# A Framework to Select Parameters for Lattice-Based Cryptography

Nabil Alkeilani Alkadri, Johannes Buchmann, Rachid El Bansarkhani, and Juliane Krämer

Technische Universität Darmstadt
Department of Computer Science, Cryptography and Computer Algebra
Hochschulstraße 10, 64289 Darmstadt, Germany
{nalkeilani_alkadri, buchmann, elbansarkhani, jkraemer}@cdc.informatik.tu-darmstadt.de

**Abstract**

Selecting parameters in lattice-based cryptography is a challenging task, which is essentially accomplished using one of two approaches. The first (very common) approach is to derive parameters assuming that the desired security level is equivalent to the bit hardness of the underlying lattice problem, ignoring the gap implied by available security reductions. The second (barely used) approach takes the gap and thus the security reduction into account. In this work, we investigate how efficient lattice-based schemes are if they respect existing security reductions. Thus, we present a framework to systematically select parameters for any lattice-based scheme using either approaches. We apply our methodology to the schemes by Lindner and Peikert (LP), by El Bansarkhani (LARA), and by Ducas et al. (BLISS). We analyze their security reductions and derive a gap of 3, 5, and 63 bits, respectively. We show how parameters impact the schemes' efficiency when involving these gaps.

**Keywords:** Lattice-Based Cryptography, Ideal Lattices, Parameter Selection, Security Reduction, Tightness, Lattice-Based Assumptions.

## 1 Introduction

During the last two decades lattice-based cryptography has been studied intensively and become a major research topic in the cryptographic community. An important reason for this is the fact that lattices offer a huge source of computational problems that are conjectured to be hard even in the presence of large-scale quantum computers, thus serving as promising candidates in the post-quantum era. Furthermore and due to their combinatorial structure, lattices can be used to realize powerful cryptographic constructions such as fully homomorphic encryption [Gen09], attribute-based encryption for any arbitrary polynomial size [GVW13], multi-linear maps [GGH13], and attribute-based signatures for expressive policies [EE16]. Moreover, lattice-based cryptographic applications can also enjoy strong security guarantees under worst-case intractability assumptions [Ajt96, Reg05].

Beside the advantageous features of lattice-based cryptography mentioned above, selecting appropriate parameters for lattice-based schemes, however, is very involved. In fact, the hardness of the underlying lattice problems depends on several parameters rather than just a simple bit length as in many number-theoretic assumptions like factoring and discrete logarithm problem. More precisely, parameters to be chosen for a lattice-based scheme $\mathcal{S}$ have to satisfy correctness of the respective algorithms of $\mathcal{S}$ (or an acceptable percentage of correctness), a desired level of efficiency, and a reasonable bit security, where efficiency and security always affect each other. A bit security $\ell$ of a scheme $\mathcal{S}$ is defined by the largest integer such that $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \geq 2^{\ell}$, for any adversary $\mathcal{A}$ running in time at most $t_{\mathcal{A}}$ and has success probability at least $\varepsilon_{\mathcal{A}}$ in breaking $\mathcal{S}$. Most lattice-based schemes come with a proof of security, which indicates that a scheme $\mathcal{S}$ is secure with respect to a certain security notion as long as some lattice problem $\mathcal{P}$ is hard to solve. However, the security of a scheme is sometimes believed to be based on the hardness of a problem with no known proof.

A very common approach for choosing parameters offering $\ell$ bits of security is to ensure that the parameters of the lattice problem $\mathcal{P}$ emerging from the security reduction offer a hardness level of $\ell$ bits, which is defined similar to the bit security for any algorithm solving $\mathcal{P}$. Note the difference between the security level of a scheme (within a certain security model) and the hardness level of the related computational problem. In other words, the desired security level $\ell$ is assumed to be identical to the hardness level of the underlying problem $\mathcal{P}$ and parameters are selected according to the best known solver. If applicable, direct attacks on $\mathcal{S}$ are involved as well, if for instance the conditions for the security reduction to hold are not sufficient.

However, a security proof of any scheme $\mathcal{S}$ reveals more than security restrictions: It actually reduces solving the underlying problem $\mathcal{P}$ to breaking $\mathcal{S}$, therefore also called a security reduction. This reduction is an algorithm $\mathcal{D}$ that employs a hypothetical adversary $\mathcal{A}$ against $\mathcal{S}$ as a subroutine in order to solve $\mathcal{P}$. Suppose that $\mathcal{A}$ takes time at most $t_{\mathcal{A}}$ to break $\mathcal{S}$ with probability at least $\varepsilon_{\mathcal{A}}$. Then, $\mathcal{D}$ solves $\mathcal{P}$ in time at most $t_{\mathcal{D}} \geq t_{\mathcal{A}}$ with probability at least $\varepsilon_{\mathcal{D}} \leq \varepsilon_{\mathcal{A}}$. This means that we need to spend more efforts to solve $\mathcal{P}$ than the efforts required to break $\mathcal{S}$. Consequently, the security reduction induces a gap between the bit security of $\mathcal{S}$ and the bit hardness of $\mathcal{P}$. This gap is called the *tightness gap* and is defined by $\gamma = \frac{t_{\mathcal{D}} \cdot \varepsilon_{\mathcal{A}}}{t_{\mathcal{A}} \cdot \varepsilon_{\mathcal{D}}} \geq 1$ [CKMS16]. If this gap is small, then the reduction is called *tight*, otherwise it is called *non-tight*. Therefore, a second approach for parameter selection is to derive the gap induced by the reduction and make sure that $\mathcal{P}$ offers a hardness of $\ell + \log \gamma$ bits. Note that this approach does not apply on schemes with no available security reduction.

For almost all lattice-based cryptographic schemes with a security reduction, parameters have been selected following the first approach, e.g., [MR09, LP11, Lyu12, GLP12, EB13, DDLL13, BG14, DEG+14, BCNS15, BCD+16, ADPS16, EB17]. To our knowledge, the only exceptions are the signature schemes TESLA [ABB+17] and Ring-TESLA [ABB+16]. In other words, the tightness gap induced by the security reduction is not taken into account by almost all lattice-based schemes and parameters were chosen according to known algorithms solving the underlying lattice problems. Therefore, security reductions were taken only as a theoretical concept and they were not considered, no matter how efficient they are. Consequently, parameters may provide less bit security than claimed and hence no reliable statement about the concrete security of the scheme can be made in practice. A typical reason for ignoring security reductions is the significant difference in time and/or probability of success between breaking the scheme and solving the underlying problem, i.e., the tightness gap is large, so that it requires to select larger parameters resulting in low performance and larger sizes of keys, signatures, ciphertexts, etc. An additional justification for choosing parameters following this approach is the unavailability of attack algorithms exploiting the tightness gap.

**Contributions.** In this work, we present a methodology to systematically select parameters for any cryptographic scheme $\mathcal{S}$, whose security is based on the hardness of some lattice problem(s) $\mathcal{P}$. Our framework covers different methods of choosing parameters, which include involving and ignoring the tightness gap of the security reduction. It can also be applied on schemes without known security reduction. We construct a function, which maps the parameters of $\mathcal{S}$ to a desired security level in three different ways (see Figure 1).

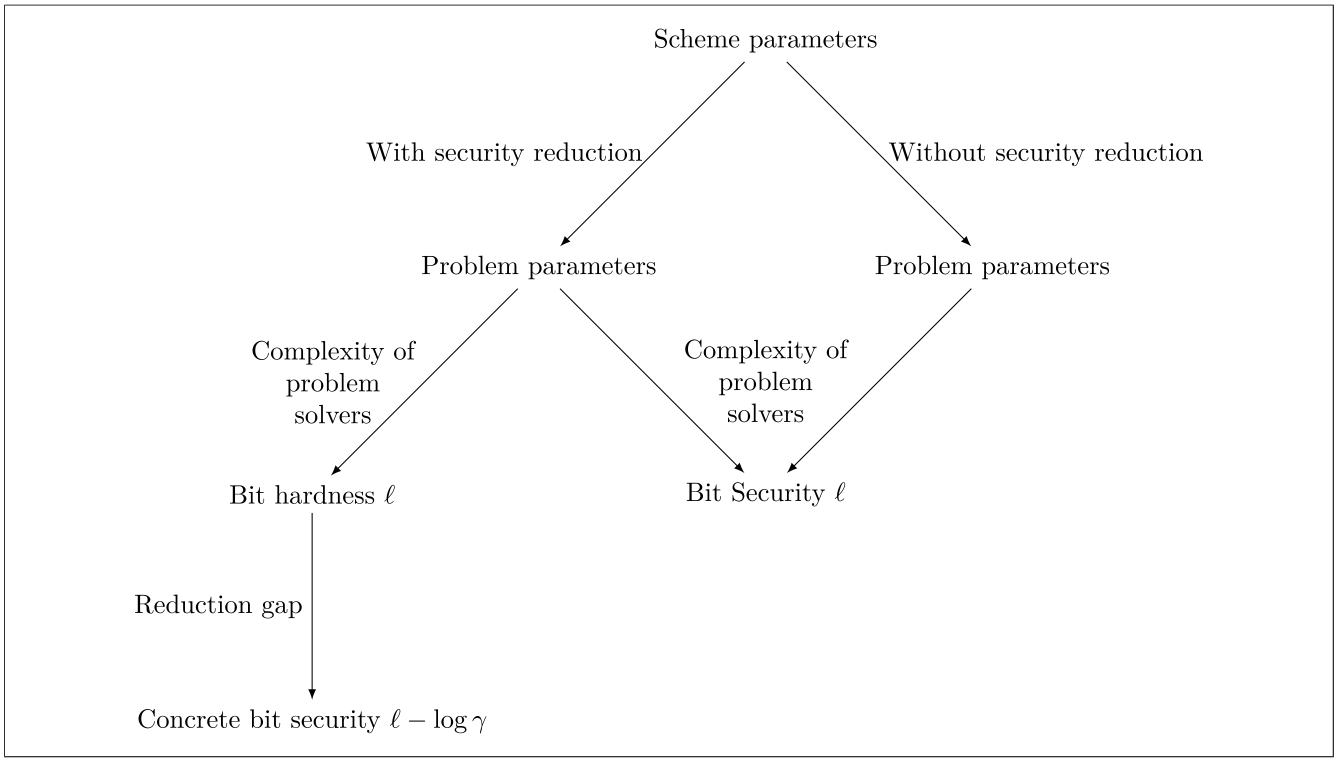

Figure 1: The different ways of selecting secure parameters in lattice-based cryptography. The first path on the left side takes the tightness gap $\gamma$ of the security reduction into account, while the second and third one ignore it with the difference that the third path on the right side completely disregards the security reduction. Therefore, the third path can be taken for schemes with no known security reduction.

Focusing on provably secure public-key encryption and digital signature schemes, we apply our methodology to the currently most efficient lattice-based schemes. More precisely, we analyze the ideal lattice variants of the respective constructions, which are suitable for practice. We consider the public-key encryption schemes proposed by Lindner and Peikert [LP11] which we denote by LP, and by El Bansarkhani [EB17] which is called LARA. Both schemes have the same key sizes and are very efficient in practice (for a detailed comparison see [EB17, Tables 4,5,6,7]). Furthermore, we consider the digital signature scheme BLISS-B by Ducas et al. [DDLL13, Duc14]. For each scheme we analyze the security proof and derive the running time and success probability of the reductions. Then, we deduce the tightness gap between the bit security of the schemes and the bit hardness of the respective underlying problem. We obtain a gap of at most 3 and 5 bits for the encryption schemes LP [LP11] and LARA [EB17], respectively, and a gap of at least 63 bits for BLISS-B [DDLL13, Duc14]. Our analysis includes the reductions from the average-case hardness of lattice problems to the security of the considered schemes and does not consider the reductions from worst-case lattice problems. More specifically, we analyze the reductions from the average-case hardness of the ring learning with errors (R-LWE) and ring short integer solution (R-SIS) problem to the security of the lattice-based schemes LP, LARA, and BLISS-B. We note that current state-of-the-art cryptanalysis tools for solving R-LWE and R-SIS related to certain lattice dimensions are more efficient than solvers targeting their underlying lattice problems in the worst-case for the same dimension. Furthermore, considering the existing reductions from worst-case hardness of lattice problems to the average-case hardness of R-LWE and R-SIS requires much larger parameters, since those reductions are highly non-tight [CKMS16]. However, our framework can also be applied to further reductions straightforwardly.

We select for LP, LARA, and BLISS-B parameters that satisfy security levels of 128 bits (for near-term security) and 256 bits (for long-term security). For each security level, we propose two types of parameters: The first one takes the tightness gap of the security reduction into account and the second one does not involve the gap when analyzing the hardness of the underlying problems. The goal is to investigate how the efficiency of the schemes suffers from increasing parameters due to the tightness gap induced by the respective security reduction. Therefore, we scrutinize the impact of the parameters on the efficiency of each scheme by providing the corresponding sizes of keys, ciphertexts, and signatures as well as timings for encryption/decryption and signing/verifying. We demonstrate that the tight reductions of both encryption schemes LP [LP11] and LARA [EB17] ensure sizes of keys and ciphertexts as well as timings for encryption and decryption that remain the same as without involving the gap, although parameters can slightly increase (see Tables 2 and 4). The non-tight reduction of BLISS-B [DDLL13, Duc14] can increase the parameters such that larger sizes of keys and signatures are obtained, while maintaining performance (see Table 7).

We note that BLISS-B is currently the fastest provably secure lattice-based signature scheme in terms of signature generation with a new signature compression technique that leads to very short signatures. It is based on the scheme by Lyubashevsky [Lyu12] with a modification on its rejection sampling and key generation. The signature scheme proposed by Güneysu et al. [GLP12] is also an efficient improvement of Lyubashevsky's scheme [Lyu12], which is optimized for embedded systems. According to the authors, its security proof is based on the reduction by Lyubashevsky. However, they only provide a proof sketch although the underlying lattice problem has been changed. Unlike BLISS-B and as stated in [DDLL13], the compression technique introduced by Güneysu et al. [GLP12] makes the scheme not strongly unforgeable anymore. Therefore, we decided to analyze the more recent modification of Lyubashevsky's scheme, i.e., BLISS-B. We also do not consider the signature schemes Ring-TESLA [ABB+16] and its recent improvement TESLA# [BLN+16]. As noted by the authors, there is a flaw in the security proof of Ring-TESLA and hence TESLA#, which still has to be fixed. We refer to [ABB+17, Table 3] for an overview and comparison of lattice-based signature schemes.

**Outline.** Section 2 gives the relevant notions and background required throughout this work. Section 3 describes our methodology for selecting parameters for any lattice-based scheme. Finally, Section 4 applies the methodology to LP, LARA, and BLISS-B and presents different parameters indicating the performance of the schemes and the (in)significant difference between sizes of keys/signatures/ciphertexts when selecting the parameters with and without considering the tightness gap of the respective security reduction.

## 2 Preliminaries

This section covers the necessary background required throughout this work. First, we give some general notation, define lattices, and the discrete Gaussian distribution, since it is a fundamental building block in lattice-based cryptography. In Section 2.1 we define the lattice problems underlying the cryptographic schemes considered in this work.

*Notation.* We let $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}$ denote the set of natural numbers, integers, rationals, and real numbers, respectively. The set of all prime numbers is denoted by $\mathbb{P}$. For a positive integer $k$, we let $[k]$ denote the set $\{1, 2, \ldots, k\}$. We denote column vectors with bold lower-case letters (e.g., $\mathbf{a}$). Furthermore, we denote matrices with bold upper-case letters (e.g., $\mathbf{A}$) and write $D^{n \times m}$ for the set of all $n \times m$ matrices over a domain $D$. For a positive integer $n$, we write $\mathbb{Z}_n$ to denote the set $\mathbb{Z}/n\mathbb{Z}$. The Euclidean norm ($\ell_2$-norm) of a vector $\mathbf{v}$ with entries $v_i$ is defined as $\|\mathbf{v}\| = (\sum_i |v_i|^2)^{1/2}$ and its $\ell_\infty$-norm as $\|\mathbf{v}\|_\infty = \max_i |v_i|$. For two vectors $\mathbf{v}, \mathbf{w}$ over some domain, we let $\langle \mathbf{v}, \mathbf{w} \rangle$ denote their inner product. For two bit strings $\mathbf{x}, \mathbf{y}$, we write $\mathbf{x} \oplus \mathbf{y}$ to denote the bitwise XOR operation on $\mathbf{x}, \mathbf{y}$ and $\mathbf{x}||\mathbf{y}$ to denote their concatenation. We write $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ for rounding up to the next integer and rounding down to the preceding one, respectively. All logarithms in this work are to base 2, i.e., $\log(\cdot) = \log_2(\cdot)$. The term *negligible* describes any function $f : \mathbb{N} \longrightarrow \mathbb{R}$ that decreases faster than the reciprocal of any polynomial $p$, i.e., there exists an $n_0 \in \mathbb{N}$ such that for all $n > n_0$, it holds $f(n) < \frac{1}{p(n)}$. With $\text{negl}(n)$ we denote a negligible function in $n$. The *statistical distance* between two distributions $X, Y$ over a countable domain $D$ is defined by the value $\frac{1}{2} \sum_{d \in D} |\operatorname{Prob}[X = d] - \operatorname{Prob}[Y = d]|$. Two distributions $X, Y$ indexed by a positive integer $n$ are called *statistically close* if their statistical distance is negligible in $n$. For some distribution $\mathcal{D}$ over some finite set $S$, we write $s \leftarrow \mathcal{D}$ to choose $s$ from $S$ according to the distribution $\mathcal{D}$ and we write $s \leftarrow_\$ S$ to choose $s$ uniformly at random from the set $S$. The uniform distribution over $S$ is denoted by $\mathcal{U}(S)$.

*Lattices.* A *lattice* is a discrete additive subgroup of the $n$-dimensional Euclidean vector space $\mathbb{R}^n$.

**Definition 2.1** (Lattice)**.** Let $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k \in \mathbb{R}^n$ be a set of linearly independent vectors, where $k \leq n$. The $n$-dimensional *lattice* $\mathcal{L}$ of rank $k$ that is generated by the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$ is the set of all integer linear combinations of $\mathbf{b}_1, \ldots, \mathbf{b}_k$, i.e.,

$$\mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_k) = \Big\{ \sum_{i=1}^{k} x_i \mathbf{b}_i \mid x_i \in \mathbb{Z} \Big\} \subset \mathbb{R}^n .$$

The vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$ are called a *basis* for the lattice $\mathcal{L}$ and they are given by a matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k] \in \mathbb{R}^{n \times k}$. Thus, we can write

$$\mathcal{L}(\mathbf{B}) = \{ \mathbf{B}\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^k \} \subset \mathbb{R}^n .$$

The *determinant* of a lattice $\mathcal{L}$, denoted by $\det(\mathcal{L})$, is given by $\sqrt{\det(\mathbf{B}^\top \cdot \mathbf{B})}$, where $\mathbf{B}$ is any basis of $\mathcal{L}$.

*Discrete Gaussian Distribution.* For any $r > 0$, the *Gaussian function* on $\mathbb{R}$ centered at $c \in \mathbb{R}$ is defined by $\rho_{r,c}(x) = \exp(-\pi \cdot \frac{(x-c)^2}{r^2})$, for all $x \in \mathbb{R}$. More generally, the Gaussian function on $\mathbb{R}^n$ centered at $\mathbf{c} \in \mathbb{R}^n$ is defined by

$$\begin{aligned} \rho_{r,\mathbf{c}} : \mathbb{R}^n &\longrightarrow (0, 1] \\ \mathbf{x} &\longmapsto \exp(-\pi \cdot \frac{\|\mathbf{x} - \mathbf{c}\|^2}{r^2}), \text{ for all } \mathbf{x} \in \mathbb{R}^n . \end{aligned}$$

The subscript $\mathbf{c}$ is taken to be $\mathbf{0}$ when omitted. The parameter $r$ is called the *Gaussian parameter*, and the standard deviation $\sigma$ is given by $\sigma = r/\sqrt{2\pi}$. The centered ($\mathbf{c} = \mathbf{0}$) *discrete Gaussian distribution* $D_{\mathcal{L},r}$ over a lattice $\mathcal{L}$ with Gaussian parameter $r$ is defined as follows: for every $\mathbf{x} \in \mathcal{L}$ the probability of $\mathbf{x}$ is given by $\rho_r(\mathbf{x})/\rho_r(\mathcal{L})$, where $\rho_r(\mathcal{L}) = \sum_{\mathbf{x} \in \mathcal{L}} \rho_r(\mathbf{x})$. We write $D_r$ to denote the centered discrete Gaussian distribution $D_{\mathbb{Z},r}$. Moreover, sampling an $n$-dimensional vector $\mathbf{x}$ from the discrete Gaussian distribution $D_{\mathbb{Z}^n,r}$ over the lattice $\mathbb{Z}^n$ is equivalent to independently sampling the coordinates of $\mathbf{x}$ from $D_r$, i.e., $D_{\mathbb{Z}^n,r} = D_r^n$. The same holds for $n$-degree polynomials via coefficient embedding.

## 2.1 Lattice Problems

In this section we describe two problems: the learning with errors (LWE) and the short integer solution (SIS) as well as their variants over rings. These are the main average-case problems underlying most modern lattice-based cryptosystems.

## Learning With Errors (LWE)

We define the learning with errors LWE problem and its ring variant. Furthermore, we define the hardness of LWE and briefly describe a method for estimating the hardness of (R-)LWE. The learning with errors problem was introduced by Regev [Reg05]. Regev's work provides amongst other results a reduction from the search to the decision version of LWE. In addition, it shows that solving the search version is at least as hard as quantumly solving certain lattice problems in the worst-case for certain moduli and Gaussian error distributions. There are many hardness results for LWE subsequent to Regev's work such as a classical reduction from worst-case lattice problems [Pei09], hardness with alternative error distribution, e.g., [MP13], and more (see [Pei16]). We only define the decision version of LWE and omit its search version, since the LWE-based schemes considered in this work were proven secure assuming the hardness of the decision version. First, we define the learning with errors distribution.

**Definition 2.2** (Learning With Errors (LWE) Distribution)**.** Let $n, q$ be positive integers, $\mathbf{s} \in \mathbb{Z}_q^n$, and $\chi$ be an error distribution over $\mathbb{Z}_q$. Define the distribution $A_{\mathbf{s},\chi}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing a vector $\mathbf{a} \leftarrow_{\$} \mathbb{Z}_q^n$ and an error $e \leftarrow \chi$, and then outputting the pair $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e \mod q)$.

In the LWE-based schemes considered in this work, the error distribution is the centered discrete Gaussian over $\mathbb{Z}$ with parameter $r = \alpha q$, where $\alpha \in (0, 1)$ is called the relative *error rate.* Therefore, in the following we define LWE with error distribution $\chi = D_{\alpha q}$.

**Definition 2.3** (Decision LWE)**.** Let $n, q$ be positive integers. Given $m = \mathrm{poly}(n)$ samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, the decision LWE problem asks to distinguish, with non-negligible advantage, whether the given samples were chosen according to the distribution $A_{\mathbf{s},D_{\alpha q}}$ or from the uniform distribution $\mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$, where $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$.

We note that the secret vector $\mathbf{s}$ in the above definition can also be selected from the error distribution [ACPS09]. An instance $\mathsf{LWE}_{n,m,q,r}$ of LWE can be described by a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a vector $\mathbf{b} \in \mathbb{Z}_q^m$, a modulus $q$, and a Gaussian parameter $r = \alpha q$. That is, the LWE samples are put together into $\mathbf{A}$ and $\mathbf{b}$, where the vectors $\mathbf{a}_i$ represent the columns of $\mathbf{A}$ and $b_i$ the entries of $\mathbf{b}$. Therefore, we can write

$$\mathsf{LWE}_{n,m,q,r} \in \Big\{ (\mathbf{A}, \mathbf{b}, q, r) : \ \mathbf{A} \in \mathbb{Z}_q^{n\times m},\ \mathbf{b} = \mathbf{A}^\top \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m,\ \mathbf{s} \in \mathbb{Z}_q^n,\ \mathbf{e} \leftarrow D_r^m,\ n, m, q \in \mathbb{N}, \\ r = \alpha q,\ \alpha \in (0, 1) \Big\} .$$

In the following we define decision R-LWE, i.e., the ring-based analogue of decision LWE. It was introduced by Lyubashevsky et al. [LPR10]. Their work shows that the decision version is as hard as the search version and provides a quantum reduction from worst-case hardness of problems on ideal lattices to the search version of R-LWE. Ideal lattices are lattices with additional algebraic structure that induces more efficient cryptographic constructions. Similar to LWE, we first define the R-LWE distribution.

**Definition 2.4** (R-LWE Distribution)**.** Let $R = \mathbb{Z}[x]/\langle f(x) \rangle$ be a polynomial ring for some $n$-degree polynomial $f(x)$ that is irreducible over $\mathbb{Z}$. Let $q$ be a positive integer modulus that defines the quotient ring $R_q = R/qR$. Further, let $s \in R_q$ and $\chi$ be an error distribution over $R_q$. Define the distribution $A_{s,\chi}$ over $R_q \times R_q$ obtained by choosing $a \leftarrow_{\$} R_q, e \leftarrow \chi$, and then outputting the pair $(a, \langle a, s \rangle + e \mod q)$.

As in LWE, the error distribution is typically $D_r^n$, where $r = \alpha q$. Moreover, the ring $R$ given in the above definition is commonly taken to be a cyclotomic ring, i.e., $f(x)$ is a cyclotomic polynomial such as $x^n + 1$ for $n$ a power of 2.

**Definition 2.5** (Decision R-LWE)**.** Let $n, q$ be positive integers. Given $m = \text{poly}(n)$ samples $(a_i, b_i) \in R_q \times R_q$, the decision R-LWE problem asks to distinguish, with non-negligible advantage, whether the given samples were chosen according to the distribution $A_{s,D_r^n}$ or from the uniform distribution $\mathcal{U}(R_q \times R_q)$, where $s \leftarrow_{\$} R_q$ (or $s \leftarrow D_r^n$).

We note that any instance of R-LWE can be seen as an instance of LWE with structured samples. We refer to [Pei16] for more details. A modified version of LWE, called *Augmented LWE* and denoted by A-LWE, was introduced by El Bansarkhani et al. [EDB15]. This work shows that A-LWE is for certain instantiations at least as hard as LWE. In fact, A-LWE samples are similar to LWE samples except that an arbitrary message is embedded into the error term while preserving its target distribution. Next, we define the hardness of decision LWE.

**Definition 2.6** (Hardness of Decision LWE)**.** Let $n, q, m$ be positive integers, $r = \alpha q$, where $\alpha \in (0, 1)$, and $D_r$ be the discrete Gaussian distribution with parameter $r$. Let $\mathcal{O}_r$ be an oracle, which upon input vector $\mathbf{s} \in \mathbb{Z}_q^n$ outputs samples from the LWE distribution $A_{\mathbf{s},D_r}$. The decision LWE problem is $(t_{\mathcal{D}}, \varepsilon_{\mathcal{D}})$-hard for an instance $\mathsf{LWE}_{n,m,q,r}$ if any polynomial time distinguisher $\mathcal{D}$ that runs in time at most $t_{\mathcal{D}}$ has advantage at most $\varepsilon_{\mathcal{D}}$ in distinguishing between $m$ samples from $\mathcal{O}_r$ and $m$ uniformly random samples from $\mathbb{Z}_q^n \times \mathbb{Z}_q$, i.e.,

$$\text{Adv}(\mathcal{D}) = \left| \text{Prob}[\mathcal{D}^{\mathcal{O}_r(\mathbf{s})}(\cdot) = 1] - \text{Prob}[\mathcal{D}^{\mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)}(\cdot) = 1] \right| \leq \varepsilon_{\mathcal{D}} \ ,$$

where the probabilities are taken over the choice of $\mathbf{s}$ from $\mathbb{Z}_q^n$ (uniformly random or from $D_r^n$) and the random coins used by $\mathcal{D}$.

We note that the hardness of decision R-LWE is defined similarly. Finally, estimating the hardness of (R-)LWE instances can be performed by using the *LWE estimator* presented by Albrecht et al. [APS15]. This estimator is a Sage module, which estimates the running time of currently existing LWE algorithms on given parameters $n, q, \alpha$, and $m$. We note that current attacks do not exploit the ring structure of R-LWE. Hence, its hardness is analyzed as LWE.

## Short Integer Solution (SIS)

We define the short integer solution (SIS) problem, its ring variant R-SIS, and describe how the hardness of (R-)SIS can be estimated. The SIS problem originally was presented by Ajtai [Ajt96]. Subsequently, several reductions from worst-case lattice problems to the average-case of SIS have been proposed improving the one by Ajtai, e.g., the reduction shown by Micciancio and Peikert [MP13]. The ring-based analogue R-SIS was introduced by Micciancio [Mic02]. The most recent work by Peikert and Rosen [PR07] proves the hardness of R-SIS assuming worst-case hardness of problems on ideal lattices with significantly better connection factors than previous results. We refer to [Pei16] for more details and hardness results for (R-)SIS.

**Definition 2.7** (Short Integer Solution (SIS) Problem)**.** Let $n, q, m$ be positive integers and $\beta$ a positive real. Given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, the SIS problem asks to find a non-zero integer vector $\mathbf{x} \in \mathbb{Z}^m$ such that

$$\mathbf{A}\mathbf{x} \equiv \mathbf{0} \pmod q, \quad \text{where} \quad \|\mathbf{x}\| \leq \beta \ .$$

An instance $\mathsf{SIS}_{n,m,q,\beta}$ of $\mathsf{SIS}$ is described by a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a modulus $q$, and a norm bound $\beta$. Therefore, we can write

$$\mathsf{SIS}_{n,m,q,\beta} \in \left\{ (\mathbf{A}, q, \beta) : \ \mathbf{A} \in \mathbb{Z}_q^{n\times m}, \ n,m,q \in \mathbb{N}, \ n \leq m, \ \beta > 0 \right\} .$$

The R-SIS is defined similarly over rings and it can be treated as a special case of SIS with structured instances.

**Definition 2.8** (R-SIS Problem)**.** Let $n, q, m$ be positive integers and $\beta$ a positive real. Further, let $R = \mathbb{Z}[x]/\langle f(x)\rangle$ be a polynomial ring for an $n$-degree polynomial $f \in \mathbb{Z}[x]$ and $R_q = R/qR$ its quotient ring. Given a uniformly random vector $(a_1, \ldots, a_m) \in R_q^m$, the R-SIS problem asks to find a non-zero vector $\mathbf{x} = (x_1, \ldots, x_m) \in R^m$ such that

$$\sum_{i=1}^{m} a_i x_i \equiv 0 \pmod q, \quad \text{where} \quad \|\mathbf{x}\| \leq \beta \ .$$

Next, we explain hardness estimation of (R-)SIS instances. The hardness of a (R-)SIS instance is usually estimated by measuring the running time of lattice basis reduction algorithms. Lattice reduction finds lattice bases consisting of reasonably short and nearly orthogonal vectors. The experiments of Gama and Nguyen [GN08] show that reduction algorithms on an $n$-dimensional lattice $\mathcal{L}$ find vectors of length $\leq \delta^n \cdot \det(\mathcal{L})^{1/n}$, where $\delta$ is a parameter called the *Hermite delta*, which depends on the output quality of the reduction algorithm being used. In this work, we consider the widely used reduction algorithm BKZ 2.0 [CN11]. As a subroutine, BKZ 2.0 requires solving the well-known shortest vector problem (SVP) in lattices of dimension at most $b$, where $b \leq n$ is called the *block size*. Furthermore, BKZ 2.0 proceeds in multiple rounds for its final output. Therefore, given the norm bound $\beta$ of an (R-)SIS instance, the corresponding Hermite delta $\delta$ can be specified by setting $\beta = \delta^n \cdot \det(\mathcal{L})^{1/n}$. Then, the running time of BKZ 2.0 required to achieve a basis of quality $\delta$ is estimated. There exists different methods to estimate the running time of BKZ 2.0, e.g., [CN11, ACF+15, APS15]. We use the following approach: Given a Hermite delta $\delta$, we proceed following [APS15, Wun16, GvVW17] and use the relation

$$\delta = \left( \frac{b \cdot (\pi b)^{\frac{1}{b}}}{2\pi e} \right)^{\frac{1}{2(b-1)}}$$

according to Chen [Che13] in order to determine the (minimal) block size $b$ required to achieve $\delta$. Since the needed number of rounds of BKZ 2.0 with block size $b$ can be reduced by applying progressive strategies (see [Che13, AWHT16]), we conservatively assume that this number of rounds can be brought down to 1. Furthermore, we do not consider running BKZ 2.0 with block sizes smaller than $b$, since its time is significantly lower. The total running time $T_{\mathsf{BKZ}}$ of BKZ 2.0 is then estimated due to [APS15] as follow:

$$\log(T_{\mathsf{BKZ}}) = 0.187281 \cdot b \cdot \log b - 1.0192 \cdot b + \log(n - b + 1) + 16.1 \ .$$

## 3 How to Select Parameters for Lattice-Based Schemes

In this section, we present a methodology for selecting parameters for lattice-based schemes. To this end, we first describe the building blocks required for our framework. In Section 3.1 and 3.2, we explain how a lattice-based scheme and its underlying problem can be characterized. Then, we describe security reductions of (lattice-based) cryptographic schemes in Section 3.3 and the types of parameters than can be chosen for any lattice-based scheme in Section 3.4. Finally, we present our methodology in Section 3.5.

### 3.1 Cryptographic Schemes

This section explains the tools required to describe a lattice-based scheme including parameters and bit security. For the rest of this section, we write $\mathcal{S}$ to denote a cryptographic scheme whose security is based on the hardness of a lattice problem. We consider any scheme $\mathcal{S}$ to be a system designed to perform certain cryptographic tasks. These tasks are carried out by a tuple of polynomial-time algorithms. One of these algorithms is designed to generate public and/or private elements, which are necessary to run the remaining algorithms. Any scheme $\mathcal{S}$ is parametrized by a finite *sequence of parameters*[1]. Parameters (or a part of them) define the domain of the public/private elements and they also serve as input to the respective algorithms that define $\mathcal{S}$. We denote the public/private elements together with the parameters by an *instance* of $\mathcal{S}$.

**Example 3.1.** A public-key encryption scheme $\Pi$ consists of three polynomial-time algorithms called key generation, encryption, and decryption. An instance of $\Pi$ consists of a sequence of parameters and a (public, private) key pair generated from the key generation algorithm. The public key is required to perform encryption, whereas decryption is accomplished by using the private key. If the instance includes, for example, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, then $n, m$, and $q$ are contained in the parameter sequence of $\Pi$. In Appendix A.1 and A.2, we formally give the definition of public-key encryption and digital signature schemes.

Any sequence of parameters includes a natural number $\ell$ called the *security parameter*, which is used to generate instances of $\mathcal{S}$. In addition, the bit security of $\mathcal{S}$ is expressed in terms of $\ell$ (see Definition 3.1). It is meaningful for choosing parameters to only consider parameters satisfying the following conditions:

- *Correctness* (or an acceptable percentage of correctness), which is given in terms of relations on the parameters. A key generation algorithm, for instance, may require some restrictions ensuring validity of keys, while a probability of decryption errors in a decryption algorithm could be made very small with an appropriate setting of parameters.
- Reasonable level of *efficiency* such as requirements on key sizes and optimized implementations for high performance.
- *Security* conditions imposed by known attacks and security proofs under certain security notions.

For the sake of parameter selection, we refer to the set of all sequences of parameters that satisfy the requirements described above as the set of *admissible parameters* of $\mathcal{S}$ and denote it by $\mathbf{Para}_{\mathcal{S}}$.

[1] Usually, it is called a set of parameters. We choose to use the term "sequence" in order to emphasize arrangement of the parameters. For example, if the parameters $(n, m, q)$ are given by $(512, 1024, 12289)$, then it should be clear that $n = 512, m = 1024$, and $q = 12289$.

**Example 3.2.** We consider the ring variant of the lattice-based public-key encryption scheme $\mathsf{LP}$ proposed by Lindner and Peikert [LP11]. This variant is parametrized by the sequence of parameters $(\ell, n, q, r, l, t)$, where $\ell \in \mathbb{N}$ is the security parameter, $n \in \mathbb{N}$ the lattice dimension, $q \in \mathbb{N}$ a modulus, $r \in \mathbb{R}$ a Gaussian parameter, $l \in \mathbb{N}$ the message length being encrypted, and $t \in \mathbb{N}$ is an error threshold. In order to obtain a negligible probability of decryption errors, parameters are restricted by conditions that upper-bound this probability. For efficiency reasons, the dimension $n$ is typically taken to be a power of 2. We refer to Section 4.1 for more details.

Any sequence of parameters from $\mathbf{Para}_{\mathcal{S}}$ provides a certain bit security, which is defined as follows:

**Definition 3.1** (Bit Security)**.** Let $\mathcal{S}$ be a cryptographic scheme and $\ell \in \mathbb{N}$. A sequence of parameters $\mathsf{para}_{\mathcal{S}} \in \mathbf{Para}_{\mathcal{S}}$ is called *$\ell$-bit secure* if $\ell$ is the largest natural number such that $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \geq 2^{\ell}$ for any adversary $\mathcal{A}$ that breaks any instance of $\mathcal{S}$ that includes $\mathsf{para}_{\mathcal{S}}$ within time $t_{\mathcal{A}}$ and success probability $\varepsilon_{\mathcal{A}}$.

Given a scheme $\mathcal{S}$, our goal is to find a general function $\mathbf{Para}_{\mathcal{S}} \to \mathbb{N}$ that maps any sequence from $\mathbf{Para}_{\mathcal{S}}$ to a security level. We describe how to determine such a function in Section 3.3 for any lattice-based scheme. In Section 4, we explicitly specify this function for the target lattice-based schemes $\mathsf{LP}$ [LP11], $\mathsf{LARA}$ [EB17], and $\mathsf{BLISS\text{-}B}$ [DDLL13, Duc14].

## 3.2 Hard Problems

In this section, we formally define the bit hardness of computational problems underlying cryptographic schemes. A computational problem $\mathcal{P}$ is said to be *hard* if there is no algorithm that efficiently finds a solution to $\mathcal{P}$. An instance of $\mathcal{P}$ defines the required input of an algorithm that solves $\mathcal{P}$ and any instance is parametrized by a finite sequence of parameters. We denote the set of all these sequences by $\mathbf{Para}_{\mathcal{P}}$. Any sequence of parameters provides a certain bit hardness, which is defined as follows:

**Definition 3.2** (Bit Hardness)**.** Let $\mathcal{P}$ be a computational problem and $\ell \in \mathbb{N}$. A sequence of parameters $\mathsf{para}_{\mathcal{P}} \in \mathbf{Para}_{\mathcal{P}}$ is called *$\ell$-bit hard* if $\ell$ is the largest natural number such that $\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \geq 2^{\ell}$ for any algorithm $\mathcal{D}$ that solves any instance of $\mathcal{P}$ parametrized by $\mathsf{para}_{\mathcal{P}}$ within time $t_{\mathcal{D}}$ and success probability $\varepsilon_{\mathcal{D}}$.

**Example 3.3.** We consider the $\mathsf{LWE}$ problem (see Section 2.1). Any instance of $\mathsf{LWE}$ is parametrized by a sequence of parameters $(n, m, q, r) \in \mathbf{Para}_{\mathsf{LWE}}$, where

$$\mathbf{Para}_{\mathsf{LWE}} = \left\{(n, m, q, r) : \ n, m, q \in \mathbb{N}, \ r = \alpha q, \ \alpha \in (0, 1)\right\}.$$

Estimating the bit hardness of any parameter sequence of a problem $\mathcal{P}$ can be defined by an *estimator function* $\mathsf{Est}_{\mathcal{P}}$. This functions maps a sequence of parameters $\mathsf{para}_{\mathcal{P}} \in \mathbf{Para}_{\mathcal{P}}$ to a hardness level $\ell \in \mathbb{N}$, i.e.,

$$\begin{aligned} \mathsf{Est}_{\mathcal{P}} : \ \mathbf{Para}_{\mathcal{P}} &\longrightarrow \mathbb{N} \\ \mathsf{para}_{\mathcal{P}} &\longmapsto \ell \end{aligned} \tag{1}$$

**Example 3.4.** We define the function $\mathsf{Est}_{\mathsf{LWE}}$ for the $\mathsf{LWE}$ problem as follows:

$$\begin{aligned} \mathsf{Est}_{\mathsf{LWE}} : \ \mathbf{Para}_{\mathsf{LWE}} &\longrightarrow \mathbb{N} \\ (n, m, q, r) &\longmapsto \ell \end{aligned}$$

where $\mathsf{Est}_{\mathsf{LWE}}$ can, for example, be instantiated by the LWE estimator [APS15].

### 3.3 Security Reductions

In this section, we describe how security proofs of cryptographic schemes work, and then define the functions required for our framework. In particular, we show how security proofs in lattice-based cryptography stem from worst-case hardness of lattice problems. Furthermore, given a lattice-based scheme $\mathcal{S}$, we construct a general function that maps parameter sequences of $\mathcal{S}$ to security levels.

Let $\mathcal{S}$ be a cryptographic scheme whose security is based on the hardness of a lattice problem $\mathcal{P}$. A security proof for $\mathcal{S}$ is a *reduction* algorithm $\mathcal{D}$ that turns an adversary $\mathcal{A}$ against $\mathcal{S}$, which runs in time at most $t_{\mathcal{A}}$ and breaks $\mathcal{S}$ with probability at least $\varepsilon_{\mathcal{A}}$, into a solver for the problem $\mathcal{P}$ that runs in time at most $t_{\mathcal{D}}$ and has success probability at least $\varepsilon_{\mathcal{D}}$. In other words, a security proof reduces solving $\mathcal{P}$ to breaking $\mathcal{S}$ with respect to a certain security notion. This means that the reduction $\mathcal{D}$ executes the adversary $\mathcal{A}$ as a subroutine and makes some extra steps in order to solve $\mathcal{P}$. Consequently, we can write

$$c_1\varepsilon_{\mathcal{A}} + c_2 \leq \varepsilon_{\mathcal{D}} \leq \varepsilon_{\mathcal{A}} \quad \text{and} \quad t_{\mathcal{A}} \leq t_{\mathcal{D}} \leq c_3 t_{\mathcal{A}} + c_4$$

for some fixed polynomials (in the security parameter of $\mathcal{S}$) $c_1, c_2, c_3$, and $c_4$. The quality of a reduction is evaluated by the quantity $\gamma = \frac{t_{\mathcal{D}}\cdot\varepsilon_{\mathcal{A}}}{t_{\mathcal{A}}\cdot\varepsilon_{\mathcal{D}}} \geq 1$, which is called the *tightness gap* [CKMS16]. It is desirable that $t_{\mathcal{A}} \approx t_{\mathcal{D}}$ and $\varepsilon_{\mathcal{A}} \approx \varepsilon_{\mathcal{D}}$, i.e., $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \approx \frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}}$. The reduction is then called *tight*, i.e., if the tightness gap is a small constant [CKMS16]. This means that breaking $\mathcal{S}$ implies solving $\mathcal{P}$ with almost the same complexity. The reduction is *non-tight* if $t_{\mathcal{A}} \ll t_{\mathcal{D}}$ or $\varepsilon_{\mathcal{A}} \gg \varepsilon_{\mathcal{D}}$ [KM06]. As a result, a security proof reveals a *reduction loss* (or *reduction gap*) in bits, which indicates the difference (gap) between the bit security of $\mathcal{S}$ and the bit hardness of $\mathcal{P}$ according to the given reduction. We quantify this reduction loss by writing

$$\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} = \gamma \cdot \frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \ . \tag{2}$$

The reduction loss is then given by $\log \gamma$, i.e., the reduction loss (reduction gap) is the logarithm (to base 2) of the tightness gap $\gamma$. If it is difficult to explicitly determine the tightness gap, then it is interesting to find out how large the reduction loss in tight reductions can be, and how small it can be in non-tight reductions. Therefore, we give an upper bound on the reduction loss in tight security reductions and a lower bound in non-tight reductions of the lattice-based schemes we consider in Section 4.

Next, we describe security reductions in lattice-based cryptography. Most modern lattice-based schemes come with a security reduction from the average-case hardness of the underlying problems such as LWE and SIS. The average-case hardness of these problems relies on reductions from the worst-case hardness of lattice problems such as the well-known shortest vector problem. A worst-case to average-case reduction is very similar to the one described above for any scheme, i.e., it reduces the hardness of solving worst-case instances of a lattice problem to the hardness of solving average-case instances of the underlying problem (see Figure 2). As stated in Section 1, we analyze in Section 4 only reductions from the average-case hardness of R-LWE and R-SIS to the security of the target lattice-based schemes.

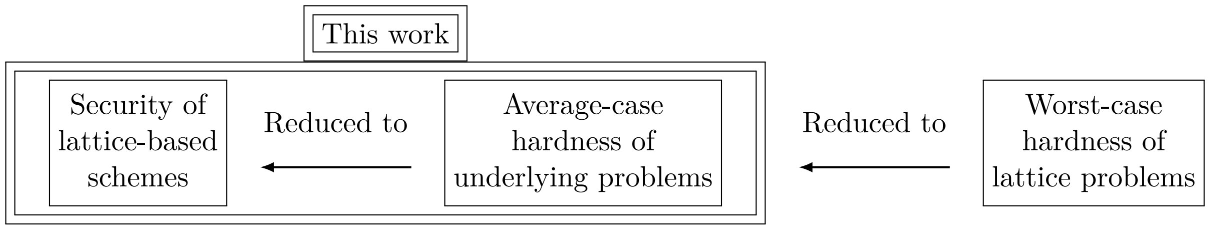
This work
Security of lattice-based schemes
Reduced to
Average-case hardness of underlying problems
Reduced to
Worst-case hardness of lattice problems

Figure 2: Security reductions of lattice-based schemes.

**Functions Specifying Security Reductions**

We construct a function that maps parameter sequences of a lattice-based scheme $\mathcal{S}$ to security levels. To this end, we need two functions that characterize a security reduction from the hardness of a lattice problem $\mathcal{P}$ to the security of $\mathcal{S}$. We define these two functions as follows:

1. A *reduction function* $\mathsf{Red}_{\mathcal{S}}$, which maps any sequence from the set of admissible parameters $\mathbf{Para}_{\mathcal{S}}$ to a sequence of parameters from $\mathbf{Para}_{\mathcal{P}}$, i.e.,

$$\begin{aligned} \mathsf{Red}_{\mathcal{S}}:\ \mathbf{Para}_{\mathcal{S}} &\longrightarrow \mathbf{Para}_{\mathcal{P}} \\ \mathsf{para}_{\mathcal{S}} &\longmapsto \mathsf{para}_{\mathcal{P}}\ . \end{aligned} \tag{3}$$

   The reduction function $\mathsf{Red}_{\mathcal{S}}$ relates the parameters of $\mathcal{S}$ to parameters of $\mathcal{P}$ by means of specific conditions and dependencies that must hold in order for the security reduction to be satisfied, i.e., breaking any instance of $\mathcal{S}$ including a sequence $\mathsf{para}_{\mathcal{S}}$ implies solving an instance of $\mathcal{P}$ parametrized by $\mathsf{para}_{\mathcal{P}} = \mathsf{Red}_{\mathcal{S}}(\mathsf{para}_{\mathcal{S}})$. If the reduction consists of combined reductions for intermediate problems, then $\mathsf{Red}_{\mathcal{S}}$ is specified by composing the intermediate reduction functions.

2. A *gap function* $\mathsf{Gap}_{\mathcal{S}}$, which maps a bit hardness of the underlying lattice problem $\mathcal{P}$ to a security level of $\mathcal{S}$ according to the security reduction, i.e.,

$$\begin{aligned} \mathsf{Gap}_{\mathcal{S}}:\ \mathbb{N}_{>\log\gamma} &\longrightarrow \mathbb{N} \\ \ell &\longmapsto \ell - \lceil \log\gamma \rceil\ , \end{aligned} \tag{4}$$

   where $\log\gamma$ is the reduction loss (in bits) described above. We point out that the hardness level $\ell$ must be larger than $\log\gamma$. Otherwise, the reduction offers no security. Note that in order to specify the gap function $\mathsf{Gap}_{\mathcal{S}}$ it is essential to deduce the running time $t_{\mathcal{D}}$ and success probability $\varepsilon_{\mathcal{D}}$, which are given in terms of the running time $t_{\mathcal{A}}$ and success probability $\varepsilon_{\mathcal{A}}$ of the adversary $\mathcal{A}$, respectively. If the reduction is composed of intermediate reductions, then the function $\mathsf{Gap}_{\mathcal{S}}$ is defined similarly, where $\log\gamma$ in this case is given by adding the reduction loss of the intermediate reductions together.

Consequently, our desired function that maps any sequence of parameters $\mathsf{para}_{\mathcal{S}} \in \mathbf{Para}_{\mathcal{S}}$ to a security level $\ell$ is given by the composition map $\mathsf{Gap}_{\mathcal{S}} \circ \mathsf{Est}_{\mathcal{P}} \circ \mathsf{Red}_{\mathcal{S}}$ (see Figure 3). The following lemma shows that this composition map indeed offers the desired security level $\ell$.

**Lemma 3.1.** *The composition map $\mathsf{Gap}_{\mathcal{S}} \circ \mathsf{Est}_{\mathcal{P}} \circ \mathsf{Red}_{\mathcal{S}}$ illustrated in Figure 3 is correct.*

$$\begin{tikzcd}
\mathbf{Para}_{\mathcal{S}} \arrow[r, "\mathsf{Red}_{\mathcal{S}}"] \arrow[d, dashed, "\mathsf{Gap}_{\mathcal{S}} \circ \mathsf{Est}_{\mathcal{P}} \circ \mathsf{Red}_{\mathcal{S}}"'] & \mathbf{Para}_{\mathcal{P}} \arrow[d, "\mathsf{Est}_{\mathcal{P}}"] \\
\mathbb{N} & \mathbb{N}_{>\log\gamma} \arrow[l, "\mathsf{Gap}_{\mathcal{S}}"]
\end{tikzcd}$$

Figure 3: Construction of the composition function $\mathsf{Gap}_{\mathcal{S}} \circ \mathsf{Est}_{\mathcal{P}} \circ \mathsf{Red}_{\mathcal{S}}$, which maps parameter sequences of a scheme $\mathcal{S}$ to security levels. A sequence of parameters $\mathsf{para}_{\mathcal{S}} \in \mathbf{Para}_{\mathcal{S}}$ maps via $\mathsf{Red}_{\mathcal{S}}$ to a parameter sequence $\mathsf{para}_{\mathcal{P}} \in \mathbf{Para}_{\mathcal{P}}$ of a problem $\mathcal{P}$ following the security reduction of $\mathcal{S}$. The bit hardness of the sequence $\mathsf{para}_{\mathcal{P}}$ is then estimated according to the map $\mathsf{Est}_{\mathcal{P}}$. The resulting bit hardness is then evaluated under the function $\mathsf{Gap}_{\mathcal{S}}$, which represents the reduction loss induced by the security reduction.

*Proof.* According to Definition 3.1, the sequence $\mathsf{para}_{\mathcal{S}} \in \mathbf{Para}_{\mathcal{S}}$ to be selected has to satisfy $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \geq 2^{\ell}$. This is performed by choosing $\mathsf{para}_{\mathcal{S}}$ such that the sequence $\mathsf{Red}_{\mathcal{S}}(\mathsf{para}_{\mathcal{S}}) \in \mathbf{Para}_{\mathcal{P}}$ is $(\ell + \lceil \log\gamma \rceil)$-bit hard. This means $\mathsf{Est}_{\mathcal{P}}(\mathsf{Red}_{\mathcal{S}}(\mathsf{para}_{\mathcal{S}})) = \ell + \lceil \log\gamma \rceil$. More precisely, using equation (2) and Definition 3.2 we obtain

$$\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} = \frac{1}{\gamma} \cdot \frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \geq \frac{1}{\gamma} \cdot 2^{\ell + \lceil \log\gamma \rceil} = 2^{-\log\gamma} \cdot 2^{\ell + \lceil \log\gamma \rceil} \geq 2^{-\lceil \log\gamma \rceil} \cdot 2^{\ell + \lceil \log\gamma \rceil} = 2^{\ell} .$$

□

In Section 3.5, we stepwise show how to select parameters with respect to the functions $\mathsf{Red}_{\mathcal{S}}, \mathsf{Est}_{\mathcal{P}}$, and $\mathsf{Gap}_{\mathcal{S}}$.

## 3.4 Parameter Types for Lattice-Based Schemes

In this section, we define the types of parameters which can be selected for lattice-based schemes. Let $\mathcal{S}$ be a cryptographic scheme whose security is based on the hardness of a lattice problem $\mathcal{P}$. The goal is to select parameters for $\mathcal{S}$ offering some security level $\ell$. In the next section, we show how this is accomplished. In the following we define two types of parameters that are strongly related to security reductions.

1. *Provably secure*: These parameters are chosen according to an available security reduction of $\mathcal{S}$ as illustrated in Figure 3. In particular, the reduction loss is deduced and the functions $\mathsf{Red}_{\mathcal{S}}, \mathsf{Gap}_{\mathcal{S}}$ defined in expressions (3) and (4) are processed properly.

2. *Non-provably secure*: Regardless of whether $\mathcal{S}$ is provided with a proof of security, parameters may still be selected without involving the gap induced by the security reduction. More precisely, one assumes that the desired security level $\ell$ is identical to the hardness level of the underlying problem. Hence, the gap function $\mathsf{Gap}_{\mathcal{S}}$ is set to be the identity. Non-provably secure parameters can be chosen following one of the below approaches:

   a. Identify the best known attack $\mathcal{A}$ on $\mathcal{S}$ and select parameters satisfying $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \geq 2^{\ell}$, where $t_{\mathcal{A}}, \varepsilon_{\mathcal{A}}$ are the running time and success probability of $\mathcal{A}$, respectively. This method is mostly carried out when $\mathcal{S}$ lacks a security proof. Note that this approach can also be involved in selecting provably secure parameters when $\mathcal{S}$ admits direct attacks.

b. Determine the best known solver $\mathcal{D}$ for the underlying problem $\mathcal{P}$ and ensure that its corresponding parameters offer a hardness of $\ell$ bits. That is, $\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \geq 2^{\ell}$, where $t_{\mathcal{D}}, \varepsilon_{\mathcal{D}}$ are the running time and success probability of $\mathcal{D}$. This method can be performed either if the security of $\mathcal{S}$ is believed to be based on the hardness of $\mathcal{P}$ (with no known security proof), or even if $\mathcal{S}$ has a security reduction. The latter can be done by either specifying the reduction function $\mathsf{Red}_{\mathcal{S}}$ properly and setting the gap function to the identity or disregarding both functions $\mathsf{Red}_{\mathcal{S}}$ and $\mathsf{Gap}_{\mathcal{S}}$.

c. Combine the approaches (a.) and (b.). This means that the parameters to be selected satisfy $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \geq 2^{\ell}$ and $\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \geq 2^{\ell}$ for any attacker $\mathcal{A}$ against $\mathcal{S}$ and any solver $\mathcal{D}$ for $\mathcal{P}$.

## 3.5 The Methodology

Finally, we describe our methodology for selecting parameters for any lattice-based scheme $\mathcal{S}$:

1. Choose a reasonable security level $\ell$.
2. Identify the security reduction for $\mathcal{S}$ and the underlying problems $\mathcal{P}_i$. If available, the reduction is mostly given by means of a theorem, which states the proof of security.
3. If exist, identify the best known attack $\mathcal{A}$ that is directly applicable on $\mathcal{S}$ in addition to its running time $t_{\mathcal{A}}$ and success probability $\varepsilon_{\mathcal{A}}$.
4. Specify the sets $\mathbf{Para}_{\mathcal{P}_i}$ for each $\mathcal{P}_i$ and the set of admissible parameters $\mathbf{Para}_{\mathcal{S}}$. This includes identifying the conditions and restrictions on the parameters that ensure correctness, efficiency, and security. Conditions for correctness are usually given along with the description of $\mathcal{S}$. Requirements for efficiency may be given separately. Security conditions are usually given with the security reduction and can be deduced from known attacks on $\mathcal{S}$, if necessary.
5. Derive the reduction function $\mathsf{Red}_{\mathcal{S}}$ defined in expression (3) by using the security reduction identified in Step 2.

   If $\mathcal{S}$ has no known security reduction, then define $\mathsf{Red}_{\mathcal{S}}$ to be a function, which simply maps any sequence of parameters from $\mathbf{Para}_{\mathcal{S}}$ to the Cartesian product of the respective parameter sequences from $\mathbf{Para}_{\mathcal{P}_i}$.
6. For each of the underlying problems $\mathcal{P}_i$ identified in Step 2, perform the following:

   6.1. Derive the gap function $\mathsf{Gap}_{\mathcal{S}}$ defined in expression (4). This requires determining the running time and success probability of the reduction if they are not explicitly given.

   If $\mathcal{S}$ has no known security reduction, then set the gap function to be the identity.

   6.2. Specify the estimator function $\mathsf{Est}_{\mathcal{P}}$ defined in expression (1), i.e., select a method that allows to estimate the bit hardness of any given sequence of parameters from $\mathbf{Para}_{\mathcal{P}_i}$.
7. Set the actual reduction loss to $\lceil \log \gamma \rceil = \max_i \{\lceil \log \gamma_i \rceil\}$, where $\lceil \log \gamma_i \rceil$ correspond to the $\mathsf{Gap}_{\mathcal{S}}$ functions derived in Step 6.1.. This is important, since breaking $\mathcal{S}$ implies solving at least one of the underlying problems. Hence, the security of $\mathcal{S}$ is restricted by the hardness of the problem that implies the largest tightness gap.

   If $\mathcal{S}$ has no known security reduction, then set $\gamma = 1$, i.e., $\lceil \log \gamma \rceil = 0$.

8. Select a sequence of parameters $\mathsf{para}_{\mathcal{S}} \in \mathbf{Para}_{\mathcal{S}}$ such that for each $\mathcal{P}_i$ the sequence of parameters $\mathsf{Red}_{\mathcal{S}}(\mathsf{para}_{\mathcal{S}})$ is $(\ell + \lceil \log \gamma \rceil)$-bit hard, i.e., $\mathsf{Est}_{\mathcal{P}}(\mathsf{Red}_{\mathcal{S}}(\mathsf{para}_{\mathcal{S}})) = \ell + \lceil \log \gamma \rceil$ (see Figure 3). This is accomplished by either starting with finding sequences $\mathsf{para}_{\mathcal{P}_i} \in \mathbf{Para}_{\mathcal{P}_i}$ that are $(\ell + \lceil \log \gamma \rceil)$-bit hard and then selecting the remaining parameters of $\mathcal{S}$, or the other way around.

   If apply, then the sequence $\mathsf{para}_{\mathcal{S}} \in \mathbf{Para}_{\mathcal{S}}$ must also satisfy $\frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \geq 2^{\ell}$, where $\mathcal{A}$ is the best known attack on $\mathcal{S}$ identified in Step 3.

## 4 Concrete Parameters for Lattice-Based Schemes

In this section we use our methodology described in Section 3 in order to select parameters for the public-key encryption schemes LP [LP11] (Section 4.1) and LARA [EB17] (Section 4.2) and for the digital signature scheme BLISS-B [DDLL13, Duc14] (Section 4.3). Therefore, we analyze the security proof of each scheme in order to derive the running time and success probability of the respective reduction and then deduce the reduction gap. For the tight security reductions of LP and LARA we obtain a gap of at most 3 and 5 bits, respectively. The non-tight security reduction of BLISS-B induces a gap of at least 63 bits according to our analysis. For each scheme, we propose two sequences of parameters that satisfy 128 bits of security (near-term security) and two sequences for 256 bits of security (long-term security). Given $\ell \in \{128, 256\}$ the first sequence of parameters provably satisfies the security level $\ell$ in accordance to the security proof of each scheme, whereas the second sequence is non-provably $\ell$-bit secure (see Section 3.4), i.e., parameters are selected such that the underlying problem is $\ell$-bit hard and the reduction loss induced by the security proof of each scheme is not taken into account. The sequences of parameters are given in Tables 2, 4, and 7, which show the corresponding bit sizes of keys, signatures, messages, and ciphertexts. Furthermore, the tables include timings for signing/verifying and encryption/decryption. Measurements of the running time are carried out on a machine specified by an Intel Core i7-6500U processor operating at 2×2.5GHz and 8GB of RAM. The benchmarks of our proposed parameters are averaged over 10,000 runs of the respective algorithms.

### 4.1 The Encryption Scheme by Lindner and Peikert

We consider the ring variant of the public-key encryption scheme LP proposed by Lindner and Peikert [LP11] (see Appendix B.1 for its description). It is IND-CPA-secure (see Appendix A.1) and its security is based on the hardness of decision R-LWE, where the public key and ciphertext represent two instances of R-LWE. The security proof of LP is given in [LP11, Theorem 3.2]. In the following theorem, we state the security proof of the ring variant of LP, in which the same Gaussian parameter is used for both key generation and encryption.

**Theorem 4.1** (Variant of [LP11, Theorem 3.2]). *The public-key encryption scheme LP is IND-CPA-secure, assuming the hardness of decision R-LWE for the instances* $\mathsf{R\text{-}LWE}_{1,1,q,r}$ *and* $\mathsf{R\text{-}LWE}_{1,2,q,r}$.

Next, we specify the sets $\mathbf{Para}_{\mathsf{LP}}$ and $\mathbf{Para}_{\mathsf{LWE}}$, where $\mathbf{Para}_{\mathsf{LP}}$ is the set of admissible parameters for LP and $\mathbf{Para}_{\mathsf{LWE}}$ is the set of all sequences of parameters for (R-)LWE (see Section 3.2). A sequence of parameters from $\mathbf{Para}_{\mathsf{LP}}$ is given by $(\ell, n, q, r, l, t)$, where $\ell$ is the security parameter, $n$ the dimension of R-LWE secrets, $q$ a modulus, $r$ a Gaussian parameter, $l$ the message length, and $t$ an error threshold for encoding and decoding of messages. Collecting all the restrictions on the parameters provided in [LP11] for correctness, efficiency, and security leads to the following set:

$$\mathbf{Para}_{\mathsf{LP}} = \Bigg\{ (\ell, n, q, r, l, t) : \ \ell \in \mathbb{N}, \ n = 2^k, \ k \in \mathbb{N}, \ q \in \mathbb{Z}, \ r = \alpha q, \ \alpha \in (0,1), \ r \geq 8, \ l \in \mathbb{N}, \ l \leq n, \\ t \in \mathbb{Z}_{\geq 1}, \ r^2 \leq \frac{\sqrt{2}\pi t}{c \cdot \sqrt{2n \ln(2/\delta)}}, \ c \in \mathbb{R}_{\geq 1}, \ \delta \in [0,1], \ \left( c \cdot \exp\left( \frac{1-c^2}{2} \right) \right)^{2n} = 2^{-40} \Bigg\} .$$

The set $\mathbf{Para}_{\mathsf{LWE}}$ has already been specified in Section 3.2 as follows:

$$\mathbf{Para}_{\mathsf{LWE}} = \Big\{ (n, m, q, r) : \ n, m, q \in \mathbb{N}, \ r = \alpha q, \ \alpha \in (0,1) \Big\} .$$

We recall from Section 2.1 that the description and hardness estimation of instances of LWE and R-LWE is performed in a similar way. We proceed with deriving the reduction function $\mathsf{Red}_{\mathsf{LP}}$ for LP (see Section 3, expression (3)). Theorem 4.1 implies the following reduction function:

$$\begin{aligned} \mathsf{Red}_{\mathsf{LP}} : \ \mathbf{Para}_{\mathsf{LP}} &\longrightarrow \mathbf{Para}_{\mathsf{LWE}} \times \mathbf{Para}_{\mathsf{LWE}} \\ (\ell, n, q, r, l, t) &\longmapsto \Big( (n, n, q, r), (n, 2n, q, r) \Big) , \end{aligned}$$

where the instances $\mathsf{R\text{-}LWE}_{1,1,q,r}$ and $\mathsf{R\text{-}LWE}_{1,2,q,r}$ transform to $\mathsf{LWE}_{n,n,q,r}$ and $\mathsf{LWE}_{n,2n,q,r}$.

## Analyzing the Security Reduction

Next, we analyze the proof of the IND-CPA security of the ring variant of LP. Our goal is to quantify the reduction loss. Therefore, we derive the success probability and running time of any distinguisher $\mathcal{D}$ for R-LWE in terms of the success probability and running time of any IND-CPA attacker $\mathcal{A}$. These are not specified in [LP11].

The IND-CPA security of LP is proven in a sequence of hybrids. This is a standard technique used in proving indistinguishability between two distributions when the hardness assumption is applied multiple times. Starting from the initial distribution (in the proof of LP, it is the IND-CPA game) the proof proceeds by showing that each hybrid is computationally or statistically indistinguishable from the previous one. More precisely, for each two successive hybrids there exists a reduction proving the hardness of distinguishing those hybrids under the computational assumption or a statistical security argument. Consequently, this shows the hardness of distinguishing the original hybrid from the final one (in the proof of LP, it is the uniform distribution).

Table 1 shows the sequence of hybrids $\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2$ and reductions $\mathcal{D}_1, \mathcal{D}_2$ that correspond to the security proof of LP. The goal is to show that the entire view of the adversary $\mathcal{A}$ is computationally indistinguishable from uniformly random for any encrypted message $\mu \in R_q$, where $R_q = \mathbb{Z}_q[x]/\langle f(x) \rangle$ for some monic $n$-degree polynomial that is irreducible over $\mathbb{Z}$ (typically $f(x) = x^n + 1$). The first hybrid $\mathcal{H}_0$ is the real IND-CPA game. In hybrid $\mathcal{H}_1$, the public key $p$ is chosen uniformly random rather than being generated as an R-LWE sample. In hybrid $\mathcal{H}_2$, the ciphertext is chosen uniformly random rather than being computed by the encryption algorithm, which outputs two R-LWE samples (see Appendix B.1). The first reduction $\mathcal{D}_1$ shows that the hybrids $\mathcal{H}_0, \mathcal{H}_1$ are computationally indistinguishable assuming the hardness of decision R-LWE (see Definition 2.6). In other words, if $\mathcal{A}$ can distinguish hybrid $\mathcal{H}_0$ from $\mathcal{H}_1$, then reduction $\mathcal{D}_1$

| | **Hybrid** $\mathcal{H}_0$ | **Hybrid** $\mathcal{H}_1$ | **Hybrid** $\mathcal{H}_2$ | **Reduction** $\mathcal{D}_1$ | **Reduction** $\mathcal{D}_2$ |
|---|---|---|---|---|---|
| 1: | $a \leftarrow_\$ R_q$ | $a \leftarrow_\$ R_q$ | $a \leftarrow_\$ R_q$ | Input $(a, p)$ | Input $((a, c_1), (p, c_2))$ |
| 2: | $r_1, r_2 \leftarrow D_r^n$ | | | | |
| 3: | $p \leftarrow r_1 - ar_2$ | $\boxed{p \leftarrow_\$ R_q}$ | $p \leftarrow_\$ R_q$ | | |
| 4: | $b \leftarrow_\$ \{0, 1\}$ | $b \leftarrow_\$ \{0, 1\}$ | $b \leftarrow_\$ \{0, 1\}$ | $b \leftarrow_\$ \{0, 1\}$ | $b \leftarrow_\$ \{0, 1\}$ |
| 5: | **if** $b = 1$ **then** | **if** $b = 1$ **then** | **if** $b = 1$ **then** | **if** $b = 1$ **then** | **if** $b = 1$ **then** |
| 6: | $e_1, e_2, e_3 \leftarrow D_r^n$ | $e_1, e_2, e_3 \leftarrow D_r^n$ | | $e_1, e_2, e_3 \leftarrow D_r^n$ | |
| 7: | $c_1 \leftarrow ae_1 + e_2$ | $c_1 \leftarrow ae_1 + e_2$ | $\boxed{c_1, c_2 \leftarrow_\$ R_q}$ | $c_1 \leftarrow ae_1 + e_2$ | |
| 8: | $c_2 \leftarrow pe_1 + e_3 + \mu$ | $c_2 \leftarrow pe_1 + e_3 + \mu$ | $c_2 \leftarrow c_2 + \mu$ | $c_2 \leftarrow pe_1 + e_3 + \mu$ | $c_2 \leftarrow c_2 + \mu$ |
| 9: | **return** $(a, p, c_1, c_2)$ | **return** $(a, p, c_1, c_2)$ | **return** $(a, p, c_1, c_2)$ | **return** $(a, p, c_1, c_2)$ | **return** $(a, p, c_1, c_2)$ |
| 10: | **else** | **else** | **else** | **else** | **else** |
| 11: | $c_1', c_2' \leftarrow_\$ R_q$ | $c_1', c_2' \leftarrow_\$ R_q$ | $c_1', c_2' \leftarrow_\$ R_q$ | $c_1', c_2' \leftarrow_\$ R_q$ | $c_1' \leftarrow c_1,\ c_2' \leftarrow c_2$ |
| 12: | **return** $(a, p, c_1', c_2')$ | **return** $(a, p, c_1', c_2')$ | **return** $(a, p, c_1', c_2')$ | **return** $(a, p, c_1', c_2')$ | **return** $(a, p, c_1', c_2')$ |

Table 1: The sequence of hybrids $\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2$ and reductions $\mathcal{D}_1, \mathcal{D}_2$ that correspond to the security proof of LP [LP11]. Reduction $\mathcal{D}_1$ is related to the hybrids $\mathcal{H}_0, \mathcal{H}_1$, whereas reduction $\mathcal{D}_2$ is associated with the hybrids $\mathcal{H}_1, \mathcal{H}_2$. An instruction in a box indicates the deference between a hybrid and its preceding one.

can distinguish the R-LWE instance R-LWE$_{1,1,q,r}$ from uniformly random samples. Similarly, reduction $\mathcal{D}_2$ proves that the hybrids $\mathcal{H}_1, \mathcal{H}_2$ are computationally indistinguishable assuming the hardness of decision R-LWE for the instance R-LWE$_{1,2,q,r}$.

In the following two lemmas we deduce the success probability and running time of the security reduction of LP.

**Lemma 4.1.** *Let $\varepsilon_\mathcal{A}$ be the success probability of any IND-CPA attacker $\mathcal{A}$ against LP. Then the success probability $\varepsilon_\mathcal{D}$ of any distinguisher $\mathcal{D}$ for R-LWE corresponding to Theorem 4.1 is given by the relation $\varepsilon_\mathcal{A} \leq 2\varepsilon_\mathcal{D}$.*

*Proof.* First, we derive the advantages of the reductions $\mathcal{D}_1, \mathcal{D}_2$ (see Table 1). We let $\mathrm{Adv}_{\mathcal{H}_i,\mathcal{H}_{i+1}}(\mathcal{D}_{i+1})$ denote the advantage of the reduction $\mathcal{D}_{i+1}$ for the hybrids $\mathcal{H}_i, \mathcal{H}_{i+1}$, where $i = 0, 1$. Furthermore, we let $\mathrm{Adv}_{\mathcal{H}_i}(\mathcal{A})$ denote the advantage of the adversary $\mathcal{A}$ in the hybrid $\mathcal{H}_i$ for $i = 0, 1, 2$. From Table 1, we see that if the input of reduction $\mathcal{D}_1$ is a sample from R-LWE, then the output of $\mathcal{D}_1$ is exactly distributed as in hybrid $\mathcal{H}_0$, and when its input is a uniformly random sample from $R_q^2$, then its output is exactly distributed as in hybrid $\mathcal{H}_1$. Similarly, if the input of reduction $\mathcal{D}_2$ is two samples from R-LWE, then the output of $\mathcal{D}_2$ is exactly distributed as in hybrid $\mathcal{H}_1$, and for two uniformly random input samples from $R_q^2$ the output distribution coincides with the one from hybrid $\mathcal{H}_2$. Hence,

$$\mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) = \left|\mathrm{Prob}[\mathcal{D}_1^{\mathcal{H}_0}(a,p) = 1] - \mathrm{Prob}[\mathcal{D}_1^{\mathcal{H}_1}(a,p) = 1]\right| = \left|\mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_1}(\mathcal{A})\right|, \tag{5}$$

$$\begin{aligned}\mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2) &= \left|\mathrm{Prob}[\mathcal{D}_2^{\mathcal{H}_1}((a,c_1),(p,c_2)) = 1] - \mathrm{Prob}[\mathcal{D}_2^{\mathcal{H}_2}((a,c_1),(p,c_2)) = 1]\right| \\ &= \left|\mathrm{Adv}_{\mathcal{H}_1}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_2}(\mathcal{A})\right| .\end{aligned}$$

We note that $\mathrm{Adv}_{\mathcal{H}_2}(\mathcal{A}) = 0$, since the public key and ciphertext in hybrid $\mathcal{H}_2$ are uniformly random, i.e.,

$$\mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2) = \mathrm{Adv}_{\mathcal{H}_1}(\mathcal{A}) . \tag{6}$$

Putting equation (6) in (5) yields

$$\mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) = \left| \mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2) \right| \geq \mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2)\ .$$

Hence

$$\mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) \leq \mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) + \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2)\ .$$

Without loss of generality, we assume that

$$\mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) \leq \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2)\ ^{2}.$$

This yields

$$\mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) \leq 2\,\mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2)\ .$$

The success probabilities $\varepsilon_{\mathcal{A}}, \varepsilon_{\mathcal{D}}$ of the adversary $\mathcal{A}$ and distinguisher $\mathcal{D}$, respectively, are given by

$$\varepsilon_{\mathcal{A}} = \mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) + \frac{1}{2} \quad \text{and} \quad \varepsilon_{\mathcal{D}} = \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2) + \frac{1}{2}\ .$$

Therefore, we obtain $\varepsilon_{\mathcal{A}} \leq 2\varepsilon_{\mathcal{D}} - \frac{1}{2} \leq 2\varepsilon_{\mathcal{D}}$.

□

**Lemma 4.2.** *Let $t_{\mathcal{A}}$ be the running time of any IND-CPA attacker $\mathcal{A}$ against LP. Then the running time $t_{\mathcal{D}}$ of any distinguisher $\mathcal{D}$ for R-LWE corresponding to Theorem 4.1 is estimated by $t_{\mathcal{D}} \leq 2t_{\mathcal{A}} + O(n \log n)$.*

*Proof.* The time $t_{\mathcal{D}}$ is the total running time of the reductions $\mathcal{D}_1, \mathcal{D}_2$. We consider the case in which reduction $\mathcal{D}_2$ runs only if $\mathcal{A}$ could not distinguish hybrid $\mathcal{H}_0$ from $\mathcal{H}_1$. Therefore, the adversary $\mathcal{A}$ is executed at most twice as a subroutine in $\mathcal{D}_1, \mathcal{D}_2$. The rest of the operations in $\mathcal{D}_1, \mathcal{D}_2$ includes polynomial addition and multiplication, sampling from discrete Gaussian distribution, and sampling random polynomials from $R_q$. Essentially, multiplication of polynomials dominates the time required for the other operations. According to [DDLL13], polynomial multiplication in dimension $n$ requires $O(n \log n)$ operations using Fast Fourier Transform (FFT). Thus, we have $t_{\mathcal{D}} \leq 2t_{\mathcal{A}} + O(n \log n)$.

□

In the next lemma we quantify the reduction loss using Lemma 4.1 and 4.2.

**Lemma 4.3.** *The tight security reduction of LP according to Theorem 4.1 induces a reduction loss of at most 3 bits.*

[2] In fact, the inequality is indeed true for LP, since the R-LWE instance in the ciphertext includes twice as much samples as the R-LWE instance in the public key, and hence it is easier.

*Proof.* The term $O(n \log n)$ included in the running time of the reduction (see Lemma 4.2) is basically the time required to simulate the IND-CPA game for the adversary $\mathcal{A}$. Therefore, there exists a positive constant $\frac{a}{b} \in \mathbb{Q} \cap (0,1)$ such that

$$O(n \log n) \leq \tfrac{a}{b} t_{\mathcal{D}} \ .$$

The latter holds because $t_{\mathcal{D}}$ must be exponential assuming the hardness of R-LWE[3]. We set $\frac{a}{b} = \frac{1}{2}$, since $\frac{1}{2} t_{\mathcal{D}}$ is already an excessive upper bound on $O(n \log n)$[4]. Thus, the running time of the reduction given in Lemma 4.2 becomes

$$t_{\mathcal{D}} \leq 4 t_{\mathcal{A}} \ .$$

Putting the latter together with the results of Lemma 4.1 in equation (2), Section 3.3, we obtain the following:

$$\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \leq 8 \cdot \frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \ .$$

This means that the tightness gap (see Section 3.3) is upper bounded by 8, which leads to a reduction loss of at most 3 bits.

□

Hence, the gap function (see Section 3, expression (4)) for LP is defined as follows:

$$\begin{aligned} \mathsf{Gap}_{\mathsf{LP}} : \ \mathbb{N}_{>\log \gamma} &\longrightarrow \mathbb{N} \\ \ell &\longmapsto \ell - 3 \ . \end{aligned}$$

**Parameter Selection**

In Table 2, we propose four sequences of parameters for LP. The first two sequences provably and non-provably offer 128 bits of security (see Section 3.4 for an explanation of provably and non-provably secure parameters). In accordance with our analysis including the function $\mathsf{Gap}_{\mathsf{LP}}$, the non-provably secure sequence offers a security of 125 bits as opposed to 128 bits. The provably secure sequence indeed offers 128 bits according to the security proof (see Theorem 4.1). Similarly, the second two sequences of parameters provably and non-provably offer 256 bits of security. We use the LWE estimator [APS15] for estimating the corresponding bit hardness of R-LWE. The estimator function $\mathsf{Est}_{\mathsf{LWE}}$ for LWE is given by

$$\begin{aligned} \mathsf{Est}_{\mathsf{LWE}} : \ \mathbf{Para}_{\mathsf{LWE}} &\longrightarrow \mathbb{N} \\ (n, m, q, r) &\longmapsto \ell \ . \end{aligned}$$

[3] All currently known algorithms for (R-)LWE require exponential running time, which is extremely larger than $O(n \log n)$.

[4] We note that giving an upper bound on $O(n \log n)$ larger than $\frac{1}{2} t_{\mathcal{D}}$, e.g., $\frac{3}{4} t_{\mathcal{D}}$, induces a larger reduction gap, whereas reducing it implies a smaller reduction gap.

| **Security level (bits)** | | 128 | | 256 | |
|---|---|---|---|---|---|
| **Provably secure** | | Yes | No | Yes | No |
| **Parameters** | | | | | |
| Dimension $n$ | | 512 | 512 | 1024 | 1024 |
| Modulus $q$ | | 239977 | 239929 | 995641 | 995587 |
| Gaussian parameter $r$ | | 9.6 | 8.6 | 12.2 | 11.6 |
| Message length $l$ | | 512 | 512 | 1024 | 1024 |
| Error threshold $t$ | | 59994 | 59982 | 248910 | 248896 |
| **Sizes (bits)** | | | | | |
| Encryption key | $n\lceil \log q \rceil$ | 9216 | 9216 | 20480 | 20480 |
| Decryption key | $n(1 + \lceil \log r \rceil)$ | 2560 | 2560 | 5120 | 5120 |
| Message | $n$ | 512 | 512 | 1024 | 1024 |
| Ciphertext | $2n\lceil \log q \rceil$ | 18432 | 18432 | 40960 | 40960 |
| **Timings (milliseconds)** | | | | | |
| Encryption | | 0.076 | 0.074 | 0.174 | 0.171 |
| Decryption | | 0.016 | 0.016 | 0.034 | 0.034 |

Table 2: Parameter sequences, sizes, and timings for LP [LP11].

Furthermore, we provide for these parameters the bit size of the corresponding encryption and decryption keys, message, and ciphertext in addition to the running time for encryption and decryption. We consider a message length of $n$ bits and the encoding method suggested in [LP11], which encodes an $n$-bit string $\mathbf{m} = (m_1, \dots, m_n)$ into an $n$-degree polynomial $\mu$ with coefficients $\mu_i = m_i \cdot \lfloor \frac{q}{2} \rfloor$ for all $i \in [n]$. This encoding function gives error tolerance $t = \lfloor \frac{q}{4} \rfloor$.

Table 2 shows that selecting provably-secure parameters according to the tight reduction of LP leads to a slight increase of the parameters. However, one observes that the related sizes of keys/ciphertext and timings of encryption/decryption do not change. Hence, our analysis shows that in case of LP, one can choose either of the two approaches for parameter selection enjoying essentially the same security level.

## 4.2 The Encryption Scheme LARA

In this section we consider the variant of the encryption scheme LARA [EB17] that is IND-CPA-secure in the random oracle model (see Appendix B.2 for its description). The scheme is a ring variant with an improved trapdoor construction as compared to the initial work of El Bansarkhani et al. [EDB15]. The security of LARA is based on the hardness of the ring version of decision A-LWE, which is at least as hard as decision R-LWE [EDB15, Theorem 2]. Hence, the security of LARA is based on the hardness of R-LWE, which follows from the theorem given below.

**Theorem 4.2** ([EDB15, Theorem 2], [EB17, Theorem 7]). *The public-key encryption scheme LARA is IND-CPA-secure in the random oracle model, assuming the hardness of decision R-LWE for the instances* $\text{R-LWE}_{1,1,q,r_1}$ *and* $\text{R-LWE}_{1,3,q,r_2}$.

Next, we specify the set of admissible parameters $\mathbf{Para}_{\mathsf{LARA}}$ for LARA and the reduction function $\mathsf{Red}_{\mathsf{LARA}}$. The set $\mathbf{Para}_{\mathsf{LWE}}$ has already been specified in Section 4.1. A sequence of parameters from $\mathbf{Para}_{\mathsf{LARA}}$ is given by $(\ell, n, q, p, r_1, r_2)$, where $\ell$ is the security parameter, $n$ the dimension of R-LWE secrets, $q$ a modulus, $p$ an integer used for encoding messages, and $r_1, r_2$ are two Gaussian parameters for key generation and encryption, respectively. Putting all the requirements on the parameters for correctness, efficiency, and security together, which are given in [EB17], we obtain the following set:

$$\mathbf{Para}_{\mathsf{LARA}} = \Big\{ (\ell, n, q, p, r_1, r_2) : \ \ell \in \mathbb{N}, \ n = 2^{n'}, \ n' \in \mathbb{N}, \ q = 2^k, \ k \in \mathbb{N}, \ p = 2^t, \ t \in \mathbb{N}, \\ r_1 = \alpha q, \ r_2 = p \cdot \sqrt{\ln(2(1+\tfrac{1}{\epsilon}))/\pi}, \ \alpha \in (0,1), \ \epsilon = \mathrm{negl}(n), \ r_2 \geq r_1 \Big\} .$$

We deduce the following reduction function from Theorem 4.2:

$$\begin{aligned} \mathsf{Red}_{\mathsf{LARA}} : \ \mathbf{Para}_{\mathsf{LARA}} &\longrightarrow \mathbf{Para}_{\mathsf{LWE}} \times \mathbf{Para}_{\mathsf{LWE}} \\ (\ell, n, q, p, r_1, r_2) &\longmapsto \big( (n, n, q, r_1), (n, 3n, q, r_2) \big) . \end{aligned}$$

## Analyzing the Security Reduction

We proceed with analyzing the security proof of LARA in order to quantify the reduction loss. Therefore, we derive the running time and success probability of the reduction, since they are not given in [EB17]. Similar to LP, the IND-CPA security of LARA is proven by using the hybrid argument (see Section 4.1). We give in Table 3 the sequence of hybrids $\mathcal{H}_0, \ldots, \mathcal{H}_5$ and reductions $\mathcal{D}_1, \ldots, \mathcal{D}_5$ that correspond to the security proof of LARA. The first hybrid $\mathcal{H}_0$ is the real IND-CPA game, where the entire view of the adversary $\mathcal{A}$ has to be computationally indistinguishable from uniformly random for any encrypted message $\mathbf{m} \in \{0,1\}^{3n \log p}$. For a modulus $q = 2^k$, the public key is given by $[a_1, a_2, 2^{k-1} - (a_1 z_1 + a_2 z_2)]$, where $a_1, a_2 \in R_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$ and $z_1, z_2 \in D^n_{r_1}$ (see Appendix B.2). This public key can be written as $a_2 \cdot [a, 1, 2^{k-1} a_2^{-1} - z]$, where $a = a_1 a_2^{-1}$, $z = a z_1 + z_2$, and $a_2$ is invertible in $R_q$, i.e., $a_2 \in R_q^{\times}$. Therefore, the public key involves the R-LWE sample $(a, z) \in R_q \times R_q$. In hybrid $\mathcal{H}_1$, the third polynomial of the public key is chosen uniformly random rather than being generated as described. In hybrid $\mathcal{H}_2$ and $\mathcal{H}_3$, the output of the functions $H_1, H_2$ is replaced with a uniformly random value. In hybrid $\mathcal{H}_4$, the error terms of the three samples in the ciphertext are sampled according to the distribution $D^n_{r_2}$ as opposed to the distribution $D_{p\mathbb{Z}^n + \mathbf{v}, r_2}$ for a uniformly random vector $\mathbf{v} \in \mathbb{Z}_p^n$. In hybrid $\mathcal{H}_5$, the ciphertext is chosen uniformly random rather being computed according to the encryption algorithm. A reduction $\mathcal{D}_{i+1}$ shows the difference between the hybrids $\mathcal{H}_i, \mathcal{H}_{i+1}$ for $i = 0, \ldots, 4$.

In the following three lemmas we derive the success probability and running time of the security proof of LARA as well as its reduction loss.

**Lemma 4.4.** *Let $\varepsilon_{\mathcal{A}}$ be the success probability of any IND-CPA attacker $\mathcal{A}$ against LARA. Then the success probability $\varepsilon_{\mathcal{D}}$ of any distinguisher $\mathcal{D}$ for R-LWE is associated with $\varepsilon_{\mathcal{A}}$ in accordance to Theorem 4.2 by the relation $\varepsilon_{\mathcal{A}} \leq 3\varepsilon_{\mathcal{D}}$.*

*Proof.* We first derive the advantages of the reductions $\mathcal{D}_1, \ldots, \mathcal{D}_5$ (see Table 3). For the first reduction we obtain

$$\mathrm{Adv}_{\mathcal{H}_0, \mathcal{H}_1}(\mathcal{D}_1) = \Big| \mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_1}(\mathcal{A}) \Big| . \qquad (7)$$

|  | **Hybrid** $\mathcal{H}_0$ | **Hybrid** $\mathcal{H}_1$ | **Hybrid** $\mathcal{H}_2$ | **Hybrid** $\mathcal{H}_3$ |
|---|---|---|---|---|
| 1: | $a_1 \leftarrow_{\$} R_q$ | $a_1 \leftarrow_{\$} R_q$ | $a_1 \leftarrow_{\$} R_q$ | $a_1 \leftarrow_{\$} R_q$ |
| 2: | $a_2 \leftarrow_{\$} R_q^{\times}$ | $a_2 \leftarrow_{\$} R_q^{\times}$ | $a_2 \leftarrow_{\$} R_q^{\times}$ | $a_2 \leftarrow_{\$} R_q^{\times}$ |
| 3: | $z_1, z_2 \leftarrow D_{r_1}^n$ |  |  |  |
| 4: | $a_3 \leftarrow 2^{k-1} - (a_1 z_1 + a_2 z_2)$ | $\boxed{a_3 \leftarrow_{\$} R_q}$ | $a_3 \leftarrow_{\$} R_q$ | $a_3 \leftarrow_{\$} R_q$ |
| 5: | $\mathbf{a} \leftarrow [a_1, a_2, a_3]$ | $\mathbf{a} \leftarrow [a_1, a_2, a_3]$ | $\mathbf{a} \leftarrow [a_1, a_2, a_3]$ | $\mathbf{a} \leftarrow [a_1, a_2, a_3]$ |
| 6: | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ |
| 7: | **if** $b = 1$ **then** | **if** $b = 1$ **then** | **if** $b = 1$ **then** | **if** $b = 1$ **then** |
| 8: | $\mathbf{x} = (x_1, \ldots, x_n) \leftarrow_{\$} \{0,1\}^n$ | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ |
| 9: | $\mathbf{y} = (y_1, \ldots, y_n) \leftarrow H_1(\mathbf{x})$ | $\mathbf{y} \leftarrow H_1(\mathbf{x})$ | $\boxed{\mathbf{y} \leftarrow_{\$} \mathbb{Z}_{2^{k-1}}^n}$ | $\mathbf{y} \leftarrow_{\$} \mathbb{Z}_{2^{k-1}}^n$ |
| 10: | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ |
| 11: | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ |
| 12: | $\mathbf{r} \leftarrow H_2(s)$ | $\mathbf{r} \leftarrow H_2(s)$ | $\mathbf{r} \leftarrow H_2(s)$ | $\boxed{\mathbf{r} \leftarrow_{\$} \{0,1\}^{3n \log p}}$ |
| 13: | $\mathbf{v} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3] \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ | $\mathbf{v} \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ | $\mathbf{v} \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ | $\mathbf{v} \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ |
| 14: | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ |
| 15: | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ |
| 16: | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ |
| 17: | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ |
| 18: | **else** | **else** | **else** | **else** |
| 19: | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ |
| 20: | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ |

|  | **Hybrid** $\mathcal{H}_4$ | **Hybrid** $\mathcal{H}_5$ | **Reduction** $\mathcal{D}_1$ | **Reduction** $\mathcal{D}_2$ |
|---|---|---|---|---|
| 1: | $a_1 \leftarrow_{\$} R_q$ | $a_1 \leftarrow_{\$} R_q$ | Input $(\mathbf{a}) \in R_q^3$ | Input $(\mathbf{a}, \mathbf{y}) \in R_q^3 \times \mathbb{Z}_{2^{k-1}}^n$ |
| 2: | $a_2 \leftarrow_{\$} R_q^{\times}$ | $a_2 \leftarrow_{\$} R_q^{\times}$ | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ |
| 3: | $a_3 \leftarrow_{\$} R_q$ | $a_3 \leftarrow_{\$} R_q$ | **if** $b = 1$ **then** | **if** $b = 1$ **then** |
| 4: | $\mathbf{a} \leftarrow [a_1, a_2, a_3]$ | $\mathbf{a} \leftarrow [a_1, a_2, a_3]$ | $\mathbf{x} = (x_1, \ldots, x_n) \leftarrow_{\$} \{0,1\}^n$ | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ |
| 5: | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ | $\mathbf{y} = (y_1, \ldots, y_n) \leftarrow H_1(\mathbf{x})$ |  |
| 6: | **if** $b = 1$ **then** | **if** $b = 1$ **then** | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ |
| 7: | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ |  | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ |
| 8: | $\mathbf{y} \leftarrow_{\$} \mathbb{Z}_{2^{k-1}}^n$ |  | $\mathbf{r} \leftarrow H_2(s)$ | $\mathbf{r} \leftarrow H_2(s)$ |
| 9: | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ |  | $\mathbf{v} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3] \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ | $\mathbf{v} \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ |
| 10: | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ |  | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ |
| 11: | $\boxed{\mathbf{e} \leftarrow D_{r_2}^{3n}}$ |  | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ |
| 12: | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ | $\boxed{\mathbf{c} \leftarrow_{\$} R_q^3}$ | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ |
| 13: | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ |
| 14: | **else** | **else** | **else** | **else** |
| 15: | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ |
| 16: | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ |

|  | **Reduction** $\mathcal{D}_3$ | **Reduction** $\mathcal{D}_4$ | **Reduction** $\mathcal{D}_5$ |
|---|---|---|---|
| 1: | Input $(\mathbf{a}, \mathbf{r}) \in R_q^3 \times \{0,1\}^{3n \log p}$ | Input $(\mathbf{a}, \mathbf{e}) \in R_q^3 \times R_q^3$ | Input $(\mathbf{a}, \mathbf{c}) \in R_q^3 \times R_q^3$ |
| 2: | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ | $b \leftarrow_{\$} \{0,1\}$ |
| 3: | **if** $b = 1$ **then** | **if** $b = 1$ **then** | **if** $b = 1$ **then** |
| 4: | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ | $\mathbf{x} \leftarrow_{\$} \{0,1\}^n$ |  |
| 5: | $\mathbf{y} \leftarrow_{\$} \mathbb{Z}_{2^{k-1}}^n$ | $\mathbf{y} \leftarrow_{\$} \mathbb{Z}_{2^{k-1}}^n$ |  |
| 6: | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ | $s_i \leftarrow y_i \Vert x_i \in \mathbb{Z}_q,\ i \in [n]$ |  |
| 7: | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ | $s \leftarrow (s_1, \ldots, s_n) \in R_q$ |  |
| 8: | $\mathbf{v} \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$ |  |  |
| 9: | $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2},\ j \in [3]$ |  |  |
| 10: | $\mathbf{e} \leftarrow [e_1, e_2, e_3]$ |  |  |
| 11: | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ | $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$ |  |
| 12: | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ | **return** $(\mathbf{a}, \mathbf{c})$ |
| 13: | **else** | **else** | **else** |
| 14: | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow_{\$} R_q^3$ | $\mathbf{c}' \leftarrow \mathbf{c}$ |
| 15: | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ | **return** $(\mathbf{a}, \mathbf{c}')$ |

Table 3: The sequence of hybrids $\mathcal{H}_0, \ldots, \mathcal{H}_5$ and reductions $\mathcal{D}_1, \ldots, \mathcal{D}_5$ that correspond to the security proof of LARA [EB17], where $H_1 : \{0,1\}^n \longrightarrow \mathbb{Z}_{2^{k-1}}^n$ and $H_2 : R_q \longrightarrow \{0,1\}^{3n \log p}$ are cryptographic hash functions modeled as random oracles and $\mathsf{encode} : \{0,1\}^{3n \log p} \longrightarrow \mathbb{Z}_p^n \times \mathbb{Z}_p^n \times \mathbb{Z}_p^n$ is a bijective encoding function. An instruction in a box indicates the deference between a hybrid and its preceding one.

The hybrids $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ are equal, since the random oracles $H_1, H_2$ are queried only once on different inputs. This means that

$$\mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2) = \mathrm{Adv}_{\mathcal{H}_2,\mathcal{H}_3}(\mathcal{D}_3) = 0 \ .$$

The advantage of reduction $\mathcal{D}_4$ is given by distinguishing between the distribution $D^n_{r_2}$ and $D_{p\mathbb{Z}^n+\mathbf{v},r_2}$ for a uniformly random vector $\mathbf{v} \in \mathbb{Z}_p^n$. According to [EDB15, Lemma 5], the statistical distance between both distributions is at most $\frac{2\epsilon}{1-\epsilon}$, where $\epsilon = \mathrm{negl}(n)$, i.e.,

$$\mathrm{Adv}_{\mathcal{H}_3,\mathcal{H}_4}(\mathcal{D}_4) = \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_4}(\mathcal{D}_4) = \Big|\mathrm{Adv}_{\mathcal{H}_1}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_4}(\mathcal{A})\Big| \leq \frac{2\epsilon}{1-\epsilon} \ . \tag{8}$$

In hybrid $\mathcal{H}_5$, the public key and ciphertext are uniformly random. Thus, we have

$$\mathrm{Adv}_{\mathcal{H}_4,\mathcal{H}_5}(\mathcal{D}_5) = \Big|\mathrm{Adv}_{\mathcal{H}_4}(\mathcal{A}) - \mathrm{Adv}_{\mathcal{H}_5}(\mathcal{A})\Big| = \mathrm{Adv}_{\mathcal{H}_4}(\mathcal{A}) \ . \tag{9}$$

Putting the equations (7), (8), and (9) together yields

$$\mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) \leq \mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) + \mathrm{Adv}_{\mathcal{H}_4,\mathcal{H}_5}(\mathcal{D}_5) + \frac{2\epsilon}{1-\epsilon} \ .$$

Since $r_2 \geq r_1$, the R-LWE instance in the ciphertext is harder than the one included in the public key, i.e.,

$$\mathrm{Adv}_{\mathcal{H}_4,\mathcal{H}_5}(\mathcal{D}_5) \leq \mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) \ .$$

This implies that

$$\mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) \leq 2\,\mathrm{Adv}_{\mathcal{H}_0,\mathcal{H}_1}(\mathcal{D}_1) + \frac{2\epsilon}{1-\epsilon} \ .$$

The success probabilities $\varepsilon_{\mathcal{A}}, \varepsilon_{\mathcal{D}}$ of the adversary $\mathcal{A}$ and distinguisher $\mathcal{D}$ are given by

$$\varepsilon_{\mathcal{A}} = \mathrm{Adv}_{\mathcal{H}_0}(\mathcal{A}) + \frac{1}{2} \quad \text{and} \quad \varepsilon_{\mathcal{D}} = \mathrm{Adv}_{\mathcal{H}_1,\mathcal{H}_2}(\mathcal{D}_2) + \frac{1}{2} \ .$$

Therefore, we obtain

$$\varepsilon_{\mathcal{A}} \leq 2\varepsilon_{\mathcal{D}} + \frac{2\epsilon}{1-\epsilon},$$

where $\epsilon = \mathrm{negl}(n)$ is taken as small as possible such that $\frac{2\epsilon}{1-\epsilon} \leq \varepsilon_{\mathcal{D}}$. This implies that $\varepsilon_{\mathcal{A}} \leq 3\varepsilon_{\mathcal{D}}$. □

**Lemma 4.5.** *Let $t_{\mathcal{A}}$ be the running time of any IND-CPA attacker $\mathcal{A}$ against LARA. Then the running time $t_{\mathcal{D}}$ of any distinguisher $\mathcal{D}$ for R-LWE is associated with $t_{\mathcal{A}}$ in accordance to Theorem 4.2 by the relation $t_{\mathcal{D}} \leq 5t_{\mathcal{A}} + O(n \log n)$.*

*Proof.* The time $t_{\mathcal{D}}$ is the total running time of the reductions $\mathcal{D}_1, \ldots, \mathcal{D}_5$ given in Table 3. Similar to LP, the operations required for these reductions essentially include executing the IND-CPA adversary $\mathcal{A}$ at most five times (see Section 4.1) and performing polynomial multiplication with time $O(n \log n)$ via FFT. This leads to a total running time of $t_{\mathcal{D}} \leq 5t_{\mathcal{A}} + O(n \log n)$. □

**Lemma 4.6.** *The tight security reduction of LARA corresponding to Theorem 4.2 induces a reduction loss of at most 5 bits.*

*Proof.* Similar to LP, we obtain an upper bound on the time $O(n \log n)$, which is given by

$$O(n \log n) \leq \tfrac{1}{2} \cdot t_{\mathcal{D}} \qquad \text{(see Section 4.1)}.$$

Hence, the running time becomes $t_{\mathcal{D}} \leq 10 \cdot t_{\mathcal{A}}$. Thus, we have

$$\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \leq 30 \cdot \frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \leq 32 \cdot \frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \ .$$

This means that the reduction gap is at most 5 bits.

□

Using Lemma 4.6, the gap function for LARA is defined as follows:

$$\begin{aligned} \mathsf{Gap}_{\mathsf{LARA}} :\ \mathbb{N}_{>\log\gamma} &\longrightarrow \mathbb{N} \\ \ell &\longmapsto \ell - 5 \ . \end{aligned}$$

**Parameter Selection**

In Table 4, we propose four sequences of parameters for LARA. The second column includes two sequences of parameters offering 128 bits of security that either involve the reduction loss or not. Similarly, the third column includes two sequences of parameters for 256 bits of security. More precisely, the parameter sequences not involving the gap function $\mathsf{Gap}_{\mathsf{LARA}}$ (non-provably secure) offer 123 and 251 bits of security as opposed to 128 and 256 bits. The provably secure sequences indeed offer 128 and 256 bits according to the security proof stated in Theorem 4.2. As for LP, we use the LWE estimator [APS15] in order to measure the bit hardness of the respective R-LWE instances.

Furthermore, we provide for these parameters the associated bit size of the encryption and decryption keys, message, and ciphertext in addition to the running time of the encryption and decryption procedures. For computing the Gaussian parameter $r_2$, we set $\epsilon$ to $2^{-128}$ and $2^{-256}$ for security level 128 and 256.

Table 4 shows that selecting parameters according to the tight reduction of LARA ensures identical sizes of keys and ciphertexts as well as equal timings of encryption and decryption. Therefore, our analysis demonstrates that either of the two approaches for parameter selection can be chosen in case of LARA, leading to essentially the same security level.

| **Security level (bits)** | | 128 | | 256 | |
|---|---|---|---|---|---|
| **Provably secure** | | Yes | No | Yes | No |
| **Parameters** | | | | | |
| Dimension $n$ | | 512 | 512 | 1024 | 1024 |
| Modulus $q = 2^k$ | | $2^{20}$ | $2^{20}$ | $2^{23}$ | $2^{23}$ |
| Message range $p = 2^t$ | | $2^6$ | $2^6$ | $2^7$ | $2^7$ |
| Gaussian parameter for public key $r_1$ | | 27 | 22 | 51 | 46 |
| Gaussian parameter for encryption $r_2$ | | 341.4 | 341.4 | 963.8 | 963.8 |
| **Sizes (bits)** | | | | | |
| Encryption key | $n \log q$ | 10240 | 10240 | 23552 | 23552 |
| Decryption key | $n(1 + \lceil \log r_1 \rceil)$ | 3072 | 3072 | 7168 | 7168 |
| Message | $3n \log p$ | 9216 | 9216 | 21504 | 21504 |
| Ciphertext | $3n \log q$ | 30720 | 30720 | 70656 | 70656 |
| **Timings (milliseconds)** | | | | | |
| Encryption | | 0.090 | 0.090 | 0.186 | 0.185 |
| Decryption | | 0.052 | 0.051 | 0.109 | 0.110 |

Table 4: Parameter sequences, sizes, and timings for LARA [EB17].

### 4.3 The Signature Scheme BLISS-B

In this section we consider the efficient variant of the signature scheme BLISS [DDLL13, Duc14] (see Appendix B.3 for its description). The scheme is strongly EUF-CMA-secure in the random oracle model (see Appendix A.2) and its security is based on the hardness of a variant of R-SIS defined by Ducas et al. [DDLL13] as an NTRU version of R-SIS. This version is inspired by the key generation algorithm of the public-key encryption scheme NTRUEncrypt [HPS98]. Unlike the standard SIS defined in Section 2.1, the matrix $\mathbf{A}$ is chosen following the distribution that picks two uniformly random polynomials $f, g \in R_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$ with exactly $d_1 = \lceil \delta_1 n \rceil$ coefficients in $\{\pm 1\}$ and $d_2 = \lceil \delta_2 n \rceil$ coefficients in $\{\pm 2\}$ for given densities $\delta_1, \delta_2 \in [0, 1)$, and all other coefficients are 0. If $f$ is not invertible modulo $q$, it is resampled. Finally, the distribution outputs $\mathbf{A} = (2 \cdot \frac{2g+1}{f}, q - 2) \in R_{2q}^{1 \times 2}$. The secret key $\mathbf{s} = (f, 2g + 1)$ is a valid solution, since

$$\mathbf{A}\mathbf{s} = (2 \cdot \frac{2g+1}{f}, q - 2) \cdot (f, 2g + 1) = \mathbf{0} \pmod{q} .$$

Ducas et al. [DDLL13] state that for certain parameters, the NTRU version of R-SIS problem is at least as hard as breaking NTRUEncrypt. We note that NTRUEncrypt has no proof of security so far[5]. With careful parameter selection, however, it is still considered a secure encryption scheme.

In [Duc14], Ducas proposes a variant of BLISS, called BLISS-B, which we analyze in this work. The variant BLISS-B improves the efficiency of BLISS by an optimized key generation and signing procedure. We denote the NTRU version of R-SIS by R-SIS$_{\mathsf{NTRU}}$, since its instances have parameters that are different from those in the standard R-SIS. The security proof of BLISS-B is given by the following theorem (parameters that are used in the theorem are defined next):

**Theorem 4.3** ([DDLL13, Theorem 4.4]). *Suppose there is a polynomial-time algorithm $\mathcal{A}$, which makes at most $q_s$ queries to the signing oracle and $q_h$ queries to the random oracle $H$, and succeeds in forging*

[5] A provably secure variant of NTRUEncrypt based on R-LWE was later proposed by Stehlé and Steinfeld [SS11].

*valid signatures of the signature scheme BLISS-B with non-negligible probability $\varepsilon_{\mathcal{A}}$. Then there exists a polynomial-time algorithm $\mathcal{D}$, which can solve the* R-SIS$_{\mathsf{NTRU}}$ *problem for modulus $q$ and norms bounded by $\beta = 2B_2 + (2^d+1)\sqrt{n}$ with probability $\varepsilon_{\mathcal{D}} \geq \frac{\varepsilon_{\mathcal{A}}^2}{2(q_h+q_s)}$.*

We specify the set of admissible parameters $\mathbf{Para}_{\mathsf{BLISS\text{-}B}}$ for BLISS-B. A sequence of parameters from $\mathbf{Para}_{\mathsf{BLISS\text{-}B}}$ is given by $(\ell, n, q, \delta_1, \delta_2, \sigma, \alpha, \kappa, d, B_2, B_\infty) \in \mathbf{Para}_{\mathsf{BLISS\text{-}B}}$, where $\ell$ is the security parameter, $n$ the dimension of the secret key, $q$ a modulus, $\delta_1$ and $\delta_2$ are two densities for $\{\pm 1\}$ and $\{\pm 2\}$ coefficients in the secret key, $\sigma$ is the standard deviation of Gaussian distribution, $\alpha$ a real number that controls the norm of signatures, $\kappa$ a Hamming weight for the random oracle, $d$ the number of dropped bits when compressing signatures, and $B_2, B_\infty$ are two bounds for verifying the $\ell_2$-norm and $\ell_\infty$-norm of signatures. Collecting all relations on the parameters from [DDLL13, Duc14] that ensure correctness, efficiency, and security, we obtain the following set:

$$\begin{aligned}\mathbf{Para}_{\mathsf{BLISS\text{-}B}} = \Bigg\{ &(\ell, n, q, \delta_1, \delta_2, \sigma, \alpha, \kappa, d, B_2, B_\infty) :\ \ell \in \mathbb{N},\ n = 2^k,\ k \in \mathbb{N},\ q \in \mathbb{P},\ q = 1 \pmod{2n},\\ &q = 1 \pmod{2^{d-1}},\ \delta_1, \delta_2 \in [0,1),\ \sigma \in \mathbb{Z},\ \alpha > 0,\ \alpha = \frac{\sigma}{\sqrt{P_{\mathsf{max}}}},\\ &P_{\mathsf{max}} = \begin{cases}(5\lceil\delta_1 n\rceil + 5)\cdot\kappa & \text{if } \delta_2 = 0,\\ (5\lceil\delta_1 n\rceil + 20\lceil\delta_2 n\rceil + 9)\cdot\kappa & \text{otherwise,}\end{cases}\\ &\kappa \in \mathbb{N},\ \kappa < n,\ \binom{n}{\kappa} \geq 2^\ell,\ d \in \mathbb{N},\ d \geq 3,\ B_2 = \eta\sqrt{2n}\sigma,\ \eta^{2n}\cdot\exp(n(1-\eta^2)) \leq 2^{-\ell},\\ &2B_\infty + (2^d+1) < \frac{q}{2},\ \frac{\sqrt{nq/2\pi e}}{\sqrt{2n\sigma_k^2/2}} < 1,\ \sigma_k = \frac{\|(f,g)\|}{\sqrt{2n}} \Bigg\}.\end{aligned}$$

Next, we specify the set of all sequences of parameters for R-SIS$_{\mathsf{NTRU}}$, denoted by $\mathbf{Para}_{\mathsf{R\text{-}SIS}_{\mathsf{NTRU}}}$. Any instance of R-SIS$_{\mathsf{NTRU}}$ is parametrized by the sequence $(n, q, \delta_1, \delta_2, \beta) \in \mathbf{Para}_{\mathsf{R\text{-}SIS}_{\mathsf{NTRU}}}$, where

$$\mathbf{Para}_{\mathsf{R\text{-}SIS}_{\mathsf{NTRU}}} = \Big\{(n, q, \delta_1, \delta_2, \beta) :\ n, q \in \mathbb{N},\ \delta_1, \delta_2, \beta > 0\Big\}.$$

We proceed with deriving the reduction function for BLISS-B. Theorem 4.3 implies the following reduction function:

$$\begin{aligned}\mathsf{Red}_{\mathsf{BLISS\text{-}B}} :\ &\mathbf{Para}_{\mathsf{BLISS\text{-}B}} \longrightarrow \mathbf{Para}_{\mathsf{R\text{-}SIS}_{\mathsf{NTRU}}}\\ (\ell, n, q, \delta_1, \delta_2, \sigma, \alpha, \kappa, d, B_2, B_\infty) &\longmapsto (n, q, \delta_1, \delta_2, 2B_2 + (2^d+1)\sqrt{n})\ .\end{aligned}$$

## Analyzing the Security Reduction

Next, we quantify the reduction loss for BLISS-B, which equivalently holds for BLISS. To this end, the running time $t_{\mathcal{D}}$ and success probability $\varepsilon_{\mathcal{D}}$ of the SIS solver $\mathcal{D}$ have to be derived in terms of the running time $t_{\mathcal{A}}$ and success probability $\varepsilon_{\mathcal{A}}$ of the forger $\mathcal{A}$. The success probability is already provided in Theorem 4.3 by the relation $\varepsilon_{\mathcal{D}} \geq \frac{\varepsilon_{\mathcal{A}}^2}{2(q_h+q_s)}$. The running time is estimated by $t_{\mathcal{D}} \approx 2t_{\mathcal{A}}$, since the security proof of BLISS uses the General Forking Lemma [BN06] in order to create a solution to SIS. This requires running the forger $\mathcal{A}$ twice (we refer to [DDLL13, Lemma 3.5] for more details). This leads to the following lemma:

**Lemma 4.7.** *The non-tight security reduction of BLISS corresponding to Theorem 4.3 induces a reduction loss of at least 63 bits.*

*Proof.* Putting the relations $\varepsilon_{\mathcal{D}} \geq \frac{\varepsilon_{\mathcal{A}}^2}{2(q_h+q_s)}$ and $t_{\mathcal{D}} \approx 2t_{\mathcal{A}}$ in equation (2), Section 3.3, we obtain the following:

$$\frac{t_{\mathcal{D}}}{\varepsilon_{\mathcal{D}}} \leq \frac{4(q_h+q_s)}{\varepsilon_{\mathcal{A}}} \cdot \frac{t_{\mathcal{A}}}{\varepsilon_{\mathcal{A}}} \ .$$

Consequently, the reduction loss depends on the number of hash and sign queries $q_h, q_s$ that the forger $\mathcal{A}$ is allowed to make, and on its success probability $\varepsilon_{\mathcal{A}}$. By setting $\varepsilon_{\mathcal{A}} = 1$, we obtain the smallest possible reduction gap. It remains to give reasonable (lower) bounds on $q_h, q_s$. Koblitz and Menezes [KM06] argue that $q_s$ is limited, since signature queries require a response from the reduction $\mathcal{D}$, whereas $q_h$ is limited to only the total running time of $\mathcal{A}$, since the random oracle corresponds to evaluating a publicly available function. They suggest $q_s$ to be between $2^{20}$ to $2^{30}$, and $q_h$ to be $2^{80}$ or at the very least $2^{50}$. Goh et al. [GJKW07] and Coron [Cor00, Cor02] propose $q_s \approx 2^{30}$ and $q_h \approx 2^{60}$ coinciding with the suggestion of Bellare and Rogaway [BR96]. Recently, Chatterjee et al. [CKMS16] give for $q_h$ the values $2^{64}$ and $2^{80}$ in the context of identity-based encryption. Therefore, we consider the values $2^{60}$ and $2^{30}$ to be reasonable and somewhat lower bounds for $q_h$ and $q_s$. This means that the reduction loss is at least $\lceil \log(4(q_h+q_s)) \rceil = \lceil \log(4(2^{60}+2^{30})) \rceil = 63$ bits. We stress that the latter term is dominated by the number of random oracle queries[6]. $\square$

Based on Lemma 4.7, the gap function for BLISS is defined as follows:

$$\begin{aligned} \mathsf{Gap}_{\mathsf{BLISS}} : \ \mathbb{N}_{>\log\gamma} &\longrightarrow \mathbb{N} \\ \ell &\longmapsto \ell - 63 \ . \end{aligned}$$

## Parameter Selection

Unlike other schemes based on (R-)SIS, BLISS-B is based on a combination of the R-SIS and the NTRU problem that stems from the key generation algorithm of NTRUEncrypt[HPS98]. Therefore, selecting parameters for BLISS-B requires to consider both problems. Forging a signature implies solving R-SIS following the security proof. In practice, this is accomplished by applying lattice reduction as described in Section 2.1. Besides forging signatures, there exist key recovery attacks, which are related to breaking NTRUEncrypt (see [DDLL13, Appendix A]). These attacks include applying lattice reduction to the so called *NTRU lattice* and its dual lattice in addition to the hybrid lattice reduction and meet-in-the-middle attack proposed by Howgrave-Graham [How07], also called the *hybrid attack*. Moreover, there exists the subfield attack against NTRU lattices. It was proposed by Gentry and Szydlo [GS02] and recently (after publication of BLISS-B) revisited by Albrecht et al. [ABD16] and then by Kirchner and Fouque [KF16]. Roughly speaking, the subfield attack exploits the existence of subfields in order to recover the secret key. However, the subfield attack should become inapplicable when the modulus $q$ is small enough [ABD16]

[6] Setting $q_s = 0$ changes the reduction gap by just one bit.

and the running time of the attack becomes polynomial when $q$ is significantly large, i.e., $q = 2^{\Omega(\sqrt{n \log \log n})}$ for power of 2 cyclotomic fields [KF16]. Albrecht et al. [ABD16] provide a vulnerability factor for this attack, which is given by

$$F = \frac{\sqrt{nq/2\pi e}}{\sqrt{2}n\sigma_k^2/2}, \tag{10}$$

where $\sigma_k^2$ represents the variance of the distribution of secret keys. For given parameters, perfect immunity to the subfield attack is achieved for $F < 1$ [ABD16].

For the sequences of parameters we propose in Table 7, we analyze the following three attacks:

1. *Lattice Reduction.* We consider lattice reduction to measure the hardness of forging signatures (see Table 5), i.e., the hardness of finding a vector of norm $\beta = 2B_2 + (2^d + 1)\sqrt{n}$.

| **Security level (bits)** | 128 | | 256 | |
|---|---|---|---|---|
| **Provably secure** | Yes | No | Yes | No |
| Lattice dimension $2n$ | 2048 | 1024 | 2048 | 2048 |
| Hermite delta $\delta = (\beta/\sqrt{q})^{\frac{1}{2n}}$ | 1.00324 | 1.00577 | 1.00307 | 1.00307 |
| Required block size | 536 | 230 | 578 | 578 |
| BKZ 2.0 cost $\log(T_{\mathsf{BKZ}})$ | 390.44 | 129.26 | 429.72 | 429.72 |

Table 5: Parameters of lattice reduction for solving the underlying $\mathsf{R\text{-}SIS_{NTRU}}$ instance.

2. *Hybrid Attack.* Since the hybrid attack is practically the best known attack on NTRU lattices [ABD16, KF16, BCLvV16, HPS+17], we consider this attack and the improvement of its analysis presented by Wunderer [Wun16] in order to estimate the hardness of recovering the secret key. More precisely, we use Wunderer's Sage code in order to estimate the hardness of finding a vector $(f, g)$, which is sufficient to recover the secret key $(f, 2g + 1)$ (see Table 6).

| **Security level (bits)** | 128 | | 256 | |
|---|---|---|---|---|
| **Provably secure** | Yes | No | Yes | No |
| NTRU lattice dimension $2n$ | 2048 | 1024 | 2048 | 2048 |
| Optimal meet-in-the-middle search dimension | 438 | 95 | 349 | 349 |
| Optimal Hermite delta | 1.00324 | 1.00490 | 1.00311 | 1.00311 |
| Optimal block size | 536 | 295 | 568 | 568 |
| Total running time $\log(T)$ | 390.95 | 179.03 | 419.72 | 419.72 |

Table 6: Parameters of the hybrid attack for recovering the secret key.

3. *Subfield Attack.* In addition to the hybrid attack and in order to be on the safe side regarding key recovery, we select parameters that provide perfect immunity to the subfield attack. More precisely, we select the number $d_1, d_2$ of entries from $\{\pm 1\}, \{\pm 2\}$ large enough such that the vulnerability factor $F$ given in equation (10) is smaller than 1. That is, for all sequences of parameters given in Table 7, we have $F = 0.99$.

In Table 7, we propose four sequences of parameters for BLISS-B. The first two sequences provably and non-provably offer 128 bits of security. More precisely, the non-provably secure sequence offers by our analysis

| **Security level (bits)** | 128 | | 256 | |
|---|---|---|---|---|
| **Provably secure** | Yes | No | Yes | No |
| **Parameters** | | | | |
| Dimension $n$ | 1024 | 512 | 1024 | 1024 |
| Modulus $q$ | 18433 | 18433 | 40961 | 40961 |
| Secret key density $\delta_1$ | 0.022 | 0.062 | 0.040 | 0.040 |
| Secret key density $\delta_2$ | 0.357 | 0.498 | 0.531 | 0.531 |
| Gaussian standard deviation $\sigma$ | 630 | 311 | 900 | 900 |
| Parameter for norm of signatures $\alpha$ | 1.675 | 0.893 | 1.288 | 1.288 |
| Hamming weight $\kappa$ | 19 | 23 | 44 | 44 |
| Number of dropped bits $d$ | 10 | 10 | 6 | 6 |
| Verification threshold $B_2$ | 34636 | 13011 | 53252 | 53252 |
| Verification threshold $B_\infty$ | 2520 | 2177 | 3600 | 3600 |
| Repetition rate $M = \exp(1/(2\alpha^2))$ | 1.19 | 1.87 | 1.35 | 1.35 |
| **Sizes (bits)** | | | | |
| Verification key $n\lceil \log q \rceil$ | 15360 | 7680 | 16384 | 16384 |
| Signing key $2n\lceil \log 5 \rceil$ | 6144 | 3072 | 6144 | 6144 |
| Signature $\lvert \mathbf{z}_1 \rvert + \lvert \mathbf{z}_2^\dagger \rvert + n^\star$ | 15360 | 6656 | 21504 | 21504 |
| **Timings (milliseconds)** | | | | |
| Signing | 0.236 | 0.837 | 0.258 | 0.258 |
| Verifying | 0.064 | 0.029 | 0.062 | 0.062 |

$^\star$ Signature sizes are given as explained in the text.

Table 7: Parameter sequences, sizes, and timings for BLISS-B [DDLL13, Duc14].

65 bits of security, where the reduction loss is not taken into account. The provably secure sequence offers 128 bits by considering the gap function $\mathsf{Gap}_{\mathsf{BLISS}}$. The second (identical) two sequences offer 256 bits of security (see below). Table 7 also includes the corresponding bit sizes of the signing and verification keys and signatures in addition to the running time of signing and verifying. Signature sizes are computed without compression and according to [EB17, Lemma 14] in combination with [LPR10, Lemma 2.4], i.e., $|\mathbf{z}_1| = n(1 + \lceil \log(\sqrt{2\pi}\sigma) \rceil)$ and $|\mathbf{z}_2^\dagger| = n(1 + \lceil \log(\frac{\sqrt{2\pi}\sigma}{2^d}) \rceil)$.

**Remark 4.1.** We give some remarks on our parameter selection proposed in Table 7. As mentioned above, perfect immunity to the subfield attack imposes larger values for the parameters $d_1, d_2$ in order to obtain a vulnerability factor satisfying $F < 1$. It also requires $d_2$ to be larger than $d_1$ and hence in contrast to the parameters proposed by Ducas et al. [DDLL13], where $d_2 < d_1$[7]. Note that perfect immunity to the subfield attack was not considered by Ducas et al. [DDLL13] because it was published after [DDLL13, Duc14]. The large values of $d_1, d_2$ require to select a larger standard deviation $\sigma$ in order to obtain a smaller repetition rate $M$ and hence speeding up the signing process. The repetition rate $M$ is the expected number of times the signing process will need to be restarted and hence for efficiency reasons it is crucial to keep it as small as possible. In the sequence of parameters that is provably 128-bit secure, the dimension $n$ is increased to 1024 as opposed to 512 in the sequence that is non-provably 128-bit secure. Increasing the dimension was necessary for practicality reasons, as the large values of $d_1, d_2$ satisfying the perfect immunity condition would imply very high repetition rates for $n = 512$, hence resulting in an impractical scheme. Doubling the dimension $n$ (note that $n$ has to be a power of 2) resulted in doubling keys and signature sizes but faster signing speed. Interestingly for the security level 256, the provably and

[7] Ducas et al. [DDLL13] included few entries from $\{\pm 2\}$ in the secret key in order to increase resistance to the hybrid attack.

non-provably sequences of parameters are identical. This is because the large values of $d_1, d_2$ resulted in sequences of parameters offering even more bits of security than desired. In summary, perfect immunity to the subfield attack was the main reason for choosing larger parameters, which imply larger keys and signatures. One could choose to have a vulnerability factor $F$, which is somewhat larger than 1, e.g., 4 or 5, in order to obtain better sizes. We have chosen to obtain perfect immunity, since our main focus is security. Our analysis shows that taking the reduction loss into account when selecting parameters for BLISS-B not always affects the performance of the scheme, and even if it does, it still results in a practical scheme as long as security and speed have more precedence than sizes.

## Acknowledgements

This work has been co-funded by the DFG as part of project P1 within the CRC 1119 CROSSING. The first author would like to thank Florian Göpfert for illuminating discussions and Léo Ducas for helpful discussions about BLISS-B and for providing its source code.

## References

[ABB+16] Sedat Akleylek, Nina Bindel, Johannes Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In *International Conference on Cryptology AFRICACRYPT 2016*, pages 44–60. Springer, 2016. 2, 4

[ABB+17] Erdem Alkim, Nina Bindel, Johannes Buchmann, Özgür Dagdelen, Edward Eaton, Gus Gutoski, Juliane Krämer, and Filip Pawlega. Revisiting TESLA in the quantum random oracle model. In *PQCrypto 2017 - Eighth International Conference on Post-Quantum Cryptography*, 2017. http://eprint.iacr.org/2015/755. 2, 4

[ABD16] Martin R Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions. In *Advances in Cryptology–CRYPTO 2016*, pages 153–178. Springer, 2016. 27, 28

[ACF+15] Martin R Albrecht, Carlos Cid, Jean-Charles Faugere, Robert Fitzpatrick, and Ludovic Perret. On the complexity of the BKW algorithm on LWE. *Designs, Codes and Cryptography*, 74(2):325–354, 2015. 8

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology–CRYPTO 2009*, pages 595–618. Springer, 2009. 6

[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In *25th USENIX Security Symposium, USENIX Security 2016.*, pages 327–343. USENIX Association, 2016. 2

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996. 1, 7

[APS15] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015. https://bitbucket.org/malb/lwe-estimator/src, commit 92d83a0. 7, 8, 11, 19, 24

[AWHT16] Yoshinori Aono, Yuntao Wang, Takuya Hayashi, and Tsuyoshi Takagi. Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In *Advances in Cryptology–EUROCRYPT 2016*, pages 789–819. Springer, 2016. 8

[BCD$^+$16] Joppe 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 *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1006–1018. ACM, 2016. 2

[BCLvV16] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU Prime. Cryptology ePrint Archive, Report 2016/461, 2016. http://eprint.iacr.org/2016/461. 28

[BCNS15] Joppe W Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *2015 IEEE Symposium on Security and Privacy, SP 2015*, pages 553–570. IEEE, 2015. 2

[BG14] Shi Bai and Steven D Galbraith. An improved compression technique for signatures based on learning with errors. In *Cryptographers' Track at the RSA Conference*, pages 28–47. Springer, 2014. 2

[BLN$^+$16] Paulo S. L. M. Barreto, Patrick Longa, Michael Naehrig, Jefferson E. Ricardini, and Gustavo Zanon. Sharper Ring-LWE signatures. Cryptology ePrint Archive, Report 2016/1026, 2016. http://eprint.iacr.org/2016/1026. 4

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 390–399. ACM, 2006. 27

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures-how to sign with RSA and Rabin. In *Advances in Cryptology–EUROCRYPT '96*, pages 399–416. Springer, 1996. 27

[Che13] Yuanmi Chen. *Réduction de réseau et sécurité concrete du chiffrement completement homomorphe*. PhD thesis, ENS-Lyon, France, 2013. 8

[CKMS16] 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. 2, 3, 11, 27

[CN11] Yuanmi Chen and Phong Q Nguyen. BKZ 2.0: Better lattice security estimates. In *Advances in Cryptology–ASIACRYPT 2011*, pages 1–20. Springer, 2011. 8

[Cor00] Jean-Sébastien Coron. On the exact security of full domain hash. In *Advances in Cryptology–EUROCRYPT 2000*, pages 229–235. Springer, 2000. 27

[Cor02] Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In *Advances in Cryptology–EUROCRYPT 2002*, pages 272–287. Springer, 2002. 27

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal Gaussians. In *Advances in Cryptology–CRYPTO 2013*, pages 40–56. Springer, 2013. 2, 3, 4, 10, 15, 18, 25, 26, 27, 29, 35, 37

[DEG$^+$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 *Progress in Cryptology–LATINCRYPT 2014*, pages 84–103. Springer, 2014. 2

[Duc14] Léo Ducas. Accelerating BLISS: the geometry of ternary polynomials. Cryptology ePrint Archive, Report 2014/874, 2014. `http://eprint.iacr.org/2014/874`. 3, 4, 10, 15, 25, 26, 29, 35, 37

[EB13] Rachid El Bansarkhani and Johannes Buchmann. Improvement and efficient implementation of a lattice-based signature scheme. In *Selected Areas in Cryptography–SAC 2013*, pages 48–67. Springer, 2013. 2

[EB17] Rachid El Bansarkhani. LARA - A design concept for lattice-based encryption. Cryptology ePrint Archive, Report 2017/049, 2017. `http://eprint.iacr.org/2017/049`. 2, 3, 4, 10, 15, 20, 21, 22, 25, 29, 35, 36

[EDB15] Rachid El Bansarkhani, Özgür Dagdelen, and Johannes Buchmann. Augmented learning with errors: The untapped potential of the error term. In *Financial Cryptography and Data Security - 19th International Conference, FC 2015, Puerto Rico, January 26 - 30, 2015*, pages 333–352. Springer, 2015. 7, 20, 23

[EE16] Rachid El Bansarkhani and Ali El Kaafarani. Post-quantum attribute-based signatures from lattice assumptions. Cryptology ePrint Archive, Report 2016/823, 2016. `http://eprint.iacr.org/2016/823`. 1

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Theory of computing, STOC 2009*, pages 169–178. ACM Press, 2009. 1

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology–EUROCRYPT 2013*, pages 1–17. Springer, 2013. 1

[GJKW07] Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient signature schemes with tight reductions to the Diffie-Hellman problems. *Journal of Cryptology*, 20(4):493–514, 2007. 27

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *Cryptographic Hardware and Embedded Systems–CHES 2012*, pages 530–547. Springer, 2012. 2, 4

[GN08] Nicolas Gama and Phong Q Nguyen. Predicting lattice reduction. In *Advances in Cryptology–EUROCRYPT 2008*, pages 31–51. Springer, 2008. 8

[GS02] Craig Gentry and Mike Szydlo. Cryptanalysis of the revised NTRU signature scheme. In *Advances in Cryptology–EUROCRYPT 2002*, pages 299–320. Springer, 2002. 27

[GvVW17] Florian Göpfert, Christine van Vredendaal, and Thomas Wunderer. A hybrid lattice basis reduction and quantum search attack on LWE. In *PQCrypto 2017 - Eighth International Conference on Post-Quantum Cryptography*, 2017. 8

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing, STOC 2013*, pages 545–554. ACM Press, 2013. 1

[How07] Nick Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In *Advances in Cryptology–CRYPTO 2007*, pages 150–169. Springer, 2007. 27

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. NTRU: A ring-based public key cryptosystem. In *Algorithmic number theory*, pages 267–288. Springer, 1998. 25, 27

[HPS$^+$17] Jeffrey Hoffstein, Jill Pipher, John M Schanck, Joseph H Silverman, William Whyte, and Zhenfei Zhang. Choosing parameters for NTRUEncrypt. In *Cryptographers' Track at the RSA Conference*, pages 3–18. Springer, 2017. 28

[KF16] Paul Kirchner and Pierre-Alain Fouque. Comparison between subfield and straightforward attacks on NTRU. Cryptology ePrint Archive, Report 2016/717, 2016. http://eprint.iacr.org/2016/717. 27, 28

[KM06] Neal Koblitz and Alfred Menezes. Another look at "provable security". II. In *International Conference on Cryptology in India*, pages 148–175. Springer, 2006. 11, 27

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In *Topics in Cryptology–CT-RSA 2011*, pages 319–339. Springer, 2011. 2, 3, 4, 10, 15, 16, 17, 20, 35, 36

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Advances in Cryptology–EUROCRYPT 2010*, pages 1–23. Springer, 2010. 6, 29

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Advances in Cryptology–EUROCRYPT 2012*, pages 738–755. Springer, 2012. 2, 4

[Mic02] Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In *Proceedings of the 43rd Symposium on Foundations of Computer Science FOCS*, pages 356–365. IEEE, 2002. 7

[MP13] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In *Advances in Cryptology–CRYPTO 2013*, pages 21–39. Springer, 2013. 6, 7

[MR09] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009. 2

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 333–342. ACM, 2009. 6

[Pei16] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016. 6, 7

[PR07] Chris Peikert and Alon Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 478–487. ACM, 2007. 7

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 84–93. ACM, 2005. 1, 6

[SS11] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *Advances in Cryptology–EUROCRYPT 2011*, pages 27–47. Springer, 2011. 25

[Wun16] Thomas Wunderer. Revisiting the hybrid attack: Improved analysis and refined security estimates. Cryptology ePrint Archive, Report 2016/733, 2016. http://eprint.iacr.org/2016/733. 8, 28

# Appendix

## A Cryptographic Definitions

This section formally defines public-key encryption schemes with their security under chosen-plaintext attacks, and digital signature schemes with their security under adaptive chosen-message attacks.

### A.1 Public-Key Encryption

**Definition A.1** (Public-Key Encryption Scheme)**.** A public-key encryption scheme with key space $\mathcal{K}$, message space $\mathcal{M}$, and ciphertext space $\mathcal{C}$ is a tuple of polynomial-time algorithms $\Pi =$(KGen, Enc, Dec) such that

- KGen is a probabilistic key generation algorithm that takes as input $1^\ell$, for a security parameter $\ell \in \mathbb{N}$, and outputs a pair of keys $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{K}$, i.e., $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\ell)$, where pk is a public key and sk is a secret or private key.
- Enc is a probabilistic encryption algorithm that takes as input a public key pk and a message $\mu \in \mathcal{M}$. It outputs a ciphertext $c \in \mathcal{C}$, i.e., $c \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu)$.
- Dec is deterministic decryption algorithm that takes as input a secret key sk and a ciphertext $c \in \mathcal{C}$. It outputs a message $\mu \in \mathcal{M}$ or a special symbol $\perp$ denoting failure, i.e., $\mu \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$ or $\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$.

A public-key encryption scheme $\Pi =$(KGen, Enc, Dec) requires the perfect correctness property, which states that the decryption algorithm always outputs correctly encrypted messages except with possibly negligible probability, i.e., for all $\ell \in \mathbb{N}, (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\ell), \mu \in \mathcal{M}$, and all $c \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu)$, it holds that $\text{Prob}[\mathsf{Dec}(\mathsf{sk}, c) \neq \mu] \leq \text{negl}(\ell)$.

**Definition A.2** (IND-CPA Security)**.** A public-key encryption scheme $\Pi =$(KGen, Enc, Dec) with security parameter $\ell \in \mathbb{N}$ is secure in terms of indistinguishability under chosen-plaintext attack, or simply IND-CPA-secure, if for any probabilistic polynomial-time adversary $\mathcal{A}$ the following IND-CPA game outputs the bit 1 only with negligible probability over $\frac{1}{2}$, i.e., $\text{Prob}[\mathsf{IND\text{-}CPA}_{\Pi,\mathcal{A}}(\ell) = 1] \leq \frac{1}{2} + \text{negl}(\ell)$.

**Game** $\mathsf{IND\text{-}CPA}_{\Pi,\mathcal{A}}(\ell)$ :

```
1: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\ell)$
2: $b \leftarrow_\$ \{0, 1\}$
3: $b' \leftarrow \mathcal{A}^{\mathcal{O}(\mathsf{pk}, b, \cdot, \cdot)}(\mathsf{pk})$, where $\mathcal{O}(\mathsf{pk}, b, \cdot, \cdot) = \mathsf{Enc}(\mathsf{pk}, \mu_b)$ for $|\mu_0| = |\mu_1|$
4: if $b' = b$ : return 1
5: else : return 0
```

As done by proving the IND-CPA security of the encryption schemes LP [LP11] and LARA [EB17], it is sufficient to show that an adversary has a negligible advantage in distinguishing a ciphertext from a uniformly random value from $\mathcal{C}$.

## A.2 Digital Signatures

**Definition A.3** (Digital Signature Scheme)**.** A digital signature scheme with key space $\mathcal{K}$, message space $\mathcal{M}$, and signature space $\mathcal{S}$ is a tuple of polynomial-time algorithms $\Sigma$ =(KGen, Sign, Verify) such that

- KGen is a probabilistic key generation algorithm that takes as input $1^\ell$, for a security parameter $\ell \in \mathbb{N}$, and outputs a pair of keys $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{K}$, i.e., $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\ell)$, where pk is a public or verification key and sk is a secret or signing key.
- Sign is a probabilistic signature algorithm that takes as input a secret key sk and a message $\mu \in \mathcal{M}$. It outputs a signature $s \in \mathcal{S}$, i.e., $s \leftarrow \mathsf{Sign}(\mathsf{sk}, \mu)$.
- Verify is a deterministic verification algorithm that takes as input a public key pk, a message $\mu \in \mathcal{M}$, and a signature $s \in \mathcal{S}$. It outputs a bit $b$, where $b = 1$ if the algorithm accepts and $b = 0$ if it rejects, i.e., $b \leftarrow \mathsf{Verify}(\mathsf{pk}, \mu, s)$, where $b \in \{0, 1\}$.

A digital signature scheme $\Sigma$ =(KGen, Sign, Verify) requires the perfect correctness property, which states that the verification algorithm always validates correctly signed messages, i.e., for all $\ell \in \mathbb{N}, (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\ell), \mu \in \mathcal{M}$, and all $s \leftarrow \mathsf{Sign}(\mathsf{sk}, \mu)$, it holds that $\mathrm{Prob}[\mathsf{Verify}(\mathsf{pk}, \mu, s) = 1] = 1$.

**Definition A.4** (EUF-CMA Security)**.** A digital signature scheme $\Sigma$ =(KGen, Sign, Verify) with security parameter $\ell \in \mathbb{N}$ is existentially unforgeable under an adaptive chosen-message attack, or EUF-CMA-secure, in the random oracle model if for any probabilistic polynomial-time adversary $\mathcal{A}$ that makes at most $q_s$ queries to the signing oracle and at most $q_h$ queries to the random oracle, the following EUF-CMA game outputs the bit 1 only with negligible probability, i.e., $\mathrm{Prob}[\mathsf{EUF\text{-}CMA}_{\Sigma,\mathcal{A}}(\ell) = 1] \leq \mathrm{negl}(\ell)$.

**Game** $\mathsf{EUF\text{-}CMA}_{\Sigma,\mathcal{A}}(\ell)$ :

```
1: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\ell)$
2: $(\mu^*, s^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),H(\cdot)}(\mathsf{pk})$, where $H(\cdot)$ is a random oracle
3: if $\mathsf{Verify}(\mathsf{pk}, s^*, \mu^*) = 1 \wedge \mu^* \notin \mathcal{Q}$ : return 1
4: else : return 0
```

```
if $\mathcal{A}$ queries $\mathsf{Sign}(\mathsf{sk}, \mu)$ :
  1: $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\mu\}$
  2: $s \leftarrow \mathsf{Sign}(\mathsf{sk}, \mu)$
  3: return $s$
```

The scheme $\Sigma$ is strongly EUF-CMA-secure if the success probability of $\mathcal{A}$ is negligible in a game defined as above except for the third line, which changes to

**if** $\mathsf{Verify}(\mathsf{pk}, s^*, \mu^*) = 1 \wedge (\mu^*, s^*) \notin \{(\mu_1, s_1), \ldots, (\mu_{q_s}, s_{q_s})\}$ : **return** 1, where $\mathcal{Q} = \{\mu_1, \ldots, \mu_{q_s}\}$.

# B Lattice-Based Encryption and Signature Schemes

In this section we recall the description of the public-key encryptions schemes LP [LP11] and LARA [EB17] as well as the digital signature scheme BLISS-B [DDLL13, Duc14].

## B.1 The Encryption Scheme LP

The ring variant of the scheme LP [LP11], given in Algorithm 1, is operated on a ring $R_q = \mathbb{Z}_q[x]/\langle f(x)\rangle$ for some $n$-degree and monic polynomial that is irreducible over $\mathbb{Z}$, typically $f(x) = x^n + 1$ for $n$ a power of 2. The scheme uses error distributions $\chi_k, \chi_e$ over $R_q$ for key generation and encryption, respectively. In Algorithm 1, both distributions are discrete Gaussian, which is a typical choice in lattice-based cryptography, i.e., $\chi_k = D^n_{r_k}, \chi_e = D^n_{r_e}$ for Gaussian parameters $r_k, r_e$. For the message space $\{0,1\}^n$, error-tolerant encoding and decoding functions are required. These functions are given by $\mathsf{encode} : \{0,1\}^n \longrightarrow R_q$ and $\mathsf{decode} : R_q \longrightarrow \{0,1\}^n$ such that $\mathsf{decode}(\mathsf{encode}(\mathbf{m}) + e \mod q) = \mathbf{m}$ for a polynomial $e \in R_q$ with entries in $(-t, t]$, where $t \geq 1$ is an integer threshold. The scheme also uses a uniformly random polynomial $a \in R_q$, which can be chosen by the user or generated by a trusted source.

**Algorithm 1** Description of the public-key encryption scheme LP [LP11].

```
KGen(1^ℓ, a) :
1: $r_1, r_2 \leftarrow D^n_{r_k}$
2: $p \leftarrow r_1 - ar_2 \pmod q$
3: $\mathsf{sk} \leftarrow r_2$
4: $\mathsf{pk} \leftarrow (a, p)$
5: return $(\mathsf{pk}, \mathsf{sk})$
```

```
Enc(a, p, m) :
1: $e_1, e_2, e_3 \leftarrow D^n_{r_e}$
2: $\mu \leftarrow \mathsf{encode}(\mathbf{m})$
3: $c_1 \leftarrow ae_1 + e_2 \pmod q$
4: $c_2 \leftarrow pe_1 + e_3 + \mu \pmod q$
5: return $(c_1, c_2)$
```

```
Dec(r_2, c_1, c_2) :
1: $\mathbf{m} \leftarrow \mathsf{decode}(c_1 r_2 + c_2)$
2: return $\mathbf{m}$
```

## B.2 The Encryption Scheme LARA

The variant of the scheme LARA [EB17] we recall in Algorithm 2 is IND-CPA-secure in the random oracle model and for simplicity is given without the high data load encryption mode, which allows to increase the message throughput per ciphertext. The scheme is operated on a ring $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$, where $n$ and $q = 2^k$ are power of 2. The message space is $\{0,1\}^{3n \log p}$, where $p$ is a power of 2. Two cryptographic hash functions (modeled as random oracles) $H_1, H_2$ are used, such that $H_1 : \{0,1\}^n \longrightarrow \mathbb{Z}^n_{2^{k-1}}$ and $H_2 : R_q \longrightarrow \{0,1\}^{3n \log p}$. The message encoding and decoding functions are given by $\mathsf{encode} : \{0,1\}^{3n \log p} \longrightarrow \mathbb{Z}^n_p \times \mathbb{Z}^n_p \times \mathbb{Z}^n_p$, $\mathsf{decode} : \mathbb{Z}^n_p \times \mathbb{Z}^n_p \times \mathbb{Z}^n_p \longrightarrow \{0,1\}^{3n \log p}$. Furthermore, the scheme uses two uniformly random polynomials $a_1 \in R_q, a_2 \in R^\times_q$, which can be generated by a trusted source or chosen by the user.

**Algorithm 2** Description of the public-key encryption scheme LARA [EB17].

```
KGen(1^ℓ, a_1, a_2) :
1: $z_1, z_2 \leftarrow D^n_{r_1}$
2: $a_3 \leftarrow 2^{k-1} - (a_1 z_1 + a_2 z_2)$
3: $\mathbf{a} \leftarrow [a_1, a_2, a_3]$
4: $\mathsf{sk} \leftarrow (z_1, z_2)$
5: $\mathsf{pk} \leftarrow \mathbf{a}$
6: return $(\mathsf{pk}, \mathsf{sk})$
```

```
Enc(a, m) :
1: $\mathbf{x} = (x_1, \dots, x_n) \leftarrow_\$ \{0,1\}^n$
2: $\mathbf{y} = (y_1, \dots, y_n) \leftarrow H_1(\mathbf{x})$
3: $s_i \leftarrow y_i \| x_i \in \mathbb{Z}_q, \ i \in [n]$
4: $s \leftarrow (s_1, \dots, s_n) \in R_q$
5: $\mathbf{r} \leftarrow H_2(s)$
6: $[\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3] \leftarrow \mathsf{encode}(\mathbf{r} \oplus \mathbf{m})$
7: $e_j \leftarrow D_{p\mathbb{Z}^n + \mathbf{v}_j, r_2}, \ j \in [3]$
8: $\mathbf{e} \leftarrow [e_1, e_2, e_3]$
9: $\mathbf{c} \leftarrow \mathbf{a}s + \mathbf{e}$
10: return $\mathbf{c}$
```

```
Dec(z_1, z_2, c) :
1: $u = 2^{k-1}s + e \leftarrow c_3 + c_1 z_1 + c_2 z_2$,
   where $e \in R_q$ small
2: if $u_i$ is closer to $q/2$ than to $0$ :
       $1 = \mathsf{LSB}(s_i) \leftarrow x_i$
3: else : $0 = \mathsf{LSB}(s_i) \leftarrow x_i$
4: $\mathbf{y} \leftarrow H_1(x_1, \dots, x_n)$
5: $s_i \leftarrow y_i \| x_i \in \mathbb{Z}_q, \ i \in [n]$
6: $s \leftarrow (s_1, \dots, s_n) \in R_q$
7: $\mathbf{r} \leftarrow H_2(s)$
8: $\mathbf{e} \leftarrow \mathbf{c} - \mathbf{a}s$
9: if $\|\mathbf{e}\| > r_2\sqrt{3n}$ : return $\perp$
10: $\mathbf{m} \leftarrow \mathsf{decode}(\mathbf{e} \mod p) \oplus \mathbf{r}$
11: return $\mathbf{m}$
```

## B.3 The Signature Scheme BLISS-B

The scheme BLISS-B [DDLL13,Duc14] we give in Algorithm 3 is operated on the rings $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ and $R_{2q} = \mathbb{Z}_{2q}[x]/\langle x^n + 1\rangle$, where $n$ is a power of two and $q = 1 \mod 2n$. For given densities $\delta_1, \delta_2 \in [0, 1)$, the integers $d_1 = \lceil \delta_1 n \rceil$ and $d_2 = \lceil \delta_2 n \rceil$ correspond to the number of entries in $\{\pm 1\}$ and $\{\pm 2\}$ (respectively) that are included in the polynomials of the secret key. A cryptographic hash function (modeled as a random oracle) $H : \{0,1\}^* \longrightarrow \mathbb{B}^n_\kappa$ is required, where $\mathbb{B}^n_\kappa$ is the set of binary vectors of length $n$ and Hamming Weight $\kappa$. For reducing the signature size, a positive integer $d$ denotes the number of dropped bits in a signature. For any integer $x \in [-q, q)$, the rounding operator $\lfloor x \rceil_d$ is defined as $\lfloor x \rceil_d = (x - [x \mod 2^d])/2^d$, where $[x \mod 2^d]$ is the unique representative of $x \mod 2^d$ in the set $[-2^{d-1}, 2^{d-1})$. A positive real $M$ denotes the repetition rate of the signing algorithm.

**Algorithm 3** Description of the digital signature scheme BLISS-B [DDLL13,Duc14].

```
KGen(1^ℓ, d_1, d_2) :
1: Sample two uniformly random polynomials $f, g$ with exactly $d_1$ entries in $\{\pm 1\}$ and $d_2$ entries in $\{\pm 2\}$
2: if $f \notin R_q^\times$ : restart
3: $\mathbf{s} \leftarrow (f, 2g + 1)$
4: $\mathbf{a}_q \leftarrow (2g + 1)/f \mod q$
5: $\mathbf{a}_1 \leftarrow 2\mathbf{a}_q$
6: $\mathbf{A} \leftarrow (\mathbf{a}_1, q - 2) \mod 2q$
7: $\mathsf{sk} \leftarrow \mathbf{s}$
8: $\mathsf{pk} \leftarrow \mathbf{A}$
9: return $(\mathsf{pk}, \mathsf{sk})$
```

```
Sign(s, A, μ) :
1: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_\sigma^n$
2: $\mathbf{u} \leftarrow \zeta \mathbf{a}_1 \mathbf{y}_1 + \mathbf{y}_2 \mod 2q$, where $\zeta(q-2) = 1 \mod 2q$
3: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rceil_d \mod p, \mu)$, where $p = \lfloor 2q/2^d \rceil$
4: $(\mathbf{v}_1, \mathbf{v}_2) \leftarrow \mathsf{GreedySC}(\mathbf{s}, \mathbf{c})$ see [Duc14, Algorithm 1]
5: $b \leftarrow_\$ \{0, 1\}$
6: $(\mathbf{z}_1, \mathbf{z}_2) \leftarrow (\mathbf{y}_1, \mathbf{y}_2) + (-1)^b \cdot (\mathbf{v}_1, \mathbf{v}_2)$
7: continue with probability $1/\left(M \exp(-\frac{\|\mathbf{v}\|^2}{2\sigma^2}) \cosh(\frac{\langle \mathbf{z}, \mathbf{v} \rangle}{\sigma^2})\right)$, otherwise restart
8: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rceil_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rceil_d) \mod p$
9: return $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
```

```
Verify(A, μ, z_1, z_2^†, c) :
1: if $\|\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger\| > B_2$ : reject
2: if $\|\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger\|_\infty > B_\infty$ : reject
3: if $\mathbf{c} = H(\lfloor \zeta \mathbf{a}_1 \mathbf{z}_1 + \zeta q \mathbf{c} \rceil_d + \mathbf{z}_2^\dagger \mod p, \mu)$ : return 1
4: else : return 0
```