The Design of Lattice-based Key-Encapsulation Mechanisms

Mark Schultz

June 15, 2021

Abstract

We survey recent trends in the design of lattice-based KEMs, where we highlight the role played by certain coding-theoretic objects that are often implicitly defined in an exceedingly simple way. This simple definition is known to be non-optimal from the perspective of the size of ciphertexts of the resulting scheme. After surveying the general design paradigm of lattice-based KEMs, we survey recent papers exploring the use of better encodings in lattice-based cryptosystems.

1 Introduction

Lattice-based cryptography has emerged as a leading candidate for both key-encapsulation mechanisms (KEMs) and digitial signature schemes that are secure against quantum-capable adversaries (known as "post-quantum cryptography"). These primitives are of vital importance to Transport Layer Security (TLS), which is a backbone of the modern internet.

The American National Institute of Standards and Technology (NIST) is currently in the process of standardizing a post-quantum KEM and signature scheme in a competition (the NIST PQC competition). This has led to a great deal of academic interest in the design of practical lattice-based KEMs. We survey a selection of papers in the area, featuring a mixture of theoretical and practical works, with the goal of exploring future directions in the development of practical lattice-based KEMs.

Throughout we will routinely refer to the cryptosystems Kyber [6], Saber [18], and LAC [32]. The first two are the lattice-based KEMs that have been selected as finalists in NIST's PQC competition, while the later is a scheme that China has recently standardized[1] via an analogous competition as a post-quantum KEM.

[1] LAC was submitted to the NIST PQC competition as well, but did not advance past the second (out of three) round.

1.1 Goals of this Survey

In this survey, we have two somewhat related goals, that in hindsight it may have been better to address separately. These are,

- examining different non-trivial encodings used in the design of lattice-cryptographic protocols, with the goal of highlighting the rate benefits of these encodings, and
- abstracting the notion of "arithmetic" and "smallness" one works with, with the intent of enabling future generalization of our lattice-cryptographic KEMs to coding-theoretic KEMs and "Mersenne Prime"-based KEMs [1].

While these both benefit from parametrizing current lattice-based schemes by an explicit underlying choice of error-tolerant encoding, trying to combine the very abstract with the very concrete perhaps has led to sub-optimal results, and it is likely too late to excise one of these topics from the survey. For this reason, we keep both of them, and the tension between "very abstract" and "very concrete", in the body of the survey, but motivate their study separately in the introduction.

1.1.1 Steps towards Uniformly covering Lattice, Code, and Mersenne Prime-based Cryptography

A paper [9] we survey in sub-section 4.1 argues that if one appropriately abstracts both the space one does arithmetic in and the notion of "small" one works with, one can uniformly describe the hardness assumptions of

- lattice-based cryptography, for example the LWE assumption, as well as
- (parts of)[2] coding-based cryptography, for example the Learning Parity with Noise assumption, and
- "Mersenne prime"-based cryptography [1], specifically the I-RLWE assumption of [15].

Historically this is not the most surprising — the LWE assumption was compared to the LPN assumption when it was introduced, and "Mersenne Prime"-based cryptography was specifically introduced as a non-standard variant[3] of lattice-based cryptography (and furthermore, can explicitly reduce its security to standard problems in lattice-based cryptography).

Despite the broad similarities at a heuristic level, work in each area often can often stay separate from the other areas. The LLL basis reduction algorithm [31] was first published in 1982, while the first analogue of a basis reduction

[2] There exist other areas of coding-based cryptography, namely "Rank Metric" codes, for which this is non-obvious but plausible.

[3] Specifically, the intention is to perform arithmetic $\sum_i a_i b^i \bmod f(b)$ rather than $\sum_i a_i x^i \bmod f(x)$, e.g. "big integer" arithmetic rather than polynomial arithmetic. The intention of this is to be able to reuse RSA coprocessors.

Scheme	Public Key Size	Ciphertext Size	Shared-key Size	PQ-Secure?
ECDH (256-bit field)	64	64	128	No
FrodoKEM-640 [10]	9,616	9,720	128	Yes!
NewHope512 [5]	928	1088	128	Yes
AKCN-E8-512 [28]	928	960	256	Yes

Figure 1: Sizes (in bytes) for a variety of IND-CPA-secure KEMs that are thought to achieve 128-bit security. The large sizes of FrodoKEM (compared to the other lattice-based schemes) are due to it relying on a more conservative hardness assumption ("unstructured" LWE vs. "structured" LWE).

algorithm for codes came out a few months ago [19], nearly 40 years later. Other examples exist, namely worst-case to average-case reductions for LPN first appeared in [14], over 10 years after Regev's reduction for LWE [38], and over 20 years after Ajtai's first worst-case to average-case reduction for lattice problems [2].

We see this state of affairs as somewhat regrettable, and describe our (lattice-based) constructions in a more general language to prepare future work where we uniformly describe *protocols* in all three areas in a uniform language. It is plausible this can be done in a way that has mild benefits purely for lattice-based cryptography, even when considered in isolation. Specifically, the modularization of constructions that this would require could reduce the number of parameters that one has to consider at a time in lattice-cryptographic protocols[4], which would be somewhat nice.

1.1.2 Smaller Lattice-based Ciphertexts from Non-trivial Encodings

As mentioned, our two topics are unfortunately only somewhat related. On the more practical side of things, we focus on the *size* of both public keys and ciphertexts in lattice-based KEMs. This is a metric that lattice-based cryptography performs poorly on compared to the elliptic curve-based state of the art (see explicit numbers in figure 1). While the impact of the order of magnitude performance degredation is application dependent, we see it as a particular pain point in the switch to post-quantum secure cryptosystems, so focus on it as an area that needs improvement.

To this end, we see two main paths forward that are relatively unexplored. Simplistically, lattice-based KEMs often implicitly depend on an underlying choice of an

- error-tolerant encoding, and a
- lossy compression mechanism.

[4] Currently, constructions are often parametrized by integers n, m, q, probability distributions χ_e, χ_s on $\mathbb{Z}_q$, and an irreducible polynomial$f(x) \in \mathbb{Z}_q[x]$.

Current schemes often set these to be roughly trivial. For example, to encode n dimensional vectors $\mathbf{x} \in \mathbb{Z}_q^n$, one often simply scales up (for error-correction properties) or scales down and rounds (for compression properties) each coordinate independently. Note that the extent to which current schemes do both of the above is even limited — while the aforementioned lossy compression mechanism is practically very common, many theorists do not even bother mentioning it when discussing constructions.

We see possibilities to improve on the practical size of lattice-based KEMs via both techniques. While we survey them both in section 6, we mention two highlights here, namely that

- the scheme AKCN-E8-512 has mildly smaller ciphertexts than NewHope512 (both displayed in fig. 1), at roughly the same computational cost, while *doubling* the size of the shared key one agrees to, and
- the non-trivial ciphertext compression scheme of [13] can compress a ciphertext $(\mathbf{u}, \mathbf{v}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$ to $\mathbb{Z}_q^n \times \mathbb{Z}_q \times \{0,1\}^n$.

The compression scheme in the second example can be computed via a surprisingly simple algorithm — it is easily quasi-linear time[5], and should have "good constants", although we have not practically implemented it yet. Still, it seems plausible one could reduce ciphertexts by a few hundred bytes at some mild (or even no) overhead compared to the state of the art. Of course, when designing practical schemes there are many other things to evaluate. It seems quite possible more complicated encodings could negatively impact the

- ease of implementing countermeasures against side-channel attacks, or the
- ease of (correct) implementation

While it is plausible points such of these may make the potential rate gains not worth it, we find that rate gains seem to be "left on the table" currently to be a compelling reason to highlight a plausible avenue to achieving them in our survey on lattice-based KEMs, with the intent of evaluating their efficacy practically in the future.

2 Preliminaries

2.1 Notation

We will use the notation s for scalars, $\mathbf{s}$ for vectors, $\mathbf{S}$ for matrices, $\mathcal{S}$ for distributions (and adversaries), and $\mathbb{S}$ for sets. We use $\lfloor x \rceil$ to denote the operation of rounding to the closest integer, extended component-wise when applied to vectors/matrices.

[5] There even exist linear time algorithms that replace a sorting algorithm call (the source of the quasi-linear complexity) to a somewhat non-standard variant of sorting that executes in $O(n)$ time, although it is not clear if this is useful for non-asymptotic settings.

2.2 The Arithmetic of this Survey

Lattice-based cryptography often uses relatively complex mathematics to design practical schemes. For instance, one of the leading hardness assumptions within lattice-based cryptography is that of Module Learning with Errors over (power-of-two cyclotomics). This combines the study of:

1. The polynomial ring $\mathbb{Z}_q[x]/(x^{2^k}+1)$.
2. Modules over this ring, i.e. "linear algebra" with coefficients in the ring.

While we could begin this survey with a careful introduction to these algebraic structures, we view this as somewhat counterproductive to our goal of highlighting the non-trivial encodings that have started appearing in the design of lattice-based KEMs.

For this reason, we write our survey working with general $\mathbb{A}$-modules for some finite commutative ring $\mathbb{A}$. We describe all protocols in terms of vectors and matrices with coefficients in $\mathbb{A}$, and encourage less algebraically-inclined readers to fix the case of $\mathbb{A} = \mathbb{F}_q$ for q prime, where things reduce to literal linear algebra. We do not generically make this simplification as lattice-based KEMs over $\mathbb{F}_q$ have quite large ciphertexts (see FrodoKEM640 in figure 1), so it is important to our purpose of desigining high-rate lattice-based KEMs to be able to later specialize $\mathbb{A}$ to $\mathbb{Z}_q[x]/(x^{2^k}+1)$ (or other similar structures).

Note that this "linear algebraic" description allows us to omit specifing what a lattice is. Readers should view this as analogous to expositions of elliptic curve protocols that compartmentalize the mathematical treatment of elliptic curves, and solely use them as the source of a group with an assumed hard problem.

2.3 Small Elements in this Survey

The paper is fairly vague on how someone should measure "smallness" in the first place, so we first discuss some nuances in defining the notion, before providing a general definition.

Typically coding theory studies *codes*, meaning subsets $C \subseteq \mathbb{F}_2^n$ (see [16] chapter 3). There are two primary notions of "small" associated with codes of interest to us, namely

- how far apart any $\mathbf{c}_1, \mathbf{c}_2 \in C$ are from eachother (often called the *minimum distance* of the code), which bounds the size of errors one can *remove* from any noisy codeword $\mathbf{c} + \mathbf{e} \mapsto \mathbf{c}$, and
- how far any arbitrary $\mathbf{x} \in \mathbb{F}_2^n$ is from some codeword $\mathbf{c}$ (often called the *covering radius* of the code), which bounds the size of errors *induced* when mapping some $\mathbf{x} \in \mathbb{F}_2^n$ to the nearest codeword.

In the above, "size" and "nearest" are measured in the Hamming metric. There is a similar story in Euclidean space, where one considers $C \subseteq \mathbb{R}^n$, and measures "size" and "nearest" in the ℓ_2 metric[6].

Given these similar stories, we present our survey in terms of codes in general metrics, with three caveats, namely that

- as we want to use the codes to encode data, we do not solely define a set $C \subseteq \mathbb{X}$ for some metric space $\mathbb{X}$, but also encoding and decoding functions

$$\mathsf{encode} : \mathbb{M} \to C \subseteq \mathbb{X}, \qquad \mathsf{decode} : \mathbb{X} \to \mathbb{M}$$

- We require C be linear, e.g. $\mathbb{M}$ and $\mathbb{X}$ to be some groups, where encode is a group homomorphism.
- We want the metric d on $\mathbb{X}$ to respect the group structure on $\mathbb{X}$, which is called being *translation-invariant*, or concretely satisfying

$$\forall a, b, c \in \mathbb{X} : d(a + c, b + c) = d(a, b)$$

As it is too strict to require that decode is linear as well[7], we instead we will require that it partitions $\mathbb{X}$ into translations of some fixed set $\mathsf{decode}^{-1}(0)$. A compact way to write this (which for Euclidean space has recently appeared in [36]) is that the code must satisfy

$$\forall m \in \mathbb{M}, \forall e \in \mathbb{X} : \mathsf{decode}(\mathsf{encode}(m) + e) = m + \mathsf{decode}(e), \tag{1}$$

so solely a "partial" linearity property. We next introduce the notion of an abstract linear code, which can be seen as a generalization of notions from lattice coding (see chapter 2 of [42]) and standard (e.g. binary, with Hamming weight) coding.

Definition 1. *Let* $\mathbb{M}, \mathbb{X}$ *be abelian groups with a translation-invariant metric* d *on* $\mathbb{X}$*, and let* $\mathsf{encode} : \mathbb{M} \to \mathbb{X}$ *be a homomorphism, and* $\mathsf{decode} : \mathbb{X} \to \mathbb{M}$ *be a function that satisfies eq.* (1)*. We call the pair* $C = (\mathsf{encode}, \mathsf{decode})$ *a* linear code *with message space* $\mathbb{M}$ *and ambient space* $\mathbb{X}$*. We define the subset* $C.\mathsf{domain} = \{x \in \mathbb{X} \mid \mathsf{decode}(x) = 0\} = \mathsf{decode}^{-1}(0)$ *to be the fundamental domain of* $\mathbb{X}$*.*

For any code C*, we define the functions:*

$$\lfloor x \rceil_C = \mathsf{encode}(\mathsf{decode}(x)), \qquad x \bmod C = x - \lfloor x \rceil_C$$

which decompose any $x \in \mathbb{X}$ *uniquely into* $\lfloor x \rceil_C \in \mathsf{encode}(\mathbb{M}) = C$*, and* $x \bmod C \in C.\mathsf{domain}$*.*

From these definitions, it is straightforward to establish codes can be used for error-correction and compression.

[6] Note that translating results to other ℓ_p metrics is not that difficult, as for $p \geq 1$ all ℓ_p metrics are equivalent up scaling by some dimension-dependent factor.

[7] If one has the simple correctness condition $\mathsf{encode}(\mathsf{decode}(m)) = m$ for all $m \in \mathbb{M}$, by basic group theory this implies that $\mathbb{M} \cong \mathbb{X}$, and no non-trivial error correction or compression can occur.

Theorem 1. *Let $C = (\mathsf{encode}, \mathsf{decode})$ be a linear code with message space $\mathbb{M}$ and ambient space $\mathbb{X}$. Then C is*

1. ***Error-Correcting:*** $\forall m \in \mathbb{M}, \forall e \in C.\mathsf{domain} : \mathsf{decode}(\mathsf{encode}(m) + e) = m$.
2. ***Lossily Compressing:*** $\forall x \in \mathbb{X}$, $\lfloor x \rceil_C$ *approximates x up to bounded error $x \bmod C$, while having a succinct representation via an element* $\mathsf{decode}(\lfloor x \rceil_C) \in \mathbb{M}$.

Note that while decode fails to be injective in general, when restricted to $\mathsf{encode}(\mathbb{M})$ decode is a bijection. There exist weakenings[8] of this notion which are interesting in practice, but we do not have space to treat them. Throughout this survey, we will freely use that a linear code C with message space $\mathbb{M}$ can be applied component-wise to give a linear code C^n with message space $\mathbb{M}^n$, justifying discussing codes in the smallest dimension they occur within.

We will frequently refer to the following code throughout this survey, which we call the *scaling code*.

Example 1 (The Scaling Code)**.** *Let $p \mid q$ (and often in particular $p = 2$), and for $m \in \mathbb{Z}_p$ define:*

$$\mathsf{encode}(k) = (q/p)k, \qquad \mathsf{decode}(c) = \lfloor (p/q)c \rceil$$

This code can correct any error contained in $[-q/2p, q/2p)$.

This construction appeared in the first cryptosystem built under the Learning with Errors assumption [38], and has been used widely thought the field since[9]. For the the first part of this survey (especially section 3) of this survey, the reader can freely replace $C = (\mathsf{encode}, \mathsf{decode})$ with this specific code with little loss in conceptual understanding.

2.4 Cryptography

We collect a variety of standard definitions of asymmetric cryptographic primitives that will form the core of our survey. Note that there are a few other cryptographic primitives that will appear sporadically[10] which we omit formal definitions of, as they are not conceptually important for the survey.

We first describe the syntax and correctness properties of Public-Key Encryption (PKE) and Key-Encapsulation Mechanisms (KEMs), before later describing their security properties. We will need a slightly non-standard notion of correctness of PKE for a variant of the Fujisaki-Okamoto transform [25] that we survey in sub-section 3.4.

[8] For example, "promise" versions of codes that are only error-correcting for e contained in the largest sphere inscribed within $C.\mathsf{domain}$, which are known as *bounded distance decoding* algorithms. Some codes we suggest to investigate in sub-section 6.1 only have efficient BDD algorithms, which are sufficient for the applications we suggest them for.

[9] The only setting one tends *not* to use this encoding is for some constructions of Fully Homomorphic Encryption, e.g. relatively advanced applications.

[10] In particular in sub-sections 3.4 and 5.1.

Definition 2 (Public-Key Encryption (PKE))**.** *Fix* $\mathbb{M}, \mathbb{PK}, \mathbb{SK}, \mathbb{CS}$ *as a message space, public-key space, secret-key space, and ciphertext space. A public-key encryption scheme (PKE) is a triple of algorithms:*

$$\mathsf{KGen} : \mathbb{N} \to \mathbb{PK} \times \mathbb{SK}, \qquad \mathsf{Enc} : \mathbb{PK} \times \mathbb{M} \to \mathbb{CS}, \qquad \mathsf{Dec} : \mathbb{SK} \times \mathbb{CS} \to \mathbb{M}$$

where $\mathsf{KGen}, \mathsf{Enc}$ *are randomized. We call a PKE scheme* δ*-correct for* $\delta \in [0, 1]$ *if:*

$$\mathbb{E}_{(\mathsf{pk},\mathsf{sk})}[\max_{m \in \mathbb{M}} \Pr[\mathsf{Dec}_{\mathsf{sk}}(c) \neq m \mid c \leftarrow_{\$} \mathsf{Enc}_{\mathsf{pk}}(m)]] \leq \delta$$

Definition 3 (Key Encapsulation Mechanism)**.** *Fix* $\mathbb{DK}, \mathbb{PK}, \mathbb{SK}, \mathbb{CS}$ *as a derived-key space, public-key space, secret-key space, and ciphertext space. A key-encapsulation mechanism (KEM) is a triple of algorithms:*

$$\mathsf{KGen} : \mathbb{N} \to \mathbb{PK} \times \mathbb{SK}, \qquad \mathsf{Encaps} : \mathbb{PK} \to \mathbb{DK} \times \mathbb{CS}, \qquad \mathsf{Decaps} : \mathbb{SK} \times \mathbb{CS} \to \mathbb{DK}$$

where $\mathsf{KGen}, \mathsf{Encaps}$ *are randomized. The KEM* $(\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ *is* ***perfectly correct*** *if* $\forall (\mathsf{pk}, \mathsf{sk}) \leftarrow_{\$} \mathsf{KGen}(1^n)$*,* $\forall (dk, c) \leftarrow_{\$} \mathsf{Encaps}(\mathsf{pk})$

$$dk = \mathsf{Decaps}(\mathsf{sk}, c)$$

Note that one can weaken the correctness notion for a KEM as well, for example to only produce shared keys with very high probability. KEMs that produce uniform keys are desirable (and some constructions we survey explicitly target this), but not required — standard techniques can convert non-uniform shared keys to uniform with a mild amount of post-processing.

Computational notions of security are defined through what are known as "security games". We detail some of the relevant ones for our purposes in figure 2.4, although delay further discussion until subsection 3.4, where we discuss the techniques that practitioners use to convert the passively-secure schemes (whether KEMs or PKE) that they construct into actively-secure KEMs.

```
IND-CPA for Encryption
$\mathsf{pk}, \mathsf{sk} \leftarrow_{\$} \mathsf{KGen}(1^n)$
$b \leftarrow_{\$} \{0, 1\}$
$(m_0^*, m_1^*, st) \leftarrow_{\$} \mathcal{A}_1(\mathsf{pk})$
$c^* \leftarrow_{\$} \mathsf{Enc}_{\mathsf{pk}}(m_b^*)$
$b' \leftarrow_{\$} \mathcal{A}_2(\mathsf{pk}, c^*, st)$
return $b' == b$
```

```
IND-CPA for KEM
$\mathsf{pk}, \mathsf{sk} \leftarrow_{\$} \mathsf{KGen}(1^n)$
$b \leftarrow_{\$} \{0, 1\}$
$(k^*, c^*) \leftarrow_{\$} \mathsf{Encaps}(\mathsf{pk})$
if $b == 1$
    $k^* \leftarrow_{\$} \mathbb{DK}$
$b' \leftarrow_{\$} \mathcal{A}(\mathsf{pk}, (c^*, k^*))$
return $b' == b$
```

Figure 2: The standard passive security notions for PKE and KEMs. Active (adaptive) security notions additionally allow the adversaries access to oracles $\mathsf{Dec}_{\mathsf{sk}}(\cdot)$ and $\mathsf{Decaps}(\mathsf{sk}, \cdot)$ on inputs that are not equal to c^*.

2.5 Learning with Errors

We will present the construction of lattice-based primitives in terms of the *LWE function*. Recall that $\mathbb{A}$ is some finite ring, which the reader should intuitively set to be $\mathbb{Z}_q$, but in practice (for efficiency purposes) is of the form $\mathbb{Z}_q[x]/(x^{2^k}+1)$.

Definition 4. *Let* $m, \ell \in \mathbb{N}$. *Let* $\mathbf{A} \in \mathbb{A}^{m\times\ell}, \mathbf{m}, \mathbf{e}, \in \mathbb{A}^m, \mathbf{s} \in \mathbb{A}^\ell$. *Define*

$$\mathsf{LWE}_{\mathbf{s}}(\mathbf{m}; \mathbf{A}, \mathbf{e}) = \mathbf{m} + \mathbf{A}\mathbf{s} + \mathbf{e}$$

to be the LWE function.

The parameters $\mathbf{A}, \mathbf{s}, \mathbf{e}$ will be randomly generated. $\mathbf{A}$ will always be uniform over $\mathbb{A}^{m\times n}$, we refer to the distribution $\mathbf{s}$ is sampled from via $\mathsf{LWE.secret}$ (this will typically be some i.i.d. distribution over $\mathbb{A}^n$, such as uniform, or "Gaussian-like").

There are two main ways to generate $\mathbf{e}$:

1. *Random Errors*: $\mathbf{e}$ is sampled with i.i.d. coordinates from some distribution on $\mathbb{A}$
2. *Deterministic Errors*: For some linear code C with ambient space $\mathbb{A}^m$, $\mathbf{e}$ is set to be $\mathbf{e} = -(\mathbf{A}\mathbf{s} + \mathbf{m} \bmod C)$, i.e. one has:

$$\begin{aligned}\mathsf{LWE}_{\mathbf{s}}(\mathbf{m}; \mathbf{A}, \mathbf{e}) &= \mathbf{A}\mathbf{s} + \mathbf{m} + \mathbf{e}\\ &= \mathbf{A}\mathbf{s} + \mathbf{m} - (\mathbf{A}\mathbf{s} + \mathbf{m} \bmod C)\\ &\overset{1}{=} \lfloor \mathbf{A}\mathbf{s} + \mathbf{m} \rceil_C\end{aligned}$$

   where 1 follows from the definition of the function $x \mapsto x \bmod C$.

By far the most common setting of LWE with deterministic errors is when $\mathbb{A} \cong \mathbb{Z}_{2^\ell}$ for some $\ell \in \mathbb{N}$, and for some $t \in \mathbb{N}$ such that $t < \ell$ the code C is given by

$$\mathsf{encode}(m) = (2^\ell/2^t)m, \qquad \mathsf{decode}(x) = \lfloor (2^t/2^\ell)x \rceil,$$

e.g. is a scaling code, where one scales by $(q/p) = 2^\ell/2^t$. For such C, it is straightforward to verify that $\lfloor x \rceil$ zeros out the t lowest-order bits of $x \in \mathbb{Z}_{2^\ell} \cong \{0,1\}^\ell$, so "rounds" the (noiseless) LWE sample $\mathbf{A}\mathbf{s}$. This motivates the common name for the hardness assumption, which is "Learning With Rounding" (LWR).

We will accomodate both[11] techniques by stating that $\mathbf{e}$ is sampled from a family of distributions that may depend on $(\mathbf{m}; \mathbf{A}, \mathbf{s})$, i.e. $\mathbf{e} \leftarrow_{\$} \mathsf{LWE.error}(\mathbf{m}; \mathbf{A}, \mathbf{s})$. Note that $\mathsf{LWE.secret}$ is instead "just" a fixed distribution. We furthermore view the parameter $\mathbf{e}$ in a call to the $\mathsf{LWE}_{\mathbf{s}}$ function as being optional — if it is omitted, one first samples $\mathbf{e} \leftarrow_{\$} \mathsf{LWE.error}(\mathbf{m}; \mathbf{A}, \mathbf{s})$, before determinstically computing $\mathsf{LWE}_{\mathbf{s}}(\mathbf{m}; \mathbf{A}, \mathbf{e})$. In this notation, we can state the (decisional) LWE problem as follows.

[11] Note that this syntax additionally allows for errors that are "partially random and partially deterministic", which we discuss in sub-section 5.2.

Definition 5. *Let $m, \ell \in \mathbb{N}$. For $\mathbf{A} \leftarrow_{\$} \mathbb{A}^{\ell \times m}$ and $\mathbf{s} \leftarrow_{\$} \mathsf{LWE.secret}$, the decisional LWE problem is to distinguish the distributions:*

$$(\mathbf{A}, \mathsf{LWE}_{\mathbf{s}}(0; \mathbf{A})) \overset{?}{\approx}_c (\mathbf{A}, \mathbf{b})$$

Where $\mathbf{b} \leftarrow_{\$} \mathbb{A}^m$, and $\mathbf{s} \leftarrow_{\$} \mathsf{LWE.secret}$.

The difficulty of this problem heavily depends on the choices of $\mathbb{A}, m, \ell$, $\mathsf{LWE.secret}$, and $\mathsf{LWE.error}$. We will not attempt to survey the hardness of the LWE problem (although direct the interested reader to [3]), and will instead show that our constructions are secure conditioned on the underlying decisional LWE problem being hard.

Throughout, while we let n be the security parameter, we will often write $\mathbb{A}^m$ as the space arithmetic occurs within. Both $\mathbb{A}$ and m can implicitly depend on the security parameter. For simplicty, we do not make this explicit, and just mention that when $\mathbb{A} \cong \mathbb{Z}_q^\ell$ (as sets), the general rule of thumb[12] is that the quantity $m\ell \log_2 q$ controls the hardness of the problem. See [11] for some results in this direction.

2.6 "Noisy" Diffie-Hellman

All Diffie-Hellman type constructions rely on the following symmetry of group exponentiation.

Example 2. *Let G be a group, and let $g \in G$. Then for any $r, s \in \mathbb{Z}$:*

$$(g^r)^s = g^{rs} = (g^s)^r$$

This leads to a simple key exchange scheme where (relative to some generator $g \in G$) users exchange elements g^r, g^s, and can agree on g^{rs}. We next describe what one might call *approximate* key agreement. This is implicit in the literature on lattice-based KEMs, and has been made explicit elsewhere (for example in [41], where a definition that fixes the Hamming metric is given). We reproduce the definition of [41] below without this fixed choice of metric.

Definition 6. *A noisy key agreement protocol between two parties A and B is a tuple $\Pi = (\mathsf{Pgen}, \mathsf{AContr}, \mathsf{BContr}, \mathsf{AConv}, \mathsf{BConv})$ of five polynomial-time algorithms where the first three are probabilistic and the last two are deterministic. The algorithms are associated with the spaces $\mathsf{ParSp}, \mathsf{ContrSp}, \mathsf{StateSp}, \mathbb{M}$, and have type signatures as follows:*

- $\mathsf{Pgen} : \mathbb{N} \to \mathsf{ParSp}$
- $\mathsf{AContr}, \mathsf{BContr} : \mathsf{ParSp} \to \mathsf{StateSp} \times \mathsf{ContrSp}$
- $\mathsf{AConv}, \mathsf{BConv} : \mathsf{StateSp} \times \mathsf{ContrSp} \to \mathbb{M}$

[12] There are many caveats here, for example this is false if the error distribution is trivial.

The algorithms are such that, with respect to tolerable noise subset $D \subseteq \mathbb{M}$ and failure probability $\epsilon \in [0,1]$, one has that

$$\Pr[S_A - S_B \in D] \geq 1 - \epsilon$$

Where the probability is over

- $\mathsf{pp} \leftarrow_{\$} \mathsf{Init}(1^n)$
- $\mathsf{state}_A, \sigma_A \leftarrow_{\$} \mathsf{AContr}(\mathsf{pp})$
- $\mathsf{state}_B, \sigma_B \leftarrow_{\$} \mathsf{BContr}(\mathsf{pp})$
- $S_A \leftarrow \mathsf{AConv}(\mathsf{state}_A, \sigma_B)$
- $S_B \leftarrow \mathsf{BConv}(\mathsf{state}_B, \sigma_A)$

We briefly discuss Diffie-Hellman key exchange in this framework.

Example 3. *Let* $\mathsf{Pgen}(1^n)$ *return* (g, G, t) *where* G *is a group,* g *is an element of* G*, and* $|\langle g \rangle| = t$*. Define*

$$\begin{aligned} \mathsf{AContr}(g, G, t) &= (\mathsf{state}_A \leftarrow_{\$} \mathsf{Unif}([t]), g^{\mathsf{state}_A}), \\ \mathsf{BContr}(g, G, t) &= (\mathsf{state}_B \leftarrow_{\$} \mathsf{Unif}([t]), g^{\mathsf{state}_B}) \\ \mathsf{AConv}(\mathsf{state}_A, \sigma_B = g^{\mathsf{state}_B}) &= \sigma_B^{\mathsf{state}_A} = g^{\mathsf{state}_A \times \mathsf{state}_B}, \\ \mathsf{BConv}(\mathsf{state}_B, \sigma_A) &= \sigma_A^{\mathsf{state}_B} = g^{\mathsf{state}_A \times \mathsf{state}_B} \end{aligned}$$

One can see directly that this is a noisy key agreement protocol with tolerable noise subset $D = \{0\}$, where 0 is the identity of the group.

The work of [41] defines a range of hard problems for noisy key agreement with tolerable noise subset D, briefly summarized as

- **State Recovery Problem**: given the public parmeters pp and σ_A (resp. σ_B), recover state_A (resp. state_B), and
- **Noisy Key Search**: given the public parameters $\mathsf{pp}, \sigma_A, \sigma_B$, find S such that $S - S_A \in D$ and $S - S_B \in D$.

In the case of Diffie-Hellman, these problems reduce to the discrete logarithm problem, and the computational Diffie-Hellman assumption.

We next define the “core” noisy key agreement protocol underlying LWE-based key exchange. In what follows, we use the notation $\mathbf{a}^t$ to denote the transpose of $\mathbf{a}$ (and similarly for matrices $\mathbf{A}$).

One can obtain a relatively simple expression of the noise between the shares $S_A - S_B$ in the protocol of figure 2.6 as follows.

Lemma 1. *Fix* $\mathbb{A}, m$*,* $\mathsf{LWE.secret}$*, and* $\mathsf{LWE.error}$*. Then, the results* S_A, S_B *of the noisy key agreement protocol of figure 2.6 satisfy*

$$S_A - S_B = \mathbf{e}_B^t \mathsf{state}_A - \mathsf{state}_B^t \mathbf{e}_A$$

```
$\mathsf{Pgen}(1^n)$
    $\mathbf{A} \leftarrow_\$ \mathbb{A}^{m\times m}$
    return $\mathbf{A}$
```

```
$\mathsf{AContr}(\mathbf{A})$
    $\mathsf{state}_A \leftarrow_\$ \mathsf{LWE.secret}$
    $\sigma_A \leftarrow_\$ \mathsf{LWE}_{\mathsf{state}_A}(0; \mathsf{A})$
    return $(\mathsf{state}_A, \sigma_A)$
```

```
$\mathsf{BContr}(\mathbf{A})$
    $\mathsf{state}_B \leftarrow_\$ \mathsf{LWE.secret}$
    $\sigma_B \leftarrow_\$ \mathsf{LWE}_{\mathsf{state}_B}(0; \mathsf{A}^t)$
    return $(\mathsf{state}_A, \sigma_A)$
```

```
$\mathsf{AConv}(\mathsf{state}_A, \sigma_B)$
    return $\sigma_B^t \mathsf{state}_A$
```

```
$\mathsf{BConv}(\mathsf{state}_B, \sigma_A)$
    return $\mathsf{state}_B^t \sigma_A$
```

Figure 3: The noisy key agreement underlying LWE-based KEMs.

where

$$\begin{aligned}\mathsf{state}_A, \mathsf{state}_B \leftarrow_\$ \mathsf{LWE.secret},\\ \mathbf{e}_A \leftarrow_\$ \mathsf{LWE.error}(0; \mathbf{A}, \mathsf{state}_A),\\ \mathbf{e}_B \leftarrow_\$ \mathsf{LWE.error}(0; \mathbf{A}^t, \mathsf{state}_B).\end{aligned}$$

Proof. Recall that $\sigma_A = \mathbf{A}\mathsf{state}_A + \mathbf{e}_A$ for $\mathbf{e}_A \leftarrow_\$ \mathsf{LWE.error}(0; \mathbf{A}, \mathsf{state}_A)$, and similarly $\sigma_B = \mathbf{A}^t\mathsf{state}_B + \mathbf{e}_B$ for $\mathbf{e}_B \leftarrow_\$ \mathsf{LWE.error}(0; \mathbf{A}^t, \mathsf{state}_B)$. We therefore have that

$$\begin{aligned}S_A &= \sigma_B^t\mathsf{state}_A = (\mathbf{e}_B^t + \mathsf{state}_B^t\mathbf{A})\mathsf{state}_A,\\ S_B &= \mathsf{state}_B^t\sigma_A = \mathsf{state}_B^t(\mathbf{A}\mathsf{state}_A + \mathbf{e}_A)\end{aligned}$$

so their difference satisfies

$$S_A - S_B = \mathbf{e}_B^t\mathsf{state}_A - \mathsf{state}_B^t\mathbf{e}_A$$

□

One still needs to produce some set D such that one can bound $\Pr[S_A - S_B \in D] \geq 1 - \epsilon$. This is typically done computationally — while one can compute theoretical bounds, they suffer from the inner products $\mathbf{e}_B^t\mathsf{state}_A$ falling outside of the standard class of "sub-Gaussian" random variables (which admit strong tail bounds known as *Chernoff bounds*) into a weaker class. Note that this isn't a practical issue — the computation of the pair (D, ϵ) can be done exactly via computing the explicit distribution of $S_A - S_B$, but it does mean that the standard theoretical bounds one may use are less tight than normal.

This does mean that there is litte practical value in (currently known) theoretical bounds on $S_A - S_B$ though, so to save space we will not reproduce them. It suffices to state that "reasonable" parameters to expect are something like $\epsilon \ll 2^{-100}$, and D of moderately large size — say for $\mathbb{A} = \mathbb{Z}_q$ of the form $[-q/d, q/d]$ for small d.

It is relatively simple to argue security of this construction relative to the problems defined in [41], as

- for the state recovery problems, one must recover state_A from $(\mathbf{A}, \mathbf{A}\mathsf{state}_A + \mathbf{e})$, e.g. solve the LWE search problem.

- For noisy key search, provided testing membership $S - S_A \in D$ is efficient[13], any adversary against the other hard problem can be used to solve decisional LWE in a rather straightforward reduction.

We do not formally treat these issues as we will not use the generic transformations of [41] later.

3 Constructing Lattice-based KEMs

We begin by formally describing archetypical examples of what have come to be known as "encryption-based" and "reconciliation-based" IND-CPA-secure KEMs, before describing the approach to convert them to IND-CCA2-secure KEMs. The general strategy will be to achieve IND-CCA2-security via appealing to the Fujisaki-Okamoto transform [23], which transforms an IND-CPA-secure PKE (and a few other primitives, discussed in sub-section 3.4) into an IND-CCA2-secure KEM. For this reason, the "encryption-based KEM" we describe will actually be an IND-CPA-secure PKE, which one can trivially convert into a KEM by encrypting a uniformly random secret key.

We explicitly mention this conversion for two reason, namely

- lattice-based KEMs are not perfectly correct, which required a re-analysis of the FO transform, and
- one large way that encryption and reconciliation-based KEMs differ is in how well they behave with respect to the FO transform.

In particular, reconciliation-based KEMs require constructing a PKE from them before applying the transform, e.g. passing through something like hybrid encryption, which leads to somewhat less practical IND-CCA2-secure constructions.

3.1 Encryption-based KEMs

We first describe a simple framework where one combines the LWE-based noisy key agreement of fig. 2.6 with a linear code with ambient space $\mathbb{A}^m$. This framework is initially due to [33], and is at the core of many of the most practical LWE-based KEMs, including all (LWE-based) NIST finalists. We delay exposition on the various optimizations that are used in practice until section 5.

One can relate to correctness of this scheme to the correctness of the noisy key agreement and the decoding region of the code C. We treat the case of $\mathsf{LWE.error}(\cdot;\cdot,\cdot)$ being a fixed distribution independent of its inputs as it is somewhat simpler to write down, but the general case is immediate from inspecting definition 2.

[13] There are some subtleties in this being true generally, namely that a natural choice of D is the domain of a linear code, so one can later correct the errors. Then efficient membership testing in D can be used to decode the code, which is not always efficient. For the cases we examine this will not matter.

```
$\mathsf{KGen}(1^n)$
    $\mathbf{A} \leftarrow_{\$} \mathsf{Pgen}(1^n)$
    $(\mathsf{state}_A, \sigma_A) \leftarrow_{\$} \mathsf{AContr}(\mathbf{A})$
    $\mathsf{pk} = (\mathbf{A}, \sigma_A)$
    $\mathsf{sk} = \mathsf{state}_A$
    return $\mathsf{sk}, \mathsf{pk}$
```

```
$\mathsf{Enc}_{\mathsf{pk}}(m)$
    $(\mathbf{A}, \sigma_A) := \mathsf{pk}$
    $(\mathsf{state}_B, \sigma_B) \leftarrow_{\$} \mathsf{BContr}(\mathbf{A}^t)$
    $v \leftarrow_{\$} \mathsf{LWE}_{\mathsf{state}_B}(\mathsf{encode}(m); \sigma_A^t)$
    return $\sigma_B, v$
```

```
$\mathsf{Dec}_{\mathsf{sk}}(\sigma_B, v)$
    return $\mathsf{decode}(v - \mathsf{sk}^t \sigma_B)$
```

Figure 4: The Lyubashevsky, Peikert, and Regev cryptosystem, parametrized by a linear code $C = (\mathsf{encode}, \mathsf{decode})$. Note that as $v \leftarrow_{\$} \mathsf{LWE}(\mathsf{encode}(m); \sigma_A^t)$ computes the LWE function with respect to $\sigma_A^t \in \mathbb{A}^{1 \times m}$, $v \in \mathbb{A}$ is a scalar, and that $\mathsf{state}_A, \mathsf{state}_B$ are elements of $\mathbb{A}^m$.

Theorem 2. *Let $n, m \in \mathbb{N}$. Let $C = (\mathsf{encode}, \mathsf{decode})$ be a linear code with message space $\mathbb{M}$ and ambient space $\mathbb{A}^m$. Let $\Pi = (\mathsf{Pgen}, \mathsf{AContr}, \mathsf{BContr}, \mathsf{AConv}, \mathsf{BConv})$ be noisy key agreement scheme with tolerable noise subset $D \subseteq \mathbb{M}$, $\mathsf{ContrSp} = \mathbb{A}^m$, $\mathsf{StateSp} = \mathrm{Supp}(\mathsf{LWE.secret})$, and failure probability ϵ. Then, the PKE as defined in fig. 3.1 δ-correct for*

$$\delta = \Pr_{e' \leftarrow_{\$} \mathsf{LWE.error}(0;0,0)}[e' + D \subseteq C.\mathsf{error}](1 - \epsilon),$$

and IND-CPA *secure under the decisional LWE assumption.*

Deriving the correctness expression is fairly mechanical in this setting. Security follows from noticing that the transcript is $(\mathbf{A}, \sigma_A, \sigma_B, v)$, and the non-uniform components (σ_A, σ_B, v) can be made uniform by appealing to the LWE assumption multiple times.

3.2 Reconciliation-based KEMs

An alternative approach to building an LWE-based KEM is to design a generic primitive that turns noisy key agreement to exact key agreement. We follow the presentation of [29], which formally defines a primitive called *Key Consensus* which serves the purposes that prior authors [20, 35] referred to as *error reconciliation*.

Definition 7 (Key Consensus)**.** *Let $\mathbb{M}$ be a metric space with metric d, and $\mathbb{K}$ be a key space, and $\mathbb{H}$ be a "Hint space". Let $\rho \in \mathbb{R}^+$. A Key Consensus (KC) protocol (of noise tolerance ρ) is a pair of algorithms:*

$$\mathsf{MakeHint} : \mathbb{M} \to \mathbb{H} \times \mathbb{K}, \qquad \mathsf{Rec} : \mathbb{M} \times \mathbb{H} \to \mathbb{K}$$

such that the following properties hold:

1. ***Correctness**: $\forall m_1, m_2$ such that $d(m_1, m_2) \leq \rho$, if $(h, k) = \mathsf{MakeHint}(m_1)$, then $\mathsf{Rec}(m_2, h) = k$.*

```
$\mathsf{KGen}(1^n)$
    $\mathbf{A} \leftarrow_\$ \mathsf{Pgen}(1^n)$
    $\mathsf{pk} \leftarrow_\$ (\mathbf{A}, \mathsf{LWE}_{\mathsf{sk}}(0; \mathbf{A}))$
    return $\mathsf{sk}, \mathsf{pk}$
```

```
$\mathsf{Encaps}(\mathsf{pk})$
    $(\mathbf{A}, \sigma_A) := \mathsf{pk}$
    $(\mathsf{state}_B, \sigma_B) \leftarrow_\$ \mathsf{BContr}(\mathbf{A}^t)$
    $v \leftarrow_\$ \mathsf{LWE}_{\mathsf{state}_{\mathbf{B}}}(0; \sigma_A^t)$
    $h, k \leftarrow_\$ \mathsf{MakeHint}(v)$
    return $k, (\sigma_B, h)$
```

```
$\mathsf{Decaps}(\mathsf{sk}, (\sigma_B, h))$
    return $\mathsf{Rec}(\mathsf{sk}^t \sigma_B, h)$
```

Figure 5: The Ding Reconcilliation-based KEM [20], where reconcilliation is done via a Key Consensus protocol.

2. ***Security:*** *if $m \leftarrow_\$ \mathbb{M}$, and $(h, k) \leftarrow_\$ \mathsf{MakeHint}(m)$, then h, k are independent of eachother, and k is uniformly distributed over $\mathbb{K}$.*

Given a KC protocol with high enough noise tolerance, it is straightforward to turn the aforementioned "noisy" Diffie-Hellman key exchange into a KEM.

Theorem 3. *Let $n \in \mathbb{N}$. Let $(\mathsf{MakeHint}, \mathsf{Rec})$ be a KC protocol with noise tolerance ρ for some $\rho > 0$. Let $\mathbb{B} \subseteq \mathbb{M}$ be the ball of radius ρ. Then if $\Pi = (\mathsf{Pgen}, \mathsf{AContr}, \mathsf{BContr}, \mathsf{AConv}, \mathsf{BConv})$ is a noisy key agreement protocol with a tolerable noise subset $D \supseteq \mathbb{B}$ and failure probability ϵ Then the KEM in fig. 3.2 is correct with probability $1 - \epsilon$, and is* IND-CPA*-secure under the decisional LWE assumption.*

Proof. Correctness follows directly from the definitions and the assumption that $\mathbb{B} \subseteq D$ (as the precondition for noisy key consensus is by design the same as the postcondition of noisy key agreement). Security again follows via appealing to the decisional LWE assumption a few times to show the transcript is computationally indistinguishable from uniformly random. □

3.3 Explicit Differences between the Frameworks

After describing the protocols in terms of noisy key agreement, it is clear there are close similarities. In both constructions, one party computes $v \leftarrow_\$ \mathsf{LWE}_{\mathsf{state}_B}(0; \sigma_A^t)$, and applies some mild post-processing to it:

$$v \mapsto \mathsf{MakeHint}(v) \qquad \text{or} \qquad v \mapsto v + \mathsf{encode}(k)$$

before returning the result. To aid understanding, we discuss two explicit examples of schemes in these frameworks. For encryption, we remind the reader of the scaling code of ex. 1 for $p \mid q$, where:

$$\mathsf{encode}(m) = (q/p)m, \qquad \mathsf{decode}(x) = \lfloor (p/q)x \rceil$$

For key consensus, we present a construction from [29].

Example 4 ("Simple" KC)**.** *Let* $q = 2^{q'}, p = 2^{p'}$ *for* $q', p' \in \mathbb{N}$*, and let* $C = (\mathsf{encode}, \mathsf{decode})$ *be the scaling code of parameters* p, q*. Define:*

$$\mathsf{MakeHint}(\mathbf{v}) = (\mathbf{v} \bmod C,\ \mathsf{decode}(\mathbf{v})), \qquad \mathsf{Rec}(\mathbf{v}', h) = \mathsf{decode}(\mathbf{v}' - h)$$

Then $(\mathsf{MakeHint}, \mathsf{Rec})$ *is a KC protocol with noise tolerance* ρ *for any* $\rho < \frac{q}{2p}$*, where distances are measured in the* ℓ_∞ *norm.*

If we look at the decryption and decapsulation functions of a reconcilliation-based KEM (using the simple KC of ex. 4) and a PKE (using the scaling code of ex. 1), we get that

$$\mathsf{Dec}_{\mathsf{sk}}(\sigma_B, v) = \mathsf{decode}(v - \mathsf{sk}^t\sigma_B), \qquad \mathsf{Decaps}(\mathsf{sk}, (\sigma_B, h)) = \mathsf{decode}(\mathsf{sk}^t\sigma_B - h),$$

i.e. they are identical (up to a sign that does not matter). Note that while the expressions are nearly identical, v, h are not. Recall that $v \in \mathbb{A}$. If for some $m, q \in \mathbb{N}$ one has that $\mathbb{A} \cong \mathbb{Z}_q^m$ as a set[14], then typically using reconcilliation, one can achieve $h \in \{0,1\}^m \subset \mathbb{Z}_q^m \cong \mathbb{A}$, i.e. one can achieve roughly a factor $\log_2 q \geq 10$ reduction in bandwidth for this particular element. Note that $\sigma_B \in \mathbb{A}^n$ is of the same size in both paradigms.

3.4 Transforming to IND-CCA2

The frameworks discussed so far yield IND-CPA-secure constructions, while many practical applications need the stronger IND-CCA notion of security. One typically achieves this via a generic "transformation", i.e. a construction that takes as input *any* IND-CPA-secure PKE and yields an IND-CCA-secure KEM.

We survey such a transformation, namely the recent analysis of the Fujisaki-Okamoto transform [22] in the quantum random oracle model for δ-correct PKE [25]. This work proceeds by decomposing the FO transform into two steps. The first step specifies the randomness that the PKE uses as a hash of the message to be encrypted, and later checks (via "re-encryption") that this was the randomness used in encryption.

Definition 8 (The T Transform)**.** *Let* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme, and let* G *be a hash function. Define the alternate encryption and decryption functions:*

1. $\mathsf{Enc}^1_{\mathsf{pk}}(m)$ *returns* $\mathsf{Enc}_{\mathsf{pk}}(m; G(m))$*, meaning encrypts* m *under* pk *using randomness* $G(m)$
2. $\mathsf{Dec}^1_{\mathsf{sk}}(c)$ *first decrypts* $m' = \mathsf{Dec}_{\mathsf{sk}}(c)$*. If this fails, or* $\mathsf{Enc}_{\mathsf{pk}}(m'; G(m')) \neq c$*, it returns* $\perp$*. Otherwise, it returns* m'*.*

Then the T *transform of* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *is* $(\mathsf{KGen}, \mathsf{Enc}^1, \mathsf{Dec}^1)$*.*

[14] This nearly always happens, although the arithmetic of $\mathbb{A}$ can be much more complex than that of the ring $\mathbb{Z}_q^m$.

The second step has a number of possible transformations — we cover the most common one among NIST candidates, namely the $U^{\not\perp}$ transform, which transforms the T transform[15] of a PKE $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$, into a KEM.

Definition 9 (The $U^{\not\perp}$ Transform)**.** *Let* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme, and let* $(\mathsf{KGen}, \mathsf{Enc}^1, \mathsf{Dec}^1)$ *be its T transform. Let H be a hash function. Define*

1. $\mathsf{Encaps}(\mathsf{pk})$*: First sample $k \leftarrow_\$ \mathbb{K}$ for a key space $\mathbb{K}$, compute $c \leftarrow \mathsf{Enc}^1_{\mathsf{pk}}(k)$, $K \leftarrow H(c, k)$, and return (c, K).*

2. $\mathsf{Decaps}_{\mathsf{sk}}(c)$ *If $\mathsf{Dec}^1_{\mathsf{sk}}(c) \neq \perp$, return $H(c, k)$, otherwise return $H(c, s)$, where s is a random seed contained in s.*

Then the $U^{\not\perp}$ transform of $(\mathsf{KGen}, \mathsf{Enc}^1, \mathsf{Dec}^1)$ *is said to be the KEM* $(\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$*.*

Combined, [25] shows that the $U^{\not\perp} \circ T$ transform takes an IND-CPA-secure encryption scheme, and transforms it to an IND-CCA-secure KEM. The reductions are fairly tight in the classical random oracle model, but are non-tight in the quantum random oracle model. There have been follow-up works improving the tightness gap [27, 26], but it is not closed yet.

As mentioned previously, a significant limitation of the FO transform is that it applies to PKE, and not KEMs, so one has to pass through hybrid encryption to apply it to a KEM. It would be quite interesting if techniques were developed of similar practical efficiency to the FO transform that convert IND-CPA-secure KEMs to IND-CCA-secure KEMs.

One significant limitation of this technique is that if one starts with an IND-CPA-secure KEM, one has to first use it to build a PKE before applying the transform. This leads to reconciliation-based constructions being less competetive[16] for building IND-CCA-secure KEMs. We discuss in sub-section 6.2 a potential avenue to side-step this difficulty.

4 Lattice-based KEMs from a Theory perspective

We next survey a few theoretical papers of interest to the construction of lattice-based KEMs.

[15] The PKE that the $U^{\not\perp}$ transform is applied to need not be precisely the T transform of an IND-CPA-secure scheme, but we make this simplification to avoid mentioning yet another security notion we do not have space to formally discuss.

[16] No NIST round 3 candidates are reconcilliation-based. There may be alternate (patent related) reasons for this. A fairly influential early "practical" scheme, NewHope [5], was reconcilliation-based, although the later NIST submission [4] was changed to be encryption-based. There additionally exist other reconcilliation-based submissions which did not reach the third round — in particular HILA5 [39], which was merged with Round2 [7] and advanced to the second round of the competition as Round5 [8].

4.1 Cryptographic Primitives from Linear Algebra

The framework of [9] that has inspired us to modularize our discussion of arithmetic and "short" elements is somewhat more specific than we have presented so far. Specifically, they consider the class of rings of the form

$$\mathbb{Z}[x]/((f(x), g(x))$$

where $g(x)$ is typically degree 1 or 2. This is somewhat different than the "typical" ring one works with in lattice-based cryptography

$$\mathbb{Z}[x]/(f(x), q) \cong \mathbb{Z}_q[x]/(f(x))$$

or in LPN-based cryptography

$$\mathbb{Z}[x]/(f(x), 2) \cong \mathbb{F}_2[x]/(f(x)).$$

The idea is that one can set $g(x) = q$ to capture both of these prior cases. One can additionally set $g(x) = x - b$. Reducing a polynomial $\sum_i a_i x^i \in \mathbb{Z}[x]$ to this setting yields the polynomial $\sum_i a_i b^i \bmod f(b)$, e.g. one performs modular (due to the $\bmod f(b)$) "big integer" (due to the $\sum_i a_i b^i$) arithmetic. The initial intention of this "big integer" variant of lattice-based cryptography is to reuse RSA coprocessors in embedded systems.

4.2 Wyner-Ziv Reconciliation for Key Exchange based on LWE

This paper [40] builds a framework for the construction of reconciliation-based KEMs. It (implicitly) does this by showing that, given linear $C_i = (\mathsf{encode}_i, \mathsf{decode}_i)$ for $i \in \{0, 1\}$ with message spaces $\mathbb{M}_i$ that are *nested*, meaning that:

$$\mathsf{encode}_0(\mathbb{M}_0) \subseteq \mathsf{encode}_1(\mathbb{M}_1), \qquad C_1.\mathsf{domain} \subseteq C_0.\mathsf{domain},$$

one can build a Key Consensus protocol from both of the codes.

Definition 10. *For $i \in \{0, 1\}$ let $C_0 \subseteq C_1$ be nested codes with message spaces $\mathbb{M}_i$ and common ambient space $\mathbb{X}$. Define*

$$\begin{aligned} \mathsf{MakeHint}(\mathbf{v}) &= (h \leftarrow \mathsf{decode}_1(\mathbf{v}) \bmod C_0, \mathsf{decode}_0(\mathbf{v} - h)) \\ \mathsf{Rec}(\mathbf{v}', h) &= \mathsf{decode}_0(\mathbf{v}' - h) \end{aligned}$$

to be the KC protocol defined by the nested linear codes $C_0 \subseteq C_1$.

Note that if one sets $\mathsf{decode}_1(\mathbf{v}) = \mathbf{v}$ to be the trivial code, one *nearly* gets back the Key Consensus protocol of ex. 4. The paper shows that with this construction, the nested condition immediately implies perfect security of the KC protocol. The story for correctness is somewhat worse — the work only examines a single pair of nested codes $C_0 \subseteq C_1$, moreover they choose C_1 to be trivial. Expanding the analysis of this protocol could be a quite interesting research direction.

4.3 Limits on KEMs built from RLWE

The final framework we survey [24] examines the class of possible reconcilliation functions $\mathsf{Rec}(\cdot)$. It is *a priori* possible that a reconciliation function could not require a hint. As generating the hint is the only[17] "interactive" step of reconcilliation-based KEM, not requiring a hint would lead to non-interactive key exchange from LWE, which would be quite interesting. Unfortunately, [24] shows that NIKE from LWE[18] is unlikely to exist, at least when the LWE moduli $q = n^{O(1)}$. Moduli this small are known as *polynomial modulus* in lattice-based cryptography, and are desirable for both security and efficiency.

The paper considers reconciliation functions which can take as input some subset of

- the other party's LWE share σ_B,
- the public matrix $\mathbf{A}$,
- your own state state_A,
- your own LWE randomness $\mathbf{e}_A \leftarrow_{\$} \mathsf{LWE.error}(0; \mathbf{A}, \mathsf{state}_A)$

In the most general setting, they notice that any reconciliation function must be unpredictable, even given the public parts of the transcript, so must *also* be a form of weak pseudorandom function. As only recently [30] have there been direct constructions of these from polynomial modulus, any such construction would be interesting in its own right. In other restricted settings they show impossibility — either information-theoretic or computational under the hardness of LWE, but due to the particular restrictions these results should be seen as "no-go" theorems for designers of protocols, rather than "hard" impossibility results for the construction of NIKE via LWE[19].

5 Lattice-based KEMs from a Practical perspective

We next discuss several optimizations that theoretical frameworks (including the ones described in section 3) tend to omit. Several of these are theoretically mundane, but most lead to quite interesting theoretical questions investigating the validity of the optimizations. We use Kyber, Saber, and LAC as a source of the practical optimizations we discuss.

[17] If one first fixes $\mathbf{A} \in \mathbb{O}_{k,q}^{n\times n}$ as a public parameter, similar to how "Diffie-Hellman groups" have been standardized in the past.

[18] We include the caveat that their result only discusses plain LWE and ring LWE, and not module LWE, or any LWR variants.

[19] Such results do not currently exist, as it is known that indistinguishability obfuscation implies NIKE, and there are (currently unbroken) candidate iO constructions from LWE[12].

5.1 Compression of Randomness

We start with the most mundane observation — one can generate the values $\mathbf{A}, \mathbf{s}$ pseudorandomly rather than randomly, for example using an Extendible Output Function to compute $\mathbf{s}$ from a short seed. While (unlike the next section) we do not see this as motivation to study new problems in the hardness of lattice-based cryptography, this optimization is still quite important for theoretical analysis to give accurate bandwidth predictions — $\mathbf{A} \in \mathbb{A}$ is often $\approx 1000 \log_2 q$ bits large in practice, and $\log_2 q \geq 10$. If one replaces this with a 128-bit seed, the initial estimate is off by roughly 2 orders of magnitude.

5.2 Compression of LWE Shares

A more theoretically interesting optimization is the compression of LWE shares σ_A, σ_B from the noisy key agreement. One can interpret the reconciliation-based framework described in subsection 4.2 as suggesting doing some form of this. This leads to a natural question — is there a notion of compression of ciphertexts in encryption-based KEMs? Briefly examining Kyber and Saber's protocols show the answer is yes — often the lower-order (noisy) bits of ciphertexts are zeroed out, which we mentioned in sub-section 2.5 is equivalent to (lossily) compressing the ciphertexts with a particularly simple linear code. This leads to few natural questions:

1. Can one use other (more effective) codes in this construction?
2. Does the induced compression error help security, and if so by how much?
3. Does the particular form of compression impact security, or solely the size of the induced errors?

In sub-section 6.2 we survey a recent paper [13] which can be interpreted as affirmatively answering the first question[20]. The Kyber authors [6] additionally give arguments (from a concrete perspective) for the second point, and other authors [30] have started investigating it theoretically, but in both settings it would be interesting to see analysis with respect to other codes C.

6 Other Interesting Constructions

Having used Kyber, Saber, and LAC as running examples of practical (encryption-based) KEMs throughout the survey, we briefly describe some interesting constructions which we find to be quite compelling, and evidence in favor of our choice to parametrize constructions with respect to the underlying linear code, rather than simply fix the scaling code everywhere.

[20] Using techniques that are quite similar to those of the initial reconciliation mechanism of [20].

6.1 Practically Evaluating Non-trivial Codes

A few years ago a paper [37] tried to apply the "Leech Lattice" to the construction of lattice-based PKE. This led to some mild bandwidth savings (roughly 17% for PKE, 10% for a KEM, but 18% if the KEM only has to transmit 240-bit keys[21]), but is limited in many ways — it was only defined over $\mathbb{Z}_q^n$, and changing it into a construction over $\mathbb{Z}_q[x]/(x^{2^k}+1)^n$ may be difficult[22].

This has motivated other authors to examine other non-trivial lattice codes, for example the E_8 lattice [28], which as we mentioned in the introduction is able to get smaller ciphertexts that transmit longer keys at roughly the same computational cost. While there is not a clear usecase for these longer keys, the fact that this highly performant (on the particular metric of key sizes) encoding exists leads to the natural question if other performant encodings exist on other (more useful) metrics, such as the size of the resulting ciphertexts.

This design strategy still fundamentally takes some finite (specifically 8)-dimensional code, and applies it coordinate-wise. It could be interesting to evaluate truly high-dimensional lattices codes for this purpose, i.e. codes built from a discrete logarithm lattice [21] or a Barnes Sloane lattice [34]. It is known that (measured via certain coding-theoretic parameters) these are asymptotically better than using the E_8 lattice coordinate-wise, so evaluating whether this is true for parameters of interest is an interesting open question.

6.2 Using Reconciliation Techniques for Encryption

Again, we previously mentioned in the introduction that recent work [13] on lattice-based Fully Homomorphic Encryption has suggested using the standard scaling code for encryption, but additionally defining a certain "hint vector" that one computes, allowing one to compress an element of $\mathbb{Z}_q^n$ to an element of $\mathbb{Z}_q \times \{0,1\}^n$. One can interpret this as a version of Ding's [20] initial reconciliation mechanism for encryption-based KEMs. As it natively works for encryption-based KEMs, one can apply the FO transform to it at to directly achieve IND-CCA security using reconciliation-like techniques.

The authors of [13] do not address this possibility. As we mention in the introduction, we believe this to be one of the most practical research directions originating from our survey, as it should be rather simple to evaluate the efficacy of the ciphertext compression of [13], and every (theoretical) metric we have checked it on so far is fairly compelling.

[21] This is because the Leech Lattice is a 24-dimensional object, and $24 \mid 240$ leads to a more efficient construction.

[22] 24 is not a power of two, so to maximize efficiency one would have to work with quotients of the form $\mathbb{Z}_q[x]/(f(x))$ for more general $f(x)$, which is a less popular setting among protocol designers.

7 Conclusion

While the NIST PQC competition is nearing its end, the design of lattice-based KEMs is headed in an exciting direction, especially with respect to techniques combining non-trivial encodings to achieve smaller bandwidth usage. Preliminary techniques already achieve ∼20% savings compared to NIST PQC candidates with no negative impacts on running time, and there are many avenues to explore yet. Moreover, adapting notation that allows these non-trivial encodings to be discussed may enable tighter interplay between various sub-fields of cryptography that are based on "cryptographic problems from linear algebra".

References

[1] Divesh Aggarwal et al. "A New Public-Key Cryptosystem via Mersenne Numbers". In: *Advances in Cryptology – CRYPTO 2018, Part III*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10993. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2018, pp. 459–482. DOI: 10.1007/978-3-319-96878-0_16.

[2] Miklós Ajtai. "Generating Hard Instances of Lattice Problems (Extended Abstract)". In: *28th Annual ACM Symposium on Theory of Computing*. Philadephia, PA, USA: ACM Press, May 1996, pp. 99–108. DOI: 10.1145/237814.237838.

[3] Martin R Albrecht, Rachel Player, and Sam Scott. "On the concrete hardness of learning with errors". In: *Journal of Mathematical Cryptology* 9.3 (2015), pp. 169–203.

[4] Erdem Alkim et al. *NewHope without reconciliation*. Cryptology ePrint Archive, Report 2016/1157. https://eprint.iacr.org/2016/1157. 2016.

[5] Erdem Alkim et al. "Post-quantum Key Exchange - A New Hope". In: *USENIX Security 2016: 25th USENIX Security Symposium*. Ed. by Thorsten Holz and Stefan Savage. Austin, TX, USA: USENIX Association, Aug. 2016, pp. 327–343.

[6] Roberto Avanzi et al. "Algorithm Specifications And Supporting Documentation". en. In: (), p. 43.

[7] Hayo Baan et al. *Round2: KEM and PKE based on GLWR*. Cryptology ePrint Archive, Report 2017/1183. https://eprint.iacr.org/2017/1183. 2017.

[8] Sauvik Bhattacharya et al. *Round5: Compact and Fast Post-Quantum Public-Key Encryption*. Cryptology ePrint Archive, Report 2018/725. https://eprint.iacr.org/2018/725. 2018.

[9] Carl Bootland et al. “A framework for cryptographic problems from linear algebra”. eng. In: *JOURNAL OF MATHEMATICAL CRYPTOLOGY* 14.1 (2020), pp. 202–217. ISSN: 1862-2976. URL: http://dx.doi.org/10.1515/jmc-2019-0032.

[10] Joppe W. Bos et al. “Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE”. In: *ACM CCS 2016: 23rd Conference on Computer and Communications Security*. Ed. by Edgar R. Weippl et al. Vienna, Austria: ACM Press, Oct. 2016, pp. 1006–1018. DOI: 10.1145/2976749.2978425.

[11] Zvika Brakerski and Nico Döttling. *Hardness of LWE on General Entropic Distributions*. Cryptology ePrint Archive, Report 2020/119. https://eprint.iacr.org/2020/119. 2020.

[12] Zvika Brakerski et al. *Candidate iO from Homomorphic Encryption Schemes*. Cryptology ePrint Archive, Report 2020/394. https://eprint.iacr.org/2020/394. 2020.

[13] Zvika Brakerski et al. “Leveraging Linear Decryption: Rate-1 Fully-Homomorphic Encryption and Time-Lock Puzzles”. In: *TCC 2019: 17th Theory of Cryptography Conference, Part II*. Ed. by Dennis Hofheinz and Alon Rosen. Vol. 11892. Lecture Notes in Computer Science. Nuremberg, Germany: Springer, Heidelberg, Germany, Dec. 2019, pp. 407–437. DOI: 10.1007/978-3-030-36033-7_16.

[14] Zvika Brakerski et al. *Worst-Case Hardness for LPN and Cryptographic Hashing via Code Smoothing*. Cryptology ePrint Archive, Report 2018/279. https://eprint.iacr.org/2018/279. 2018.

[15] Gu Chunsheng. *Integer Version of Ring-LWE and its Applications*. Cryptology ePrint Archive, Report 2017/641. https://eprint.iacr.org/2017/641. 2017.

[16] John Horton Conway and Neil James Alexander Sloane. *Sphere packings, lattices and groups*. Vol. 290. Springer Science & Business Media, 2013.

[17] Hovav Shacham and Alexandra Boldyreva, eds. *Advances in Cryptology – CRYPTO 2018, Part III*. Vol. 10993. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2018.

[18] Jan-Pieter D’Anvers et al. “Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM”. In: *AFRICACRYPT 18: 10th International Conference on Cryptology in Africa*. Ed. by Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi. Vol. 10831. Lecture Notes in Computer Science. Marrakesh, Morocco: Springer, Heidelberg, Germany, May 2018, pp. 282–305. DOI: 10.1007/978-3-319-89339-6_16.

[19] Thomas Debris-Alazard, Léo Ducas, and Wessel P.J. van Woerden. *An Algorithmic Reduction Theory for Binary Codes: LLL and more*. Cryptology ePrint Archive, Report 2020/869. https://eprint.iacr.org/2020/869. 2020.

[20] Jintai Ding. *New cryptographic constructions using generalized learning with errors problem*. Cryptology ePrint Archive, Report 2012/387. https://eprint.iacr.org/2012/387. 2012.

[21] Léo Ducas and Cécile Pierrot. "Polynomial time bounded distance decoding near Minkowski's bound in discrete logarithm lattices". In: *Designs, Codes and Cryptography* 87.8 (Aug. 2019), pp. 1737–1748. ISSN: 1573-7586. DOI: 10.1007/s10623-018-0573-3. URL: https://doi.org/10.1007/s10623-018-0573-3.

[22] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *Advances in Cryptology – CRYPTO'99*. Ed. by Michael J. Wiener. Vol. 1666. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 1999, pp. 537–554. DOI: 10.1007/3-540-48405-1_34.

[23] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *Journal of Cryptology* 26.1 (Jan. 2013), pp. 80–101. DOI: 10.1007/s00145-011-9114-1.

[24] Siyao Guo et al. *Limits on the Efficiency of (Ring) LWE based Non-Interactive Key Exchange*. Cryptology ePrint Archive, Report 2020/1555. https://eprint.iacr.org/2020/1555. 2020.

[25] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. "A Modular Analysis of the Fujisaki-Okamoto Transformation". In: *TCC 2017: 15th Theory of Cryptography Conference, Part I*. Ed. by Yael Kalai and Leonid Reyzin. Vol. 10677. Lecture Notes in Computer Science. Baltimore, MD, USA: Springer, Heidelberg, Germany, Nov. 2017, pp. 341–371. DOI: 10.1007/978-3-319-70500-2_12.

[26] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. *Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model*. Cryptology ePrint Archive, Report 2019/134. https://eprint.iacr.org/2019/134. 2019.

[27] Haodong Jiang et al. "IND-CCA-Secure Key Encapsulation Mechanism in the Quantum Random Oracle Model, Revisited". In: *Advances in Cryptology – CRYPTO 2018, Part III*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10993. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2018, pp. 96–125. DOI: 10.1007/978-3-319-96878-0_4.

[28] Zhengzhong Jin and Yunlei Zhao. *AKCN-E8: Compact and Flexible KEM from Ideal Lattice*. Cryptology ePrint Archive, Report 2020/056. https://eprint.iacr.org/2020/056. 2020.

[29] Zhengzhong Jin and Yunlei Zhao. *Optimal Key Consensus in Presence of Noise*. Cryptology ePrint Archive, Report 2017/1058. https://eprint.iacr.org/2017/1058. 2017.

[30] Sam Kim. “Key-Homomorphic Pseudorandom Functions from LWE with Small Modulus”. In: *Advances in Cryptology – EUROCRYPT 2020, Part II*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12106. Lecture Notes in Computer Science. Zagreb, Croatia: Springer, Heidelberg, Germany, May 2020, pp. 576–607. DOI: 10.1007/978-3-030-45724-2_20.

[31] Arjen K Lenstra, Hendrik Willem Lenstra, and László Lovász. “Factoring polynomials with rational coefficients”. In: *Mathematische annalen* 261.ARTICLE (1982), pp. 515–534.

[32] Xianhui Lu et al. *LAC: Practical Ring-LWE Based Public-Key Encryption with Byte-Level Modulus*. Cryptology ePrint Archive, Report 2018/1009. https://eprint.iacr.org/2018/1009. 2018.

[33] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. “A Toolkit for Ring-LWE Cryptography”. In: *Advances in Cryptology – EUROCRYPT 2013*. Ed. by Thomas Johansson and Phong Q. Nguyen. Vol. 7881. Lecture Notes in Computer Science. Athens, Greece: Springer, Heidelberg, Germany, May 2013, pp. 35–54. DOI: 10.1007/978-3-642-38348-9_3.

[34] Ethan Mook and Chris Peikert. “Lattice (List) Decoding Near Minkowski’s Inequality”. In: *arXiv preprint arXiv:2010.04809* (2020).

[35] Chris Peikert. “Lattice Cryptography for the Internet”. In: *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*. Ed. by Michele Mosca. Waterloo, Ontario, Canada: Springer, Heidelberg, Germany, Oct. 2014, pp. 197–219. DOI: 10.1007/978-3-319-11659-4_12.

[36] Maxime Plançon and Thomas Prest. “Exact Lattice Sampling from Non-Gaussian Distributions”. In: *Public-Key Cryptography – PKC 2021*. Ed. by Juan A. Garay. Cham: Springer International Publishing, 2021, pp. 573–595. ISBN: 978-3-030-75245-3.

[37] Alex van Poppelen. *Cryptographic decoding of the Leech lattice*. Cryptology ePrint Archive, Report 2016/1050. https://eprint.iacr.org/2016/1050. 2016.

[38] Oded Regev. “On lattices, learning with errors, random linear codes, and cryptography”. In: *37th Annual ACM Symposium on Theory of Computing*. Ed. by Harold N. Gabow and Ronald Fagin. Baltimore, MA, USA: ACM Press, May 2005, pp. 84–93. DOI: 10.1145/1060590.1060603.

[39] Markku-Juhani O. Saarinen. “HILA5: On Reliability, Reconciliation, and Error Correction for Ring-LWE Encryption”. In: *SAC 2017: 24th Annual International Workshop on Selected Areas in Cryptography*. Ed. by Carlisle Adams and Jan Camenisch. Vol. 10719. Lecture Notes in Computer Science. Ottawa, ON, Canada: Springer, Heidelberg, Germany, Aug. 2017, pp. 192–212. DOI: 10.1007/978-3-319-72565-9_10.

[40] Charbel Saliba, Laura Luzzi, and Cong Ling. *Wyner-Ziv reconciliation for key exchange based on Ring-LWE*. Cryptology ePrint Archive, Report 2020/076. https://eprint.iacr.org/2020/076. 2020.

[41] Alan Szepieniec, Reza Reyhanitabar, and Bart Preneel. *Key Encapsulation from Noisy Key Agreement in the Quantum Random Oracle Model.* Cryptology ePrint Archive, Report 2018/884. https://eprint.iacr.org/2018/884. 2018.

[42] Ram Zamir et al. *Lattice Coding for Signals and Networks: A Structured Coding Approach to Quantization, Modulation and Multiuser Information Theory*. Cambridge University Press, 2014. DOI: 10.1017/CBO9781139045520.