APPLICATION OF THE MLWE IN THE CRYPTOSYSTEM CRYSTALS – KYBER AS KEM AND DILITHIUM AS SIGNATURE

RAZAFIMAHEFA Fenofahasoavana Ombana[1], RANDRIAMITANTSOA Paul August[2], RANDRIANANDRASANA Marie Emile[3]

[1] PhD Student, TASI, ED-STII, Antananarivo, Madagascar
[2] Thesis Director, TASI, ED-STII, Antananarivo, Madagascar
[3] Thesis Codirector, TASI, ED-STII, Antananarivo, Madagascar

ABSTRACT

Quantum computers, but still with a low number of qubits, are starting to emerge as we write. It is more crucial than ever to work on new cryptosystems. Cryptosystems that will be based on mathematical problems that are robust enough to withstand the increased computational power that these new types of computers will provide. In the course of our research, we came across the mathematical object that is the Lattice. The latter is one of the promising candidates in terms of fundamental problem for post-quantum cryptosystems. But the mathematical tool on which we will open this article is the MLWE or Module Learning With Error. The LWE (or Learning With Error), in the first instance, is a generalization of the problem Learning from parity with error. It has been shown that this problem is equivalent, in terms of difficulty, to the SIVP or Shortest Independent Vector Problem of a lattice. Coming back to MLWE, we have another designation, Learning With Error other Module Lattices. Module lattices can be considered as lattices between those used in the LWE problem definitions and those used for the R-LWE problem (Ring LWE). The Module-LWE offers a compromise between the two extremes of the LWE and the R-LWE. Thereafter we will focus on the CRYSTALS cryptosystem, a cryptosystem based on the MLWE. Our choice to study this cryptosystem is based on the fact that it was submitted to the NIST (National Institute of Standards and Technology) project of post-quantum cryptosystem standardization. CRYSTALS was able to progress to step 3 of the project. But above all, it is the only one among the few that have progressed to this final stage, to have both a KEM type cryptosystem (Key Encapsulation Mechanism) and another signature type one.

Keyword: *Quantum Computer, Post Quantum Cryptosystem, Module LWE, CRYSTALS-Kyber, CRYSTALS-Dilithium, Key Encapsulation Mechanism (KEM), Signature.*

---

1. The MLWE

Most lattice-based cryptographic schemes are built on the assumed difficulty of Short Integer Solution (SIS) and Leaning With Error (LWE). Their effectiveness can be improved by shifting the difficulty assumptions to the more compact Ring-SIS and Ring-LWE problems. However, this change in difficulty assumptions is accompanied by a weakening of security: SIS and LWE are known to be at least as difficult as standard (worst case) problems on Euclidean lattices (classical version of lattices), while Ring-SIS and Ring-LWE have the same level of difficulty only for their restrictions to special classes of ideal lattices, corresponding to the ideal of some polynomial rings. [7]

The present subtitle of our work consists of the definition of Module-SIS and, especially, Module-LWE problems, which link SIS with Ring-SIS, and LWE with Ring-LWE, respectively. These average case problems are at least as difficult as the standard lattice problems restricted to module lattices (the latter, module lattices, in turn, serve as a middle ground between arbitrary lattices and ideal lattices). It is important to note that the worst case to average case reductions for modulus problems are strong, in the sense that there are inverse reductions. This property is not known to hold in the context of Ring-SIS / Ring-LWE: problems with ideal lattices could be easy without affecting the difficulty of Ring-SIS / Ring-LWE.

1.1. LWE variants

1.1.1. LWE

Let $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ denote the segment $[0,1)$ with the addition operation modulo 1. For a probability density function χ on $\mathbb{T}$ and a vector $s \in \mathbb{Z}_q^n$, we posit $A_{s,\chi}$ the distribution on $\mathbb{Z}_q^n \times \mathbb{T}$ obtained by choosing a vector $a \in \mathbb{Z}_q^n$ uniformly and in a random fashion, then $e \in \mathbb{T}$ by respect to χ, and returning $\left(a, \frac{1}{q}\langle a, s\rangle + e\right)$.

Definition 01:

The search version of the Learning With Error problem $\mathrm{SLWE}_{q,\chi}$ is as follows: let $s \in \mathbb{Z}_q^n$ be a secret; given several samples from $A_{s,\chi}$, the goal is to find s.

The decision version of the Learning With Error problem $\mathrm{LWE}_{q,\chi}$ is as follows: let $s \in \mathbb{Z}_q^n$ be chosen in a uniformly random fashion; the goal is to distinguish between several independent samples from $A_{s,\chi}$ and the same number of independent samples from $U(\mathbb{Z}_q^n \times \mathbb{T})$. Where U refers to a uniform distribution, on the set $\mathbb{Z}_q^n \times \mathbb{T}$ in this case therefore.

It is also possible to interpret the LWE in terms of linear algebra: Suppose the number of samples required $\left(a_i, \frac{1}{q}\langle a_i, s\rangle + e_i\right)$ from $A_{s,\chi}$ is m, then we consider the matrix $A \in \mathbb{Z}_q^{m\times n}$ whose rows are the a_i. And we will consider the vector $e = (e_1, \dots, e_m)^T$. Then the SLWE is as follows: [3], [4]

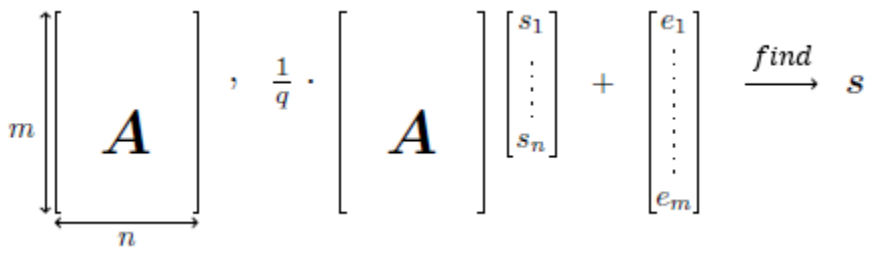

Figure 01: The LWE in terms of linear algebra.

Theorem 01:

Let $\varepsilon(n) = n^{-\omega(1)}$, $\alpha \in (0,1)$ and $q \geq 2$ such that $\alpha q \geq 2\sqrt{n}$. There is a quantum reduction from solving $\mathrm{GIVP}_{\sqrt{8}n/\alpha}^{\eta_\varepsilon}$ in polynomial time (in the worst case, with high probability) to solving $\mathrm{SLWE}_{q,D_\alpha}$ in polynomial time with nontrivial probability.

Assuming q is prime, $q \leq \mathrm{poly}(n)$, and χ is a probability density function on $\mathbb{T}$. Then there exists a reduction in polynomial time from $\mathrm{SLWE}_{q,\chi}$ to $\mathrm{LWE}_{q,\chi}$.

The following theorem overrides the condition that q must be prime.

Theorem 02:

Let $\alpha > 0,\ \varepsilon \in (0, 1/2),\ m \geq n \geq 1,\ p \geq 25$ and $q \in [p, 2p)$. There exists a polynomial-time reduction from $\mathrm{LWE}_{q,\alpha}$ to $\mathrm{LWE}_{p,\Psi_{\leq\beta}}$ where $\beta = C\alpha\sqrt{n}\sqrt{\log(n/\varepsilon)\log(nm/\varepsilon)}$, for a positive constant C, and which loses at most ε in advantage.

1.1.2. R-LWE

Let ψ be a distribution on $\mathbb{T}_{R^\vee} = K_{\mathbb{R}}/R^\vee$ and $s \in R_q^\vee$. We denote by $A_{s,\psi}^{(R)}$ the distribution over $R_q \times \mathbb{T}_{R^\vee}$ obtained by choosing $a \in R_q$ in a uniformly random manner and $e \in \mathbb{T}_{R^\vee}$ with respect to ψ, and returning $(a, (a.s)/q + e)$.

Definition 02:

Let $q \geq 2$ and Ψ be a family of distributions on $\mathbb{T}_{R^\vee}$. The search version of the Ring Learning With Error problem $\text{R-SLWE}_{q,\Psi}$ is as follows: let $s \in R_q^\vee$ be secret, and $\psi \in \Psi$; given several samples from $A_{s,\psi}^{(R)}$, the goal is to find s.

Let Υ be a distribution over a family of noise distributions on $\mathbb{T}_{R^\vee}$. The decision version of the Ring Learning With Error problem $\text{R-SLWE}_{q,\Upsilon}$ is as follows: suppose $s \in R_q^\vee$ is uniformly random, and ψ sampled from Υ ; the goal is to distinguish between multiple independent samples from $A_{s,\psi}^{(R)}$ and the same number of independent samples from $U\left(R_q, \mathbb{T}_{R^\vee}\right)$.

Theorem 03:

Let $\varepsilon(n) = n^{-\omega(1)}$, $\alpha \in (0,1)$ and $q \geq 2$ of a known factorization such that $\alpha q > \omega\left(\sqrt{\log n}\right)$. There is a quantum reduction from solving the $\text{Id-GIVP}_\gamma^{\eta_\varepsilon}$ in polynomial time (in the worst case, with high probability) to solving the $\text{R-SLWE}_{q,\Psi_{\leq\alpha}}$ in polynomial time with nontrivial probability with $\gamma = \sqrt{n}.\,\omega\left(\sqrt{\log n}\right)/\alpha$.

Assuming q is prime, $q \leq \mathrm{poly}(n)$, and $q = 1 \bmod v$. Then there exists a polynomial time reduction from $\text{R-SLWE}_{q,\Psi_{\leq\alpha}}$ to $\text{R-LWE}_{q,\Upsilon_\alpha}$.

1.1.3. LWE over modules

The M-LWE generalizes the LWE and the R-LWE. The variable n and d denote the dimension of R and the rank of the $M \subseteq R^d$ module, respectively. We pose $N = nd$ the dimension of the lattice modulus.

Let ψ be a probability distribution on $\mathbb{T}_{R^\vee}$ and $s \in \left(R_q^\vee\right)^d$ a vector. We define by $A_{q,s,\psi}^{(M)}$ the distribution on $\left(R_q\right)^d \times \mathbb{T}_{R^\vee}$ obtained by choosing a vector $a \in \left(R_q\right)^d$ uniformly random, and $e \in \mathbb{T}_{R^\vee}$ by respect to ψ, and returning $\left(a, \frac{1}{q}\langle a, s\rangle + e\right)$.

Definition 03:

Let $q \geq 2$ and Ψ be a family of distributions on $\mathbb{T}_{R^\vee}$. The search version of the Module Learning With Error problem $\text{M-SLWE}_{q,\Psi}$ is as follows: Let $s \in \left(R_q^\vee\right)^d$ be a secret vector and $\psi \in \Psi$; given arbitrarily many samples from $A_{q,s,\psi}^{(M)}$, the goal is to find s.

For an integer $q \geq 2$ and a distribution Υ over a family of distributions on $K_{\mathbb{R}}$. The decision version of the Module Learning With Error problem $\text{M-SLWE}_{q,\Upsilon}$ is as follows: Let $s \in \left(R_q^\vee\right)^d$ be uniformly random and ψ sampled over Υ; the goal is to distinguish between arbitrarily many random samples from $A_{q,s,\psi}^{(M)}$ and the same number of samples from $U\left(R_q^d \times \mathbb{T}_{R^\vee}\right)$.

As was the case with the LWE and R-LWE, the M-LWE problem can be interpreted in terms of linear algebra. It (the M-LWE) consists of taking the LWE matrix A of the form:

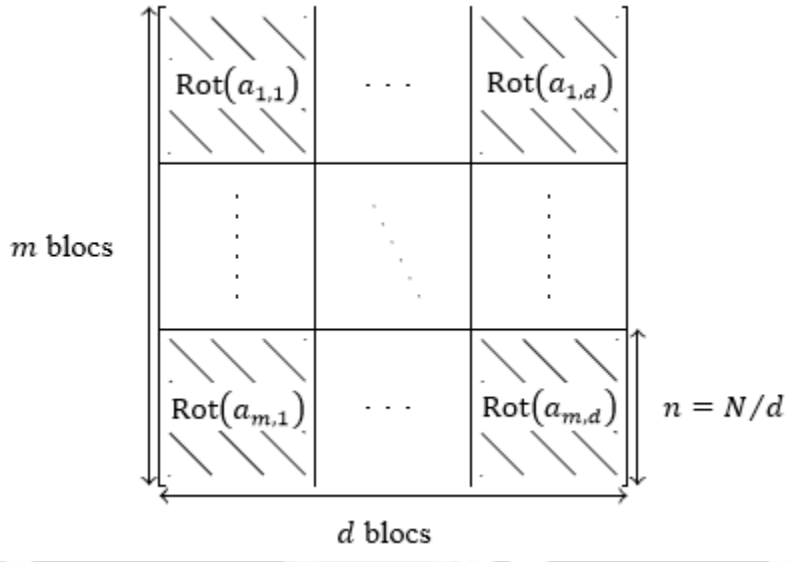

Figure 02: The M-LWE problem can be interpreted in terms of linear algebra.

The difficulty of M-LWE is supported by the two theorems that follow.

Theorem 04:

Let $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ and $q \geq 2$ of a known factorization such that $\alpha q > 2\sqrt{d}.\omega\left(\sqrt{\log n}\right)$. There is a quantum reduction from solving Mod-GIVP$_{\gamma}^{\eta_\varepsilon}$ in polynomial time (in the worst case, with high probability) to solving M-SLWE$_{q,\Psi_{\leq\alpha}}$ in polynomial time with a nontrivial advantage with $\gamma = \sqrt{8Nd}.\omega\left(\sqrt{\log n}\right)/\alpha$.

Assuming q is prime, $q \leq \text{poly}(N)$ and $q = 1 \bmod v$. Then there exists a polynomial time reduction from M-SLWE$_{q,\Psi_{\leq\alpha}}$ to M-LWE$_{q,\Upsilon_\alpha}$.

When $n = N$ and $d = 1$, Theorem 04 is equivalent to Theorem 03. When $n = 1$ and $d = N$, it (Theorem 04, again) is equivalent to Theorem 01.

Theorem 05:

Let $p, q \in \left[2, 2^{N^{O(1)}}\right]$ and $\alpha, \beta \in (0,1)$ such that $\beta \geq \alpha.\max\left(1, \frac{q}{p}\right).n^{1/4}N^{1/2}.\omega(\log^2 N)$ and $\alpha q \geq \omega\sqrt{\log(N)/n}$. There is a polynomial time reduction from M-LWE$_{q,\Upsilon_\alpha}$ to M-LWE$_{q,\Upsilon_\beta}$.

1.2. Reduction of Mod-GIVP to M-SLWE

The reduction from Mod-GIVP to M-SLWE uses the following intermediate problem, where ϕ denotes a real-valued function on a lattice and γ depends on the dimension, called the *Module Discrete Gaussian Sampling Problem* M-DGS$_{\gamma}^{\phi}$: given a module lattice M of dimension N, and a number $r > \gamma \cdot \phi$, the goal is to return a sample from $D_{M,r}$. The reduction is done in 2 (two) steps:

$$\text{Mod-GIVP}_{\sqrt{8Nd}\cdot\omega\left(\sqrt{\log n}\right)/\alpha}^{\eta_\varepsilon} \xrightarrow{\textbf{Lemma 01}} \text{M-DGS}_{\sqrt{2d}\cdot\omega\left(\sqrt{\log n}\right)/\alpha}^{\eta_\varepsilon} \xrightarrow{\textbf{Lemma 02}} \text{M-SLWE}_{q,\Psi_{\leq\alpha}}$$

Lemma 01:

For any $\varepsilon = \varepsilon(N) \leq 1/10$ and any γ and ϕ such that $\gamma \cdot \phi(M) \leq \sqrt{2}\eta_\varepsilon(M)$, there exists a polynomial-time reduction of the Mod-GIVP$^{\phi}_{2\sqrt{N}\cdot\gamma}$ to M-DGS$^{\phi}_{\gamma}$. [3]

Lemma 02:

Let $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ and q be an integer such that $\alpha q \geq 2\sqrt{d} \cdot \omega(\sqrt{\log n})$. Assuming we have access to an oracle that solves M-SLWE$_{q,\Psi_{\leq\alpha}}$, given a polynomial number of samples. Then there exists a polynomial-time quantum algorithm for M-DGS$^{\eta_\varepsilon}_{\sqrt{2d}\cdot\omega(\sqrt{\log n})/\alpha}$.

Lemma 03:

Let $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ and $q \geq 2$. Assume that we have access to an oracle that solves M-SLWE$_{q,\Psi_{\leq\alpha}}$ in polynomial time with nontrivial probability. Then there exists a polynomial-time quantum algorithm that, given an N -dimensional lattice modulus M, a number $r > 2q\eta_\varepsilon(M)$ and poly(N) samples from $D_{M,r}$, produces a sample from $D_{M,\frac{r\sqrt{d}\cdot\omega(\sqrt{\log n})}{\alpha q}}$ with nontrivial probability.

To prove **Lemma 03** we use an intermediate problem, Mod-BDD$_\delta$ (BDD for *Bounded Distance Decoding*): given a lattice modulus M, $\delta < \lambda_1(M)/2$ and any point $y \in \mathbb{R}^n$ of the form $y = x + e$ for some $x \in M$ and $\|e\|_{2,\infty} \leq \delta$, find x. Note that the norm $\ell_{2,\infty}$ is used here, because it will help us state **Lemma 06** more appropriately.

Another intermediate problem q-Mod-BDD$_\delta$ will be used: given a lattice modulus M and a point $y \in \mathbb{R}^n$ in the distance (by respect to the norm $\ell_{2,\infty}$) δ from M, give as output the coset in M/qM of the nearest vector y. The proof of **Lemma 03** consists of the reduction sequence below (note that δ here takes the value $\frac{\alpha q\cdot\omega(\sqrt{\log n})}{\sqrt{2n}r}$).

Echantillons à partir de $D_{M,\frac{r\sqrt{d}\cdot\omega(\sqrt{\log n})}{\alpha q}}$ $\xrightarrow[\text{(quantique)}]{\textbf{Lemma 04}}$ Mod-BDD$_{M^\vee,\delta}$ $\xrightarrow{\textbf{Lemma 05}}$ q-Mod-BDD$_{M^\vee,\delta}$ $\xrightarrow{\textbf{Lemma 06}}$ M-SLWE$_{q,D_\alpha}$ Echantillon à partir de $D_{M,r}$

Lemma 04:

There exists an efficient quantum algorithm that, given an N-dimensional lattice modulus M, a number $\delta < \lambda_1(M^\vee)\omega(\sqrt{\log n})/(2\sqrt{n})$, and an oracle that solves Mod-BDD$_\delta$ on $M^\vee$, yields at the output samples from $D_{M,r\sqrt{d}\cdot\omega(\sqrt{\log n})/(\sqrt{2}\delta)}$.

Lemma 05:

For any $q \geq 2$, there exists a polynomial-time reduction from Mod-BDD$_\delta$ to q-Mod-BDD$_\delta$.

Lemma 06:
Let $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ and $q \geq 2$. Let $M \subseteq R^d$ be an R-module, and $r > \sqrt{2}q \cdot \eta_\varepsilon(M)$. Given an access to an oracle sampler from the distribution $D_{M,r}$, there is a probabilistic reduction from q-Mod-BDD$_{M^\vee,\frac{\alpha q\cdot\omega(\sqrt{\log n})}{\sqrt{2n}r}}$ to M-SLWE$_{q,\Psi_{\leq\alpha}}$.

The principle of the reduction is to construct from y, the input of q-Mod-BDD, and from some continuous or discrete Gaussian samples, the (a,b) peers distributed according to $A^{(M)}_{q,s,\psi}$, where s will directly depend on the x vector closest to y. To produce such samples (a,b) according to the desired distribution, we combine the

corresponding proofs for LWE and R-LWE. Then call the M-SLWE oracle to return s and allow us to recover the information about x.

Lemma 07:

Let $\varepsilon > 0$ and $s = \Theta(x \bmod qM^{\vee})$. There exists $\psi \in \Psi_{\leq\alpha}$ such that the statistical distance between $A_{q,s,\psi}^{(M)}$ and the distribution of (a, b) is at most 6ε.

All this concludes the statement of the first part of our Theorem 05.

For the second part of the theorem (existence of a reduction from M-SLWE to M-LWE), we invite our readers to consult the proof in our bibliography (paragraph 4.3 in [3]).

2. The cryptosystem CRYSTALS

The CRYSTALS or Cryptographic Suite for Algebraic Lattices encapsulates two types of cryptosystems: Kyber, a KEM (Key Encapsulation Mechanism, to recall) having the specificity of being classified as IND-CCA2 secure, and Dilithium a EUF-CMA secure signature algorithm. Both algorithms are based on difficult problems related to lattice modules, and were therefore built to resist possible attacks from quantum computers.

CRYSTALS, as mentioned in the introduction to this book, was submitted to the NIST Post-Quantum Cryptography Project. It was able to progress to stage 3 of the said standardization project. But above all, it is the only one among the few that have progressed to this final stage, to have both a KEM and a signature type cryptosystem. [5], [6], [8]

2.1. CRYSTALS-Kyber, the KEM type cryptosystem

2.1.1. Performance Analysis

In this section, we discuss the implementation aspects of Kyber and report performance results for two implementations: the *ANSI C* (ANSI for *American National Standards Institute*) reference implementation and an optimized implementation using *AVX2* vector instructions.

2.1.1.1. Kyber performance on Intel Haswell processors

Table 01 shows the Intel Haswell performance results of the reference implementation and an optimized Kyber AVX2 implementation and Kyber variant 90s as well as the key and ciphertext sizes. All benchmarks were obtained on a core of an Intel Core i7-4770K (Haswell) processor clocked at 3492 MHz (as indicated by /proc/cpuinfo) with TurboBoost and hyperthreading disabled. The machine used has 32 GB RAM (for *Random Access Memory*) and runs Debian GNU/Linux (GNU from *GNU's Not Unix*) with version 4.19.0 of the Linux kernel. Both implementations were compiled with gcc version 8.3.0. All cycle counts shown are the median of the cycle counts of 10000 executions of the respective function. The implementations are not optimized for memory usage, but in general, Kyber has very modest memory requirements.

Table 01 : Performance of Kyber on Intel Haswell processors

Kyber512					
Sizes (in Bytes)		Hashwell Cycles (ref)		Hashwell Cycles (AVX2)	
sk :	1632 (ou 32)	gen :	122684	gen :	33856
pk :	800	enc :	154524	enc :	45200
ct :	768	dec :	187960 (ou ≈ 288912)	dec :	34572 (ou ≈ 59088)
Kyber512 – 90s					
Sizes (in Bytes)		Hashwell Cycles (ref)		Hashwell Cycles (AVX2)	
sk :	1632 (ou 32)	gen :	213156	gen :	21880

pk :	800	enc :	249084	enc :	28592
ct :	768	dec :	277612 (ou ≈ 405268)	dec :	20980 (ou ≈ 38752)
Kyber768					
Sizes (in Bytes)		Hashwell Cycles (ref)		Hashwell Cycles (AVX2)	
sk :	2400 (ou 32)	gen :	199408	gen :	52732
pk :	1184	enc :	235260	enc :	67624
ct :	1088	dec :	274900 (ou ≈ 425492)	dec :	53156 (ou ≈ 82220)
Kyber768 – 90s					
Sizes (in Bytes)		Hashwell Cycles (ref)		Hashwell Cycles (AVX2)	
sk :	2400 (ou 32)	gen :	389760	gen :	30460
pk :	1184	enc :	432764	enc :	40140
ct :	1088	dec :	473984 (ou ≈ 671864)	dec :	301108 (ou ≈ 52512)
Kyber1024					
Sizes (in Bytes)		Hashwell Cycles (ref)		Hashwell Cycles (AVX2)	
sk :	3168 (ou 32)	gen :	307148	gen :	73544
pk :	1568	enc :	346648	enc :	97324
ct :	1568	dec :	396584 (ou ≈ 617848)	dec :	79128 (ou ≈ 115332)
Kyber1024 – 90s					
Sizes (in Bytes)		Hashwell Cycles (ref)		Hashwell Cycles (AVX2)	
sk :	3168 (ou 32)	gen :	636380	gen :	43212
pk :	1568	enc :	672644	enc :	56566
ct :	1568	dec :	724144 (ou ≈ 1009448)	dec :	44328 (ou ≈ 71180)

In addition, *ref* refers to the C reference implementation, AVX2 to the implementation using AVX2 vector instructions; sk stands for secret key, pk for public key and ct for ciphertext. In parentheses are approximate values when including key generation in decapsulation to avoid having to store expanded secret keys. In this scenario, we store only the initial seed d at the very beginning of the key generation algorithm $\mathrm{Kyber.CPAPKE.KeyGen}()$. The approximate cycle counts for this scenario are calculated as the sum of the cycle counts for standard decapsulation and key generation minus the number of cycles required to generate the public seed ρ matrix **A**. Note that this is a very conservative estimate; actual implementations of the approach may also save, for example, sampling the 32 bytes of randomness.

2.1.1.2. Performances de Kyber sur les processeurs ARM Cortex-M4

Table 02 reports the cycle count and RAM usage of a C implementation (clean) and an optimized implementation (m4) of Kyber on an ARM Cortex-M4. All benchmarks are obtained using the pqm4 framework on a Discovery STM32F407 board. We do not report here benchmarks for the 90s version of Kyber on the Cortex-M4, as pqm4 does not yet include a constant-time implementation of AES.

Table 02: Performance of Kyber on ARM Cortex-M4 processors

Kyber512				
	M4 Cycles (clean)	M4 RAM (clean)	M4 Cycles (m4)	M4 RAM (m4)
gen :	655595	6020	463068	2844
enc :	865256	8668	561518	2484
dec :	961648	9444	519237	2508
Kyber768				
	M4 Cycles (clean)	M4 RAM (clean)	M4 Cycles (m4)	M4 RAM (m4)
gen :	1087897	10052	756224	3292

enc :	1373744	13212	915676	2980
dec :	1491214	14308	853001	3004
Kyber1024				
	M4 Cycles (clean)	M4 RAM (clean)	M4 Cycles (m4)	M4 RAM (m4)
gen :	1696314	15180	1213303	3804
enc :	2057522	18844	1407769	3492
dec :	2199958	20420	1326409	3516

2.1.2. Security level estimation

In this paragraph, we have mainly relied on our bibliographies [1], [10] and [11].

Table 03 lists the security levels as defined in Section 4.A.5 of the NIST call for proposals [12] for the different parameter sets of Kyber. Our assertions are based on the cost estimates of the most well-known attacks against the MLWE problem underlying Kyber, as detailed in subsection 5.1 of our bibliography [1]. More precisely, the classical and quantum difficulty of the kernel-SVP are listed and used to derive the security levels.

All Kyber parameter sets have a certain probability of decryption failure. These failures are a security problem (see Section 5.5 of [1]) and thus the probabilities with which they occur must be small. But since in classical ROM the probability of decryption failure is information theoretic, we see no need to decrease it with the security parameter. In particular, the decryption failure for our level 3 and 5 parameter sets is less than 2^{-160}, which means that if 280 instances of Kyber were executed every second from now until our sun becomes a white dwarf, the odds are still strongly in favor of there never being a decryption failure. We therefore exclude these attacks from our claims about NIST security estimates.

The impact of deterministic noise caused by $\mathrm{Compress_q}$ *on Kyber512*. Each coefficient of $\mathbf{e_1}$ (and $\mathbf{e_2}$) in the Kyber. CPAPKE encryption algorithm behaves like a binomial distribution with parameter $\eta_2 = 2$, with variance $\eta_2/2 = 1$. The parameter $d_u = 10$ implies that the $\mathrm{Compress}_q$ function maps the elements modulo q to a set of size 2^{10} where the difference between all the even numbers of elements in said set is 3 or 4. This implies that the error added by the $\mathrm{Compress}_q$ function for each coefficient is either uniform over$\{-1,0,1\}$, $\{-1,0,1,2\}$ or $\{-2,-1,0,1\}$. It therefore has a variance at least as large as the uniform distribution on the set $\{-1,0,1\}$, or $2/3$. This makes the total variance of each coefficient of $\mathbf{e_1}$ plus the deterministic error at least $1 + 2/3 = 5/3$. The remaining secret and noise terms have binomial distributions with parameter $\eta_1 = 3$ for a variance of $\eta_1/2 = 3/2 < 5/3$. Taking into account the errors added by $\mathrm{Compress}_q$, we therefore compute the Kyber512 difficulty by assuming that the variance of each secret/noise coefficient is $3/2$.

Table 03: Classical and quantum difficulty of the different parameter sets proposed for Kyber.

	Kyber512	Kyber768	Kyber1024
NIST security Level	1	3	5

Core-SVP methodology, Primal attack only			
Lattice attack dim. d	1003	1424	1885
BKZ-blocksize β (BKZ for Block Korkine Zolotarev)	403	625	877
Core-SVP classical hardness	118	182	256
Core-SVP quantum hardness	107	165	232

Refined estimate for classical attacks using			
Lattice attack dim. d	1025	1467	1918
BKZ-blocksize β	413	637	894

Sieving dimension $\beta' = \beta - d_{4f}$	375	586	829
$\log_2$(gates)	151.5	215.1	287.3
$\log_2$(memory in bits)	93.8	138.5	189.7

MAXDEPTH impact. The best-known quantum accelerations for the sifting algorithm, which are more detailed in the cost analysis (refer to subsection 5.1.1 of [1]), are only slightly affected by the depth limitation of a quantum circuit, as it uses Grover search on small sets (compared to search the entire AES key space). For the operating estimates of the core-SVP difficulty to match the quantum gate cost of breaking AES at the respective security levels, a quantum computer would have to support a maximum depth of 70 to 80. When limiting the maximum depth to smaller values, or when considering classical attacks, the estimates of the kernel-SVP difficulty are smaller than the number of gates for attacks against AES. A recent study of the concrete cost of sieving suggests that the quantum accelerations of these algorithms are tenuous, regardless of the value of MAXDEPTH.

2.2. CRYSTALS-Dilithium, the signature type cryptosystem

2.2.1. Security

The security of our signature algorithm can be proved through ROM, based on the difficulty of two problems [2], [9]. The first is the standard LWE problem (on polynomial rings) which requires distinguishing $(\mathbf{A}, \mathbf{t} \coloneqq \mathbf{As_1} + \mathbf{s_2})$ from $(\mathbf{A}, \mathbf{u})$, where $\mathbf{u}$ is uniformly random. The other problem is what has been called the SelfTargetMSIS problem, which is the problem of finding a vector $\begin{bmatrix} \mathbf{z} \\ c \\ \mathrm{v} \end{bmatrix}$ with small coefficients and a satisfying message μ :

$$\mathrm{H}\left(\mu || [\mathbf{A}|\boldsymbol{t}|\mathbf{I}] \cdot \begin{bmatrix} \mathbf{z} \\ c \\ \mathrm{v} \end{bmatrix}\right) = c \qquad \textbf{(01)}$$

where $\mathbf{A}$ and t are uniformly random and $\mathbf{I}$ is the identity matrix. In ROM, one can obtain a (non-narrow) reduction using the forking lemma of the standard MSIS problem to find a z_0 with small coefficients satisfying $\mathbf{A}z_0 = 0$ at SelfTargetMSIS. This exact approach can be followed to prove that Dilithium is secure in ROM depending on the difficulty of MLWE and SIS.

In QROM, where the adversary can query H in overlay, the situation is somewhat different. It has been shown in Dilithium that it is still based on MLWE and SelfTargetMSIS in QROM, even with a narrow reduction when the algorithm is deterministic. But one can no longer use the forking lemma directly (since it is a type of rewinding) to give a quantum reduction from MSIS to SelfTargetMSIS. There are still good reasons to believe that the SelfTargetMSIS problem, and thus Dilithium, is secure in QROM. First, there are no natural signature schemes built from Σ protocols using the Fiat-Shamir transformation that are secure in ROM but not in QROM. Furthermore, it is possible to set the parameters of Dilithium (while leaving the algorithm design unchanged) so that the SelfTargetMSIS problem becomes theoretically difficult, thus leaving this version of Dilithium secure in the MLWE-based QROM only. An instantiation of these parameters results [2] in an algorithm with signatures and public keys that are 2x and 5x larger respectively. Although we do not consider this a good compromise, the existence of such a system gives us additional confidence in the security of an optimised version of Dilithium.

Recently, new work has narrowed the gap between security in ROM and QROM even further. This work [2] showed that if the underlying Σ protocol collapses and has a particular strength, then its Fiat-Shamir transformation is a secure signature in the QROM. The particular soundness of the Dilithium protocol is directly implied by the difficulty of MSIS. Moreover, the said work, again, states a probable protocol collapse Σ of Dilithium. Again in this work is shown that the collapse property has a reduction with respect to MLWE. The reduction is rather non-narrow, but it gives even more affirmation that there is nothing uncertain in the construction of Dilithium or any natural algorithm built via the Fiat-Shamir framework that can be proven safe in ROM. In our view, it is certain that the distinction between secure signatures in ROM and QROM will soon be treated in the same way as the distinction

between secure schemes in the standard model and ROM - there will be theoretical differences, but the security in practice will be the same.

In **Table 04**, the formulas for the public key and signature sizes are in our bibliography [2]. The numbers in parentheses for the SIS difficulty correspond to the strongly unforgeable (i.e., it is difficult to find a second distinct signature for a previously seen message/signature pair) version of the signature scheme. For the length of the coefficient corresponding to the parenthesis-free version, we always refer our readers to our bibliography [2] for more details. Due to the rejection sampling, there is a noticeable difference between the median and average signature times, so we measure both.

Table 04: Output sizes, safety and performance of Dilithium.

NIST Security Level	2	3	5
Output Size			
public key size (bytes)	1312	1952	2592
Signature size (bytes)	2420	3293	4595
LWE Hardness (Core-SVP and refined)			
BKZ block size b (GSA)	423	624	863
Classical Core-SVP	123	182	252
Quantum Core-SVP	112	165	229
BKZ block size b (simulation)	433	624	863
$\log_2$ Classical Gates	159	217	285
$\log_2$ Classical Memory	98	139	187
SIS Hardness (Core-SVP)			
BKZ block size b	423 (417)	638 (602)	909 (868)
Core-SVP Classique	123 (121)	186 (176)	265 (253)
Core-SVP quantum	112 (110)	169 (159)	241 (230)
Performance (Unoptimized Reference Code, Skylake)			
Gen median cycles	300,751	544,232	819,475
Sign median cycles	1,081,174	1,713,783	2,383,399
Sign average cycles	1,355,434	2,348,703	2,856,803
Verify median cycles	327,362	522,267	871,609
Performance (AVX2, Skylake)			
Gen median cycles	124,031	256,403	298,050
Sign median cycles	259,172	428,587	538,986
Sign average cycles	333,013	529,106	642,192
Verify median cycles	118,412	179,424	279,936
Performance (AVX2+AES, Skylake)			
Gen median cycles	70,548	153,856	153,936
Sign median cycles	194,892	296,201	344,578
Sign average cycles	251,144	366,470	418,157
Verify median cycles	72,633	102,396	151,066

2.2.2. Number of iterations

We would now like to calculate the probability that step 21 assigns $(\mathbf{z}, \mathbf{h})$ to $\perp$. The probability that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ can be calculated by considering each coefficient separately. For each coefficient σ of $c\mathbf{s}_1$ the corresponding coefficient of $\mathbf{z}$ will be between $-\gamma_1 + \beta + 1$ and $-\gamma_1 + \beta - 1$ (inclusive) as long as the corresponding coefficient of $\mathbf{y}_i$ is between $-\gamma_1 + \beta + 1 - \sigma$ and $-\gamma_1 + \beta - 1 - \sigma$. The weight of this range is $2(\gamma_1 - \beta) - 1$ and the coefficient of $\mathbf{y}$ has $2\gamma_1 - 1$ possibilities. Thus the probability that any coefficient of $\mathbf{y}$ is in the correct range is :

$$\left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_1 - 1}\right)^{256 \cdot l} = \left(1 - \frac{\beta}{\gamma_1 - 1/2}\right)^{ln} \approx e^{-256 \cdot \beta l / \gamma_1} \quad (02)$$

where we use the fact that our γ_1 values are larger compared to $1/2$.

We then tackle the calculation of the probability that we have:

$$\|\mathbf{r}_0\|_\infty = \left\|\text{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)\right\|_\infty < \gamma_2 - \beta$$

If we assume (heuristically) that the lower order bits are uniformly distributed modulo $2\gamma_2$, then there are:

$$\left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_1 - 1}\right)^{256 \cdot k} \approx e^{-256 \cdot \beta k/\gamma_2}$$

probability that all coefficients are in the correct range (using the fact that our β values are large compared to $1/2$). Therefore, the probability that step 21 will succeed is:

$$\approx e^{-256 \cdot \beta (l/\gamma_1 + k/\gamma_2)} \quad \textbf{(03)}$$

Table 05: Parameter of Dilithium.

NIST Security Level	2	3	5
Parameters			
q [modulus]	8380417	8380417	8380417
d [dropped bits from **t**]	13	13	13
τ [# of 1's in c]	39	49	60
challenge entropy $\left[\log\binom{256}{\tau} + \tau\right]$	192	225	257
γ_1 [**y** coefficient range]	2^{17}	2^{19}	2^{19}
γ_1 [low-order rounding range]	$(q-1)/88$	$(q-1)/32$	$(q-1)/32$
(k, l) [dimensions of **A**]	(4,4)	(6,5)	(8,7)
η [secret key range]	2	4	2
β $[\tau \cdot \eta]$	78	196	120
ω [max. # of 1's in the hint **h**]	80	55	75
Répétitions (from the equation (08))	4.25	5.1	3.85

It is more difficult to formally calculate the probability that step 24 will result in a restart. The parameters were set so that heuristically $(\mathbf{z}, \mathbf{h}) = \perp$ with a probability between 1% and 2%. Therefore, the vast majority of loop repetitions will be caused by step 21.

We would like to emphasize that the expected number of iterations is independent of the secret key $\mathbf{s}_1$, $\mathbf{s}_2$ and thus no information about them can be obtained by an attack that counts the iterations.

3. Conclusions

This paper has allowed us to focus on the mathematic object that is the MLWE (Module Learning With Error) and its application as basis of the CRYSTALS-Kyber and Dilithium cryptosystems.

We have opened our writing on the development of the MLWE.

The section has put in exergue that there is a reduction (an equivalence in terms of difficulty or hardness then) from the Mod-GIVP (Generalized Independent Vector Problem over Module Lattices) and the M-LWE. To be more precise, the reduction was detailed for the reduction from Mod-GIVP to M-SLWE (Search version of the LWE over Module Lattices). The reader was then sent to our reference [3] for the second part of the reduction, which is M-SLWE to M-LWE.

We did then focus mainly on the cryptosystem CRYSTALS.

We started with the KEM (Key Encapsulation Mechanism) type one, the CRYSTALS-Kyber.

We advanced the specification of the two parts that compose it: Kyber.CPAPKE and Kyber.CCAKEM. Each of these two parts in their turn is subdivided into 03 algorithms: key generation, encryption, and decryption. Let us note in passing the fact that the operations of multiplication in the said algorithms, are carried out in the domain NTT (Number Theoretic Transform). The multiplication in R_q is performed in an efficient way (at a lower cost).

In terms of performance, it is at its peak for CRYSTALS-Kyber when it is written taking into account the advantages of processors supporting AVX2 technology for Intel Hashwell processors, and m4 technology for ARM processors.

Then came the case of CRYSTALS-Dilithium which falls in the category of signature cryptosystem.

In addition to the notion of NTT, which is still relevant, hash operations are also involved in a more marked way. We will list among others the ball hash which is represented in algorithmic terms in the function SampleInBall(ρ). ρ representing a seed.

Finally, the actual algorithm of CRYSTALS-Dilithium is given. It consists of 03 subparts: key generation, signature and verification.

4. References

[1]. R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler and D. Stehlé, *« CRYSTALS-Kyber - Algorithm Specifications And Supporting Documentation (version 2.0) »*, https://pq-crystals.org/kyber/index.shtml, Mar. 2019;

[2]. S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler and D. Stehlé, *« CRYSTALS-Dilithium - Algorithm Specifications and Supporting Documentation »*, https://pq-crystals.org/dilithium/index.shtml, Mar. 2021;

[3]. A. Langlois and D. Stehle, *« Worst-Case to Average-Case Reductions for Module Lattices »*, https://eprint.iacr.org/2012/090, Aug. 2013;

[4]. O. Regev, *« On Lattices, Learning with Errors, Random Linear Codes, and Cryptography »*, https cims.nyu.edu ~regev papers qcrypto.pdf, May 2009;

[5]. *« Easy explanation of IND- security notions? »*, https://crypto.stackexchange.com/questions/26689/easy-explanation-of-ind-security-notions, Oct. 2020;

[6]. *« EUF-CMA and SUF-CMA »*, https://blog.cryptographyengineering.com/euf-cma-and-suf-cma/, Oct. 2020;

[7]. V. Lyubashevsky, C. Peikert and O. Regev, *« On Ideal Lattices and Learning with Errors Over Rings »*, https://www.iacr.org/archive/eurocrypt2010/66320288/66320288.pdf, Nov. 2020;

[8]. S. Diop, B. O. Sané, M. Seck, *« NTRU-LPR IND-CPA A New Ideal Lattice-based Scheme »*, https://eprint.iacr.org/2018/109.pdf, Nov. 2020;

[9]. *« pq-crystals/security-estimates »*, https://github.com/pq-crystals/security-estimates, Feb. 2021;

[10]. J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler and D. Stehlé, *« CRYSTALS-Kyber a CCA-secure module-lattice-based KEM »*, https://eprint.iacr.org/2017/634.pdf, Feb. 2021;

[11]. *« FIPS PUB 202 – SHA-3 standard: Permutation-based hash and extendable-output functions »*, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf, Feb. 2021;

[12]. *« Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process »*, https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf, Mar. 2021;