Application du MLWE dans le cryptosystème CRYSTALS - Kyber comme KEM et Dilithium comme signature

Razafimahefa F. O.[1], Randriamitantsoa P. A.[2], Randriamitantsoa A. A[3]

Laboratoire de Recherche en Télécommunication, Automatique, Signal et Images (LR-TASI)

École Doctorale en Sciences et Techniques de l'Ingénierie et de l'Innovation (ED-STII)

Équipe d'Accueil Doctorale Télécommunication, Automatique, Signal et Images (EAD-TASI)

Université d'Antananarivo

BP 1500, Ankatso – Antananarivo 101 – Madagascar

[1]ombana.razafimahefa@gmail.com , [2]rpauguste@gmail.com, [3]andriau23@gmail.com

Résumé

Les ordinateurs quantiques, mais toujours avec un faible nombre de qubits, commencent à émerger au moment où nous écrivons. Il est plus crucial que jamais de travailler sur de nouveaux cryptosystèmes. Des cryptosystèmes qui seront basés sur des problèmes mathématiques suffisamment robustes pour résister à la puissance de calcul accrue que ces nouveaux types d'ordinateurs fourniront. Au cours de nos recherches, nous sommes tombés sur l'objet mathématique qu'est la Lattice. Cette dernière est l'un des candidats prometteurs en termes de problème fondamental pour les cryptosystèmes post-quantiques. Mais l'outil mathématique sur lequel nous allons ouvrir cet article est le MLWE ou Module Learning With Error. Le LWE (ou Learning With Error), en premier lieu, est une généralisation du problème Learning from parity with error. Il a été montré que ce problème est équivalent, en termes de difficulté, au SIVP ou Shortest Independent Vector Problem d'une lattice. Pour en revenir au MLWE, nous avons une autre désignation, Learning With Error other Module Lattices. Les modules lattices peuvent être considérées comme des lattices entre celles utilisées dans les définitions du problème LWE et celles utilisées pour le problème R-LWE (Ring LWE). Le Module-LWE offre un compromis entre les deux extrêmes du LWE et du R-LWE. Par la suite, nous nous concentrerons sur le cryptosystème CRYSTALS, un cryptosystème basé sur le MLWE. Notre choix d'étudier ce cryptosystème est basé sur le fait qu'il a été soumis au projet du NIST (National Institute of Standards and Technology) de standardisation des cryptosystèmes post-quantiques. CRYSTALS a pu progresser jusqu'à l'étape 3 du projet. Mais surtout, il est le seul parmi les rares à avoir progressé jusqu'à cette étape finale, à avoir à la fois un cryptosystème de

type KEM (Key Encapsulation Mechanism) et un autre de type signature.

Mots clés : *Ordinateur quantique, cryptosystème post-quantique, module LWE, CRYSTALS-Kyber, CRYSTALS-Dilithium, mécanisme d'encapsulation de clés (KEM), signature.*

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.

1 Le MLWE

La plupart des schémas cryptographiques basés sur les lattices sont construits sur la difficulté supposée des Short Integer Solution (SIS) et du Leaning With Error (LWE). Leur efficacité peut être améliorée en déplaçant les hypothèses de difficulté vers les problèmes plus compacts que constituent les Ring-SIS et Ring-LWE. Cependant, ce changement d'hypothèses de difficulté s'accompagne d'un affaiblissement de la sécurité : SIS et LWE sont connus pour être au moins aussi difficiles que les problèmes standard (dans le cas le plus défavorable) sur les lattices euclidiennes (version classique des lattices),

alors que Ring-SIS et Ring-LWE ont le même niveau de difficulté uniquement pour leurs restrictions à des classes spéciales d'ideal lattices, correspondant aux ideal de certains rings polynomiaux. [7]

Le présent sous-titre consiste en la définition des problèmes Module-SIS et, surtout, Module-LWE, qui relient respectivement SIS et Ring-SIS, et LWE avec Ring-LWE. Ces problèmes de cas moyen (average case, en anglais pour rappel) sont au moins aussi difficiles que les problèmes de lattices standards limités aux module lattices (ces derniers, les module lattices, à leur tour, servent de juste milieu entre les lattices arbitraires et les ideal lattices). Il est important de noter que les réductions allant du cas le plus défavorable au cas moyen pour les problèmes de module sont fortes, en ce sens qu'il existe des réductions inverses. Cette propriété n'est pas connue pour tenir dans le contexte de Ring-SIS / Ring-LWE : des problèmes d'ideal lattices pourraient se révéler faciles sans affecter la difficulté de Ring-SIS / Ring-LWE.

1.1 Variantes du LWE

1.1.1 LWE

Soit $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ dénotant le segment $[0,1)$ avec l'opération addition modulo 1. Pour une fonction de densité de probabilité χ sur $\mathbb{T}$ et un vecteur $s \in \mathbb{Z}_q^n$, nous posons $A_{s,\chi}$ la distribution sur $\mathbb{Z}_q^n \times \mathbb{T}$ obtenu en choisissant un vecteur $a \in \mathbb{Z}_q^n$ uniformément et d'une manière aléatoire, puis $e \in \mathbb{T}$ par respect à χ, et retournant $\left(a, \frac{1}{q}\langle a, s\rangle + e\right)$.

Définition 01 :

La version recherche du problème Learning With Error $\mathrm{SLWE}_{q,\chi}$ est comme suit : soit $s \in \mathbb{Z}_q^n$ un secret ; étant donnés plusieurs échantillons à partir de $A_{s,\chi}$, le but est de trouver s.

La version décision du problème Learning With Error $\mathrm{LWE}_{q,\chi}$ est comme suit : soit $s \in \mathbb{Z}_q^n$ choisi d'une manière uniforme et aléatoire ; le but est de faire la distinction entre plusieurs échantillons indépendants à partir de $A_{s,\chi}$ et le même nombre d'échantillons indépendants à partir de $U(\mathbb{Z}_q^n \times \mathbb{T})$. Où U fait référence à une distribution uniforme, sur l'ensemble $\mathbb{Z}_q^n \times \mathbb{T}$ dans le cas présent donc.

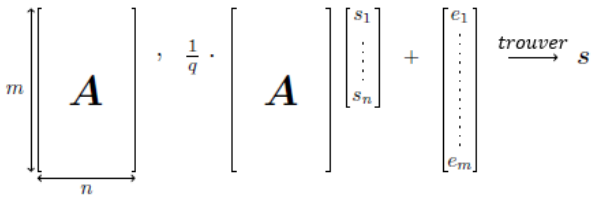

Figure 01 : *Le LWE en termes d'algèbre linéaire.*

Il est également possible d'interpréter le LWE en termes d'algèbre linéaire : Supposons que le nombre d'échantillons requis $\left(a_i, \frac{1}{q}\langle a_i, s\rangle + e_i\right)$ à partir de $A_{s,\chi}$ est m, alors nous considérons la matrice $A \in \mathbb{Z}_q^{m \times n}$ dont les lignes sont les a_i. Et

l'on considèrera le vecteur $e = (e_1, \dots, e_m)^T$. Alors le SLWE est comme nous le montre la **Figure 01**. [3], [4]

Théorème 01 :

Soit $\varepsilon(n) = n^{-\omega(1)}$, $\alpha \in (0,1)$ et $q \geq 2$ de telle sorte que $\alpha q \geq 2\sqrt{n}$. Il existe une réduction quantique de la résolution du $\mathrm{GIVP}^{\eta_\varepsilon}_{\sqrt{8}n/\alpha}$ en un temps polynomial (dans le cas le plus défavorable, avec une probabilité élevée) vers la résolution du $\mathrm{SLWE}_{q,D_\alpha}$ en un temps polynomial avec une probabilité non négligeable.

En assumant que q est premier, $q \leq \mathrm{poly}(n)$, et que χ soit une fonction de densité de probabilité sur $\mathbb{T}$. Alors il existe une réduction en un temps polynomial du $\mathrm{SLWE}_{q,\chi}$ au $\mathrm{LWE}_{q,\chi}$.

Le théorème qui va suivre permet d'outrepasser la condition que q doit être premier.

Théorème 02 :

Soit $\alpha > 0$, $\varepsilon \in (0, 1/2)$, $m \geq n \geq 1$, $p \geq 25$ et $q \in [p, 2p)$. Il existe une réduction à temps polynomial du $\mathrm{LWE}_{q,\alpha}$ au $\mathrm{LWE}_{p,\Psi_{\leq\beta}}$ où $\beta = C\alpha\sqrt{n}\sqrt{\log(n/\varepsilon)\log(nm/\varepsilon)}$, pour une constante positive C, et qui perd tout au plus ε en avantage.

1.1.2 R-LWE

Soit ψ une distribution sur $\mathbb{T}_{R^\vee} = K_{\mathbb{R}}/R^\vee$ et $s \in R_q^\vee$. Nous dénotons par $A^{(R)}_{s,\psi}$ la distribution sur $R_q \times \mathbb{T}_{R^\vee}$ obtenu en choisissant $a \in R_q$ d'une manière uniformément aléatoire et $e \in \mathbb{T}_{R^\vee}$ par respect à ψ, et retournant $(a, (a.s)/q + e)$.

Définition 02 :

Soit $q \geq 2$ et Ψ une famille de distributions sur $\mathbb{T}_{R^\vee}$. La version recherche du problème Ring Learning With Error $\text{R-SLWE}_{q,\Psi}$ est comme suit : soit $s \in R_q^\vee$ secret, et $\psi \in \Psi$; étant donné plusieurs échantillons à partir de $A^{(R)}_{s,\psi}$, le but est de trouver s.

Soit Υ une distribution sur une famille de distribution de bruits sur $\mathbb{T}_{R^\vee}$. La version décision du problème Ring Learning With Error $\text{R-SLWE}_{q,\Upsilon}$ est comme suit : supposons que $s \in R_q^\vee$ soit uniformément aléatoire, et ψ échantillonné à partir de Υ ; le but est de distinguer entre plusieurs échantillons indépendants partir de $A^{(R)}_{s,\psi}$ et le même nombre d'échantillons indépendants à partir de $U(R_q, \mathbb{T}_{R^\vee})$.

Théorème 03 :

Soit $\varepsilon(n) = n^{-\omega(1)}$, $\alpha \in (0,1)$ et $q \geq 2$ d'une factorisation connue de telle sorte que $\alpha q > \omega(\sqrt{\log n})$. Il existe une réduction quantique de la résolution du $\text{Id-GIVP}^{\eta_\varepsilon}_{\gamma}$ en un temps polynomial (dans le cas le plus défavorable, avec une probabilité élevée) vers la résolution du $\text{R-SLWE}_{q,\Psi_{\leq\alpha}}$ en un temps polynomial avec une probabilité non négligeable avec $\gamma = \sqrt{n}.\omega(\sqrt{\log n})/\alpha$.

En supposant que q est premier, $q \leq \text{poly}(n)$, et que $q = 1 \bmod v$. Alors il existe une réduction à temps polynomial de R-SLWE$_{q,\Psi_{\leq\alpha}}$ au R-LWE$_{q,\Upsilon_\alpha}$.

1.1.3 LWE over module

Le M-LWE généralise le LWE et le R-LWE. La variable n et d dénotent respectivement la dimension de R et le rang du module $M \subseteq R^d$. Nous posons $N = nd$ la dimension de la module lattice.

Soit ψ une probabilité de distribution sur $\mathbb{T}_{R^\vee}$ et $s \in \left(R_q^\vee\right)^d$ un vecteur. Nous définissons par $A_{q,s,\psi}^{(M)}$ la distribution sur $\left(R_q\right)^d \times \mathbb{T}_{R^\vee}$ obtenu en choisissant un vecteur $a \in \left(R_q\right)^d$ uniformément aléatoire, et $e \in \mathbb{T}_{R^\vee}$ par respect à ψ, et retournant $\left(a, \frac{1}{q}\langle a, s\rangle + e\right)$.

Définition 03 :

Soit $q \geq 2$ et Ψ une famille de distributions sur $\mathbb{T}_{R^\vee}$. La version recherche du problème Module Learning With Error M-SLWE$_{q,\Psi}$ est comme suit : Soit $s \in \left(R_q^\vee\right)^d$ un vecteur secret et $\psi \in \Psi$; étant donné arbitrairement plusieurs échantillons à partir de $A_{q,s,\psi}^{(M)}$, le but est de trouver s.

Pour un entier $q \geq 2$ et une distribution Υ sur une famille de distribution sur $K_{\mathbb{R}}$. La version décision du problème Module Learning With Error M-SLWE$_{q,\Upsilon}$ est comme suit : Soit $s \in \left(R_q^\vee\right)^d$ uniformément aléatoire et ψ échantillonné sur Υ ; le but est de distinguer entre arbitrairement plusieurs échantillons aléatoire à partir de $A_{q,s,\psi}^{(M)}$ et le même nombre d'échantillons à partir de $U\left(R_q^d \times \mathbb{T}_{R^\vee}\right)$.

Comme c'était le cas du LWE et du R-LWE, le problème M-LWE peut être interprété en termes d'algèbre linéaire. Il (le M-LWE) consiste à prendre la matrice A du LWE de la forme :

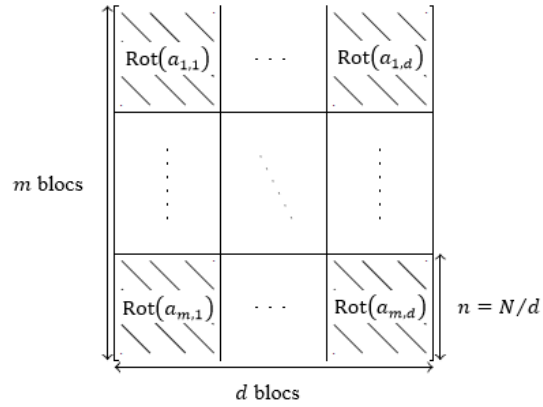

Figure 02 : *Le problème M-LWE peut être interprété en termes d'algèbre linéaire.*

La difficulté du M-LWE est soutenue par les deux théorèmes qui vont suivre.

Théorème 04 :

Soit $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ et $q \geq 2$ d'une factorisation connue de telle sorte que $\alpha q > 2\sqrt{d}.\omega\left(\sqrt{\log n}\right)$. Il y a une réduction quantique de la résolution du Mod-GIVP$_\gamma^{\eta_\varepsilon}$ en un temps polynomial (dans le cas le plus défavorable, avec une probabilité élevée) vers la résolution du M-

$\mathrm{SLWE}_{q,\Psi_{\leq\alpha}}$ en un temps polynomial avec un avantage non négligeable avec $\gamma = \sqrt{8Nd}.\,\omega(\sqrt{\log n})/\alpha$.

En assumant que q est premier, $q \leq \mathrm{poly}(N)$ et que $q = 1 \bmod v$. Alors il existe une réduction à temps polynomial depuis $\mathrm{M\text{-}SLWE}_{q,\Psi_{\leq\alpha}}$ vers $\mathrm{M\text{-}LWE}_{q,\Upsilon_\alpha}$.

Théorème 05 :

Soit $p, q \in \left[2, 2^{N^{O(1)}}\right]$ et $\alpha, \beta \in (0,1)$ de telle sorte que $\beta \geq \alpha.\max\left(1, \frac{q}{p}\right).n^{1/4}N^{1/2}.\,\omega(\log^2 N)$ et $\alpha q \geq \omega\sqrt{\log(N)/n}$. Il existe une réduction à temps polynomial de $\mathrm{M\text{-}LWE}_{q,\Upsilon_\alpha}$ vers $\mathrm{M\text{-}LWE}_{q,\Upsilon_\beta}$.

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

Lemme 01 :

Pour tout $\varepsilon = \varepsilon(N) \leq 1/10$ et tout γ et ϕ de telle sorte que $\gamma \cdot \phi(M) \leq \sqrt{2}\eta_\varepsilon(M)$, il existe une réduction à temps polynomial du $\mathrm{Mod\text{-}GIVP}^{\phi}_{2\sqrt{N}\cdot\gamma}$ au $\mathrm{M\text{-}DGS}^{\phi}_{\gamma}$. [3]

Lemme 02 :

Soit $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ et q un entier de telle sorte que $\alpha q \geq 2\sqrt{d} \cdot \omega(\sqrt{\log n})$. En assumant que nous avons accès à un oracle qui résout $\mathrm{M\text{-}SLWE}_{q,\Psi_{\leq\alpha}}$, étant donnés un nombre d'échantillons polynomial. Alors il existe un algorithme quantique à temps polynomial pour $\mathrm{M\text{-}DGS}^{\eta_\varepsilon}_{\sqrt{2d}\cdot\omega(\sqrt{\log n})/\alpha}$.

1.2 Réduction du Mod-GIVP au M-SLWE

La réduction du Mod-GIVP au M-SLWE utilise le problème intermédiaire qui va suivre, où ϕ dénote une fonction à valeur réelle sur une lattice et γ dépend de la dimension, appelé *Module Discrete Gaussian Sampling Problem* $\mathrm{M\text{-}DGS}^{\phi}_{\gamma}$: étant donné une module lattice M de dimension N, et un nombre $r > \gamma \cdot \phi$, le but est de retourner un échantillon à partir de $D_{M,r}$. La réduction se fait en 2 (deux) étapes :

Lemme 03 :

Soit $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ et $q \geq 2$. Assumons que nous avons accès à un oracle qui résout $\mathrm{M\text{-}SLWE}_{q,\Psi_{\leq\alpha}}$ en un temps polynomial avec une probabilité non négligeable. Alors, il existe un algorithme quantique à temps polynomial qui, étant donnée une module lattice M de dimension N, un nombre $r > 2q\eta_\varepsilon(M)$ et $\mathrm{poly}(N)$ échantillons à partir de $D_{M,r}$, produit un échantillon à partir de $D_{M,\frac{r\sqrt{d}\cdot\omega(\sqrt{\log n})}{\alpha q}}$ avec une probabilité non négligeable.

Pour prouver le **Lemme 03** nous utilisons un problème intermédiaire, le $\mathrm{Mod\text{-}BDD}_\delta$ (*BDD* pour *Bounded Distance Decoding*) : étant donné une module lattice M, $\delta < \lambda_1(M)/2$ et tout point

$y \in \mathbb{R}^n$ de la forme $y = x + e$ pour un certain $x \in M$ et $\|e\|_{2,\infty} \leq \delta$, trouver x. Il est à noter que la norme $\ell_{2,\infty}$ est utilisée ici, parce que cela nous aidera à énoncer le **Lemme 06** plus convenablement.

Un autre problème intermédiaire q-Mod-BDD$_\delta$ sera utilisé : étant donné une module lattice M et un point $y \in \mathbb{R}^n$ dans la distance (par respect à la norme $\ell_{2,\infty}$) δ de M, donner en sortie le coset dans M/qM du vecteur le plus proche y. La démonstration du **Lemme 03** consiste en la séquence de réduction ci-dessous (il est à noter que δ ici prend la valeur $\frac{\alpha q \cdot \omega(\sqrt{\log n})}{\sqrt{2n}r}$) :

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

Lemme 04 :

Il existe un algorithme quantique efficace qui, étant donnée une module lattice M de dimension N, un nombre $\delta < \lambda_1(M^\vee)\omega(\sqrt{\log n})/(2\sqrt{n})$, et un oracle qui résout Mod-BDD$_\delta$ sur $M^\vee$, donne à la sortie des échantillons à partir de $D_{M,r\sqrt{d}\cdot\omega(\sqrt{\log n})/(\sqrt{2}\delta)}$.

Lemme 05 :

Pour tout $q \geq 2$, il existe une réduction à temps polynomial du Mod-BDD$_\delta$ au q-Mod-BDD$_\delta$.

Lemme 06 :

Soit $\varepsilon(N) = N^{-\omega(1)}$, $\alpha \in (0,1)$ et $q \geq 2$. Soit $M \subseteq R^d$ un R-module, et $r > \sqrt{2}q \cdot \eta_\varepsilon(M)$. Etant donné un accès à un échantillonneur oracle à partir de la distribution $D_{M,r}$, il existe une réduction probabiliste du q-Mod-BDD$_{M^\vee,\frac{\alpha q\cdot\omega(\sqrt{\log n})}{\sqrt{2n}r}}$ au M-SLWE$_{q,\Psi_{\leq\alpha}}$.

Le principe de la réduction est de construire à partir de y, l'entrée de q-Mod-BDD, et à partir de quelques échantillons Gaussiens continus ou discret, les pairs (a, b) distribués selon $A^{(M)}_{q,s,\psi}$, où s va directement dépendre du vecteur x le plus proche de y. Pour produire de tels échantillons (a, b) suivant la distribution désirée, nous combinons les preuves correspondants pour LWE et R-LWE. Puis d'appeler l'oracle du M-SLWE pour retourner s et nous permettre de recouvrir l'information sur x.

Lemme 07 :

Soit $\varepsilon > 0$ et $s = \Theta(x \bmod qM^\vee)$. Il existe $\psi \in \Psi_{\leq\alpha}$ de telle sorte que la distance statistique entre

$A_{q,s,\psi}^{(M)}$ et la distribution de (a, b) est tout au plus 6ε.

Tout ceci conclue la démonstration de la première partie de notre **Théorème 04**.

Pour la seconde partie du théorème (existence d'une réduction du M-SLWE au M-LWE), nous invitons nos lecteurs à consulter la démonstration dans notre bibliographie (paragraphe 4.3 dans [3]).

2 Le cryptosystème CRYSTALS

Le cryptosystème *CRYSTALS* ou *Cryptographic Suite for Algebraic Lattices* encapsule deux types de cryptosystèmes : *Kyber*, un KEM (Key Encapsulation Mechanism, pour rappel) ayant la spécificité d'être classifié sécurisé IND-CCA2 (nous verrons la définition de ce terme un peu plus loin dans notre rédaction), et *Dilithium* un algorithme de signature sécurisé EUF-CMA (même cas que l'IND-CCA2, une définition sera donnée plus loin). Les deux algorithmes sont basés sur les problèmes difficiles relatifs aux modules lattices, ont été par conséquent construits pour résister aux éventuels attaques d'ordinateurs quantiques.

Le CRYSTALS, comme mentionnés dans l'introduction du présent ouvrage, a été soumis au projet NIST de cryptographie post-quantique. Il a pu progresser jusqu'à l'étape 3 dudit projet de standardisation. Mais surtout, le seul parmi le quelques-uns qui ont progresser à cette étape finale, à posséder à la fois un cryptosystème type KEM et signature. [5], [6], [8]

2.1 CRYSTALS-Kyber, cryptosystèmetype KEM

2.1.1 Analyses de performance

Dans cette section, nous examinons les aspects de mise en œuvre de Kyber et rapportons les résultats de performance de deux implémentations : l'implémentation de référence ANSI C (ANSI pour *American National Standards Institute*) et une implémentation optimisée à l'aide d'instructions vectorielles AVX2.

2.1.1.1 Performances de Kyber sur les processeurs Intel Haswell

Le **Tableau 01** présente les résultats de performance d'Intel Haswell de l'implémentation de référence et d'une implémentation optimisée AVX2 de Kyber et de la variante 90s de Kyber ainsi que les tailles des clés et des textes chiffrés. Tous les benchmarks ont été obtenus sur un cœur d'un processeur Intel Core i7-4770K (Haswell) cadencé à 3492 MHz (comme indiqué par /proc/cpuinfo) avec TurboBoost et l'hyperthreading désactivés. La machine utilisée a 32 Go de RAM (pour *Random Access Memory*)et exécute Debian GNU/Linux (GNU de *GNU's Not Unix*) avec la version 4.19.0 du

noyau Linux. Les deux implémentations ont été compilées avec gcc version 8.3.0. Tous les décomptes de cycles indiqués sont la médiane des décomptes de cycles de 10000 exécutions de la fonction respective. Les implémentations ne sont pas optimisées pour l'utilisation de la mémoire, mais en général, Kyber n'a que des besoins en mémoire très modestes.

En addition, *ref* fait référence à l'implémentation de référence C, *AVX2* à l'implémentation utilisant des instructions vectorielles AVX2 ; sk signifie clé secrète, pk pour clé publique et ct pour texte chiffré. Entre parenthèses se trouvent des valeurs approximatives lors de l'inclusion de la génération de clé dans la décapsulation pour éviter d'avoir à stocker des clés secrètes développées. Dans ce scénario, nous stockons uniquement la graine initiale d au tout début de l'algorithme de génération de clé $\mathrm{Kyber.CPAPKE.KeyGen}()$. Les nombres de cycles approximatifs pour ce scénario sont calculés comme la somme des nombres de cycles pour la décapsulation standard et la génération de clé moins le nombre de cycles nécessaires pour générer la matrice $\mathbf{A}$ de la graine publique ρ. Notez qu'il s'agit d'une estimation très prudente ; les mises en œuvre réelles de l'approche peuvent également économiser, par exemple, l'échantillonnage des 32 octets de caractère aléatoire.

Kyber512					
Tailles (en Octets)		Cycles Hashwell (ref)		Cycles Hashwell (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 – années 90					
Tailles (en Octets)		Cycles Hashwell (ref)		Cycles Hashwell (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					
Tailles (en Octets)		Cycles Hashwell (ref)		Cycles Hashwell (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 – années 90					
Tailles (en Octets)		Cycles Hashwell (ref)		Cycles Hashwell (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					
Tailles (en Octets)		Cycles Hashwell (ref)		Cycles Hashwell (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 – années 90					
Tailles (en Octets)		Cycles Hashwell (ref)		Cycles Hashwell (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)

Tableau 01 : *Performances de Kyber sur les processeurs Intel Haswell.*

2.1.1.2 Performance of Kyber on ARM Cortex-M4 CPUs

Le **Tableau 02** rapporte le nombre de cycles et l'utilisation de la RAM d'une implémentation C (propre) et d'une implémentation optimisée (m4) de Kyber sur un ARM Cortex-M4. Tous les benchmarks sont obtenus en utilisant le framework pqm4 sur une carte Discovery STM32F407. Nous ne rapportons pas ici les benchmarks de la version des années 90 de Kyber sur le Cortex-M4, car pqm4 n'inclut pas encore d'implémentation en temps constant d'AES.

Kyber512				
	M4 Cycles (propre)	M4 RAM (propre)	M4 Cycles (m4)	M4 RAM (m4)
gen :	655595	6020	463068	2844
enc :	865256	8668	561518	2484
dec :	961648	9444	519237	2508
Kyber768				
	M4 Cycles (propre)	M4 RAM (propre)	M4 Cycles (m4)	M4 RAM (m4)
gen :	1087897	10052	756224	3292
enc :	1373744	13212	915676	2980
dec :	1491214	14308	853001	3004
Kyber1024				
	M4 Cycles (propre)	M4 RAM (propre)	M4 Cycles (m4)	M4 RAM (m4)
gen :	1696314	15180	1213303	3804
enc :	2057522	18844	1407769	3492
dec :	2199958	20420	1326409	3516

Tableau 02 : *Performances de Kyber sur les processeurs Intel Haswell.*

2.1.2 Estimation de niveau de sécurité

Dans notre présent paragraphe, nous nous sommes principalement appuyé sur nos bibliographies [1], [10] et [11].

Le **Tableau 03** répertorie les niveaux de sécurité selon la définition de la section 4.A.5 de l'appel à propositions du NIST [11] pour les différents ensembles de paramètres de Kyber. Nos affirmations sont basées sur les estimations de coût des attaques les plus connues contre le problème MLWE sous-jacent à Kyber, comme détaillé dans la sous-section 5.1 de notre bibliographie [1]. Plus précisément, y est listée, la difficulté classique et quantique du noyau-SVP et les utilisons pour dériver les niveaux de sécurité.

	Kyber512	Kyber768	Kyber1024
Niveau de sécurité NIST	1	3	5
Méthodologie noyau-SVP, attaque primitive uniquement			
Attaque lattice dim. d	1003	1424	1885
BKZ-blocksize β (BKZ pour Block Korkine Zolotarev)	403	625	877
Noyau-SVP difficulté classique	118	182	256
Noyau-SVP difficulté quantique	107	165	232
Estimation affinée pour les attaques classiques			
Attaque lattice dim. d	1025	1467	1918
BKZ-blocksize β	413	637	894
Tamisage 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

Tableau 03 : *Difficulté classique et quantique des différents ensembles de paramètres proposés pour Kyber.*

Tous les jeux de paramètres de Kyber ont une certaine probabilité d'échec de décryptage. Ces échecs sont un problème de sécurité (voir Section 5.5 de [1]) et donc les probabilités avec lesquelles ils se produisent doivent être faibles. Mais comme dans le ROM classique, la probabilité d'échec de décryptage est théorique de l'information, nous ne voyons pas la nécessité de la diminuer avec le paramètre de sécurité. En particulier, l'échec de décryptage pour nos ensembles de paramètres de niveau 3 et 5 est inférieur à 2^{-160}, ce qui signifie que si 280 instances de Kyber étaient exécutées toutes les

secondes à partir de maintenant jusqu'à ce que notre soleil devienne une naine blanche, les chances sont toujours fortement favorables à ce qu'il n'y ait jamais d'échec de décryptage. Nous excluons donc ces attaques de nos affirmations concernant les estimations de sécurité du NIST.

L'impact du bruit déterministe causé par $\mathrm{Compress}_q$ *sur Kyber512*. Chaque coefficient de $\mathbf{e_1}$ (et $\mathbf{e_2}$) dans l'algorithme de cryptage $\mathrm{Kyber.CPAPKE}$ se comporte comme une distribution binomiale avec le paramètre $\eta_2 = 2$, avec une variance $\eta_2/2 = 1$. Le paramètre $d_u = 10$ implique que la fonction $\mathrm{Compress}_q$ mappe les éléments modulo q à un ensemble de taille 2^{10} où la différence entre tous les pairs d'éléments de ce dudit ensemble est de 3 ou 4. Cela implique que l'erreur ajoutée par la fonction $\mathrm{Compress}_q$ pour chaque coefficient est soit uniforme sur $\{-1,0,1\}$, $\{-1,0,1,2\}$ ou $\{-2,-1,0,1\}$. Elle a donc une variance au moins aussi grande que la distribution uniforme sur l'ensemble $\{-1,0,1\}$, soit $2/3$. Cela rend la variance totale de chaque coefficient de $\mathbf{e_1}$ plus l'erreur déterministe au moins $1 + 2/3 = 5/3$. Les autres termes secret et bruit ont des distributions binomiales avec le paramètre $\eta_1 = 3$ pour une variance de $\eta_1/2 = 3/2 < 5/3$. En tenant compte des erreurs ajoutées par $\mathrm{Compress}_q$, nous calculons donc la difficulté de Kyber512 en supposant que la variance de chaque coefficient secret/erreur est $3/2$.

L'impact de MAXDEPTH. Les accélérations quantiques les plus connues pour l'algorithme de tamisage, qui sont plus détaillés dans l'analyse des coûts (se référer à la sous-section 5.1.1 de [1]), ne sont que légèrement affectées par la limitation de la profondeur d'un circuit quantique, car elle utilise la recherche de Grover sur des ensembles de petite taille (comparé pour rechercher dans tout l'espace de clés d'AES). Pour que les estimations de fonctionnement de la difficulté noyau-SVP correspondent au coût de la porte quantique de la rupture d'AES aux niveaux de sécurité respectifs, un ordinateur quantique devrait prendre en charge une profondeur maximale de 70 à 80. Lors de la limitation de la profondeur maximale à des valeurs plus petites, ou lors de l'examen d'attaques classiques, les estimations de la difficulté noyau-SVP sont plus petites que le nombre de portes pour les attaques contre AES. Une étude récente sur le coût concret du tamisage suggère que les accélérations quantiques de ces algorithmes sont ténues, indépendamment de la valeur de MAXDEPTH.

2.2 CRYSTALS-Dilithium, the signature type cryptosystem

2.2.1 Sécurité

La sécurité de notre algorithme de signature peut être prouvée par le biais du ROM, sur la base de la difficulté de deux problèmes [2], [9]. Le premier est le problème standard LWE (sur des anneaux polynomiaux) qui demande de distinguer $(\mathbf{A}, \mathbf{t} \coloneqq \mathbf{As_1} + \mathbf{s_2})$ de $(\mathbf{A}, \mathbf{u})$, où $\mathbf{u}$ est uniformément aléatoire. L'autre problème est ce qu'on a appelé le problème SelfTargetMSIS, qui est le problème de trouver un vecteur $\begin{bmatrix}\mathbf{z}\\ c\\ \mathbf{v}\end{bmatrix}$ avec de petits coefficients et un message μ satisfaisant :

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

où $\mathbf{A}$ et t sont uniformément aléatoires et $\mathbf{I}$ est la matrice d'identité. Dans le ROM, on peut obtenir une réduction (non étroite) en utilisant le *lemme d'embranchement* (*forking lemma* en anglais) du problème MSIS standard de trouver un z_0 avec de petits coefficients satisfaisant $\mathbf{A}z_0 = 0$ au SelfTargetMSIS. On peut suivre cette approche exacte pour prouver que Dilithium est sécurisé dans le ROM en fonction de la difficulté du MLWE et SIS.

Niveau de sécurité NIST	2	3	5
Taille de sortie			
taille de la clé publique (bytes)	1312	1952	2592
taille de la signature (bytes)	2420	3293	4595
Difficulté du LWE (Core-SVP, et pour la version améliorée également)			
taille de bloc BKZ b (GSA)	423	624	863
Core-SVP classique	123	182	252
Core-SVP quantum	112	165	229
taille de bloc BKZ b (simulation)	433	624	863
$\log_2$ portes classiques	159	217	285
$\log_2$ mémoires classiques	98	139	187
Difficulté SIS (Core-SVP)			
taille de bloc BKZ 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 (Code de référence non optimisé, Skylake)			
Gen cycle médian	300,751	544,232	819,475
Sign cycle médian	1,081,174	1,713,783	2,383,399
Sign cycle moyen	1,355,434	2,348,703	2,856,803
Verify cycle médian	327,362	522,267	871,609
Performance (AVX2, Skylake)			
Gen cycle médian	124, 031	256, 403	298, 050
Sign cycle médian	259, 172	428, 587	538, 986
Sign cycle moyen	333, 013	529, 106	642, 192
Verify cycle médian	118, 412	179, 424	279, 936
Performance (AVX2+AES, Skylake)			
Gen cycle médian	70, 548	153, 856	153, 936
Sign cycle médian	194, 892	296, 201	344, 578
Sign cycle moyen	251, 144	366, 470	418, 157
Verify cycle médian	72, 633	102, 396	151, 066

Tableau 04 : *Tailles de sortie, sécurité et performances de Dilithium.*

Dans le QROM, où l'adversaire peut interroger H en superposition, la situation est un peu différente. Il a été montré dans que Dilithium est toujours basé sur MLWE et SelfTargetMSIS dans le QROM, même avec une réduction étroite lorsque l'algorithme est déterministe. Mais on ne peut plus utiliser directement le lemme d'embranchement (puisqu'il s'agit d'un type de rembobinage) pour donner une réduction quantique de MSIS à SelfTargetMSIS. Il y a encore de bonnes raisons de croire que le problème SelfTargetMSIS, et donc Dilithium, est sécurisé dans le QROM. Premièrement, il n'y a pas de schémas de signature naturelle construits à partir de protocoles Σ utilisant la transformation Fiat-Shamir qui sont sécurisés dans le ROM mais pas dans le QROM. En outre, il est possible de définir les paramètres de Dilithium (tout en laissant la conception de l'algorithme inchangée) afin que le problème SelfTargetMSIS devienne théoriquement difficile, laissant ainsi cette version de Dilithium sécurisée dans le QROM basé uniquement sur MLWE. Une instanciation de ces paramètres aboutit [2] à un algorithme avec des signatures et des clés publiques qui sont respectivement 2x et 5x plus grandes. Bien que nous ne considérions pas cela comme un bon compromis, l'existence d'un tel système nous

donne une confiance supplémentaire dans la sécurité d'une version optimisée de Dilithium.

Récemment, de nouveaux travaux ont réduit encore plus l'écart entre la sécurité dans le ROM et le QROM. Ces travaux [2] ont montré que si le protocole Σ sous-jacent s'effondre et a une solidité particulière, alors sa transformation Fiat-Shamir est une signature sécurisée dans le QROM. La solidité particulière du protocole Dilithium est directement impliquée par la difficulté du MSIS. De plus, lesdits travaux, toujours, énonce un probable effondrement de protocole Σ de Dilithium. Encore dans ces travaux est montré que la propriété d'effondrement a une réduction par rapport à MLWE. La réduction est plutôt non étroite, mais elle donne encore plus d'affirmation qu'il n'y a rien d'incertain dans la construction de Dilithium ou de tout algorithme naturel construit via le cadre Fiat-Shamir dont la sécurité peut être prouvée dans le ROM. À notre avis, il est certain que la distinction entre les signatures sécurisées dans le ROM et QROM sera bientôt traitée de la même manière que la distinction entre les schémas sécurisés dans le modèle standard et le ROM - il y aura des différences théoriques, mais la sécurité dans la pratique sera la même.

Dans le **Tableau 04**, les formules pour les tailles de la clé publique et de la signature sont dans notre bibliographie [2]. Les chiffres entre parenthèses pour la difficulté SIS correspondent à la version fortement infalsifiable (c'est-à-dire qu'il est difficile de trouver une deuxième signature distincte pour une paire message/signature déjà vue) du schéma de signature. Pour la longueur du coefficient correspondant à la version sans parenthèse, nous renvoyons toujours nos lecteurs à notre bibliographie [2] pour plus de détails. En raison de l'échantillonnage de rejet, il y a une différence notable entre les temps de signature médians et moyens, et nous mesurons donc les deux.

2.2.2 Nombre d'itérations

Nous aimerions maintenant calculer la probabilité que l'étape 21, de l'algorithme détaillé de signature de Dilithium, assigne $(\mathbf{z}, \mathbf{h})$ à $\perp$. La probabilité que $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ peut être calculée en considérant chaque coefficient séparément. Pour chaque coefficient σ de $c\mathbf{s}_1$ le coefficient correspondant de $\mathbf{z}$ sera entre $-\gamma_1 + \beta + 1$ et $-\gamma_1 + \beta - 1$ (inclusivement) tant que le coefficient correspondant de $\mathbf{y}_i$ est entre $-\gamma_1 + \beta + 1 - \sigma$ et $-\gamma_1 + \beta - 1 - \sigma$. Le poids de cette fourchette est $2(\gamma_1 - \beta) - 1$ et la coefficient de $\mathbf{y}$ a $2\gamma_1 - 1$ possibilités. Ainsi la probabilité pour que tout coefficient de $\mathbf{y}$ soit dans la bonne fourchette est de :

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

où nous utilisons le fait que nos valeurs de γ_1 sont plus grandes comparées à $1/2$.

Nous nous attaquons par la suite au calcul de la probabilité que nous avons :

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

Si nous assumons (heuristiquement) que les bits d'ordre inférieur sont uniformément distribués modulo $2\gamma_2$, alors il y a :

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

probabilité que tous les coefficients soient dans la bonne fourchette (en utilisant le fait que nos valeurs de β sont grandes par rapport à $1/2$). Par conséquent, la probabilité que l'étape 21 réussisse est :

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

Il est plus difficile de calculer formellement la probabilité que l'étape 24 aboutisse à un redémarrage. Les paramètres ont été fixés de telle sorte qu'heuristiquement $(\mathbf{z}, \mathbf{h}) = \perp$ avec une probabilité comprise entre 1% et 2%. Par conséquent, la grande majorité des répétitions de la boucle seront causées par l'étape 21.

Nous tenons à souligner que le nombre d'itérations prévu est indépendant de la clé secrète $\mathbf{s}_1$, $\mathbf{s}_2$ et donc aucune information à leur sujet ne peut être obtenue par une attaque qui compte les itérations.

Niveau de sécurité NIST	2	3	5
Paramètres			
q [module]	8380417	8380417	8380417
d [bits de $\mathbf{t}$ ignorés]	13	13	13
τ [# de bits 1s dans c]	39	49	60
défi en termes d'entropie $\left[\log\binom{256}{\tau}+\tau\right]$	192	225	257
γ_1 [fourchette de coefficient de $\mathbf{y}$]	2^{17}	2^{19}	2^{19}
γ_1 [fourchette d'approximation pour l'ordre inférieur]	$(q-1)/88$	$(q-1)/32$	$(q-1)/32$
(k,l) [dimension de $\mathbf{A}$]	(4,4)	(6,5)	(8,7)

η [fourchette de la clé secrète]	2	4	2
β $[\tau \cdot \eta]$	78	196	120
ω [# de bits 1s max dans l'indice **h**]	80	55	75
Itérations (depuis l'équation (03))	4.25	5.1	3.85

Tableau 05 : *Jeux de paramètres de Dilithium.*

3 Conclusion

Cet article nous a permis de nous concentrer sur l'objet mathématique qu'est le MLWE (Module Learning With Error) et son application comme base des cryptosystèmes CRYSTALS-Kyber et CRYSTALS-Dilithium.

Nous avons ouvert notre rédaction sur le développement du MLWE.

La section a mis en exergue qu'il existe une réduction (une équivalence en termes de difficulté ou de dureté donc) du Mod-GIVP (Generalized Independent Vector Problem over Module Lattices) et du M-LWE. Pour être plus précis, la réduction a été détaillée pour la réduction de Mod-GIVP à M-SLWE (version Search du LWE sur les treillis de modules). Le lecteur a ensuite été renvoyé à notre référence [3] pour la deuxième partie de la réduction, qui est M-SLWE vers M-LWE.

Nous nous sommes ensuite concentrés principalement sur le système de cryptage CRYSTALS.

Nous avons commencé par celui de type KEM (Key Encapsulation Mechanism), le CRYSTALS-Kyber.

Nous avons avancé la spécification des deux parties qui le composent : Kyber.CPAPKE et Kyber.CCAKEM. Chacune de ces deux parties est à son tour subdivisée en 03 algorithmes : génération de clé, chiffrement et déchiffrement. Notons au passage le fait que les opérations de multiplication dans lesdits algorithmes, sont effectuées dans le domaine NTT (Number Theoretic Transform). La multiplication dans R_q est réalisée de manière efficace (à moindre coût).

En termes de performance, elle est à son maximum pour CRYSTALS-Kyber lorsqu'il est écrit en tenant compte des avantages des processeurs supportant la technologie AVX2 pour les processeurs Intel Hashwell, et la technologie m4 pour les processeurs ARM.

Vient ensuite le cas de CRYSTALS-Dilithium qui entre dans la catégorie des cryptosystèmes type signature.

Outre la notion de NTT, qui est toujours d'actualité, les opérations de hachage

interviennent de manière plus marquée. Nous citerons entre autres le hachage en boule qui est représenté en termes algorithmiques par la fonction SampleInBall(ρ). ρ représentant une graine.

Enfin, l'algorithme réel de CRYSTALS-Dilithium est donné. Il est composé de 03 sous-parties : génération de clé, signature et vérification.

4 Bibliographie

[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. Diop1, B. O. Sané1 and M. Seck1, *« 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]. *« 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;