PROVABLE QUANTUM-SECURE CRYPTOGRAPHY

Reduction-Respecting Parameters for Lattice-Based Cryptosystems

By Fletcher Gates, BMath.

A Thesis Submitted to the School of Graduate Studies in Partial Fulfillment of the Requirements for the Degree Master of Science

McMaster University © Copyright by Fletcher Gates, October 2018

McMaster University MASTER OF SCIENCE (2018) Hamilton, Ontario (Mathematics)

TITLE: Reduction-Respecting Parameters for Lattice-Based Cryptosystems

AUTHOR: Fletcher Gates, BMath. (University of Waterloo)

SUPERVISOR: Dr. Douglas Steblia

NUMBER OF PAGES: ix, 37

#### Lay Abstract

The advent of quantum computing poses a serious threat to modern cryptography, as most cryptosystems in use today are vulnerable to attacks by quantum algorithms. Recently proposed cryptosystems based on lattices are conjectured to be resistant to attacks by quantum computers. These cryptosystems also have a conditional security guarantee: if the cryptosystem can be broken by an attack, then a reduction exists which uses that attack to solve variants of the shortest vector problem (Regev, STOC 2005; Peikert, ePrint 2008). As these problems have no known efficient solutions, breaking the cryptosystem should be hard. However this guarantee only holds if the cryptosystem is constructed using parameters which satisfy conditions given in the reduction. Current proposals do not do this, and so cannot claim even a conditional security guarantee. We analyze two reductions and select parameters for a cryptosystem which satisfy these conditions. We also investigate the runtime necessary for a reduction to give meaningful security assurances for current cryptosystems.

#### Abstract

One attractive feature of lattice-based cryptosystems is the existence of security reductions relating the difficulty of breaking the cryptosystem to the difficulty of solving variants of the shortest vector problem (Regev, STOC 2005; Peikert, ePrint 2008). As there are no known polynomial-time algorithms which solve these lattice problems, this implies the asymptotic security of the cryptosystem. However, current lattice-based cryptosystems using the learning with errors (LWE) problem select parameters for which the reduction to the underlying lattice problem gives no meaningful assurance of concrete security. We analyze the runtime of the algorithm constructed in the reductions and select parameters for a cryptosystem under which the reductions give 128-bit security. While the resulting LWE-based cryptosystem is somewhat cumbersome, requiring a dimension of n = 1460, this is less than 2 times the dimension in the recently proposed Frodo cryptosystem (Bos et al., ACM CCS 2016), and could be implemented without catastrophic damage to communication times. We also investigate the runtime necessary for a reduction to give meaningful security assurances for current cryptosystems.

#### Acknowledgements

First and foremost I would like to extend my gratitude to Dr. Douglas Stebila. He has been a kind and generous mentor, and has given me a great deal of guidance over the past two years. I am delighted to have been able to work in a field that was entirely new to me and without him I would not have had that opportunity. While this thesis marks the end of our work together, I hope I will be a friend to him for many years to come.

I also wish to thank Dr. Eric Sawyer, who has taken me under his wing as a new PhD student. I am thoroughly enjoying our work together, and I am thrilled at the opportunity to continue studying at McMaster.

Lastly I want to thank my friends and family, whose support has been invaluable. In particular I would like to thank Brian Goncalves; he welcomed me to McMaster on my first day and within weeks was one of my closest friends. Without his companionship the past two years would have been far less fun.

# Contents

Li	st of	Abbreviations and Symbols	viii
Li	st of	Figures	ix
1	Intr	oduction	1
	1.1	History	2
	1.2	Outline	3
2	Bac	kground	4
	2.1	Cryptography	4
	2.2	Learning with errors	7
		2.2.1 Constructing cryptography from LWE	9
	2.3	Lattices	11
3	Ana	lysis of Regev's reduction	15
	3.1	Discussion	18
4	Ana	lysis of Peikert's reduction	20
	4.1	Discussion	24

5	Implementation results	<b>25</b>
	5.1 Parameter Selection from Regev's Reduction	27
	5.2 Parameter Selection from Peikert's Reduction	29
6	Future considerations	31
	6.1 Conclusions	34
Bi	ibliography	35

# List of Abbreviations and Symbols

LWE	Learning With Errors		
IND-CPA	Indistinguishability under Chosen Plaintext Attack		
IND-CCA	Indistinguishability under Chosen Ciphertext Attack		
LLL	Lenstra Lenstra Lovász		
BKZ	Block Korkine Zolotarev		
SVP	Shortest Vector Problem		
SIVP	Shortest Independent Vectors Problem		
GapSVP	Gap Shortest Vector Problem		
CVP	Closest Vector Problem		
GapCVP	Gap Closest Vector Problem		
DGS	Discrete Gaussian Sampling		

# List of Figures

3.0.1 Summary of Regev's SIVP $_{\gamma}$ to average-case decision LWE reduction	18
5.1.1 Sample parameters derived from Regev's reduction	28
6.0.1 Secure dimensions for an SVP to search LWE reduction with tightness gap $n^t$	32
6.0.2 Secure dimensions for an SVP to average-case decision LWE reduction with tight- ness gap $x^2yn^t$	33
6.0.3 Secure dimensions for an SVP to average-case decision LWE reduction with tight- ness gap $xyn^t$	33
6.0.4 Sample parameters derived from Regev's reduction with a factor of $x$ removed	34

### Chapter 1

## Introduction

Much of the world's digital infrastructure is dependent on cryptography that is both secure and highly efficient. While current implementations based on prime factorization [21] and discrete logarithms [8] perform this task admirably, they are vulnerable to Shor's quantum algorithm [22]. With large-scale quantum computers now on the horizon, it is vital that new quantum-secure cryptosystems are developed and standardized.

At present, lattices are one of the foremost primitives proposed for building quantum-secure cryptography. One attractive property of lattice-based cryptosystems is the existence of security reductions which relate the hardness of breaking a cryptosystem to that of solving some well-studied underlying problem, e.g. the shortest vector problem (SVP). These reductions—most famously Regev's in [20]—are non-tight and give only a conditional guarantee of security.

Informally: say we have an algorithm A with runtime run(A) that breaks a lattice-based cryptosystem, and an algorithm W for solving SVP using  $run_A(W)$  calls to A as a subroutine. If  $run(A) \cdot run_A(W)$  is less than the best known running time for an SVP-solver, then we have produced a superior algorithm for solving SVP. Defeating the cryptosystem is therefore at least as hard as finding such an algorithm.

However, this only gives a meaningful security guarantee if the condition on  $run(A) \cdot run_A(W)$ is met. Currently, most proposals for lattice-based cryptosystems use parameters which do not respect this inequality for any known reduction (see [5] for more discussion of non-tight security reductions and their significance). Chatterjee et al. [6] gave a brief analysis of the tightness gap in Regev's quantum reduction, but to our knowledge there has been no serious attempt to determine how large the gap is between a provably secure cryptosystem system and current implementations.

Our primary contribution is the selection of parameters for a lattice-based cryptosystem which respect the reduction given by Regev. This is done by producing a runtime analysis of the reduction and comparing against the best known runtimes for solving SVP. We discuss potential vulnerabilities of and improvements to our results, and investigate the necessary runtime of a reduction which could provide a security assurance for current lattice-based cryptosystems. We also analyze a more recent reduction by Peikert [18] and demonstrate why it is unsuitable for selecting concrete parameters.

#### 1.1 History

The use of lattices in cryptography dates back to work by Ajtai and Dwork [1], [2] in 1996. Although the resulting cryptosystems were unwieldy, needing public keys with size  $O(n^4)$  for a lattice of dimension n, they were the first whose security is provably implied by the difficulty of well-studied lattice problems. A significant improvement came from Regev in 2005 [20] where he presented the learning with errors problem (LWE) alongside a reduction of LWE to two variants of SVP and a public-key encryption system using LWE as a framework. This work was a dramatic improvement over the system of Ajtai and Dwork, needing public keys only of size O(n).

Further improvement came in 2010 when Lindner and Peikert [19] showed that keys could be sampled from a small distribution without compromising security. Also in 2010 Lyubashevsky, Peikert, and Regev [17] introduced the ring-LWE problem. This variant allows yet faster implementations by using lattices which contain additional structure, but it is so far unclear whether the additional structure could be leveraged by attackers. Lattices have also been used to instantiate richer forms of cryptography; in 2009 Gentry [11] gave a fully-homomorphic encryption system and in 2013 Garg et al. [10] extended this work to produce functional encryption. For a more detailed survey of lattice-based cryptography see [19]

### 1.2 Outline

Chapter 2 covers all the necessary background to read this work, including a brief introduction to cryptography. In chapter 3 we expand on Chatterjee et al.'s [6] analysis of Regev's reduction, then discuss potential vulnerabilities of and improvements to our result. Chapter 4 provides a similar analysis and discussion for Peikert's reduction. In chapter 5 we select a set of parameters which respect the reduction given in chapter 3, and demonstrate that Peikert's reduction does not yield practical parameters. Chapter 6 considers future avenues of improvement to these results and closes with a brief discussion of the gap between our results and current implementations.

### Chapter 2

### Background

As most relevant parameters are undefined at 0, we define  $\mathbb{N}$  to be the natural numbers excluding 0. We also denote by ln the natural logarithm and by log the base-2 logarithm. Vectors are written with bold lower-case (e.g.  $\mathbf{x}$ ) and matrices with bold upper-case letters (e.g.  $\mathbf{A}$ ). All lengths are measured under the Euclidian metric. We let  $B(\mathbf{x}, d)$  denote the set of points  $\mathbf{y} \in \mathbb{R}^n$ such that  $||\mathbf{x} - \mathbf{y}|| < d$ . A polynomial P parametrized by an integer n is said to be negligible if for any  $c \in \mathbb{N}$ , P(n) is eventually bounded above by  $\frac{1}{n^c}$ . Similarly P(n) is overwhelming if 1 - P(n)is negligible. Typically these notions will refer to probabilities. For functions f(n), g(n) we write  $f(n) = \tilde{O}(g(n))$  if  $f(n) = O(g(n) \cdot \log^c g(n))$  for some  $c \in \mathbb{R}$ . We say that an algorithm A is quantum if it can only be run on a quantum computer, and conversely we say that A is classical if it can be run on a classical computer.

### 2.1 Cryptography

An encryption system is a means by which two parties can communicate securely over an insecure channel. Encryption systems are broadly divided into two categories: symmetric encryption systems and asymmetric (or public-key) encryption systems. In a symmetric system the two parties are required to have previously agreed on some shared secret key. Common symmetric systems such as 3DES and AES offer a high degree of security relative to the amount of computation required, but cannot be used if the two parties do not have a shared secret. Public-key encryption systems require significantly more computation to transmit the same amount of data, but do not require a pre-established shared secret.

Public key encryption systems are in practice not often used to send long encrypted messages due to the relatively high amount of computation involved. Instead, public key encryption is primarily used to transmit a key so that highly efficient symmetric encryption algorithms can be used. Because of this, a full-fledged public key encryption system is often overkill. A key exchange protocol is a specialized set of public-key algorithms which, instead of encrypting arbitrary messages, allow two parties to efficiently agree on a shared secret key. The Diffie-Hellman protocol [8] is a straightforward example and is the foundation of many current standards. We will use the term "cryptosystem" as shorthand to refer to any of these constructions.

Formally, a public-key encryption system is a set of three algorithms (KeyGen, Enc, Dec) dependent on a security parameter n:

- Key generation (KeyGen): Given n, output a public key pk and corresponding secret key sk.
- 2. Encryption (Enc): Given a public key pk and message m, output a ciphertext c.
- 3. Decryption (Dec): Given a secret key sk and ciphertext c, output a message d.

For this to be useful we want the decryption of a ciphertext to return the original message. The system is therefore said to be *correct* if for pk and sk output by KeyGen, we have d = Dec(sk, Enc(pk, m)) = m with overwhelming probability.

A baseline notion of security for such a system is *indistiguishability under chosen plaintext attack* (IND-CPA), first introduced in [12]. Informally, suppose that an adversary chooses two messages and is then shown a ciphertext which is the encryption of one of the two messages. An attack is said to break indistinguishability if the adversary is able to determine which of two chosen messages corresponds to the given ciphertext. If it is infeasible for an adversary to do so, then the encryption algorithm leaks no information about any message it encrypts and is said to be IND-CPA secure. More formally, this definition uses the following experiment:

- 1. Generate a public key pk and secret key sk from KeyGen, and give pk to the adversary.
- 2. The adversary chooses any two messages  $m_0, m_1$  of the same length.
- Choose b ∈ {0,1} with equal probability and let c = Enc(m<sub>b</sub>). Give the resulting ciphertext c to the adversary.
- 4. The adversary guesses whether b = 0 or b = 1.

If the adversary makes a random guess then they succeed with probability 1/2. The *advantage* generated by an algorithm A used by an adversary is therefore defined as:

 $Adv(A) = \Pr[A \text{ succeeds in the IND-CPA experiment}] - 1/2.$ 

We also denote by run(A) the number of operations required to execute A. Similarly for an algorithm W using A as a subroutine, we denote by  $run_A(W)$  the number of calls made by W to A. This is the notion of security we will use, but we emphasize that IND-CPA security is a baseline and does not alone guarantee that a system is suitable for practical application. In particular many applications require security against active adversaries, modelled by the *indistinguishability* under chosen ciphertext attack (IND-CCA) security notion.

**Definition 1** (Security). Let  $t, k \in \mathbb{N}$  and  $\epsilon \in [0, 1]$ . A cryptosystem C is  $(t, \epsilon)$  secure if for all algorithms A with  $run(A) \leq t$ ,  $Adv(A) \leq \epsilon$ . C is said to have k-bit security if it is  $(t, \epsilon)$  secure for all  $\frac{t}{\epsilon} \leq 2^k$ .

It will also be useful to consider the negation of the above definition: a cryptosystem C is *k*-bit *insecure* if there exists an A and  $\frac{t}{\epsilon} \leq 2^k$  such that run(A) < t and  $Adv(A) \geq \epsilon$ . The current standard for security is 128 bits, as  $2^{128}$  operations is still considered infeasible even for large networks. An increase to 256-bit security in the near future may be needed if quantum computers capable of executing Grover's algorithm [13] are developed.

It has historically been difficult to prove that an encryption system actually attains a desired level of security. In some cases however, we can relate the difficulty of breaking indistinguishability to the difficulty of solving some other well-known problem. Such a relation is called a *security*  reduction, and offers a guarantee of security given that the related problem remains difficult to solve efficiently. A reduction is given as an algorithm W for solving the related problem which uses as a subroutine a hypothetical algorithm A capable of breaking distinguishability of the cryptosystem. The reduction is said to be *tight* if  $run_A(W) = 1$ , and otherwise is said to be *non-tight*. These notions are formalized as follows:

**Definition 2** (Reduction). Let A be an algorithm which solves some problem  $P_1$  (with overwhelming success rate). We say problem  $P_2$  reduces to  $P_1$  if there exists an algorithm W which solves  $P_2$  with overwhelming success rate using A as an oracle.

**Definition 3** (Tightness gap). Let  $P_1$  and  $P_2$  be problems, A be an algorithm which solves  $P_1$ , and W be an algorithm that reduces  $P_2$  to  $P_1$ . We say that W has tightness gap k if  $run_A(W) = k$ .

We note in the above definitions that the algorithms are assumed to succeed essentially always and that the tightness gap is purely a function of runtime. A more nuanced view of tightness gap can be taken as in  $(t, \epsilon)$ -security, where the reduction of  $P_2$  to  $P_1$  causes potential degredation in both running time and success probability, but quantifying this adds significant complication to the analysis. This however presents us with a difficulty: some of the algorithms we analyze here have negligible but non-zero failure rates. When an algorithm's failure rate can only be made arbitrarily small, in the abstract there is no reason to prefer one small failure rate over another. We chose  $2^{-32}$  to be an acceptable failure rate in these cases, but this choice is essentially arbitrary. If a practical application requires a particular failure rate, the appropriate calculations are easily modified. Moreover, because we are considering algorithms with negligible failure rates, even a drastic reduction (say to  $2^{-256}$ ) ultimately produces only a small change in overall runtime.

#### 2.2 Learning with errors

Cryptographic assumptions—difficult mathematical problems such as factorization which underlie cryptosystems—can be loosely thought of as problems which obscure information. Consider the following toy problem: given that 2x = 42, compute x. With only one unknown, this is trivially solved. Now consider the following modification: given that  $2x = 41 \pm 1$ , compute x. The addition of a small error term makes it impossible to determine with certainty what the correct value of x should be. The *learning with errors* problem, first formalized by Regev [20], is a more sophisticated application of this idea.

**Definition 4** (Learning with errors). Let  $n, m, q \in \mathbb{N}$  and  $\chi$  be a probability distribution on  $\mathbb{Z}_q^m$ . Generate  $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ ,  $\mathbf{s} \in \mathbb{Z}_q^n$  uniformly at random, and  $\mathbf{e} \in \mathbb{Z}_q^m$  sampled from  $\chi$ . Define  $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ . The goal of (search) LWE is, given  $(\mathbf{A}, \mathbf{b})$ , recover  $\mathbf{s}$ . The goal of (decision) DLWE is, given  $(\mathbf{A}, \mathbf{v})$ , determine whether  $\mathbf{v} = \mathbf{b}$  or  $\mathbf{v}$  was sampled uniformly at random from  $\mathbb{Z}_q^m$ .

In the case of decision LWE, an algorithm is said to *accept* an input if it guesses that the input was generated as  $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ . For fixed  $\mathbf{A}$ ,  $\mathbf{s}$  these inputs are said to be generated from the *LWE distribution*, where the randomness is over the sampling of  $\mathbf{e}$ . Similarly an algorithm is said to *reject* if it guesses that the input was generated from the uniform distribution on  $\mathbb{Z}_q^m$ . We then define the avantage against decision LWE analogously to advantage in the IND-CPA experiment, and we also define  $(t, \epsilon)$  hardness and k-bit hardness analogously to  $(t, \epsilon)$  security and k-bit security.

In principle  $\chi$  can be any distribution, but for cryptographic purposes we want the error term sampled from  $\chi$  to be small relative to q. As seen in the toy example even a small error term is sufficient to obscure s, but error terms too large relative to q make it difficult to reliably recover the correct message during decryption. The standard choice of distribution is the *discrete Gaussian distribution*.

**Definition 5** (Discrete Gaussian distribution). Let  $G_{\alpha}$  be a Gaussian distribution with mean 0 and standard deviation  $\sigma_G = \frac{\alpha}{\sqrt{2\pi}}$ . Let  $N_{\alpha}$  be the probability distribution defined on the interval [0,1) given by sampling from  $G_{\alpha}$  and reducing modulo 1. Then the discrete Gaussian distribution  $D_{\alpha}$  defined on  $\mathbb{Z}_q$  is obtained by sampling from  $N_{\alpha}$ , multiplying by q, and rounding to the nearest integer modulo q. Also let  $D_{\alpha}^n$  denote the distribution on  $\mathbb{Z}_q^n$  obtained by individually sampling each coordinate from  $D_{\alpha}$ .

More recently, Applebaum et al. [3] showed that  $\mathbf{s}$  can be sampled from  $D^n_{\alpha}$  (or more generally from  $\chi$ ) without compromising security. Since recovering  $\mathbf{e}$  is sufficient to recover  $\mathbf{s}$ , the intuition is that additional variance in the sampling of  $\mathbf{s}$  beyond that in the sampling of  $\mathbf{e}$  should not contribute additional security. To avoid a potential source of confusion, notice that this formulation of LWE differs slightly from the one originally given by Regev. In [20, Section 2], the problem is given in terms of vectors **a** and scalars  $e, b = \mathbf{a} \cdot \mathbf{s} + e$ . The challenge then is to recover **s** (or to distinguish *b* from random) using an *arbitrary* number of samples from the resulting distribution, where in our formulation only a fixed number *m* of samples are given. For our purposes the fixed choice of *m* is appropriate; in a cryptosystem using LWE *m* will be determined by the size of a public key and an attacker will not be able to request an arbitrary number of samples.

#### 2.2.1 Constructing cryptography from LWE

Here we give two examples demonstrating how cryptosystems can be constructed from the LWE problem. The first is a public-key encryption system given by Regev in [20, Section 5]. The second is a simple implementation of Frodo, the key exchange protocol given in [4] by Bos et al. We emphasize that these are primitive examples; practical implementations would contain many additional optimizations to deliver improved runtime and additional security features.

Let  $n, q \in \mathbb{N}$  where q is prime and  $n^2 < q < 2n^2$ . Also let  $D^n_{\alpha}$  be the discrete Gaussian distribution with  $\alpha = 1/(\sqrt{n}\log^2 n)$ . These parameters define the cryptosystem and are public knowledge. To have a rough idea of scale assume  $n \approx 1000$ ; the exact choice of n would be determined by the desired security level. The system is as follows:

- 1. Alice chooses  $\mathbf{s} \in \mathbb{Z}_q^n$ ,  $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$  uniformly at random and samples  $\mathbf{e} \in \mathbb{Z}_q^n$  from  $D_{\alpha}^n$ . Let  $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ . Alice's secret key is  $\mathbf{s}$ , and her public key is  $(\mathbf{A}, \mathbf{b})$ .
- 2. Bob chooses  $\mathbf{x} \in \mathbb{Z}_2^n$  uniformly at random. To encrypt the bit 0, Bob sends  $(\mathbf{x}^T \mathbf{A}, \mathbf{x} \cdot \mathbf{b})$ . To encrypt the bit 1, Bob sends  $(\mathbf{x}^T \mathbf{A}, \mathbf{x} \cdot \mathbf{b} + \lfloor \frac{q}{2} \rfloor)$ .
- 3. Alice receives  $(\mathbf{a}, b)$  and computes  $d = b \mathbf{a} \cdot \mathbf{s}$ . The decryption of  $(\mathbf{a}, b)$  is 0 if d is closer to 0 than to  $\lfloor \frac{q}{2} \rfloor$  modulo q, and otherwise the decryption is 1.

In the decryption algorithm we have  $\mathbf{a} = \mathbf{x}^T \mathbf{A}$  and  $b = \mathbf{x}^T (\mathbf{A} \cdot \mathbf{s} + \mathbf{e}) + \lfloor \frac{q}{2} \rfloor \cdot m$  for a message bit  $m \in \{0, 1\}$ . Therefore  $d = b - \mathbf{a} \cdot \mathbf{s} = \mathbf{e} + \lfloor \frac{q}{2} \rfloor \cdot m \approx \lfloor \frac{q}{2} \rfloor \cdot m$ , so the decryption algorithm is correct with high probability as the error distribution is small relative to q. The security of this system follows from the hardness of LWE. If search LWE is difficult then an adversary cannot recover Alice's secret key **s** from her public key  $(\mathbf{A}, \mathbf{b})$ . Similarly, an adversary who can solve decision LWE can trivially break indistinguishability as encryptions of 0 are close to the LWE distribution while encryptions of 1 are not.

We note that there is a wrinkle here: if the adversary can determine the binary vector  $\mathbf{x}$  then they can trivially determine whether the encrypted bit is 0 or 1. The security of this system therefore also relies on the difficulty of the subset sum problem rather than following immediately from the hardness of decision LWE (see [9] for a discussion of the subset sum problem in cryptography). This is a quirk of this particular implementation and is not a general feature of lattice-based cryptosystems. Regev's system is still IND-CPA secure; see [20, Section 5] for the full proof. For comparison we also give a slightly simplified version of the Frodo key exchange protocol, based on the Lindner-Peikert cryptosystem [16].

Let  $n, q, \bar{n}, \bar{m} \in \mathbb{N}$  and  $D^n_{\alpha}$  be the discrete Gaussian distribution. Here  $n, q, D^n_{\alpha}$  are LWE parameters as before, and  $\bar{m}, \bar{n}$  are chosen so that  $\bar{m} \cdot \bar{n}$  is at least the number of bits required for the shared secret key. Again the exact parameters depend on the desired security level, but for sake of example take  $n = 750, q = 2^{15}, \alpha = 1.75$ .

- 1. Alice generates  $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$  uniformly at random and  $\mathbf{S}, \mathbf{E} \in \mathbb{Z}_q^{n \times \bar{n}}$  sampled from  $D_{\alpha}^n$ . Alice also computes  $\mathbf{B} = \mathbf{A} \cdot \mathbf{S} + \mathbf{E}$  and sends  $(\mathbf{A}, \mathbf{B})$  to Bob.
- 2. Bob generates  $\mathbf{S}', \mathbf{E}' \in \mathbb{Z}_q^{\bar{m} \times n}$  sampled from  $D_{\alpha}^n$  and computes  $\mathbf{B}' = \mathbf{S}' \cdot \mathbf{A} + \mathbf{E}'$ . Bob also samples  $\mathbf{E}'' \in \mathbb{Z}_q^{\bar{m} \times \bar{n}}$  from  $D_{\alpha}^n$  and computes  $\mathbf{V} = \mathbf{S}' \cdot \mathbf{B} + \mathbf{E}''$ . Lastly Bob computes  $\mathbf{C} = \langle \mathbf{V} \rangle$ and sends  $(\mathbf{B}', \mathbf{C})$  to Alice.
- 3. Alice computes  $r(\mathbf{B}' \cdot \mathbf{S}, \mathbf{C}) = \mathbf{K}$  and Bob computes  $|\mathbf{V}| = \mathbf{K}$  to arrive at a shared secret.

The matrices  $\mathbf{B}' \cdot \mathbf{S}$  computed by Alice and  $\mathbf{V}$  computed by Bob are approximately equal modulo q, but not identical. The rounding functions  $\langle \cdot \rangle$ ,  $\lfloor \cdot \rceil$  run by Bob and the reconciliation function r run by Alice allow both parties to extract the same key  $\mathbf{K}$ . We will not present the details of these functions here, but they can be found in [4].

#### 2.3 Lattices

An *n*-dimensional *lattice* L is the set of all integer linear combinations of a set of n linearly independent vectors in  $\mathbb{R}^n$ . L is therefore also a discrete additive subgroup of  $\mathbb{R}^n$ . The prototypical example of a lattice is  $\mathbb{Z}^n \subset \mathbb{R}^n$ . Lattices can be fully described by a basis set B, and as with real vector spaces this basis representation is not unique. The *minimum distance* of L, denoted  $\lambda_1(L)$ , is the shortest nonzero length of a vector in L. Correspondingly  $\lambda_n(L)$  is the shortest length of a set of n independent vectors in L, where the length of a set of vectors is the length of the longest vector in the set. Finding these lengths, and the vectors they represent, constitute the majority of standard lattice problems.

It is important to note that a given basis is not guaranteed to contain a vector of length  $\lambda_1(L)$ . As an example, consider the lattice in  $L \subset \mathbb{R}^2$  given by the basis  $\mathbf{B} = \{(10, 0), (10, 1)\}$ . This gives  $\lambda_1(L) = 1$  but the shortest basis vector has length 10. These highly non-orthogonal bases make lattices difficult to work with in high dimension and are the source of most cryptographically useful lattice problems. The LLL algorithm [14] (and more recently the BKZ algorithm [7]) can find a reasonably orthogonal basis for a given lattice in polynomial time, but these algorithms are slow to run in high dimensions and are not guaranteed to produce a sufficiently short basis to make lattice problems efficiently solvable.

The dual of an *n*-dimensional lattice L, denoted  $L^*$ , is defined to be the set of all  $\mathbf{y} \in \mathbb{R}^n$ such that  $\mathbf{x} \cdot \mathbf{y} \in \mathbb{Z}$  for all  $\mathbf{x} \in L$ . Notice that this set is discrete and closed under addition, so  $L^*$ is also a lattice. If  $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$  is a basis for L, the dual basis of  $L^*$  is the set  $\{\mathbf{x}_1^*, ..., \mathbf{x}_n^*\}$  where  $\mathbf{x}_i \cdot \mathbf{x}_i^* = 1$  and  $\mathbf{x}_i \cdot \mathbf{x}_j^* = 0$  for  $i \neq j$ .

**Definition 6** (Smoothing parameter). Let L be an n-dimensional lattice and  $\epsilon > 0$  be a real number. The smoothing parameter  $\eta_{\epsilon}(L)$  is the smallest  $s \in \mathbb{R}_+$  such that

$$\sum_{\mathbf{x} \in L^* \setminus \{0\}} \exp(-\pi ||s\mathbf{x}||^2) \le \epsilon.$$

Informally,  $\eta_{\epsilon}(L)$  measures the necessary width of a discrete Gaussian distribution on L so that the distribution behaves like a continuous Gaussian distribution to within an approximation factor  $\epsilon$ . We do not make extensive use of dual lattices or the smoothing parameter, but they are used in Regev's reduction and so will be needed for a complete description. A useful result is the following bound on  $\eta_{\epsilon}$ , which ensures that  $\eta_{\epsilon}$  is roughly proportional to  $\lambda_n(L)$ .

**Lemma 1** ([20], Lemma 2.12, Claim 2.13). For any n-dimensional lattice L and any  $\epsilon > 0$ ,

$$\sqrt{\frac{\ln 1/\epsilon}{\pi}} \cdot \frac{\lambda_n(L)}{n} \le \eta_\epsilon \le \sqrt{\frac{\ln \left(2n(1+1/\epsilon)\right)}{n}} \cdot \lambda_n(L)$$

Regev [20] (and more recently Peikert [18]) showed that the existence of an algorithm which solves LWE implies the existence of algorithms which solve hard lattice problems. This makes LWE attractive for cryptography; lattice problems are well-studied, so one can hope that they are unlikely to admit major improvements in the foreseeable future. While this does not constitute an absolute guarantee of security, cryptosystems based on LWE enjoy a more solid theoretical foundation than many other potential constructions.

Here we present some of the standard lattice problems, all of which are conjectured to be hard. There are both exact and approximate versions for some problems; in the approximation problems,  $\gamma \in \mathbb{R}_+$  is the approximation factor. Ultimately, our goal is to relate the difficulty of solving these problems to the difficulty of solving LWE.

**Definition 7** (Shortest vector problem). Let L be a lattice. The goal of SVP is to output a vector in L with length  $\lambda_1(L)$ . The goal of  $SVP_{\gamma}$  is to output a vector in L with length at most  $\gamma \cdot \lambda_1(L)$ .

**Definition 8** (Shortest independent vectors problem). Let L be a lattice. The goal of SIVP is to output a set of n independent vectors in L with length at most  $\lambda_n(L)$ , where n is the dimension of L. The goal of SIVP<sub> $\gamma$ </sub> is to output a set of n independent vectors in L with length at most  $\gamma \cdot \lambda_n(L)$ .

**Definition 9** (Gap shortest vector problem). Let L be a lattice and d > 0. The goal of GapSVP is to determine whether  $\lambda_1(L) \leq d$  or  $\lambda_1(L) > d$ . The goal of GapSVP $\gamma$  is to determine whether  $\lambda_1(L) \leq d$  or  $\lambda_1(L) > \gamma \cdot d$ 

**Definition 10** ( $\zeta$ -to- $\gamma$  Gap shortest vector problem). Let L be a lattice given by a basis  $\mathbf{B}$  where every Gram-Schmidt vector  $\widetilde{\mathbf{b}}_i$  has length at least 1. Also let  $d \ge 1$  and  $\zeta \ge \gamma \ge 1$  with  $\gamma \cdot d \le \zeta$ . The goal of  $GapSVP_{\zeta,\gamma}$  is to determine whether  $1 \le \lambda_1(L) \le d$  or  $\gamma \cdot d < \lambda_1(L) \le \zeta$ . This variant of GapSVP was introduced as part of Peikert's classical reduction [18]. For  $\zeta > 2^{n/2}$ , GapSVP<sub> $\zeta,\gamma$ </sub> is equivalent to the more standard GapSVP<sub> $\gamma$ </sub>. For smaller (polynomial) values of  $\zeta$  the additional promise that  $\lambda_1(L) \leq \zeta$  makes the problem potentially easier to solve. The best known algorithms are however unable to leverage this information, and still have exponential running time even for very small  $\zeta$ .

**Definition 11** (Closest vector problem). Let *L* be a lattice and  $\mathbf{x} \notin L$  be a vector in  $\mathbb{R}^n$ . The goal of CVP is to output a vector  $\mathbf{y} \in L$  such that  $||\mathbf{x} - \mathbf{y}|| \leq ||\mathbf{x} - \mathbf{z}||$  for all  $\mathbf{z} \in L$ .

We also use the notation  $\text{CVP}_{\phi}$  to refer to a CVP instance  $(L, \mathbf{x})$  with the additional promise that there exists  $\mathbf{y} \in L$  with  $||\mathbf{x} - \mathbf{y}|| < \phi$ . This is a less common formulation, but is used in the iterative step of Regev's reduction. Note that a CVP instance  $(L, \mathbf{x})$  which does not satisfy the above condition may still be given as input to an algorithm which solves  $\text{CVP}_{\phi}$ , but the algorithm will not necessarily return the true closest vector.

**Definition 12** (Gap closest vector problem). Let L be a lattice,  $\mathbf{x} \notin L$  be a vector in  $\mathbb{R}^n$ , and d > 0. The goal of GapCVP is to determine whether there exists a vector  $\mathbf{y} \in L$  such that  $||\mathbf{x} - \mathbf{y}|| \leq d$ . The goal of GapCVP<sub> $\gamma$ </sub> is to determine whether there exists a vector  $\mathbf{y} \in L$  such that that  $||\mathbf{x} - \mathbf{y}|| \leq d$  or whether  $||\mathbf{x} - \mathbf{y}|| > \gamma \cdot d$  for all  $\mathbf{y} \in L$ .

Regev also gives a modified version of  $\operatorname{GapCVP}_{\gamma}$ , denoted  $\operatorname{GapCVP}_{\gamma}'$ , which additionally promises that if  $||\mathbf{x} - \mathbf{y}|| > \gamma \cdot d$  for all  $\mathbf{y} \in L$  then also  $\lambda_1(L) > \gamma \cdot d$ .

In 2.2 we defined the discrete Gaussian distribution  $D_{\alpha}^{n}$  over the lattice  $\mathbb{Z}_{q}^{n}$ . The discrete Gaussian distribution can also be defined more generally as  $D_{L,r}$  over an arbitrary lattice for some width parameter r; this is the definition used in the iterative step in Regev's quantum reduction. Informally, if  $\mathbf{x} \in L$  is a lattice point and  $G_r$  is the probability density function of a continuous Gaussian distribution with standard deviation r, then  $D_{L,r}$  samples  $\mathbf{x}$  with probability proportional to  $G_r(\mathbf{x})$ . We also note that r is close to but not exactly the standard deviation of  $D_{L,r}$ , but for visualization purposes the reader can think of the lattice as being  $\mathbb{Z}_{q}^{n}$  and the width parameter r as being a standard deviation. The details of the more general definition are not necessary to understand this work but can be found in [20, Section 2].

**Definition 13** (Discrete Gaussian sampling). Let L be a lattice and r > 0 be a width parameter. The goal of  $DGS_r$  is to output samples  $\mathbf{x} \in L$  which are distributed according to  $D_{L,r}$ . Note that if one can efficiently generate samples from  $D_{L,r}$  for arbitrary r then one can also trivially solve SVP by choosing successively smaller values for r and generating samples from each  $D_{L,r}$ . Eventually r will be sufficiently small that  $D_{L,r}$  assigns all its weight to the origin, so once **0** has been sampled many times consecutively one can halt and output the smallest non-zero vector generated by the sampling algorithm. With high probability this vector will have length  $\lambda_1(L)$ . Other standard lattice problems can be solved similarly, so solving DGS<sub>r</sub> for small r is at least as hard as other lattice problems.

### Chapter 3

### Analysis of Regev's reduction

In [20, Sections 3, 4], Regev gives a chain of reductions from  $\text{SIVP}_{\gamma}$  to decision LWE, with each reduction invoking some polynomial number of calls to the algorithm output by the previous reduction. Chatterjee et al. [6] give a brief analysis of Regev's reduction, which we expand upon here. As the DGS<sub>r</sub> to LWE reduction is extremely technical we only give a brief overview of its structure here. All other reductions are presented fully.

**Theorem 1.** Let  $n, m \in \mathbb{N}$ ,  $q \in \mathbb{N}$  be prime,  $\alpha \in (0,1)$  with  $\alpha q > \sqrt{n}$ , and  $x, y \in \mathbb{R}^+$  with  $x \ge 2$  and  $y \ge 1$ . Also let  $D = D_{\alpha}$  be the discrete Gaussian distribution on  $\mathbb{Z}_q$  centered around 0 with standard deviation  $\sigma = \alpha q/\sqrt{2\pi}$ . Let W be an algorithm for solving decision LWE given m LWE samples and with errors sampled from D. Suppose that on some proportion 1/y of all possible keys  $\mathbf{s} \in \mathbb{Z}_q^n$ , W accepts with probabilities  $a_{LWE}$ ,  $a_U \in [0,1]$  on inputs from the LWE and uniform distributions respectively, and that these acceptance rates differ by at least 1/x. Then there exists a quantum algorithm W<sup>\*</sup> which solves  $SIVP_{\gamma}$  for  $\gamma = \tilde{O}(n/\alpha)$  with tightness gap  $276x^2yn^8mq\log(\alpha q/\sqrt{n})^{-1}$ .

*Proof.* We consider the following sequence of reductions:

Worst-case to average-case decision LWE. Regev gives the following: for  $t \in \mathbb{Z}_q^n$ , define the function  $f_{\mathbf{t}} : \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m \to \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$  by  $f_{\mathbf{t}}(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{b} + \mathbf{A} \cdot \mathbf{t})$ . Repeat the following process ny times: choose  $\mathbf{t} \in \mathbb{Z}_q^n$  uniformly at random, then estimate the acceptance probability of W on the uniform distribution and on the modified input data  $f_{\mathbf{t}}(\mathbf{A}, \mathbf{v})$  using the Chernoff bound. This estimate requires calling  $W O(nx^2)$  times on each distribution. If the two estimates differ by at least 1/2x then halt and return *accept*, otherwise continue. If W does not accept in any of the ny repetitions, then return *reject*.

Notice that if  $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ , then  $f_{\mathbf{t}}(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \cdot (\mathbf{s} + \mathbf{t}) + \mathbf{e})$  is still an LWE sample but using the new key  $\mathbf{s} + \mathbf{t}$ . The choice of ny repetitions is to ensure that with overwhelming probability at least one of the uniformly random translated keys  $\mathbf{s} + \mathbf{t}$  is in the 1/y portion of "good" keys on which W achieves meaningful advantage. In fact ny is needlessly large; with ky repetitions, the probability of never finding a good key is  $(\frac{y-1}{y})^{ky} < 1/e^k$ . Choosing k = 23 already yields a failure rate less than  $2^{-32}$ , which we consider sufficient for practical purposes.

Lastly we would like to evaluate the  $O(nx^2)$  term. We need to take enough samples to estimate both acceptance probabilities to within 1/8x of their true values. The (additive) Chernoff bound allows us to estimate the likelihood that a given number of samples produce an estimate that is *not* within the desired margin.

**Lemma 2** (Chernoff bound). Suppose  $X_1, ..., X_k$  are independent and identically distributed random variables taking values in 0, 1. If  $\mu = E[X_i]$  and  $\epsilon > 0$  then

$$Pr\left(\frac{1}{k}\sum X_i \ge \mu + \epsilon\right) \le \left(\left(\frac{\mu}{\mu + \epsilon}\right)^{\mu + \epsilon} \left(\frac{1 - \mu}{1 - \mu - \epsilon}\right)^{1 - \mu - \epsilon}\right)^k.$$

So we want to choose k large enough that this probability is negligible for  $\mu = a_U$  and  $\epsilon = 1/8x$ . While the inequality is difficult to evaluate algebraically, Regev gives  $O(nx^2)$  as a sufficient sample size. In fact testing a range of reasonable values for  $a_U \in [0, 1]$  and  $x \ge 2$  shows that in each case taking  $k = nx^2$  samples of each distribution is sufficient to achieve a failure rate less than  $2^{-32}$ , which we consider sufficient. The tightness gap for this reduction is therefore  $23y \cdot 2 \cdot nx^2 = 46nx^2y$ ; let  $W_1$  denote the resulting algorithm.

Search LWE to worst-case decision LWE. Let  $(\mathbf{A}, \mathbf{b})$  be an LWE instance, and fix some  $k \in \mathbb{Z}_q$  and choose  $l \in \mathbb{Z}_q$  uniformly at random. For a single LWE sample  $(\mathbf{a}, b)$ , consider the transformation  $g_{k,l} : \mathbb{Z}_q^n \times \mathbb{Z}_q \to \mathbb{Z}_q^n \times \mathbb{Z}_q$  by  $g_{k,l}(\mathbf{a}, b) = (\mathbf{a} + (l, 0, ..., 0), b + k \cdot l)$ . If  $k = s_1$ , the first coordinate of  $\mathbf{s}$ , then  $g_{k,l}(\mathbf{a}, b)$  is also an LWE sample using the key  $\mathbf{s}$ . Otherwise, if  $k \neq s_1$  then  $k \cdot l$  is uniform on  $\mathbb{Z}_a$  (since  $\mathbb{Z}_q$  is a field) and therefore  $b + k \cdot l$  is also uniform. By running  $W_1$  on each of  $g_{0,l}, ..., g_{q-1,l}$ , we can determine the correct value of  $s_1$ . Repeating this procedure

on each coordinate therefore recovers s with a tightness gap of at most nq; let this algorithm be  $W_2$ . We also note that this is the only reduction which requires q to be prime.

Continuous search LWE to discrete search LWE. Given  $W_2$  which solves LWE with errors sampled from D we can also solve LWE with errors are sampled from  $N_{\alpha}$ , the equivalent continuous Gaussian distribution reduced modulo 1. Let  $(\mathbf{A}, \mathbf{b})$  be such an LWE instance. We can produce an LWE instance  $(\mathbf{A}, \mathbf{b}')$  with errors from D by taking the fractional part of each coordinate of  $\mathbf{b}$ , multiplying by q and rounding to the nearest integer. Since this only changes the error term, we can recover  $\mathbf{s}$  for  $(\mathbf{A}, \mathbf{b})$  by running  $W_2$  on  $(\mathbf{A}, \mathbf{b}')$ . This requires only a single call so the reduction is tight; let this algorithm be  $W_3$ .

**DGS**<sub>r</sub> to **LWE.** From  $W_3$ , [20, Thm 3.1] gives an iterative quantum algorithm for solving  $DGS_r$ , where  $r = \sqrt{2n} \cdot \eta_{\epsilon}(L)/\alpha$  and L is any *n*-dimensional lattice. Let  $r_i = r \cdot (\alpha q/\sqrt{n})^i$ . For  $r_i$  sufficiently large, [20, Lemma 3.2] shows how to efficiently sample from  $D_{L,r_i}$ . Then given m samples from  $D_{L,r_i}$ , the iterative step produces m samples from the smaller distibution  $D_{L,r_{i-1}}$ . The algorithm repeats until we have produced samples from  $D_{L,r_0} = D_{L,r}$ .

Regev gives the condition  $\alpha q > 2\sqrt{n}$ , under which the iterative part of the reduction runs 3n times to produce a sufficiently small sample. However the lower bound on  $\alpha q$  can be relaxed; the algorithm still produces successively smaller samples as long as  $\alpha q > \sqrt{n}$ . This will allow us to choose a smaller value for  $\sigma$  at the cost of needing an increasingly large number of iterations as this relation approaches equality. With the relaxed condition, we iterate  $3n \cdot \log (\alpha q/\sqrt{n})^{-1}$  times to produce one sample.

The iterative step of the reduction contains two parts. Given samples from  $D_{L,r_i}$ , Regev constructs an algorithm which solves  $\text{CVP}_{\alpha q/(\sqrt{2}r_i)}$  on  $L^*$  with tightness gap  $n^2$  [20, Lemma 3.4]. Then given a  $\text{CVP}_{\alpha q/(\sqrt{2}r_i)}$  solver for  $L^*$  Regev produces a *quantum* algorithm to produce samples from  $D_{L,r_i\sqrt{n}/(\alpha q)} = D_{L,r_{i-1}}$ , and this reduction is tight [20, Lemma 3.14]. To produce the required *m* samples, this reduction then has total tightness gap  $3n^3m \cdot \log(\alpha q/\sqrt{n})^{-1}$ ; let this algorithm be  $W_4$ .

SIVP<sub> $\gamma$ </sub> to DGS<sub>r</sub>. Let *L* be a lattice. Applying the LLL algorithm [14] to *L* yields a set of linearly independent basis vectors of length at most  $2^n \lambda_n(L)$ ; let  $\tilde{\lambda}_n$  be the length of the longest

Reduction	Tightness gap	Parameters	Restrictions
Worst-case to average-case dLWE	$46nx^2y$	$n,q,\sigma = \alpha q/\sqrt{2\pi}$	none
Search to decision LWE	nq	$n,q,\sigma=\alpha q/\sqrt{2\pi}$	q prime
Continuous to discrete LWE	1	$n,q,\sigma = \alpha q/\sqrt{2\pi}$	none
$\mathrm{DGS}_r$ to LWE	$3n^3m\log\left(\alpha q/\sqrt{n}\right)^{-1}$	$r = \sqrt{2n} \cdot \eta_{\epsilon}(L) / \alpha$	$\alpha q > \sqrt{n}$
$\mathrm{SIVP}_{\gamma}$ to $\mathrm{DGS}_r$	$2n^3$	$\gamma = \tilde{O}(n/\alpha)$	none

Figure 3.0.1: Summary of Regev's SIVP $_{\gamma}$  to average-case decision LWE reduction

resulting basis vector. Now for  $i \in \{0, ..., 2n\}$  call  $W_4$   $n^2$  times with input  $(L, r_i)$  for  $r_i = \tilde{\lambda}_n 2^{-i}$ and let  $S_i$  be the resulting set of vectors. By [20, Corollary 3.16] each  $S_i$  contains a set of nindependent vectors with overwhelming probability and by [20, Lemma 2.12] at least one of the sets  $S_i$  contains a basis set with length at most  $\tilde{O}(n/\alpha) \cdot \lambda_n(L)$ . The resulting algorithm  $W^*$ therefore solves SIVP<sub> $\gamma$ </sub> for  $\gamma = \tilde{O}(n/\alpha)$  and the construction of  $W^*$  from  $W_4$  has tightness gap  $2n^3$ .

By chaining these reductions we can construct  $W^*$  using  $46nx^2y \cdot nq \cdot 1 \cdot 3n^3m \log (\alpha q/\sqrt{n})^{-1} \cdot 2n^3$ calls to W. Therefore the tightness gap of Regev's reduction is  $276x^2yn^8mq \log (\alpha q/\sqrt{n})^{-1}$ .  $\Box$ 

The tightness gap, relevant parameters, and any additional restrictions in each reduction are summarized in figure 3.0.1.

#### 3.1 Discussion

With the above analysis, we can now compare the tightness of the reduction with the best known runtimes for solving underlying lattice problems to achieve the desired security guarantee. However this presents a problem; while the basic lattice problems (SVP and CVP) are wellstudied and have known runtime estimates, the same is not true for some of the more obscure lattice problems. In particular, Regev's reduction uses  $SIVP_{\gamma}$  as its underlying lattice problem. As solving SIVP trivially also solves SVP, any SIVP solver will have runtime at least as bad as the best known SVP solvers. Consequently we can use the known runtimes for SVP, and at worst we will have overestimated the necessary dimension. More worryingly, the approximation problem  $\text{SIVP}_{\gamma}$  is potentially easier to solve than  $\text{SIVP}_{\gamma}$ To the best of our knowledge the only known efficient algorithms for solving  $\text{SIVP}_{\gamma}$  require an exponential approximation factor while Regev's reduction only gives  $\gamma = \tilde{O}(n/\alpha) \approx \tilde{O}(n^{1.5})$ , so we might hope that no algorithm exists to abuse the weakening of the underlying problem. None the less, this presents a vulnerability in our result. Further runtime analysis of algorithms solving  $\text{SIVP}_{\gamma}$  would allow for a more accurate computation and could potentially leverage the greater difficulty of SIVP to achieve smaller parameters.

Regev also gives a reduction of DGS to GapSVP $_{\gamma}$ . This could yield a potentially better result, but presents two additional problems. While we believe there is no known GapSVP $_{\gamma}$  solution better than solving the equivalent SVP problem, GapSVP $_{\gamma}$  is a potentially easier problem and so using SVP runtimes in our analysis would yield an underestimate of the required parameters rather than an overestimate. The reduction to GapSVP $_{\gamma}$  is also less straightforward than the reduction to SIVP $_{\gamma}$ ; Regev actually gives a reduction to GapCVP' $_{\gamma}$  using a polynomial number of calls to a DGS oracle, then invokes a known polynomial-time reduction from GapSVP $_{\gamma}$  to GapCVP' $_{\gamma}$ . Using this result would therefore require computing the exact number of calls and the runtime of the GapSVP $_{\gamma}$  to GapCVP' $_{\gamma}$  reduction.

Another potential area for improvement lies in the worst-case to average-case decision LWE reduction. Regev's formulation allows an adversary to request an arbitrary number of LWE samples, and the reduction requires the adversary to have at least  $nx^2$  samples to make a sufficiently accurate estimate. In practice an attacker would only have access to the m samples provided in a public key  $(\mathbf{A}, \mathbf{b})$ , and  $nx^2 \gg m$  unless W achieves a very large advantage. Since Regev's reduction guarantees security even with a much larger number of samples revealed, implementations with  $m \approx n$  will likely have more security than guaranteed by the reduction. Quantifying this gap might allow for the selection of smaller parameters.

### Chapter 4

### Analysis of Peikert's reduction

One clear limitation of Regev's reduction is that it is inherently quantum, and so cannot offer security guarantees based on the classical hardness of lattice problems. This would be irrelevant if we were only interested in quantum security, but as it seems unlikely that powerful quantum computers will be available in the immediate future we are also interested in the classical security of lattice-based cryptosystems. In [18] Peikert gives a purely classical reduction from average-case decision LWE to  $\zeta$ -to- $\gamma$  GapSVP. Here we analyze the tightness gap of Peikert's reduction.

Intuitively speaking, Peikert's reduction works by using Regev's  $\text{CVP}_{\phi}$  solver to construct a  $\zeta$ -to- $\gamma$  GapSVP solver. If a lattice L has a large minimum distance  $\lambda_1(L)$  relative to the GapSVP parameter d then the  $\text{CVP}_{\phi}$  solver, given an input  $\mathbf{x}$  near a lattice vector  $\mathbf{l}$ , should always correctly return  $\mathbf{l}$ . If on the other hand  $\lambda_1(L)$  is small relative to d, then there is some other lattice point  $\mathbf{l}'$  close to  $\mathbf{l}$ . With some non-negligible probability, generating an input  $\mathbf{x}$  near  $\mathbf{l}$  will yield an  $\mathbf{x}$  which could also have been generated from  $\mathbf{l}'$ . The  $\text{CVP}_{\phi}$  solver will therefore return a lattice vector other than the original  $\mathbf{l}$  on a non-negligible portion of its possible inputs. If we run the  $\text{CVP}_{\phi}$  solver many times on uniformly random inputs and one of the inputs returns an incorrect output, we can conclude that the  $\lambda_1(L)$  must be small. Similarly, if the  $\text{CVP}_{\phi}$  solver never returns an incorrect vector then we can conclude that  $\lambda_1(L)$  must be large.

To formalize this, for a given lattice L we need to know how many instances  $(L, \mathbf{x})$  to generate so that, if  $\lambda_1(L)$  is small, the  $\text{CVP}_{\phi}$  solver returns a lattice vector other than  $\mathbf{l}$  in at least one instance with overwhelming probability. **Theorem 2.** Let  $n \in \mathbb{N}$ ,  $d \in \mathbb{R}$  with d > 1, and define  $d' = d \cdot \sqrt{n/(4 \log n)}$ . Suppose that L is an n-dimensional lattice with  $\lambda_1(L) \leq d$  and W' is an algorithm for solving  $CVP_{\phi}$  for  $\phi > 2d'$ . For  $\mathbf{l} \in L$  and  $\mathbf{x} \in B(\mathbf{l}, d')$  both sampled uniformly at random, calling W' on  $(L, \mathbf{x})$  returns  $\mathbf{l}' \neq \mathbf{l}$ with probability at least

$$\frac{\Gamma(\frac{n}{2}+1)}{\sqrt{\pi}\Gamma(\frac{n+1}{2})} \left(2 \cdot \int_{0}^{\arccos\left(\sqrt{\frac{\log n}{n}}+\frac{1}{2}\right)} \sin^{n}(t)dt - \int_{0}^{\operatorname{arccos}\left(2\sqrt{\frac{\log n}{n}}+\frac{1}{2}\right)} \sin^{n}(t)dt\right)$$

Proof. Under the condition  $\lambda_1(L) \leq d$ , we want to generate such a lattice L which minimizes the probability that W' returns some  $\mathbf{l}' \neq \mathbf{l}$ . Since  $\mathbf{x} \in B(\mathbf{l}, d')$  is chosen uniformly it suffices to minimize the number of lattice points inside  $B(\mathbf{l}, d')$ . For simplicity assume that L has an orthogonal basis with  $|b_1| = d$  and  $|b_2|, ..., |b_n| \gg d$ . We want to find the portion of  $B(\mathbf{l}, d')$  which does not overlap with  $B(\mathbf{l} + \mathbf{b}_1, d')$  or  $B(\mathbf{l} - \mathbf{b}_1, d')$ . Let V be the volume of  $B(\mathbf{l}, d')$ . First we compute the volume  $V_1$  of overlap between  $B(\mathbf{l}, d')$  and  $B(\mathbf{l} + \mathbf{b}_1, d')$ . The volume of an n-sphere S is given by:

$$V_S = \frac{\pi^{n/2} r^n}{\Gamma(\frac{n}{2} + 1)}$$

**Definition 14** (Spherical cap, [15]). Let S be an n-dimensional sphere with radius r and P be an (n-1)-dimensional plane bisecting S. A spherical cap T is the lesser volume in S sectioned off by P. The height h of T is measured along the radius of S perpendicular to P. In particular,  $0 < h \leq r$ . The volume of a spherical cap T is given by

$$V_T = \frac{\pi^{\frac{n-1}{2}} r^n}{\Gamma(\frac{n+1}{2})} \int_0^{\arccos\frac{r-h}{r}} \sin^n(t) dt.$$

Now  $V_1$  is comprised of two *n*-spherical caps in  $B(\mathbf{l}, d')$ , each with height (d' - d)/2. Thus we have

$$V_1 = 2 \cdot \frac{\pi^{\frac{n-1}{2}} d'^n}{\Gamma(\frac{n+1}{2})} \int_0^{\arccos\frac{d'+d}{2d'}} \sin^n(t) dt.$$

Then by symmetry, for  $V_2$  the volume of overlap between  $B(\mathbf{l}, d')$  and  $B(\mathbf{l} - \mathbf{b}_1, d')$  we have  $V_1 = V_2$ . Since d' > d we know that  $B(\mathbf{l} + \mathbf{b}_1, d')$  and  $B(\mathbf{l} - \mathbf{b}_1, d')$  overlap, so we have  $V_3$  the

volume of overlap between  $B(\mathbf{l} + \mathbf{b}_1, d')$  and  $B(\mathbf{l} - \mathbf{b}_1, d')$  which is again a pair of spherical caps with height (d' - 2d)/2. This gives

$$V_{3} = 2 \cdot \frac{\pi^{\frac{n-1}{2}} d'^{n}}{\Gamma(\frac{n+1}{2})} \int_{0}^{\arccos\frac{d'+2d}{2d'}} \sin^{n}(t) dt.$$

Now by inclusion-exclusion we compute V' the volume of overlap between  $B(\mathbf{l}, d')$  and  $B(\mathbf{l} + \mathbf{b}_1, d') \cup B(\mathbf{l} - \mathbf{b}_1, d')$ 

$$V' = V_1 + V_2 - V_3 = 2 \cdot \frac{\pi^{\frac{n-1}{2}} d'^n}{\Gamma(\frac{n+1}{2})} \left( 2 \cdot \int_0^{\arccos \frac{d'+d}{2d'}} \sin^n(t) dt - \int_0^{\arccos \frac{d'+2d}{2d'}} \sin^n(t) dt \right).$$

Lastly we compute the portion V'/V of B(l, d') contained within  $B(\mathbf{l} + \mathbf{b}_1, d') \cup B(\mathbf{l} - \mathbf{b}_1, d')$ and substitute  $d' = d \cdot \sqrt{n/(4 \log n)}$  to simplify:

$$\begin{split} V'/V &= 2 \cdot \frac{\frac{\pi}{2} \frac{n-1}{2} \frac{d'^n}{\Gamma(\frac{n+1}{2})}}{\frac{\pi^{n/2} d'^n}{\Gamma(\frac{n}{2}+1)}} \left( 2 \cdot \int_{0}^{\arccos \frac{d'+d}{2d'}} \sin^n(t) dt - \int_{0}^{\arccos \frac{d'+2d}{2d'}} \sin^n(t) dt \right) \\ &= \frac{2\Gamma(\frac{n}{2}+1)}{\sqrt{\pi}\Gamma(\frac{n+1}{2})} \left( 2 \cdot \int_{0}^{\operatorname{arccos} \frac{d \cdot \sqrt{n/(4\log n)} + d}{2d \cdot \sqrt{n/(4\log n)}}} \sin^n(t) dt - \int_{0}^{\operatorname{arccos} \frac{d \cdot \sqrt{n/(4\log n)} + 2d}{2d \cdot \sqrt{n/(4\log n)}}} \sin^n(t) dt \right) \\ &= \frac{2\Gamma(\frac{n}{2}+1)}{\sqrt{\pi}\Gamma(\frac{n+1}{2})} \left( 2 \cdot \int_{0}^{\operatorname{arccos} \left(\sqrt{\frac{\log n}{n}} + \frac{1}{2}\right)} \sin^n(t) dt - \int_{0}^{\operatorname{arccos} \left(2\sqrt{\frac{\log n}{n}} + \frac{1}{2}\right)} \sin^n(t) dt \right) \\ &= \frac{2\Gamma(\frac{n}{2}+1)}{\sqrt{\pi}\Gamma(\frac{n+1}{2})} \left( 2 \cdot \int_{0}^{\operatorname{arccos} \left(\sqrt{\frac{\log n}{n}} + \frac{1}{2}\right)} \sin^n(t) dt - \int_{0}^{\operatorname{arccos} \left(2\sqrt{\frac{\log n}{n}} + \frac{1}{2}\right)} \sin^n(t) dt \right). \end{split}$$

Since  $\mathbf{l} \in L$  and  $\mathbf{x} \in B(\mathbf{l}, d')$  are sampled uniformly at random, any  $\mathbf{x}$  in the volume of overlap can be generated in at least two equally likely ways. Therefore the probability that W' returns  $\mathbf{l}$ given  $\mathbf{x}$  is at most 1/2, and we have

$$\Pr\left[W'(L,\mathbf{x})\neq\mathbf{l}\right] \geq \frac{\Gamma(\frac{n}{2}+1)}{\sqrt{\pi}\Gamma(\frac{n+1}{2})} \left(2 \cdot \int_{0}^{\arccos\left(\sqrt{\frac{\log n}{n}}+\frac{1}{2}\right)} \sin^{n}\left(t\right)dt - \int_{0}^{\arccos\left(2\sqrt{\frac{\log n}{n}}+\frac{1}{2}\right)} \sin^{n}\left(t\right)dt\right).$$

Note that this bound is independent of the  $\zeta$ -to- $\gamma$  GapSVP parameter d. This is natural,

as we chose  $d = \lambda_1(L)$  when constructing L and the geometric argument given holds under any scaling of L by a constant factor. From now on we denote this lower bound by P(n). We now have the primary result for Peikert's reduction.

**Theorem 3.** Let  $n, m \in \mathbb{N}$ ,  $q \in \mathbb{N}$  be prime,  $\alpha \in (0, 1)$  with  $\alpha q > \sqrt{n}$ , and  $x, y \in \mathbb{R}^+$  with  $x \ge 2$ and  $y \ge 1$ . Also let  $D = D_{\alpha}$  be the discrete Gaussian distribution on  $\mathbb{Z}_q$  centered around 0 with standard deviation  $\sigma = \alpha q/\sqrt{2\pi}$ . Suppose that W is an algorithm for solving decision LWE given m LWE samples with errors sampled from D, and suppose that W has acceptance rates on some portion 1/y of secret keys which differ by at least 1/x. If  $\gamma \ge n/(\alpha \sqrt{\log n})$  and  $\zeta \ge \gamma$  then there exists a classical algorithm W<sup>\*</sup> which solves  $\zeta$ -to- $\gamma$  GapSVP with overwhelming probability and has tightness gap  $46n^5qx^2yN$ , for  $N \in \mathbb{N}$  chosen so that  $N \ge 32\ln(2) \cdot P(n)^{-1}$ .

Proof. Peikert's reduction proceeds in two parts. From W we construct an algorithm  $W_3$  for solving continuous search LWE as in Theorem 1 and use  $W_3$  to construct an algorithm W' for solving  $\operatorname{CVP}_{\phi}$ . The tightness gap in constructing  $W_3$  is  $46n^2qx^2y$ . Now we choose  $r = q\sqrt{2}/d\gamma$ and generate m samples from  $D_{L^*,r}$  (this can be done efficiently by [18, Prop 2.13]). Invoking the  $\operatorname{CVP}_{\phi}$ -to-DGS<sub>r</sub> reduction from the iterative step in Theorem 1 using  $W_3$  and these samples gives an algorithm W' which solves  $\operatorname{CVP}_{\phi}$  for  $\phi = \alpha q/r = d\alpha \gamma/\sqrt{2} \ge dn/\log n > 2d'$  (where  $d' = d \cdot \sqrt{n/(4\log n)}$  as before). From Theorem 1 the tightness gap in constructing W' from  $W_3$ is  $n^2$ .

Now let  $W^*$  be the following algorithm: for i = 1, ..., N, select  $\mathbf{l}_i \in L$  and  $\mathbf{x}_i \in B(\mathbf{l}_i, d')$  both uniformly at random, and run W' on  $(L, \mathbf{x}_i)$ . If the output is any  $\mathbf{l}' \neq \mathbf{l}_i$  halt and conclude that  $1 \leq \lambda_1(L) \leq d$ , otherwise continue. If all N repetitions run without halting, then conclude that  $\gamma \cdot d < \lambda_1(L) \leq \zeta$ . If  $\gamma \cdot d < \lambda_1(L) \leq \zeta$ , then since  $\phi > 2d'$  we are guaranteed that  $(L, \mathbf{x})$  is a valid  $\operatorname{CVP}_{\phi}$  instance and W' will correctly return  $\mathbf{l}$ . Otherwise suppose  $1 \leq \lambda_1(L) \leq d$ . Then  $W^*$  will halt with probability at least 1/2 when given an input  $(L, \mathbf{x}_i)$  such that there exists an  $\mathbf{l}' \neq \mathbf{l}_i$ with  $||\mathbf{x}_i - \mathbf{l}'|| \leq d')$ . By Theorem 2 the probability of sampling such an  $\mathbf{x}_i$  is at least P(n), so to achieve an error rate less than  $2^{-32}$  over N repetitions we have

$$Pr[W^* \text{ fails to halt}] \leq [1 - P(n)]^N$$
$$\leq [1 - P(n)]^{32\ln(2) \cdot P(n)^{-1}}$$
$$= \left( [1 - P(n)]^{P(n)^{-1}} \right)^{32\ln(2)}$$
$$< \left( \frac{1}{e} \right)^{32\ln(2)}$$
$$= 2^{-32}.$$

#### 4.1 Discussion

The most attractive feature of Peikert's reduction is that it is entirely classical. The best known runtimes for solving SVP classically are significantly slower than for quantum solutions; this would allow us to choose a smaller n for implementations where classical security is the primary concern. Only when quantum computers become sufficiently powerful to carry out attacks would we need to revert to the larger quantum-secure values for n. Another difference from Regev's reduction is that the underlying lattice problem is  $\zeta$ -to- $\gamma$  GapSVP rather than SIVP<sub> $\gamma$ </sub>. Since we will ultimately select parameters based on the best known runtime for an SVP solver and a solution to SVP trivially implies a solution to  $\zeta$ -to- $\gamma$  GapSVP, parameters selected according to Peikert's reduction would be potentially more vulnerable to future breakthroughs than Regev's reduction.

Unfortunately, Peikert's reduction is of little use when it comes to selecting concrete parameters for a cryptosystem. While the reduction runs in polynomial time, it is not faster than the best known runtimes for SVP even in very large dimensions ( $n \approx 4000$ ). The difficulty arises in the N term; the portion of  $B(\mathbf{l}, d')$  which is within distance d' of another lattice point becomes extremely small in large dimensions necessitating a correspondingly large number of repetitions. We will discuss this in more detail in Section 5.2.

### Chapter 5

### Implementation results

Before we can select parameters for a cryptosystem, we need to refine our desired notion of security. An attacker can make tradeoffs between the runtime of their algorithm, the advantage it achieves, and the number of secret keys against which it achieves that advantage. While we want it to be difficult for an attacker to succeed meaningfully in any of these capacities, demanding that it be difficult to achieve even a negligible improvement in any capacity is extremely conservative and will result in a much slower cryptosystem.

Ultimately we want to construct an inequality of the form  $run(A) \cdot run_A(W) < 2^l$  for a reduction W making  $run_A(W)$  calls to an algorithm A, where  $2^l$  is the best known runtime for a solution to the lattice problem used by W. We therefore need to express the maximum allowable runtime of an attack given all other security parameters. Expanding on the definition of  $(t, \epsilon)$ -security, we have the following lemmas:

**Lemma 3.** Let A be an algorithm for solving average-case decision LWE. Suppose that A achieves advantage at least 1/a on a portion 1/y of all secret keys and no advantage on the remaining secret keys. Then A breaks k-bit hardness of decision LWE if  $run(A) < \frac{2^k}{ay}$ .

Proof. Let S be the set of all possible keys and  $T \subseteq S$  be the portion of keys against which A achieves advantage at least 1/a. Denote by  $Adv_T(A)$  the advantage achieved by A on T. Then  $Adv(A) = \frac{1}{y} \cdot Adv_T(A) \geq \frac{1}{ay}$ . Let  $\epsilon = \frac{1}{ay}$  and  $t = \frac{2^k}{ay}$ . Then we have  $\epsilon \leq Adv(A)$  and t > run(A) such that  $\frac{t}{\epsilon} \leq 2^k$ , so A breaks k-bit hardness for decision LWE.

If secret keys are sampled uniformly at random from S then we have  $|T| = \frac{1}{y}|S|$ , but it is more generally correct to think of T as being a subset of S such that the distribution from which secret keys are drawn samples from T with probability 1/y.

**Lemma 4.** Let A be an algorithm for solving average-case decision LWE. Suppose that on some set T of secret keys A accepts with probabilities  $a_{LWE}$ ,  $a_U \in [0,1]$  on inputs from the LWE and uniform distributions respectively. Also suppose that  $a_{LWE} - a_U \ge 1/x$ . Then A achieves advantage 1/2x on T.

*Proof.* Let  $(\mathbf{A}, \mathbf{t})$  be an arbitrary decision LWE instance. Recalling the definition of advantage, we have

$$\begin{aligned} Adv(A) &= \Pr[A \text{ wins the decision LWE experiment}] - \frac{1}{2} \\ &= \frac{1}{2} \Pr[A \text{ wins } | \mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}] + \frac{1}{2} \Pr[A \text{ wins } | \mathbf{t} \text{ sampled uniformly at random}] - \frac{1}{2} \\ &= \frac{1}{2} a_{LWE} + \frac{1}{2} (1 - a_U) - \frac{1}{2} \\ &= \frac{1}{2} (a_{LWE} - a_U) \\ &\geq \frac{1}{2x}. \end{aligned}$$

Note that the implicit assumption  $a_{LWE} > a_U$  is without loss of generality, since if  $a_{LWE} < a_U$ then A achieves negative advantage and we can immediately create a better algorithm by reversing the outputs of A. From these lemmas we have the immediate corollary:

**Corollary 1.** Let  $k \in \mathbb{N}$ , A be an algorithm for solving average-case decision LWE, and W be a reduction from average-case decision LWE to a lattice problem P. Suppose that A has acceptance rates on some portion 1/y of secret keys which differ by at least 1/x, so 1/a = 1/2x is the advantage attained by A on the portion of keys. Lastly suppose that  $run(A) < \frac{2^k}{ay}$ . Then W solves P using A as a subroutine with  $run(W) < \frac{2^k}{ay} \cdot run_A(W) \leq \frac{2^{k-1}}{xy} \cdot run_A(W)$ .

#### 5.1 Parameter Selection from Regev's Reduction

By using the above corollary and the best known running times for solutions to SVP, we can derive parameters for a cryptosystem under which a reduction will guarantee that breaking the cryptosystem is at least as hard as solving SVP. From [4, Section 4.1] we have a runtime of  $2^{l \cdot n}$  for an SVP solver in dimension n, with  $l = \log \sqrt{3/2}$  for the best known classical runtime,  $l = \log \sqrt{13/9}$  the best known quantum runtime, and  $l = \log \sqrt{4/3}$  as a plausible worst case runtime anticipating future improvements. Note that Regev's reduction itself is quantum, so we should be considering quantum solvers. The values listed for a classical solver would be applicable if Regev's reduction could be made classical, and would constitute a significant improvement. This brings us to our main result.

**Theorem 4.** Let  $n, m \in \mathbb{N}$ ,  $q \in \mathbb{N}$  be prime,  $\alpha \in (0, 1)$  with  $\alpha q > \sqrt{n}$ , and  $x, y \in \mathbb{R}^+$  with  $x \ge 2$ and  $y \ge 1$ . Also let  $D = D_{\alpha}$  be the discrete Gaussian distribution on  $\mathbb{Z}_q$  centered around 0 with standard deviation  $\sigma = \alpha q/\sqrt{2\pi}$ . Let A be an algorithm for solving average-case decision LWE and suppose that on some proportion 1/y of all possible keys  $\mathbf{s} \in \mathbb{Z}_q^n$ , A has acceptance rates which differ by at least 1/x. Let  $l \in \{\log \sqrt{3/2}, \log \sqrt{13/9}, \log \sqrt{4/3}\}$  be an SVP runtime parameter. Then there exists an algorithm  $W^*$  which solves  $SIVP_{\gamma}$  for  $\gamma = \tilde{O}(n/\alpha)$  with  $run(W^*) < 2^{l \cdot n}$  if  $run(A) < \frac{2^k}{2x \cdot y}$  and

$$2^{k-1} \cdot 276xn^8 mq \log \left(\alpha q/\sqrt{n}\right)^{-1} < 2^{l \cdot n}.$$
(5.1.1)

*Proof.* By Theorem 1 Regev's reduction W has tightness gap  $run_A(W) = 276x^2yn^8mq\log(\alpha q/\sqrt{n})^{-1}$ , and so this is an immediate consequence of Corollary 1.

This tells us that, for a set of parameters which satisfy the conditions of Theorem 4, if we assume that there is no solution to  $\text{SIVP}_{\gamma}$  with running time less than  $2^{l\cdot n}$  then there is no algorithm which breaks 128-bit hardness of decision LWE. There remains one obvious problem: we have produced a guarantee for the hardness of decision LWE, but ultimately what we want is a guarantee of security for a cryptosystem. An algorithm which achieves non-negligible advantage in decision LWE trivially wins the IND-CPA experiment, but it is not clear that the reverse holds. We omit this concern for two reasons: the reduction from IND-CPA to decision LWE would depend on the particular implementation of the cryptosystem in question, and regardless

Setting	Sample Parameters
Best known classical, $l = \log \sqrt{3/2}$	$n = 1310,  q \approx 2^{20},  \sigma = 15$
Best known quantum, $l = \log \sqrt{13/9}$	$n = 1460, q \approx 2^{21}, \sigma = 15.5$
Worst case quantum, $l = \log \sqrt{4/3}$	$n = 1870, q \approx 2^{22}, \sigma = 17.5$

Figure 5.1.1: Sample parameters derived from Regev's reduction

of the implementation the tightness gap should be extremely small. As an example, a reduction for the public-key encryption scheme presented in section 2.2.1 can be found in [20, Lemma 5.4] and is tight.

To give sample parameters which satisfy these contraints, we will make some simplifications. For 128-bit security we have  $run(A) \leq \frac{2^{127}}{xy}$  so the most conservative choice of parameters is  $x = 2^{127}, y = 1$ . This corresponds to an algorithm which achieves a small advantage against any possible secret key, with a tradeoff between advantage and runtime. With regard to Dthe parameter of interest for practical implementations is  $\sigma$ , so we can make the substitution  $\alpha q = \sigma \sqrt{2\pi}$ . This yields the constraint  $\sigma > \sqrt{n/2\pi}$  which ensures that the iterative step in the reduction functions properly. Most constructions have  $m \approx n$  and  $q \approx n^2$  so we use these as approximations. This gives us the simplified constraint

$$2^{255} \cdot 276n^{11} \log \left(\sigma \sqrt{2\pi/n}\right)^{-1} < 2^{l \cdot n}.$$
(5.1.2)

We note that, other than being prime, the only condition on q is that  $\alpha q > \sqrt{n}$ . Since q is only a linear factor in the reduction it has only a small effect on the inequality. Rather than the above condition, the approximate value of q will ultimately be determined by the correctness requirements of a cryptosystem, as q must be large enough relative to  $\sigma$  that samples from  $D_{\sigma}$ are small relative to q. Figure 5.1.1 gives a set of LWE parameters derived from the runtimes of best known classical, best known quantum, and hypothetical worst-case quantum SVP solvers. Values for n were searched for in increments of 10 and values of  $\sigma$  were searched for in increments of 1/2 due to the approximations already made.

#### 5.2 Parameter Selection from Peikert's Reduction

Unfortunately Peikert's reduction does not yield a useful concrete security guarantee. Recall from Theorem 3 that  $N \in \mathbb{N}$  is the number of samples that must be drawn from a ball centered at  $\mathbf{l} \in L$  of radius d' so that with overwhelming probability at least one sample lies within distance d' of a lattice point other than 1. For a given sample, the probability of this occurring is bounded below by P(n) as defined in Theorem 2. Even for dimensions as small as n = 800 we have  $P(n) < 2^{-250}$  with N inversely proportional to P(n). N remains larger than the best known runtimes for SVP as high as dimension n = 4000, at which point even computing P(n) becomes difficult.

We might hope that this is a result of a particularly bad lower bound, as we only considered a worst possible lattice when computing P(n). However, consider the argument from Theorem 3 on the lattice  $L = \mathbb{Z}^n$  for a ball centred on **0**. For n = 800, as an example, we have d = 1 and  $d' = \sqrt{n/(4 \log n)} \approx 4.6$ . To get a rough estimate of the number of vectors within d' of **0**, we have:

**Lemma 5.** Let  $n, k \in \mathbb{N}$ .  $\mathbb{Z}^n$  has  $2^k \cdot \binom{n}{k}$  vectors of length  $\sqrt{k}$  with coordinates in  $\{1, 0, -1\}$ .

*Proof.* Let  $\mathbf{v} \in \mathbb{Z}^n$  have coordinates in  $\{1, 0, -1\}$ . If  $\mathbf{v}$  has length  $\sqrt{k}$ , then  $\mathbf{v}$  must have k non-zero coordinates where each non-zero coordinate may be either 1 or -1. There are  $2^k \cdot \binom{n}{k}$  ways to choose such a vector.

These are not the only short vectors in  $\mathbb{Z}^n$ , but for  $d' \approx 4.6$  it yields a reasonable portion of the vectors closest to **0**. Notice that if  $\mathbf{v} \in \mathbb{Z}^n$  is such a vector and  $\mathbf{u} = k \cdot \mathbf{v}$  for k > 1, then  $B(\mathbf{u}, d') \cap B(\mathbf{0}, d') \subset B(\mathbf{v}, d') \cap B(\mathbf{0}, d')$ , so any such **u** may be ignored. Now the number of  $\{1, 0, -1\}$  vectors of length less than  $d' \approx 4.6$  is less than

$$\sum_{i=1}^{21} 2^i \cdot \binom{800}{i} \approx 2^{160}.$$

So even assuming that each pair  $B(\mathbf{v}, d'), B(-\mathbf{v}, d')$  produces pairwise disjoint overlaps with  $B(\mathbf{0}, d')$ , by Theorem 2 the total portion of  $B(\mathbf{0}, d')$  overlapping with some  $B(\mathbf{v}, d')$  is at most

 $2^{160} \cdot 2^{-250} = 2^{-90}$ , which means that the number of needed samples is still too large to produce useful parameters under Peikert's reduction. This also assumes that all overlaps are equally sized, when in fact the amount of overlap created by any  $B(\mathbf{v}, d')$  will be significantly smaller if  $||\mathbf{v}|| > 1$ . We emphasize that these calculations were extremely rough, and intended only to demonstrate that even a very well-behaved lattice produces a large running time.

### Chapter 6

### **Future considerations**

In section 3.1 we discussed several considerations which might impact the security guaranteed by Regev's reduction. Here we instead consider possible improvements to the reduction itself, in particular considering how small the tightness gap would have to be to give a meaningful guarantee of security for current implementations. As an example, Bos et al. [4] recommend n = 752,  $q = 2^{15}$ ,  $\sigma = \sqrt{1.75}$  as parameters for a quantum-secure implementation of the Frodo key exchange protocol.

While we focus primarily on improving n, the parameters q and  $\sigma$  are also potential areas for improvement. Regev's search LWE to worst-case decision LWE reduction requires q to be prime, and Regev's DGS<sub>r</sub> to search LWE reduction requires  $\sigma > \sqrt{n/(2\pi)}$ . Ideally we would like to take q to be a power of 2, as reducing integers modulo  $2^k$  in binary can be done very efficiently. The condition on  $\sigma$  yields values much larger than in current implementations, and the larger error distribution will necessitate a corresponding larger value of q to maintain correctness.

Now we want to consider hypothetical improvements to Regev's reduction. While a reduction W could in principle use any standard lattice problem, we will assume that W solves SVP for ease of notation. Suppose that A is an algorithm for solving decision LWE with acceptance rates differing by 1/x on a portion 1/y of secret keys. Then for a W to imply k-bit hardness of decision LWE assuming a best known runtime of  $2^{l \cdot n}$  for SVP, from Corollary 1 we have

$$\frac{2^{127}}{xy} \cdot run_A(W) < 2^{l \cdot n}$$

Runtime exponent	Classical setting	Quantum setting	Worst-case quantum setting
t = 0	n = 1040	n = 1150	n = 1470
t = 1	n = 1080	n = 1190	n = 1530
t = 2	n = 1110	n = 1230	n = 1580
t = 3	n = 1150	n = 1270	n = 1630
t = 4	n = 1180	n = 1310	n = 1690

Figure 6.0.1: Secure dimensions for an SVP to search LWE reduction with tightness gap  $n^t$ 

where as before  $n \in \mathbb{N}$  is a dimension and  $l \in \{\log \sqrt{3/2}, \log \sqrt{13/9}, \log \sqrt{4/3}\}$  is a runtime parameter based on the desired setting. From Regev's reduction we will consider three possible forms for W, and in each case we will let  $n^t$  be a tightness gap for  $t \in \mathbb{R}$ .

The weakest possibility we consider is an improvement to the reduction from SVP to search LWE. This is the most complex part of Regev's reduction, and so might reasonably be considered the most likely place to find improvements. By comparison the reduction from search LWE to average-case decision LWE is well known and relatively straightforward. From Theorem 1 the search LWE to average-case decision LWE reduction has tightness gap  $276x^2yn^2q$ , so W gives a concrete security guarantee if

$$2^{127} \cdot 276xqn^{2+t} < 2^{l \cdot n}. \tag{6.0.1}$$

By taking  $q \approx n^2$  and  $x = 2^{128}$  as in section 5.1 we can compute the smallest dimension under which W gives a concrete guarantee. However even a tight SVP to search LWE reduction still yields n > 1000 in every setting. To get a better result, we can instead consider improving the entire SVP to average-case decision LWE reduction. Because of the tradeoff between runtime and advantage we expect that any reduction to average-case decision LWE will include an xyterm, but the extra factor of x in Regev's reduction significantly inflates the overall runtime of the reduction. We therefore consider two cases: an SVP to average-case decision LWE reduction with tightness gap  $x^2yn^t$ , and one with tightness gap  $xyn^t$ . These yield the following inequalities:

$$2^{127} \cdot xn^t < 2^{l \cdot n}. \tag{6.0.2}$$

$$2^{127} \cdot n^t < 2^{l \cdot n}. \tag{6.0.3}$$

Runtime exponent	Classical setting	Quantum setting	Worst-case quantum setting
t = 0	n = 880	n = 970	n = 1230
t = 1	n = 910	n = 1000	n = 1280
t=2	n = 940	n = 1040	n = 1330
t = 3	n = 980	n = 1080	n = 1380
t = 4	n = 1010	n = 1120	n = 1440
t = 5	n = 1040	n = 1160	n = 1490
t = 6	n = 1070	n = 1200	n = 1540

Figure 6.0.2: Secure dimensions for an SVP to average-case decision LWE reduction with tightness gap  $x^2yn^t$ 

Runtime exponent	Classical setting	Quantum setting	Worst-case quantum setting
t = 0	n = 440	n = 490	n = 620
t = 1	n = 470	n = 510	n = 660
t=2	n = 500	n = 550	n = 710
t = 3	n = 530	n = 590	n = 750
t = 4	n = 560	n = 620	n = 800
t = 5	n = 600	n = 660	n = 850
t = 6	n = 630	n = 700	n = 900

Figure 6.0.3: Secure dimensions for an SVP to average-case decision LWE reduction with tightness gap  $xyn^t$ 

While we do not expect the values in 6.0.2 or 6.0.3 to correspond to actual reductions for small values of t, the extreme results in figure 6.0.3 suggest that it is unlikely that we can both remove the additional factor of x while also significantly improving the overall reduction. For a final set of values, we consider the case where Regev's reduction is improved only by removing the additional factor of x. Restating 5.1.2 with the factor of  $x = 2^{128}$  removed gives

$$2^{127} \cdot 276n^{11} \log \left(\sigma \sqrt{2\pi/n}\right)^{-1} < 2^{l \cdot n} \tag{6.0.4}$$

where  $\sigma > \sqrt{n/(2\pi)}$  as before. This change alone brings the quantum-secure dimension down to only n = 900, and would constitute a significant improvement over the values from section 5.1.

Setting	Sample Parameters
Best known classical, $l = \log \sqrt{3/2}$	$n = 820,  q \approx 2^{19},  \sigma = 12$
Best known quantum, $l = \log \sqrt{13/9}$	$n = 900, q \approx 2^{20}, \sigma = 13$
Worst case quantum, $l = \log \sqrt{4/3}$	$n = 1170, q \approx 2^{21}, \sigma = 14.5$

Figure 6.0.4: Sample parameters derived from Regev's reduction with a factor of x removed

In particular we observe that the improvement from removing x is comparable to replacing the entire reduction with one of tightness gap  $x^2y$ , which we expect to be impossible. We propose that the worst-case to average-case reduction is therefore of significant interest for future work, either producing a reduction with smaller tightness gap or proving that no such reduction exists.

### 6.1 Conclusions

In summary, Regev's reduction guarantees that breaking a lattice-based cryptosystem is at least as hard as solving SIVP<sub> $\gamma$ </sub> in dimensions as low as n = 1460 for q and  $\sigma$  chosen appropriately. Moreover there are several potential improvements to this result which could bring this number closer to those being used in practice, currently  $n \approx 750$ . Peikert's reduction, while entirely classical, unfortunately has tightness gap too large to give a concrete security assurance in any practical dimension.

Ultimately we do not expect that the parameters we propose here will become common in everyday use. While the resulting cryptosystems would not be unusable, they would be significantly slower than current implementations. Instead we view this work as a worst-case security assurance; even if efficient attacks against current implementations are found, lattice based cryptosystems will still be viable as long as SVP remains hard. This assurance partially mitigates the risk of new attacks being discovered, which is particularly welcome as lattice-based cryptography is still a relatively new field.

# Bibliography

- Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In 28th ACM STOC, pages 99–108. ACM Press, May 1996.
- [2] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In 29th ACM STOC, pages 284–293. ACM Press, May 1997.
- [3] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Heidelberg, August 2009.
- [4] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, ACM CCS 16, pages 1006–1018. ACM Press, October 2016.
- [5] Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Another look at tightness II: Practical issues in cryptography. Cryptology ePrint Archive, Report 2016/360, 2016. http://eprint.iacr.org/2016/360.
- [6] Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Concrete analysis of Regev's worst-case/average-case reduction. http://cacr.uwaterloo.ca/~ajmeneze/ anotherlook/papers/WCACanalysis.pdf, 2016.

- [7] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, ASIACRYPT 2011, volume 7073 of LNCS, pages 1–20. Springer, Heidelberg, December 2011.
- [8] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions* on Information Theory, 22(6):644–654, 1976.
- [9] Sebastian Faust, Daniel Masny, and Daniele Venturi. Chosen-ciphertext security from subset sum. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 35–46. Springer, Heidelberg, March 2016.
- [10] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In 54th FOCS, pages 40–49. IEEE Computer Society Press, October 2013.
- [11] Craig Gentry. A Fully Homomorphic Encryption Scheme. PhD thesis, Stanford, CA, USA, 2009. AAI3382729.
- [12] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing, 17(2):281–308, April 1988.
- [13] Lov K. Grover. A fast quantum mechanical algorithm for database search. In 28th ACM STOC, pages 212–219. ACM Press, May 1996.
- [14] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. Mathematische Annalen, 261(4):515–534, Dec 1982.
- [15] S. Li. Concise Formulas for the Area and Volume of a Hyperspherical Cap. Asian Journal of Mathematics & Statistics, 4(1):66–70, 2011.
- [16] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption.
   In Aggelos Kiayias, editor, CT-RSA 2011, volume 6558 of LNCS, pages 319–339. Springer, Heidelberg, February 2011.

- [17] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, May 2010.
- [18] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. Cryptology ePrint Archive, Report 2008/481, 2008. http://eprint.iacr.org/2008/481.
- [19] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. http://eprint.iacr.org/2015/939.
- [20] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, 37th ACM STOC, pages 84–93. ACM Press, May 2005.
- [21] Ronald Linn Rivest, Adi Shamir, and Leonard Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Commun. ACM, 21(2):120–126, February 1978.
- [22] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In 35th FOCS, pages 124–134. IEEE Computer Society Press, November 1994.