important from a practical point of view is a variant called the "unique Shortest Vector Problem" (uSVP). The difference to SVP is that in uSVP, we know in advance that $\lambda_2(\Lambda) > \alpha\lambda_1(\Lambda)$ for a fixed factor $\alpha > 1$. In order to make this factor explicit, we also write $\alpha-$uSVP. Experiments show that uSVP gets easier with increasing "gap" α. Since SVP can be seen as 1-uSVP, this implies that uSVP is easier than SVP.

A detailed analysis of the computational complexity of uSVP is given by Albrecht et al. [AFG13]. They claimed that the attack succeeds with high probability if

$$\frac{\lambda_2(\Lambda)}{\lambda_1(\Lambda)} \geq \tau\delta^m,$$

where $\tau \approx 0.4$ is a constant depending on the unique-SVP solver used. Applying the Gaussian heuristic shows that the attack succeeds if

$$\tau\delta^m \leq \frac{\frac{\Gamma(1+n/2)^{1/m}}{\sqrt{\pi}}\det(\Lambda)^{1/m}}{\lambda_1(\Lambda)}.$$

This equation can be simplified using Sterling's approximation. Sterling approximated the factorial of a natural number x via

$$x! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.$$

Applying this for $\Gamma(1 + m/2) = (m/2)!$ leads to

$$\begin{aligned}\Gamma(1+\frac{m}{2})^{1/m} = \left(\left(\frac{m}{2}\right)!\right)^{1/m} &= \left(\sqrt{2\pi\frac{m}{2}}\left(\frac{\frac{m}{2}}{e}\right)^{\frac{m}{2}}\right)^{1/m} = \left(\sqrt{2\pi\frac{m}{2}}\right)^{1/m}\left(\frac{m}{2e}\right)^{\frac{1}{2}} \\ &\approx \sqrt{m/(2e)}.\end{aligned}$$

Consequently, we assume that an algorithm solves uSVP correctly if

$$\tau\delta^m \leq \frac{\sqrt{m}\det(\Lambda)^{1/m}}{\sqrt{2\pi e}\lambda_1(\Lambda)}. \tag{2.6}$$

Sections 2.3.4 and 2.3.5 show how to construct uSVP instances from LWE instances. A detailed analysis of the complexity of those instances is given in Sections 3.1.3 and 3.1.4.

2.2.2. Short Integer Solution (SIS)

In the last decade, researchers developed many tools that allow easier constructions of secure lattice-based cryptographic schemes. A major part of this are the intermediate problems LWE and SIS. Those problems have been proven to be hard in the average-case, as long as certain underlying lattice problems are hard in the worst case. Consequently, a scheme based on the average-case hardness of LWE or SIS inherits the worst-case hardness.

The Short Integer Solution (SIS) problem is the oldest of the intermediate problems and is mainly used to build signature schemes. The most prominent example of such a signature scheme is BLISS [DDLL13]. In this thesis, SIS is mainly important for the security analysis of the Bai/Galbraith signature scheme (see Chapter 7). For parameters $n, m, q \in \mathbb{Z}$ and $\nu \geq 0$, $\mathrm{SIS}_{n,m,q,\nu}$ is the problem of finding a nonzero vector $\mathbf{u} \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{u} \equiv 0 \mod q$ and $\|\mathbf{u}\| \leq \nu$. If not stated otherwise, the norm applied in the definition of the SIS problem is the Euclidean norm.

2.2.3. Learning with Errors (LWE)

The second intermediate problem is LWE. While SIS is mainly used to build signature schemes, LWE allows the construction of public-key encryption schemes. Furthermore, several modern signature schemes (e.g., the line of work following TESLA [ABBD15]) are also based on the hardness of LWE. In some sense, one can say that LWE is more powerful than SIS.

Definition 1 (LWE Distribution)**.** *Let n, q be positive integers, χ be a probability distribution on $\mathbb{Z}_q$, and $\mathbf{s} \in \mathbb{Z}_q^n$. We denote by $L_{\mathbf{s},\chi}^{(n)}$ the probability distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing $\mathbf{a}$ from the uniform distribution on $\mathbb{Z}_q^n$, choosing e according to χ, and returning $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.*

For parameters $n, m, q \in \mathbb{Z}$ and a probability distribution χ on $\mathbb{Z}_q$, Search-$\mathrm{LWE}_{n,m,q,\chi}$ is the problem of finding $\mathbf{s} \in \mathbb{Z}_q^n$ given m pairs $(\mathbf{a}_i, c_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ sampled according to $L_{\mathbf{s},\chi}^{(n)}$ for $\mathbf{s} \xleftarrow{\$} \chi^n$. While LWE is defined for arbitrary distributions χ, there are few very important special cases. The most common case is to use a discrete or discretized Gaussian distribution.

Definition 2. *For a positive parameter $\sigma \in \mathbb{R}$, we write D_σ for the discrete Gaussian distribution over the integers. It is defined by*

$$\Pr[D_\sigma = x] \sim \exp\left(-\frac{1}{2\sigma^2}\right).$$

Slightly abusing notation, we also write D_σ for the Gaussian distribution over $\mathbb{Z}_q$ which is defined by sampling from $\mathbb{Z}$ and taking the result modulo q. To simplify notation, we write Search-LWE$_{n,m,q,\sigma}$ for Search-LWE$_{n,m,q,D_\sigma}$ throughout this paper. A second important case is binary LWE. It is defined by Search-binLWE = Search-LWE$_{n,m,q,\mathcal{U}_{\{0,1\}}}$, where $\mathcal{U}_{\{0,1\}}$ denotes the uniform distribution on $\{0,1\}$.

LWE plays a major role in lattice-based cryptography, since it allows the construction of all sorts of cryptographic algorithms. The arguably most important primitives in asymmetric cryptography are signature schemes and encryption schemes, and LWE allows efficient constructions of both of them. In this thesis, we discuss parameter selection for a signature scheme (Chapter 7) and two encryption schemes (Chapter 8 and 9).

In fact, there is a third important line of work, namely key exchange protocols. All LWE-based key exchange protocols follow basically the original construction by Jintai Ding [Din12]. LWE instances used for practical instantiations are very similar to instances used for encryption schemes. Consequently, the techniques introduced in this work can also be used for good instantiations of Ding-like key exchange protocols, including new constructions like Peikert's key exchange [Pei14] and the new hope protocol [ADPS16]. We want to point out that this is in contrast to signature schemes, that typically require significantly different parameters (e.g., a substantially bigger modulus) and are therefore addressed independently.

2.3. Algorithms for LWE

This section gives an overview of the best known algorithms to solve LWE. Section 2.3.1 treats the BKW algorithm by Blum, Kalai, and Wassermann [BKW03], a very direct way to solve LWE. An alternative is to interpretate LWE as an instance of the Bounded Distance Decoding (BDD) problem, see Section 2.3.3. There are also indirect ways to solve LWE. Those are to either reduce it to the unique Shortest Vector Problem (uSVP) (see Section 2.3.4), or to reduce it first to the inhomogeneous Short Integer Solution (iSIS) problem and then to uSVP (see Section 2.3.5).

The above attacks can be split into two different areas. On the one hand are the more algebraic attacks, namely BKW and the attack by Arora and Ge [AG11]. An important characterization of these attacks is that they do not use basis reduction as subroutine. This is a desirable feature, since basis reduction algorithms are hard to analyze (see Section 2.1.2). Consequently, theory can predict the behavior of those attacks very well, and the theoretic analysis give very good runtime estimations.

On the other hand, we have lattice-based attacks. They can be characterized as the attacks that utilize basis reduction. Consequently, the gap between theory and practice for basis reduction algorithms carries over to lattice-based LWE attacks. We deal with this problem by using the well-established extrapolation methods to give lower bounds on the capabilities of basis reduction algorithms introduced in Section 2.1.2. Consequently, the analysis of those attacks are of theoretical nature, but extended with extrapolations of experimental data.

It is important to note that all runtime estimates presented in this work are valid for classical computers. While LWE is believed to be hard, this does not mean that quantum computers can not provide any speed-up to known algorithms. This statement only implies that there is no efficient (i.e., polynomial-time) quantum algorithm known for LWE. So far, the existence of such an algorithm is considered to be unlikely.

In fact, the only known way to speed up solvers for post-quantum hard problems is to apply Grover's algorithm. Grover's algorithm is an quantum algorithm for finding elements in an unstructured search space. It's advantage is that it can find a wanted element in runtime roughly $\mathcal{O}(\sqrt{n})$, while classical algorithms require at least $\mathcal{O}(n)$. While this speedup is by far not enough to break LWE-based crypto completely, it is still too big to be ignored.

Grover's algorithm can speed up certain subroutines of classical LWE attacks, potentially resulting in an quantum attack that outperforms classical approaches. Identifying the attacks that can benefit from this speed-up is content of current research. However, this research requires detailed knowledge of the classical algorithm. Due to the modular nature of the approaches presented in this work, new results about quantum speed-ups can be easily included by replacing runtime estimates of the affected subroutines by the estimations of their quantum counterparts. Promising research directions considering quantum speed-ups are given in the conclusion (Chapter 10).

In many scenarios, Grover can be used to speed up a brute-force subroutine. Some recent developments show that in fact, several lattice solvers (mainly those using some sieving subroutine) can benefit from this speed-up. However, the impact on the concrete hardness of LWE seems to be limited until know, and is not considered in this work.

2.3.1. The Blum Kalai Wasserman Algorithm

In 2003, Blum, Kalai and Wasserman [BKW03] published a new attack on the Learning with Parity Noise (LPN) problem, a predecessor of LWE. The attack was generalized and applied to LWE by Albrecht et al. in 2013 [ACF+15]. Improvements for LWE with particularly small secret [AFFP14] and by applying multidimensional Fourier transforms [DTV15] followed.

The main drawback of BKW at the moment is that it requires too many samples

a	b	$\log_2$(samples)
1	256	3221
2	128	1611
4	64	805
8	32	403
16	16	201
32	8	81026
64	4	347965698915015

Table 2.1.: Number of samples required to break an LWE instance with $n = 256$, $q = 4093$, $\sigma = 8.35/\sqrt{2\pi}$

to be applied to real-world LWE instances. Consequently, BKW only plays a minor role throughout this thesis. This section introduces the main idea of BKW, and aims at giving an intuitive understanding of why BKW is not practical in its current state. For details, we refer to the recent research papers [AFFP14, DTV15].

BKW is an algebraic attack on LWE. We do not give a formal explanation of BKW here, interested readers are referred to [ACF+15, DTV15]. In the most recent work on BKW, Duc et al. [DTV15] also show how to apply BKW on LWE instances with "few samples". They claim that a successful attack requires at least

$$\max\left(8 \cdot b \log(aq)\left(1 - \frac{2\pi^2\sigma^2}{q^2}\right)^{-2^a}, \frac{3}{2}q^b\right) \tag{2.7}$$

samples, where a, b are attack parameters satisfying $a \cdot b = n$. Obviously, the choice of a and b can be used to tune the trade-off between the two terms. The key observation is that the second term is exponential in b, while the second term is super exponential in a. Consequently, the whole formula is at least exponential in $\sqrt{n}$.

Besides this theoretical (asymptotic) argument, inserting concrete values into Equation 2.7 supports the assumption that BKW requires too many samples to be practical. Table 2.1 shows the necessary number of samples for an LWE instance proposed by Lindner and Peikert [LP11] for LWE-based encryption. Keep in mind that this numbers take the new technique that allows to run BKW on few samples (at the cost of additional memory requirements) into account.

2.3.2. The Distinguishing Attack

One of the oldest attacks on LWE is the so-called "distinguishing attack". It is easy to understand, shows a direct relation between LWE and aSVP in certain lattices, is easy to anlyse, and can be used in combination with any aSVP algorithm. Despite

Lindner and Peikert's claim that it is outperformed by the decoding attack [LP11], recent improvements on lattice sieving algorithms lead to runtime estimations that show that it could actually be the most promising approach on certain instances. Those arguments are somewhat debatable (since they ignore memory consumption), but still show that one must not ignore the distinguishing attack.

The goal is to distinguish whether the vector $\mathbf{b}$ comes from an LWE distribution or was sampled uniformly random in $\mathbb{Z}_q^m$, given $\mathbf{A}$ and $\mathbf{b}$. In order to do this, the algorithm creates a small vector in the dual lattice

$$\Lambda_q(\mathbf{A})^\perp = \{\mathbf{v} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{v} = 0 \mod q\}. \tag{2.8}$$

Afterwards, it checks whether $|\langle \mathbf{v}, \mathbf{b} \rangle|$ is small or not. To see why, assume $\mathbf{b}$ was sampled uniformly random. In this case, $\langle \mathbf{v}, \mathbf{b} \rangle$ is uniformly random in $\mathbb{Z}_q$ and in general not small. Now, assume $\mathbf{b}$ comes from an LWE distribution. This leads to $\langle \mathbf{v}, \mathbf{b} \rangle = \langle \mathbf{e}, \mathbf{b} \rangle$, which is small if $\mathbf{v}$ is small enough.

The common instantiation of this attack is to represent the inner product as an integer between $-q/2$ and $q/2$ and output "LWE" if the absolute value is smaller than $q/4$, and "uniform" otherwise. The effort necessary to compute a vector of a certain length can be estimated with Equation 2.5 or similar results. Lindner and Peikert [LP11] gave an analysis for the attack on LWE with Gaussian error, an analysis for LWE with binary error is given in Chapter 5.

2.3.3. The Decoding Attack

A very natural way to attack LWE is based on viewing it as a special instance of CVP. To this end, consider the lattice $\Lambda_q(\mathbf{A})$ as defined in Section 2.1.1. Since $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \mod q$, we know that there is an integer vector $x \in \mathbb{Z}^m$ such that $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} + q\mathbf{b}$. It is easy to verify that $\mathbf{A}\mathbf{s} \in \Lambda_q(\mathbf{A})$ and $q\mathbf{x} \in \Lambda_q(\mathbf{A})$. Since a lattice is an additive subgroup, also $\mathbf{A}\mathbf{s} + q\mathbf{x} \in \Lambda_q(A)$, and consequently $\mathbf{b}$ lies within distance $\|\mathbf{e}\|$ from a lattice point. Since $\mathbf{e}$ is typically very small, solving the CVP problem in the lattice $\Lambda_q(\mathbf{A})$ with target vector $\mathbf{b}$ returns the target vector $\mathbf{A}\mathbf{s} + q\mathbf{x}$, from which $\mathbf{s}$ can be recovered easily.

The standard approach to solve CVP (and its variants) is to use basis reduction followed by a decoding algorithm. Since basis reduction algorithms have been introduced before, we focus on the second step here. The classical decoding algorithm is the nearest-plane algorithm introduced by Babai [Bab86]. A generalization was presented by Lindner and Peikert in 2011 [LP11]. On a lattice basis $\mathbf{B}$ and a target vector $\mathbf{t}$ as input, it returns a lattice vector that is somewhat close to $\mathbf{t}$.

To understand the exact output guarantees, it is necessary to introduce the fundamental parallelepiped. For a basis $\mathbf{B} \in \mathbb{Z}^{m \times n}$, the fundamental parallelepiped is defined as

$$\mathcal{P}(\mathbf{B}) = \left\{ \mathbf{v} = \sum_{i=1}^{m} \alpha_i \mathbf{b}_i \mid \forall i \in [m] : 0 \leq \alpha_i < 1 \right\},$$

Algorithm 1: Nearest Plane

Input : $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\} \subset \mathbb{R}^m$, // the lattice basis
$\widetilde{\mathbf{B}} = \left\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_m\right\}$, // Gram-Schmidt basis of $\mathbf{B}$
$k \in \mathbb{Z}$, // current subdimension satisfying $k \leq m$
$\mathbf{t} \in \mathbb{R}^m$ // the target vector

Output: A lattice vector $\mathbf{v}$ such that $\mathbf{t} - \mathbf{v} \in \mathcal{P}_{1/2}(\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_{k-1}\})$

```
begin
    if k = 0 then
        return 0
    end
    else
        t ← π_{span({b_1,...,b_k})}(t)                    // optional step
        Let c be the integer closest to ⟨b̃_k, t⟩ / ⟨b̃_k, b̃_k⟩;
        return (c · b_k + NearestPlanes(B, B̃, k − 1, t − c · b_k))
    end
end
```

and the shifted fundamental parallelepiped as

$$\mathcal{P}_{1/2}(\mathbf{B}) = \left\{ \mathbf{v} = \sum_{i=1}^{m} \alpha_i \mathbf{b}_i \mid \forall i \in [m] : -\frac{1}{2} \leq \alpha_i < \frac{1}{2} \right\} .$$

This notation is used for lattice bases as well as for Gram-Schmidt bases. Note that the orthogonality of Gram-Schmidt vectors implies that the corresponding fundamental parallelepipeds are m-dimensional rectangles.

For a vector space U, let $\pi_U(\mathbf{t})$ be the orthogonal projection of $\mathbf{t}$ on U. When called on target vector $\mathbf{t} \in \mathbb{R}^m$ and a lattice basis $\mathbf{B} \in \mathbb{Z}^{m \times m}$, the (polynomial-time) algorithm nearest plane (see Algorithm 1) returns the unique lattice point $\mathbf{v}$ such that $\mathbf{t} - \mathbf{v}$ lies in the shifted Gram-Schmidt fundamental parallelepiped $\mathcal{P}_{1/2}(\widetilde{\mathbf{B}})$. Consequently, when called on an LWE instance $\mathbf{b} = \mathbf{As} + \mathbf{e}$ with a basis $\mathbf{B}$ of $\Lambda_q(\mathbf{A})$, nearest plane returns the desired vector $\mathbf{v} \in \mathbb{Z}^m$ such that $\mathbf{v} = \mathbf{As}$ if (and only if) $\mathbf{e} \in \mathcal{P}_{1/2}(\widetilde{\mathbf{B}})$. This shows that the success probability of nearest plane heavily depends on the quality of the input basis: a bad input basis comes with a "long and skinny" Gram-Schmidt fundamental parallelepiped and consequently with a small success probability.

Consequently, already the basic decoding attack (basis reduction on $\Lambda_q(\mathbf{A})$, followed by nearest plane) comes with a trade-off between runtime and success probability: increasing the time spend on basis reduction increases the success probability, while decreasing it leads to a smaller success probability.

In 2011, Lindner and Peikert [LP11] presented nearest planes, a generalisation of

Algorithm 2: NearestPlanes

Input : $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\} \subset \mathbb{R}^m$, // the lattice basis
$\widetilde{\mathbf{B}} = \left\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_m\right\}$, // Gram-Schmidt basis of $\mathbf{B}$
$k \in \mathbb{Z}$, // current subdimension satisfying $k \leq m$
$\mathbf{d} \in (\mathbb{Z}^+)^m$, // recursive branching vector
$\mathbf{t} \in \mathbb{R}^m$ // the target vector

Output: A lattice vector $\mathbf{v}$ such that $\mathbf{t} - \mathbf{v} \in \mathcal{P}^{\mathbf{d}}_{1/2}(\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_{k-1}\})$

begin
  if *k = 0* **then**
    return 0
  end
  else
    $\mathbf{t} \leftarrow \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t})$ // optional step
    Let $\{c_1, \ldots, c_{d_k}\} \in \mathbb{Z}^{d_k}$ be the distinct integers closest to $\frac{\langle \tilde{\mathbf{b}}_k, \mathbf{t} \rangle}{\langle \tilde{\mathbf{b}}_k, \tilde{\mathbf{b}}_k \rangle}$;
    return $\bigcup_{i \in [d_k]} \left(c_i \cdot \mathbf{b}_k + \mathrm{NearestPlanes}(\mathbf{B}, \widetilde{\mathbf{B}}, k-1, \mathbf{d}, \mathbf{t} - c_i \cdot \mathbf{b}_k) \right)$
  end
end

nearest plane that provides the attacker with additional power. Instead of calculating one coefficient c in every iteration, nearest planes (see Algorithm 2) calculates several c_i and recurses on all of them. The attacker can choose the number of coefficients for every step by an additional attack parameter $\mathbf{d} \in (\mathbb{Z}^+)^m$.

The output guarantee of nearest planes can easily be formulated with the following generalization of the shifted Gram-Schmidt rectangle: for a basis $\mathbf{B} \in \mathbb{Z}^{m \times n}$ and a vector $\mathbf{d} \in (\mathbb{Z}^+)^m$, the generalized fundamental parallelepiped is defined as

$$\mathcal{P}^{\mathbf{d}}_{1/2}(\mathbf{B}) = \left\{ \mathbf{v} = \sum_{i=1}^{m} \alpha_i \mathbf{b}_i \mid \forall i \in [m] : -\frac{d_i}{2} \leq \alpha_i < \frac{d_i}{2} \right\} .$$

The output guarantee of nearest planes is as follows:

Theorem 1. *Let* $\mathbf{b} = \mathbf{As} + \mathbf{e} \mod q$ *be an LWE instance. Furthermore, let* $\mathbf{B} \in \mathbb{Z}^{m \times m}$ *be a basis of* $\Lambda_q(\mathbf{A})$ *with Gram-Schmidt basis* $\widetilde{\mathbf{B}}$ *and* $\mathbf{d} \in (\mathbb{Z}^+)^m$ *be a vector. On input* $\mathbf{B}, \widetilde{\mathbf{B}}, m, \mathbf{d}$ *and* $\mathbf{b}$*, nearest planes returns a vector* $\mathbf{v} \in \mathbb{Z}^m$ *such that* $\mathbf{v} = \mathbf{As} \mod q$ *if and only if* $\mathbf{e} \in \mathcal{P}^{\mathbf{d}}_{1/2}(\widetilde{\mathbf{B}})$.

The advanced decoding attack (basis reduction followed by nearest planes) allows two trade-offs: The parameter $\mathbf{d}$ can be used to tune the trade-off between the time spend on basis reduction and on nearest planes, while the hermite delta δ determines the trade-off between runtime and success probability.

Lindner and Peikert estimate the success probability of the overall attack via

$$p_{\text{succ}} = \prod_{i \in [m]} \text{erf}\left(\frac{\mathbf{d}_i \|\widetilde{\mathbf{b}}_i\| \sqrt{\pi}}{2\sigma}\right) \tag{2.9}$$

More details about the optimal choice of the attack parameters are given in Section 3.1.2.

2.3.4. Kannan's Reduction to unique SVP

An other purely lattice-based approach is to reduce the LWE instance to an instance of the unique shortest vector problem (uSVP). In a nutshell, the idea of the attack is to add the vector $\mathbf{b}$ to the lattice

$$\Lambda_q(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m \mid \exists \mathbf{x} \in \mathbb{Z}^n : \mathbf{A}\mathbf{x} = \mathbf{v} \mod q\}.$$

Since $\mathbf{A}\mathbf{s}$ and $\mathbf{b}$ are lattice vectors, $\mathbf{e}$ is a very short lattice vector, and one can apply a solver for unique-SVP to recover $\mathbf{e}$ (the typical solver is BKZ2.0).

2.3.5. Reduction to Inhomogeneous SIS

A different embedding approach is to see LWE as an instance of the Inhomogeneous Short Integer Solution (ISIS) problem. In order to do this, consider the equation

$$\mathbf{b} = \begin{pmatrix}\mathbf{A} & \mathbf{I}_m\end{pmatrix} (\mathbf{s}, \mathbf{e})^T \mod q.$$

Consequently, the vector $(\mathbf{s}, \mathbf{e})$ is a solution to ISIS with matrix $\begin{pmatrix}\mathbf{A} & \mathbf{I}_m\end{pmatrix}$ and target vector $\mathbf{b}$.

A established way to solve ISIS (and the reason for the title of this subsection) is to reduce it to an SVP problem by considering the equation

$$\begin{pmatrix}\mathbf{A} & \mathbf{I}_m & \mathbf{b}\end{pmatrix} (\mathbf{s}, \mathbf{e}, -1)^T = 0 \mod q.$$

Since $\mathbf{s}$ and $\mathbf{e}$ are small, $(\mathbf{s}, \mathbf{e}, -1)^T$ is a small vector in the lattice $\Lambda_q^{\perp}(\begin{pmatrix}\mathbf{A} & \mathbf{I}_m & \mathbf{b}\end{pmatrix})$ (see Equation (2.2) for the definition), so the remaining problem is to solve an uSVP instance in dimension $m + n + 1$ with determinant q^m (with high probability, see [BBD09]).

Part I.

Hardness of LWE

3 LWE Instances with Few Samples

In the original definition of LWE, a solver has access to arbitrary many samples. However, this is typically not true in real world applications. Concrete cryptographic schemes like the encryption scheme by Lindner and Peikert [LP11] or the signature scheme by Ducas et al. [DDLL13] only provide a bounded number of samples. A natural research question is therefore: how does a restricted number of samples influence the hardness of LWE?

In this chapter, we address this question by evaluating how a restricted number of samples influences the complexity of LWE solvers. Some attacks, like the Arora and Ge algorithm [AG11], are unsuitable in this setting. Other attacks can still be applied, but perform much worse. We show that some attacks depend more on the optimal attack dimension than others, and that this changes the picture of which attacks perform best on what instances completely.

Organisation Section 3.1 shows how to estimate the runtime of existing LWE solvers on LWE with few samples. Afterwards, Section 3.2 gives an overview of the estimated runtime of solvers for concrete instances with few samples.

3.1. Known Solvers

In this section, we focus on attacks than can be applied on LWE instances with few samples. Attacks requiring many samples like BKW or Arora-Ge are excluded.

3.1.1. Distinguishing Attack

As mentioned in the introduction, the distinguishing attack is one of the easiest attacks on LWE. Lindner and Peikert [LP11] claim that a short vector $\mathbf{v}$ in $\Lambda_q^\perp(\mathbf{A})$ is enough to distinguish an LWE instance from a uniformly random sample with advantage close to $\exp(-\pi(\|\mathbf{v}\|\sigma/q)^2)$ Obviously, this advantage grows with decreasing

size of $\mathbf{v}$. Consequently, the optimal dimension is the dimension that allows to find the shortest vector $\mathbf{v}$.

Arbitrary many samples Micciancio and Regev [MR09] show that a basis reduction algorithm that is capable to find a vector of hermite delta δ finds the smallest vector if run in dimension $\sqrt{n\log(q)/\log(\delta)}$. Using the equation $\det(\Lambda_q^\perp) = q^n$ (which is true withhigh probability [BBD09]) leads to

$$\|\mathbf{v}\| \approx \delta^m \cdot q^{n/m} = \delta^{\sqrt{n\log(q)/\log(\delta)}} q^{\frac{n\sqrt{\log(q)}}{\sqrt{n\log(n)}}} = 2^{2\sqrt{n\log_2(q)\log_2(\delta)}} \tag{3.1}$$

for the best achievable vector.

It is often better to amplify a small advantage by repeating the attack than running an attack with high advantage in the first place. Following the proposal by Albrecht et al. [ACF+15], we assume that it takes ε^2 repetitions of an attack with advantage ε to solve the problem with sufficiently high probability. Following Equation 3.1, the advantage for a given hermite delta δ is given by

$$\varepsilon_\delta = \exp(-\pi(2^{2\sqrt{n\log_2(q)\log_2(\delta)}}\sigma/q)^2).$$

The only thing remaining is to (numerically) find the hermite delta δ that minimizes $\frac{\mathrm{ops}_{\mathrm{BKZ}}(\delta)}{\varepsilon_\delta}$, where $\mathrm{ops}_{\mathrm{BKZ}}(\delta)$ is the runtime estimation of basis reduction (see Equation 2.5).

Bounded number of samples If there are not enough samples available to run the attack in the best dimension, it is commonly known [MR09] that the best strategy is to use all available samples. In this case, the norm of the best achievable vector is estimated by

$$\|\mathbf{v}\| \approx q^{n/m}\delta^m. \tag{3.2}$$

As in the "arbitrary many samples"-case, we repeat an attack with advantage ε ε^2 many times. In our case, the optimal hermite delta δ is the one that minimizes $\frac{\mathrm{ops}_{\mathrm{BKZ}}(\delta)}{\varepsilon_\delta}$, with

$$\varepsilon_\delta = \exp(-\pi(q^{n/m}\delta^m\sigma/q)^2).$$

3.1.2. Decoding Attack

As mentioned in the previous chapter, the decoding attack [LP11] is one of the most important attacks in practice. One of the problems when running it in practice is that selecting good attack parameters is non-trivial. Recall that the attacker can control the following parameters:

- $\mathbf{d}$, the number of recursive nearest-planes calls per level

- δ, the quality of the basis after basis reduction
- m, the number of samples

We discuss the optimal choice of those parameters in the following.

Since the runtime of nearest planes is nearly linear in $D = \prod_i d_i$, the value D is a natural attack parameter. Lindner and Peikert claim that it is optimal to choose $\mathbf{d}$ such that $\min_{i\in[m]}\{d_i \cdot \widetilde{\mathbf{b}}_i\}$ is maximized, so selecting D is in fact equivalent to selecting $\mathbf{d}$.

Finding the optimal value for δ is also not easy. However, an easy calculation shows that selecting δ such that the runtimes of basis reduction and nearest planes is about equal is at least close to optimal (up to a factor of two). Finding such a value for δ can be done easily using binary search.

Arbitrary many samples The last choice concerns m. Using arguments similar to those from the distinguishing-attack (see Section 3.1.1), Lindner and Peikert claim that it is at least close to optimal to use

$$m = \sqrt{n \log(q)/\log(\delta)}.$$

The expected runtime is then given by

$$\frac{\mathrm{ops}_{\mathrm{BKZ}}(\delta) + \mathrm{ops}_{\text{nearest planes}}(\delta, \mathbf{d})}{p_{\mathrm{succ}}(\delta, \mathbf{d}, m)} \tag{3.3}$$

for $m = \sqrt{n \log(q)/\log(\delta)}$ with $\mathrm{ops}_{\text{nearest planes}}(\delta, \mathbf{d}) = 2^{15} \prod_i \mathbf{d}_i$ (see [LP11]) and p_{succ} as in Equation 2.9.

Bounded number of samples When there are not enough samples to use the optimal attack dimension, Lindner and Peikert claim that one should always use all samples available. They do not give a formal proof of this statement, but it is intuitive and confirmed by all experiments so far. With this assumption, the expected runtime of the attack is given by Equation 3.3, but with m as the given number of samples. We want to point out at this point that the minor-looking difference has a huge impact, since the choice of m influences the choice of $\mathbf{d}$ as well as the values of $\widetilde{\mathbf{b}}_i$.

3.1.3. Kannan's Embedding Attack

Following Equation (2.6), we assume that the attack succeeds of (and only if)

$$\tau \delta^m \leq \frac{\sqrt{m}\det(\Lambda_q(\mathbf{A}))^{1/m}}{\sqrt{2\pi e}\|\mathbf{e}\|}.$$

With high probability (see [MR09]), the determinant of the q-ary lattice is given by q^{m-n}. Additionally, we assume that the error norm is given by $\|\mathbf{e}\| \approx \sqrt{m} \cdot c$ for a constant c that is independent of n, m, and q. This is true for all uniform errors, but also for Gaussian errors. With this assumptions, the attack is therefore successful if

$$\frac{q^{1-n/m}}{\sqrt{2e\pi}c\tau\delta^m} \geq 1. \tag{3.4}$$

Arbitrary many samples Standard arguments show that the left-hand side of Inequality (3.4) is maximized for $m = \sqrt{n\log(q)/\log(\delta)}$ [MR09]. In that case, $\delta = \exp(n\log(q)/m^2) = q^{n/m^2}$, which leads to

$$\frac{q^{1-n/m}}{\sqrt{2e\pi}c\tau\delta^m} = \frac{q^{1-n/m}}{\sqrt{2e\pi}c\tau}q^{-n/m} = \frac{q^{1-2n/m}}{\sqrt{2e\pi}c\tau}.$$

The next step is to simplify Inequality (3.4) for the optimal value of m:

$$\begin{aligned}\frac{q^{1-2n/m}}{\sqrt{2e\pi}c\tau} \geq 1 &\Leftrightarrow q^{1-2n/m} \geq \sqrt{2e\pi}c\tau \\ &\Leftrightarrow (1-2n/m)\log(q) \geq \log(\sqrt{2e\pi}c\tau) \\ &\Leftrightarrow 2n\log(q) \leq m\left(\log(q) - \log(\sqrt{2e\pi}c\tau)\right)\end{aligned}$$

After inserting $m = \sqrt{n\log(q)/\log(\delta)}$, we have:

$$\begin{aligned}\frac{q^{1-2n/m}}{\sqrt{2e\pi}c\tau} \geq 1 &\Leftrightarrow 2n\log(q) \leq \sqrt{\frac{n\log(q)}{\log(\delta)}}\left(\log(q) - \log(\sqrt{2e\pi}c\tau)\right) \\ &\Leftrightarrow 2\sqrt{n\log(q)}\sqrt{\log(\delta)} \leq \log(q) - \log(\sqrt{2e\pi}c\tau) \\ &\Leftrightarrow \sqrt{\log(\delta)} \leq \frac{\log(q) - \log(\sqrt{2e\pi}c\tau)}{2\sqrt{n\log(q)}} \\ &\Leftrightarrow \log(\delta) \leq \frac{\left(\log(q) - \log(\sqrt{2e\pi}c\tau)\right)^2}{4n\log(q)}\end{aligned}$$

Bounded number of samples If the LWE instance does not provide sufficiently many samples to run the attack in the optimal dimension, using all samples available is the best choice. In that case, Equation (3.4) is equivalent to

$$\delta \leq \sqrt[m]{\frac{q^{1-n/m}}{\sqrt{2\pi e}c\tau}}. \tag{3.5}$$

3.1.4. Reduction to uSVP via ISIS

As introduced in the background (see Section (2.3)), there is a second technique to reduce LWE to uSVP: via ISIS. It requires to solve a uSVP problem in dimension $m+n+1$ with determinant q^m. Following Equation (2.6) again, we assume that the attack is successful if

$$\sqrt{2\pi e}\lambda_1(\Lambda)\tau\delta^{m+n+1} \leq \sqrt{m+n+1}\det(\Lambda)^{1/(m+n+1)}.$$

Inserting the determinant and the norm $\|(\mathbf{s},\mathbf{e},-1)^T\| \approx c\sqrt{m+n+1}$ leads to

$$\sqrt{2\pi e}c\sqrt{m+n+1}\tau\delta^{m+n+1} \leq \sqrt{m+n+1}q^{\frac{m}{m+n+1}},$$

which is equivalent to

$$\frac{q^{\frac{m}{m+n+1}}}{\sqrt{2\pi e}c\tau\delta^{m+n+1}} \geq 1. \tag{3.6}$$

Arbitrary many samples The remaining task is to identify the smallest δ, such that there is an $m > 0$ that fulfills this inequality. In order to do this, we first identify m such that the function

$$f(m) = \frac{q^{\frac{m}{m+n+1}}}{\delta^{m+n+1}}$$

is maximized. The first derivative is given by

$$f'(m) = \delta^{-n-m-1}q^{m/(n+m+1)}\left(\frac{(n+1)\log(q)}{(n+m+1)^2} - \log(\delta)\right).$$

An easy calculation shows that this is only zero for $m = \pm\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}} - n - 1$. Since we are only interested in positive values for m, the only candidate is

$$m = \sqrt{\frac{(n+1)\log(q)}{\log(\delta)}} - n - 1. \tag{3.7}$$

Standard analytic techniques show that this is indeed the maximum for positive values of m (see Figure 3.1).

Figure 3.1 plots the left hand side of Inequality (3.6) for $n = 256, q = 256, c = \frac{1}{\sqrt{2}}$. Obviously, $\delta = 1.005$ should be enough to break the instance, while $\delta = 1.006$ would not suffice. In order to find the minimal δ in a more systematic way, we insert the

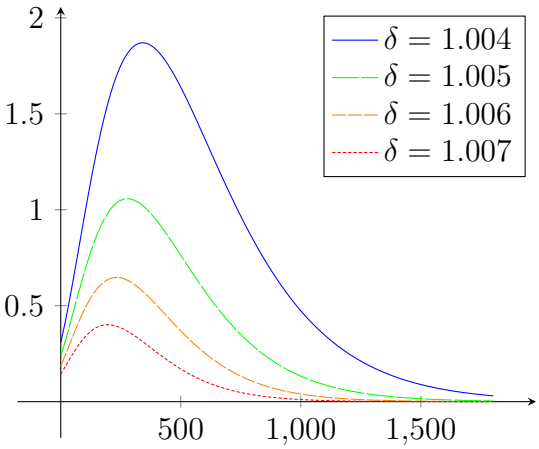


Figure 3.1.: Left hand side of Inequality (3.6) for $n = 256, q = 256, c = \frac{1}{\sqrt{2}}$

optimal m from Equation (3.7) into Inequality (3.6) and have:

$$\begin{aligned}
&\frac{q^{\frac{m}{m+n+1}}}{\sqrt{2\pi e}c\tau\delta^{m+n+1}} \geq 1 \Leftrightarrow q^{\frac{m}{m+n+1}} \geq \sqrt{2\pi e}c\tau\delta^{m+n+1} \\
&\Leftrightarrow q^{\frac{\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}}-n-1}{\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}}}} \geq \sqrt{2\pi e}c\tau\delta^{\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}}} \\
&\Leftrightarrow \log(q)\frac{\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}}-n-1}{\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}}} \geq \log(\sqrt{2\pi e}c\tau) + \log(\delta)\sqrt{\frac{(n+1)\log(q)}{\log(\delta)}} \\
&\Leftrightarrow \log(q)\left(1 - \frac{(n+1)\sqrt{\log(\delta)}}{\sqrt{(n+1)\log(q)}}\right) \geq \log(\sqrt{2\pi e}c\tau) + \sqrt{(n+1)\log(q)\log(\delta)} \\
&\Leftrightarrow \log(q) - \sqrt{(n+1)\log(q)\log(\delta)} \geq \log(\sqrt{2\pi e}c\tau) + \sqrt{(n+1)\log(q)\log(\delta)} \\
&\Leftrightarrow 2\sqrt{(n+1)\log(q)\log(\delta)} \leq \log(\sqrt{2\pi e}c\tau) - \log(q) \\
&\Leftrightarrow \log(\delta) \leq \frac{\left(\log(\sqrt{2\pi e}c\tau) - \log(q)\right)^2}{4(n+1)\log(q)}
\end{aligned}$$

It is interesting to note that the embedding attack via ISIS requires nearly, but not exactly the same hermite delta as the direct embedding approach (it is slightly bigger). On the other hand, it solves uSVP in a lattice of a smaller dimension, and therefore my be still favorable in practice.

Bounded number of samples If the attacker does not have access to enough samples, Figure 3.1 indicates that it is optimal to use all samples. In that case,

Inequality (3.6) is equivalent to

$$\delta \leq \sqrt[m+n+1]{\frac{q^{\frac{m}{m+n+1}}}{\sqrt{2\pi e}c\tau}}.$$

3.2. Concrete Complexity of LWE with Few Samples

We modified the BKZ simulator by Albrecht et al. [APS15] to predict the runtime of LWE with a bounded number of samples. In order to compare the attack, it was also necessary to add the reduction via ISIS to the simulator[1].

Figure 3.2 shows the runtime of the different attacks for different number of samples. We want to emphasize that in order to illustrate the influence of the number of samples, m is the number of *used*, and not the number of *available* samples. A real attacker could of course ignore additional samples and attack the problem in the optimal dimension, if he has enough samples available. The graph a follows a similar pattern for all attacks. However, the optimal number of samples is different for every attack.

A particularly interesting attack is the reduction to uSVP via ISIS. For the instance considered, it is not the best attack if enough samples are available. However, for a limited number of samples, the picture changes: If less than 650 samples (which is about 2.5 times n) samples are available, the attack via ISIS has the best performance. This is much more than a theoretical result, since several cryptographic primitives (like the encryption scheme by Lindner and Peikert [LP11]) only provide $2n$ samples.

Of course, Figure 3.2 is only valid for a specific LWE instance. However, for nearly all instances observed, the best attack changes depending on how many samples are available (additional examples are given in Section 7.2). Considering this when selecting parameters for a cryptographic application can lead to an improved performance (see Chapter 7).

[1]The changes will be included to the simulator soon

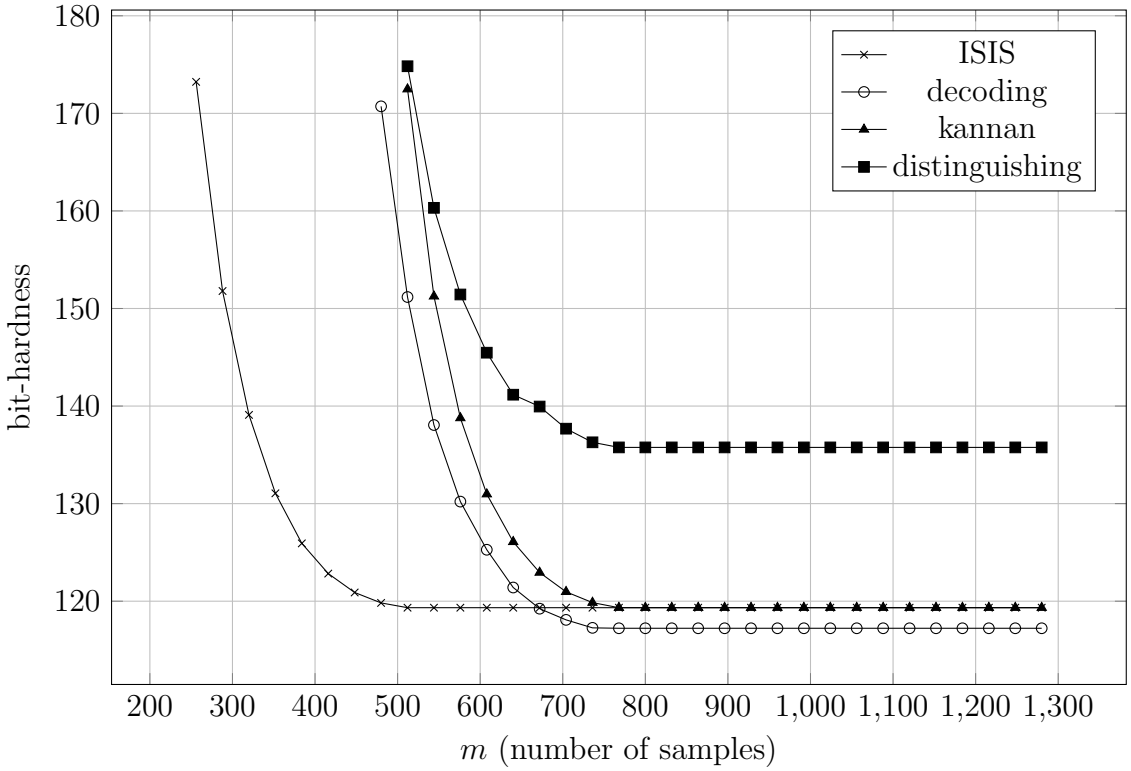


Figure 3.2.: Hardness of LWE instances with dimension of secret $n = 256$, modulus $q = 65537$, and error parameter $\sigma = 0.0009765625 \cdot \sqrt{2\pi}q$, depending on the number of available samples m

4 Attacking LWE with Parallel Enumeration

In order to be able to select optimal and secure parameters for LWE-based schemes, it is crucial to understand the complexity of the existing attack algorithms on real-world environments. In particular, it is important to investigate how they perform on modern, highly parallel computing architectures. This leaves us with two important questions: can LWE solvers benefit from parallelization? And (if yes), how does this influence the parameter selection for cryptographic schemes?

This section addresses those questions with respect to the nearest planes algorithm [LP11], which is part of the decoding attack (see Section 2.3.3). It shows that parallel computing architectures can be used to decrease the runtime significantly. We compare our results with existing estimated lower bounds for the achievable runtime and show that those bounds (despite being considered conservative) can actually be reached and even outperformed in practice. Furthermore, we show ways to further improve the parallel implementations. This shows that care must be taken not to underestimate the power of modern highly parallel computing devices.

To investigate the impact of a parallel nearest planes algorithm, we combine our results with existing methods that estimate the runtime of basis reduction (the second part of the decoding attack). We show that even a massive speed-up for the nearest planes algorithm does not lead to a significant improvement of the complete decoding attack. Consequently, further research should focus on developing parallel versions of basis reduction algorithms, rather than improving the nearest planes step.

Organisation Section 4.1 presents a parallel implementation of nearest planes on a shared-memory device that outperforms the efficiency bound proposed by Lindner and Peikert [LP11]. Section 4.2 gives hints how to further improve the efficiency of the implementation. To this end, we discuss alternative ways to traverse the search tree build in nearest planes. However, we do not give such an implementation, for reasons given in Section 4.3. In this section, we simulate the impact an additional speed-up for nearest planes or basis reduction would have on the overall running time of the decoding attack. We show that even a tremendous improvement of the

former only leads to a modest speedup for the overall attack, while speed ups of the latter lead to improvements nearly linearly.

The parallel implementation of nearest planes and the discussions of further improvements was published at BalkanCryptSec 2014 [BBD+14], the simulation of different speed-ups is original in this thesis.

4.1. Parallel Version of Enumeration

4.1.1. On the Necessity of the Projection

Two versions of nearest planes appeared in the wild. Some papers have line 6 of Algorithm 2, some skip this step ([LP11]). Interestingly, both variants lead to the same result. We give a prove for nearest plane here, the result for nearest planes follows similarly. Let $\mathbf{t}_i$ be the target vector in the ith recursive call of nearest plane (i.e. the call for $k = m - i$) as given in Algorithm 1, and $\tilde{\mathbf{t}}_i$ be the corresponding target vector in a nearest plane version without line 6.

Theorem 2. *Let $\mathbf{B}, \widetilde{\mathbf{B}}, m, \mathbf{d}$, and $\mathbf{t}$ be a correct input for Algorithm 1. Let $\mathbf{t}_k$ be the target vector in the recursive call of nearest plane with input k as given in Algorithm 1, and $\mathbf{t}'_k$ be the corresponding target vector in a nearest plane version without line 6. For all $k \in [m]$,*

$$\pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}_k) = \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}'_k).$$

Proof. We proof the statement by recursion over k. It is obviously true for $k = m$ and $k = m - 1$. Assume

$$\pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}_k) = \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}'_k)$$

for a fixed k. Let $\mathbf{t}^*_k$ be the value of $\mathbf{t}$ after line 6 in the recursive call with input k, i.e., $\mathbf{t}^*_k = \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}_k)$. The next step is to investigate the inner product calculated in line 7. Note that

$$\langle \tilde{\mathbf{b}}_k, \mathbf{t}^*_k \rangle = \langle \tilde{\mathbf{b}}_k, \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}_k) \rangle = \langle \tilde{\mathbf{b}}_k, \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_k\})}(\mathbf{t}'_k) \rangle = \langle \tilde{\mathbf{b}}_k, \mathbf{t}'_k \rangle,$$

and consequently the value of the coefficient c is the same in both nearest plane versions. The next target values are given by $\mathbf{t}_{k-1} = \mathbf{t}^*_k - c\mathbf{b}_k$ and $\mathbf{t}'_{k-1} = \mathbf{t}'_k - c\mathbf{b}_k$. The rest follows, since

$$\begin{aligned}
\pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}_{k-1}) &= \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}^*_k - c\mathbf{b}_k) \\
&= \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}_k - c\mathbf{b}_k) \\
&= \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}_k) - \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(-c\mathbf{b}_k) \\
&= \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}'_k) - \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(-c\mathbf{b}_k) \\
&= \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}'_k - c\mathbf{b}_k) \\
&= \pi_{\mathrm{span}(\{\mathbf{b}_1,\ldots,\mathbf{b}_{k-1}\})}(\mathbf{t}'_{k-1}).
\end{aligned}$$

□

Theorem 2 shows that both variants of nearest plane lead to the same result. Consequently, we ignore the projection-step in the following.

4.1.2. Returning the Error Vector

Despite the fact that knowing the secret $\mathbf{s}$ is equivalent to knowing the error $\mathbf{e}$, LWE typically asks for $\mathbf{s}$. Likewise, CVP typically asks to recover the close lattice vector, and not the error vector $\mathbf{e}$. In the original definition, nearest plane was formulated to satisfy this requirement. So far, we followed this tradition (see Algorithms 1 and 2 in order to make this work more accessable to readers familiar with existing literature. Asking for the error comes with several advantages. For one thing, it is more natural for some attacks to recover $\mathbf{e}$ (see also Chapter 5). For another thing, recovering $\mathbf{e}$ leads to some advantages when implementing some of the attacks, including nearest planes.

To see why, recall that Algorithm 2 recovers a close lattice vector. This is done by calling nearest planes recursively many times, which can be seen traversing a tree in a depth-first manner, as depicted in Figure 4.1. Every node stands for one nearest plane call, the level in the tree corresponds to the parameter k of the call. While going down the tree, every node stands for calculating the values of c_i and the new target vectors for the next recursive call. Assume a leaf is reached. In theory, every leaf stands for one lattice vector that is somewhat close to the original target vector. Unfortunately, calculating this candidate solutions requires information from all nodes between the leaf and the root of the tree. Therefore, it is necessary to climb back up the tree and collect this information, which is an unpleasant overhead.

In our implementation, we followed a different approach. In order to explain this, we investigate what happens to the target vector $\mathbf{t}$ in nearest plane without the optional projection step. In every recursive call, nearest planes subtracts a basis vector of $\mathbf{t}$. Consequently, the target vector for the last call (where $k = 0$) is given by $t - \sum_{k=1}^{m} c_i^k b_k$, where $c_i^k b_k$ is the lattice vector that gets calculated by Algorithm 2. Of course, this means that $\mathbf{e} = t - \sum_{k=1}^{m} c_i^k b_k$ is the error vector corresponding to this target vector. Therefore, the last call of nearest plane may be lacking the information to calculate the lattice vector, but it has immediate access to the error vector.

So far, we saw that every leaf node has the information necessary to calculate one error vector. The next step is to make this local information globally accessible. Unfortunately, this can not be done easily in the recursive version of nearest planes. The problem is that the nodes do not know about their position in the tree, so they do not know the correct place to store their result. This could be solved with a global counter, but this solution could lead to additional problems in the parallel version, since several threads would read and write this counter. Our solution is to abandon

the recursive structure and create an iterative version of nearest plane instead. The result is given in Algorithm 3. Besides the recursive structure, there is an additional difference: instead traversing the tree in a depth-first fashion, Algorithm 3 follows a breadth-first approach. The difference is depicted in Figure 4.1. This leads to a simpler program, but comes with some problems that are discussed in Section 4.2.

4.1.3. Parallelization

The search tree created by the nearest plane algorithm comes with two properties that lead to an efficient parallelization: the tree is entirely symmetric, and the size of the tree and every subtree is known in advance. A natural approach is to traverse the tree sequentially, until we reached a level that contains enough subtrees to keep the desired number of sub processes busy (called the adequate level). Afterwards, every process processes one of the subtrees, and writes all collected error vectors in a global array. Every process gets an id, which allows him to write the collected information to the right position in the array. This approach ensures that we do not create any data races that could potentially break the workflow of the algorithm. The whole algorithm is given in Algorithm 4.

Algorithm 3: Sequential, Recursive Nearest Planes

Input : $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\} \subset \mathbb{R}^m$, `// the lattice basis`
$\mathbf{d} \in (\mathbb{Z}^+)^m$, `// recursive branching vector`
$\mathbf{t} \in \mathbb{R}^m$ `// the target vector`
Output: All error vectors $\mathbf{e} \in \mathcal{P}^{\mathbf{d}}_{1/2}(\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_m\})$ such that $\mathbf{t} - \mathbf{e} \in \Lambda(\{\mathbf{b}_1, \ldots, \mathbf{b}_m\})$

begin
  calculate Gram-Schmidt basis $\widetilde{\mathbf{B}} = \left\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_m\right\}$;
  $len = 1$;
  for $k = n;\ k \geq 1;\ k = k - 1$ **do**
    for $i = 0;\ i < len;\ i = i + 1$ **do**
      Let $\{c_1, \ldots, c_{d_k}\} \in \mathbb{Z}^{d_k}$ be the distinct integers closest to $\frac{\langle \tilde{\mathbf{b}}_k, \mathbf{t} \rangle}{\langle \tilde{\mathbf{b}}_k, \tilde{\mathbf{b}}_k \rangle}$;
      for $j = 1;\ j \leq \mathbf{d}_k;\ j = j + 1$ **do**
        $\mathbf{t}^*_{i \cdot \mathbf{d}_k + j} = \mathbf{t}_i - c_j \cdot \mathbf{b}_k$;
      end
    end
    $\mathbf{t} = \mathbf{t}^*$;
    $len = len \cdot \mathbf{d}_k$;
  end
  return $\mathbf{t}$;
end

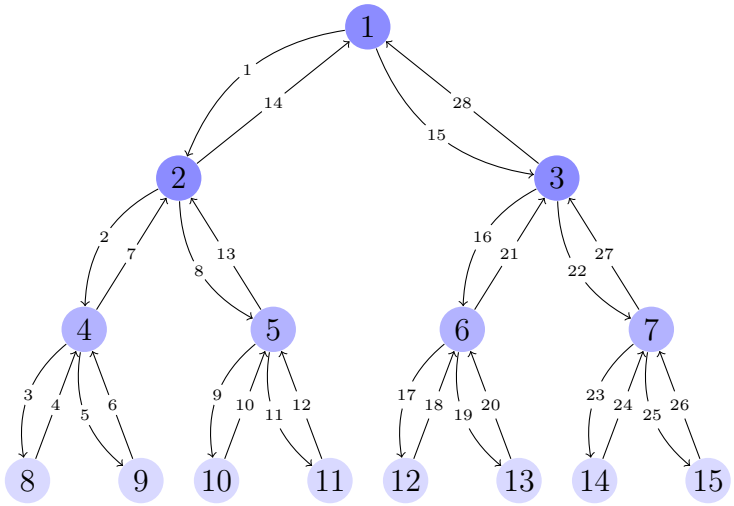


Figure 4.1.: Workflow of Algorithm 2 (arrows) and Algorithm 3 (node numbers)

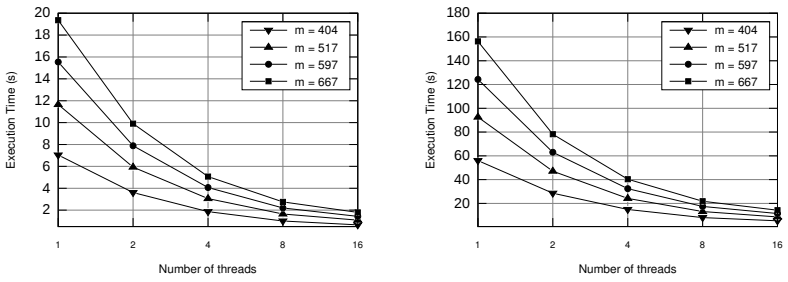
In fact, the implementation of Algorithm 4 presented by Bischof et al. [BBD+14] scales very well on a multi-core architecture. For completeness, the results are given in Figure 4.2 and Table 4.1.

4.2. Further Possible Improvements

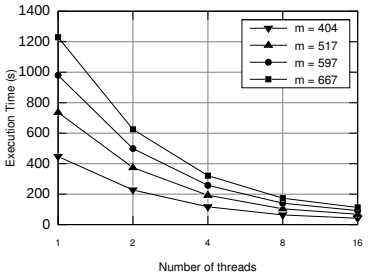
Despite its good scalability, our implementation is not optimal yet. From an implementation point of view, the main bottleneck seems to be the memory consumption. The main reason for this is that the breadth-first approach requires to store all nodes, while a depth-first algorithm can be implemented with a significantly smaller memory footprint.

A second improvement could be achieved by applying recent theoretic improvements presented by Liu and Nguyen [LN13].

They show that nearest planes can be viewed as an instance of enumeration (more commonly studied with regard to solving the exact shortest vector problem) and apply known improved variants of enumeration to nearest planes to obtain theoretical and practical improvements over [LP11]. In particular, those improvements are randomization and pruning. The idea of randomization is to apply the attack many times with parameters that provide only a small success probability with random bases. Applying this approach with a parallel implementation of nearest plane is easily possible. The idea of pruning is to cut off parts of the search trees that contribute significantly to the running time but only slightly to the success probability. This leads to unbalanced search trees and makes parallelization more difficult, but not impossible as the parallel implementation of the pruned enumeration by Kuo et



(a) $\#enum = 2^{12}$ (b) $\#enum = 2^{15}$



(c) $\#enum = 2^{18}$

Figure 4.2.: Performance of our implementation executing the nearest planes algorithm on random lattices, for dimensions 404, 517, 597 and 667, with $\#enum = 2^{12}$ in (a), $\#enum = 2^{15}$ in (b) and $\#enum = 2^{15}$ in (c). Runtime in seconds (less is better).

al. [KSD+11] shows (see also [DS10, HSB+10]).

4.3. Bounds on the Speedup

Before working on a parallel implementation of a lattice attack, it is important to answer two questions:

1. Is the algorithm suitable for parallelization?

2. What impact will a parallel implementation have?

Concerning enumeration as subroutine of the decoding attack, the first question was answered in the last section. The second question is important for very pragmatic reasons. Consider a computer scientist with a limited amount of time that wants to improve the decoding attack. The decoding attack consists of two subroutines (basis reduction and enumeration), but he can (due to time restrictions) only improve one of them. Which one should he choose?

A different way of looking at the problem is from the perspective of a researcher that wants to select parameter for an LWE-based scheme. Since the parameters have to withstand attacks in a foreseeable future, he wants to take possible improvements into considerations. How would an improvement by a factor of 2^x for the enumeration attack influence the hardness of his LWE instance? And how about an improvement for basis reduction?

In order to answer the second question, we perform the following thought experiment: Assume there is a parallel version of enumeration that scales perfectly on arbitrary many cores, and fix an LWE instance. How does the runtime of the decoding attack on 2^x cores depend on x?

The reason that answering this question is not trivial is that the decoding attack consists of two steps: Basis reduction followed by enumeration. Consequently, it is unclear how a speedup of a part of the attack (in this case: enumeration) carries over to a speedup to the complete attack.

The answer for this question is given in Figure 4.3. It gives hardness estimations for an LWE instance provided by Linder and Peikert [LP11], given different speedups for basis reduction and enumerations. The hardness estimations have been created using a modified version of the LWE hardness estimator by Albrecht et al. [APS15]. As the figure shows, the hardness of the instance is predicted to be 145 bits. Of course, speeding up both parts on the decoding attack by ten bits would drop the hardness to 135 bits, so the values on the diagonal are not surprising. The more interesting conclusion can be taken from the values on the lowest row and the first column. They show for example that speeding up enumeration by a factor of 2^{10} would drop the hardness by only two bits, while the same improvement of basis reduction leads to a significantly bigger speedup of 2^7. In other terms: to achieve the same hardness drop one can get from speeding up basis reduction by a factor of 2^{10}, one would have to improve enumeration by more than 2^{30}.

Following Figure 4.3, we conclude that further improvements of the enumeration step is not of high priority. Consequently, we did not spend time to develop a more optimized parallel implementation of enumeration.

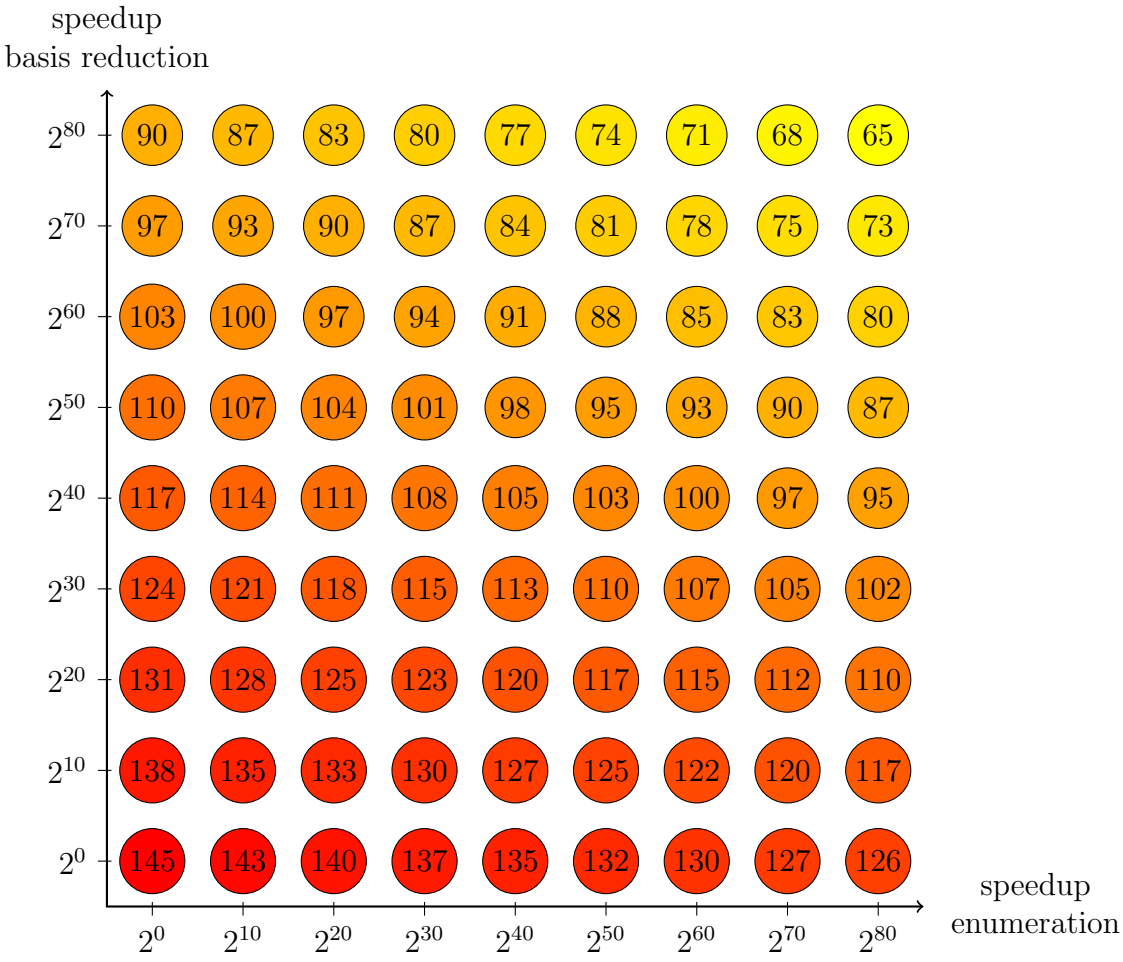


Figure 4.3.: Runtime of the decoding attack depending on speedups for enumeration and basis reduction for LWE with $n = 256, q = 4093, \sigma = 3.33$

Algorithm 4: Nearest Planes

Input : $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\} \subset \mathbb{R}^m$, // the lattice basis
$\mathbf{d} \in (\mathbb{Z}^+)^m$, // recursive branching vector
$\mathbf{t} \in \mathbb{R}^m$ // the target vector
$al \in \mathbb{N}$ // the adequate level

Output: All error vectors $\mathbf{e} \in \mathcal{P}^{\mathbf{d}}_{1/2}(\{\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_m\})$ such that $\mathbf{t} - \mathbf{e} \in \Lambda(\{\mathbf{b}_1, \ldots, \mathbf{b}_m\})$

```
1  begin
2      calculate Gram-Schmidt basis B̃ = {b̃_1, ..., b̃_m};
3      OpenMP_parallel_region
4          OpenMP_single_region
5              len = 1;
6              for k = n; k ≥ n − al; k = k − 1 do
7                  for i = 0; i < len; i = i + 1 do
8                      Let {c_1, ..., c_{d_k}} ∈ Z^{d_k} be the distinct integers closest to
                       ⟨b̃_k, t⟩ / ⟨b̃_k, b̃_k⟩;
9                      for j = 1; j ≤ d_k; j = j + 1 do
10                         t*_{i·d_k+j} = t_i − c_j · b_k;
11                     end
12                 end
13                 t = t*;
14                 len = len · d_k;
15             end
16             for node = 0; node ≤ ∏_{i=n−al}^{n} i; node = node + 1 do
17                 create_task
18                     len = 1;
19                     for k = n − al − 1; k ≥ 1; k = k − 1 do
20                         for i = 0; i < len; i = i + 1 do
21                             Let {c_1, ..., c_{d_k}} ∈ Z^{d_k} be the distinct integers closest
                               to ⟨b̃_k, t⟩ / ⟨b̃_k, b̃_k⟩;
22                             for j = 1; j ≤ d_k; j = j + 1 do
23                                 t*^{node}_{i·d_k+j} = t_i − c_j · b_k;
24                             end
25                         end
26                         t = t*;
27                         len = len · d_k;
28                     end
29                 end
30             end
31         end
32     end
33     return t;
34 end
```

	$m = 404$						$m = 517$					
	number of enumerations											
	2^{12}		2^{15}		2^{18}		2^{12}		2^{15}		2^{18}	
Threads	R	S	R	S	R	S	R	S	R	S	R	S
1	7.04	1.00	56.03	1.00	446.93	1.00	11.65	1.00	92.63	1.00	736.19	1.00
2	3.61	1.95	28.54	1.96	227.43	1.97	5.93	1.96	47.14	1.96	373.78	1.97
4	1.87	3.77	14.88	3.77	117.18	3.81	3.06	3.81	24.19	3.83	192.71	3.82
8	1.01	6.99	8.04	6.97	63.81	7.00	1.65	7.07	13.16	7.04	104.20	7.06
16	0.66	10.71	5.36	10.45	42.01	10.64	1.08	10.75	8.64	10.72	67.93	10.84

	$m = 597$						$m = 667$					
	number of enumerations											
	2^{12}		2^{15}		2^{18}		2^{12}		2^{15}		2^{18}	
Threads	R	S	R	S	R	S	R	S	R	S	R	S
1	15.54	1.00	124.31	1.00	979.78	1.00	19.36	1.00	156.34	1.00	1229.38	1.00
2	7.88	1.97	63.03	1.97	499.26	1.96	9.91	1.95	78.20	2.00	624.76	1.97
4	4.07	3.82	32.37	3.84	257.64	3.80	5.07	3.82	40.36	3.87	321.76	3.82
8	2.21	7.05	17.45	7.12	140.85	6.96	2.75	7.04	21.83	7.16	173.44	7.09
16	1.43	10.86	11.46	10.85	92.28	10.62	1.79	10.80	14.22	10.99	112.54	10.92

Table 4.1.: Runtime in seconds (R) and speed-up (S) for our implementation for LWE instances proposed in [LP11]

5 LWE with Binary Error

In his original work [Reg05], Regev defined LWE with an arbitrary error distribution. However, he gave a worst-case reduction for Gaussian errors only, and most subsequently proposed schemes also use Gaussian error. In this chapter, we discuss the hardness of LWE with binary error by presenting a new attack and comparing its efficiency with existing approaches.

In 2013, Micciancio and Peikert [MP13] showed the worst-case hardness of LWE with different distributions including binary error. In a related line of work, several proposals replaced the Gaussian error with a distribution that can be sampled easier [CGW14, GLP12]. This typically leads to more efficient schemes, since sampling from a Gaussian distribution is rather complicated, time-consuming, often requires big lookup tables, and may even insert side-channel vulnerabilities [BCG$^+$13, BHLY16].

On the other hand, history shows that using special instances of a problem to boost performance often leads to weaker schemes. This is particularly true for LWE, where a growing line of work identified easy or at least easier-than-expected instances [EHL14, ELOS15, CLS15, LL15]. It is therefore crucial for the trustworthyness of LWE-based cryptography to carefully analyze special instances that can be used for schemes.

This chapter investigates the hardness of LWE with binary error (binLWE[1]). Binary errors can be sampled extremely easily and efficiently, and in addition often allow to use a smaller modulus, which typically leads to smaller key sizes [BGG$^+$16]. However, we show that binLWE is easier than expected. To this end, we propose a new attack that can outperform all existing methods for typical parameter choices. Since the performance of most attacks on binLWE was never investigated before, we give a careful analysis for this schemes. Finally, we explain how to estimate the hardness of binLWE, which allows to select instances that withstand all nowaday known attacks.

Organisation Section 5.1 presents the new attack and analysis its performance. Section 5.2 compares the new attack with existing approaches. To this end, we

[1]Note that other definitions of binLWE appeared before in the literature (e.g., [BG14c])

give the first comprehensive study on the performance of existing LWE solvers on binLWE. We show that the new attack outperforms all existing approaches for most realistic parameter sets. Consequently, the techniques presented before can be used to estimate the hardness of binLWE.

This chapter is based on a publication that appeared in Africacrypt 2016 [BGPW16].

5.1. The Hybrid Lattice-Reduction and Meet-in-the-Middle Attack

In this section we present and analyze the hybrid attack on LWE with binary error. The attack is described in Algorithm 5 of Section 5.1.1, where $\mathrm{NP}_{\mathbf{B}}(\mathbf{t})$ stands for the nearest plane, called on input basis $\mathbf{B}$ and target vector $\mathbf{t}$. In Theorem 3 of Section 5.1.2 we analyze the expected runtime of the hybrid attack. Section 5.1.3 shows how to optimize the attack parameters and perform a trade-off between precomputation and the actual attack in order to minimize the runtime of the attack.

5.1.1. The Hybrid Attack

In the following we describe the hybrid attack on LWE with binary error. The attack is presented in Algorithm 5.

Let $m, n, q \in \mathbb{N}$ and let

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}\tilde{\mathbf{s}} + \mathbf{e} \mod q) \tag{5.1}$$

with $\mathbf{A} \in \mathbb{Z}_q^{m \times n}, \mathbf{b} \in \mathbb{Z}_q^m, \tilde{\mathbf{s}} \in \{0,1\}^n$ and $\mathbf{e} \in \{0,1\}^m$ be an LWE instance with binary error $\mathbf{e}$ and binary secret $\tilde{\mathbf{s}}$. In order to obtain a smaller error vector we can subtract the vector $(1/2) \cdot \mathbf{1}$ consisting of all $1/2$ entries from Equation (5.1). This yields a new LWE instance $(\mathbf{A}, \mathbf{b}' = \mathbf{A}\tilde{\mathbf{s}} + \mathbf{e}' \mod q)$, where $\mathbf{b}' = \mathbf{b} - (1/2) \cdot \mathbf{1}$ and $\mathbf{e}' = \mathbf{e} - (1/2) \cdot \mathbf{1}$. The new error vector $\mathbf{e}'$ now has norm $\sqrt{m/4}$ instead of the expected norm $\sqrt{m/2}$ of the original error vector $\mathbf{e}$. For $r \in \{1, \ldots, n-1\}$, we can split the secret $\tilde{\mathbf{s}} = \begin{pmatrix} \mathbf{v} \\ \mathbf{s} \end{pmatrix}$ and the matrix $\mathbf{A} = (\mathbf{A}_1 | \mathbf{A}_2)$ into two parts and rewrite this LWE instance as

$$\mathbf{b}' = (\mathbf{A}_1 | \mathbf{A}_2) \begin{pmatrix} \mathbf{v} \\ \mathbf{s} \end{pmatrix} + \mathbf{e}' = \mathbf{A}_1 \mathbf{v} + \mathbf{A}_2 \mathbf{s} + \mathbf{e}' \mod q, \tag{5.2}$$

where $\mathbf{v} \in \{0,1\}^r, \mathbf{s} \in \{0,1\}^{n-r}, \mathbf{A}_1 \in \mathbb{Z}_q^{m \times r}, \mathbf{A}_2 \in \mathbb{Z}_q^{m \times (n-r)}, \mathbf{b}' = \mathbf{b} - (1/2) \cdot \mathbf{1} \in \mathbb{Q}^m$, and $\mathbf{e}' = \mathbf{e} - (1/2) \cdot \mathbf{1} \in \{-1/2, 1/2\}^m$.

The main idea of the attack is to guess $\mathbf{v}$ and solve the remaining LWE instance $(\mathbf{A}_2, \tilde{\mathbf{b}} = \mathbf{b}' - \mathbf{A}_1 \mathbf{v} = \mathbf{A}_2 \mathbf{s} + \mathbf{e}' \mod q)$, which has binary secret $\mathbf{s}$ and error $\mathbf{e}' \in \{-1/2, 1/2\}^m$. The new LWE instance obtained in this way turns out to be

considerably easier to solve, since the determinant $\det(\Lambda_q(\mathbf{A}_2)) = q^{m-n+r}$ of the new lattice is significantly bigger than the determinant $\det(\Lambda_q(\mathbf{A})) = q^{m-n}$ of the original lattice (see Section 6.1 of [BG14c]). The newly obtained LWE instance is solved by solving a close vector problem in the lattice $\Lambda_q(\mathbf{A}_2)$. In more detail, $\tilde{\mathbf{b}} = \mathbf{A}_2\mathbf{s}+q\mathbf{w}+\mathbf{e}'$ for some vector $\mathbf{w} \in \mathbb{Z}^m$ is close to the lattice vector $\mathbf{A}_2\mathbf{s} + q\mathbf{w} \in \Lambda_q(\mathbf{A}_2)$ since $\mathbf{e}'$ is small. Hence one can hope to find $\mathbf{e}'$ by running the nearest plane algorithm in combination with a sufficient basis reduction as a precomputation (see [LP11]).

The guessing of $\mathbf{v}$ is sped up by a Meet-in-the-Middle approach, i.e., guessing binary vectors $\mathbf{v}_1 \in \{0,1\}^r$ and $\mathbf{v}_2 \in \{0,1\}^r$ such that $\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2$. In order to recognize matching guesses $\mathbf{v}_1$ and $\mathbf{v}_2$ that sum up to $\mathbf{v}$, one searches for collisions in (hash) boxes. The addresses of these boxes are determined in the following way.

Definition 3. *Let $m \in \mathbb{N}$. For a vector $\mathbf{x} \in \mathbb{R}^m$ the set $\mathcal{A}_{\mathbf{x}}^{(m)} \subset \{0,1\}^m$ is defined as*

$$\mathcal{A}_{\mathbf{x}}^{(m)} = \left\{ \mathbf{z} \in \{0,1\}^m \;\middle|\; \begin{array}{l} (\mathbf{z})_i = 1 \text{ for all } i \in \{1,\dots,m\} \text{ with } (\mathbf{x})_i > -1/2, \text{ and} \\ (\mathbf{z})_i = 0 \text{ for all } i \in \{1,\dots,m\} \text{ with } (\mathbf{x})_i < -1/2 \end{array} \right\}.$$

Intuitively, for $\mathbf{x}_2$ obtained during Algorithm 5, the set $\mathcal{A}_{\mathbf{x}_2}^{(m)}$ captures all the possible sign vectors of $\mathbf{x}_2$ added up with a vector in $\{-1/2, 1/2\}^m$ (where 1 represents a non-negative and 0 a negative sign). For $\mathbf{x}_1$ obtained during Algorithm 5, the set $\mathcal{A}_{\mathbf{x}_1}^{(m)}$ consists only of the sign vector of $\mathbf{x}_1$. This is due to the fact that $\mathbf{x}_2 \in \mathbb{Z}^m + \{1/2\}^m$, whereas $\mathbf{x}_1 \in \mathbb{Z}^m$. This leads to the desired collisions, as can be seen in the upcoming Lemma 2.

5.1.2. Runtime Analysis

In this section we analyze the runtime and success probability of the attack presented in Algorithm 5. We start by presenting our main result.

Theorem 3. *Let $n, m, q, c \in \mathbb{N}$, and $1 \leq \delta \in \mathbb{R}$ be fixed. Consider the following input distribution of $(q, r, \mathbf{A}, \mathbf{b}, \mathbf{B})$ for Algorithm 5. The modulus q and the attack parameter $r = 4c$ are fixed, $\mathbf{A} = (\mathbf{A}_1|\mathbf{A}_2)$, where $\mathbf{A}_1 \xleftarrow{\$} \mathbb{Z}_q^{m\times r}$, $\mathbf{A}_2 \xleftarrow{\$} \mathbb{Z}_q^{m\times(n-r)}$, $\mathbf{b} = \mathbf{A}\begin{pmatrix}\mathbf{v}\\ \mathbf{s}\end{pmatrix} + \mathbf{e} \mod q$, where $\mathbf{v} \xleftarrow{\$} \{0,1\}^r$, $\mathbf{s} \xleftarrow{\$} \{0,1\}^{n-r}$, $\mathbf{e} \xleftarrow{\$} \{0,1\}^m$, and $\mathbf{B}$ is some lattice basis of $\Lambda_q(\mathbf{A}_2)$ with Hermite delta δ. Let all notations be as in the above description of the input distribution. Assume that the approximations given in Heuristic 1, Heuristic 2 and Heuristic 4 are in fact equations.Then, if Algorithm 5 terminates, it finds a valid binary error vector of the LWE with binary error instance $(\mathbf{A}, \mathbf{b})$. The probability that Algorithm 5 terminates is*

$$p_0 = 2^{-r}\binom{r}{2c}\prod_{i=1}^{m}\left(1 - \frac{2}{B(\frac{m-1}{2}, \frac{1}{2})}\int_{-1}^{\max(-r_i,-1)}(1-t^2)^{\frac{m-3}{2}}dt\right),$$

Algorithm 5: The Hybrid Attack

Input: $q, r \in \mathbb{Z}$
$\mathbf{A} = (\mathbf{A}_1 | \mathbf{A}_2)$, where $\mathbf{A}_1 \in \mathbb{Z}_q^{m \times r}, \mathbf{A}_2 \in \mathbb{Z}_q^{m \times (n-r)}$
$\mathbf{b} \in \mathbb{Z}_q^m$
$\mathbf{B}$, a lattice basis of $\Lambda_q(\mathbf{A}_2)$

calculate $c = \lfloor r/4 \rceil$;
calculate $\mathbf{b}' = \mathbf{b} - (1/2) \cdot \mathbf{1}$;
while *true* **do**
  guess a binary vector $\mathbf{v}_1 \in \{0,1\}^r$ with c ones ;
  calculate $\mathbf{x}_1 = -\operatorname{NP}_{\mathbf{B}}(-\mathbf{A}_1\mathbf{v}_1) \in \mathbb{R}^m$;
  calculate $\mathbf{x}_2 = \operatorname{NP}_{\mathbf{B}}(\mathbf{b}' - \mathbf{A}_1\mathbf{v}_1) \in \mathbb{R}^m$;
  store $\mathbf{v}_1$ in all the boxes addressed by $\mathcal{A}_{\mathbf{x}_1}^{(r)} \cup \mathcal{A}_{\mathbf{x}_2}^{(r)}$;
  for *all* $\mathbf{v}_2 \neq \mathbf{v}_1$ *in all the boxes addressed by* $\mathcal{A}_{\mathbf{x}_1}^{(r)} \cup \mathcal{A}_{\mathbf{x}_2}^{(r)}$ **do**
    Set $\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2$ and calculate $\mathbf{x} = (1/2) \cdot \mathbf{1} + \operatorname{NP}_{\mathbf{B}}(\mathbf{b}' - \mathbf{A}_1\mathbf{v}) \in \mathbb{R}^m$;
    if $\mathbf{x} \in \{0,1\}^m$ *and* $\exists \tilde{\mathbf{s}} \in \{0,1\}^n : \mathbf{b} = \mathbf{A}\tilde{\mathbf{s}} + \mathbf{x} \mod q$ **then**
      return $\mathbf{x}$;
    end
  end
end

where $B(\cdot,\cdot)$ denotes the Euler beta function (see [Olv10]) and

$$r_i = \frac{\delta^{-2(i-1)+m} q^{\frac{m-n+r}{m}}}{2\sqrt{m/4}}.$$

In case that Algorithm 5 terminates, the expected number of operations is

$$2^{16} \binom{r}{c} \left(p \binom{2c}{c} \right)^{-1/2},$$

where

$$p = \prod_{i=1}^{m} \left(1 - \frac{1}{r_i B(\frac{m-1}{2}, \frac{1}{2})} J(r_i, m) \right),$$

and

$$J(r_i, m) = \begin{cases} \int_{-r_i-1}^{r_i-1} \int_{-1}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dy dz \\ \quad + \int_{r_i-1}^{-r_i} \int_{z-r_i}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dy dz & \text{for } r_i < \frac{1}{2} \\ \int_{-r_i-1}^{-r_i} \int_{-1}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dy dz & \text{for } r_i \geq \frac{1}{2}, \end{cases}$$

Proof. By definition, every output of Algorithm 5 is a valid binary error vector of the given LWE with binary error instance. The rest follows directly from the upcoming Lemma 3, Heuristic 2, and Heuristic 4. □

Remark 1. *Algorithm 5 gets some basis* $\mathbf{B}$ *as input. This basis has a certain quality, given by the Hermite delta* δ*. In practice, we can improve the attack by providing a basis with better, i.e., smaller, Hermite delta. We achieve this by running a basis reduction (e.g., BKZ) on* $\mathbf{B}$ *in a precomputation step. More details about the time necessary to achieve a certain Hermite delta and the trade-off between the runtime of the precomputation and the hybrid attack is given in Section 5.1.3.*

We postpone the proof of Theorem 3 to the end of this subsection, since we first need to develop some necessary tools. We start by giving the following definition, which is crucial to our analysis as the notion will be used frequently throughout this section.

Definition 4. *Let* $m \in \mathbb{N}$*. A vector* $\mathbf{x} \in \mathbb{Z}^m$ *is called* $\mathbf{y}$*-admissible for some vector* $\mathbf{y} \in \mathbb{Z}^m$ *if* $\mathrm{NP}(\mathbf{x}) = \mathrm{NP}(\mathbf{x} - \mathbf{y}) + \mathbf{y}$*.*

Intuitively, $\mathbf{x}$ being $\mathbf{y}$-admissible means that running the nearest plane algorithm on $\mathbf{x}$ and running it on $\mathbf{x} - \mathbf{y}$ yields the same lattice vector, since then we have $\mathbf{x} - \mathrm{NP}(\mathbf{x}) = (\mathbf{x} - \mathbf{y}) - \mathrm{NP}(\mathbf{x} - \mathbf{y})$.

We provide the following useful result about Definition 4.

Lemma 1. *Let* $\mathbf{t}_1 \in \mathbb{R}^m, \mathbf{t}_2 \in \mathbb{R}^m$ *be two arbitrary target vectors. Then the following are equivalent.*

1. $\mathrm{NP}(\mathbf{t}_1) + \mathrm{NP}(\mathbf{t}_2) = \mathrm{NP}(\mathbf{t}_1 + \mathbf{t}_2)$.
2. $\mathbf{t}_1$ *is* $\mathrm{NP}(\mathbf{t}_1 + \mathbf{t}_2)$*-admissible.*
3. $\mathbf{t}_2$ *is* $\mathrm{NP}(\mathbf{t}_1 + \mathbf{t}_2)$*-admissible.*

Proof. Let $\mathbf{t} = \mathbf{t}_1 + \mathbf{t}_2$ and $\mathbf{y} = \mathrm{NP}(\mathbf{t})$. By symmetry it suffices to show

$$\mathrm{NP}(\mathbf{t}_1) + \mathrm{NP}(\mathbf{t}_2) = \mathbf{y} \quad \Leftrightarrow \quad \mathrm{NP}(\mathbf{t}_1) = \mathrm{NP}(\mathbf{t}_1 - \mathbf{y}) + \mathbf{y}.$$

By definition, $\mathbf{t} - \mathbf{y}$ is a lattice vector and therefore $\mathrm{NP}(\mathbf{x} - (\mathbf{t} - \mathbf{y})) = \mathrm{NP}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^m$. This leads to

$$\mathrm{NP}(\mathbf{t}_1 - \mathbf{y}) = \mathrm{NP}(\mathbf{t}_1 - \mathbf{y} - (\mathbf{t} - \mathbf{y})) = \mathrm{NP}(\mathbf{t}_1 - \mathbf{t}) = \mathrm{NP}(-\mathbf{t}_2) = -\mathrm{NP}(\mathbf{t}_2).$$

Using this, one direction of the equivalence follows from

$$\mathrm{NP}(\mathbf{t}_1) + \mathrm{NP}(\mathbf{t}_2) = \mathrm{NP}(\mathbf{t}_1 - \mathbf{y}) + \mathbf{y} + \mathrm{NP}(\mathbf{t}_2) = -\mathrm{NP}(\mathbf{t}_2) + \mathbf{y} + \mathrm{NP}(\mathbf{t}_2) = \mathbf{y},$$

and the other from

$$\mathrm{NP}(\mathbf{t}_1) = \mathbf{y} - \mathrm{NP}(\mathbf{t}_2) = \mathbf{y} + \mathrm{NP}(\mathbf{t}_1 - \mathbf{y}).$$

□

As we will see in our analysis, the expected runtime heavily depends on the following probability. Let all notations be as in Theorem 3 and $\mathbf{e}' = \mathbf{e} - (1/2) \cdot \mathbf{1}$. For

$$W = \{\mathbf{w} \in \{0,1\}^r : \text{exactly } c \text{ entries of } \mathbf{w} \text{ are } 1\} \tag{5.3}$$

we define

$$p := \begin{cases} \Pr_{\mathbf{v}_1 \leftarrow W}[-\mathbf{A}_1\mathbf{v}_1 \text{is } \mathbf{e}'\text{-admissible} | \mathbf{v} - \mathbf{v}_1 \in W] & \text{if } \Pr_{\mathbf{v}_1 \leftarrow W}[\mathbf{v} - \mathbf{v}_1 \in W] > 0 \\ 0 & \text{else.} \end{cases} \tag{5.4}$$

Note that the analysis of the original attack on the NTRU encryption proposed by Howgrave-Graham [How07] also requires to calculate the probability p. In the original work, this is done experimentally. Replacing this probability estimation with the analytic methodology presented in the following removes the dependency on experimental support in the analysis of the hybrid attack. A first mathematical calculation of the probability p has already been presented by Hirschhorn et al. in [HHHW09]. However, their analysis requires an additional assumption that we no longer need.

Success Probability

In this subsection we determine the probability that Algorithm 5 terminates. We start by giving a sufficient condition for this event.

Lemma 2. *Let all notations be as in Theorem 3 and let* $\mathbf{b}' = \mathbf{b} - (1/2) \cdot \mathbf{1}$ *and* $\mathbf{e}' = \mathbf{e} - (1/2) \cdot \mathbf{1}$. *Assume that* $\mathbf{v}_1$ *and* $\mathbf{v}_2$ *are guessed in separate loops of Algorithm 5 and satisfy* $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}$. *Also let* $\mathbf{t}_1 = -\mathbf{A}_1\mathbf{v}_1$ *and* $\mathbf{t}_2 = \mathbf{b}' - \mathbf{A}_1\mathbf{v}_2$ *and assume* $\mathrm{NP}(\mathbf{t}_1) + \mathrm{NP}(\mathbf{t}_2) = \mathrm{NP}(\mathbf{t}_1 + \mathbf{t}_2) = \mathbf{e}'$ *holds. Then* $\mathbf{v}_1$ *and* $\mathbf{v}_2$ *collide in at least one box chosen during Algorithm 5 and the algorithm outputs the error vector* $\mathbf{e}$ *of the given LWE instance.*

Proof. According to the notation used in Algorithm 5, let $\mathbf{x}_1 = -\mathrm{NP}(\mathbf{t}_1)$ correspond to $\mathbf{v}_1$ and $\mathbf{x}_2 = \mathrm{NP}(\mathbf{t}_2)$ correspond to $\mathbf{v}_2$. By assumption we have $\mathbf{x}_1 = \mathbf{x}_2 - \mathbf{e}'$. Using the definition it is easy to verify that $\mathbf{x}_1$ and $\mathbf{x}_2$ share at least one common address, since $\mathbf{e}' \in \{-1/2, 1/2\}^m$. Therefore $\mathbf{v}_1$ and $\mathbf{v}_2$ collide in at least one box. Again by assumption, we obtain $\mathbf{x} = \mathrm{NP}(\mathbf{b}' - \mathbf{A}_1\mathbf{v}) = \mathrm{NP}(\mathbf{t}_1 + \mathbf{t}_2) = \mathbf{e}'$. Hence the algorithm outputs the error vector $\mathbf{e}$. □

In the following lemma we give a lower bound on the probability that Algorithm 5 terminates in case $\mathrm{NP}(\mathbf{b}' - \mathbf{A_1}\mathbf{v}) = \mathbf{e}'$. This condition is necessary for the algorithm to terminate. We then determine the probability that this condition is fulfilled and combine both probabilities to the overall success probability.

Lemma 3. *Let all notations be as in Theorem 3 and let* $\mathbf{b}' = \mathbf{b} - (1/2) \cdot \mathbf{1}$ *and* $\mathbf{e}' = \mathbf{e} - (1/2) \cdot \mathbf{1}$*. Assume that if* $\mathbf{v}$ *has exactly* $2c$ *one-entries, then* $p > 0$*, where* p *is as defined in Equation* (5.4)*. If* $\mathrm{NP}(\mathbf{b}' - \mathbf{A_1 v}) = \mathbf{e}'$*, then Algorithm 5 terminates with probability at least*

$$\tilde{p_0} = 2^{-r} \binom{r}{2c}.$$

Proof. We show that Algorithm 5 terminates if $\mathbf{v}$ consists of exactly $2c$ one-entries. The probability of this happening is exactly $\tilde{p_0}$, since there are 2^r binary vectors of length r, and $\binom{r}{2c}$ of them have exactly $2c$ one-entries. Assume that v consists of exactly $2c$ one-entries. The claim follows directly from Lemma 1 and Lemma 2. Since $p > 0$ there exist binary vectors $\mathbf{v}_1, \mathbf{v}_2 \in \{0,1\}^r$, each containing exactly c one-entries, such that $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}$ and $-\mathbf{A_1 v_1}$ is $\mathbf{e}'$-admissible. These vectors will eventually be guessed during Algorithm 5 if it does not terminate before. By Lemma 1 they satisfy

$$\mathrm{NP}(-\mathbf{A}_1\mathbf{v}_1) + \mathrm{NP}(\mathbf{b}' - \mathbf{A}_1\mathbf{v}_2) = \mathrm{NP}(\mathbf{b}' - \mathbf{A}_1\mathbf{v}) = \mathbf{e}'.$$

Lemmas 2 now guarantees that Algorithm 5 then outputs the error vector $\mathbf{e}$. □

It remains to calculate the probability that in fact it holds that $\mathrm{NP}(\mathbf{b}' - \mathbf{A_1 v}) = \mathbf{e}'$, which is equivalent to $\mathbf{e}' \in \mathcal{P}_{1/2}(\overline{\mathbf{B}})$. The probability that this is the case is calculated later, see Equation 5.6. Using Equation 5.6 and assuming independence we obtain the following estimate.

Heuristic 1. *Let all notations be as in Theorem 3. The probability that Algorithm 5 terminates is approximately*

$$p_0 \approx 2^{-r} \binom{r}{2c} \prod_{i=1}^{m} \left(1 - \frac{2}{B(\frac{m-1}{2}, \frac{1}{2})} \int_{-1}^{\max(-r_i,-1)} (1-t^2)^{\frac{m-3}{2}} dt\right).$$

Estimating the Number of Loops

The next step is to estimate the number of loops until the attack terminates.

Heuristic 2. *Let all notations be as in Theorem 3 and let* $\mathbf{b}' = \mathbf{b} - (1/2) \cdot \mathbf{1}$ *and* $\mathbf{e}' = \mathbf{e} - (1/2) \cdot \mathbf{1}$*. Assume that* $\mathrm{NP}(\mathbf{b}' - \mathbf{A_1 v}) = \mathbf{e}'$*, and that* $\mathbf{v}$ *consists of exactly* $2c$ *one-entries. Then the expected number of loops of Algorithm 5 is*

$$L \approx \binom{r}{c} \left(p \binom{2c}{c}\right)^{-1/2},$$

and the probability p, as given in Equation (5.4)*, is*

$$p \approx \prod_{i=1}^{m} \left(1 - \frac{1}{r_i B(\frac{m-1}{2}, \frac{1}{2})} J(r_i, m)\right),$$

with $B(\cdot,\cdot)$, $J(\cdot,\cdot)$, and r_i defined as in Theorem 3.

In the following, we justify the heuristic. Assume that $\mathbf{v}$ consists of exactly $2c$ one-entries. In addition to W (see Equation (5.3)), define the set

$$V = \{\mathbf{v}_1 \in W : \mathbf{v} - \mathbf{v}_1 \in W \text{ and } -\mathbf{A}_1\mathbf{v}_1 \text{ is } \mathbf{e}'\text{-admissible}\}.$$

Note that W is the set from which Algorithm 5 samples the vectors $\mathbf{v}_1$. Lemma 2 shows that the attack succeeds if two vectors $\mathbf{v}_1, \mathbf{v}_2 \in V$ satisfying $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}$ are sampled in different loops of Algorithm 5. Since otherwise the probability of success is close to zero, for simplicity we assume that the attack is only successful in this case. Therefore we need to estimate the necessary number of loops in Algorithm 5 until some $\mathbf{v}_1, \mathbf{v}_2 \in V$ with $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}$ are found. Note that by Lemma 1 if $\mathbf{v}_1 \in V$, then also $\mathbf{v}_2 = \mathbf{v} - \mathbf{v}_1 \in V$.

We start by calculating the probability that a vector sampled during Algorithm 5 lies in V. By definition of p, this probability is given by

$$\Pr_{\mathbf{v}_1 \xleftarrow{\$} W}[\mathbf{v}_1 \in V] = p_1 p, \text{ where } p_1 := \Pr_{\mathbf{v}_1 \xleftarrow{\$} W}[\mathbf{v} - \mathbf{v}_1 \in W].$$

Therefore we expect to sample a vector $\mathbf{v}_1 \in V$ every $\frac{1}{p_1 p}$ loops in Algorithm 5. The above equation also implies $p_1 p = \frac{|V|}{|W|}$, which gives us

$$|V| = p_1 p |W| = p_1 p \binom{r}{c}.$$

The probability p_1 is given by $p_1 = \binom{2c}{c} / \binom{r}{c}$ as can be seen in the following lemma.

Lemma 4. *Let all notations be as in Theorem 3. Let $\mathbf{v} \in \{0,1\}^r$ be binary with exactly $2c$ one-entries and W be defined as in Equation* (5.3)*. Then*

$$p_1 := \Pr_{\mathbf{v}_1 \xleftarrow{\$} W}[\mathbf{v} - \mathbf{v}_1 \in W] = \binom{2c}{c} / \binom{r}{c}.$$

Proof. There are exactly $\binom{r}{c}$ binary vectors of length r containing exactly c ones. For such a vector $\mathbf{v}_1$, the vector $\mathbf{v}_2 = \mathbf{v} - \mathbf{v}_1$ is binary and has exactly c one entries if and only if for all $i \in \{1, \ldots, r\}$ satisfying $(\mathbf{v}_1)_i = 1$, we also have $(\mathbf{v})_i = 1$. In other terms, the index set of one-entries of $\mathbf{v}_1$ has to be a subset of the index set of one-entries of $\mathbf{v}$, and there are exactly $\binom{2c}{c}$ such subsets. □

Therefore by the birthday paradox, the expected number of loops in Algorithm 5 until some $\mathbf{v}_1, \mathbf{v}_2 \in V$ with $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}$ are found can be estimated by

$$L \approx \frac{1}{p_1 p}\sqrt{|V|} = \frac{\sqrt{\binom{r}{c}}}{\sqrt{p_1 p}} = \binom{r}{c}\left(p\binom{2c}{c}\right)^{-1/2}.$$

It remains to approximate the probability p which we do in the following. Let $\mathbf{v}_1 \in \{0,1\}^r$ and $\mathbf{B}$ be some basis of $\Lambda_q(\mathbf{A}_2)$. By Lemma 1 there exist unique $\mathbf{u}_1, \mathbf{u}_2 \in \Lambda_q(\mathbf{A}_2)$ such that $\mathrm{NP}_{\mathbf{B}}(-\mathbf{A}_1\mathbf{v}_1) = -\mathbf{A}_1\mathbf{v}_1 - \mathbf{u}_1 \in \mathcal{P}_{1/2}(\overline{\mathbf{B}})$ and $\mathrm{NP}_{\mathbf{B}}(-\mathbf{A}_1\mathbf{v}_1 - \mathbf{e}') + \mathbf{e}' = -\mathbf{A}_1\mathbf{v}_1 - \mathbf{u}_2 \in \mathbf{e}' + \mathcal{P}_{1/2}(\overline{\mathbf{B}})$. Without loss of generality, in the following we assume $\mathbf{u}_1 = \mathbf{0}$, or equivalently $-\mathbf{A}_1\mathbf{v}_1 \in \mathcal{P}_{1/2}(\overline{\mathbf{B}})$. Now $-\mathbf{A}_1\mathbf{v}_1$ is $\mathbf{e}'$-admissible if and only if $\mathbf{u}_2 = \mathbf{u}_1 = \mathbf{0}$, which is equivalent to $\mathbf{e}' + \mathbf{A}_1\mathbf{v}_1 \in \mathcal{P}_{1/2}(\overline{\mathbf{B}})$. Therefore p is equal to the probability that $\mathbf{e}' + \mathbf{A}_1\mathbf{v}_1 \in \mathcal{P}_{1/2}(\overline{\mathbf{B}})$, which we determine in the following.

There exists some orthonormal transformation that aligns $\mathcal{P}_{1/2}(\overline{\mathbf{B}})$ along the standard axes of $\mathbb{R}^m$. By applying this transformation, we may therefore assume that $\mathcal{P}_{1/2}(\overline{\mathbf{B}})$ is aligned along the standard axes of $\mathbb{R}^m$ and that in consequence $\mathbf{e}'$ is a uniformly random vector of length $\sqrt{m/4}$. Because $\mathbf{A}_1$ is uniformly random in $\mathbb{Z}_q^{m\times r}$ we may further assume that $\mathbf{A}_1\mathbf{v}_1$ is uniformly random in $\mathcal{P}_{1/2}(\overline{\mathbf{B}})$, since without loss of generality we assume $\mathbf{A}_1\mathbf{v}_1 \in \mathcal{P}_{1/2}(\overline{\mathbf{B}})$. This gives rise to the following heuristic.

Heuristic 3. *The probability p as defined in Equation 5.4 (with respect to a reduced basis with Hermite delta δ) is*

$$p \approx \Pr_{\mathbf{t}\xleftarrow{\$}R,\, \mathbf{e}'\xleftarrow{\$}S_m(\sqrt{m/4})}[\mathbf{t} + \mathbf{e}' \in R],$$

where

$$S_m(\sqrt{m/4}) = \{\mathbf{x} \in \mathbb{R}^m \mid \|\mathbf{x}\| = \sqrt{m/4}\}$$

is the surface of a sphere with radius $\sqrt{m/4}$ centered around the origin and

$$R = \{\mathbf{x} \in \mathbb{R}^m \mid \forall i \in \{1,\ldots,m\} : -R_i/2 \le x_i < R_i/2\}$$

is the search rectangle with edge lengths

$$R_i = \delta^{-2(i-1)+m} q^{\frac{m-n+r}{m}}.$$

In the heuristic, the edge lengths are implied by the Geometric Series Assumption.

We continue calculating the approximation of p given in Heuristic 3. Let R and R_i be as defined in Heuristic 3. We can rewrite the approximation given in Heuristic 3 as

$$p \approx \Pr_{t_i\xleftarrow{\$}[-R_i/2,R_i/2],\mathbf{e}'\xleftarrow{\$}S_m(\sqrt{m/4})}[\forall i \in \{1,\ldots,m\} : t_i + e'_i \in [-R_i/2, R_i/2]].$$

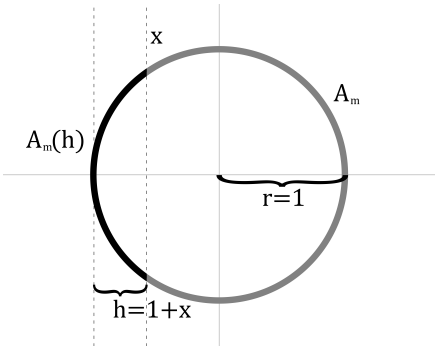


Figure 5.1.: Two-dimensional hyperspherical cap

Rescaling everything by a factor of $1/\sqrt{m/4}$ leads to

$$p \approx \Pr_{t_i \xleftarrow{\$} [-r_i, r_i], \mathbf{e}' \xleftarrow{\$} S_m(1)} [\forall i \in \{1, \ldots, m\} : t_i + e'_i \in [-r_i, r_i]],$$

where

$$r_i = \frac{R_i}{2\sqrt{m/4}} = \frac{\delta^{-2(i-1)+m} q^{\frac{m-n+r}{m}}}{2\sqrt{m/4}}. \tag{5.5}$$

Unfortunately, the distributions of the coordinates of $\mathbf{e}$ are not independent, which makes calculating p extremely complicated. In practice, however, the probability that $e_i \in [-R_i/2, R_i/2]$ is big for all but the last few indices i. This is due to the fact that by the Geometric Series Assumption typically only the last values R_i are small. Consequently, we expect the dependence of the remaining entries not to be strong. This assumption was already established by Howgrave-Graham [How07] and appears to hold for all values of R_i appearing in practice.

It is therefore reasonable to assume that

$$p \approx \prod_{i=1}^{m} \Pr_{t_i \xleftarrow{\$} [-r_i, r_i], e'_i \xleftarrow{\$} D_m} [t_i + e'_i \in [-r_i, r_i]],$$

were D_m denotes the distribution on the interval $[-1, 1]$ obtained by the following experiment: sample a vector $\mathbf{w}$ uniformly at random on the unit sphere and then output the first (equivalently, any arbitrary but fixed) coordinate of $\mathbf{w}$.

Next we explore the density function of D_m. The probability that $e'_i \leq x$ for some $-1 < x < 0$, where $e'_i \xleftarrow{\$} D_m$, is given by the ratio of the surface area of a hyperspherical cap of the unit sphere in $\mathbb{R}^m$ with height $h = 1 + x$ and the surface area of the unit sphere. This is illustrated in Figure 5.1 for $m = 2$. The surface area of a hyperspherical cap of the unit sphere in $\mathbb{R}^m$ with height $h < 1$ is given by (see [Li11])

$$A_m(h) = \frac{1}{2} A_m I_{2h-h^2}\left(\frac{m-1}{2}, \frac{1}{2}\right),$$

where $A_m = 2\pi^{m/2}/\Gamma(m/2)$ is the surface area of the unit sphere and

$$I_x(a,b) = \frac{\int_0^x t^{a-1}(1-t)^{b-1}dt}{B(a,b)}$$

is the regularized incomplete beta function (see [Olv10]) and $B(a,b)$ is the Euler beta function.

Consequently, for $-1 < x < 0$, we have

$$\begin{aligned}
\Pr_{e_i' \xleftarrow{\$} D_m}[e_i' \le x] &= \frac{A_m(1+x)}{A_m} \\
&= \frac{1}{2} I_{2(1+x)-(1+x)^2}\left(\frac{m-1}{2}, \frac{1}{2}\right) \\
&= \frac{1}{2} I_{1-x^2}\left(\frac{m-1}{2}, \frac{1}{2}\right) \\
&= \frac{1}{2B(\frac{m-1}{2}, \frac{1}{2})} \int_0^{1-x^2} t^{\frac{m-3}{2}}(1-t)^{-1/2}dt \\
&= \frac{1}{2B(\frac{m-1}{2}, \frac{1}{2})} \int_{-1}^{x} (1-t^2)^{\frac{m-3}{2}}(1-(1-t^2))^{-1/2}(-2t)dt \\
&= -\frac{1}{B(\frac{m-1}{2}, \frac{1}{2})} \int_{-1}^{x} (1-t^2)^{\frac{m-3}{2}}|t|^{-1}tdt \\
&= \frac{1}{B(\frac{m-1}{2}, \frac{1}{2})} \int_{-1}^{x} (1-t^2)^{\frac{m-3}{2}}dt. \qquad (5.6)
\end{aligned}$$

Together with

$$\Pr_{t_i \xleftarrow{\$} [-r_i, r_i]}[t_i \le x] = \int_{-r_i}^{x} \frac{1}{2r_i} dy,$$

we can use a convolution to obtain

$$\Pr_{t_i \xleftarrow{\$} [-r_i, r_i], e_i' \xleftarrow{\$} D_m}[t_i + e_i' \le x] = \frac{1}{2r_i B(\frac{k-1}{2}, \frac{1}{2})} \int_{-r_i-1}^{x} \int_{\max(-1, z-r_i)}^{\min(1, z+r_i)} (1-y^2)^{\frac{m-3}{2}} dydz.$$

Since

$$\Pr_{t_i \xleftarrow{\$} [-r_i, r_i], e_i' \xleftarrow{\$} D_m}[t_i + e_i' \in [-r_i, r_i]] = 1 - 2\left(\Pr_{t_i \xleftarrow{\$} [-r_i, r_i], e_i' \xleftarrow{\$} D_m}[t_i + e_i' < -r_i]\right),$$

it suffices to calculate the integral

$$J(r_i, m) = \int_{-r_i-1}^{-r_i} \int_{\max(-1, z-r_i)}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dydz \qquad (5.7)$$

in order to calculate p, which we do in the following. For the lower end of the inner integral, we have to distinguish two cases. If $r_i < 1/2$, we can split it into

$$J(r_i, m) = \int_{-r_i-1}^{r_i-1} \int_{-1}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dydz + \int_{r_i-1}^{-r_i} \int_{z-r_i}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dydz,$$

while in the simpler case $r_i > 1/2$ we have

$$J(r_i, m) = \int_{-r_i-1}^{-r_i} \int_{-1}^{z+r_i} (1-y^2)^{\frac{m-3}{2}} dydz.$$

This concludes our calculation of the probability p. All integrals can be calculated symbolically using sage [S+14], which allows an efficient calculation of p.

Time Spend per Loop cycle

With the estimation of the number of loops given, the remaining task is to estimate the time spend per loop cycle. Each cycle consists of four steps:

1. Guessing a binary vector.
2. Running the nearest plane algorithm (twice).
3. Calculating $\mathcal{A}_{\mathbf{x}_1}^{(r)} \cup \mathcal{A}_{\mathbf{x}_1'}^{(r)}$.
4. Dealing with collisions in the boxes.

We assume that the runtime of one inner loop of Algorithm 5 is dominated by the runtime of the nearest plane algorithm, as argued in the following. It is well known that sampling a binary vector is extremely fast. Furthermore, note that only very few of the 2^n addresses contain a vector, since filling a significant proportional would take exponential time. Consequently, collisions are extremely rare, and lines 8-11 of Algorithm 5 do not contribute much to the overall runtime.

An estimation by Howgrave-Graham [How07] shows that for typical instances, the runtime of the nearest plane algorithm exceeds the time spent for storing the collision. We therefore omit the latter from our considerations.

Lindner and Peikert [LP11] estimated the time necessary to run the nearest plane algorithm to be about 2^{-16} seconds, which amounts to about 2^{15} bit operations on their machine. This leads to the following heuristic for the runtime of the attack.

Heuristic 4. *The average number of operations per inner loop in Algorithm 5 is* $N \approx 2^{16}$.

Total Runtime

We are now able to prove our main theorem.

Proof (Theorem 3): By definition, every output of Algorithm 5 is a valid binary error vector of the given LWE with binary error instance. The rest follows directly from Heuristic 1, Heuristic 2, and Heuristic 4. ■

5.1.3. Minimizing the Expected Runtime

As previously mentioned in Remark 1, we can perform a basis reduction to obtain a lattice basis with smaller Hermite delta δ before running the actual attack in order to speed up the attack.

The Hermite delta δ determines the trade-off between the runtime of the pre-computation and the actual attack. More precisely, choosing a smaller value for δ increases the runtime of the basis reduction, but at the same time decreases the runtime of the actual attack, since it increases the success probability and probability that a vector is $\mathbf{e}'$-admissible. From the attacker's perspective it is necessary to optimise the choice of δ and r, the Meet-in-the-Middle dimension.

The Meet-in-the-Middle dimension r balances the trade-off between the Meet-in-the-Middle and the lattice part of the attack. On the one hand, increasing r increases the complexity of Meet-in-the-Middle part, since more entries of the secret have to be guessed. On the other hand, it also increases the determinant of the lattice, making CVP easier and thereby increasing the probability that vectors are $\mathbf{e}'$-admissible. Finding the optimal values for r and δ is therefore, at first sight, non-trivial. We perform this task in the following way. For each r we find the optimal δ that minimizes the runtime. We then take the optimal r and the corresponding δ to determine the overall minimal runtime. Since there are only finitely many possible values for r, this can be performed numerically. Figure 5.2 shows the expected runtime for the attack depending on the Meet-in-the-Middle dimension r.

5.2. Performance of Known Attacks

In this section we consider other approaches to solve LWE with binary error and compare these algorithms to Algorithm 5. In particular we give upper bounds for the runtimes of the algorithms. A comparison of the most practical attacks, including the hybrid attack, is given in Table 5.1.

Much of the analyzes below are in a similar spirit to that given in the survey [APS15] for methods of solving standard LWE. However it is often necessary to specifically adapt the analysis for the binary error case. Note that to solve LWE with binary error, in addition to algorithms for standard LWE, one may also be able

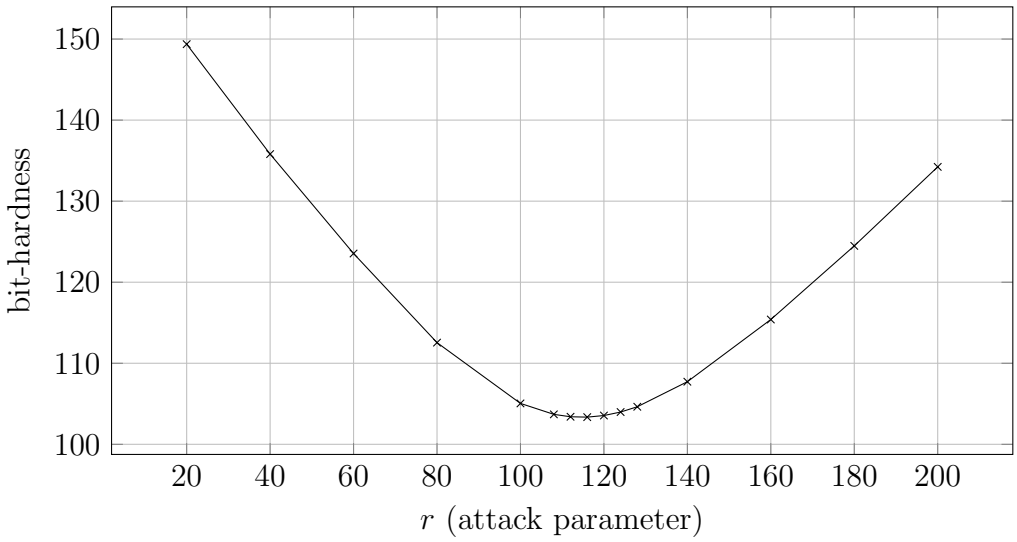


Figure 5.2.: Hardness of LWE instances with dimension of secret $n = 256$, number of samples $m = 512$ and modulus $q = 256$ for different values of r

to apply algorithms for the related Inhomogeneous Short Integer Solution problem. A discussion of these algorithms is given in [BGLS14].

5.2.1. Number of Samples

Some algorithms require a large number of LWE samples to be available in order to run. However it is well known (see, e.g., [DM13, MP13]) that if one has at least $m = \mathcal{O}(n^2)$ samples, the algorithm of Arora and Ge [AG11] solves LWE with binary error in polynomial time. Recall also that for reducing LWE with binary error to worst-case problems on lattices, one must restrict the number of samples to be $m = n\,(1 + \Omega(1/\log n))$ [MP13, Theorem 1.2]. On the other hand, with slightly more than linear samples, such as $m = \mathcal{O}(n \log\log n)$, the algorithm given in [ACF+14] is subexponential. Therefore if a scheme bases its security on the hardness of LWE with binary error, it is reasonable to expect that one has only access to at most linearly many samples. In the analysis below, we assume we the available number of samples is m, where m is linear in n. For concreteness, we fix $m = 2n$.

5.2.2. Algorithms for solving LWE

There are several approaches one could use to solve LWE or its variants (see the survey [APS15]). One may employ combinatorial algorithms such as the BKW [BKW03, ACF+15] algorithm and its variants [AFFP14, DTV15, GJS15, KF15]. However, all these algorithms require far more samples than are available in the binary error case, and are therefore ruled out. In the comparison we also omit a

Meet-in-the-Middle attack [APS15] or attacks based on the algorithm of Arora and Ge [AG11, ACF+14], as they will be slower than other methods, but nevertheless discuss them for completeness.

Distinguishing attack

We determine how small $\mathbf{v}$ must be for a successful distinguishing attack (see Section 2.3.2) as follows. Recall that our errors are chosen uniformly at random from $\{0, 1\}$. So they follow a Bernoulli distribution with parameter $1/2$, and have expectation $1/2$ and variance $1/4$. Consider the distribution of $\langle \mathbf{v}, \mathbf{e} \rangle$. Since the errors e_i are chosen independently, its expectation is $\frac{1}{2}\sum_{i=1}^{m} v_i$ and its variance is $\frac{1}{4}\sum_{i=1}^{m} v_i^2$. Since $\langle \mathbf{v}, \mathbf{e} \rangle$ is the sum of many independent random variables, asymptotically it follows a normal distribution with those parameters. Since the distinguishing attack success is determined by the variance and not the mean, and we can account for the mean, we assume it is zero. Then we can use the result of [LP11] to say that we can distinguish a Gaussian from uniform with advantage close to $\exp(-\pi(\|\mathbf{v}\| \cdot s/q)^2)$, where s is the width parameter of the Gaussian. In our case $s^2 = 2\pi \cdot \frac{1}{4}$ so we can distinguish with advantage close to $\epsilon = \exp(-\pi^2\|\mathbf{v}\|^2/2q^2)$. Therefore to distinguish with advantage ϵ we require a vector $\mathbf{v}$ of length $\|\mathbf{v}\| = q \cdot \frac{\sqrt{2\ln(1/\epsilon)}}{\pi}$.

We calculate a basis of the scaled dual lattice Λ and find a short vector $\mathbf{v} \in \Lambda$ by lattice basis reduction. With high probability the lattice Λ has rank m and volume q^n [MR09, APS15]. By definition of the Hermite delta we therefore have $\|\mathbf{v}\| = \delta^m q^{n/m}$. So the Hermite delta we require to achieve for the attack to succeed with advantage ϵ is given by $\delta^m q^{n/m} = q \cdot \frac{\sqrt{2\ln(1/\epsilon)}}{\pi}$. Assuming that the number of samples m is large enough to use the 'optimal subdimension' $m = \sqrt{n\log(q)/\log(\delta)}$ [MR09], we rearrange to obtain

$$\log \delta = \frac{\left(\log(q) + \log\left(\frac{\sqrt{2\ln(1/\epsilon)}}{\pi}\right)\right)^2}{4n\log(q)}.$$

To establish the estimates for the runtime of this attack given in Table 5.1, we assume one has to run the algorithm about $1/\epsilon$ times to succeed, and consider δ as a function of ϵ. The overall running time is then given by $1/\epsilon$ multiplied the estimated time, according to Lindner and Peikert [LP11], to achieve $\delta(\epsilon)$. We pick the optimal ϵ such that this overall running time is minimized.

It is possible that we do not have enough samples to use the 'optimal subdimension'. We firstly calculate δ assuming we have as many samples as we need for the 'optimal subdimension', then check that the m this corresponds to is indeed less than or equal to $2n$. If so, we use the runtime estimate for that δ. If not, we take $m = 2n$ and derive a Hermite delta using $\delta^{2n} q^{n/2n} = q \cdot \frac{\sqrt{2\ln(1/\epsilon)}}{\pi}$.

The number of operation necessary to achieve this Hermite delta is estimated using Equation (2.5).

Reducing to uSVP

Applying Kannan's reduction to attack our LWE instances is promising, since we have a very short error vector $\mathbf{e}$, which leads to a big gap $\frac{\lambda_1(\Lambda_q(A))}{\|\mathbf{e}\|}$. Since the error vector follows a binary distribution, we assume that its euclidian norm is given by $\|\mathbf{e}\| = \sqrt{m}/\sqrt{2}$. For all instances considered, the optimal number of samples exceeds $2n$, so we assume that the attack is successful if

$$\delta \leq \sqrt[m]{\frac{q^{1-n/m}}{\sqrt{\pi e}\tau}},$$

see Inequality (3.5) in Section 3.1.3.

Decoding

Recall (see also Section 2.3.3) that the decoding attack requires about $2^{15} \cdot \prod_{i=1}^{m} d_i$ operations. However, the analysis of the success probability on LWE with binary error is more complicated. By definition of the search parallelepiped, the attack succeeds if (and only if) the error $\mathbf{e}$ lies in the search rectangle $\mathcal{P}^{\mathbf{d}}_{1/2}(\mathbf{B})$. Under the same assumption as in Section 5.1.2 (and using the same error transformation), this probability can be estimated via

$$p_{\text{decoding}} \approx \prod_{i=1}^{m} \left(\Pr_{\mathbf{e}_i \xleftarrow{\$} D_m} [\mathbf{e}_i \in [-r_i, r_i]] \right)$$

where

$$r_i = d_i \frac{\delta^{-2(i-1)+m} q^{\frac{m-n}{m}}}{2\sqrt{m/4}}.$$

Together with Equation (5.6), this leads to

$$p_{\text{decoding}} \approx \prod_{i=1}^{m} \left(1 - \frac{2}{B(\frac{m-1}{2}, \frac{1}{2})} \int_{-1}^{\max(-r_i,-1)} (1-t^2)^{\frac{m-3}{2}} dt \right).$$

A standard way to increase the runtime of the attack is to use basis reduction (like BKZ2.0) as precomputation. Predicting the runtime of BKZ2.0 according to Equation (2.5) leads to the runtime estimation

$$T_{\text{decoding}} \approx \frac{2^{1.8/\log_2(\delta)-110} \cdot 2.3 \cdot 10^9 + 2^{15} \prod_{i=1}^{m} d_i}{p_{\text{decoding}}}.$$

Using the same numeric optimization techniques as presented above to minimize the expected runtime leads to the complexity estimates given in Table 5.1.

Meet-in-the-Middle Attack

We adapt the analysis in [APS15] to determine an upper bound on the complexity of a Meet-in-the-Middle attack on LWE with binary error. The proof follows [APS15] entirely analogously.

Theorem 4. *[APS15, Theorem 2] Let n, q parametrise an LWE instance with binary error. If there are m samples satisfying $m/q < 1/C$ for some constant $C > 1$ and $(2/q)^m \cdot 2^{n/2} = poly(n)$, then there is Meet-in-the-Middle algorithm which solves search LWE with binary error with non-negligible probability which runs in time $\mathcal{O}\left(2^{n/2}\left(\frac{n}{2}(5m+1)+(m+1)\log m\right)\right)$ and requires memory $m \cdot 2^{n/2}$.*

Proof. Given m samples $(\mathbf{a}_k, \langle \mathbf{a}_k, \mathbf{s}\rangle + e_k)$ split $\mathbf{a}_k = \mathbf{a}_k^l || \mathbf{a}_k^r$ in half and for each possibility $\mathbf{s}_i^l$ of the first half of $\mathbf{s}$ compute inner product of the first half of $\mathbf{a}_k$ and $\mathbf{s}_i^l$. Let the output of guess $\mathbf{s}_i^l$ for each of the m samples be $\mathbf{u}_{\mathbf{s}_i^l} = (\langle \mathbf{a}_1, \mathbf{s}_1^l \rangle, \ldots, \langle \mathbf{a}_m, \mathbf{s}_m^l \rangle)$. Store a table T whose entries map vectors $\mathbf{u}_{\mathbf{s}_i^l}$ to $\mathbf{s}_i^l$. Generating this table costs $m \cdot 2n \cdot 2^{n/2}$ operations since there are $2^{n/2}$ candidate secrets and for each we calculate m inner products. Sort the table into lexicographical ordering component-wise, which costs $\mathcal{O}\left(m \cdot 2^{n/2} \log\left(m \cdot 2^{n/2}\right)\right)$ operations. Now for each candidate $\mathbf{s}_j^r$ for the second half of the secret, and for each sample, compute $c_k - \langle \mathbf{a}_k^r, \mathbf{s}_j^r \rangle$, to form the vector $\mathbf{v}_{\mathbf{s}_j^r} = (c_1 - \langle \mathbf{a}_1^r, \mathbf{s}_j^r \rangle, \ldots, c_m - \langle \mathbf{a}_m^r, \mathbf{s}_j^r \rangle)$. Sort $\mathbf{v}_{\mathbf{s}_j^r}$ into T, which costs $\log |T| = \log\left(m \cdot 2^{n/2}\right) = \log m + n/2$ operations. Since there are $2^{n/2}$ possible second $\mathbf{s}_j^r$ the total cost of this step is $2^{n/2}(\log m + n/2)$. When $\mathbf{s}_j^r$ is sorted into the list, check which $\mathbf{u}_{\mathbf{s}_i^l}$ it is between. If $\mathbf{v}_{\mathbf{s}_j^r}$ and $\mathbf{u}_{\mathbf{s}_i^l}$ have a binary difference, return $\mathbf{s}_i^l$ and treat $\mathbf{s}_i^l || \mathbf{s}_j^r$ as a candidate secret, and check if it is correct. This procedure would then identify the correct secret, so long as there is not a wrap around mod q, since if $\mathbf{s}_{i'}^l || \mathbf{s}_{j'}^r$ is the correct secret then $\mathbf{v}_{\mathbf{s}_{j'}^r} - \mathbf{u}_{vecs_{i'}^l} = (e_1, \ldots, e_m) \mod q$ which is a binary vector. Let $b_k = \langle \mathbf{a}_k, \mathbf{s}\rangle \mod q$, then a wrap around error will not occur as long as $b_k \neq q-1$. The probability that one component has $b_k = q-1$ is $1/q$ so by the union bound the probability that at least one component has $b_k = q-1$ is $\leq m/q$. We want to bound m so that this event happens only with probability at most $1/C$ for some constant C, i.e., $m/q < 1/C$. It remains to consider the chance of a false positive, that is, an incorrect candidate secret $\mathbf{s}_i^l$ being suggested for some $\mathbf{s}_j^r$. Since a_k is uniformly random, for any $\mathbf{s}_i^l$, the vector $\mathbf{u}_{\mathbf{s}_i^l}$ is also random with each component taking one of q values. A wrong $\mathbf{s}_j^r$ will produce a $\mathbf{v}_{\mathbf{s}_j^r}$ that matches to $\mathbf{u}_{\mathbf{s}_i^l}$ only if its difference is 0 or 1 on every component. Therefore the chance of a false positive is $(2/q)^m$. There are $2^{n/2} - 1$ wrong choices for $\mathbf{s}_i^l$, so we expect to test $(2/q)^m \cdot 2^{n/2}$ candidates per $\mathbf{s}_j^r$. Hence we require $(2/q)^m \cdot 2^{n/2} = poly(n)$. □

Theorem 4 gives an upper bound on the complexity of a Meet-in-the-Middle attack, but note that it also takes at least $m \cdot 2n \cdot 2^{n/2}$ operations just to generate the table, excluding the costs of the other steps, e.g., sorting. Hence, we do not include

Instance	n	q	$\log_2(T_{\text{Hybrid}})$	$\log_2(T_{\text{Decoding}})$	$\log_2(T_{\text{uSVP}})$	$\log_2(T_{\text{Disting.}})$
I	128	256	**41**	67	82	**37**
II	160	256	**55**	77	122	**62**
III	192	256	**71**	88	162	**85**
IV	224	256	**87**	**102**	165	109
V	256	256	**103**	**117**	203	132
VI	288	256	**120**	**136**	254	154
VII	320	256	**136**	**158**	327	176
VIII	352	256	**153**	**185**	443	198

Table 5.1.: Comparison of attacks on LWE with binary error using at most $m = 2n$ samples. $\log_2(T_{\text{attack}})$ denotes the bit operations required to perform the algorithm described in 'attack' [MR09].

estimates for the runtime of this approach in Table 5.1, as there is always a faster choice.

Arora-Ge algorithm

The basic idea of the Arora-Ge algorithm [AG11] is setting up a system of nonlinear equations of which the secret is a root, and then solving the system. Solving may be via linearisation (as in [AG11]) or by Gröbner basis methods (as in [ACF+14]). The authors of [ACF+14] consider the complexity of their algorithm for solving LWE with binary error for various numbers of samples (see [ACF+14, Theorem 7]. In particular, if $m = 2n$ their algorithm solves LWE with binary error in time $\mathcal{O}(n^2 \cdot 2^{0.43\omega n})$ where $2 \leq \omega < 3$ is the linear algebra constant. Although this is an upper bound, it is significantly more than the cost of the other attacks. Therefore we do not expect that the actual runtime is smaller than for the other possible approaches, so we omit this algorithm from consideration in Table 5.1.

5.2.3. Comparison

Table 5.1 shows the runtime of the hybrid attack compared with some of the possible attacks described above on at most $m = 2n$ samples of LWE with binary error. For algorithms requiring lattice reduction, we choose whichever is the fewer of $m = 2n$ or the 'optimal subdimension' $m = \sqrt{n \log(q)/\log(\delta)}$ [MR09].

6 The LWE Challenge

The results and runtimes of solvers for lattice problem in practice are often much better than predicted by the proven bounds. This gap already appears in the basic LLL algorithm [LLL82], and carries over to other basis reduction algorithms and solvers for other lattice problems, especially for LWE. In this light, it is surprising that there was no exhaustive evaluation of the behavior of existing LWE solvers in practice so far. This makes comparing those attacks a very complicated task even for experts in the area.

In order to close this gap, we present the LWE challenge. It provides standardized LWE instances experts can use to prove the feasibility of their attacks. The instances have been created in a new fashion using a multi-party protocol that guarantees that none of the parties involved in the creation process has any advantage over other researchers.

The parameters have been chosen carefully such that the instances are at the same time representative for instances proposed for real-world cryptographic applications, and at the border between barely solvable and unsolvable instances. New solutions can be submitted directly to the challenge and are shown in a hall of fame. This will motivate researchers to compare their results to existing approaches in a realistic setting, revealing strengths and weaknesses of the attacks. At the same time, the comparison in the hall of fame serves as an easily accessable overview for non-experts.

In addition to the presentation of the LWE challenge, we give an overview and an interpretation of the first results.

Organisation Section 6.1 explains the parameter selection for the LWE instances provided by the challenge. The presentation of those instances is explained in Section 6.2. We investigated the submissions to the LWE challenge and present the first results in Section 6.3.

The LWE challenge was presented at AsiaPKC 2016 [BBG+16], the interpretation of the first results is original in this thesis.

6.1. Instances Provided by the LWE Challenge

In order to allow meaningful conclusions about the hardness of LWE, the parameter choice for the provided problem instances is crucial. On the one hand, if the provided instances are too easy, the attacks that are easier to implement would dominate the hall of fame, and not the attacks with the best runtime. On the other hand, if the instances are too hard, none of them would get broken. In both cases, the challenge would fail to provide useful information about the practical hardness of LWE.

In this section, we first explain how we chose the parameters of the instances, i.e., n, m, q, and α. Then, we show that every instance has (with high probability) a unique solution.

6.1.1. Choice of Parameters

LWE is parametrized by several different parameters. On the one hand, this is a big advantage: it allows to generate instances that are crafted specifically for a certain application, which leads to more efficient schemes. An example for this are the different moduli used in signature schemes: while there are examples of secure schemes with a modulus length of about 14 bits (see [DDLL13]), other techniques require the modulus to be about 30 bits long or even longer (see [BG14b]). On the other hand, the flexibility of LWE makes estimating its hardness much more complicated, since there is not one best algorithm for all instances [APS15]. This makes the selection of the right instances provided by the challenge a non-trivial task.

Fortunately, both theoretical and experimental results show that the hardness of LWE mainly depends on the secret dimension n, and the error parameter α (see [Reg05, BG14c, BLP+13, APS15]). The number of samples m and the modulus q appear to play a minor role. In the following, we explain our parameter choices in detail.

Known attacks on LWE (See Section2.3) can be roughly divided in two classes: lattice-based attacks (like the decoding attack [Bab86, LP11, LN13], the distinguishing attack [LP11], or the embedding approach [AFG13]) work with few samples, while other approaches (like BKW [ACF+15] or the Arora-Ge algorithm [AG11]) often require subexponentially many (or even more). In theory, this is not a big issue, since an attacker has access to arbitrarily many samples in the original definition of LWE. However, nearly all practical applications only provide a limited number of samples (e.g.,[LP11, DDLL13, BG14b]). Consequently, we consider modifying the latter attacks to run with less samples an important challenge. In fact, progress in this direction by lowering the required number of samples [DTV15] or generating new samples [Lyu05] shows that it may be possible to overcome this problem.

To motivate further progress, the LWE challenge provides only $m = n^2$ samples

per instance. This is enough to run all sample-efficient solvers, but should exclude the sample-consuming ones. Besides motivating further research for sample-efficient (and therefore realistic) attacks, this is leads to a more realistic picture about the limitations of current attacks in practice.

While being important for the correctness and the efficiency of many schemes, the modulus q appears to play a minor role for the hardness of LWE. Following the original proposal by Regev [Reg05], the LWE challenge is restricted to instances with q being the smallest prime that is bigger than n^2. Prime numbers are the most frequently choice in practical schemes (e.g., [Reg05, LP11, DDLL13, BG14b]). Fortunately, it is not necessary to include other values for q (like powers of 2). The reason for this is a technique introduced by Brakerski and Vaikuntanathan [BV14] called modulus switching. It allows an attacker to transfer an LWE instance with modulus q to an LWE instance with an arbitrary different modulus q' with the same secret $\mathbf{s}$.

The next choice concerns the size of the error. In the literature, the standard deviation of the gaussian error is either given by the standard deviation σ, or by the relative error size $\alpha = \sigma/q$. Following Regev's original proposal, we select α as error size parameter instead of σ. This choice is supported by modulus switching: When switching the modulus q to q', the relative error rate α remains constant except for a small factor, which shows that the hardness of LWE depends on α rather than on σ.

The last choice concerns concrete values for n and α. In the first cryptographic application of LWE [Reg05], Oded Regev proposed to to choose $\alpha = o(\frac{1}{\sqrt{n}\log(n)})$. For the instances provided by the challenge, the lattice dimension n ranges from 40 to 120, and the relative error size α ranges from 0.005 to 0.070. They are chosen such that they capture the proportion of n and α proposed by Regev (see Figure 6.1).

At the same time, the hardness of the instances lies in a reasonable range, i.e., the easiest instances can be solved fairly easy using standard techniques, while the hardest challenges are likely to remain unsolved for at least several years. The hardness estimates are based on the simulator by Albrecht et al. [APS15], that estimates the runtime of the known attacks on a given LWE instance. On the one hand, the applicability of this simulator should be taken with a grain of salt, since it was crafted for LWE instances with higher hardness levels (like the instances proposed for cryptographic applications). On the other hand, it should at least give an idea of the hardness of the instances, and comparing the performance of the attacks in practice to the values predicted by theory is an interesting future work made possible by the challenge. Additionally, we ran attacks on LWE instances with the easiest parameter sets to confirm that they are breakable within a reasonable time. The challenge was build such that more instances can be included once our estimations prove wrong or better algorithms which significantly decrease the hardness of LWE are developed.

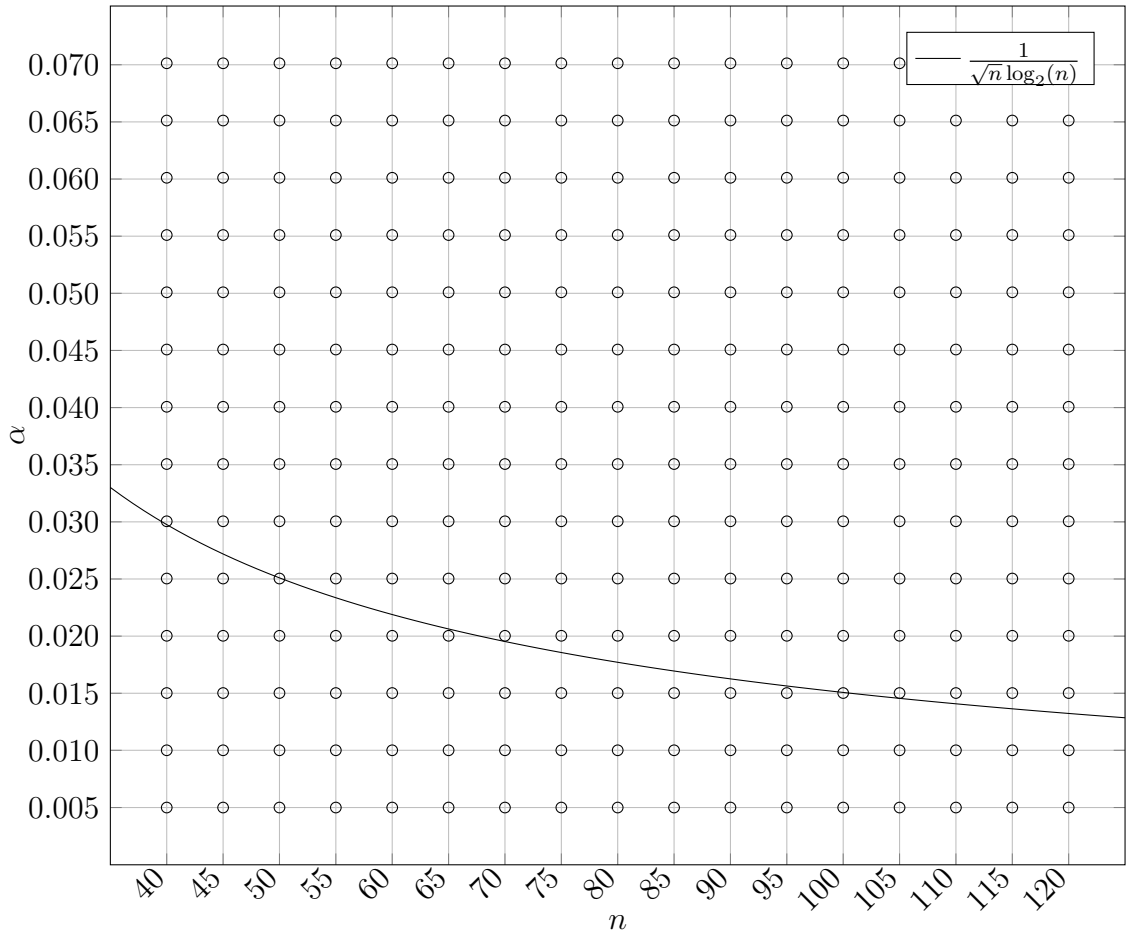


Figure 6.1.: Values for the error rate α and the dimension n as proposed by Regev (solid line) and provided by the LWE challenge (small circles)

6.1.2. Uniqueness and Correctness of Solutions

LWE is typically instantiated such that the solution is unique. This sounds surprising at first glance because for every $\mathbf{s} \in \mathbb{Z}_q^n$, there is an error vector $\mathbf{e} \in \mathbb{Z}_q^m$ such that $\mathbf{As} + \mathbf{e} = \mathbf{b}$. Since there is no bound for values sampled according to a Gaussian distribution, each $\mathbf{s}$ could be the secret. However, for a typical instantiation of LWE, there is only one vector $\mathbf{s}$ that leads to a reasonable error $\mathbf{e}$, by which we mean that all other errors are much bigger and therefore only sampled with a negligible probability.

For the LWE challenge, uniqueness is a little bit easier to define. The challenge accepts a submission $\mathbf{s}$ if (and only if) the corresponding error $\mathbf{e} = \mathbf{b} - \mathbf{As}$ satisfies $\|\mathbf{e}\| \leq 2\sqrt{m}\sigma$ with $\sigma = \alpha q$. This is justified by the fact that Lemma 2.2 in [DDLL13] by Ducas et al. bounds the size of a Gaussian distributed vector as

$$\Pr[\|\mathbf{e}\| > 2\sqrt{m}\sigma; \mathbf{e} \xleftarrow{\$} D_{\mathbb{Z}^m,\sigma}] < 2\left(2\exp(-3/2)\right)^m < 2^{-m+1}. \tag{6.1}$$

Note that this probability is extremely small for our values of m ranging from 1600 to 14400. Consequently, correct solutions get accepted with overwhelming probability.

In the following, we show that all challenges have (with high probability) one unique solution. For an arbitrary lattice $\Lambda \subset \mathbb{Z}^m$, let $\lambda_1(\Lambda)$ be the norm of the shortest non-zero vector in Λ. To see why the solutions are unique, imagine two secret-error tuples satisfying

$$\mathbf{As}_1 + \mathbf{e}_1 = \mathbf{b} = \mathbf{As}_2 + \mathbf{e}_2 \mod q$$

and $\|\mathbf{e}_i\| \leq 2\sqrt{m}\sigma$. The triangle inequality immediately leads to

$$\|\mathbf{A}(\mathbf{s}_1 - \mathbf{s}_2)\| \leq 4\sqrt{m}\sigma,$$

which shows that the lattice

$$\Lambda_q(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m \mid \exists \mathbf{w} \in \mathbb{Z}^n : \mathbf{Aw} = \mathbf{v} \mod q\}$$

contains two points $\mathbf{v}_1, \mathbf{v}_2$ satisfying $\|\mathbf{v}_1 - \mathbf{v}_2\| \leq 4\sqrt{m}\sigma$. Consequently, the existence of two LWE solutions would imply

$$\lambda_1(\Lambda_q(\mathbf{A})) \leq 4\sqrt{m}\sigma. \tag{6.2}$$

The following Lemma bounds the probability that such an extraordinary short lattice vector exists.

Lemma 5. *Let $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ be a uniformly random matrix. If $q^{1-\frac{n}{m}} \geq 1250$ then*

$$\Pr\left[\lambda_1(\Lambda_q(\mathbf{A})) \leq \frac{1}{5}\sqrt{m}q^{1-\frac{n}{m}}\right] \leq 0.9^{-m} + q^{n-m}.$$

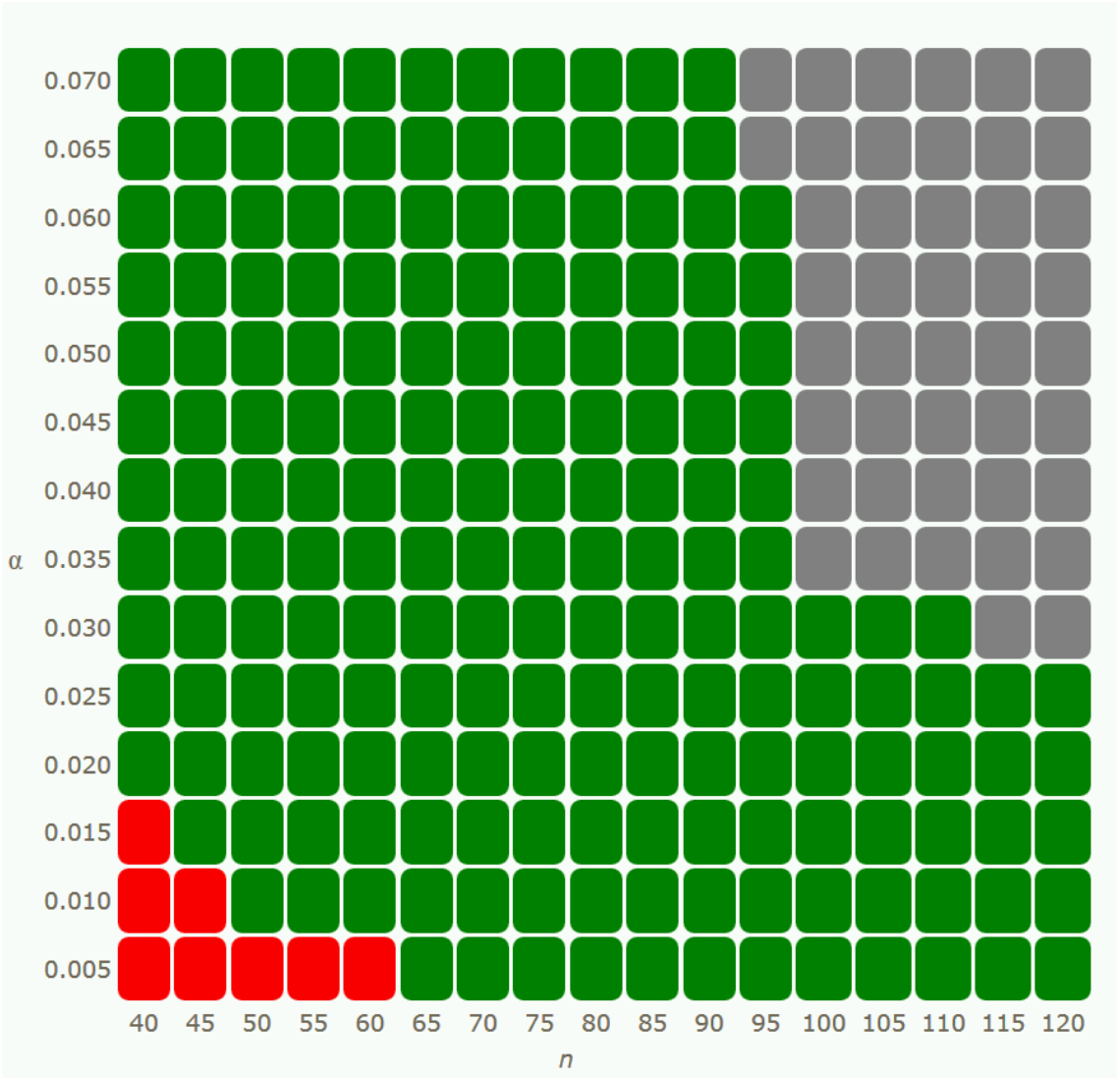


Figure 6.2.: The LWE challenge table. The lattice dimension n is shown on the x-axis, the relative error size α is shown on the y-axis. Green points stand for unsolved instances, while red points indicate that the respective instance has already been solved. By clicking on one of the points, the respective challenge can be downloaded. By clicking on a green point, additionally a solution to the challenge can be submitted.

Proof. The idea of the proof is to consider the probability of a random integer vector being in the lattice and take a union bound over all short vectors. Assume we have $\det(\Lambda_q(\mathbf{A})) = q^{m-n}$ (which happens with probability at least $1 - q^{n-m}$). The probability of a random integer vector to be in the lattice is q^{n-m}. The tricky part is to bound the number of integer vectors inside the ball of radius $1/5\sqrt{m}q^{1-\frac{n}{m}}$.

Note that using cubes of edge length 1 centered at each integer vector, one can tile the entire space. However, not all cubes centered at integer points inside the ball are completely enclosed by the ball. So we consider a second ball with radius larger than the first one by an addititive factor of $\frac{1}{2}\sqrt{m}$ to ensure that all cubes centered inside the original ball are entirely enclosed in the second ball. This allows us to bound the number of integer points inside the first ball by the volume of the second ball. Note that we can obtain the necessary extension of the radius by multiplying the radius of the first ball with 1.002 due to the condition in the lemma. So by the union bound and Stirling approximation of the Gamma function we have

$$\begin{aligned}
&\Pr\left[\lambda_1(\Lambda_q(\mathbf{A})) \leq \frac{1}{5}\sqrt{m}q^{1-\frac{n}{m}} \,|\, \det(\Lambda_q(\mathbf{A})) = q^{m-n}\right] \\
&\leq \frac{\pi^{m/2}}{\Gamma(\frac{m}{2}+1)}\left(0.2004\sqrt{m}q^{\frac{m-n}{m}}\right)^m q^{n-m} \\
&\leq \Gamma\left(\frac{m}{2}+1\right)^{-1}\left(0.2004^2\pi m\right)^{m/2} \\
&\leq \left(\frac{0.2004^2\pi m}{m/2+1}\right)^{m/2} \\
&\leq (\sqrt{2\pi e}0.2004)^m \\
&\leq 0.9^m.
\end{aligned}$$

□ □

Corollary 1. *Let $\mathbf{A} \in \mathbb{Z}_q^{m\times n}, \mathbf{b} \in \mathbb{Z}_q^m$ be an LWE instance with parameters n, q, α, and m. If $m = n^2$, $q^{1-\frac{n}{m}} \geq 1250$, and*

$$\alpha \sqrt[n]{q} < 1/20, \tag{6.3}$$

the probability that two different vectors $\mathbf{s}_1 \in \mathbb{Z}_q^n, \mathbf{s}_2 \in \mathbb{Z}_q^n$ satisfy

$$\|\mathbf{b} - \mathbf{A}\mathbf{s}_i\| \leq \sqrt{m}\alpha q$$

is bounded by $0.9^{-n^2} + q^{n-n^2}$.

Proof. Follows directly from Equation (6.2), Lemma 5, and the easy calculation

$$\begin{aligned}
& 4\sqrt{m}\sigma < 1/5\sqrt{m}\cdot q^{1-n/m} \\
\Leftrightarrow \quad & 20\alpha q < q^{1-1/n} \\
\Leftrightarrow \quad & \alpha\sqrt[n]{q} < 1/20.
\end{aligned}$$

□

While all proposed instances meet the condition in Lemma 5, not all satisfy Equation (6.3). However, this does not mean that the solutions of the other challenges are not unique. To the contrary, the Gaussian heuristic strongly indicates that all solutions are unique: it estimates the length of the shortest non-zero vector in the above lattice to be

$$\lambda_1(\Lambda_q(\mathbf{A})) \approx \frac{\Gamma(1+m/2)^{1/m}}{\sqrt{\pi}} q^{1-n/m}.$$

Consequently, for all our instances, two valid solutions would imply a lattice vector shorter than 0.7 times the Gaussian heuristic. However, the existence of such a short vector is very unlikely. This is, among others, confirmed by the results of the SVP challenge: despite big efforts by many researchers, no one was able to find a lattice vector shorter than 0.8 times the prediction of the Gaussian heuristic so far.

6.2. The Challenge Web Page

In this section, we explain the challenge web page in more detail and show how one can participate in the LWE challenge.

6.2.1. How to Download Challenges

The LWE challenge website provides a challenge table that is shown in Figure 6.2. This table contains all available instances, ordered by lattice dimension n and relative error rate α. The green points of the challenge table stand for unsolved instances. By clicking on a green point, the respective instance can be downloaded.

On the day of the release of the LWE challenge website, all points of the challenge table will be green. Once an instance is solved, i.e., the correct solution has been submitted, the respective point turns red. Hence, while the challenge table provides the challenge instances, it also serves as a visual representation of the development of the LWE challenge and hence, of the LWE problem.

In addition to using the challenge table, LWE instances can also be downloaded directly in the download section (right column). After selecting n and α, a click on the download button leads directly to the file containing the corresponding challenge.

The LWE challenge website also provides some smaller instances at the download section. These toy challenges could for example be used to test the correctness of an LWE solver implementation.

Format of the Challenges The LWE challenges are provided in the following format: in the first three rows, the integers n, m, and q are listed. In the fourth row, the real α is found. It is written in US notation, i.e., with a period as decimal point. In the fifth row the vector $\mathbf{b}$ - which actually is a column vector - is given and finally in the sixth row, the matrix $\mathbf{A}$ starts. It consists of m rows which n

entries each. A description of the format of the instances can also be found at the download section on the LWE challenge website.

6.2.2. How to Submit Solutions

The LWE challenge accepts a submitted solution $\mathbf{s}$ for a challenge $\mathbf{A}, \mathbf{b}$ with parameters n, m, α, and q if (and only if) $\|\mathbf{b} - \mathbf{As}\| \leq 2\sqrt{m}\alpha q$. Equation (6.1) shows that such an $\mathbf{s}$ gets accepted with overwhelming probability if $\mathbf{b}$ was actually created as $\mathbf{As} + \mathbf{e} \mod q$. On the other hand, as mentioned in Section 6.1, with very high probability all instances have a unique solution.

Analogous to downloading instances, there are two possibilities for submitting a solution: first, the solution can be submitted by clicking on the respective green point in the challenge table on the website. This will lead to a submission form where the lattice dimension and the Gaussian parameter are already entered. Second, the solution can be submitted by following the submission link on the right side of the start page of the LWE challenge website. This link leads to a submission form where both the lattice dimension and the Gaussian parameter can be selected independently.

Note that for the solved instances, i.e., those represented with a red point in the challenge table, no solution can be submitted. Solutions for the toy instances can not be submitted either.

Hall of Fame At the bottom of the LWE challenge website's start page, we show the latest five successful submissions. We also provide a list with all successful submissions, i.e., a hall of fame. It can be found by clicking on a link below the latest submissions. Here, all correct solutions are listed in a chronological order, together with some meta information.

Once an instance is broken, there is no way to find a better solution. Therefore, old results will not be suppressed and hence, these early contributions will stay visible. This will prevent a picture similar to the one on the original lattice challenge, where only few names dominate the hall of fame since as long as the shortest vector is not found, better submissions can replace older solutions.

6.3. First Results of the LWE Challenge

About one month after publication, eight instances of LWE have been broken. All solutions come from the same group, and have been produced using the embedding technique.

This picture follows the expected pattern. Also for other lattice challenges, the first result came from the easiest to implement algorithms, rather than the algorithms with the best expected performance. Those algorithms showed up a few

0.015	4749 / 52				
0.010	220 / 2.4	1614 / 3.8			
0.005	2 / 0.2	5 / 0.24	48 / 0.27	2201 / 0.31	1382690 / 30
α / n	40	45	50	55	60

Table 6.1.: Overview of successful attacks times / attack time predictions (both in seconds)

months later, followed by new developments. We expect the same to happen for the LWE challenge.

A second interesting aspect is the runtime of the attacks. Since the authors gave details about the running time, we can compare their concrete runtimes with predictions stemming from the LWE estimator by Albrecht et al. [APS15]. This comparison is given in Table 6.1. It shows that there is a significant gap between the predicted hardness of the instances, and the concrete hardness is practice. It is necessary to point out that this is probably mainly due to the fact that the main purpose of the tool by Albrecht et al. is to predict the hardness of instances used in cryptography, which are way harder than the instances considered in the challenge. Consequently, we expect the gap between the runtime estimations and experimental runtimes to shrink for harder instances. If this is not the case, it may question the accuracy of the existing hardness estimations and lead to interesting new results.

Part II.

Selecting Parameters

7 Signature Scheme by Bai/Galbraith

Selecting parameters for LWE-based cryptographic schemes is often a complicated and error-prone process. Even for experts in the field, it is easy to overestimate the security of the scheme, or to miss opportunities to optimize the parameter choice. In this chapter, we discuss this issue for the parameter selection for the signature scheme by Bai and Galbraith [BG14b].

In the original paper, the authors give concrete parameters aiming for 128 bits of security. However, due to an error in the analysis of the attacks, they underestimate its runtime, which leads directly to an overestimation of the security. We address this problem by giving a corrected security analysis based on state-of-the-art estimations of attack performances [LP11, AFG13, BG14b, APS15].

In addition, we show that careful parameter selection can lead to an instantiation that is at the same time more secure and more efficient than the one given by Bai and Galbraith. The cause of this improvement is two-fold: the first relies on the fact that the hardness of the scheme is not only build on LWE, but also on SIS. We show that the original parameter set does not balance the hardness of both problems perfectly, and doing so leads to an improved performance. Second, as mentioned in Chapter 3, the number of samples provided significantly influences the hardness of the LWE instance. This fact was used before to exclude certain attacks [LP11]. We show that in addition to that, a careful quantitative analysis of the remaining attacks leads to more realistic hardness estimations, which allows to select parameters that lead to more efficient implementations.

Organisation Section 7.1 introduces the scheme given by Bai and Galbraith [BG14b] and the slightly modified variant considered in the rest of this chapter. An updated analysis of the old parameter set is given in Section 7.2, together with a comparison to the old analysis. Finally, Section 7.3 presents our improved methodology to select parameters and the new parameter sets.

This chapter is based on a publication at Latincrypt 2014 [DBG+14].

7.1. The Scheme

The Bai-Galbraith digital signature scheme [BG14b] (BG signature) is based on the Fiat-Shamir paradigm which transforms an identification scheme into a signature scheme [FS87] and closely follows previous proposals by Lyubashevsky et al. [Lyu12, GLP12, DDLL13, Lyu09]. The hardness of breaking the BG signature scheme, in the random oracle model, is reduced to the hardness of solving standard worst-case computational assumptions on lattices. The explicit design goal of Bai and Galbraith is having short signatures.

7.1.1. Description of the BG Signature Scheme

For easy reference, the key generation, signing, and the verification algorithm of the BG signature scheme are given in Figure 7.1. Our proposed parameter set is summarized in Table 7.3. An analysis of the original parameter sets is given in Section 7.2. However, the algorithms have been simplified and redundant definitions have been removed (e.g., we just use σ as standard deviation and do not differentiate between $\sigma_{\mathbf{E}}, \sigma_{\mathbf{S}}$ and set $n = k$, folllowing the proposal by Bai and Galbraith in both cases).

During key generation two secret matrices $\mathbf{S} \in \mathbb{Z}^{n \times n}, \mathbf{E} \in \mathbb{Z}^{m \times n}$ are sampled from a discrete Gaussian distribution $D_\sigma^{n \times n}$ and $D_\sigma^{m \times n}$, respectively. A rejection condition CHECK_E enforces certain constraints on $\mathbf{E}$, which are necessary for correctness and short signatures (see Section 7.1.2). Finally, the public key $\mathbf{T} = \mathbf{AS} + \mathbf{E}$ and the secret key matrices $\mathbf{S}, \mathbf{E}$ are returned where $\mathbf{AS}$ is the only matrix-matrix multiplication necessary in the scheme. As we choose $\mathbf{A} \in \mathbb{Z}^{m \times n}$ as a global constant, it does not have to be sampled during key generation and is also not included in the public key and secret key.

For signing, the global constant $\mathbf{A}$ as well as secret keys $\mathbf{S}, \mathbf{E}$ are required (no usage of $\mathbf{T}$ in this variant). The vector $\mathbf{y}$ is sampled uniformly random from $[-B, B]^n$. For the instantiation of the random oracle H (using a hash function) only the higher order bits of $\mathbf{Ay}$ are taken into account and hashed together with the message μ. The algorithm $F(c)$ takes the binary output of the hash c and produces a vector $\mathbf{c}$ of weight ω (see [DDLL13] for a definition of $F(c)$). In a different way than [BG14b] $\mathbf{w}$ is computed following an idea that has also been applied in [GLP12]. Instead of computing $\mathbf{w} = \mathbf{Az} - \mathbf{Tc} \pmod q$ we calculate $\mathbf{w} = \mathbf{v} - \mathbf{Ec} \pmod q$, where $\mathbf{v} = \mathbf{Ay} \pmod q$. This is also the reason why $\mathbf{E}$ has to be included into the secret key $sk = (\mathbf{S}, \mathbf{E}) \in \mathbb{Z}^{n \times n} \times \mathbb{Z}^{m \times n}$. Thus, the large public key $\mathbf{T} \in \mathbb{Z}^{m \times n}$ is not needed anymore for signing and the operations become simpler. The test whether $|[\mathbf{w}_i]_{2^d}| > 2^{d-1} - L_{BG}$ ($L_{BG} = 7\omega\sigma$ in [BG14b]) ensures that the signature verification will not fail on a generated signature ($\mathbf{w}$ is never released) and the last line ensures that the signature is uniformly distributed within the allowed range $[-B+U, B-U]^n$

Algorithm KeyGen	Algorithm Sign	Algorithm Verify
INPUT: $\mathbf{A}, n, m, q, \sigma$ OUTPUT: $sk = (\mathbf{S}, \mathbf{E}), pk = (\mathbf{T})$ 1. $\mathbf{S} \xleftarrow{\$} D_\sigma^{n \times n}$ 2. $\mathbf{E} \xleftarrow{\$} D_\sigma^{m \times n}$ 3. **if** CHECK_E$(\mathbf{E}) = 0$ **then** Restart 4. $\mathbf{T} = \mathbf{AS} + \mathbf{E} \pmod q$ 5. **return** $sk = (\mathbf{S}, \mathbf{E}), pk = (\mathbf{T})$	INPUT: $\mu, \mathbf{A}, \mathbf{S}, \mathbf{E}, B, U, d, w, \sigma$ OUTPUT: $(\mathbf{z}, c)$ 1. $\mathbf{y} \xleftarrow{\$} [-B, B]^n$ 2. $\mathbf{v} = \mathbf{Ay} \pmod q$ 3. $c = H(\lfloor \mathbf{v} \rceil_d, \mu)$ 4. $\mathbf{c} = F(c)$ 5. $\mathbf{z} = \mathbf{y} + \mathbf{Sc}$ 6. $\mathbf{w} = \mathbf{v} - \mathbf{Ec} \pmod q$ 7. **if** $\lvert[\mathbf{w}_i]_{2^d}\rvert > 2^{d-1} - L$ **then** Restart 8. **return** $(\mathbf{z}, c)$ **if** $\lVert\mathbf{z}\rVert_\infty \leq B - U$	INPUT: $\mu, \mathbf{z}, c, \mathbf{A}, \mathbf{T}, B, U, d$ OUTPUT: Accept/Reject 1. $\mathbf{c} = F(c)$ 2. $\mathbf{w} = \mathbf{Az} - \mathbf{Tc} \pmod q$ 3. $c' = H(\lfloor \mathbf{w} \rceil_d, \mu)$ 4. **if** $c' = c$ and $\lVert\mathbf{z}\rVert_\infty \leq B - U$ **then return** 1 5. **return** 0

Figure 7.1.: The BG signature scheme [BG14b]; see Section 7.1.2 for implementations of CHECK_E.

for $U = 14 \cdot \sigma\sqrt{\omega}$.

For verification the higher order bits of $\mathbf{w} = \mathbf{Az} - \mathbf{Tc} = \mathbf{Ay} - \mathbf{Ec}$ are hashed and a valid signature $(\mathbf{z}, c)$ is accepted if and only if $\mathbf{z}$ is small, i.e., $\lVert\mathbf{z}\rVert_\infty \leq B - U$, and $c = c'$ for $c' := H(\lfloor \mathbf{w} \rceil_d, \mu)$. For the security proof and standard attacks we refer to the original work [BG14b].

7.1.2. Optimizing Rejection Sampling

In the original signature scheme [BG14b] CHECK_E_{BG} restarts the key generation if $\lvert\mathbf{E}_{i,j}\rvert > 7\sigma$ for any (i, j) and the rejection condition in Line 7 of Sign is $\lvert[\mathbf{w}_i]_{2^d}\rvert > 2^{d-1} - L_{BG}$ for $L_{BG} = 7w\sigma$. This ensures that it always holds that $\lfloor \mathbf{Ay} \rceil_d = \lfloor \mathbf{Ay} - \mathbf{Ec} \rceil_d$ and thus verification works even for the short signature. However, in practice the acceptance probability of $(1 - 14\omega\sigma/2^d)^m$ has a serious impact on performance and leaves much room for improvement. On first sight it would seem most efficient to test during signing whether $\lfloor \mathbf{Ay} \rceil_d = \lfloor \mathbf{Ay} - \mathbf{Ec} \rceil_d$ and just reject signatures that would not be verifiable. However, in this case the proof structure given in the full version of [BG14b] does not work anymore. In Game 1, sign queries are replaced by a simulation (in the random oracle model) which is not allowed to use the secret key and later on has to produce valid signatures even for an invalidly chosen public key (Game 2).

The authors of [DBG+14] propose an optimization (similar to [DDLL13]) that rejects $\mathbf{E}$ during key generation only if the error generated by $\mathbf{Ec}$ in $\lfloor \mathbf{Ay} \rceil_d = \lfloor \mathbf{Ay} - \mathbf{Ec} \rceil_d$ for the worst-case $\mathbf{c}$ is larger than a threshold L. Thus, the CHECK_E_{new}

Table 7.1.: Parameter sets proposed by Bai and Galbraith [BG14b]

Parameter Sets by Bai/Galbraith					
Parameter	Instance I	Instance II	Instance III	Instance IV	Instance V
n	640	576	512	512	400
m	1137	969	945	1014	790
σ	58	68	66	224	70
ω	18	18	19	19	20
d	24	24	24	26	24
B	2201370	2322422	2058115	6985118	1748695
q	$2^{34.34}$	$2^{33.10}$	$2^{30.84}$	$2^{32.66}$	$2^{28.71}$
U	3446	4039	4028	13670	4383

algorithm works the following: Using $\max_k(\cdot)$ which returns the k-th largest value of a vector we compute thresholds $t_h = \sum_{k=1}^{\omega} \max_k(|\mathbf{E}_h|), \forall h \in [0, m]$ where $\mathbf{E}_h$ is the h-th row of $\mathbf{E}$ and reject if one or more t_h are larger than L. Thus the rejection probability for the close-to-uniform $\mathbf{w}$ is independent of $\mathbf{c}$ and $\mathbf{E}$ and does not leak any information. When L is chosen such that only a small percentage of secret keys are rejected the LWE instances generated by the public key are still hard due to the same argument on the bounded number of samples as in [BG14b, DDLL13]. The acceptance probability of $\mathbf{w}$ in Line 7 of Sign is $(1 - 2L/2^d)^m$. Table 7.3 shows concrete values for our choice of L_{new} and the original L_{BG}.

7.2. Revisiting the Old Parameter Sets

In the original definition of LWE, an attacker has access to arbitrary many LWE samples (i.e. he can choose the value of m). Most hardness analyzes ignore the upper bound on the number of samples (if there is one), which leads to a conservative security estimation. As we showed in Chapter 3, taking this upper bound into consideration, however, has some interesting impacts. We predicted the runtime of the attacks on LWE with and without this bound on m. As can be seen in Table 7.2, the decoding attack is the fastest attack on all instances (see Table 7.1) if we look at the original definition of LWE with arbitrary big m, followed by the standard embedding approach and the embedding approach in [COT14].

However, considering the bound on m changes this picture completely. The decoding attack is suddenly the slowest approach, and the previously slowest attack becomes the fastest way to break most of the instances. This is due to the fact that m is big enough to run the embedding approach from [COT14] in the optimal

Security Level								
	Instance I		Instance II		Instance III		Instance IV	
Attack	Dim.	Sec.	Dim.	Sec.	Dim.	Sec.	Dim.	Bit Sec.
LWE with optimal m								
Decoding	1432	86	1305	80	1176	76	1238	83
Embedding	1511	101	1381	94	1244	90	1314	100
Embedding	945	122	876	114	801	114	875	123
LWE with given m								
Decoding	1137	168	969	216	945	129	1014	130
Kannan	1137	126	969	138	945	111	1014	119
ISIS	945	122	876	114	801	114	875	123
SIS								
Lattice reduction		301		225		192		206

Table 7.2.: Runtime of attacks on the LWE instances from [BG14b] with arbitrary many and a bounded number of samples for the decoding attack [LP11], Kannans embedding [AFG13] and the embedding via ISIS [BG14b].

dimension (in fact, only 945 out of the 1137 samples are needed for an optimized attack on Instance I). Running the decoding attack on the same instance, on the other hand, would require nearly 300 samples more than provided by the scheme. Interestingly, the standard embedding approach would ideally make use of even more samples, but suffers less from the restriction.

A second issue with the old parameter sets is that the hardness of LWE and SIS is very unbalanced. Attacking the underlying SIS instance is always several orders of magnitude harder than attacking the LWE instance. Since breaking one of the assumptions suffice to break the scheme, this wastes a lot of potential. Consequently, balancing the hardness of SIS and LWE is one major goal for the new parameter set.

The biggest issue with the old parameter sets, however, is the error in the security analysis. When estimating the hardness of the embedding attack (following the approach presented in Section 2.3.5), Bai and Galbraith did not take the heuristic constant τ into consideration. This may look like a minor problem, but has an huge impact on the security estimates: Analyzing Instance I in Table 7.1 with the techniques introduced in Section 3.1.4 for arbitrary many samples leads to $\delta \approx 1.0061$. Bai and Galbraith, however, estimated the necessary hermit delta by $\delta \approx 1.0056$. Converting this to bit security estimations shows that the original work overestimates the hardness of Instance I by nearly 20 bits.

7.3. Security Analysis and Parameter Selection

In the original work [BG14b], Bai and Galbraith proposed five different parameter sets (see Table 7.2) to instantiate their signature scheme. In this section we revisit their security analysis and propose a new instantiation that is optimized for software implementations on modern server and desktop computers (Intel/AMD) and also mobile processors (ARM). The security analysis has been refined due to the following reasons: First, as mentioned before, a small negligence in the assessment of the underlying LWE instances leads to a slightly wrong hardness estimation, which was acknowledged by the authors after publication [BG14a]. Second, an important attack, namely the decoding attack, was not considered in [BG14b]. We justify that indeed the decoding attack is less efficient than the one considered if one takes into account the limited number of samples m given to the attack algorithms.

Third, the hardness of the two underlying problems (namely SIS and LWE) is unbalanced for the original parameter sets. Table 7.2 shows that the hardness of the underlying SIS always significantly exceeds the hardness of the corresponding LWE instance. Even when considering the upper bound on the number of samples, this gap reaches up to 179 bits for the first parameter sets. Since the security of the scheme corresponds to the hardness of the easier problem, this is a suboptimal situation.

In Table 7.3 we propose a parameter set for an instantiation of the signature scheme from Section 7.1 with 128 bits of security, for which we provide evidence in the next section. The hardness of the LWE instances obtained from the parameters proposed in the original work [BG14b] is analyzed in Section 7.2.

7.3.1. Hardness of LWE

We estimated the hardness of the LWE instances by considering the decoding attack, Kannan's embedding approach, and the reduction via ISIS, while taking into consideration the bounded number of samples. Those sample restriction are also the reason we did not consider other attacks like BKW or Arora-Ge. Using the techniques introduced before (see Chapter 2 and 3), we come to the estimation

$$\delta \leq \sqrt[m+n+1]{\frac{q^{\frac{m}{m+n+1}}}{\sqrt{2\pi e}\sigma\tau}}.$$

Together with the runtime estimation of BKZ taken from Albrecht et al. [AFG13], this leads to the results presented in Table 7.4.

7.3.2. Hardness of SIS

Instead of recovering the secret key, which corresponds to solving an instance of LWE, an attacker could also try to forge a signature directly and thus solve an SIS instance. We predict the hardness of SIS for the well-known lattice-reduction attack (see for example [BBD09]) like it was done in [BG14b]. This attack views SIS as a variant of the (approximate) shortest-vector problem and finds the short vector by applying a basis reduction. As stated by Bai and Galbraith, forging a signature through this attack requires to find a reduced basis with Hermite factor

$$\delta = (D/q^{m/(m+n)})^{1/(n+m+1)}, \tag{7.1}$$

with $D = (\max(2B, 2^{d-1}) + 2E'\omega)$ for E' satisfying $(2E')^{m+n} \geq q^m 2^{132}$. Applying Equation (2.5), we estimate that a successful forger requires to perform about 2^{159} operations (see Table 7.4).

7.3.3. An Instantiation for Software Efficiency

Choosing optimal parameters for the scheme is a non-trivial multi-dimensional optimization problem and our final parameter set is given in Table 7.3. Since the probability that the encoding function F maps two random elements to the same value must be negligible (i.e. smaller than 2^{-128}), we choose ω such that

$$2^{\omega}\binom{n}{\omega} \geq 2^{128}. \tag{7.2}$$

Since $\mathbf{Sc}$ is distributed according to a Gaussian distribution with parameter

$$\sigma_{Sc} = \sqrt{\omega}\sigma \tag{7.3}$$

we can bound its entries by

$$U = 14\sigma_{Sc}\,. \tag{7.4}$$

Following the proposal by Bai and Galbraith, we set

$$B = n \cdot U \tag{7.5}$$

Consequently, $B - U$ is lower bounded by $14\sqrt{\omega}\sigma(n-1)$, such that the acceptance probability of a signature P_{acc} (Line 8 in Figure 7.1) is at least

$$P_{acc} = \left(\frac{2(B-U)+1}{2B}\right)^m = \left(\frac{2 \cdot 14\sqrt{\omega}\sigma(n-1)+1}{2 \cdot 14\sqrt{\omega}\sigma n+1}\right)^m \approx \left(1-\frac{1}{n}\right)^m .$$

Experimental results show that choosing

$$L = 3\omega\sigma \tag{7.6}$$

leads to an success probability of line 3 in KeyGen of about 99%. The next important choice to be made is the value for the parameter d. It has a determining influence on the trade-off between runtime and key sizes: The success probability in the signing algorithm (Line 7 in Figure 7.1) is given by $(1-2L/2^d)^m$, which means that large values for d lead to a high success probability, and thereby to fewer rejections implying better running times. Bounding this success probability by 1/3 leads to the restriction

$$(1-2L/2^d)^m \geq 1/3, \tag{7.7}$$

which can be used to determine the smallest possible value for d.

On the other hand, the security proof requires

$$(2B)^n q^{m-n} \geq 2^{(d+1)m+\kappa} \tag{7.8}$$

to be satisfied, which means that increasing d implies larger values for q, hence, worsening runtime and key sizes. With this, we collected all conditions necessary to conclude all parameters from n, σ, and m. An overview on the dependencies is given in Figure 7.2.

Our goal is to come up with a parameter set that ensures at least 128 bits of security. Furthermore, dimensions n resp. m are multiples of four to support four parallel operations in vector registers, and multiples of seven for additional platform-dependent optimization of the implementation. The secret dimension n is the main security parameter: one the one hand, increasing it increases the hardness of both SIS and LWE. One the other hand, n also has the biggest influence on key and signature sizes, and also on the runtime of the scheme. Consequently, keeping n as small as possible is the main design goal.

The important second parameter influencing the hardness of SIS is the dimension m. However, increasing m also means providing more samples to attack LWE, therefore decreasing the hardness of LWE. This property can be used to fine-tune the hardness of LWE and SIS. Figure 7.3 shows the lower bound on q depending on the choice of m and σ. In order to support small key sizes and efficient operations, we want q to be as small as possible, so the natural candidates for σ are the values right before the lower bound on q performs a jump. We can see that tempering m moves this jumps and influences the natural choices for σ and q. We found $n = 532, m = 840$, and $\sigma = 43$ to be the best choice, since it balances the hardness of LWE and SIS pretty well (see Table 7.4), and at the same time permits to use the modulus $q = 2^{29} - 3$, which leads to very efficient modulo operations. The choice $n = 532$ leads to $\omega = 18$, which results in the lower bound $\log_2(B) \geq 20.4$ that allows our choice $B = 2^{21} - 1$.

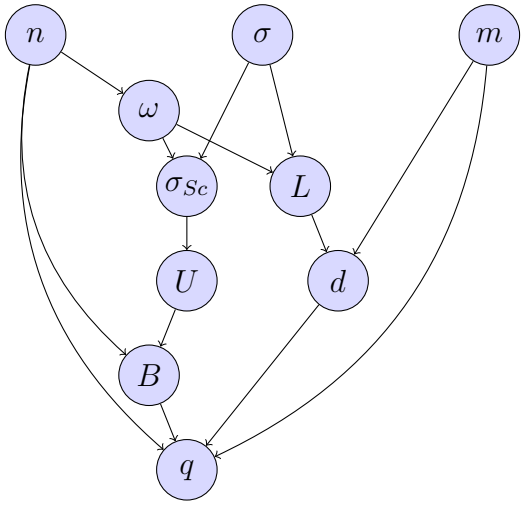


Figure 7.2.: Dependencies of the parameters

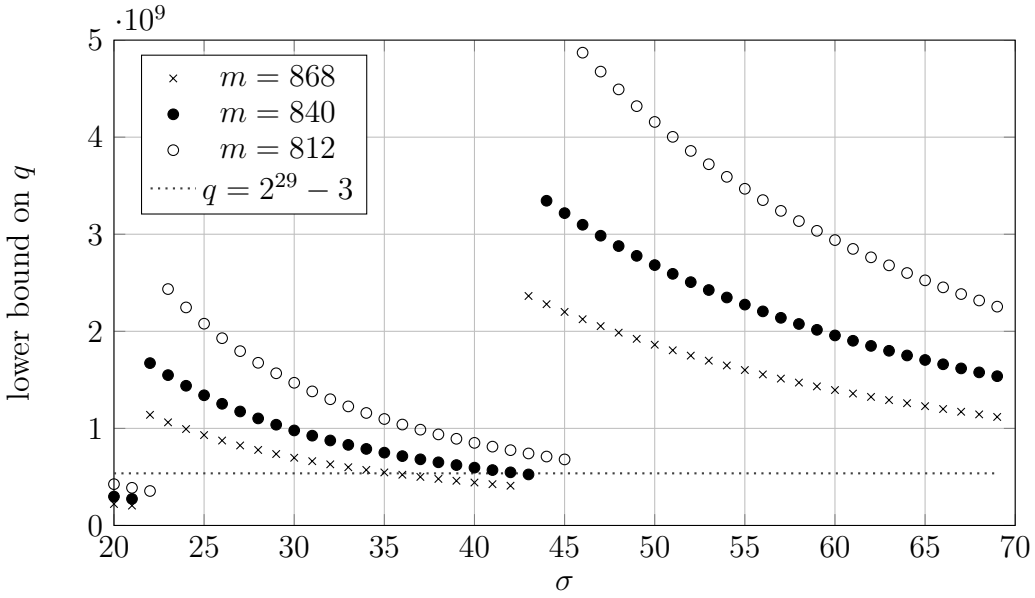


Figure 7.3.: Lower bound on q for $n = 532$ and various values of m

Table 7.3.: The parameter set we use for 128 bits of security. Note that signature and key sizes refer to fully compressed signature and keys. Our software uses slightly a larger (padded) signature and keys to support faster loads and stores aligned to byte boundaries.

Parameter Selection		
Parameter	Bound	Value
n		532
m		840
σ		43
ω	$2^{\omega}\binom{n}{\omega} \geq 2^{128}$	18
d	d is s.t. $(1 - 2L/2^d)^m \geq 1/3$	23
σ_{Sc}	$\sqrt{\omega}\sigma$	182.43
B	close to power of two $\geq 14\sqrt{\omega}\sigma(n-1)$	$2^{21} - 1$
q	$\geq \left(2^{(d+1)m+\kappa}/(2B)^n\right)^{1/(m-n)}$	$2^{29} - 3$
U	$14 \cdot \sigma\sqrt{\omega}$ (Prob. of acceptance Line 8 of Sign: 0.51)	2554.1
L_{BG}	$7w\sigma$ (Prob. of acceptance Line 3 of KeyGen: ≈ 1) (Prob. of acceptance Line 7 of Sign: 0.337)	5418
L_{new}	$3w\sigma$ (Prob. of acceptance Line 3 of KeyGen: 0.99) (Prob. of acceptance Line 7 of Sign: 0.628)	2322
public-key size	$m \cdot n \cdot \lceil \log_2(q) \rceil$	1.54 MiB
secret-key size	$(n^2 + n \cdot m) \lceil \log_2(14 \cdot \sigma) \rceil$	0.87 MiB
signature size	$n \cdot \lceil \log_2(2B) \rceil + 256$	11960 bits

Table 7.4.: Security of our parameter set

Security Level		
Problem	Attack	Bit Security
LWE	Decoding [LP11]	271
	Embedding [AFG13]	192
	Embedding [BG14b]	130
SIS	Lattice reduction [BG14b]	159

8 A New and Highly Efficient Encryption Scheme

Many cryptographic schemes are not based in standard LWE, but on variants like ring-LWE [LP11], or LWE with modified secret [DDLL13] or error distribution [BGG+16, CGW14]. The reason for this is that the additional structure often allows to increase the efficiency of the scheme in many regards (e.g., key sizes, message/signature sizes, runtimes). It can even allow to construct new primitives, with Gentry's fully homomorphic encryption being the most prominent example. In this chapter, we show that using variants of LWE can also help to transfer existing solutions to new environments.

To this end, we present a variant of the well-known Lindner/Peikert encryption scheme [LP11] that is based on binary LWE as introduced in Chapter 5. Using the hardness estimations from Chapter 5, we select parameters that lead to a secure instantiation, that is at the same time suitable for usage on microcontrollers. Based on this results, Buchmann et al. [BGG+16] showed that a microcontroller implementation in fact leads to an extremely practical solution outperforming even established approaches like elliptic-curve cryptography or RSA.

Organisation Section 8.1 introduces the new scheme, which is a variant of encryption scheme proposed by Lindner and Peikert [LP11]. Our variant requires a new correctness proof that is given in Section 8.2. Finally, Section 8.3 proposes a concrete instantiation based on the hardness results for binLWE from Chapter 5 and the correctness results from Section 8.2. For completeness, it also contains a comparison of the implementation given by Buchmann et al. [BGG+16] with established alternatives.

This chapter is based in a paper that appeared AsiaPKC@AsiaCCS 2016. This paper also contains the microcontroller implementation by Güneysu, Oder, and Pöppelmann.

$\textsc{Gen}(\mathbf{a})$:	Choose $\mathbf{r}_1, \mathbf{r}_2$ uniformly at random among the polynomials in $\mathcal{R}_q$ with binary entries and let $\mathbf{p} = \mathbf{r}_1 - \mathbf{a}\mathbf{r}_2 \in \mathcal{R}_q$. The public key is $\mathbf{p}$ and the secret key is $\mathbf{r}_2$.
$\textsc{Enc}(\mathbf{a}, \mathbf{p}, \mathbf{m} \in \{0,1\}^n)$:	Choose $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ uniformly at random among the polynomials in $\mathcal{R}_q$ with binary entries. Let $\bar{\mathbf{m}} = \textsc{encode}(\mathbf{m}) \in \mathcal{R}_q$, and compute the ciphertext $[\mathbf{c}_1 = \mathbf{a}\mathbf{e}_1 + \mathbf{e}_2, \mathbf{c}_2 = \mathbf{p}\mathbf{e}_1 + \mathbf{e}_3 + \bar{\mathbf{m}}] \in \mathcal{R}_q^2$.
$\textsc{Dec}(\mathbf{c} = [\mathbf{c}_1, \mathbf{c}_2], \mathbf{r}_2)$:	Output $\textsc{decode}(\mathbf{c}_1\mathbf{r}_2 + \mathbf{c}_2) \in \{0,1\}^n$.

Figure 8.1.: The Scheme R-BinLWEEnc

8.1. Ring-LWE Based Public Key Encryption Scheme

In this section we introduce the required notation, describe our public key encryption scheme, show the probability analysis for decoding failures, and provide parameters.

8.1.1. Preliminaries

Since its introduction by Regev [Reg05], the learning with error problem (LWE) served as a fundamental building block for an astonishing variety of cryptographic schemes. To set up an LWE-distribution for integers n, q, and an error distribution ψ over $\mathbb{Z}_q$, one samples a secret $\mathbf{s} \in \mathbb{Z}_q^n$ according to ψ^n. To create an LWE-sample, one samples a vector $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, an error $e \in \mathbb{Z}_q$ according to ψ, and outputs the tuple $(\mathbf{a}, b)$ with $b = \mathbf{a}^T\mathbf{s} + e$. The LWE-samples can be collected to get $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$ for an $m \times n$ matrix $\mathbf{A}$, the secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ and two vectors $\mathbf{e}, \mathbf{b} \in \mathbb{Z}_q^m$. Note that LWE can also be defined with different distributions for error $\mathbf{s}$ and secret $\mathbf{e}$, but it is known that any LWE instance can be transformed into an LWE instance with secrets distributed according to the error distribution.

The most efficient lattice-based schemes are based on a more structured variant of LWE, called Ring-LWE [LPR10]. While certain properties can be established for various rings, we define $\mathcal{R}$ as the ring $\mathbb{Z}[x]/\langle x^n + 1\rangle$ and $\mathcal{R}_q$ as $\mathbb{Z}_q[x]/\langle x^n + 1\rangle$ for an integer q, $\mathbb{Z}_q = \mathbb{Z}/(q\mathbb{Z})$, and a power of two n. We write elements $\mathbf{p} \in \mathcal{R}_q$ with maximum degree $n-1$ as $\mathbf{p} = \sum_{i=0}^{n-1}[\mathbf{p}]_i x^i$ with $[\mathbf{p}]_i \in (-\lfloor q/2\rfloor + 1, \lfloor q/2\rfloor)$ being the i-th coefficient.

To setup a Ring-LWE-distribution for integers n, q, and an error distribution ψ over $\mathcal{R}_q$, one samples a secret $\mathbf{s} \in \mathcal{R}_q$ according to ψ. To create a Ring-LWE-sample, one samples a polynomial $\mathbf{a} \in \mathcal{R}_q$ uniformly at random, and an error $\mathbf{e} \in \mathcal{R}_q$ according to ψ, and outputs the tuple $(\mathbf{a}, \mathbf{b})$ with $\mathbf{b} = \mathbf{a}\mathbf{s} + \mathbf{e}$.

The search variant of the (Ring-)LWE problem is to find $\mathbf{s}$, given arbitrary many (Ring-)LWE-samples. The decision variant is to distinguish between arbitrary many (Ring-)LWE-samples and the same number of samples with uniform $\mathbf{a}$ and b (respectively $\mathbf{a}$ and $\mathbf{b}$).

Typical choices for ψ are discrete (or discretized) Gaussian distributions or uniform distributions over a small set. Complementary to Ring-LWE, we define the Ring-BLWE problem as the instance of Ring-LWE, where ψ is the uniform distribution on the polynomials in $\mathcal{R}_q$ with binary coefficients (i.e., coefficients in $\{0, 1\}$).

8.1.2. The Scheme

In [LP11, LPR10] a semantically secure public key encryption scheme (from now referred to as R-LWEEnc) is described whose security is based on the hardness of the Ring-LWE problem. While it has been shown that high-performance implementations are possible [GFS+12, RVM+14, dCRVV15], the scheme still has the disadvantage of large ciphertexts and requires complex sampling of discrete Gaussian noise. However, its simple structure can be used as starting point to derive a variant which is much better suited for practice. The main difference is that we now use binary errors and secrets instead of errors or secrets chosen from a Gaussian distribution. While just using binary noise in the R-LWEEnc scheme seems straightforward, the choice of the encoding and decoding functions and the decryption error analysis is not. The scheme (from now referred to as R-BinLWEEnc) is defined in Figure 8.1. It is parameterized by integers n and q and uses a uniformly random chosen global constant polynomial $\mathbf{a} \in \mathcal{R}_q$. Furthermore, it requires a pair of error-tolerating encoding and decoding functions. Our instantiations of these functions are given in Equations (8.1) and (8.2), the justification for this choices in Section 8.2.

As we replace Gaussian by binary noise our scheme is now based on the Ring-BLWE problem whose hardness is assessed in Section 8.2.2. Following [LP11], we can conclude that the scheme remains semantically secure as long as decisional Ring-BLWE in dimension n with modulus q is hard. We note that in R-BinLWEEnc, $\mathbf{r}_1$ in Gen is noise and hence not needed anymore after key generation. The scheme uses the error-tolerant encoding and decoding functions

encode: $\{0,1\}^n \to \mathcal{R}_q$,

$$(m_0, \ldots, m_{n-1}) \mapsto \sum_{i=0}^{n-1} m_i \cdot (q/2)x^i \tag{8.1}$$

and decode : $\mathcal{R}_q \to \{0,1\}^n$ as given in Equation (8.2). Note that decode differs from the decoding function by Lindner and Peikert. This is due to the

fact that the binary distribution is (unlike the Gaussian distribution) typically not centered around zero and this asymmetry of the error leads to an asymmetry of the coefficients. In the next section, we discuss the implications of the different decoding function on the correctness of the scheme.

8.2. Correctness and Security of R-BinLWEEnc

Similar to the correctness result of [LP11], the message is decrypted correctly if (and only if)

$$\text{DECODE}(\text{ENCODE}(\mathbf{m}) + \mathbf{e}_1\mathbf{r}_1 + \mathbf{e}_2\mathbf{r}_2 + \mathbf{e}_3) = \mathbf{m}.$$

Note that for all $i \in \{1, 2\}$ and $k \in \{0, \ldots, n-1\}$

$$[\mathbf{e}_i\mathbf{r}_i]_k = \sum_{j=0}^{k} [\mathbf{e}_i]_j [\mathbf{r}_i]_{k-j} - \sum_{j=k+1}^{n-1} [\mathbf{e}_i]_j [\mathbf{r}_i]_{n+k-j},$$

and the coefficients $[\mathbf{e}_i]_j, [\mathbf{r}_i]_j$ are statistically independent and binary. Consequently, the coefficients $[\mathbf{e}_i\mathbf{r}_i]_k$ are approximately distributed according to a Gaussian distribution modulo q (this is basically a random walk). Therefore, the distribution of the coefficients of the noise polynomial $\mathbf{n} = \mathbf{e}_1\mathbf{r}_1 + \mathbf{e}_2\mathbf{r}_2 + \mathbf{e}_3$ is likewise close to a Gaussian distribution. The natural choice for the decoding function is therefore to determine the expected values $\mathrm{E}([\mathbf{n}]_k)$ and decode all coefficients closer to $\text{ENCODE}(0) + \mathrm{E}([\mathbf{n}]_k)$ to zero and the elements closer to $\text{ENCODE}(1) + \mathrm{E}([\mathbf{n}]_k)$ to one. Since

$$\begin{aligned} \mathrm{E}([\mathbf{e}_i\mathbf{r}_i]_k) &= \mathrm{E}(\sum_{j=0}^{k} [\mathbf{e}_i]_j [\mathbf{r}_i]_{k-j} - \sum_{j=k+1}^{n-1} [\mathbf{e}_i]_j [\mathbf{r}_i]_{n+k-j}) \\ &= (-n + 2k + 2)/4, \end{aligned}$$

the desired expected value is

$$\begin{aligned} \mathrm{E}([\mathbf{n}]_k) &= \mathrm{E}([\mathbf{e}_1\mathbf{r}_1]_k) + \mathrm{E}([\mathbf{e}_2\mathbf{r}_2]_k) + \mathrm{E}([\mathbf{e}_3]_k) \\ &= k - n/2 + 3/2. \end{aligned}$$

Therefore, we choose the decoding function

DECODE: $\mathcal{R}_q \to \{0,1\}^n$,

$$\sum_{k=0}^{n-1} \alpha_k x^k \mapsto (m_0, \ldots, m_{n-1}) \tag{8.2}$$

with

$$m_k = \begin{cases} 0 & \text{if } |\alpha_k - k - \lfloor \frac{n-3}{2} \rceil| \leq \frac{q}{4} \\ 1 & \text{else.} \end{cases}$$

Please note that since all the above probability distribution have finite support, it is possible to calculate the probability of decoding errors exactly. In fact, we used Sage [S+14], a computer algebra program, to calculate the probabilities in Table 8.1.

8.2.1. Parameter Selection

In Table 8.1 we provide three parameter sets for R-BinLWEEnc based on the security analysis (see Section 8.2.2). For comparison, we also include selected NTRU [HHHW09] and Ring-LWE Encryption (R-LWEEnc) [LP11] parameter sets. Note that it is also possible to use a q between 128 and 512 to balance security and error probability between the different parameter sets given in Table 8.1. While even a small failure probability like 2^{-32} is clearly undesirable in practice, some applications are able to deal well with such a small probability (e.g., interactive applications already have to account for data corruption during transmission). It can further be seen that our proposal leads to smaller key and ciphertext sizes than R-LWEEnc and is comparable to NTRU. Only the size optimized (but computationally more complex) parameter set NTRU ($d_f = 113$) allows a slightly smaller ciphertext. For a more detailed comparison of different lattice-based encryption schemes we refer to [CWB14]. Additionally, we would like to note that the ciphertext size could presumably be reduced in future work by removing redundant information in the ciphertext [PG13] or by techniques presented by Peikert [Pei14].

8.2.2. Hardness Assessment of Binary LWE

In this section, we discuss the theoretical and concrete hardness of Ring-BLWE.

Theoretical Hardness of Ring-BLWE

When Regev [Reg05] introduced LWE, he provided a quantum reduction that showed its worst-case hardness if at least one of two well-known lattice problems (namely gapSVP, the decisional variant of the shortest vector problem SVP, and SIVP, the shortest independent vector problem) is hard in the average case. Nowadays, there

Set/Scheme	n	q	Bit Sec.	Failure Probability	Size [bits]			
					Message	Secret Key	Public Key	Ciphertext
R-BinLWEEnc-I	256	128	94	2^{-10}	256	256	1,792	3,584
R-BinLWEEnc-II	256	256	84	2^{-32}	256	256	2,048	4,096
R-BinLWEEnc-III	512	256	190	2^{-18}	512	512	4,096	8,192
NTRU ($d_f = 113$) [HHHW09]	401	2048	112	$< 2^{-112}$	401	636	4,401	4,401
NTRU ($d_f = 49$) [HHHW09]	541	2048	112	$< 2^{-112}$	541	858	5,951	5,951
NTRU ($d_f = 38$) [HHHW09]	659	2048	112	$< 2^{-112}$	659	1045	7,249	7,249
R-LWEEnc [GFS+12]	256	7681	106	$\approx 2^{-7}$	256	1,504	3,304	6,608
R-LWEEnc [GFS+12]	512	12289	157	$\approx 2^{-7}$	512	3,062	6,956	13,912

Table 8.1.: Proposed Parameter Sets for R-BinLWEEnc. Hardness Result for LWE taken from [LN13]

are classical reductions [BLP+13] and worst case results for LWE with uniformly distributed error [MP13], for LWE instances with leaky secret [GKPV10], and for Ring-LWE [LPR10].

Two of the above reductions can in principle be applied for LWE with binary error or secret. Goldwasser et al. [GKPV10] gave a reduction from LWE with binary (and possibly leaky) secret to LWE with uniform secret. However, these results are solely valid for LWE with Gaussian error and therefore not applicable to Ring-BLWE. Micciancio and Peikert [MP13] showed the worst-case hardness of LWE with uniform error distribution if the number of samples is restricted. Unfortunately, their worst case result for binary errors requires a strong restriction on the number of samples and furthermore does not transfer to the ring setting.

Consequently, R-BinLWEEnc is not worst-case secure, but only based on the average-case hardness of Ring-BLWE. Other examples of the common practice to base the security on average-case problems are the signature schemes by Lyubashevsky et al. [GLP12, DDLL13] and the NTRU encryption scheme [HPS98]. Likewise, the parameter sets proposed by Lindner and Peikert for their encryption scheme [LP11] have not been shown to provide worst-case security.

Attacks on Ring-BLWE

Despite several algorithms dedicated to problems in ideal lattices [LS14, EHL14], there is still no breakthrough result that can break schemes based on ideal lattices considerably faster. In the lattice challenges [LRBN], the current record for solving the shortest integer solution problem in standard lattices is dimension 140, while the record for SVP in ideal lattices is dimension 128. In particular it has not been shown that any known attack considerably profits from the additional structure of Ring-LWE. Hence, according to current knowledge, Ring-LWE is assumed to be not easier than LWE.

The hardness of BLWE was recently studied by Buchmann et al. [BGPW16]. The authors present a new attack, and compare it with existing approaches like the distinguishing attack [MR09, LP11], the embedding approach [Kan87], the decoding attack [LP11], and the meet-in-the-middle attack [APS15]. Additionally, they argue that algebraic attacks like the Arora-Ge algorithm [AG11, ACF+14] or the BKW approach [BKW03, ACF+14] can not be applied to Ring-BLWE instances with few samples.

Applying the methods presented in [BGPW16] leads to the hardness estimations given in Table 8.1. For all instances, the hybrid approach [BGPW16] performed best.

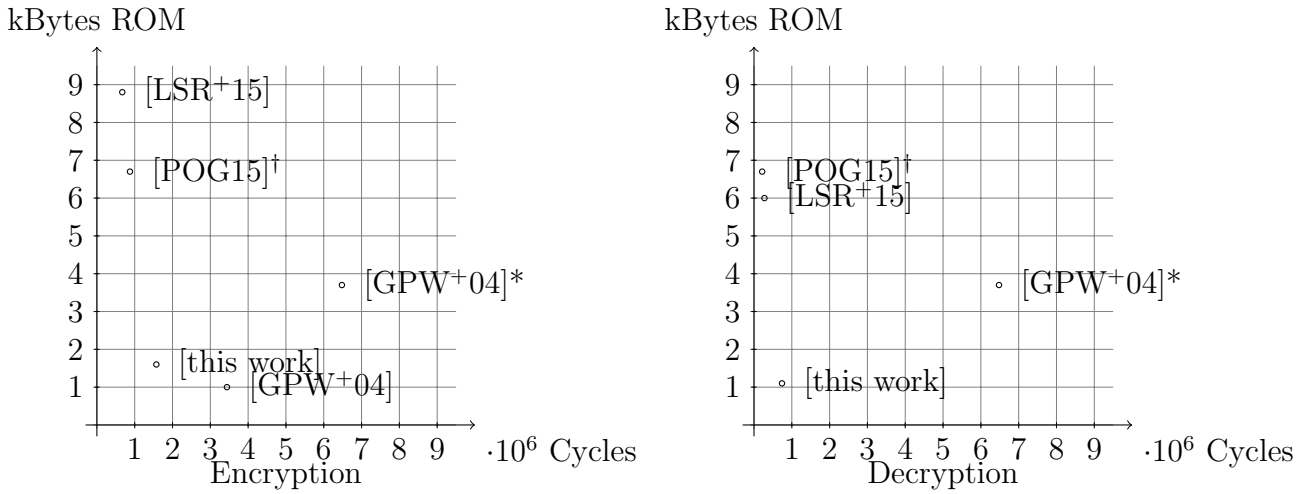


Figure 8.2.: Comparison of memory consumption and speed of AVR implementations. Implementation [GPW$^+$04]* is a point multiplication on an elliptic curve. Implementation [POG15]† gives no separate data about memory consumption for encryption and decryption.

8.3. Instantiation and Results

To evaluate the performance of R-BinLWEEnc on typical IoT devices, Buchmann et al. implemented the scheme on an 8-bit Atmel AVR ATXmega128A1 microcontroller using AVR-GCC 4.7 with optimization flag `-Os` and on the ARM Cortex-M0 using armcc V5.06 with optimization flag `-O3`.

Results for the C implementation are given in Table 5.1. Our ARM implementation includes assembly optimization of the multiplication. For the AVR implementation, an assembly implementation did not provide significant savings due to the very simple nature of the polynomial multiplication algorithm and the straightforward mapping of polynomial coefficients to the `uint8_t` data type. Due to the higher level of optimization, as described in the work by Buchman et al., the ARM implementation runs faster than the AVR implementation. Storing two key coefficients in one 32-bit data word instead of one coefficient in an 8-bit data word also doubles the memory requirement for the key storage.

The trade-off between memory consumption and speed for various ATXmega implementations is given in Figure 8.2. The points correspond to the implementation by Buchmann et al., two implementations of an instantiation of Ring-LWE with Gaussian error ($n = 256, q = 7681$, [LSR$^+$15, POG15]), an RSA-1024 implementation and an elliptic-curve point-multiplication [GPW$^+$04]. Table 5.1 also contains an implementation of a code-based scheme [HvMG13], which was omitted in the figure since it is several orders of magnitude less space efficient than the other imple-

mentations. For the same reason, we omitted the RSA decryption implementation. Figure 8.2 highlights the extremely small memory footprint of our implementation.

In [LSR+15] two different optimization goals are given. The high-speed implementation outperforms the new implementation on the same platform by a factor of 2.5. But this comes to no surprise since our main design goal is a low memory footprint and their high-speed implementation includes large precomputed tables for the number-theoretic transform. Their memory-efficient implementation performs comparable the new implementation (1,532,823 / 673,489 cycles for encryption / decryption) but is still much larger than our implementation (8,5 / 6,0 kBytes for encryption/decryption). The implementation of [POG15] is 1.8 times faster for encryption than the new one (and 3.4 times faster for decryption) but also applies the number-theoretic transform with precomputed twiddle factors and therefore requires much more memory.

Translating the implementation results for RSA and ECC given in [GPW+04] to cycle counts, it turns out that an ECC secp160r1 operation requires 6.5 million cycles. RSA-1024 encryption with public key $e = 2^{16} + 1$ is the only implementation that outperforms ours in terms of memory consumption, but instead only provides two times slower encryption and nearly 120 times slower decryption with Chinese Remainder Theorem (CRT). Compared to an implementation of a code-based scheme [HvMG13] Buchmann et al. also achieve a much better performance and require less flash memory.

9 A Double Secure Encryption Scheme

Many modern public-key cryptographic schemes come with a security proof that ensures that the ability to break the scheme implies the ability to solve an instance of an underlying problem. This can be seen as a way to define the attack surface for the scheme: in order to break the scheme, it is necessary to solve the problem instance. Unfortunately, typically the opposite is true, too: breaking the assumption is enough to break the scheme. This is clearly unfavorable, since we can not prove that any problem used for public-key cryptography is in fact hard.

On the one hand, this is particularly problematic for rather young security assumptions like LWE. In fact, the missing trust in the hardness of LWE is one of the main problems that prohibited a wide-spread application of LWE-based schemes so far. On the other hand, the hardness of well-established problems like factoring or discrete logarithms is threatened by the possibility of quantum computers. Until know, everybody had to select either a rather young (but potentially post-quantum) hardness assumption, or a well-established, but not quantum-resistant one.

In this chapter, we introduce a new way do address this problem: we show that it is possible to combine the LWE problem with a number theoretic problem (the representation problem, RP) to obtain a new computational problem called Learning With Errors in the Exponent (LWEE). We proof that this problem remains hard as long as either LWE *or* RP is hard. Additionally, we show the usability of the new assumption by constructing a public-key encryption scheme whose security can be reduced to the hardness of LWEE.

The new scheme allows a very flexible parameter selection. We show that it is possible to select parameters such that the scheme is based on the number-theoretic assumption only (the classical way), on LWE only (the post-quantum way), or on bot assumptions (the double-secure way). While the parameter selection for the post-quantum instantiation can be done similar to the selection in Chapter 8, the double-secure instantiation requires a different approach. This is due to the fact the scheme needs an exponential gap between the error size and the modulus. Such LWE instances are typically not used for standard LWE encryption, but are very common in advanced schemes like fully homomorphic encryption. Consequently,

we use techniques proposed to instantiate one such scheme to identify appropriate parameters for the double-secure instantiation.

The second difference between this chapter and Chapter 8 is that the results here are mainly of theoretical interest. We take this into consideration by giving asymptotic parameters instead of focusing parameters optimized for efficient implementation.

Organization Section 9.1 introduces the the representation problem, the second problem our scheme will be based on. Afterwards, Section 9.2 defines the new LWEE assumption and proves reductions from LWE and RP. The remaining parts of the chapter deal with the new encryption scheme. First, Section 9.3 presents the scheme and gives the reduction from LWEE. Second, Section 9.4 explains the parameter selection, including the hardness estimation used for the LWE instances.

This chapter is based on a publication on a publication on ICISC 2015 [DGG15].

9.1. The Representation Problem

In this section, we introduce the well-established search representation problem, and give the first decisional variant. We also introduced the rank hiding problem, that is used to show that the rank hiding problem is hard, even when several samples are given.

9.1.1. Definition and Properties

The representation problem in a group $\mathbb{G}$ assumes that given l random group elements $g_1, \ldots, g_l \in \mathbb{G}$ and $h \in \mathbb{G}$ it is hard to find a representation $\mathbf{x} \in \mathbb{Z}_q^\ell$ such that $h = \prod_{i=1}^{\ell} g_i^{x_i}$ holds. Brands shows an electronic cash system based on the problem. Recently, the assumption was extensively applied to show leakage resiliency [KV09, ADVW13, DV14].

We now state a more general version of the search representation problem where vector $\mathbf{x} \xleftarrow{\$} \chi^\ell$ is sampled from a distribution χ with (at least) min-entropy and where an adversary is given $m \geq 1$ samples instead of a single one.

Definition 5 (Search Representation Problem)**.** *Let χ be a distribution over $\mathbb{Z}_q$, and ℓ, m be integers. Sample $\mathbf{M} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$ and $\mathbf{x} \xleftarrow{\$} \chi^\ell$. The Search Representation Problem ($\mathsf{SRP}_{\mathbb{G},\chi,\ell,m}$) is (t,ϵ)-hard if any algorithm $\mathcal{A}$, running in time t, upon input $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{Mx}})$, outputs $\mathbf{x}' \in \mathbb{Z}_q^\ell$ such that $g^{\mathbf{Mx}'} = g^{\mathbf{Mx}}$ with probability at most ϵ. If χ is the uniform distribution, we sometimes skip χ in the index and say that $\mathsf{SRP}_{\mathbb{G},\ell,m}$ is (t,ϵ)-hard.*

Brands proves the equivalence of the representation problem and the discrete logarithm problem for uniform χ and $m = 1$. It is easy to verify that the reduction holds for every distribution for which the discrete logarithm problem holds.

To establish relations to the learning with errors in the exponent problem (cf. Section 9.2.2), we need a decisional variant of the representation problem. To our surprise, the decisional version has not been defined before, although the assumption is a natural generalization of the decisional Diffie-Hellman problem to ℓ-tuples (similar in spirit as the ℓ-linear problem in $\mathbb{G}$ [Sha07]). Given ℓ random group elements $g_1, \ldots, g_\ell \in \mathbb{G}$ together with $h \in \mathbb{G}$ and $g^{x_1}, \ldots, g^{x_\ell} \in \mathbb{G}$ where $x_1, \ldots, x_\ell \xleftarrow{\$} \mathbb{Z}_q^*$, it is hard to decide if $h = \prod_{i=1}^{\ell} g_i^{x_i}$ or h is a random group element in $\mathbb{G}$. Our definition below generalizes this problem to the case, where $m \geq 1$ samples are given to an adversary and $x_1, \ldots, x_\ell$ are sampled from any min-entropy distribution χ.

Definition 6 (Decisional Representation Problem)**.** *Let χ be a distribution over $\mathbb{Z}_q^*$, and ℓ, m be integers. Sample $\mathbf{M} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, $\mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^m$, and $\mathbf{x} \xleftarrow{\$} \chi^\ell$. The Decisional Representation ($\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$) problem is (t,ϵ)-hard if*

$$(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{Mx}}) \approx_{(t,\epsilon)} (g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{h}}) \; .$$

If χ is the uniform distribution over $\mathbb{Z}_q^*$, we say $\mathsf{DRP}_{\mathbb{G},\ell,m}$ is (t,ϵ)-hard.

Remark 2. *$\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ can be stated in the framework of the Matrix-DDH assumption recently introduced by Escala et al. [EHK+13] and thus we put another class of hardness problems to the arsenal of their expressive framework.*

We now give evidence that the family of $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ problems is a class of progressively harder problems (with increasing ℓ).

Proposition 1. *If $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ is (t,ϵ)-hard, then for any $\ell, m \geq 1$ with $t' \approx t$ and distribution χ with min-entropy $\mathsf{DRP}_{\mathbb{G},\chi,\ell+1,m}$ is (t',ϵ)-hard.*

Proof. Suppose there exists an adversary $\mathcal{A}$ which solves the $\mathsf{DRP}_{\mathbb{G},\chi,\ell+1,m}$ problem in time t with probability ϵ. We show that in this case, there exists an adversary $\mathcal{B}$ with black-box access to $\mathcal{A}$ which solves the $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ problem with probability ϵ.

Adversary $\mathcal{B}$ is given as challenge the tuple $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{h}}) \in \mathbb{G} \times \mathbb{G}^{m\times\ell} \times \mathbb{G}^{\ell} \times \mathbb{G}^m$. She invokes adversary $\mathcal{A}$ with input the group $\mathbb{G}$ and its generator g. Adversary $\mathcal{A}$ expects as challenge a tuple $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{h}}) \in \mathbb{G} \times \mathbb{G}^{m\times\ell+1} \times \mathbb{G}^{\ell+1} \times \mathbb{G}^m$. To this end, $\mathcal{B}$ samples $x_{\ell+1}$ according distribution χ, and $\mathbf{a} = (a_1, \ldots, a_m)$ uniformly from $\mathbb{Z}_q^m$. Adversary $\mathcal{B}$ provides $\mathcal{A}$ with the challenge $(g, g^{\mathbf{M}'}, g^{\mathbf{x}'}, g^{\mathbf{h}'})$ where $g^{\mathbf{M}'} = (g^{\mathbf{M}'}, g^{\mathbf{a}})$, $g^{\mathbf{x}'} = (g^{\mathbf{x}}, g^{x_{\ell+1}})$, and $g^{h_i'} = g^{h_i} \cdot g^{a_i x_{\ell+1}}$ for $i \in [m]$. Note that $g^{x_{\ell+1}}$ is distributed as expected as we choose $x_{\ell+1} \xleftarrow{\$} \chi$. Moreover, $g^{\mathbf{a}}$ is uniformly distributed in $\mathbb{G}^m$. If the $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ tuples are such that $g^h = \prod_{i=1}^{\ell} g_i^{x_i}$, then $g^{h'}$ in the $\mathsf{DRP}_{\mathbb{G},\chi,\ell+1,m}$

distribution is computed correctly. This follows from the fact that for all $i \in [m]$ we have

$$g^{\mathbf{h}'} = g^{\mathbf{h}} \cdot (g^{\mathbf{a}})^{x_{\ell+1}} = g^{\mathbf{M}\mathbf{x}} \cdot (g^{\mathbf{a}})^{x_{\ell+1}} = g^{\mathbf{M}'\mathbf{x}'}$$

given $g^{\mathbf{h}} = g^{\mathbf{M}\mathbf{x}}$. In case $g^{\mathbf{h}}$ is a random group element, so is $g^{\mathbf{h}'}$, since $\mathbf{a}, x_{\ell+1}$ are sampled independently of h. Hence, $\mathcal{B}$ outputs in her game what $\mathcal{A}$ guesses, and wins with $\mathcal{A}$'s advantage ϵ. The running time of $\mathcal{B}$ is essentially the same as $\mathcal{A}$ merely adding the time to sample $O(m)$ uniform group elements. □

Remark 3. *$\mathsf{DRP}_{\mathbb{G},\chi,1,1}$-problem with χ being the uniform distribution over $\mathbb{Z}_q$ coincides with the decisional Diffie-Hellman (*DDH*) problem. Hence, we obtain the corollary that for uniform distributions χ, the decisional Diffie-Hellman problem implies the representation problem $\mathsf{DRP}_{\mathbb{G},\chi,\ell,1}$ for $\ell \geq 1$. In fact, Proposition 1 suggests a stronger argument. Assuming the decisional Diffie-Hellman problem holds for well-spread and min-entropy distributions χ, then the $\mathsf{DRP}_{\mathbb{G},\chi,\ell,1}$ holds for χ and $\ell \geq 1$.*

While Proposition 1 shows that the DRP problem progressively increases with ℓ, the following proposition states that the problem remains hard with increasing number of samples m. More precisely, we show that $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m+1}$ is hard as long as $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ and the Rank Hiding problem $\mathsf{RH}_{\mathbb{G},m,m+1,m+1,2\ell+1}$ (cf. Definition 7) is hard. RH was introduced by Naor and Segev [NS09] (and later extended by Agrawal et al. [ADVW13]).[1] and proven to be equivalent to the $\mathsf{DDH}_{\mathbb{G},\chi}$ assumption for groups of prime order and uniform χ [NS09].

Definition 7 (Rank Hiding)**.** *Let $\mathbb{G}$ be a group of order q with generator g, and $i, j, n, m \in \mathbb{N}$ satisfying $i \neq j$ and $i, j \geq 1$. The Rank Hiding problem ($\mathsf{RH}_{\mathbb{G},i,j,m,n}$) is (t, ϵ)-hard if*

$$\{(\mathbb{G}, q, g, g^{\mathbf{M}}) : \mathbf{M} \xleftarrow{\$} \mathsf{Rk}_i(\mathbb{Z}_q^{m\times n})\} \approx_{(t,\epsilon)} \{(\mathbb{G}, q, g, g^{\mathbf{M}}) : \mathbf{M} \xleftarrow{\$} \mathsf{Rk}_j(\mathbb{Z}_q^{m\times n})\}$$

where $\mathsf{Rk}_k(\mathbb{Z}_q^{m\times n})$ returns an $m \times n$ matrix uniformly random from $\mathbb{Z}_q^{n\times m}$ with rank $k \leq \min(n, m)$.

Proposition 2. *If $\mathsf{RH}_{\mathbb{G},m,m+1,m+1,2\ell+1}$ is (t, ϵ)-hard and $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ is (t', ϵ')-hard in a cyclic group $\mathbb{G}$ of order q, then for any distribution χ_e and any $m > 0$ with $t' \approx t$ and $\epsilon'' \leq (1-\epsilon)^{-1}\epsilon'$ $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m+1}$ is (t, ϵ'')-hard.*

Proof. We prove by contradiction. We assume that $\mathsf{RH}_{\mathbb{G},m,m+1,m+1,2\ell+1}$ is (t, ϵ)-hard and $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ is (t', ϵ')-hard. However, we assume that there is an algorithm $\mathcal{A}$ which solves $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m+1}$ in time t with probability $\epsilon'' > (1-\epsilon)^{-1}\epsilon'$.

[1] The assumption was first introduced by Boneh et al. [BHHO08] under the Matrix DDH assumption.

We then build an algorithm $\mathcal{B}$ with black-box access to $\mathcal{A}$ which solves the $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ problem in time $t' \approx t$ with probability larger than ϵ' as follows. The algorithm $\mathcal{B}$ is given a DRP instance $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{h}})$ for uniform matrix $\mathbf{M} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$ and has to decide whether $g^{\mathbf{h}}$ equals $g^{\mathbf{Mx}}$ or was chosen uniformly from $\mathbb{G}^m$. Algorithm $\mathcal{B}$ now prepares a DRP instance for $\mathcal{A}$ by adding a row to the matrix $g^{\mathbf{M}}$ and vector $g^{\mathbf{h}}$ as follows. It chooses a random index $0 \leq i \leq m$ and samples a random coefficient vector $\mathbf{y} \in \mathbb{Z}_q^m$. Let $\mathbf{u} = g^{\mathbf{y}^\top\mathbf{M}} = g^{y_1\mathbf{m}_1} \cdot \ldots \cdot g^{y_m\mathbf{m}_m}$ and $v = g^{\langle\mathbf{h}_i,\mathbf{y}\rangle}$. Create the matrix $g^{\mathbf{M}'} \in \mathbb{G}^{(m+1)\times\ell}$ by inserting $\mathbf{u}$ before the ith column of $g^{\mathbf{M}}$, and $\mathbf{h}' \in \mathbb{G}^{m+1}$ by inserting $\mathbf{v}$ before the ith entry of $\mathbf{h}$. Now, $\mathcal{B}$ invokes $\mathcal{A}$ upon input $(g, g^{\mathbf{M}'}, g^{\mathbf{x}}, g^{\mathbf{h}'})$.

At this point, we stress that $\mathcal{A}$ will accept the input and work properly even if $(g, g^{\mathbf{M}'}, g^{\mathbf{x}}, g^{\mathbf{h}'})$ is of different rank. In fact, an honestly generated DRP instance for the $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m+1}$ problem will have a rank $\min(\ell, m+1)$ matrix (with overwhelming probability), while our input matrix has rank $\min(\ell, m)$ (with overwhelming probability). Since by assumption there is no algorithm that can distinguish those two inputs (matrices) in time t with a probability greater than ϵ, algorithm $\mathcal{A}$, which also runs in time t, must work for the given input with probability greater than $(1-\epsilon)$. Algorithm $\mathcal{A}$ returns a guess $b \in \{0,1\}$ for its challenge which in turn constitutes the guess of $\mathcal{B}$ for its challenge instance $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{h}})$. Since $\mathcal{A}$ successfully wins its challenge in time t with probability ϵ'', we have constructed an algorithm $\mathcal{B}$ which breaks $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ in time $t' \approx t$ with probability $(1-\epsilon)\epsilon'' > \epsilon'$. This leads to a contradiction to $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m}$ being (t', ϵ')-hard. Hence, $\mathsf{DRP}_{\mathbb{G},\chi,\ell,m+1}$ must be $(t', (1-\epsilon)^{-1}\epsilon')$-hard.

□

9.2. Learning with Errors in the Exponent

This section deals with the new problem LWEE and its hardness. After the formal definition of LWEE, we give reductions from RP and LWE. The last part gives heuristic evidence that LWEE is computationally harder than each of the underlying problems.

9.2.1. Definition

For self-containment, the assumption is stated both as a search and decision problem over a group $\mathbb{G}$ of order q, and exponents sampled from distributions χ_e, χ_s over $\mathbb{Z}$. We demonstrate the versatility and general utility of the decisional version in Section 9.3.

Definition 8 (Learning with Errors in the Exponent)**.** *Let $\mathbb{G}$ be a group of order q where g is a generator of $\mathbb{G}$. Let n, m, q be integers and χ_e, χ_s be distributions*

over $\mathbb{Z}$. *For any fixed vector* $\mathbf{s} \in \mathbb{Z}_q^n$, *define the* LWEE *distribution* $L^{\mathsf{LWEE}}_{\mathbb{G},n,q,\chi_e}$ *to be the distribution over* $\mathbb{G}^n \times \mathbb{G}$ *obtained such that one first draws vector* $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$ *uniformly,* $e \xleftarrow{\$} \chi_e$ *and returns* $(g^{\mathbf{a}}, g^b) \in \mathbb{G}^n \times \mathbb{G}$ *with* $b = \langle \mathbf{a}, \mathbf{s} \rangle + e$. *Let* $(g^{\mathbf{a}_i}, g^{b_i})$ *be samples from* $L^{\mathsf{LWEE}}_{\mathbb{G},n,q,\chi_e}$ *and* c_i *be uniformly sampled from* $\mathbb{Z}_q^*$ *for* $0 \leq i < m = poly(\kappa)$.

- *The Search Learning With Errors in the Exponent* ($\mathsf{SLWEE}_{\mathbb{G},n,m,q,\chi_e}(\chi_s)$) *problem is* (t, ϵ)*-hard if any algorithm* $\mathcal{A}$, *running in time* t, *upon input* $(g^{\mathbf{a}_i}, g^{b_i})_{i \in [m]}$, *outputs* $\mathbf{s}$ *with probability at most* ϵ.

- *The Decision Learning With Errors in the Exponent* ($\mathsf{DLWEE}_{\mathbb{G},n,m,q,\chi_e}(\chi_s)$) *problem is* (t, ϵ)*-hard if* $(g^{\mathbf{a}_i}, g^{b_i})_{i \in [m]} \approx_{(t,\epsilon)} (g^{\mathbf{a}_i}, g^{c_i})_{i \in [m]}$ *for a random secret* $\mathbf{s} \xleftarrow{\$} \chi_s^n$. *If* χ_s *is the uniform distribution over* $\mathbb{Z}_q$, *we write* $\mathsf{DLWEE}_{\mathbb{G},n,m,q,\chi_e}$.

We let $\mathsf{Adv}^{\mathsf{DLWEE/SLWEE}}_{\mathbb{G},n,m,q,\chi_e,\chi_s}(t)$ *denote a bound on the value* ϵ *for which the decisional/search* LWEE *problem is* (t, ϵ)*-hard.*

One may interpret learning with errors in the exponent in two ways. One way is to implant an error term from a distribution χ_e into the Diffie-Hellman exponent. Another way to look at LWEE is as compressing an LWE instance within some group $\mathbb{G}$ of order q.

9.2.2. Relations to Group and Lattice Problems

We connect the representation and learning with errors problem to learning with errors in the exponent. The essence is that there exist tight reductions from the search (resp. decision) learning with errors in the exponent problem to either the search (resp. decision) representation problem and the search (resp. decision) learning with errors problem. This has several interesting property preserving implications. As a corollary we infer that for appropriate parameter choices LWEE preserves the *hardness* and *robustness* properties of the representation and/or learning with errors problem. Essentially then LWEE boils down to the security of either of the two underlying problems. This way, the cryptosystem can be instantiated to leverage leakage resistance and post-quantum hardness thanks LWE [GKPV10, Reg05]. On the flip side, the cryptosystem may offer short instance sizes through the underlying RP problem (when instantiated on elliptic curves). Of particular interest for many emerging applications is the partnering of the two hardness assumptions. One may choose parameters such that both RP and LWE hold. We call the case *double-hard*, which appeals to provide in some sense hedged security.

Following four propositions summarize our main results.

Proposition 3. *If* $\mathsf{SRP}_{\mathbb{G},\chi_s,\ell,m}$ *is* (t, ϵ)*-hard in a cyclic group* $\mathbb{G}$ *of order* q, *then for any distribution* χ_e *and any number of samples* $m > 0$ $\mathsf{SLWEE}_{\mathbb{G},\ell,m,q,\chi_e}(\chi_s)$ *is* (t', ϵ)*-hard with* $t' \approx t$.

Proof. Suppose there exists an adversary $\mathcal{A}$ which solves the $\mathsf{SLWEE}_{\mathbb{G},\ell,m,q,\chi_e}(\chi_s)$ problem in time t with probability ϵ. We show that in this case, there exists an adversary $\mathcal{B}$ with black-box access to $\mathcal{A}$ which solves the $\mathsf{SRP}_{\mathbb{G},\chi_s,\ell,m}$ problem in time $\approx t$ with probability ϵ.

Adversary $\mathcal{B}$ is given as challenge a SRP instance $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{Mx}})$ and is asked for a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ such that $g^{\mathbf{Mx}'} = g^{\mathbf{Mx}}$. Adversary $\mathcal{B}$ invokes $\mathcal{A}$ with input the group $\mathbb{G}$ and its generator g. Whenever $\mathcal{A}$ asks for the i-th sample from the $L^{\mathsf{LWEE}}_{\mathbb{G},\ell,m,\chi_e}$ distribution, adversary $\mathcal{B}$ returns the i-th row of $g^{\mathbf{M}}$ and the i-th element of $g^{\mathbf{Mx}}$ with some noise $e_i \leftarrow \chi_e$, i.e., $g^{\mathbf{a}_i} := g^{\mathbf{M}[i]}$ and $g^{b_i} := g^{(\mathbf{Mx})_i} \cdot g^{e_i}$. Note that $(g^{\mathbf{a}_i}, g^{b_i})$ as such corresponds to the $L^{\mathsf{LWEE}}_{\mathbb{G},\ell,m,\chi_e}$ distribution. The vector $g^{\mathbf{a}_i}$ is uniformly distributed as the input $g^{\mathbf{M}}$ for SRP is uniformly distributed. Moreover, we have

$$g^{b_i} = g^{(\mathbf{Mx})_i} \cdot g^{e_i} = g^{\langle \mathbf{M}[i], \mathbf{x} \rangle + e_i} .$$

Note that $\mathcal{B}$ can provide $\mathcal{A}$ enough samples since both algorithms get m samples from their respective distributions.

Eventually, adversary $\mathcal{A}$ will output an element $\mathbf{s} \in \mathbb{Z}_q^\ell$ such that for all $i \in \{1, \ldots, m\}$ it holds $g^{\langle \mathbf{a}_i, \mathbf{s} \rangle + e'_i} = g^{b_i}$ where $e'_i \xleftarrow{\$} \chi_e$. Now, since there can exist only a single vector $\mathbf{s}$ which can fulfill the equation $g^{\langle \mathbf{a}_i, \mathbf{s} \rangle + e'_i} = g^{b_i}$ for errors $e' \xleftarrow{\$} \chi_e$, we must have $\mathbf{s} = \mathbf{x} = (x_1, \ldots, x_\ell)$. Hence, $\mathcal{B}$ outputs $\mathbf{s}$ as the solution vector for her instance.

The running time of $\mathcal{B}$ is almost identical to $\mathcal{A}$, and the success probability is equal, too. The proposition follows accordingly. □

Proposition 4. *If* $\mathsf{SLWE}_{n,m,q,\chi_e}(\chi_s)$ *is* (t, ϵ)*-hard, then for any cyclic group* $\mathbb{G}$ *of order* q *with known (or efficiently computable) generator* $\mathsf{SLWEE}_{\mathbb{G},n,m,q,\chi_e}(\chi_s)$ *is* (t', ϵ)*-hard with* $t' \approx t$*.*

Proof. Suppose there exists an adversary $\mathcal{A}$ which solves the $\mathsf{SLWEE}_{\mathbb{G},n,m,\chi_e}(\chi_s)$ problem in time t with probability ϵ. We show that in this case, there exists an adversary $\mathcal{B}$ with black-box access to $\mathcal{A}$ which solves the $\mathsf{SLWE}_{n,m,q,\chi_e}(\chi_s)$ problem in time $\approx t$ with probability ϵ.

Adversary $\mathcal{B}$ is allowed to ask for samples $(\mathbf{a}_i, b_i)$ which are distributed either according to the $L^{\mathsf{LWE}}_{n,m,q,\chi_e}$ distribution or distributed uniformly in $(\mathbb{G}^n \times \mathbb{G})$. Adversary $\mathcal{B}$ invokes adversary $\mathcal{A}$ with input $\mathbb{G}$ (the group of order q) and samples a random generator g for that group. When $\mathcal{A}$ asks for i-th sample $(g^{\mathbf{a}_i}, g^{b_i})$, $\mathcal{B}$ asks for samples $(\mathbf{a}_i, b_i)$ in his own game and returns to $\mathcal{A}$ the tuple $(g^{\mathbf{a}_i}, g^{b_i})$.

Eventually, $\mathcal{A}$ outputs the secret $\mathbf{s}$, which $\mathcal{B}$ forwards to his own game as output. Time complexity of $\mathcal{B}$ is the time required by $\mathcal{A}$ plus taking exponentiations, which is a negligible cost. □

Proposition 5. *If* $\mathsf{DRP}_{\mathbb{G},\chi_s,\ell,m}$ *is* (t, ϵ)*-hard in a cyclic group* $\mathbb{G}$ *of order* q*, then for any distribution* χ_e *and any number of samples* $m > 0$ $\mathsf{DLWEE}_{\mathbb{G},\ell,m,\chi_e}(\chi_s)$ *is* (t', ϵ)*-hard with* $t' \approx t$*.*

Proof. Suppose there exists an adversary $\mathcal{A}$ which solves the $\mathsf{DLWEE}_{\mathbb{G},\ell,m,\chi_e}(\chi_s)$ problem in time t with probability ϵ. We show that in this case, there exists an adversary $\mathcal{B}$ with black-box access to $\mathcal{A}$ which solves the $\mathsf{DRP}_{\mathbb{G},\chi_s,\ell,m}$ problem in time $\approx t$ with probability ϵ.

Adversary $\mathcal{B}$ is given as challenge a DRP instance $(g, g^{\mathbf{M}}, g^{\mathbf{x}}, g^{\mathbf{h}})$ and has to decide whether $\mathbf{h}$ equals $\mathbf{Mx}$ or was chosen uniformly at random from $\mathbb{Z}_q^m$. Adversary $\mathcal{B}$ invokes $\mathcal{A}$ with input the group $\mathbb{G}$. Whenever $\mathcal{A}$ asks for the i-th sample from the $L^{\mathsf{LWEE}}_{\mathbb{G},\ell,m,\chi_e}$ distribution, adversary $\mathcal{B}$ returns the i-th row of $g^{\mathbf{M}}$ and the i-th element of $g^{\mathbf{h}}$ with some noise $e_i \leftarrow \chi_e$, i.e., $g^{\mathbf{a}_i} := g^{\mathbf{M}[i]}$ and $g^{b_i} := g^{h_i} \cdot g^{e_i}$. Note that $(g^{\mathbf{a}_i}, g^{b_i})$ as such corresponds to the $L^{\mathsf{LWEE}}_{\mathbb{G},\ell,m,\chi_e}$ distribution. The vector $g^{\mathbf{a}_i}$ is uniformly distributed as the input $g^{\mathbf{M}}$ for DRP is uniformly distributed. Moreover, we have

$$g^{b_i} = g^{h_i} \cdot g^{e_i} = g^{(Mx)_i} \cdot g^{e_i} = g^{\langle \mathbf{M}[i], \mathbf{x} \rangle + e_i}$$

if $g^{\mathbf{h}} = g^{\mathbf{Mx}}$. Otherwise, $g^{\mathbf{b}_i}$ is distributed uniformly in $\mathbb{G}$ since $\mathbf{h}$ is. Note that $\mathcal{B}$ can provide $\mathcal{A}$ enough samples since both algorithms get m samples from their respective distributions.

Hence, when adversary $\mathcal{A}$ outputs a bit d, adversary $\mathcal{B}$ outputs d in her decisional representation problem. If $\mathcal{A}$ guessed correctly, so does $\mathcal{B}$. The running time of $\mathcal{B}$ is almost identical to $\mathcal{A}$, and the success probability is equal, too. The proposition follows accordingly.

□

Proposition 6. *If* $\mathsf{DLWE}_{n,m,q,\chi_e}(\chi_s)$ *is* (t,ϵ)*-hard, then for any cyclic group* $\mathbb{G}$ *of order* q *with known (or efficiently computable) generator* $\mathsf{DLWEE}_{\mathbb{G},n,m,\chi_e}(\chi_s)$ *is* (t',ϵ)*-hard with* $t' \approx t$*.*

Proof. Suppose there exists an adversary $\mathcal{A}$ which solves the $\mathsf{DLWEE}_{\mathbb{G},n,m,\chi_e}(\chi_s)$ problem in time t with probability ϵ. We show that in this case, there exists an adversary $\mathcal{B}$ with black-box access to $\mathcal{A}$ which solves the $\mathsf{LWE}_{n,m,q,\chi_e}(\chi_s)$ problem in time $\approx t$ with probability ϵ.

Adversary $\mathcal{B}$ is allowed to ask for samples $(\mathbf{a}_i, b_i)$ which are distributed either according to the $L^{\mathsf{LWE}}_{n,m,q,\chi_e}$ distribution or distributed uniformly in $(\mathbb{G}^n \times \mathbb{G})$. Adversary $\mathcal{B}$ invokes adversary $\mathcal{A}$ with input $\mathbb{G}$ (the group of order q) and samples a random generator g for that group. When $\mathcal{A}$ asks for i-th sample $(g^{\mathbf{a}_i}, g^{b_i})$, $\mathcal{B}$ asks for samples $(\mathbf{a}_i, b_i)$ in his own game and returns to $\mathcal{A}$ the tuple $(g^{\mathbf{a}_i}, g^{b_i})$.

Eventually, $\mathcal{A}$ outputs a bit b, which $\mathcal{B}$ forwards to his own game as output. It is easy to verify that the samples $(\mathbf{a}_i, b_i)$ are distributed according to $L^{\mathsf{LWE}}_{n,m,q,\chi_e}$ if and only if the samples $(g^{\mathbf{a}_i}, g^{b_i})$ are distributed according to $L^{\mathsf{LWEE}}_{\mathbb{G},n,m,\chi_e}$. Hence, $\mathcal{B}$ wins whenever $\mathcal{A}$ does while having approximately the same running time $\approx t$. □

9.2.3. On the Generic Hardness of LWEE

With Proposition 3-6 in our toolbox we conjecture LWEE to be harder than either of the underlying RP or LWE problems. The argument is heuristic and based on what is known about the hardness of each intractability problem.

Fix parameters such that RP and LWE problem instances give κ bits security. The only obvious known approach today to solve the LWEE instance is to first compute the discrete logarithm of samples $(g^{\mathbf{a}_i}, g^{b_i})$ and then solve the LWE problem for samples $(\mathbf{a}_i, b_i)$. Note that an adversary must solve n^2+n many discrete logarithms because the secret vector $\mathbf{s}$ is information-theoretically hidden, if less than n samples of LWE are known. Solving $N := n^2+n$ discrete logarithms in generic groups of order q takes time $\sqrt{2Nq}$ while computing a single discrete logarithm takes time $\sqrt{\pi q/2}$ [KS01, HMCD04].[2] In fact, this bound is proven to be optimal in the generic group model [Yun14]. Note, parameters for LWEE are chosen such that computing a single discrete logarithm takes time 2^κ. Hence, in order to solve the LWEE instance for $N = \mathcal{O}(\kappa^2)$, one requires time $\frac{2}{\sqrt{\pi}}\sqrt{N} \cdot 2^\kappa + 2^\kappa > 2^{\kappa+2\log(\kappa)}$. This shows that generically the concrete instance of LWEE is logarithmically harder in the security parameter κ.

9.3. Public Key Encryption from LWEE

This section introduces the new encryption scheme based on LWEE. We begin with some intuition on the ideas that are the foundation of the scheme, before we give the formal definition of our scheme. Afterwards, we show correctness and security of the scheme.

9.3.1. The High-Level Idea

The idea behind our scheme is reminiscent of Regev's public-key encryption scheme. In a nutshell, the public key is an LWEE instance $(g^{\mathbf{A}}, g^{\mathbf{As}+\mathbf{x}}) \in \mathbb{G}^{n\times n} \times \mathbb{G}^n$. Similarly to [LPS10, LP11] and as opposed to Regev [Reg05], for efficiency reason we avoid the use of the leftover hash lemma –instead we impose one further LWEE instance– and make use of a square matrix A. Ciphertexts consist of two LWEE instances $C = (\mathbf{c}_0, c_1)$ where $\mathbf{c_0} = g^{\mathbf{Ar}+\mathbf{e}_0}$ encapsulates a random key $\mathbf{r} \in \mathbb{Z}_q^n$ and $c_1 = g^{\langle \mathbf{b},\mathbf{r}\rangle + e_1} \cdot g^{\alpha\mu}$ encrypts the message μ (we discuss the exact value of α below). The tricky part is the decryption algorithm. All known LWE-based encryption schemes require some technique to deal with the noise terms. Otherwise, decryption is prone to err. Regev's technique ensures small error terms. One simply rounds $c_1 - \mathbf{c}_0\mathbf{s}$ to

[2] Solving N-many discrete logarithms is easier than applying N times a DL solver for a single instance.

some reference value c_b indicating the encryption of bit b. While rounding splendidly works on integers, the technique fails in our setting.

Our approach explores a considerably different path. Instead of rounding, we synthesize the pesky error terms. To this end, we adapt the trapdoor technique of Joye and Libert [JL13] and recover partial bits of the discrete logarithm (by making use of the Pohlig-Hellman algorithm [PH78]). The main idea is to tweak the modulus in a smart way. Given composite modulus $N = pq$ with p', q', such that $p = 2^k p' + 1$ and $q = 2^k q' + 1$ are prime, there exists an efficient algorithm for recovering the k least significant bits of the discrete logarithm. We choose the parameters so that the sum of all error terms in the exponent is (with high probability) at most $2^{k-\ell}$. This leads to a "gap" between error bits and those bits covert by the discrete log instance. We plant the message in this gap by shifting it to the $2^{k-\ell}$'s bit, where ℓ is the size of the message we want to decrypt. Hence, we choose $\alpha = 2^{k-\ell}$ in our construction to shift the message bits accordingly. We leave it as an interesting open problem to instantiate the scheme in prime order groups.

9.3.2. Our Construction

The scheme is parameterized by positive integers $n, k, \ell < k$ and Gaussian parameters σ_s, σ_e.

KeyGen: Sample prime numbers p' and q', such that $p = 2^k p' + 1$ and $q = 2^k q' + 1$ are prime. Set $N = pq$ and $M = 2^k p' q'$. Sample $\mathbf{s} \xleftarrow{\$} D^n_{\sigma_s}$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}^{n \times n}_M$ and $\mathbf{x} \xleftarrow{\$} D^n_{\sigma_e}$ and compute $\mathbf{b} = \mathbf{A}^\top \mathbf{s} + \mathbf{x}$. Sample $g \in \mathbb{J}_N \setminus \mathbb{RR}_N$ of order M. The public key consists of $\mathsf{pk} = (g, g^{\mathbf{A}}, g^{\mathbf{b}}, N)$, and the secret key of $\mathsf{sk} = (p, \mathbf{s})$.

Encrypt(pk, μ): To encrypt ℓ bits $\mu \in \{0, 1, \ldots 2^\ell - 1\}$ given public key pk choose $\mathbf{r} \xleftarrow{\$} D^n_{\sigma_s}$, $\mathbf{e}_0 \xleftarrow{\$} D^n_{\sigma_e}$ and $e_1 \xleftarrow{\$} D_{\sigma_e}$. Use $g^{\mathbf{A}}$, $\mathbf{r}$ and $\mathbf{e}_0$ to compute $g^{\mathbf{Ar}+\mathbf{e}_0}$, and $g^{\mathbf{b}}$, $\mathbf{r}$ and e_1 to compute $g^{\langle \mathbf{b}, \mathbf{r} \rangle + e_1}$. The ciphertext is $\mathbf{c}_0, c_1$ with

$$\mathbf{c_0} = g^{\mathbf{Ar}+\mathbf{e}_0}, \quad c_1 = g^{\langle \mathbf{b}, \mathbf{r} \rangle + e_1} \cdot g^{2^{k-\ell}\mu} .$$

Decrypt($\mathsf{sk}, (\mathbf{c}_0, c_1)$): To decrypt the ciphertext $(\mathbf{c}_0, c_1)$ given secret key $\mathsf{sk} = (p, \mathbf{s})$, first compute $g^{\langle \mathbf{s}, \mathbf{Ar}+\mathbf{e}_0 \rangle}$ and then $h = c_1 / g^{\langle \mathbf{s}, \mathbf{Ar}+\mathbf{e}_0 \rangle}$. Run Algorithm 6 to synthesize $v = \log_g(h) \mod 2^k$ and return $\left\lfloor \frac{v}{2^{k-\ell-1}} \right\rceil$.

9.3.3. Correctness

To show correctness of our construction we build upon two facts. First, Algorithm 6 synthesizes the k least significant bits of a discrete logarithm. The algorithm's correctness for a modulus being a multiple of 2^k is proven in [JL13, Section 3.2]. Second, noise in the exponent does not overlap with the message. To this end, we bound the size of the noise with following lemma.

Algorithm 6:

Input: Generator g of a group with order $p-1=2^k p'$, p and k
Output: k least significant bits of $\log_g(h)$
begin
  $a=0, B=1$;
  for $i \in \{1,\ldots,k\}$ **do**
    $z \leftarrow \mathbb{L}(h,p)_{2^i} \mod p$;
    $t \leftarrow \mathbb{L}(g,p)^a_{2^i} \mod p$;
    if $z \neq t$ **then**
      $a \leftarrow a+B$;
    end
    $B \leftarrow 2B$;
  end
  return a
end

Lemma 6 (adapted from [LP11, Lemma 3.1])**.** *Let c, T be positive integers such that*

$$\sigma_s \cdot \sigma_e \leq \frac{\pi}{c}\frac{T}{\sqrt{n\ln(2/\delta)}} \quad \text{and} \quad \left(c\cdot\exp(\frac{1-c^2}{2})\right)^{2n} \leq 2^{-40}\,.$$

For $\mathbf{x},\mathbf{s} \xleftarrow{\$} D^n_{\sigma_e}, \mathbf{r}, \mathbf{r}_0 \xleftarrow{\$} D^n_{\sigma_e}, e_1 \xleftarrow{\$} D_{\sigma_e}$, we have $|\langle \mathbf{x},\mathbf{r}\rangle - \langle \mathbf{s},\mathbf{e}_0\rangle + e_1| < T$ with probability at least $1-\delta-2^{-40}$.

We are now ready to prove the following theorem.

Theorem 5. *Let c, T be as in Lemma 6. Then, the decryption is correct with probability at least $1-\delta-2^{-40}$.*

9.3.4. Ciphertext Indistinguishability

Theorem 6. *Let $X=\{X_\kappa\}_{\kappa\in\mathbb{N}}$ and $Y=\{Y_\kappa\}_{\kappa\in\mathbb{N}}$ be two distribution ensembles. We say X and Y are (t,ϵ)-computationally indistinguishable if for every PPT distinguisher $\mathcal{A}$ with running time t, there exists a function $\epsilon(\kappa)$ such that $|\Pr[\mathcal{A}(X)=1]-\Pr[\mathcal{A}(Y)=1]| \leq \epsilon(\kappa)$ (and we write $X \approx_{(t,\epsilon)} Y$). If $\mathcal{A}$ is PPT and $\epsilon(\kappa)$ is negligible, we simply say X and Y are (computationally) indistinguishable (and we write $X \approx Y$).*

Let $\mathbb{G}=\langle g\rangle$ be the cyclic group of composite order generated by g. If the decisional LWEE problem $\mathsf{DLWEE}_{\mathbb{G},n,n+1,q,D_{\sigma_e}}(D_{\sigma_s})$ is (t,ϵ)-hard, then the above cryptosystem is $(t,2\epsilon)$-indistinguishable against chosen plaintext attacks.

Proof. In a high level, our proof works as follows. Instead of showing IND-CPA security via a direct argument we show that the distribution $(\mathsf{pk}, \mathbf{c}_0, c_1)$ is indistinguishable from the uniform distribution over $(\mathbb{G}^{n\times n} \times \mathbb{G}^{2n+1})$. That is, a ciphertext $(\mathbf{c}_0, c_1)$ under public key pk appears completely random to an adversary. This holds, in particular, in the IND-CPA experiment when the adversary chooses the underlying plaintext. We prove the theorem via a series of hybrid arguments, Hybrid_0 to Hybrid_2, where in each consecutive argument we make some slight changes with the provision that the adversary notices the changes with negligible probability only. In the following, we use the abbreviations $\mathbf{u} = \mathbf{A}\mathbf{r} + \mathbf{e}_0$ and $v = \langle \mathbf{b}, \mathbf{r}\rangle + e_1 + 2^{k-\ell}\mu$.

Hybrid_0: In this hybrid we consider the original distribution of the tuple

$$(\mathsf{pk}, (\mathbf{c}_0, \mathbf{c}_1)) = (g^{\mathbf{A}}, g^{\mathbf{b}}, g^{\mathbf{u}}, g^{v}).$$

Hybrid_1: In this hybrid we modify the distribution and claim

$$(g^{\mathbf{A}}, g^{\mathbf{b}}, g^{\mathbf{u}}, g^{v}) \approx_c (g^{\mathbf{A}'}, g^{\mathbf{b}'}, g^{\mathbf{A}'\mathbf{r}+\mathbf{e}_0}, g^{\langle \mathbf{b}'\cdot\mathbf{r}\rangle+e_1} \cdot g^{2^{k-\ell}\mu})$$

for a uniformly sampled elements $g^{\mathbf{A}'}, g^{\mathbf{b}'} \in \mathbb{G}^{n\times n} \times \mathbb{G}^n$. We argue that any successful algorithm distinguishing between Hybrid_0 and Hybrid_1 can be easily turned into a successful distinguisher $\mathcal{B}$ in the $\mathsf{DLWEE}_{\mathbb{G},n,n,q,D_{\sigma_e}}(D_{\sigma_s})$ problem. The DLWEE-adversary $\mathcal{B}$ is given as challenge the tuple $(g^{\mathbf{A}}, g^{\mathbf{b}})$ and is asked to decide whether there exist vectors $\mathbf{s} \xleftarrow{\$} D_{\sigma_s}$, $\mathbf{x} \xleftarrow{\$} D^n_{\sigma_e}$ such that $g^{\mathbf{b}} = g^{\mathbf{A}^\top\mathbf{s}+\mathbf{x}}$ or $g^{\mathbf{b}}$ was sampled uniformly from $\mathbb{G}^n$.

Let $\Pr[\mathsf{Hybrid}_i(t)]$ denote the probability of any algorithm with runtime t to win the IND-CPA experiment in hybrid i. Then, we have

$$\Pr[\mathsf{Hybrid}_0(t)] \leq \Pr[\mathsf{Hybrid}_1(t)] + \mathsf{Adv}^{\mathsf{DLWEE}}_{\mathbb{G},n,n,q,D_{\sigma_e},D_{\sigma_s}}(t)\ .$$

Hybrid_2: In this hybrid we modify the distribution and claim

$$(g^{\mathbf{A}'}, g^{\mathbf{b}'}, g^{\mathbf{A}'\mathbf{r}+\mathbf{e}_0}, g^{\langle \mathbf{b}'\cdot\mathbf{r}\rangle+e_1} \cdot g^{2^{k-1}\mu}) \approx_c (g^{\mathbf{A}''}, g^{\mathbf{b}''}, g^{\mathbf{u}'}, g^{v'} \cdot g^{2^{k-1}\mu})$$

for a uniformly sampled elements $g^{\mathbf{A}''}, g^{\mathbf{b}''}, g^{\mathbf{u}'}, g^{v'} \cdot g^{\mu} \in \mathbb{G}^{(n+1)\times n} \times \mathbb{G}^{n+1}$. We argue that any successful algorithm distinguishing between Hybrid_1 and Hybrid_2 can be easily turned into a successful distinguisher $\mathcal{B}$ against the $\mathsf{DLWEE}_{\mathbb{G},n,n+1,q,D_{\sigma_e}}(D_{\sigma_s})$ problem. Note that $g^{\mathbf{b}'}, g^{\langle \mathbf{b}'\cdot\mathbf{r}\rangle+e_1}$ is an additional sample from the LWEE distribution from which $g^{\mathbf{A}'}, g^{\mathbf{A}'\mathbf{r}+\mathbf{e}_0}$ is sampled.

We have

$$\Pr[\mathsf{Hybrid}_1(t)] \leq \Pr[\mathsf{Hybrid}_2(t)] + \mathsf{Adv}^{\mathsf{DLWEE}}_{\mathbb{G},n,n+1,q,D_{\sigma_e},D_{\sigma_s}}(t)\ .$$

Note that now all exponents are uniformly distributed, and, in particular, independent of μ and thus, independent of b in the IND-CPA game. Hence, any algorithm has in Hybrid_2 exactly a success probability of $1/2$.

This completes the proof of semantic security. □

9.4. Parameter Selection

Here, we review the decisional representation and learning with errors problem. Our aim is to give empirical arguments of the hardness of learning with errors in the exponent and justify the parameter choices in the main paper.

9.4.1. Hardness of the Decisional Representation Problem

Little is known about the decisional representation problem. Proposition 1 (main paper) shows that $(\ell+1)$-DRP for any ℓ is generically at least as hard as the ℓ-DRP problem. As ℓ-DRP for $\ell = 1$ coincides with DDH, we lay our argumentation on the well-studied decisional Diffie-Hellman problem.

Decisional Diffie-Hellman. We start by recalling groups in which the DDH problem is believed to be instractable. Boneh gives several examples in [Bon98]. Among them are the following ones:

1. In the cyclic subgroup $\mathbb{QR}(p) \subset \mathbb{Z}_p^*$ of quadratic residues in $\mathbb{Z}_p^*$, where $p = 2p' + 1$ with p and p' both prime, DDH is believed to be intractable.

2. Let $N = pq$ for primes $p, q, \frac{(p-1)}{2}, \frac{(q-1)}{2}$. The cyclic subgroup T in $\mathbb{Z}_N^*$ of non-prime order $(p-1)(q-1)/2$ is believed to be a DDH-hard group. The same is claimed for subgroup $\mathbb{QR}_N \subset \mathbb{Z}_N^*$ of order $(p-1)(q-1)/4$, which even holds if p, q is known, and thus, the hardness of DDH is independent of the factorization [KY05].

3. The elliptic curve $E_{a,b}/\mathbb{F}_p$ where $|E_{a,b}|$ and p are prime is believed to resist against DDH attacks.

Note that one might believe that the multiplicative group $\mathbb{Z}_p^*$ with prime p is a safe choice. However, this group has an even order which is also publicly known. Hence, one can evaluate the Legendre symbol on g^a and g^b and compare the result with the given challenge g^c. This gives a significant non-negligible advantage to a distinguisher. Moreover, the group of signed quadratic residues $\mathbb{QR}_N^+ := \{|x| \ : \ x \in \mathbb{QR}_N\}$, introduced in [FS97] and revisited in [HK09] is publicly recognizable and thus non DDH-hard.

Trapdoor Decisional Diffie-Hellman. While many cryptographic applications can be instantiated in the above groups, our encryption scheme requires a special DDH-hard group where DDH is easy given a secret trapdoor. The requirement is reminiscent of trapdoor decisional Diffie-Hellman (TDDH) groups, introduced by Dent and Galbraith [DG06] and studied further by Seurin [Seu13]. Informally, TDDH groups satisfy two properties: (i) the DDH problem is assumed to be hard without

a trapdoor, (ii) DDH becomes easy but CDH remains hard given a trapdoor. Thus, anyone in possession of the trapdoor is able to efficiently solve the DDH problem. We remark that for our construction groups satisfying property (i) suffice, and we do not necessarily require hardness of CDH.

Looking at TDDH groups, there are several candidates:

1. Dent and Galbraith [DG06] gave two constructions based on hidden pairings. Here, the trapdoor permits to compute pairings on a specific elliptic curve what is assumed to be infeasible without the trapdoor. One such construction of a TDDH is as follows. Let $N = pq$ be an Blum integer, i.e., the product of two primes $p \equiv q \equiv 3 \mod 4$, where there exists two large primes p' and q' such that $p'|(p+1)$ and $q'|(q+1)$. The order of an elliptic curve $E : y^2 = x^3 + x$ over the ring $\mathbb{Z}_N$ is $|E(\mathbb{Z}_N)| = (p+1)(q+1)$. The group $E(\mathbb{Z}_N)$ with the generator point $P = (x_P, y_P) \in E(\mathbb{Z}_N)$ of order $p'q'$ is assumed DDH-hard. However, if one is given the trapdoor $\tau = (p, p', q, q')$, one can solve the DDH problem by the Chinese Remainder Theorem. A tuple $(A, B, C) \in E(\mathbb{Z}_N)^3$ is a true DDH tuple iff the elements reduce modulo p and q to valid tuples in the subcurve $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$. Those two checks can be performed with the knowledge of the trapdoor using Weil or Tate pairing [MOV93, FMR99]. In fact, given the trapdoor one can also efficiently test subgroup memberships. Dent and Galbraith [DG06] also consider an elliptic curve E over $\mathbb{F}_{2^{mn}}$ with mn being odd. Again, a hidden pairing allows one to solve DDH with the knowledge of a trapdoor.

2. Seurin [Seu13] continues the study and identifies additional trapdoor groups. Let $N = pq$ where p, q are safe primes, i.e., p and q are of the form $p = 2p' + 1$ and $q = 2q' + 1$ where p', q' are prime. The DDH problem in the group $\mathbb{QR}_{N^2}$ of quadratic residues modulo N^2 is hard given the description of $\mathbb{Z}_N^*$ if factoring N is hard. The use of a trapdoor $\tau = (p, q)$ which is the factorization of N enables to solve the DDH efficiently. $\mathbb{QR}_{N^2}$ is a cyclic group of order $\mathsf{ord}(\mathbb{QR}_{N^2}) = Np'q'$.

3. Let N be as before. The subgroup $\mathbb{J}_N$ of $\mathbb{Z}_N^*$ consists of all elements $x \in \mathbb{Z}_N^*$ such that $\mathbb{J}(x, N) = 1$. This subgroup has order $\mathsf{ord}(\mathbb{J}_N) = \phi(N)/2 = 2p'q'$. Moreover, $\mathbb{J}_N$ is cyclic because all prime factors of $\phi(N)/4 = p'q'$ are (pairwise) distinct [HK09]. Given the description of $\mathbb{J}_N$ with generator $g \in \mathbb{J}_N$, it is assumed one cannot solve the DDH problem in $\mathbb{J}_N$ without knowledge of the factorization of N. The trapdoor here is thus defined as $\tau = (p, q)$ or $\tau = \mathsf{ord}(\mathbb{J}_N)$. The assumption known as the quadratic residues problem appeared first in the security proof of the Goldwasser and Micali cryptosystem [GM82]. Joye and Libert [JL13] generalize the Goldwasser-Micali encryption scheme to groups $\mathbb{J}_N$ where p and q are *k-quasi-safe-primes*. That is, p (resp. q) are of the form $p = 2^k p' + 1$ (resp. $q = 2^k q' + 1$) where p, p', q, q' are all prime.

Security / Group	$E_{a,b}/\mathbb{F}_p$ $a, b \xleftarrow{\$} \mathbb{F}_p$	$\mathbb{QR}(p) \subset \mathbb{Z}_p^*$ p safe prime	$\mathbb{J}_N \subset \mathbb{Z}_N^*$ p, q safe primes	$\mathbb{J}_N \subset \mathbb{Z}_N^*$ p, q k-safe-primes
80-bit	$\log p \approx 160$	$\log p \approx 1130$	$\log N = 1130$ $\log p \approx \log q \geq 160$	$\log N = 1130$ with $k < 202$ $\log p \approx \log q \geq 565$
128-bit	$\log p \approx 256$	$\log p \approx 3000$	$\log N = 3000$ $\log p \approx \log q \geq 256$	$\log N = 3000$ with $k < 622$ $\log p \approx \log q \geq 1500$
256-bit	$\log p \approx 512$	$\log p \approx 15000$	$\log N = 15000$ $\log p \approx \log q \geq 512$	$\log N = 15000$ with $k < 3494$ $\log p \approx \log q \geq 7500$

Table 9.1.: Example instantiation of some DDH-hard groups for different security levels

They assume that without knowing the factorization of N, random elements of $\mathbb{QR}_N$ are computationally indistinguishable from elements in $\mathbb{J}_N/\mathbb{QR}_N$. This assumption is believed to hold for the group even when the distinguisher is given k.

Parameters. For all the above groups, there exists no specialized DDH distinguisher. In fact, the best algorithms to solve the DDH problem is to solve the DL problem in that group. Some groups above are assumed to be hard only if also factoring a composite number $N = pq$ of two large primes is hard or the quadratic residue assumption holds.

In Table 9.1 we give instantiations for four of the above groups for different security levels. We select

(a) the elliptic curve $E_{a,b}/\mathbb{F}_p$ where $|E_{a,b}|$ and p are prime,

(b) the subgroup $\mathbb{QR}(p)$ of quadratic residues in $\mathbb{Z}_p^*$, where $p = 2p' + 1$ with p and p' both prime,

(c) the subgroup $\mathbb{J}_N$ of $\mathbb{Z}_N^*$ defined as $\{x \in \mathbb{Z}_N^* \mid \mathbb{J}(x, N) = 1\}$ where $N = pq$ and p, q being safe primes, and

(d) the subgroup $\mathbb{J}_N$ of $\mathbb{Z}_N^*$ defined as $\{x \in \mathbb{Z}_N^* \mid \mathbb{J}(x, N) = 1\}$ where $N = pq$ and p, q being k-quasi-safe-primes,

where the latter subgroup is particularly important for the instantiation of our encryption scheme. Note that the remaining groups are appealing to instantiate LWEE in other applications.

When instantiating the group in finite fields or in elliptic curves, one chooses the Number Field Sieve (NFS) algorithm (for finite fields such as in (b)) or the Pohlig-Hellman [PH78] (and resp. Pollard-Rho [Pol78]) algorithm (in a generic group such as in (a)). For the groups (c) and (d) we use the results from [Mil75, Bac84].

Bach [Bac84] and Miller [Mil75] show that if there exists a PPT algorithm $\mathcal{A}$ solving the DL problem for a composite modulus on all inputs, then a PPT algorithm exists which solves the Factoring problem with arbitrarily high probability. Hence, we demand that factoring is hard for which the best algorithm is NFS, too. If we consider the computation of discrete logarithms in subgroups T (of order p) of a multiplicative group $\mathbb{G}$ (of order q), DL in T is hard if the NFS attack in $\mathbb{G}$ and the generic Pollard-Rho attack for groups of order $|T| = p$ is hard. Moreover, in the group $\mathbb{J}_N$ in (d) we have that $(p-1)$ and $(q-1)$ share common factors, namely 2^k, for which one can apply McKee and Pinch's algorithm [MP98], factoring $N = pq$ in essentially $O(N^{1/4}/2^k)$ operations. This is also observed in [Gir91, LL95, JL13]. Furthermore, the Coppersmith algorithm [Cop96, Cop97] (based on LLL) factors N efficiently if $k > \frac{1}{2}\min(\log_2 p, \log_2 q)$. For this reason we pick primes p, q of similar bit length and hinder both attack algorithms. NFS [CS06] has the running time $L_p[1/3, \sqrt[3]{64/9}]$ for modulus p where the complexity function $L_p[t, s]$ is defined by $L_p[t, s] = e^{s(1+o(1))(\ln p)^t(\ln\ln p)^{1-t}}$. The Pohlig-Hellman [PH78] and Pollard-Rho [Pol78] algorithms take time roughly $\sqrt{p}$ for computing individual discrete logarithms.

When estimating security parameters we take previously known attacks and timings into account by saying that if computing discrete logarithms in groups of order p takes time t, then we expect that computing DLs in groups of order p' takes time roughly $t' \approx t\frac{L_{p'}[1/3, \sqrt[3]{64/9}]}{L_p[1/3, \sqrt[3]{64/9}]}$. If the difference between p' and p is not too large, the term $o(1)$ goes to zero. A similar strategy has been recommended in [LV00].

We take as reference the 2009 factorization of a 768-bit modulus, which offers roughly 66 security bits ($t \approx 2^{66}$). We stress that the parameters suggested in Table 9.1 should be handled with care. If one selects parameters for cryptographic constructions based on the hardness of DRP or LWEE, respectively, then the tightness of security reduction to the underlying problem takes an important role. Assume the security reduction says that if an adversary $\mathcal{A}$ breaks the security of the cryptographic scheme in time t with probability ϵ, then one can solve the $\mathsf{DRP}_{\mathbb{G},\ell}$ problem in time t' with probability ϵ'. In order that the scheme offers κ security bits, the parameters have to be chosen such that $(t'\epsilon)/(t\epsilon) \leq 2^\kappa$. Thus, one has to compensate a non-tight reduction by strengthening the underlying hardness assumption.

9.4.2. Hardness of the Learning with Errors Problem

Determining the hardness of lattice-based problems is a delicate issue. There are several reasons for this. First, lattice problem instances typically are defined over multiple parameters. Thus, solvers rather depend on the particular configuration of the problem instance. Second, there are merely a few theoretical results known about the behavior and running time of lattice algorithms, and those predictions are not close to practical running times.

There are several approaches to solve the LWE problem. Not all of them are directly related to lattices. We start by explaining the "non-lattice-related" solvers. Arora and Ge [AG11] introduced an algebraic attack which, unfortunately, requires way too many samples to be applied in practice. Recently, Albrecht et al. [ACF+14] gave an improved version using Groebner basis techniques that can be applied to LWE with fewer samples, but it is still much slower than other known algorithms. Another non-lattice approach is the attack by Blum, Kalai and Wasserman [BKW03]. While it may be the asymptotically fastest approach for LWE, it requires subexponentially many samples and can therefore not be applied here.

The most basic lattice-based approach is the distinguishing approach [LP11]. However, Lindner and Peikert [LP11] showed that it is outperformed by a direct decoding attack. There are several variants of this attack, ranging from the basis nearest-plane algorithm by Babai [Bab86] to the latest enumeration variant using known techniques from SVP solvers by Liu and Nguyen [LN13]. The third approach is to use the so-called embedding technique to reduce LWE to an instance of the unique-shortest vector problem [AFG13, BG14b].

So far, there is not one single attack performing best for all scenarios. Since our scheme is based on a very limited number of samples, we focus on lattice attacks, specifically on the decoding attack and the embedding technique. Albrecht et al. [APS15] evaluate the attacks for concrete LWE instances. It turns out that the embedding technique outperforms the decoding attack only in very few cases (and the advantage in those rare cases is not big). We consequently concentrate on the decoding attack in this work.

Nearest-Plane Approach. Linder-Peikert's attack [LP11] is a generalized Babai's nearest plane algorithm [Bab86]. It was further improved by Liu and Nguyen [LN13] by adapting two techniques from known SVP solvers: randomization and (extreme) pruning. While randomization leads to a significant improvement, the speedup of pruning is rather small (and the pruned version is much harder to analyze). We will, therefore, consider in the following the randomized, but none-pruned version. The attack consists of two steps: (a) a basis reduction to precompute a good basis of a lattice defined by the matrix $\mathbf{A}$, and (b) a probabilistic search algorithm with a success probability related to the quality of the basis. Lindner-Peikert's approach inherently allows a trade-off between the time spend on the basis reduction and the search algorithm. That trade-off is controlled by the Hermite factor δ, which measures the quality of the basis. We say that a basis $\mathcal{B} = \{b_0, \ldots, b_{m-1}\}$ of an m-dimensional lattice Λ has Hermite factor δ, if $\|b_0\| = \delta^m \det(\Lambda)^{1/m}$. For a given probability p and Hermite factor δ, one can compute the effort of the search algorithm needed to succeed at least with success p. Lindner and Peikert claim in [LP11] that it takes about 2^{-16} seconds to perform one "search-step" (for readers familiar with the nearest-plane algorithm: to search one parallelepiped spanned by

the Gram-Schmidt orthogonalized basis). This allows us to estimate the running time of the search step, given p and δ. It is folklore that the running time of a basis reduction depends mainly on the desired Hermite factor of the reduced basis. The original paper considers the BKZ basis-reduction algorithm [SE94]. There have however been several improvements to BKZ. Most improvements are summarized in the remarkable work by Chen and Nguyen [CN11]. The BKZ 2.0 algorithm comes together with a simulation algorithm that can be used to predict its behavior. Albrecht et al. [AFG13] used the results of [LN13] to give an easy formula that roughly estimates the running time t necessary to compute a basis with given Hermite factor. They conjecture that the time t can be approximated by $\log_2(t) = 0.009/\log_2^2 \delta_0 - 27$.

Parameters. Since we are now able to estimate the total running time of the attack, given the desired success probability and Hermite factor, we can use a numerical method to obtain the best parameters and thereby the expected running time necessary to break the LWE instance. Given that the computers used for these experiments execute about 2^{10} operations per second, this can be used to estimate the bit security of LWE instances. Table 9.2 summarizes the results.

Security / Parameters	n	modulus	σ
80-bit	240	327680	33.98
128-bit	320	327680	32.01
256-bit	550	327680	28.55

Table 9.2.: Example instantiation of LWE for different security levels

Exponential Gap Between Error and Modulus. For our double hardness instantiation, we have to estimate the security of LWE instances with an exponential gap between the error size and the modulus. The hardness of LWE with exponentially small gap between error and modulus is not well understood today. Brakerski and Vaikuntanathan [BV11] say that if the error is a $1/2^{n^\varepsilon}$ fraction of the modulus N, the best known algorithm runs in time approximately $2^{n^{1-\varepsilon}}$. With the methodology, we can perform a binary search for the smallest dimension that suits our needs. Table 9.3 gives LWE instances that are suitable for double hardness instantiation of our scheme.

9.4.3. Candidate Instantiations of our Encryption Scheme

We give three possible instantiations to derive a system with short key sizes, post-quantum security or double hardness. Throughout this section we instantiate our

Security / Parameters	n	log(modulus)	$\log(\sigma)$
80-bit	67000	927	97
128-bit	270000	2378	306
256-bit	2500000	11506	1741

Table 9.3.: Example instantiation of LWE for different security levels

scheme such that the encryption scheme from Section 9.3.2 encrypts only a single bit. Nonetheless, parameters can easily be upscaled to many bits.

Table 9.4.: Key sizes in kilobytes (kB) for our encryption scheme basing security on DRP or LWE, respectively.

Sizes	DRP-based instantiation			LWE-based instantiation		
/ Security	80-bit	128-bit	256-bit	80-bit	128-bit	256-bit
public-key size	0.565 kB	1.500 kB	7.500 kB	235 kB	417 kB	1233 kB
secret-key size	0.212 kB	0.563 kB	2.813 kB	0.976 kB	1.302 kB	2.237 kB
ciphertext size	0.283 kB	0.750 kB	3.750 kB	0.980 kB	1.306 kB	2.241 kB

The Classical Way. Here, we instantiate our encryption scheme such that the underlying DRP is intractable, and neglecting the hardness of the underlying LWE. In Section 9.4.1, we recall some groups where we believe DRP is hard to solve. Our encryption scheme works in the group $\mathbb{J}_N := \{x \in \mathbb{Z}_N \ : \ \mathbb{J}(x, N) = 1\}$ for $N = pq$ with p, q being k-safe primes. In fact, we can even take safe primes p, q (i.e., $k = 1$) since we do not need any noise in the exponent if we neglect the underlying LWE hardness. Thus, we embed the message to the least significant bit in the exponent. For this reason, we can sample $g \xleftarrow{\$} \mathbb{J}_N/\mathbb{QR}_N$ where $\langle g \rangle$ has order $2p'q'$. Since the LWE instance within LWEE is not an issue here we select $n = m = 1$, $\sigma_s = \infty$ and $\sigma_e = 0$.

We obtain 80-bit security for the underlying DRP problem if we choose safe primes p and q such that $\log p = \log q = 565$ (see Table 9.1 for more details). Table 9.4 lists possible key sizes for our encryption scheme. Recall that the public key consists of $\mathsf{pk} = (g, g^{\mathbf{A}}, g^{\mathbf{b}}, k, N)$ (i.e., 4 group elements if we fix $k = 1$) and the secret key of $\mathsf{sk} = (p, \mathbf{s})$.

The Post-Quantum Way. Here we give example instantiations of our encryption scheme when it is based on a presumably quantum-resistant LWEE assumption. That is, we select parameters such that the underlying LWE assumption is intractable without relying on the hardness of DRP. For this, we modify the scheme slightly

by choosing fixed values for p' and q' instead of sampling. A good choice is $k = 15$, since it allows to choose $p' = 2$ and $q' = 5$, which are very small prime numbers such that $2^k p' + 1$ and $2^k q' + 1$ are prime. For the LWE modulus, this leads to $M = 2^k p' q' = 327680$. Like Lindner and Peikert [LP11], we choose the Gaussian parameter such that the probability of decoding errors is bounded by 1%. We choose furthermore the same parameter for error and secret distribution (i.e. $\sigma_s = \sigma_e = \sigma$), since a standard argument reduces LWE with arbitrary secret to LWE with secret chosen according to the error distribution. For this choice of k, p' and q', we obtain 80-bit security by choosing $n = 240$ and $\sigma = 33.98$. Table 9.4 lists the key sizes when our encryption scheme is instantiated such that its security is based on LWE only (see Table 9.2 for more information about the concrete hardness of LWE).

The Hardest Way (Double-Hardness). The most secure instantiation of our encryption is such that even if one of the problems DRP or LWE is efficiently solvable at some point, our encryption scheme remains semantically secure. Selecting parameters for double hardness, however, is non-trivial.

To select appropriate parameters for the case of double hardness, we apply the following approach: For a given security level (say $\kappa = 80$), we select N such that the Number Field Sieve needs at least 2^κ operations to factor N. A possible choice is $\log N = 1130$ (See Table 9.1). Since factoring N must also be hard for McKee-Pinch's algorithm, which works well when $(p-1)$ and $(q-1)$ share common factor, k must be chosen such that $N^{1/4} 2^{-k} \geq 2^\kappa$, i.e. $k \leq \frac{\log(N)}{4} - \kappa$. This leads to $k = 203$. Given N and k, we can calculate the sizes of the primes $\log(p') \approx \log(q') \approx 362$ and $log(p) \approx \log(q) \approx 565$ and the LWE modulus $\log(M) \approx 927$. Taking $n = 67000$ and $\sigma = 2^{97}$, Lemma 6 shows that the algorithm decrypts correctly with high probability.

10 Conclusion

In this work, we introduced new methods to select parameters for LWE-based schemes. We showed that parameter selection is a challenging, but rewarding task.

Concerning the challenges, our results show that cautious parameter selection is necessary, especially if the underlying security assumption is modified to achieve additional speed-up. We also showed that it is crucial to take modern hardware architecture into consideration. Of course, new developments of LWE cryptanalysis is a group effort that requires contributions of a bigger community. Consequently, we introduced the LWE challenge that allows to bunch and monitor the progress.

On the other hand, a careful parameter selection can lead to significant efficiency improvements. We showed that LWE instances used for typical cryptographic schemes do not provide enough samples for many attacks. In addition, even the attacks that can succeed with the limited number of samples suffer from the limitation, which results in increased run times. Taking this into consideration leads to significant higher attack runtimes, which we used to give improved hardness estimations for LWE.

Furthermore, we used the new hardness results to construct and instantiate improved cryptographic schemes. The schemes may be of independent interest and cover both main aspects of efficiency (speed and memory consumption), and in addition show that advanced security properties are possible.

The first important aspect of efficiency is the speed of the algorithms. Our improvement of the signature scheme by Bai and Galbraith's lattice-based signature scheme [BG14b] can compete with the best alternatives in terms of speed, despite the fact that it is based on standard lattices. Building on our results, Bindel et al. invented TESLA [ABBD15] and ring-TESLA [ABB+16], that are among the most promising candidates for lattice-based signature schemes.

The second aspect of efficiency is memory consumption. In order to show that LWE-based encryption schemes can have an extremely small memory footprint, we introduced a new encryption scheme based on the proposal by Lindner and Peikert [LP11]. Our new scheme only requires about one kByte of ROM, which is at least comparable to all existing alternatives (see Figure 8.2). With more than 20 encryptions and decryptions per seconds on low-cost microcontrollers, it is also fast

enough for most use-cases and a good candidate for public-key encryption in the internet of things.

Future work As mentioned in the introduction, an important future work is to investigate potential speed ups by Grover's search algorithm. A good starting point is to take a closer look at parallel attacks, since it is a good rule-of-thumb that those attacks can benefit from Grover. In particular, this is true for the parallel decoding attack presented in Chapter 4. However, Section 4.3 shows that a speed up in this area would not lead to a significant overall improvement.

More promising attempts try to speed up basis reduction algorithms, in particular BKZ. They are used as subroutines in many attacks (see Sections 2.3.2, 2.3.3, 2.3.4, 2.3.5, and Chapter 5), and have significant influence on the overall runtime for most of them. Since the only exponential-runtime part of BKZ is an SVP solver for projected sublattices, research focuses on those at the moment. A second application of SVP solvers are the embedding attacks (see Section 2.3.4 and 2.3.5).

At the moment, two different types of SVP solvers are used: Enumeration and sieving algorithms. Despite significant speed ups for sieving algorithms (see [Laa15, BL16]), enumeration still seems to be preferable in practice. However, no significant progress was made concerning improved enumeration on quantum computers so far. In contrast to this, sieving algorithms can benefit from Grover. An early result was presented by Laarhofen et al. [LMvdP13], and new estimates where recently used to estimate the hardness of the "new hope protocol" [ADPS16], an LWE-based key exchange protocol based on the proposal by Ding [Din12].

Algebraic attacks on LWE like BKW could also benefit from Grover's algorithm. In fact, BKW looks like a perfect candidate at first sight, since a big part of it consists of finding collisions in LWE samples. Indeed, Grover-like algorithms can speed up collision finding from $\mathcal{O}(\sqrt{2^n})$ to $\mathcal{O}(\sqrt[3]{2^n})$. Unfortunately, this is not directly applicable to BKW, since BKW requires exponentially many collisions, and the speed up of Grover collision finding vanishes in this case.

Bibliography

[ABB+16] Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In Pointcheval et al. [PNR16], pages 44–60. 3, 109

[ABBD15] Erdem Alkim, Nina Bindel, Johannes A. Buchmann, and Özgür Dagdelen. TESLA: tightly-secure efficient signatures from standard lattices. *IACR Cryptology ePrint Archive*, 2015:755, 2015. iii, 3, 11, 109

[ABV+11] Shweta Agrawal, Xavier Boyen, Vinod Vaikuntanathan, Panagiotis Voulgaris, and Hoeteck Wee. Fuzzy identity based encryption from lattices. *IACR Cryptology ePrint Archive*, 2011:414, 2011. iii

[ACF+14] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. Algebraic algorithms for LWE problems. *IACR Cryptology ePrint Archive*, 2014:1018, 2014. 52, 53, 56, 85, 105

[ACF+15] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. On the complexity of the BKW algorithm on LWE. *Des. Codes Cryptography*, 74(2):325–354, 2015. 13, 14, 22, 52, 58

[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange—a new hope. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 327–343, Austin, TX, August 2016. USENIX Association. iii, 12, 110

[ADVW13] Shweta Agrawal, Yevgeniy Dodis, Vinod Vaikuntanathan, and Daniel Wichs. On continual leakage of discrete log representations. In *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, pages 401–420, 2013. 90, 92

[AFFP14] Martin R. Albrecht, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. Lazy modulus switching for the BKW algorithm on LWE. In Hugo Krawczyk, editor, *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings*, volume 8383 of *Lecture Notes in Computer Science*, pages 429–445. Springer, 2014. 13, 14, 52

[AFG13] Martin R. Albrecht, Robert Fitzpatrick, and Florian Göpfert. On the efficacy of solving LWE by reduction to unique-svp. In Hyang-Sook Lee and Dong-Guk Han, editors, *Information Security and Cryptology - ICISC 2013 - 16th International Conference, Seoul, Korea, November 27-29, 2013, Revised Selected Papers*, volume 8565 of *Lecture Notes in Computer Science*, pages 293–310. Springer, 2013. 1, 8, 10, 58, 69, 73, 74, 78, 105, 106

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Aceto et al. [AHS11], pages 403–415. 12, 21, 52, 53, 56, 58, 85, 105

[AHS11] Luca Aceto, Monika Henzinger, and Jiri Sgall, editors. *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I*, volume 6755 of *Lecture Notes in Computer Science*. Springer, 2011. 112

[AM15] Diego F. Aranha and Alfred Menezes, editors. *Progress in Cryptology - LATINCRYPT 2014 - Third International Conference on Cryptology and Information Security in Latin America, Florianópolis, Brazil, September 17-19, 2014, Revised Selected Papers*, volume 8895 of *Lecture Notes in Computer Science*. Springer, 2015. 116, 118

[APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *J. Mathematical Cryptology*, 9(3):169–203, 2015. 27, 35, 51, 52, 53, 55, 58, 59, 66, 69, 85, 105

[Bab86] László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986. 15, 58, 105

[Bac84] Eric Bach. Discrete logarithms and factoring. Technical Report UCB/CSD-84-186, EECS Department, University of California, Berkeley, Jun 1984. 103, 104

[BBD09] Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen, editors. *Post-Quantum Cryptography*. Springer, 2009. 18, 22, 75, 124

[BBD+14] Christian H. Bischof, Johannes A. Buchmann, Özgür Dagdelen, Robert Fitzpatrick, Florian Göpfert, and Artur Mariano. Nearest planes in practice. In Berna Ors and Bart Preneel, editors, *Cryptography and Information Security in the Balkans - First International Conference, BalkanCryptSec 2014, Istanbul, Turkey, October 16-17, 2014, Revised Selected Papers*, volume 9024 of *Lecture Notes in Computer Science*, pages 203–215. Springer, 2014. 1, 30, 33

[BBG+16] Johannes A. Buchmann, Niklas Büscher, Florian Göpfert, Stefan Katzenbeisser, Juliane Krämer, Daniele Micciancio, Sander Siim, Christine van Vredendaal, and Michael Walter. Creating cryptographic challenges using multi-party computation: The LWE challenge. In Keita Emura, Goichiro Hanaoka, and Rui Zhang, editors, *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography, AsiaPKCAsiaCCS, Xi'an, China, May 30 - June 03, 2016*, pages 11–20. ACM, 2016. 1, 57

[BCG+13] Johannes A. Buchmann, Daniel Cabarcas, Florian Göpfert, Andreas Hülsing, and Patrick Weiden. Discrete ziggurat: A time-memory tradeoff for sampling from a gaussian distribution over the integers. In Lange et al. [LLL14], pages 402–417. 2, 39

[BG14a] Shi Bai and Steven Galbraith. Personal communication and e-mail exchanges, 2014. 74

[BG14b] Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *Topics in Cryptology - CT-RSA 2014 - The Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings*, volume 8366 of *Lecture Notes in Computer Science*, pages 28–47. Springer, 2014. 6, 58, 59, 69, 70, 71, 72, 73, 74, 75, 78, 105, 109

[BG14c] Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary LWE. In Willy Susilo and Yi Mu, editors, *Information Security and Privacy - 19th Australasian Conference, ACISP 2014, Wollongong, NSW, Australia, July 7-9, 2014. Proceedings*, volume 8544 of *Lecture Notes in Computer Science*, pages 322–337. Springer, 2014. 39, 41, 58

[BGG+16] Johannes A. Buchmann, Florian Göpfert, Tim Güneysu, Tobias Oder, and Thomas Pöppelmann. High-performance and lightweight lattice-based public-key encryption. In Richard Chow and Gökay Saldamli, editors, *Proceedings of the 2nd ACM International Workshop on IoT*

Privacy, Trust, and Security, IoTPTSAsiaCCS, Xi'an, China, May 30, 2016, pages 2–9. ACM, 2016. 1, 5, 6, 39, 79

[BGLS14] Shi Bai, Steven D. Galbraith, Liangze Li, and Daniel Sheffield. Improved exponential-time algorithms for inhomogeneous-SIS. *IACR Cryptology ePrint Archive*, 2014:593, 2014. 52

[BGPW16] Johannes A. Buchmann, Florian Göpfert, Rachel Player, and Thomas Wunderer. On the hardness of LWE with binary error: Revisiting the hybrid lattice-reduction and meet-in-the-middle attack. In Pointcheval et al. [PNR16], pages 24–43. 1, 40, 85

[BGV11] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. Fully homomorphic encryption without bootstrapping. *IACR Cryptology ePrint Archive*, 2011:277, 2011. iii

[BHHO08] Dan Boneh, Shai Halevi, Mike Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In David Wagner, editor, *Advances in Cryptology - CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer Berlin Heidelberg, 2008. 92

[BHLY16] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. Flush, gauss, and reload - A cache attack on the BLISS lattice-based signature scheme. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *Lecture Notes in Computer Science*, pages 323–345. Springer, 2016. 39

[BKW03] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003. 12, 13, 52, 85, 105

[BL16] Anja Becker and Thijs Laarhoven. Efficient (ideal) lattice sieving using cross-polytope LSH. In Pointcheval et al. [PNR16], pages 3–23. 110

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 575–584. ACM, 2013. 4, 58, 85

[Bon98] Dan Boneh. The decision Diffie-Hellman problem. In JoeP. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer Berlin Heidelberg, 1998. 101

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 505–524. Springer Berlin Heidelberg, 2011. 106

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM J. Comput.*, 43(2):831–871, 2014. iii, 59

[CG13] Ran Canetti and Juan A. Garay, editors. *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*. Springer, 2013. 116, 124

[CGW14] Daniel Cabarcas, Florian Göpfert, and Patrick Weiden. Provably secure LWE encryption with smallish uniform noise and secret. In Keita Emura, Goichiro Hanaoka, and Yunlei Zhao, editors, *ASIAPKC'14, Proceedings of the 2nd ACM Wookshop on ASIA Public-Key Cryptography, June 3, 2014, Kyoto, Japan*, pages 33–42. ACM, 2014. 2, 39, 79

[CLS15] Hao Chen, Kristin E. Lauter, and Katherine E. Stange. Attacks on search RLWE. *IACR Cryptology ePrint Archive*, 2015:971, 2015. 39

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. SV, 2011. `http://www.iacr.org/archive/asiacrypt2011/70730001/70730001.pdf`. 106

[Cop96] Don Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In Ueli Maurer, editor, *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 178–189. Springer Berlin Heidelberg, 1996. 104

[Cop97] Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997. 104

[COT14] Alain Couvreur, Ayoub Otmani, and Jean-Pierre Tillich. Polynomial time attack on Wild McEliece over quadratic extensions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 17–39. SV, 2014. `https://eprint.iacr.org/2014/112/`. 72

[CS06] An Commeine and Igor Semaev. An algorithm to solve the discrete logarithm problem with the number field sieve. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 174–190. Springer Berlin Heidelberg, 2006. 104

[CWB14] Daniel Cabarcas, Patrick Weiden, and Johannes Buchmann. On the efficiency of provably secure NTRU. In Mosca [Mos14], pages 22–39. 83

[DBG+14] Özgür Dagdelen, Rachid El Bansarkhani, Florian Göpfert, Tim Güneysu, Tobias Oder, Thomas Pöppelmann, Ana Helena Sánchez, and Peter Schwabe. High-speed signatures from standard lattices. In Aranha and Menezes [AM15], pages 84–103. 1, 6, 69, 71

[dCRVV15] Ruan de Clercq, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. Efficient software implementation of ring-lwe encryption. In Wolfgang Nebel and David Atienza, editors, *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE 2015, Grenoble, France, March 9-13, 2015*, pages 339–344. ACM, 2015. 81

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Canetti and Garay [CG13], pages 40–56. iii, 11, 21, 58, 59, 61, 70, 71, 72, 79, 85

[DG06] Alexander W. Dent and Steven D. Galbraith. Hidden pairings and trapdoor DDH groups. In Florian Hess, Sebastian Pauli, and Michael Pohst, editors, *Algorithmic Number Theory*, volume 4076 of *Lecture Notes in Computer Science*, pages 436–451. Springer Berlin Heidelberg, 2006. 101, 102

[DGG15] Özgür Dagdelen, Sebastian Gajek, and Florian Göpfert. Learning with errors in the exponent. In Soonhak Kwon and Aaram Yun, editors, *Information Security and Cryptology - ICISC 2015 - 18th International Conference, Seoul, South Korea, November 25-27, 2015, Revised Selected Papers*, volume 9558 of *Lecture Notes in Computer Science*, pages 69–84. Springer, 2015. 1, 90

[Din12] Jintai Ding. A simple provably secure key exchange scheme based on the learning with errors problem. *IACR Cryptology ePrint Archive*, 2012:688, 2012. 12, 110

[DM13] Nico Döttling and Jörn Müller-Quade. Lossy codes and a new variant of the learning-with-errors problem. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2013. 52

[DS10] Özgür Dagdelen and Michael Schneider. Parallel enumeration of shortest lattice vectors. In *Euro-Par 2010 - Parallel Processing*, volume 6272 of *Lecture Notes in Computer Science*, pages 211–222. Springer Berlin Heidelberg, 2010. 34

[DTV15] Alexandre Duc, Florian Tramèr, and Serge Vaudenay. Better algorithms for LWE and LWR. In Oswald and Fischlin [OF15], pages 173–202. 13, 14, 52, 58

[DV14] Özgür Dagdelen and Daniele Venturi. A second look at Fischlin's transformation. In *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings*, pages 356–376, 2014. 90

[EHK+13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge L. Villar. An algebraic framework for Diffie-Hellman assumptions. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 129–147, 2013. 91

[EHL14] Kirsten Eisenträger, Sean Hallgren, and Kristin E. Lauter. Weak instances of PLWE. In Antoine Joux and Amr M. Youssef, editors, *Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers*, volume 8781 of *Lecture Notes in Computer Science*, pages 183–194. Springer, 2014. 39, 85

[ELOS15] Yara Elias, Kristin E. Lauter, Ekin Ozman, and Katherine E. Stange. Provably weak instances of Ring-LWE. In Gennaro and Robshaw [GR15], pages 63–92. 39

[FBB+14] Robert Fitzpatrick, Christian H. Bischof, Johannes A. Buchmann, Özgür Dagdelen, Florian Göpfert, Artur Mariano, and Bo-Yin Yang.

Tuning gausssieve for speed. In Aranha and Menezes [AM15], pages 288–305. 2

[FMR99] G. Frey, M. Müller, and H.-G. Rück. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *Information Theory, IEEE Transactions on*, 45(5):1717–1719, 1999. 102

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO '86*, volume 263 of *LNCS*, pages 186–194. SV, 1987. `http://www.cs.rit.edu/~jjk8346/FiatShamir.pdf`. 70

[FS97] R. Fischlin and C.P. Schnorr. Stronger security proofs for RSA and Rabin bits. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 267–279. Springer Berlin Heidelberg, 1997. 101

[GF05] Harold N. Gabow and Ronald Fagin, editors. *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*. ACM, 2005. 125

[GFS+12] Norman Göttert, Thomas Feller, Michael Schneider, Johannes Buchmann, and Sorin A. Huss. On the design of hardware building blocks for modern lattice-based encryption schemes. In Prouff and Schaumont [PS12], pages 512–529. 81, 84

[Gir91] Marc Girault. An identity-based identification scheme based on discrete logarithms modulo a composite number. In IvanBjerre Damgard, editor, *Advances in Cryptology - EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 481–486. Springer Berlin Heidelberg, 1991. 104

[GJS15] Qian Guo, Thomas Johansson, and Paul Stankovski. Coded-bkw: Solving LWE using lattice codes. In Gennaro and Robshaw [GR15], pages 23–42. 52

[GKPV10] Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In Andrew Chi-Chih Yao, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 230–240. Tsinghua University Press, 2010. 85, 94

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Prouff and Schaumont [PS12], pages 530–547. 39, 70, 85

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, STOC '82, pages 365–377, New York, NY, USA, 1982. ACM. 102

[GN08] Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 31–51. SV, 2008. `ftp://ftp.di.ens.fr/pub/users/pnguyen/Euro08.pdf`. 9

[GPW+04] Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and Sheueling Chang Shantz. Comparing elliptic curve cryptography and RSA on 8-bit cpus. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, volume 3156 of *LNCS*, pages 119–132. Springer, 2004. 86, 87

[GR15] Rosario Gennaro and Matthew Robshaw, editors. *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*. Springer, 2015. 117, 118, 121

[HHHW09] Philip S. Hirschhorn, Jeffrey Hoffstein, Nick Howgrave-Graham, and William Whyte. Choosing ntruencrypt parameters in light of combined lattice reduction and MITM approaches. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *Applied Cryptography and Network Security, 7th International Conference, ACNS 2009, Paris-Rocquencourt, France, June 2-5, 2009. Proceedings*, volume 5536 of *Lecture Notes in Computer Science*, pages 437–455, 2009. 44, 83, 84

[HK09] Dennis Hofheinz and Eike Kiltz. The group of signed quadratic residues and applications. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 637–653. Springer Berlin Heidelberg, 2009. 101, 102

[HMCD04] Yvonne Hitchcock, Paul Montague, Gary Carter, and Ed Dawson. The efficiency of solving multiple discrete logarithm problems and the im-

plications for the security of fixed elliptic curves. *International Journal of Information Security*, 3(2):86–98, 2004. 97

[How07] Nick Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 150–169. Springer, 2007. 44, 48, 50

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In Joe Buhler, editor, *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998. 85

[HSB+10] Jens Hermans, Michael Schneider, Johannes Buchmann, Frederik Vercauteren, and Bart Preneel. Parallel shortest lattice vector enumeration on graphics cards. In DanielJ. Bernstein and Tanja Lange, editors, *Progress in Cryptology – AFRICACRYPT 2010*, volume 6055 of *Lecture Notes in Computer Science*, pages 52–68. Springer Berlin Heidelberg, 2010. 34

[HvMG13] Stefan Heyse, Ingo von Maurich, and Tim Güneysu. Smaller keys for code-based cryptography: QC-MDPC McEliece implementations on embedded devices. In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings*, volume 8086 of *LNCS*, pages 273–292. Springer, 2013. 86, 87

[JL13] Marc Joye and BenoÃ®t Libert. Efficient cryptosystems from 2^k-th power residue symbols. In Thomas Johansson and PhongQ. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 76–92. Springer Berlin Heidelberg, 2013. 98, 102, 104

[KAF+10] Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Joppe W. Bos, Pierrick Gaudry, Alexander Kruppa, Peter L. Montgomery, Dag Arne Osvik, Herman J. J. te Riele, Andrey Timofeev, and Paul Zimmermann. Factorization of a 768-bit RSA modulus. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August*

15-19, 2010. Proceedings, volume 6223 of *Lecture Notes in Computer Science*, pages 333–350. Springer, 2010. 5

[Kan87] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, August 1987. 85

[KF15] Paul Kirchner and Pierre-Alain Fouque. An improved BKW algorithm for LWE with applications to cryptography and lattices. In Gennaro and Robshaw [GR15], pages 43–62. 52

[KS01] Fabian Kuhn and Rene Struik. Random walks revisited: Extensions of pollard's rho algorithm for computing multiple discrete logarithms. In *8th Annual Workshop on Selected Areas in Cryptography (SAC), Toronto, Ontario, Canada*, August 2001. 97

[KSD+11] Po-Chun Kuo, Michael Schneider, Özgür Dagdelen, Jan Reichelt, Johannes Buchmann, Chen-Mou Cheng, and Bo-Yin Yang. Extreme enumeration on gpu and in clouds. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems - CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 176–191. Springer Berlin Heidelberg, 2011. 34

[KV09] Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, pages 703–720, 2009. 90

[KY05] Aggelos Kiayias and Moti Yung. Efficient secure group signatures with dynamic joins and keeping anonymity against group managers. In Ed Dawson and Serge Vaudenay, editors, *Progress in Cryptology Mycrypt 2005*, volume 3715 of *Lecture Notes in Computer Science*, pages 151–170. Springer Berlin Heidelberg, 2005. 101

[Laa15] Thijs Laarhoven. Sieving for shortest vectors in lattices using angular locality-sensitive hashing. In Gennaro and Robshaw [GR15], pages 3–22. 110

[Li11] Shengqiao Li. Concise formulas for the area and volume of a hyperspherical cap. *Asian Journal of Mathematics and Statistics*, 4(1):66–70, 2011. 48

[LL95] Chae Hoon Lim and Pil Joong Lee. Security and performance of server-aided RSA computation protocols. In *CRYPTO*, pages 70–83, 1995. 104

[LL15] Kim Laine and Kristin E. Lauter. Key recovery for LWE in polynomial time. *IACR Cryptology ePrint Archive*, 2015:176, 2015. 39

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982. 57

[LLL14] Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors. *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*. Springer, 2014. 113, 125

[LLLS13] Fabien Laguillaumie, Adeline Langlois, Benoît Libert, and Damien Stehlé. Lattice-based group signatures with logarithmic signature size. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 41–61. Springer, 2013. iii

[LMvdP13] Thijs Laarhoven, Michele Mosca, and Joop van de Pol. Solving the shortest vector problem in lattices faster using quantum search. In Philippe Gaborit, editor, *Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013, Limoges, France, June 4-7, 2013. Proceedings*, volume 7932 of *Lecture Notes in Computer Science*, pages 83–101. Springer, 2013. 110

[LN13] Mingjie Liu and Phong Q. Nguyen. Solving BDD by enumeration: An update. In Ed Dawson, editor, *Topics in Cryptology - CT-RSA 2013 - The Cryptographers' Track at the RSA Conference 2013, San Francisco,CA, USA, February 25-March 1, 2013. Proceedings*, volume 7779 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2013. 33, 58, 84, 105, 106

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011. iii, 5, 6, 9, 10, 14, 15, 16, 21, 22, 23, 27, 29, 30, 33, 35, 38, 41, 50, 53, 58, 59, 69, 73, 78, 79, 81, 82, 83, 85, 97, 99, 105, 108, 109

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *LNCS*, pages 1–23. Springer, 2010. iii, 80, 81, 85

[LPS10] Vadim Lyubashevsky, Adriana Palacio, and Gil Segev. Public-key cryptographic primitives provably as secure as subset sum. In *Theory of Cryptography*, pages 382–400. Springer, 2010. 97

[LRBN] R. Lindner, M. Rueckert, P. Baumann, and L. Nobach. Lattice challenge. `http://www.latticechallenge.org/`. 85

[LS14] H.W. Lenstra and A. Silverberg. Revisiting the Gentry-Szydlo algorithm. In JuanA. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014*, volume 8616 of *LNCS*, pages 280–296. Springer Berlin Heidelberg, 2014. 85

[LSR+15] Zhe Liu, Hwajeong Seo, Sujoy Sinha Roy, Johann Großschädl, Howon Kim, and Ingrid Verbauwhede. Efficient ring-lwe encryption on 8-bit AVR processors. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, volume 9293 of *LNCS*, pages 663–682. Springer, 2015. 86, 87

[LV00] Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 446–465. Springer Berlin Heidelberg, 2000. 104

[Lyu05] Vadim Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In Chandra Chekuri, Klaus Jansen, José D. P. Rolim, and Luca Trevisan, editors, *Approximation, Randomization and Combinatorial Optimization, Algorithms and Techniques, 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2005 and 9th InternationalWorkshop on Randomization and Computation, RANDOM 2005, Berkeley, CA, USA, August 22-24, 2005, Proceedings*, volume 3624 of *Lecture Notes in Computer Science*, pages 378–389. Springer, 2005. 58

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. SV, 2009. `http://www.di.ens.fr/~lyubash/papers/FSAbortAsiacryptconf.pdf`. 70

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. SV, 2012. `https://eprint.iacr.org/2011/537`. 70

[Mil75] Gary L. Miller. Riemann's hypothesis and tests for primality. In *Proceedings of seventh annual ACM symposium on Theory of computing*, STOC '75, pages 234–239, New York, NY, USA, 1975. ACM. 103, 104

[Mos14] Michele Mosca, editor. *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, volume 8772 of *LNCS*. Springer, 2014. 116, 125

[MOV93] A.J. Menezes, T. Okamoto, and S.A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *Information Theory, IEEE Transactions on*, 39(5):1639–1646, 1993. 102

[MP98] JF McKee and RGE Pinch. Further attacks on server-aided RSA cryptosystems. *Unpublished manuscript*, 1998. 104

[MP13] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In Canetti and Garay [CG13], pages 21–39. 4, 39, 52, 85

[MR09] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In Bernstein et al. [BBD09], pages 147–191. 22, 24, 53, 56, 85

[NS09] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pages 18–35, 2009. 92

[OF15] Elisabeth Oswald and Marc Fischlin, editors. *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*. Springer, 2015. 117

[Olv10] Frank WJ Olver. *NIST handbook of mathematical functions*. Cambridge University Press, 2010. 42, 49

[Pei14] Chris Peikert. Lattice cryptography for the internet. In Mosca [Mos14], pages 197–219. 12, 83

[PG13] Thomas Pöppelmann and Tim Güneysu. Towards practical lattice-based public-key encryption on reconfigurable hardware. In Lange et al. [LLL14], pages 68–85. 83

[PH78] S.C. Pohlig and M.E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance (corresp.). *IEEE Transactions on Information Theory*, 24(1):106–110, 1978. 98, 103, 104

[PNR16] David Pointcheval, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors. *Progress in Cryptology - AFRICACRYPT 2016 - 8th International Conference on Cryptology in Africa, Fes, Morocco, April 13-15, 2016, Proceedings*, volume 9646 of *Lecture Notes in Computer Science*. Springer, 2016. 111, 114

[POG15] Thomas Pöppelmann, Tobias Oder, and Tim Güneysu. High-performance ideal lattice-based cryptography on 8-bit atxmega microcontrollers. In Kristin E. Lauter and Francisco Rodríguez-Henríquez, editors, *Progress in Cryptology - LATINCRYPT 2015 - 4th International Conference on Cryptology and Information Security in Latin America, Guadalajara, Mexico, August 23-26, 2015, Proceedings*, volume 9230 of *LNCS*, pages 346–365. Springer, 2015. 86, 87

[Pol78] John M Pollard. Monte Carlo methods for index computation (mod p). *Mathematics of computation*, 32(143):918–924, 1978. 103, 104

[PS12] Emmanuel Prouff and Patrick Schaumont, editors. *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*. Springer, 2012. 118, 119

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Gabow and Fagin [GF05], pages 84–93. iii, 3, 5, 39, 58, 59, 80, 83, 94, 97

[RVM+14] Sujoy Sinha Roy, Frederik Vercauteren, Nele Mentens, Donald Donglong Chen, and Ingrid Verbauwhede. Compact Ring-LWE cryptoprocessor. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International*

Workshop, Busan, South Korea, September 23-26, 2014. Proceedings, volume 8731 of *LNCS*, pages 371–391. Springer, 2014. 81

[S+14] W. A. Stein et al. *Sage Mathematics Software (Version 6.3)*. The Sage Development Team, 2014. `http://www.sagemath.org`. 50, 83

[SE94] C.P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66(1-3):181–199, 1994. 106

[Seu13] Yannick Seurin. New constructions and applications of trapdoor DDH groups. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography - PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 443–460. Springer Berlin Heidelberg, 2013. 101, 102

[Sha07] Hovav Shacham. A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. `http://eprint.iacr.org/`. 91

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. 3

[Yun14] Aaram Yun. Generic hardness of the multiple discrete logarithm problem. Cryptology ePrint Archive, Report 2014/637, 2014. `http://eprint.iacr.org/`. 97

Wissenschaftlicher Werdegang

Januar 2015 - heute Wissenschaftlicher Mitarbeiter in der Arbeitsgruppe von Prof. Johannes Buchmann, Fachbereich Informatik, Fachgebiet Theoretische Informatik - Kryptographie und Computeralgebra an der Technischen Universität Darmstadt im Sonderforschungsbereich 1119 CROSSING - Kryptographiebasierte Sicherheitslösungen als Grundlage für Vertrauen in heutigen und zukünftigen IT-Systemen der Deutschen Forschungsgemeinschaft (DFG)

Februar 2014 - Dezember 2014 Wissenschaftlicher Mitarbeiter in der Arbeitsgruppe von Prof. Johannes Buchmann, Fachbereich Informatik, Fachgebiet Theoretische Informatik - Kryptographie und Computeralgebra an der Technischen Universität Darmstadt im Software Campus

Oktober 2012 - Januar 2014 Promotionsstipendium bei Center for Advanced Security Research Darmstadt - CASED

September 2006 - September 2012 Studium der Mathematik (Diplom) mit Nebenfach Informatik an der Julius-Maximilians-Universität Würzburg

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit – abgesehen von den in ihr ausdrücklich genannten Hilfen – selbständig verfasst habe.

Darmstadt, August 2016 _____