Formally Verified Correctness Bounds for Lattice-Based Cryptography

Manuel Barbosa
mbb@fc.up.pt
University of Porto (FCUP)
Porto, Portugal
INESC TEC, Porto, Portugal
Porto, Portugal

Matthias J. Kannwischer
matthias@kannwischer.eu
Chelpis Quantum Corp
Taipei, Taiwan

Thing-han Lim
potsrevenmil@gmail.com
Academia Sinica
Taipei, Taiwan

Peter Schwabe
peter@cryptojedi.org
MPI-SP
Bochum, Germany
Radboud University
Nijmegen, The Netherlands

Pierre-Yves Strub
pierre-yves.strub@pqshield.com
PQShield
Paris, France

Abstract

Decryption errors play a crucial role in the security of KEMs based on Fujisaki-Okamoto because the concrete security guarantees provided by this transformation directly depend on the probability of such an event being bounded by a small real number. In this paper we present an approach to formally verify the claims of statistical probabilistic bounds for incorrect decryption in lattice-based KEM constructions. Our main motivating example is the PKE encryption scheme underlying ML-KEM. We formalize the statistical event that is used in the literature to heuristically approximate ML-KEM decryption errors and confirm that the upper bounds given in the literature for this event are correct. We consider FrodoKEM as an additional example, to demonstrate the wider applicability of the approach and the verification of a correctness bound without heuristic approximations. We also discuss other (non-approximate) approaches to bounding the probability of ML-KEM decryption.

CCS Concepts

• **Security and privacy → Logic and verification**; **Cryptography**.

Keywords

Computer-Aided Cryptography, Formal Verification, EasyCrypt

1 Introduction

The transition to post-quantum cryptography (PQC) has seen an important development in 2024 with the publication of the first PQC standards FIPS-203 [31], FIPS-204 [32] and FIPS-205 [33] by NIST. The standardized algorithms are called, ML-KEM, ML-DSA and SLH-DSA, which match the Kyber, Dilithium and SPHINCS+ submissions with minor changes, respectively. These algorithms will see large-scale deployment in the near future in many practical applications as mitigation for the potential arrival of a quantum computer. Key Encapsulation Mechanisms (KEM), such as ML-KEM are arguably the most critical components in the PQC transition, as they protect against so-called *harvest now, decrypt later* attacks which allow an attacker to decrypt data exchanged today with a future quantum computer. For this reason, ML-KEM is already being deployed by software giants such as Google and AWS [5, 12], and the number of deployed implementations is expected to grow fast in the near future. Another competitor in the NIST PQC competition for KEMs is called FrodoKEM. Although not selected by NIST for standardization, FrodoKEM's conservative design—its security is based on the standard Learning With Errors (LWE) assumption, rather than the Module LWE (MLWE) assumption used by ML-KEM—has led to endorsement of entities such as the German Federal Office for Information Security (BSI) [23] and the French National Agency for the Security of Information Systems (ANSSI) [1] for adoption in the transition to PQC. Additionally, ISO/IEC has approved its standardization in the revision of ISO/IEC 18033-2 [24].

Widely deployed cryptographic (de facto) standards such as ML-KEM and FrodoKEM will be critical security components in the ITC infrastructure of the coming decades, and so it is crucial that their design is validated to the highest level of assurance. For ML-KEM, several recent works have looked at formally verifying both the design and efficient implementation of the standard. In particular, Almeida et al. [4] presented formally verified proofs of cryptographic security (IND-CCA) and correctness—the guarantee that decapsulation inverts encapsulation—in EasyCrypt. Alternative proofs of IND-CPA security and correctness were given by

Kreuzer [29] in Isabelle. However, in both of these works, there is one aspect of the security and correctness claims that support the ML-KEM design that is not formally verified: the concrete values for the probability of a failed decryption. Both works account for the probability of a failed decryption by defining a statistical event over the distribution of a complex noise expression, and then proving that bounding the probability of such an event yields an upper bound for decryption failures. However, neither work provides a means to compute or even upper-bound this concrete probability to a high-level of assurance. In this paper we address this gap. We begin by recalling the importance of decryption errors in post-quantum KEM security.

The importance of decryption errors. Unlike Diffie–Hellman and RSA-based constructions, which typically yield perfectly correct cryptographic constructions, lattice-based constructions often allow for a low probability of error in order to optimize the compromise between security and performance. One might think that a decryption error would represent only an inconvenience for practical applications, e.g., in that it would cause message transmission to sometimes fail. However, it is well known that, when freely exposed to an adversary, decryption errors can lead to devastating attacks on lattice-based constructions [7, 9, 13, 14, 20–22]. Put differently, lattice-based KEM constructions such as ML-KEM and FrodoKEM are supported by IND-CCA security proofs where the overall bound on an attacker's advantage in breaking the KEM must typically account for the probability that the attacker can cause a decryption error to occur. This means that, in order to have a concrete security bound for the construction, one must bound the probability of a decryption error.

Intuitively, it is easy to explain why this is the case. Both ML-KEM and FrodoKEM internally use the Fujisaki–Okamoto [25] transformation, where IND-CCA security is achieved by having the decapsulation algorithm check consistency of a recovered decryption result via re-encryption. Informally, decapsulation checks that $C = \mathrm{Enc}(pk, M; H(M))$, where $M = \mathrm{Dec}(sk, C)$ and $H(M)$ is used to derive all randomness required by encryption pseudo-randomly. If the check succeeds, then decapsulation proceeds; otherwise the ciphertext is rejected. Indeed, correct decryption and re-encryption is taken as evidence that C was honestly constructed by the adversary starting from M, rather than mauling another ciphertext from which it is trying to extract information. The soundness of this technique crucially depends on the adversary not being able to exploit decryption errors, which is why the probability of a correctness error appears in the security bound for the IND-CCA construction.

Bounding the probability of decryption errors. Among the algorithms considered for the last round of the NIST PQC competition, four of them were very close in structure: Kyber [35], Saber [15], FrodoKEM [30], and NTRU LPRime [8].[1] All of these schemes start from a lattice-based IND-CPA encryption scheme and then apply the Fujisaki–Okamoto transform outlined above. However, while NTRU LPRrime selects parameters avoiding decryption errors altogether, the other three proposals support the soundness of their designs and parameter choices by computing exact bounds for statistical events that permit setting upper bounds for the probability of a decryption failure of the IND-CPA scheme—which affects, not only the decryption failure probability of the IND-CCA scheme, but also the corresponding security bound.[2] Interestingly, the bounds for all three schemes were computed using somewhat similar Python scripts, which trace their origins back to the script used to bound the failure probability of NewHope [3].[3]

For FrodoKEM, the computation performed by this script can be described as follows. The IND-CPA scheme decryption procedure of FrodoKEM recovers $M' = C_2 - C_1 S$ where M', C_1 and C_2 are matrices of (binary) field elements and M' encodes a message in the most significant bits of its entries. Here, (C_1, C_2) are produced by the encryption procedure as $C_1 = S'A + E'$ and $C_2 = S'B + E'' + M$, where matrices A and $B = AS + E$ are fixed by the public encryption key, the S matrix is the secret key, and S', E, E' and E'' are noise matrices sampled from distributions with very small support—every finite field element produced by these distributions is an element close to 0 chosen from a small set of possibilities. A straightforward linear algebra argument shows that $\mathrm{Decode}(M') = \mathrm{Decode}(M)$ if the noise expression $E''' = S'E + E'' - E'S$ results in a matrix where *all* entries are field elements with a small norm, i.e., they are small enough that the entries in M and M' have the same most significant bits. The Python script brute-force computes the probability mass function of a coefficient in E''' and computes the tail probability of a value exceeding the correctness threshold. The overall correctness bound follows from arguing that all entries in E''', individually, have the same distribution, and computing a union bound. We note that these computations are performed using high-precision floating-point arithmetic and result in values of the order of 2^{-200}.

The cases of ML-KEM and Saber are slightly more intricate due to the use of rounding, but the principle is the same. Prior to this work, the correctness of the above simplification steps—which are crucial to allow an efficient computation of the error—and therefore computed bounds that support the ML-KEM standard, and the FrodoKEM and Saber proposals have not been subject to formal verification.

Our Contributions. Our main contribution is an EasyCrypt formalization that permits connecting the formal definition of a decryption error for a KEM construction to an efficiently computable specification of a statistical event that provably yields an upper-bound for this security-critical parameter. More in detail, our individual contributions are the following.

- We provide a framework to reason in EasyCrypt about distributions over a restricted class of matrix expressions, and proving that the relevant events related to decryption errors can be expressed as a union bound over events that can be checked for only one of the matrix entries. We extend this result to cases where matrix entries are expressions in a certain class of polynomial rings, in which the event is checked for only one of the polynomial coefficients. This framework reduces the problem of

[1] FrodoKEM and NTRU LPRrime were not finalists, but kept as an alternate candidates.

[2] The results in this paper focus on the IND-CPA public-key encryption scheme subcomponents of the above algorithms. This means that we can talk interchangeably about Kyber (round 3) and ML-KEM, since there is no difference in their IND-CPA subcomponents. To avoid confusion, and because we believe this is where the interest lies for practical applications, we will mostly refer to ML-KEM from this point onwards when we talk about our results.

[3] See https://github.com/newhopecrypto/newhope-usenix/blob/master/scripts/failure.py

bounding the probability of decryption errors to the problem of comparing the absolute value of a finite field element sampled from a distribution, to a fixed threshold.
- We propose an approach to connect EasyCrypt specifications of probability bounds as above to OCaml computations that are guaranteed by construction to provide a concrete upper bound. We then build on this feature to compute upper bounds for decryption failure probabilities for FrodoKEM and ML-KEM. The algorithm has a reasonable execution time, whenever the distributions have a simple description and small enough support. On a modest personal machine, the more costly computations we performed for FrodoKEM take a few hours to complete. Our results can be seen as a formally verified implementation of the Python scripts used to obtain the upper bounds presented in the NIST post-quantum submissions.
- We show that, for FrodoKEM, our EasyCrypt development permits connecting the formal definition of correctness to a fully concrete correctness bound, where all statistical terms can be computed: our correctness theorem relates the adversary's advantage in winning the correctness game for the KEM to a description of the computation required to determine the probability value, which is then carried out in OCaml. As a side contribution, we give a computer-verified security proof for the IND-CPA component of FrodoKEM that goes down to a variant of the standard LWE problem (rather than MLWE as in ML-KEM [10] or LWR as in Saber [27]). This proof is similar in structure to those given in [4, 27, 29] but, to the best of our knowledge, such a proof had not been previously verified. In particular, our proof includes a hybrid argument that reduces the LWE problem to the multi-instance LWE problem required for FrodoKEM.
- We revisit the formally verified correctness proofs for ML-KEM in [4, 29] and resolve one of the proof goals left for future work: formally verifying that the simplified (heuristic) computations for the correctness bounds given in the documentation that suported this algorithm in the NIST PQC competition are correct. This shows the generality of our method and extends the formal verification results for ML-KEM [4] to cover all the correctness claims that supported it in the NIST competition. We also provide a new (more conservative) bound for ML-KEM decryption errors that can be justified under the MLWE assumption, i.e., we prove that this bound is correct unless MLWE can be broken.[4]

Related Work. Two previous works presented formally verified proofs of security and correctness for ML-KEM [4, 29]. Although these works covered security and correctness guarantees, none of them addressed the problem of proving that the concrete bounds for decryption failures claimed for the construction hold. We are not aware of prior work formally verifying any of the FrodoKEM security and correctness claims.

The impact of decryption failures in lattice-based KEM security has been studied in the literature from two perspectives: a provable security perspective, and an attack perspective. In this work we are interested in the provable security perspective, i.e., how one can obtain a concrete (formally verified) bound for the decryption error probability that is relevant for the setting of parameters of lattice-based KEMs. Our results confirm that, for FrodoKEM this is easy to do, whereas for ML-KEM obtaining a formal proof comes at a cost of significantly overestimating the probability.[5]

Alternatively, Hövelmanns, Hülsing, and Majenz [26] observe that the notion of cryptographic correctness (i.e., absence of decryption failures) used in Fujisaki-Okamoto security proofs may be too strong, in that it requires the bound to hold against an adversary that learns the secret key. The authors propose an alternative (weaker) definition that removes this requirement, but fundamentally modifies the way in which decryption failures are estimated: one needs to bound the difference in probability of failure with respect to another key pair. We are not aware of concrete bounds computed for these definitions, but it is an interesting direction for future work to formally verify their correctness. In this work we therefore work with the more standard (stronger) notion of decryption failure probability and study how bounds can formally verified for ML-KEM and FrodoKEM.

A sequence of works[13, 14, 16, 18, 19] studies the potential of exploiting decryption failures in lattice-based schemes, and Saber and Kyber in particular, in both single and multi-target scenarios. These works also investigate how to obtain good estimates for decryption failure probabilities, and various estimation techniques are proposed to deal with correlations between rounding noise across coefficients. In particular, these works point out that assuming independence across coefficients may be overly optimistic. We work in the simpler setting of single-key attack models, and consider only the most basic technique for approximate probability estimation in ML-KEM, which consists of assuming that all rounding errors across coefficients are independent. This was the approach used in the Kyber submission to NIST. We leave it as an interesting direction for future work to formally verify the correctness of other approximate estimation techniques. The impact of decryption failures in other families of cryptographic constructions have also been studied, e.g. in [36] for code-based cryptography, but these analyses are so far out of reach of our formal framework.

Structure of this paper. In Section 2 we provide some necessary background on ML-KEM, FrodoKEM, and EasyCrypt. Then in Section 3 and in Section 4 we describe the proofs that were formally verified in EasyCrypt. Finally, in Section 5 we discuss our approach to computing upper bounds in a formally verified way, and present our results for ML-KEM and FrodoKEM.

Access to development. The EasyCrypt and OCaml code described in this paper are submitted as supplementary material.

2 Preliminaries

We now briefly discuss the mechanized reasoning tools we use for our proofs and give an overview of the IND-CPA encryption schemes that underlie the FrodoKEM and ML-KEM constructions, which is all that we need to present our work on formally verifying the correctness bounds for both schemes. The cryptographic definitions used are standard and we try to keep as our presentation

[4]Proving the claim that the heuristic bound, which is computed over a simplified distribution, applies to ML-KEM is an open problem. Our new bound provably applies, but it is significantly larger than the heuristic one.

[5]We do not exclude that a better provably secure bound can be established using different techniques, but we leave this as an open problem.

Game COR:
1: $(pk, sk) \hookleftarrow_{\$} \mathsf{Gen}^{O}()$
2: $m \hookleftarrow_{\$} \mathcal{A}^{O}(pk, sk)$
3: $c \hookleftarrow_{\$} \mathsf{Enc}^{O}(pk, m)$
4: **return** $(m \neq \mathsf{Dec}^{O}(sk, c))$

Game IND-CPA:
1: $(pk, sk) \hookleftarrow_{\$} \mathsf{Gen}^{O}()$
2: $(m_0, m_1, st) \hookleftarrow_{\$} \mathcal{A}_1^{O}(pk)$
3: $b \hookleftarrow_{\$} \{0,1\}$
4: $c^* \hookleftarrow_{\$} \mathsf{Enc}^{O}(pk, m_b)$
5: $b' \hookleftarrow_{\$} \mathcal{A}_2^{O}(c^*, st)$
6: **return** $b' = b$

Figure 1: Correctness and Security of a PKE in the Random Oracle Model.

Algorithm 1 K-PKE.Gen(): key generation

Ensure: Secret key $sk \in \mathcal{R}_q^k$ and public key $pk \in \hat{\mathcal{R}}_q^{\,k} \times \{0,1\}^{256}$
1: $d \hookleftarrow_{\$} \{0,1\}^{256}$
2: $(\rho, \sigma) \leftarrow \mathsf{G}(d)$
3: $\hat{\mathbf{A}} \leftarrow \mathsf{Parse}(\mathsf{XOF}(\rho))$
4: $\mathbf{s}, \mathbf{e} \leftarrow \mathsf{CBD}_{\eta_1}(\mathsf{PRF}(\sigma))$ ▷ $\mathbf{s}, \mathbf{e} \in \mathcal{R}_q^k$
5: $\hat{\mathbf{s}} \leftarrow \mathsf{NTT}(\mathbf{s})$
6: $\hat{\mathbf{e}} \leftarrow \mathsf{NTT}(\mathbf{e})$
7: $\hat{\mathbf{t}} \leftarrow \hat{\mathbf{A}}\hat{\mathbf{s}} + \hat{\mathbf{e}}$
8: **return** $sk = \hat{\mathbf{s}}$ and $pk = (\hat{\mathbf{t}}, \rho)$

Algorithm 2 K-PKE.Enc(pk, m): encryption

Require: Public key $pk = (\hat{\mathbf{t}}, \rho) \in \mathcal{R}_q^k \times \{0,1\}^{256}$, message $m \in \{0,1\}^{256}$
Ensure: Ciphertext $c \in \mathcal{R}_{d_u}^k \times \mathcal{R}_{d_v}$
1: $r \hookleftarrow_{\$} \{0,1\}^{256}$
2: $\hat{\mathbf{A}} \leftarrow \mathsf{Parse}(\mathsf{XOF}(\rho))$
3: $\mathbf{r} \leftarrow \mathsf{CBD}_{\eta_1}(\mathsf{PRF}(r))$ ▷ $\mathbf{r} \in \mathcal{R}_q^k$
4: $\mathbf{e}_1, e_2 \leftarrow \mathsf{CBD}_{\eta_2}(\mathsf{PRF}(r))$ ▷ $\mathbf{e}_1 \in \mathcal{R}_q^k, e_2 \in \mathcal{R}_q$
5: $\hat{\mathbf{r}} \leftarrow \mathsf{NTT}(\mathbf{r})$
6: $\mathbf{u} \leftarrow \mathsf{NTT}^{-1}(\hat{\mathbf{A}}^T\hat{\mathbf{r}}) + \mathbf{e}_1$
7: $v \leftarrow \mathsf{NTT}^{-1}(\hat{\mathbf{t}}^T\hat{\mathbf{r}}) + e_2 + \mathsf{ToPoly}(m)$
8: $\mathbf{c}_1 \leftarrow \mathsf{Compress}_q(\mathbf{u}, d_u)$
9: $c_2 \leftarrow \mathsf{Compress}_q(v, d_v)$
10: **return** $c = (\mathbf{c}_1, c_2)$

Algorithm 3 K-PKE.Dec(sk, c): decryption

Require: Secret key $sk = \hat{\mathbf{s}} \in \mathcal{R}_q^k$ and ciphertext $c = (\mathbf{c}_1, c_2) \in \mathcal{R}_{d_u}^k \times \mathcal{R}_{d_v}$
Ensure: Message $m \in \{0,1\}^{256}$
1: $\tilde{\mathbf{u}} \leftarrow \mathsf{Decompress}_q(\mathbf{c}_1, d_u)$
2: $\tilde{v} \leftarrow \mathsf{Decompress}_q(c_2, d_v)$
3: $m \leftarrow \mathsf{ToMsg}(\tilde{v} - \mathsf{NTT}^{-1}(\hat{\mathbf{s}}^T\mathsf{NTT}(\tilde{\mathbf{u}})))$
4: **return** m

of the constructions close to the specifications of the algorithms found in the literature [30, 35].

2.1 Public-Key Encryption

Syntax. A public-key encryption scheme consists of three algorithms PKE = (Gen, Enc, Dec) and a finite message space $\mathcal{M}$. The key generation algorithm Gen outputs a key pair (pk, sk). The encryption algorithm Enc, on input pk and a message $m \in \mathcal{M}$, outputs a ciphertext $c \hookleftarrow_{\$} \mathsf{Enc}(pk, m)$. The decryption algorithm Dec, on input sk and a ciphertext c, outputs either a message $m \leftarrow \mathsf{Dec}(sk, c) \in \mathcal{M}$ or a special symbol $\perp \notin \mathcal{M}$.

Correctness. Correctness of a PKE is defined as in Figure 1 (left). We give the definition in the Random Oracle Model, as this is what we are going to use. Note that the adversary gets the secret key as an input. We say a PKE is δ-correct if, for all (possibly computationally unbounded) adversaries $\mathcal{A}$ placing at most q queries to the random oracle, we have that $\Pr[\mathsf{COR}_{\mathsf{PKE}}^{\mathcal{A}} \Rightarrow 1] \leq \delta(q)$.

Security. In this paper we are only considering IND-CPA security. We define the IND-CPA game as in Figure 1 (right), and the IND-CPA advantage function of an adversary $\mathcal{A} = (\mathcal{A}_1, A_2)$ against PKE as

$$\mathsf{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathcal{A}) = |\Pr[\mathsf{IND\text{-}CPA}_{\mathsf{PKE}}^{\mathcal{A}} \Rightarrow 1] - 1/2|\,.$$

2.2 The IND-CPA PKE underlying ML-KEM

We give a high-level algorithmic description of K-PKE, the IND-CPA-secure public-key encryption scheme underlying ML-KEM, in Algorithms 1 to 3. For a more implementation-oriented description that operates on byte arrays, see [31, Algs. 12–14].

ML-KEM works in the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$ with $q = 3329$ and $n = 256$. The core operations are on small-dimension vectors and matrices over $\mathcal{R}_q$; the dimension depends on the parameter k, which is different for different parameter sets of ML-KEM: ML-KEM-512 (NIST security level 1) uses $k = 2$, ML-KEM-768 (NIST security level 3) uses $k = 3$, and ML-KEM-1024 (NIST security level 5) uses $k = 4$. We denote elements in $\mathcal{R}_q$ with regular lower-case letters (e.g., v); vectors over $\mathcal{R}_q$ with bold-face lower-case letters (e.g., $\mathbf{u}$), and matrices over $\mathcal{R}_q$ with bold-face upper-case letters (e.g., $\mathbf{A}$).

In these descriptions, XOF is an extendable output function that in ML-KEM is instantiated with SHAKE-128 [34]. Parse interprets outputs of XOF as sequence of 12-bit unsigned integers and runs rejection sampling to obtain coefficients that look uniformly random modulo q. CBD_η denotes sampling coefficients from a centered binomial distribution with parameter η;[6] extension from coefficients to (vectors of) polynomials is done by sampling each coefficient independently from CBD_η. For example, both ML-KEM-768 and ML-KEM-1024 use $\eta_1 = \eta_2 = 2$. The sampling routine is parameterized by a pseudorandom function PRF_k with key k. NTT is the number-theoretic transform of a polynomial in $\mathcal{R}_q$. Both input and output of NTT can be written as a sequence of 256 coefficients in $\mathbb{Z}_q$ and typical implementations perform the transform inplace. However, output coefficients do not have any meaning as a polynomial in $\mathcal{R}_q$. We therefore denote the output domain as $\hat{\mathcal{R}}_q$; we apply the same notation for elements in $\hat{\mathcal{R}}_q$, e.g., $\hat{u} = \mathsf{NTT}(u)$. Application of NTT to vectors and matrices over $\mathcal{R}_q$ is done element-wise.

[6] This means we have $B(n, p)$ with $p = 1/2$, $n = 2\eta$ and expected value shifted to 0.

Compress_q compresses elements in $\mathcal{R}_q$ (or $\mathcal{R}_q^k$) by rounding coefficients to a smaller modulus 2^{d_v} (or 2^{d_u}). For ML-KEM-768 we have $d_v = 4$ and $d_u = 10$. For ML-KEM-1024 we have $d_v = 5$ and $d_u = 11$. Decompress_q is an approximate inverse of Compress_q. For an integer $x \in [0..3329)$, these functions are defined as:

$$\text{Compress}_q(x, d) = \lfloor (2^d/q) \cdot x \rceil \mod 2^d$$
$$\text{Decompress}_q(x, d) = \lfloor (q/2^d) \cdot x \rceil .$$

ToPoly maps 256-bit strings to elements in $\mathcal{R}_q$ by mapping a zero bit to a zero coefficient and mapping a one bit to a $\frac{q}{2}$ coefficient; ToMsg rounds coefficients to bits to recover a message from a noisy version of a polynomial generated by ToPoly.

2.3 The IND-CPA PKE underlying FrodoKEM

FrodoKEM is based on the algebraically unstructured LWE problem. It uses an error distribution that closely approximates a wide Gaussian distribution, parameterized to guarantee that the best known attacks on the resulting LWE instance require a computational effort that is well beyond the one mandated by the target security level. In addition, FrodoKEM is designed with simplicity in mind [2], as evidenced by: (1) its use of integer modulo $q \leq 2^{16}$, which is always a power of 2; (2) the main operations in the scheme consisting of simple matrix-vector multiplications, unlike the more complex operations in systems based on algebraically structured LWE variants; and (3) its straightforward encoding of secret bits by multiplying by $q/2^B$ (for B bits), avoiding the complex bandwidth-saving optimizations required by some Ring-LWE-based and Module-LWE-based schemes.

FrodoKEM is parameterized by the pseudorandom function (PRF) used to generate the public matrix $\mathbf{A}$. Two options are available for generating $\mathbf{A}$: AES-128 and SHAKE-128.

2.3.1 Technical description of FrodoKEM.. We give a high-level description of the IND-CPA PKE scheme underlying FrodoKEM in Algorithm 4, Algorithm 5 and Algorithm 6. FrodoKEM works under a quotient ring $\mathbb{Z}_q$ and the main operations are on matrices over $\mathbb{Z}_q$, where FrodoKEM-640 uses $q = 2^{15}$ while FrodoKEM-976 and FrodoKEM-1344 use $q = 2^{16}$. Gen generates a pseudorandom matrix $\mathbf{A}$ either by SHAKE128 or AES128. The SHA-3-derived extendable output function SHAKE is either SHAKE128 or SHAKE256 determined by the parameter set (FrodoKEM-640 uses SHAKE128 and FrodoKEM-976, FrodoKEM-1344 use SHAKE256). The SampleMatrix function samples an n_1-by-n_2 matrix with each entry sampled from the error distribution χ, which is a discrete and symmetric distribution centered at zero and closely approximating a moderately wide Gaussian distribution (denoted as $\chi_{\text{FrodoKEM}-640}$, $\chi_{\text{FrodoKEM}-976}$, $\chi_{\text{FrodoKEM}-1344}$ for the 3 NIST security levels respectively).

The Encode function encodes a bit string into a matrix and the Decode function decodes a matrix into a bit string using the following encoding and decoding functions, given $2^B \leq q$ and $0 \leq k < 2^B$:

$$\text{Encode}(k) = k \cdot q/x^B \qquad \text{Decode}(c) = \lfloor c \cdot 2^B/q \rceil \mod 2^B$$

Algorithm 4 FrodoPKE.Gen(): key generation

Ensure: Key pair $(pk, sk) \in (\{0,1\}^{len_{\text{seed}_\text{A}}} \times \mathbb{Z}_q^{n\times\bar{n}}) \times \mathbb{Z}_q^{n\times\bar{n}}$

1: $\text{seed}_\mathbf{A} \hookleftarrow \{0,1\}^{len_{\text{seed}_\text{A}}}$
2: $\mathbf{A} \leftarrow \text{Gen}(\text{seed}_\mathbf{A})$
3: $\text{seed}_{\mathbf{SE}} \hookleftarrow \{0,1\}^{len_{\text{seed}_{\mathbf{SE}}}}$
4: $(\mathbf{r}^{(0)}, ..., \mathbf{r}^{(2n\bar{n}-1)}) \leftarrow \text{SHAKE}(0x5F||\text{seed}_{\mathbf{SE}}, 2n\bar{n} \cdot len_\chi)$
5: $\mathbf{S}^T \leftarrow \text{SampleMatrix}((\mathbf{r}^{(0)}, ..., \mathbf{r}^{(n\bar{n}-1)}), \bar{n}, n, T_\chi)$
6: $\mathbf{E} \leftarrow \text{SampleMatrix}((\mathbf{r}^{(n\bar{n})}, ..., \mathbf{r}^{(2n\bar{n}-1)}), n, \bar{n}, T_\chi)$
7: $\mathbf{B} = \mathbf{AS} + \mathbf{E}$
8: **return** $(pk, sk) \leftarrow ((\text{seed}_\mathbf{A}, \mathbf{B}), \mathbf{S}^T)$

Algorithm 5 FrodoPKE.Enc(pk, μ): encryption

Require: Public key $pk = (\text{seed}_\mathbf{A}, \mathbf{B}) \in \{0,1\}^{len_{\text{seed}_\text{A}}} \times \mathbb{Z}_q^{n\times\bar{n}}$ and message $\mu \in \mathcal{M}$
Ensure: Ciphertext $\mathbf{c} = (\mathbf{C}_1, \mathbf{C}_2) \in \mathbb{Z}_q^{\bar{m}\times n} \times \mathbb{Z}_q^{\bar{m}\times\bar{n}}$

1: $\mathbf{A} \leftarrow \text{Gen}(seed_A)$
2: $\text{seed}_{\mathbf{SE}} \hookleftarrow \{0,1\}^{len_{\text{seed}_{\mathbf{SE}}}}$
3: $(\mathbf{r}^{(0)}, ..., \mathbf{r}^{(2n\bar{n}-1)}) \leftarrow \text{SHAKE}(0x96||\text{seed}_{\mathbf{SE}}, (2\bar{m}n+\bar{m}\bar{n}) \cdot len_\chi)$
4: $\mathbf{S}' \leftarrow \text{SampleMatrix}((\mathbf{r}^{(0)}, ..., \mathbf{r}^{(\bar{m}n-1)}), \bar{m}, n, T_\chi)$
5: $\mathbf{E}' \leftarrow \text{SampleMatrix}((\mathbf{r}^{(\bar{m}n)}, ..., \mathbf{r}^{(2\bar{m}n-1)}), \bar{m}, n, T_\chi)$
6: $\mathbf{E}'' \leftarrow \text{SampleMatrix}((\mathbf{r}^{(2\bar{m}n)}, ..., \mathbf{r}^{(2\bar{m}n+\bar{m}\bar{n}-1)}), \bar{m}, \bar{n}, T_\chi)$
7: $\mathbf{B}' = \mathbf{S}'\mathbf{A} + \mathbf{E}'$
8: $\mathbf{V}' = \mathbf{S}'\mathbf{B} + \mathbf{E}''$
9: **return** $\mathbf{c} \leftarrow (\mathbf{C}_1, \mathbf{C}_2) = (\mathbf{B}', \mathbf{V} + \text{Encode}(\mu))$

Algorithm 6 FrodoPKE.Dec($sk, \mathbf{c}$): decryption

Require: Ciphertext $\mathbf{c} = (\mathbf{C}_1, \mathbf{C}_2) \in \mathbb{Z}_q^{\bar{m}\times n} \times \mathbb{Z}_q^{\bar{m}\times\bar{n}}$ and secret key $sk = \mathbf{S}^T \in \mathbb{Z}_q^{\bar{n}\times n}$
Ensure: Decrypted message $\mu' \in \mathcal{M}$

1: $\mathbf{M} = \mathbf{C}_2 - \mathbf{C}_1\mathbf{S}$
2: **return** message $\mu' \leftarrow \text{Decode}(\mathbf{M})$

2.4 The EasyCrypt proof assistant

EasyCrypt[7] [6] is a proof assistant for formalizing proofs of cryptographic properties. Its primary feature is the Probabilistic Relational Hoare Logic (pRHL), which we use throughout to prove equivalences between games. pRHL is designed to support reasoning about equivalences of probabilistic programs while reasoning only locally (within oracles) and without reasoning about the distribution of specific variables—essentially keeping track only of the fact that variables in one program are distributed identically to variables in the other, but not keeping track of what that distribution may be. This logic has proved highly expressive for the bulk of cryptographic proof work. However, some steps require more global reasoning (about the entire execution) or keeping track of the distribution of individual variables. Logical rules to support such reasoning steps are implemented in EasyCrypt, but are often unwieldy to apply in concrete context. The EasyCrypt team has, over the years, developed a number of generic libraries that abstract those more complex reasoning rules into "game transformations" or

[7] https://easycrypt.info

equivalence results that can be instantiated as part of other proofs. Our proof makes use, in particular, of the Hybrid theory, which provides a formalized and generic argument for bounding the distance between two games that differ only in one oracle, but where the transition must be done query-by-query for the purpose of the proof. We also rely on the PROM theory, which provides a generic argument, initially intended to apply to programmable random oracles, that encapsulates the widely used argument that one can move the sampling of a value that is independent of the adversary's view.

3 Analysis of K-PKE

All the results stated in this section are formally verified in EasyCrypt. Our proofs are conducted over the simplified specification of the PKE shown in Figure 2. We factor out the encoding and decoding of ring elements (and vectors thereof) to operators ⋆_encd and ⋆_decd. For public keys and secret keys, these operators are simply assumed to form a bijection, which was formally proved in [4], and implies that they play no role in the security and correctness analyses. For ciphertexts and messages, we will see that the definitions of encoding and decoding vary with different variants of ML-KEM. However, one can express all results generically so as to cover all variants, and only deal with a full instantiation when a computation of a probability is needed. We assume an arbitrary distribution $\mathcal{SD}$ over seeds, and consider the case where all the ring elements are sampled from the same binomial distribution $\mathcal{B}$ (shown as $\mathcal{B}^k$ when applied to vectors).[8] Finally, the matrix $\mathbf{A}$ is taken as the output of a random oracle.

We justify the simplifications in this specification as follows. In practice, the sampling procedure for $\mathbf{A}$ is public, so there is really no way to argue that $\mathbf{A}$ has a distribution that looks uniform to an adversary. However, modeling sampling procedure as a random oracle allows formally relating the security of ML-KEM to the standard MLWE problem, and it is aligned with the intuition of using SHA-3-based rejection sampling to compress $\mathbf{A}$ into a small seed. In our analysis we will also take $\mathcal{B}$ to be the binomial distribution, rather than the SHA-3-based sampling procedure used in practice. It was proved in [4] that this procedure generates noise that is computationally close to the binomial distribution if the specific variant of SHA-3 used in that process is a secure PRF—and this holds statistically in the random oracle model. To summarize: our results rely heavily on the random oracle heuristic to justify that the distributions over which we perform the computations are good approximations of those occurring in ML-KEM. Nevertheless, the simplifications we introduce in this way are natural and they are aligned with prior analyses of ML-KEM [11].

3.1 Security analysis

We have formally verified a security proof of the K-PKE alternative to that given in [4]. The difference in this formalization is that we establish a direct connection to the standard MLWE assumption, which we are able to do because we model the sampling of $\mathbf{A}$ as coming from a random oracle. We do not claim any novelty here, but we present the result because the intuition helps understand the need for and difficulties associated to building a reduction to MLWE when reasoning about correctness in the rest of the section.

The proof is carried out in the random oracle model (ROM) in two steps. We first define the MLWE problem in the ROM in the natural way: the adversary has access to a random oracle mapping a seed to a matrix $\mathbf{A}$, and receives as challenge a vector $\mathbf{t} = \mathbf{As} + \mathbf{e}$ and a random seed sd that was used by the challenger to retrieve $\mathbf{A}$ from the random oracle. A simple reduction permits proving that distinguishing $\mathbf{t}$ from a vector sampled uniformly at random amounts is equivalent to the standard MLWE problem where $\mathbf{A}$ is given directly to the adversary a part of the challenge: the reduction just lazily simulates the RO itself, programming $\mathbf{A}$ in the point defined by sd.

The IND-CPA proof then proceeds in two hops justified using MLWE in the ROM. The first hop replaces the $\mathbf{t}$ vector in the public key with a uniform vector. The second hop uses the fact that $\mathbf{t}$ can now be seen as an extra row in an MLWE challenge matrix and replaces both MLWE samples computed in the ciphertext $(\mathbf{u}, \langle \mathbf{t}, \mathbf{r} \rangle + e_2)$ with uniform values. Both steps are justified by reductions that receive an MLWE challenge and construct for the adversary a perfect interpolation between the two games involved in the hop: if the reduction is given an MLWE sample, the adversary is run in the game on the left, and otherwise it is run in the game on the right. In the final game it is clear that all information about the message is information-theoretically hidden from the adversary, as this is masked by a value sampled uniformly and independently at random. So the probability of correctly guessing the challenge bit b is exactly $1/2$. Combining all the proof steps, we can express the security of the K-PKE in the ROM in terms of the standard MLWE problem.

3.2 Correctness Analysis

The proof of correctness first rearranges the correctness game in Figure 1 instantiated with the K-PKE into the form shown in Figure 3 (left). A simple algebraic argument permits showing that the noise that remains added to the message m is given by the expression assigned to $\tilde{n}$ in the figure, and $\lfloor q/4 \rfloor - 1$ is the maximum noise value above which a decoding error can occur in the message recovery. Note that the noise expression includes two terms $\mathbf{c}_u$ and c_v that capture the inaccuracy introduced by encoding ciphertexts via rounding to a smaller noise: these are expressed as additive noise, each of them defined as the difference between the original value and the decoded value.

Our goal is to provide an upper-bound for the probability that the noise threshold is exceeded in at least one of the 256 coefficients in $\tilde{n}$. The way that this is typically done in the literature is to argue that, although the joint distribution of all 256 coefficients is complex, the distribution of each coefficient individually can actually be proven to be the same. Moreover, this distribution is simple enough that one can just exhaustively compute the probability mass function over all elements in the support to obtain an exact upperbound for the probability ϵ of one coefficient in $\tilde{n}$ exceeding the noise threshold. A union bound then permits obtaining a final bound of $256 \cdot \epsilon$.

As we will see in the next section, this argument applies directly in the case of FrodoKEM, because the noise expression nicely

[8] This means our proof doesn't strictly cover ML-KEM-512, but it can be easily extended to do so.

Algorithm $\text{Gen}^{\mathcal{O}}()$:
1: $sd \hookleftarrow \mathcal{SD}$
2: $\mathbf{A} \leftarrow \mathcal{O}(sd)$
3: $\mathbf{s}, \mathbf{e} \hookleftarrow \mathcal{B}^k$
4: $\mathbf{t} \leftarrow \mathbf{As} + \mathbf{e}$
5: $pk \leftarrow \text{pk\_encd}(\mathbf{t}, sd)$
6: $sk \leftarrow \text{sk\_encd}(\mathbf{s})$
7: **return** (pk, sk)

Algorithm $\text{Enc}^{\mathcal{O}}(pk, m)$:
1: $(\mathbf{t}, sd) \leftarrow \text{pk\_decd}(pk)$
2: $\mathbf{A} \leftarrow \mathcal{O}(sd)$
3: $\mathbf{r}, \mathbf{e}_1 \hookleftarrow \mathcal{B}^k$
4: $e_2 \hookleftarrow \mathcal{B}$
5: $\mathbf{u} \leftarrow \mathbf{A}^\top \mathbf{r} + \mathbf{e}_1$
6: $v \leftarrow \langle \mathbf{t}, \mathbf{r} \rangle + e_2 + \text{m\_encd}(m)$
7: $c \leftarrow \text{c\_encd}(\mathbf{u}, v)$
8: **return** c

Algorithm $\text{Dec}^{\mathcal{O}}(sk, c)$:
1: $\mathbf{s} \leftarrow \text{sk\_decd}(sk)$
2: $(\mathbf{u}, v) \leftarrow \text{c\_decd}(c)$
3: $m \leftarrow \text{m\_decd}(v - \langle \mathbf{s}, \mathbf{u} \rangle)$
4: **return** m

Figure 2: Abstract specification of K-PKE.

Game $\text{COR}_{\text{ML-KEM}}$:
1: $sd \hookleftarrow \mathcal{SD}$
2: $\mathbf{s}, \mathbf{e} \hookleftarrow \mathcal{B}^k$
3: $\mathbf{r}, \mathbf{e}_1 \hookleftarrow \mathcal{B}^k$
4: $e_2 \hookleftarrow \mathcal{B}$
5: $\mathbf{A} \leftarrow \mathcal{O}(sd)$
6: $\mathbf{t} \leftarrow \mathbf{As} + \mathbf{e}$
7: $pk \leftarrow \text{pk\_encd}(\mathbf{t}, sd)$
8: $sk \leftarrow \text{sk\_encd}(\mathbf{s})$
9: $m \hookleftarrow \mathcal{A}^{\mathcal{O}}(pk, sk)$
10: $\mathbf{u} \leftarrow \mathbf{A}^\top \mathbf{r} + \mathbf{e}_1$
11: $v \leftarrow \langle \mathbf{t}, \mathbf{r} \rangle + e_2 + \text{m\_encd}(m)$
12: $(\mathbf{c}_u, c_v) \leftarrow \text{c\_decd}(\text{c\_encd}(\mathbf{u}, v)) - (\mathbf{u}, v)$
13: $\tilde{n} \leftarrow \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 \rangle - \langle \mathbf{s}, \mathbf{c}_u \rangle + e_2 + c_v$
14: **return** $\|\tilde{n}\|_\infty > \lfloor q/4 \rfloor - 1$

Game $\text{COR}^{\text{heuristic}}_{\text{ML-KEM}}$:
1:
2: $\mathbf{s}, \mathbf{e} \hookleftarrow \mathcal{B}^k$
3: $\mathbf{r}, \mathbf{e}_1 \hookleftarrow \mathcal{B}^k$
4: $e_2 \hookleftarrow \mathcal{B}$
5:
6:
7:
8:
9:
10: $\mathbf{u} \hookleftarrow \mathcal{U}(R_q^k)$
11: $v \hookleftarrow \mathcal{U}(R_q)$
12: $(\mathbf{c}_u, c_v) \leftarrow \text{c\_decd}(\text{c\_encd}(\mathbf{u}, v)) - (\mathbf{u}, v)$
13: $\tilde{n} \leftarrow \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 \rangle - \langle \mathbf{s}, \mathbf{c}_u \rangle + e_2 + c_v$
14: **return** $\|\tilde{n}\|_\infty > \lfloor q/4 \rfloor - 1$

Figure 3: Rearranged correctness experiment for ML-KEM (left). Heuristic approximation (right)

decomposes into summations and products of values independently sampled from distributions with small support. However, in the case of ML-KEM there is a problem: the distribution of the noise is affected by $\mathbf{c}_u$ and c_v, which break this convenient behavior of the noise expression. We now consider three alternatives to addressing this problem.

3.2.1 Heuristic approximation. The solution proposed in [11] and used in the Kyber submission to the NIST competition simply assumes that one can take the probability of the event defined in the experiment in Figure 3 (right) as an upper-bound for the correctness error. The assumption here is that $(\mathbf{u}, v)$ are taken to be values sampled uniformly at random and independently from everything else in the noise expression. This immediately allows brute-forcing the probability computation: what is crucial here is that the distribution of the error introduced by rounding each ciphertext coefficient is independent from other coefficients, which allows deriving a simple (computable) description of the final noise distribution.

Justifying that this assumption is reasonable seems, at first sight, to follow from the MLWE problem: in fact, in the security proof we described above, $(\mathbf{u}, v)$ and shown to be computationally close to uniform after the two game hops we described above. However, the same game-hopping reasoning does not directly apply here: to transform the COR experiment we would need to build a reduction $\mathcal{B}$ from MLWE, and this algorithm would need to 1) provide the secret key to the adversary, and 2) construct its guess based on whether the COR experiment would return true or false. We note that, not only does $\mathcal{B}$ not know the secret key $\mathbf{s}$ to give to $\mathbf{A}$, but also point 2) implies computing the noise expression explicitly using the secret key $\mathbf{s}$ plus all ephemeral noise values not revealed by the MLWE experiment. In Section 5 we explain how we obtain a formally verified computed bound for the heuristic explained above that confirms the accuracy of the claims in [11]. We conclude this section on how one could provably bound the probability above without the heuristic.

3.2.2 Removing the adversary. The discussion above shows that, unless one removes the need to provide adversary $\mathcal{A}$ with the secret key, there is little hope of relying on the MLWE assumption in this

Game $\text{COR}^1_{\text{ML-KEM}}$:
1: $sd \hookleftarrow \mathcal{SD}$
2: $\mathbf{s}, \mathbf{e} \hookleftarrow \mathcal{B}^k$
3: $\mathbf{r}, \mathbf{e}_1 \hookleftarrow \mathcal{B}^k$
4: $e_2 \hookleftarrow \mathcal{B}$
5: $\mathbf{A} \leftarrow \mathcal{O}(sd)$
6: $\mathbf{u} \leftarrow \mathbf{A}^\top \mathbf{r} + \mathbf{e}_1$
7: $\mathbf{c}_u \leftarrow \text{c\_decd}(\text{c\_encd}(\mathbf{u})) - \mathbf{u}$
8: $\tilde{n}_1 \leftarrow \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 \rangle + e_2$
9: $\tilde{n}_2 \leftarrow \langle \mathbf{s}, \mathbf{c}_u \rangle$
10: **return** $\|\tilde{n}_1 - \tilde{n}_2\|_\infty > \lfloor q/4 \rfloor - 1 - t^{\max}_{c_v}$

Game $\text{COR}^2_{\text{ML-KEM}}$:
1: $\mathbf{s} \hookleftarrow \mathcal{B}^k$
2: $\mathbf{u} \hookleftarrow \mathcal{U}(R_q^k)$
3: $\mathbf{c}_u \leftarrow \text{c\_decd}(\text{c\_encd}(\mathbf{u})) - \mathbf{u}$
4: $\tilde{n}_2 \leftarrow \langle \mathbf{s}, \mathbf{c}_u \rangle$
5: **return** $\|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}$

Figure 4: Removing the adversary (left). Provable bound under MLWE (right).

setting. However, it is clear that the adversary's influence in the outcome of the experiment is limited to choosing m, so one can just consider the worst case value of c_v, i.e. the smallest integer $t^{\max}_{c_v}$ such that

$$\Pr[\text{COR}_{\text{ML-KEM}} : \|c_v\|_\infty > t^{\max}_{c_v}] = 0$$

And redefining the experiment to output $\|\tilde{n}'\|_\infty > \lfloor q/4 \rfloor - 1 - t^{\max}_{c_v}$, where

$$\tilde{n}' := \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 \rangle - \langle \mathbf{s}, \mathbf{c}_u \rangle + e_2$$

We show the resulting game $\text{COR}^1_{\text{ML-KEM}}$ in Figure 4 (left)[9]. Clearly, it is straightforward to prove that upperbounding the probability that the (reduced) noise threshold is reached in this new experiment also yields an upper-bound for the original experiment and therefore for the correctness of the K-PKE. We note that one can still not immediately justify that $\mathbf{u}$ can be assumed to be uniform under MLWE: a reduction from MLWE would still need to decide whether the noise threshold is exceeded in order to produce a guess, and this implies explicitly computing $\tilde{n}'$. Nevertheless, it is still interesting to assess how much is lost by max-ing out the adversary's influence in the bound, so we also consider this case in Section 5.[10]

[9] We split the noise expression of $\tilde{n}'$ into two sub-expressions, because this is useful for the discussion that follows.
[10] To be precise, this heuristic bound can be defined in terms of the experiment in Figure 4 (right), considering the event returned by experiment Figure 4 (left).

3.2.3 A provable bound. To obtain a (sub-optimal) provable bound, we introduce two noise thresholds t and $t_{\mathbf{c}_u}$, where the latter is allocated to the term depending on $\mathbf{c}_u$ and the former to the remaining terms in the noise expression. Clearly, setting $t + t_{\mathbf{c}_u} = \lfloor q/4 \rfloor - 1 - t_{c_v}^{\max}$, the probability that the experiment returns true can be upper-bounded by taking the union bound as follows:

$$\Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \top] \leq \Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \|\tilde{n}_1\|_\infty > t] + \Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}] .$$

In other words, we can analyze the two events independently. Furthermore, the first probability corresponding to $\tilde{n}_1$ is already in a form that can be computed by exhaustive evaluation: and note that this intuitively corresponds to the probability of a decryption error if no rounding was used by ML-KEM. On the other hand, the term corresponding to $\tilde{n}_2$ can now be justified using MLWE: we introduce an additional experiment $\mathrm{COR}^2_{\text{ML-KEM}}$ in Figure 4 (right) and prove that a simple reduction to MLWE can be used to justify that:

$$|\Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}] - \Pr[\mathrm{COR}^2_{\text{ML-KEM}} : \|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}]| \leq \epsilon_{\text{LWE}} .$$

The reduction is now trivial, since it only needs to generate $\mathbf{s}$ itself to check if an error occurred. Finally, we can use $\mathrm{COR}^2_{\text{ML-KEM}}$ to brute-force the probability of exceeding threshold $t_{\mathbf{c}_u}$, with the guarantee that any error significant introduced by the approximation would imply an attack on MLWE:

$$\Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \top] \leq \Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \|\tilde{n}_1\|_\infty > t] + \Pr[\mathrm{COR}^2_{\text{ML-KEM}} : \|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}] + \epsilon_{\text{LWE}} .$$

Note that this provable bound comes at a cost: we are over-approximating the decryption error by declaring the adversary the winner whenever noise terms, which could cancel each-other out in the full noise expression, exceed individually a partial threshold. We will see the impact in Section 5. Another way to interpret this bound is the following: we can statistically bound the error when no rounding is used, so we exclude this possibility with a conservative threshold. Then, the only way that an error could occur is caused by the rounding component and we are able to use the MLWE assumption to formally exclude this possibility. In this analysis we lose precision because, clearly, the summation of both noise components could still be small enough not to cause an error, even if one of them exceeds its threshold.

4 Analysis of FrodoKEM PKE

We now turn our attention to FrodoKEM PKE. We proceed in the same way as we did for ML-KEM, first introducing the abstract specification over which we conducted the analysis, and then describing the security and correctness proofs. The main difference to ML-KEM is that, here, we can present a security proof that goes down to the standard LWE problem and, most importantly, we can give a functional correctness proof that doesn't require any heuristic approximations (except for the ROM) and yields a machine-checked proof that the PKE decryption error probability is bound by the numbers claimed by the designers of FrodoKEM. We believe that, even though these observations are folklore knowledge in the community, it is interesting to highlight the fact that there are provable security costs inherent to the introduction of optimizations in the design of post-quantum schemes. As before, all the results stated in this section have been machine-checked in EasyCrypt.

Algorithm $\mathrm{Gen}^O()$:
1: $sd \hookleftarrow \mathcal{SD}$
2: $\mathbf{A} \leftarrow O(sd)$
3: $\mathbf{S} \hookleftarrow \mathcal{M}_{\mathcal{X}}^{n \times \bar{n}}$
4: $\mathbf{E} \hookleftarrow \mathcal{M}_{\mathcal{X}}^{m \times \bar{n}}$
5: $\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$
6: $pk \leftarrow \mathrm{pk\_encd}(\mathbf{B}, sd)$
7: $sk \leftarrow \mathrm{sk\_encd}(\mathbf{S})$
8: **return** (pk, sk)

Algorithm $\mathrm{Enc}^O(pk, m)$:
1: $(\mathbf{B}, sd) \leftarrow \mathrm{pk\_decd}(pk)$
2: $\mathbf{A} \leftarrow O(sd)$
3: $\mathbf{S}' \hookleftarrow \mathcal{M}_{\mathcal{X}}^{\bar{m} \times m}$
4: $\mathbf{E}' \hookleftarrow \mathcal{M}_{\mathcal{X}}^{\bar{m} \times n}$
5: $\mathbf{E}'' \hookleftarrow \mathcal{M}_{\mathcal{X}}^{\bar{m} \times \bar{n}}$
6: $\mathbf{U} \leftarrow \mathbf{S}'\mathbf{A} + \mathbf{E}'$
7: $\mathbf{V} \leftarrow \mathbf{S}'\mathbf{B} + \mathbf{E}'' + \mathrm{m\_encd}(m)$
8: $c \leftarrow \mathrm{c\_encd}(\mathbf{U}, \mathbf{V})$
9: **return** c

Algorithm $\mathrm{Dec}^O(sk, c)$:
1: $\mathbf{S} \leftarrow \mathrm{sk\_decd}(sk)$
2: $(\mathbf{U}, \mathbf{V}) \leftarrow \mathrm{c\_decd}(c)$
3: $m \leftarrow \mathrm{m\_decd}(\mathbf{V} - \mathbf{US})$
4: **return** m

Figure 5: Abstract specification of FrodoKEM PKE.

4.0.1 The specification. The specification is given in Figure 5. We follow very much the same approach as for ML-KEM in modeling the encoding and decoding operations, and the introduction of the random oracle to sample $\mathbf{A}$. The main difference is the use of a general theory for matrices that allows fixing the dimensions dynamically, rather than working with hardwired dimensions in the type. This allows us to reason about the relations between various LWE definitions and reductions involving matrices of different dimensions, as well as distributions over matrices of different dimensions, in a unified context. Note in the figure the use of $\mathcal{M}_{\mathcal{X}}^{a \times b}$ to denote the lifting of a distribution over field elements $\mathcal{X}$ to matrices of dimension $a \times b$. Again, the encoding and decoding of public and secret keys is irrelevant for our results and is treated as an abstract bijection. Moreover, for FrodoKEM, the encoding and decoding of ciphertexts is also a bijection, so the only encoding/decoding operators that need to be taken into consideration for the correctness bound are the ones applied to the message.

4.1 Security analysis

We have formally verified a security proof of the FrodoKEM PKE. The novelty here compared to prior machine-checked security proofs for post-quantum PKEs is pushing the reduction down to standard LWE, relying on the fact that we are assuming the matrix $\mathbf{A}$ to be produced by a random oracle. The proof is carried out in the random oracle model in three steps.

4.1.1 From LWE to Matrix LWE.. We first define the standard LWE problem and the Matrix LWE problem. Both provide the adversary with a challenge $(\mathbf{A}, \mathbf{B})$, and the adversary must guess whether this challenge comes from a real or an ideal distribution. In the real distribution, the challenge is constructed as $\mathbf{B} = \mathbf{AS} + \mathbf{E}$ where $\mathbf{A}$, $\mathbf{S}$ and $\mathbf{E}$ are matrices over some ring R_q. Matrix $\mathbf{A}$ is sampled from the uniform distribution, whereas $\mathbf{S}$ and $\mathbf{E}$ are sampled from some arbitrary distribution $\mathcal{X}$ lifted to the appropriate matrix dimensions. In the ideal distribution $\mathbf{A}$ and $\mathbf{B}$ are sampled independently and uniformly at random. For the Matrix LWE problem, we have $\mathbf{A} \in$

$R_q^{m\times n}$, $\mathbf{S} \in R_q^{n\times \bar{n}}$, and $\mathbf{B}, \mathbf{E} \in R_q^{m\times \bar{n}}$. In the standard LWE problem we have $\bar{n} = 1$, so $\mathbf{S}$, $\mathbf{B}$ and $\mathbf{E}$ are column vectors. We prove the following theorem:

Theorem 4.1. *For every adversary $\mathcal{A}$ attacking the Matrix LWE problem, there exists an adversary $\mathcal{B}$ such that:*

$$\Pr[\mathsf{LWE}_\chi^{m,n,\bar{n}}(\mathcal{A}) \Rightarrow 1|b=1] - \Pr[\mathsf{LWE}_\chi^{m,n,\bar{n}}(\mathcal{A}) \Rightarrow 1|b=0] \leq$$
$$\bar{n} \cdot (\Pr[\mathsf{LWE}_\chi^{m,n,1}(\mathcal{B}) \Rightarrow 1|b=1] - \Pr[\mathsf{LWE}_\chi^{m,n,1}(\mathcal{B}) \Rightarrow 1|b=0])$$

The proof follows a hybrid argument, where at each step one of the columns of the Matrix LWE challenge is flipped from the real distribution to the ideal distribution. This proof, which is straightforward on paper, required some effort to machine check. Notably, we needed to develop a general theory of distributions over matrices that permits reasoning about composing distributions over submatrices. Once this library was in place, we first re-expressed the Matrix LWE assumption as an experiment that samples the column vectors of the challenge matrix $\mathbf{B}$ in a while loop, one at a time—the new library allowed us to prove equivalence by induction. From that point on, we relied on the EasyCrypt libraries for general hybrid arguments, with extra support from the PROM theory when we needed to argue that the crucial i-th sample involved in a hybrid step could be pre-sampled outside of the loop—this is needed to construct a reduction to LWE for the i-th step: the challenge sample is given upfront, and then needs to be programmed into the i-th loop iteration.

4.1.2 IND-CPA security in the ROM.. The second step in the proof is to show that the Matrix LWE assumption in the ROM follows from Matrix LWE assumption in the standard model and therefore from LWE. This proof step is similar to the one we presented in the previous section for MLWE. Finally, the IND-CPA security proof for the FrodoKEM PKE follows the same structure as the one for the K-PKE, comprising two hops. The first hop uses the Matrix LWE assumption (in the ROM) to justify sampling matrix $\mathbf{B}$ in the public key as a uniform matrix. The second hop uses the Matrix LWE assumption (in the ROM) to justify making the ciphertext uniform. By showing that the adversary's advantage in the final game is 0, and plugging in the previous results on Matrix LWE, we obtain the following theorem for FrodoKEM.

Theorem 4.2. *The FrodoKEM PKE is IND-CPA secure under the LWE assumption in the Random Oracle Model. More precisely, for every adversary $\mathcal{A}$ against FrodoKEM, there exist adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that:*

$$\Pr[\mathsf{IND\text{-}CPA}^{\mathrm{RO}}_{\mathit{FrodoKEM}}(\mathcal{A}) \Rightarrow 1|b=1] -$$
$$\Pr[\mathsf{IND\text{-}CPA}^{\mathrm{RO}}_{\mathit{FrodoKEM}}(\mathcal{A}) \Rightarrow 1|b=0] \leq$$
$$\bar{n} \cdot (\Pr[\mathsf{LWE}_\chi^{m,n,1}(\mathcal{B}_1) \Rightarrow 1|b=1] -$$
$$\Pr[\mathsf{LWE}_\chi^{m,n,1}(\mathcal{B}_1) \Rightarrow 1|b=0]) +$$
$$\bar{m} \cdot (\Pr[\mathsf{LWE}_\chi^{m,n+\bar{n},1}(\mathcal{B}_2) \Rightarrow 1|b=1] -$$
$$\Pr[\mathsf{LWE}_\chi^{m,n+\bar{n},1}(\mathcal{B}_2) \Rightarrow 1|b=0])$$

Again, although the proof is straightforward on paper and conceptually identical to the proof for the K-PKE, there were some machine-checking challenges we needed to overcome in order to conclude it. In particular, the second hop in the security proof requires again to reason about the distributions of sub-matrices: the public key $(\mathbf{A}, \mathbf{B})$ is now seen as a monolythic LWE public matrix $[\mathbf{AB}]$, so one must reason compositionally about the distribution of the Matrix LWE challenge, when proving that the reduction matches the distribution of the security games over which the hop is being carried out—in these games $\mathbf{A}$ and $\mathbf{B}$ are constructed separately.

Game $\mathrm{COR}^{\mathrm{provable}}_{\mathrm{FrodoKEM}}$:
1: $\mathbf{S} \hookleftarrow \mathcal{M}_\chi^{n\times\bar{n}}$
2: $\mathbf{E} \hookleftarrow \mathcal{M}_\chi^{m\times\bar{n}}$
3: $\mathbf{S}' \hookleftarrow \mathcal{M}_\chi^{\bar{m}\times m}$
4: $\mathbf{E}' \hookleftarrow \mathcal{M}_\chi^{\bar{m}\times n}$
5: $\mathbf{E}'' \hookleftarrow \mathcal{M}_\chi^{\bar{m}\times\bar{n}}$
6: $(\mathbf{c_U}, \mathbf{c_V}) \leftarrow \mathsf{c\_decd}(\mathsf{c\_encd}(\mathbf{U}, \mathbf{V})) - (\mathbf{U}, \mathbf{V})$
7: $\mathbf{N} \leftarrow \mathbf{S}'\mathbf{E} - \mathbf{E}'\mathbf{S} + \mathbf{E}''$
8: **return** $\exists ij, \neg(-q/2^{B+1} \leq \mathbf{N}[i,j] < q/2^{B+1})$

Figure 6: Provable statistical bound for FrodoKEM

4.2 Correctness Analysis

We carry out our analysis of the FrodoKEM PKE correctness in pretty much the same way as was presented for ML-KEM. However, in this case, the analysis is much simpler. Indeed, a simple algebraic argument allows us to show that the experiment in Figure 6 provides a provable upper-bound for the failure probability in Frodo KEM, for any adversary. Indeed, the absence of any compression in ciphertexts gives us a nice cancellation in the decryption process, ending up with an error distribution that can be characterized based only on the distributions of the noise matrices. This allows us to formally connect the probability of a decryption failure with a machine-checked computed probability bound for this statistical event, as we will describe in the next section.

5 Computing formally-verified upper-bounds

In order to provide a machine-checked computation of an upper bound for each of the statistical events defined in Section 3 and Section 4, three steps are needed:

(1) Prove that the statistical event can be upper-bounded using a union bound and, in some cases, reducing the problem to the probability that a single integer modulo q is within a prescribed range. I.e., for ML-KEM we need to consider only the distribution of one polynomial coefficient, and in FrodoKEM we need only consider the distribution of one matrix entry.
(2) Prove that the probability above can be computed using an explicit functional formula over the reals, which essentially represents the construction of the probability mass function, followed by the computation of the tail probability.
(3) Extract the specification obtained in EasyCrypt to an OCaml program and execute it to compute the probability upper bound. One of the current limitations of our work is that this step is not done automatically: we have carefully crafted an OCaml program that syntactically closely matches the EasyCrypt specifications (with some caveats described below) and leave it as a direction for future work to automate this process.

We now describe how we achieve these goals.

5.1 Modular reasoning about distributions

Our formalization starts with a few basic definitions of distribution combiners. Throughout our formal development we consider only *lossless* distributions, where the summation of the probabilities of all values in the support adds up to 1. When performing approximate computations, this property may, of course, not be preserved.

Let $\mathcal{D}_i$, for $i \in 1, 2, \ldots$ denote arbitrary distributions over a ring R. Then we can define the distributions induced by addition, subtraction, multiplication and inner products as

$$\begin{aligned}
\mathcal{D}_1 \oplus \mathcal{D}_2 &:= \{a + b | a \hookleftarrow_\$ \mathcal{D}_1; b \hookleftarrow_\$ D_2\} \\
\mathcal{D}_1 \ominus \mathcal{D}_2 &:= \{a - b | a \hookleftarrow_\$ \mathcal{D}_1; b \hookleftarrow_\$ D_2\} \\
\mathcal{D}_1 \otimes \mathcal{D}_2 &:= \{a \cdot b : a \hookleftarrow_\$ \mathcal{D}_1; b \hookleftarrow_\$ D_2\} \\
\langle \mathcal{D}_1, \mathcal{D}_2 \rangle_n &:= \left\{ \sum_{i=1}^{n} \mathbf{a}_i \cdot \mathbf{b}_i : \mathbf{a} \hookleftarrow_\$ \mathcal{D}_1^n; \mathbf{b} \hookleftarrow_\$ D_2^n \right\} \\
\langle \mathcal{D}_1, \mathcal{D}_2 \rangle_n^{\mathcal{I}} &:= \left\{ \sum_{i=1}^{n} -1^{i \in \mathcal{I}} \cdot \mathbf{a}_i \cdot \mathbf{b}_i : \mathbf{a} \hookleftarrow_\$ \mathcal{D}_1^n; \mathbf{b} \hookleftarrow_\$ D_2^n \right\}
\end{aligned}$$

Here, the last distribution is a generalization of the inner product, where some terms are added and other terms are subtracted. This distribution is useful to describe multiplication in the polynomial ring underlying ML-KEM.

We note that, independently of the cardinality of the ring, if distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ have small enough support, then the probability mass functions of the distributions resulting from a small number of applications of these combiners can be computed by exhaustive evaluation. In particular, this is the case when $\mathcal{D}_i$ is one of the following distributions:

(1) The binomial distribution $\mathcal{B}$ over $\mathbb{Z}_q$ as used in all the variants of ML-KEM.
(2) The distribution of the rounding error resulting from rounding a uniform element in $\mathbb{Z}_q$ to a smaller modulus, required to analyze the probability of error in all of the ML-KEM variants.
(3) The distribution of the noise $\mathcal{X}$ over $\mathbb{Z}_q$ used in all of the FrodoKEM variants.

We first proved the following general result in EasyCrypt, for any ring and any $\mathcal{D}_1$ and $\mathcal{D}_2$:

$$\langle \mathcal{D}_1, \mathcal{D}_2 \rangle_n = \bigoplus_{1}^{n} \mathcal{D}_1 \otimes \mathcal{D}_2$$

This result permits computing the distribution of an inner product using a standard double and add algorithm, starting from the probability mass function of $\mathcal{D}_1 \otimes \mathcal{D}_2$ and computing the $\bigoplus$ combiner $O(\lg n)$ times.

We also define a restricted class of distributions, called good, if they satisfy the following two requirements, which intuitively mean that the distribution is symmetric and centered around zero and, furthermore, that zero is not the only element in the support:[11]

$$\text{good}(\mathcal{D}) \Rightarrow \begin{cases} \Pr[x = 0 | x \hookleftarrow_\$ \mathcal{D}] < 1 \\ \forall c, \Pr[x = c | x \hookleftarrow_\$ \mathcal{D}] = \Pr[x = -c | x \hookleftarrow_\$ \mathcal{D}] \end{cases}.$$

Note that both $\mathcal{B}$ and $\mathcal{X}$ mentioned above satisfy this property, which we prove in EasyCrypt, but the distribution of rounding errors in ML-KEM does not.

The following property is straightforward to prove in EasyCrypt for any ring:

$$\begin{aligned}
\text{good}(\mathcal{D}_2) \Rightarrow \quad & \mathcal{D}_1 \oplus \mathcal{D}_2 = \mathcal{D}_1 \ominus \mathcal{D}_2 \\
\text{good}(\mathcal{D}_1) \Rightarrow \text{good}(\mathcal{D}_2) \Rightarrow \quad & \text{good}(\mathcal{D}_1 \oplus \mathcal{D}_2)
\end{aligned}$$

Furthermore, if working over a field, which is the case of $\mathbb{Z}_q$ in ML-KEM (but not in FrodoKEM) we prove that, for any $\mathcal{I}$ and any n, have that:

$$\begin{aligned}
\text{good}(\mathcal{D}_1) \Rightarrow \text{good}(\mathcal{D}_2) \Rightarrow \quad & \langle \mathcal{D}_1, \mathcal{D}_2 \rangle_n = \langle \mathcal{D}_1, \mathcal{D}_2 \rangle_n^{\mathcal{I}} \\
\text{good}(\mathcal{D}_1) \Rightarrow \text{good}(\mathcal{D}_2) \Rightarrow \quad & \text{good}(\langle \mathcal{D}_1, \mathcal{D}_2 \rangle_n)
\end{aligned}$$

The existence of multiplicative inverses for all non-zero elements permits showing that multiplication preserves the good property, and the result then follows by induction.

Equipped with these general results, we can now look at how they permit proving the correctness of simple and efficiently computable formulas for the probability bounds defined in Section 3 and Section 4.

5.2 When all noise coefficients are alike

5.2.1 ML-KEM without rounding. The most elegant result can be established for the probability defined in Section 3 as

$$\Pr[\text{COR}^1_{\text{ML-KEM}} : \|\tilde{n}_1\|_\infty > \lfloor q/4 \rfloor - 1 - t^{\max}_{c_v} - t_{\mathbf{c}_u}]$$

We recall that this is intuitively the probability that the noise expression in ML-KEM exceeds a threshold t, if one does not consider the rounding noise. The noise expression in this case is given by:

$$\tilde{n}_1 := \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 \rangle + e_2$$

Here, $\mathbf{e}$, $\mathbf{r}$, and $\mathbf{s}$ are vectors of polynomials of size k, where each polynomial has 256 coefficients. e_2 and $\tilde{n}_1$ are each a single polynomial. Each coefficient in each of the input polynomials is sampled independently at random from the binomial distribution $\mathcal{B}$. Using the general results above we can prove the following theorem.

Theorem 5.1. *The distribution of each coefficient of $\tilde{n}_1$ is given by*

$$\langle \mathcal{B}, \mathcal{B} \rangle_{256k} \oplus \langle \mathcal{B}, \mathcal{B} \rangle_{256k} \oplus \mathcal{B}$$

Sketch. We first consider the operations over the ML-KEM polynomial ring. Addition is done coefficient-wise, so one can easily propagate the good property. Multiplication in the polynomial ring can be defined, for each coefficient of the result, as a generalized inner product over the coefficients. More precisely, if we see two polynomials a and b as vectors of coefficients of size 256, and we

[11] This latter requirement ensures that we are actually working with non-trivial noise distributions throughout the computation. It was recently pointed out to us that this requirement, although intuitive, may be unnecessarily strong if the goal is only to simplify the final probability expressions. Removing it would simplify the proofs of preservation of good and may allow deriving more general results, e.g., for commutative rings, that can be important for other use cases. This is not relevant for our examples, but the adaptation is simple and will be considered in future work.

slightly abuse notation, it is well known that one can write the formula for coefficient i of the product as:

$$(a \cdot b)_i = \langle a, b \rangle^{\mathcal{I}}_{256} \quad \text{for} \quad \mathcal{I} = \{k > i \mid k \in [0..256)\}$$

The EasyCrypt proof relies on this fact and the goodness of $\mathcal{B}$ to derive that a coefficient of the product of two polynomials sampled from $\mathcal{B}^{256}$ is given by $\langle \mathcal{B}, \mathcal{B} \rangle_{256}$. The proof, for a fixed i, is by induction on the summation that builds the coefficient value and relies on the general properties of distributions given above. A second inductive proof over k allows us to use the properties we established for ring addition and multiplication and extend the result to vectors of polynomials. Note that the final expression does not use $\ominus$ at all, which is possible due to the propagation of the good property throughout the whole computation. □ □

The above result has the nice property that the distribution of all noise coefficients is the same. This means that one can use a union bound and derive that:

$$\begin{aligned}\Pr[\mathrm{COR}^1_{\text{ML-KEM}} : \|\tilde{n}_1\|_\infty > t] \;\leq\; & \\ & 256 \cdot \Pr[\, |c| > t \mid c \hookleftarrow \langle \mathcal{B}, \mathcal{B} \rangle_{256k} \oplus \langle \mathcal{B}, \mathcal{B} \rangle_{256k} \oplus \mathcal{B} \,] \\ & \text{where} \;\; t = \lfloor q/4 \rfloor - 1 - t^{\max}_{c_v} - t_{\mathbf{c}_u} \,. \end{aligned} \tag{1}$$

5.2.2 FrodoKEM.. The case of FrodoKEM is similar to the proof above, with the important caveat that we are working over two-dimensional structures. For our machine-checked proof this introduces additional complexity, so we developed a theory of distributions over matrices that permits seeing distributions over matrices as distributions over lists. Using this framework, we proved a result that is the analogue of the one presented above for the ML-KEM polynomial ring, but expressed over the ring of matrices used by FrodoKEM. Consider the expression for the noise matrix we obtained in Section 4:

$$\mathbf{N} := \mathbf{S}'\mathbf{E} - \mathbf{E}'\mathbf{S} + \mathbf{E}''$$

Here $\mathbf{S}'$ is a matrix of dimensions $\bar{n} \times n$, $\mathbf{E}$ has dimensions $n \times \bar{n}$, $\mathbf{E}'$ has dimensions $\bar{n} \times n$, $\mathbf{S}$ has dimensions $n \times \bar{n}$ and $\mathbf{E}''$ has dimensions $\bar{n} \times \bar{n}$.[12] We prove the following theorem:

Theorem 5.2. *The distribution of each coefficient of* $\mathbf{N}$ *is given by*

$$\langle \mathcal{X}, \mathcal{X} \rangle_n \ominus \langle \mathcal{X}, \mathcal{X} \rangle_n \oplus \mathcal{X}$$

The proof is similar in structure to the one presented for the ML-KEM expression, but conceptually simpler because matrix multiplication can be expressed directly using simple (rather than generalized) inner products. We cannot, however propagate the good property from the input distribution to the inner product distribution because we are not working over a field. For this reason the final expression of the noise distribution still uses $\ominus$.

Nevertheless, we still obtain the nice result that all noise coefficients are equally distributed, and so we can derive the following upper bound for the error probability:

$$\begin{aligned}&\Pr[\mathrm{COR}^{\text{provable}}_{\text{FrodoKEM}} : \exists ij, \neg(-q/2^{B+1} \leq \mathbf{N}[i,j] < q/2^{B+1})] \;\leq \\ &\bar{n}^2 \cdot \Pr[\, \neg(-q/2^{B+1} \leq c < q/2^{B+1}) \mid c \hookleftarrow \langle \mathcal{X}, \mathcal{X} \rangle_n \ominus \langle \mathcal{X}, \mathcal{X} \rangle_n \oplus \mathcal{X} \,]\end{aligned} \tag{2}$$

[12] In comparison to the general result shown in Section 4 that applies to a PKE based on LWE using arbitrary, yet consistent, matrix dimensions, we focus here on the concrete case of FrodoKEM where we have $n = m$ and $\bar{m} = \bar{n}$.

We note that both the results for ML-KEM and FrodoKEM are obtained generically, and so we can use them for different variants of each construction.

5.3 Dealing with rounding in ML-KEM

5.3.1 The provable bound. We now return to ML-KEM and assess the impact of rounding in the analysis. Let us begin with the isolated event associated with a rounding error in the ciphertext component $\mathbf{u}$, that we defined in Section 3 as:

$$\Pr[\mathrm{COR}^2_{\text{ML-KEM}} : \|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}] \quad \text{where} \quad \tilde{n}_2 := \langle \mathbf{s}, \mathbf{c}_u \rangle$$

The distribution of the error is defined as

$$\mathcal{D}_{\mathbf{c}_u} := \{\, \mathbf{c}_u \mid \mathbf{c}_u \leftarrow \mathrm{c\_decd}(\mathrm{c\_encd}(\mathbf{u})) - \mathbf{u}; \mathbf{u} \hookleftarrow \mathcal{U}(R^k_q) \}$$

For concreteness, when rounding a coefficient to 10 bits, as in ML-KEM-768, this distribution can be defined by the following frequency list

$$\{(-2, 128), (-1, 1024), (0, 1024), (1, 1024), (2, 129)\}$$

where the first element in each pair represents the element in $\mathbb{Z}_q$ and the second represents the number of occurrences. Probabilities can be obtained by dividing the second elements by 3329. When rounding to 11 bits, as in ML-KEM-1024 we get $\{(-1, 640), (0, 2048), (1, 641)\}$. So, these distributions have small support, but they do not satisfy the good definition due to the lack of symmetry.

The implication of this is that we cannot simplify the description of the distribution beyond the statement in the following theorem.

Theorem 5.3. *The distribution of coefficient i in $\tilde{n}_2$ is given by*

$$\langle \mathcal{B}, \mathcal{D}_{\mathbf{c}_u} \rangle_{k(i+1)} \ominus \langle \mathcal{B}, \mathcal{D}_{\mathbf{c}_u} \rangle_{k(255-i)}$$

The machine-checked proof is tedious, as it requires reasoning about the associativity of $\oplus$ and $\ominus$ when applied to general distributions. Using this property we first aggregate the terms which are added and those which are subtracted in each ring multiplication, and then aggregate them again across the inner products of vectors $\mathbf{s}$ and $\mathbf{c}_u$ of size k. The resulting probability distribution now depends on the coefficient index, which means that the computation of the bound cannot be simplified beyond the following summation over all coefficients:

$$\begin{aligned}&\Pr[\mathrm{COR}^2_{\text{ML-KEM}} : \|\tilde{n}_2\|_\infty > t_{\mathbf{c}_u}] \;\leq \\ &\textstyle\sum_{i=0}^{255} \Pr[\, |c| > t_{\mathbf{c}_u} \mid c \hookleftarrow \langle \mathcal{B}, \mathcal{D}_{\mathbf{c}_u} \rangle_{k(i+1)} \ominus \langle \mathcal{B}, \mathcal{D}_{\mathbf{c}_u} \rangle_{k(255-i)} \,]\end{aligned} \tag{3}$$

The above bound, combined with the one we obtained in the previous section and stated in Equation (1) allows us to obtain a provably secure correctness bound for ML-KEM under the MLWE assumption, as discussed in Section 3. However, even optimizing for the most favorable $t_{\mathbf{c}_u}$ this bound is significantly worse than the heuristic approximations that we will discuss next.

5.3.2 The heuristic bounds. The two heuristic bounds we consider are defined as follows.

$$\Pr[\mathrm{COR}^{\text{heuristic}}_{\text{ML-KEM}} : \|\tilde{n}\|_\infty > \lfloor q/4 \rfloor - 1] \tag{4}$$

$$\Pr[\mathrm{COR}^{1}_{\text{ML-KEM}} : \|\tilde{n}'\|_\infty > \lfloor q/4 \rfloor - 1 - t^{\max}_{c_v}] \tag{5}$$

$$\text{where } \tilde{n} := \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 + \mathbf{c}_u \rangle + e_2 + c_v$$

$$\tilde{n}' := \langle \mathbf{e}, \mathbf{r} \rangle - \langle \mathbf{s}, \mathbf{e}_1 + \mathbf{c}_u \rangle + e_2$$

The first one corresponds to assuming that both $\mathbf{c}_u$ and c_v result from rounding uniform (vectors of) ring elements, as is done in the original Kyber analysis [11]. In the second one we make the assumption only for $\mathbf{c}_u$ and max-out the c_v component. The formulas we obtain for the noise expressions are given by the following theorems.

Theorem 5.4. *The (heuristic approximation) of the distribution of coefficient i in $\tilde{n}$ is given by*

$$\langle \mathcal{B}, \mathcal{B} \rangle_{256k} \ominus (\langle \mathcal{B}, \mathcal{B} \oplus \mathcal{D}_{\mathbf{c}_u} \rangle_{k(i+1)} \ominus \langle \mathcal{B}, \mathcal{B} \oplus \mathcal{D}_{\mathbf{c}_u} \rangle_{k(255-i)}) \oplus \mathcal{B} \oplus \mathcal{D}_{c_v}$$

Theorem 5.5. *The (heuristic approximation) of the distribution of coefficient i in $\tilde{n}'$ is given by*

$$\langle \mathcal{B}, \mathcal{B} \rangle_{256k} \ominus (\langle \mathcal{B}, \mathcal{B} \oplus \mathcal{D}_{\mathbf{c}_u} \rangle_{k(i+1)} \ominus \langle \mathcal{B}, \mathcal{B} \oplus \mathcal{D}_{\mathbf{c}_u} \rangle_{k(255-i)}) \oplus \mathcal{B}$$

Since the distribution depends on the index of the coefficient, the overall upper bound is obtained by computing the probability for each i and summing all 256 values to obtain the union bound. In the next section we describe how we perform floating point computations that are guaranteed to provide an upper bound for the above mathematical quantities defined in this section. We will conclude the section and the technical part of the paper with our numeric results.

5.4 Computing the upper bounds.

Our formal development provides rigorous upper bounds for the statistical events defined in Section 3 and Section 4. This development relies on the explicit construction of discrete probability distributions and the computation of concrete upper bounds of some probability events defined on them. However, while EasyCrypt allows for exact computations in theory, performing them in practice within the tool is not possible.[13] To overcome this limitation, we have mirrored the constructive definitions of these distributions in OCaml, enabling practical computation of failure probabilities. We took care to keep the OCaml definitions syntactically as close as possible to their EasyCrypt counterparts to ensure correctness and maintain a strong link between the formal proofs and the numerical computations.

This can be observed in the development provided as supplementary material: the expression that describes how a probability distribution is constructed in EasyCrypt is easy to match to the expression that does this in OCaml. We deviated from a strict translation of the EasyCrypt computations only to introduce two optimizations that are yet unverified: computing the distribution of an n-fold summation of identically and independently distributed values by a double-and-add algorithm, and using memoization to avoid the repeated computation of some intermediate results. We plan to provide EasyCrypt proofs that these optimizations are correct in the future.

The OCaml code then emulates the required computations over the reals by using the MPFR library, using a precision of 500 bits and enforcing a rounding mode toward infinity to emulate the computations over the reals. This ensures that we always overapproximate the mass functions of the distributions, thereby guaranteeing that the computed failure probabilities serve as a valid upper bound.

One other potential direction for future improvements is developing a robust extraction mechanism for EasyCrypt that enables efficient computation outside its virtual machine while preserving formal guarantees. Alternatively, implementing a more efficient evaluator directly within EasyCrypt could make direct computations feasible without resorting to extraction to OCaml.

5.5 Results and Discussion

The results of our verified probability computations are given in Table 1. We show in blue the results that confirm the claims in the submissions to the NIST post-quantum competition. For FrodoKEM these correspond to Equation (2) and they are given in the **Provable** column—this is because we can formally connect them to the definition of cryptographic correctness required for security proofs of CCA security. For ML-KEM, the claims in the NIST submissions correspond to the heuristically simplified distributions (in column **Heur.** $\mathbf{c}_u$, c_v) captured by the bound in Equation (4), i.e., assuming that both rounding errors $\mathbf{c}_u$ and c_v result from rounding uniform elements.

The **Provable** bounds for ML-KEM have been computed as the summation of two probabilities given by Equation (1) and Equation (3). Recall that, in the analysis, we assign a threshold of $t_{\mathbf{c}_u}$ to the rounding error noise term, and a threshold of $\lfloor q/4 \rfloor - 1 - t^{\max}_{c_v} - t_{\mathbf{c}_u}$ to the noise terms unrelated to rounding. To obtain the final bound we tried all possible values of $t_{\mathbf{c}_u}$ and selected the thresholds that provided the best upper bounds. We illustrate the observed behavior in Figure 7. The optimal value for $t_{\mathbf{c}_u}$ is 296 in ML-KEM-768. This corresponds to partial error probabilities of 2^{-81} and 2^{-82}. The optimal value for $t_{\mathbf{c}_u}$ is 240 in ML-KEM-1024. This corresponds to partial error probabilities of 2^{-96} and 2^{-97}. Finally, we also report for ML-KEM the heuristic bound that results from max-ing out c_v and assuming only that $\mathbf{c}_u$ is computed over a uniform $\mathbf{u}$. This corresponds to Equation (5).

Remark. We emphasize that our search for a provable bound for ML-KEM is not motivated by a belief that previously claimed bounds are incorrect, but rather to emphasize that, so far and to the best of our knowledge, they have not been formally justified under MLWE in a way that is compatible with the definition of cryptographic correctness required for CCA security proofs. The intuition of such a proof would be that, if the simplification used to compute the claimed heuristic bounds was wrong, then one should be able to break MLWE. We prove that this is indeed the case, but only when the simplification is done separately on different noise terms, which has the cost of yielding a significantly worse

[13] Exact computations over the reals in general remain out of reach due to constraints such as the need for unbounded rational numbers. The particular examples that we handle in this paper might be within reach for a powerful machine with a well optimized implementation, but we decided to adopt a more pragmatic approach.

Table 1: Results of probability computations. Provable bounds: proved in EasyCrypt to apply to the cryptographic definition of correctness. Heuristic (Heur.) bounds: assume that the errors introduced by rounding one or both of the ciphertext elements in ML-KEM are distributed as if one rounded a uniform value. Values in blue confirm the claims in the submissions to the NIST competition.

Algorithm	Variant	Provable	Heur. $\mathbf{c}_u, c_v$	Heur. $\mathbf{c}_u$
ML-KEM	768	2^{-80}	2^{-164}	2^{-158}
ML-KEM	1024	2^{-95}	2^{-174}	2^{-169}
FrodoKEM	640	2^{-138}	-	-
FrodoKEM	976	2^{-199}	-	-
FrodoKEM	1344	2^{-252}	-	-

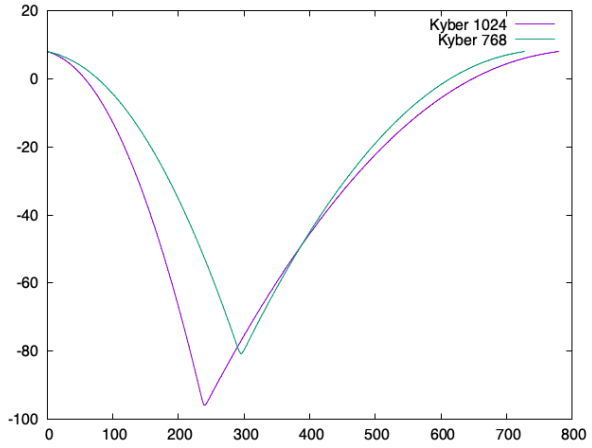

Figure 7: Behavior of the provable bound (y axis) computed for ML-KEM-768 & ML-KEM-1024 for varying values of $t_{\mathbf{c}_u}$ (x axis).

bound.[14] The take-away message is that to obtain stronger provably secure guarantees for the current parameters one needs better ways to estimate the probability of failure without relying on the MLWE assumption, i.e., without assuming that rounding is applied to uniform and independently distributed coefficients.

6 Conclusions and Future Work

Further Discussion. A natural question to ask is whether our results somehow formally verify the implementations of the Python scripts that were previously used to compute the correctness bounds we corroborate. Strictly speaking this is not the case, as there are several points in which the code we use for computations differs from the original Python scripts. For example, the original Python "cleans up" intermediate distributions by removing points with very low mass, and it performs composition of distribution in different ways than we do. We did not initially have an intuition on the potential impact of these differences. However, the fact that our code produces results that are close enough to the original scripts give us good indications that they are indeed correct.

In terms of technical challenges, the main hurdle we faced was in showing that the distribution combiners can be applied to ML-KEM, where noise expressions are computed via polynomial ring operations, by leveraging the cyclotomic structure of the ring. Indeed, deriving that all coefficients in the noise expression follow a distribution with a simple enough description that allows efficient computation was, to the best of our knowledge, never proved in a machine-checked setting.

Take-Away Messages. When we set out to do this work, our primary motivation was to unambiguously formalize the claims about correctness errors in ML-KEM, which is clearly the most practically relevant algorithm. We have achieved this: we have formalized the (simplified) distribution of noise that is used in the literature supporting ML-KEM, and we have a mechanized proof that the reported probability bounds for this distribution are correct. This is perhaps the most important result for practice in the immediate future.

Our secondary motivation was to clarify what the claimed bounds for ML-KEM mean from a provable security point of view. We do this in two ways: 1) we highlight the fact that the simplified distribution above is a heuristic approximation, i.e., that it is an assumption that currently underlies the security of ML-KEM (this has been already noted in the literature); and 2) we provide a worst-case scenario for removing this assumption by using MLWE to simplify the distribution in a way that is compatible to the provable security results for ML-KEM. The take-away message from these results is not that our worst-case bound is the correct one to use for parameter selection, but rather that further investigation is needed on how to bound the error probability without relying on MLWE.

To further clarify the area we decided to look at FrodoKEM for two reasons: 1) it is also being endorsed for practical uses by public entities and 2) its conservative design permits obtaining an efficiently computable bound for the failure probability that can be directly plugged into IND-CCA2 security proofs. The second point, we believe, highlights a tradeoff between optimization and provable security that in our opinion was not well understood in the past.

Future Work. There are many interesting directions for future work. Our current approach to connecting the EasyCrypt development to OCaml code requires human intervention (and validation), and so it is natural to consider either a fully automatic extraction mechanism, or an EasyCrypt extension that can perform such computations directly inside the tool. Our techniques should naturally extend to probability bounds computed explicitly for other lattice-based constructions and ML-DSA in particular. Also, we did not yet consider Saber because it seems not to have the same immediate practical relevance as ML-KEM and FrodoKEM; however, formally verifying the correctness bounds for this algorithm may also raise interesting questions on how to deal with conditional probabilities when simplifying the analyses of error distributions, as discussed in [17, 28]. A more exploratory direction is to consider concrete probability bounds claimed for other families of cryptographic primitives such as code-based and multivariate polynomial cryptography.

[14] Indeed, according to Figure 7, our approach to obtain a justification via MLWE cannot result in a better upper bound than the one reported in the **Provable** column in Table 1.

Acknowledgements

Most of the work was carried out while Manuel Barbosa was at MPI-SP and while Thing-han Lim was at Chelpis Quantum Corp. The research was supported by Deutsche Forschungsgemeinschaft (DFG, German research Foundation) as part of the Excellence Strategy of the German Federal and State Governments – EXC 2092 CASA - 390781972 and by the German Federal Ministry of Education and Research (BMBF) in the framework of the 6GEM research hub under grant number 16KISK038.

References

[1] Agence nationale de la sécurité des systèmes d'information (ANSSI). 2023. *ANSSI Views on the Post-Quantum Cryptography Transition: Follow-up Position Paper.* Technical Report. Agence nationale de la sécurité des systèmes d'information (ANSSI). https://cyber.gouv.fr/sites/default/files/document/follow_up_position_paper_on_post_quantum_cryptography.pdf Accessed: 2025-02-11.

[2] Erdem Alkim, Joppe Bos, Léo Ducas, Patrick Longa, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Chris Peikert, Ananth Raghunathan, and Douglas Stebila. 2021. FrodoKEM Learning With Errors Key Encapsulation Algorithm Specifications And Supporting Documentation. https://frodokem.org/files/FrodoKEM-specification-20210604.pdf Accessed: 2024-12-27.

[3] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. 2016. Post-quantum Key Exchange - A New Hope. 327–343.

[4] José Bacelar Almeida, Santiago Arranz Olmos, Manuel Barbosa, Gilles Barthe, François Dupressoir, Benjamin Grégoire, Vincent Laporte, Jean-Christophe Léchenet, Cameron Low, Tiago Oliveira, Hugo Pacheco, Miguel Quaresma, Peter Schwabe, and Pierre-Yves Strub. 2024. Formally Verifying Kyber - Episode V: Machine-Checked IND-CCA Security and Correctness of ML-KEM in EasyCrypt. 384–421. https://doi.org/10.1007/978-3-031-68379-4_12

[5] Nouri Alnahawi, Johannes Müller, Jan Oupický, and Alexander Wiesmaier. 2024. A Comprehensive Survey on Post-Quantum TLS. 1, 2 (2024), 6. https://doi.org/10.62056/ahee0iuc

[6] Gilles Barthe, Benjamin Grégoire, Sylvain Heraud, and Santiago Zanella Béguelin. 2011. Computer-Aided Security Proofs for the Working Cryptographer. 71–90. https://doi.org/10.1007/978-3-642-22792-9_5

[7] Daniel J. Bernstein, Leon Groot Bruinderink, Tanja Lange, and Lorenz Panny. 2018. HILA5 Pindakaas: On the CCA Security of Lattice-Based Encryption with Error Correction. 203–216. https://doi.org/10.1007/978-3-319-89339-6_12

[8] Daniel J. Bernstein, Billy Bob Brumley, Ming-Shing Chen, Chitchanok Chuengsatiansup, Tanja Lange, Adrian Marotzke, Bo-Yuan Peng, Nicola Tuveri, Christine van Vredendaal, and Bo-Yin Yang. 2020. *NTRU Prime.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[9] Nina Bindel and John M. Schanck. 2020. Decryption Failure Is More Likely After Success. 206–225. https://doi.org/10.1007/978-3-030-44223-1_12

[10] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. 2018. CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018.* IEEE, 353–367. https://doi.org/10.1109/EuroSP.2018.00032

[11] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. 2018. CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM. 353–367. https://doi.org/10.1109/EuroSP.2018.00032

[12] Sofía Celi, Armando Faz-Hernández, Nick Sullivan, Goutam Tamvada, Luke Valenta, Thom Wiggers, Bas Westerbaan, and Christopher A. Wood. 2021. Implementing and Measuring KEMTLS. 88–107. https://doi.org/10.1007/978-3-030-88238-9_5

[13] Jan-Pieter D'Anvers and Senne Batsleer. 2022. Multitarget Decryption Failure Attacks and Their Application to Saber and Kyber. 3–33. https://doi.org/10.1007/978-3-030-97121-2_1

[14] Jan-Pieter D'Anvers, Qian Guo, Thomas Johansson, Alexander Nilsson, Frederik Vercauteren, and Ingrid Verbauwhede. 2019. Decryption Failure Attacks on IND-CCA Secure Lattice-Based Schemes. 565–598. https://doi.org/10.1007/978-3-030-17259-6_19

[15] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, Jose Maria Bermudo Mera, Michiel Van Beirendonck, and Andrea Basso. 2020. *SABER.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[16] Jan-Pieter D'Anvers, Mélissa Rossi, and Fernando Virdia. 2020. (One) Failure Is Not an Option: Bootstrapping the Search for Failures in Lattice-Based Encryption Schemes. 3–33. https://doi.org/10.1007/978-3-030-45727-3_1

[17] Jan-Pieter D'Anvers, Angshuman Karmakar Sujoy Sinha Roy, and Frederik Vercauteren. 2018. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. Cryptology ePrint Archive, Report 2018/230. https://eprint.iacr.org/2018/230

[18] Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede. 2018. The impact of error dependencies on Ring/Mod-LWE/LWR based schemes. Cryptology ePrint Archive, Report 2018/1172. https://eprint.iacr.org/2018/1172

[19] Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede. 2018. On the impact of decryption failures on the security of LWE/LWR based schemes. Cryptology ePrint Archive, Report 2018/1089. https://eprint.iacr.org/2018/1089

[20] Jintai Ding, Saed Alsayigh, R V Saraswathy, Scott Fluhrer, and Xiaodong Lin. 2017. Leakage of signal function with reused keys in RLWE key exchange. In *2017 IEEE International Conference on Communications (ICC).* 1–6. https://doi.org/10.1109/ICC.2017.7996806

[21] Jintai Ding, Scott R. Fluhrer, and Saraswathy RV. 2018. Complete Attack on RLWE Key Exchange with Reused Keys, Without Signal Leakage. 467–486. https://doi.org/10.1007/978-3-319-93638-3_27

[22] Scott Fluhrer. 2016. Cryptanalysis of ring-LWE based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085. https://eprint.iacr.org/2016/085

[23] Federal Office for Information Security (BSI). 2024. Cryptographic Mechanisms: Recommendations and Key Lengths. Technical Guideline. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf.

[24] FrodoKEM Team. 2024. FrodoKEM: Practical Quantum-Secure Key Encapsulation from Generic Lattices. https://frodokem.org/ Accessed: 2024-12-27, see the "News" section..

[25] Eiichiro Fujisaki and Tatsuaki Okamoto. 1999. Secure Integration of Asymmetric and Symmetric Encryption Schemes. 537–554. https://doi.org/10.1007/3-540-48405-1_34

[26] Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. 2022. Failing Gracefully: Decryption Failures and the Fujisaki-Okamoto Transform. 414–443. https://doi.org/10.1007/978-3-031-22972-5_15

[27] Andreas Hülsing, Matthias Meijers, and Pierre-Yves Strub. 2022. Formal Verification of Saber's Public-Key Encryption Scheme in EasyCrypt. 622–653. https://doi.org/10.1007/978-3-031-15802-5_22

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

[29] Katharina Kreuzer. 2024. Verification of Correctness and Security Properties for CRYSTALS-KYBER. In *37th IEEE Computer Security Foundations Symposium, CSF 2024.* IEEE, 511–526. https://doi.org/10.1109/CSF61375.2024.00016

[30] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. 2020. *FrodoKEM.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[31] National Institute of Standards and Technology. 2024. FIPS PUB 203 – ML-KEM: Module-Lattice-Based Key-Encapsulation Mechanism Standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf.

[32] National Institute of Standards and Technology. 2024. FIPS PUB 204 – ML-DSA: Module-Lattice-Based Digital Signature Standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf.

[33] National Institute of Standards and Technology. 2024. FIPS PUB 205 – SLH-DSA: Stateless Hash-Based Digital Signature Standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf.

[34] NIST 2015. FIPS PUB 202 – SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf.

[35] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehlé. 2020. *CRYSTALS-KYBER.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[36] Tianrui Wang, Anyu Wang, and Xiaoyun Wang. 2023. Exploring Decryption Failures of BIKE: New Class of Weak Keys and Key Recovery Attacks. 70–100. https://doi.org/10.1007/978-3-031-38548-3_3