CRYSTALS (CRYptographic SuiTe for Algebraic LatticeS): A New (Classical) Post-Quantum Public-Key Cryptosystem Standard

Advanced Topics in Information Security II / Cryptography and Security Protocols

Ph.D. (Doctoral Program) in Information Security

Instituto Superior Técnico, University of Lisbon (2022/2023 - 2nd Semester)

Rúben Barreiro[1⋆] and Paulo Mateus[1,2⋆⋆]

[1] Instituto Superior Técnico, University of Lisbon, Portugal
{ruben.andre.letra.barreiro,paulo.mateus}
@tecnico.ulisboa.pt
[2] Instituto de Telecomunicações, Portugal
paulo.mateus@lx.it.pt

⋆ Student Identification: IST1108107

⋆⋆ Acknowledgments and thanks to Prof. Paulo Mateus, who is the lecturer of the course "Advanced Topics in Information Security II / Cryptography and Security Protocols" at Instituto Superior Técnico, University of Lisbon, Portugal.

Abstract. In the last few years, Quantum Computing has been growing exponentially, and several quantum algorithms are threatening most of the cryptography we are using nowadays. In 1994, Peter Shor proposed one of them, known nowadays simply as Shor's algorithm. This algorithm is capable of factoring large integer numbers into smaller prime numbers and finding discrete logarithms in a polylogarithmic time. These two problems are generally considered hard on a classical computer, and we apply this computational hardness assumption as the basis of several cryptosystems we used since the 1970s. For instance, the security factor of popular cryptosystems, such as Rivest-Shamir Adleman (RSA), Diffie-Hellman (DH), and Elliptic-Curve Cryptography (ECC), rely on this assumption. However, this hardness assumption does not hold for quantum computers due to the expected impact of Shor's algorithm.
Concerning this imminent threat, the scientific community is putting enormous efforts into selecting new quantum-resistant candidates, arising two new leading families of quantum-resistant cryptography, known as (Classical) Post Quantum Cryptography and Quantum Cryptography. For the former, we consider mathematical approaches and problems believed to be computationally hard to solve in both classical and quantum computers. From this family of candidates, we have a popular set of cryptographic primitives based on mathematical problems related to algebraic lattices, commonly known as Lattice-based Cryptography.
In this paper/report, we discuss Lattice-based Cryptography in detail, its fundamentals, as well as the mathematical and computational hardness assumptions involved, presenting a detailed description and analysis of CRYptographic SuiTe for Algebraic LatticeS (CRYSTALS) public-key cryptosystem, namely its variants for asymmetric encryption (CRYSTALS-Kyber) and digital signatures (CRYSTALS-Dilithium), already selected as the new (and first-ever) standards for public-key cryptography to be adopted at the beginning of the post-quantum era.

1 Background

Recently, due to the emergence of Quantum Computing and its expected future impact on Modern Cryptography, from the threats arising from some quantum algorithms such as Grover's, Simon's, Brassard-Hoyer-Tapp (BHT), and Shor's algorithms, new quantum-resistant cryptographic primitives are being studied.

These new alternatives of Modern Cryptography are mainly focused on Key Exchanges and Public-Key Cryptography since the expected impacts of Grover's, Simon's, and BHT algorithms on Symmetric Cryptography and Cryptographic Hash Functions are minor when compared to the impact Shor's algorithm will have on Asymmetric Cryptography and Public-Key Cryptosystems we use.

We support this claim since we expect symmetric encryption and cryptographic hash algorithms to have their security strength only reduced by half and by a third, respectively. For instance, symmetric encryption algorithms, such as Advanced Encryption Standard (AES), are currently believed to be quantum-resistant when we use them with key sizes doubled regarding what we consider secure nowadays (i.e., 256 bits are considered quantum-resistant) in order to resist to brute-force attacks made by Grover's and Simon's algorithms. Similarly, cryptographic hash algorithms, such as Secure Hash Algorithm (SHA), are currently believed to be quantum-resistant when we use them with output sizes tripled regarding what we consider secure nowadays (i.e., 384 bits or more, are believed to be quantum-resistant) in order to resist to collision and birthday attacks made by BHT algorithm (also known as Quantum Birthday Attack).

In the opposite direction, we expect the Asymmetric and Public-Key Cryptography we currently use nowadays to be easily "broken" by Shor's algorithm in polylogarithmic time, running in a sufficiently powerful Quantum Computer with a large processing capacity, no matter how large the keys used are nor how we change the security parameters. This claim is valid since Shor's algorithm can solve the factorization of large integer numbers into smaller prime numbers as well as the discrete logarithm problem in a very efficient fashion, while most of the currently used public-key cryptosystems, digital signature schemes, and key exchange protocols in our daily lives, such as RSA, DH, ElGamal, Digital Signature Algorithm (DSA), and other primitives based on ECC, have their security relied precisely on the computational hardness from these two mathematical problems when we consider only classical contexts. However, for quantum contexts, all these mentioned cryptographic primitives will become obsolete, and that is the reason why we need to replace them.

From this incoming need, two new major families of Modern Cryptography arose, one named (Classical) Post-Quantum Cryptography and the other named Quantum Cryptography, as the more suitable candidates for replacement of the cryptographic primitives we use nowadays, which we will start to refer to as (Classical) Pre-Quantum Cryptography to simplify the presented nomenclature.

The former is very similar to the cryptographic primitives we currently use in (Classical) Pre-Quantum Cryptography and also uses classical information (in the form of bits). However, (Classical) Post-Quantum Cryptography has its basis in mathematical problems and computational assumptions we believe to be hard to be solved by a classical computer and even by a considerably powerful quantum computer. In particular, we have several categories of (Classical) Post-Quantum Cryptography differing on what type of mathematical problems they have their computational hardness assumptions based on, such as Lattice-based Cryptography, Code-based Cryptography, Hash-based Cryptography, Isogeny-based Cryptography, Multivariate Cryptography, Non-Commutative (Group-based) Cryptography, and Zero-Knowledge Proofs (ZKPs). Since some cryptographic primitives for symmetric encryption and hash computation, such as AES and SHA, for example, are quantum-resistant when used with sufficiently large sizes for keys and outputs, as mentioned before, we can consider them primitives of (Classical) Post-Quantum Cryptography as well.

The latter follows a very different approach when compared to the previously mentioned (Classical) Pre-Quantum and Post-Quantum Cryptography. Namely, it uses quantum information (in the form of quantum bits, also known simply as qubits, or quantum modes, also known simply as qumodes) in addition to the usual classical information. Moreover, it does not depend on any mathematical problems and computational hardness assumptions since it relies on postulates and fundamentals from Quantum Mechanics, as well as laws of physics. More specifically, it uses well-studied quantum phenomena such as Quantum Superposition and Quantum Entanglement, as well as postulates and properties ruled from them, such as Wave Interference, Heisenberg's Uncertainty Principle, Observer's Effect, Measurement Statistics (e.g., Non-Locality), No-Go Theorems (e.g., No-Cloning and No-Deleting Theorems), Monogamy of Entanglement, Special Relativity, among others. Since these quantum phenomena rule nature and reality themselves, they cannot simply be avoided or modified by an attacker, as well as retire any dependence of cryptographic primitives on future advances in computational power and respective continuous cryptoanalysis, reasons why it offers theoretical unconditional security and can achieve perfect security in the future. In particular, from Quantum Cryptography, we can build several cryptographic primitives, such as Quantum Key Distribution (QKD), Semi-Quantum Key Distribution (SQKD), Quantum Conference Key Agreement (QCKA), Semi-Quantum Conference Key Agreement (SQCKA), Quantum Oblivious Transfer (QOT), Quantum Digital Signature (QDS), Quantum Hashing, Quantum Bit Commitment, among others. However, due to the current state of quantum technology, the development and implementation of these physical apparatuses and systems still have their limitations, and there are gaps between its currently available setups and the ideal ones.

For a global perspective of the current state of Modern Cryptography regarding both classical and quantum contexts, we have the following summary table:

Type	Cryptographic Primitive	Security Strength (in bits)		Is Long-Term Secure?	Possible Countermeasures
		Classical	Quantum		
Symmetric Cryptography	*AES-128*	128	64	No	Use larger symmetric key sizes (with doubled sizes)
	AES-192	192	96	No	
	AES-256	256	128	Yes	
Cryptographic Hash/Digest Functions	*SHA3-224*	112	≈ 74	No	Use larger output digest sizes (with tripled sizes)
	SHA3-256	128	≈ 85	No	
	SHA3-384	192	128	Yes	
	SHA3-512	256	≈ 170	Yes	
	SHAKE128(d)	$\min(\frac{d}{2}, 128)$	$\min(\frac{d}{3}, 128)$	Yes, if $\frac{d}{3} \geq 128$	
	SHAKE256(d)	$\min(\frac{d}{2}, 256)$	$\min(\frac{d}{3}, 256)$	Yes, if $\frac{d}{3} \geq 128$	
Asymmetric Cryptography	*RSA*			No	No direct countermeasures and new solutions are needed, such as (Classical) Post-Quantum Cryptography or Quantum Cryptography
	DH			No	
	DSA			No	
	Elliptic Curve Diffie-Hellman (ECDH)			No	
	Elliptic Curve Digital Signature Algorithm (ECDSA)			No	
	ElGamal			No	

Even if this cryptographic threat does not seem to have immediate repercussions since the current quantum computers are still very limited in terms of processing capacity and noise, the whole scenario changes if we consider the Harvest Now - Decrypt Later (HNDL) paradigm, where an attacker can store information protected by these cryptographic primitives, waiting for a sufficiently powerful quantum computer to appear in the future, to then, be able to "break" such encryption and stole private information, what can lead to critical leakage of confidential information in military and health services, where that information usually needs to be kept as private for several years.

2 Lattice-based Cryptography

Lattice-based Cryptography is the generic term for cryptographic primitives that use geometric groups, known as lattices, in their construction or security proof. In the current state-of-the-art, we believe these mathematical problems involving lattices are computationally hard to be solved efficiently using either classical or quantum computers. For this reason, they are considered prominent candidates for future asymmetric cryptosystems of (Classical) Post-Quantum Cryptography.

Unlike more widely used and known public-key schemes such as the RSA (Rivest-Shamir-Adleman), DH (Diffie-Hellman), El Gamal, or other popular classical public-key cryptosystems of ECC (Elliptic-Curve Cryptography) that we currently use daily, which could, theoretically, be defeated using Shor's quantum algorithm for factoring on a sufficiently powerful quantum computer, some lattice-based constructions appear to be quantum-resistant. Namely, we consider many lattice-based cryptosystems to be (asymptotically) and provable secure, under the assumption that it is not known how to solve efficiently some well-studied lattice problems based on their worst-case computational hardness via worst-case-to-average-case reductions. Furthermore, these asymmetric cryptosystems have robust security proofs based on worst-case computational hardness, relatively efficient implementations, simplicity, and flexibility, as well as offer possible ways to achieve the well-desired Homomorphic Encryption, considered one of the holy grails of cryptography by the scientific community.

2.1 What is a Lattice?

First of all, it is necessary to define what is a lattice geometric group and structure. A lattice is a set of points in m-dimensional space with a periodic geometry. We give an example illustration of a lattice in the following Fig. 1:

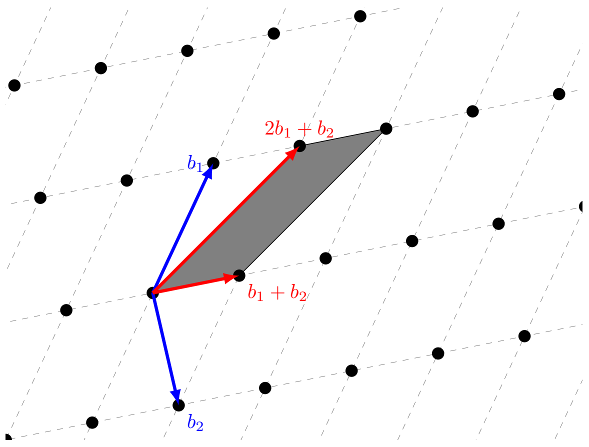

Fig. 1: An illustrative example of a lattice geometric group, with a possible choice of 2 bases and 2 linear combinations of them.

Formally, given n-linearly independent vectors from a basis $\{\vec{b_1}, \vec{b_2}, \ldots, \vec{b_n}\}$ of $\mathbb{R}^n$, a lattice structure $\mathcal{L} \subset \mathbb{R}^n$ generated by them is a linear combination of the basis vectors $\vec{b}$ and a given set of integer coefficients x, defined as follows:

$$\begin{aligned}\mathcal{L}(\vec{b_1}, \vec{b_2}, \ldots, \vec{b_n}) &= \left\{ \sum_{i=1}^{n} c_i \cdot \vec{b_i} : x_i \in \mathbb{Z},\ \vec{b_i} \in \mathbb{R}^n \right\} = \\ &= x_1 \cdot \vec{b_1} + x_2 \cdot \vec{b_2} + \cdots + x_n \cdot \vec{b_n} = \\ &= x_1 \cdot \begin{pmatrix} b_{(1,1)} \\ b_{(1,2)} \\ \vdots \\ b_{(1,n)} \end{pmatrix} + x_2 \cdot \begin{pmatrix} b_{(2,1)} \\ b_{(2,2)} \\ \vdots \\ b_{(2,n)} \end{pmatrix} + \cdots + x_n \cdot \begin{pmatrix} b_{(n,1)} \\ b_{(1,2)} \\ \vdots \\ b_{(n,n)} \end{pmatrix}\end{aligned} \tag{2.1}$$

Indeed, the basis can be represented by the matrix $B = \big(b_1, b_2, \ldots, b_n\big)$, having the basis vectors b_i as their columns, where $1 \leq i \leq n$. Using the matrix notation, a lattice structure generated by a matrix $B \in \mathbb{R}^{(n\times n)}$ can also be defined by $\mathcal{L}(B) = \{\ B \cdot x : B \in \mathbb{R}^{(n\times n)},\ x \in \mathbb{Z}^n\ \}$, using a matrix-multiplication as follows:

$$\begin{aligned}
\mathcal{L}(B) &= \Big\{\ B \cdot x : B \in \mathbb{R}^{(n\times n)},\ x \in \mathbb{Z}^n\ \Big\} = \\
&= \begin{pmatrix} b_{(1,1)} & b_{(2,1)} & \cdots & b_{(n,1)} \\ b_{(1,2)} & b_{(2,2)} & \cdots & b_{(n,2)} \\ \vdots & \vdots & \ddots & \vdots \\ b_{(1,n)} & b_{(2,n)} & \cdots & b_{(n,n)} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \\
&= \begin{pmatrix} b_{(1,1)} \cdot x_1 + b_{(2,1)} \cdot x_2 + \ldots + b_{(n,1)} \cdot x_n \\ b_{(1,2)} \cdot x_1 + b_{(2,2)} \cdot x_2 + \ldots + b_{(n,2)} \cdot x_n \\ \vdots \quad + \quad \vdots \quad + \ddots + \quad \vdots \\ b_{(1,n)} \cdot x_1 + b_{(2,n)} \cdot x_2 + \ldots + b_{(n,n)} \cdot x_n \end{pmatrix} = \\
&= \begin{pmatrix} b_{(1,1)} \cdot x_1 \\ b_{(1,2)} \cdot x_1 \\ \vdots \\ b_{(1,n)} \cdot x_1 \end{pmatrix} + \begin{pmatrix} b_{(2,1)} \cdot x_2 \\ b_{(2,2)} \cdot x_2 \\ \vdots \\ b_{(2,n)} \cdot x_2 \end{pmatrix} + \cdots + \begin{pmatrix} b_{(n,1)} \cdot x_n \\ b_{(n,2)} \cdot x_n \\ \vdots \\ b_{(n,n)} \cdot x_n \end{pmatrix} = \qquad (2.2) \\
&= \begin{pmatrix} b_{(1,1)} \\ b_{(1,2)} \\ \vdots \\ b_{(1,n)} \end{pmatrix} \cdot x_1 + \begin{pmatrix} b_{(2,1)} \\ b_{(2,2)} \\ \vdots \\ b_{(2,n)} \end{pmatrix} \cdot x_2 + \cdots + \begin{pmatrix} b_{(n,1)} \\ b_{(n,2)} \\ \vdots \\ b_{(n,n)} \end{pmatrix} \cdot x_n = \\
&= x_1 \cdot \begin{pmatrix} b_{(1,1)} \\ b_{(1,2)} \\ \vdots \\ b_{(1,n)} \end{pmatrix} + x_2 \cdot \begin{pmatrix} b_{(2,1)} \\ b_{(2,2)} \\ \vdots \\ b_{(2,n)} \end{pmatrix} + \cdots + x_n \cdot \begin{pmatrix} b_{(n,1)} \\ b_{(1,2)} \\ \vdots \\ b_{(n,n)} \end{pmatrix} = \\
&= \mathcal{L}(\vec{b_1}, \vec{b_2}, \ldots, \vec{b_n}) \text{ (see Equation (2.1))}
\end{aligned}$$

Given a particular $(n \times n)$ unimodular matrix U, the bases B and $B \cdot U$ generate the same lattice structure. In fact, we have the equality $\mathcal{L}(B) = \mathcal{L}(B')$ if and only if there is a unimodular matrix U such that $B' = BU$. In particular, any lattice structure admits multiple bases, and this mathematical fact represents the core of many lattice-based cryptographic applications and primitives proposed.

The determinant of a lattice structure is given by the absolute value of the determinant of the basis matrix B, that is, $\det(\mathcal{L}(B)) = |\det(B)|$. The value of the determinant is independent of the choice of the basis and geometrically corresponds to the inverse of the density of the points of the lattice structure in $\mathbb{R}^n$. The dual of a lattice structure $\mathcal{L} \in \mathbb{R}^n$ denoted $\mathcal{L}^*$, is the lattice structure given by the set of all column n-vectors $\vec{y} \in \mathbb{R}^n$ that satisfies $\langle \vec{x}, \vec{y} \rangle \in \mathbb{Z}$ for all column n-vectors $\vec{x} \in \mathcal{L}$. Additionally, we can see that for any $B \in \mathbb{R}^n$, $\mathcal{L}(B)^* = \mathcal{L}((B^{-1})^T)$. Therefore, we also obtain the equality $\det(\mathcal{L}^*) = \frac{1}{\det(\mathcal{L})}$.

Other lattice structures that are particularly important in Lattice-Based Cryptography are q-ary (modular) lattice groups. These q-ary lattice groups are lattice structures $\mathcal{L}$ satisfying $\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$ for some prime integer q. In other words, we determine the membership of a vector $x \in \mathcal{L}$ by $x \mod q$. Then, we define such lattice groups in one-to-one correspondence with linear codes in $\mathbb{Z}_q^n$. Most constructions of Lattice-Based Cryptography use q-ary (modular) lattice groups as their hard-on-average computational problem. Note that any lattice structure of integer elements $\mathcal{L} \subseteq \mathbb{Z}^n$ is a q-ary lattice group for some prime integer q. For example, whenever q is an integer multiple of the determinant $\det(\mathcal{L})$. However, we are more concerned with q-ary (modular) lattice groups with a prime integer q much smaller than $\det(\mathcal{L})$ in this specific configuration. Namely, given a matrix $X \in Z_q^{(m \times n)}$ for some integer numbers q, m, and n, we can mathematically define two n-dimensional q-ary lattice structures as follows:

1. $\Lambda_q(X) = \left\{\vec{y} \in \mathbb{Z}^n : \vec{y} = X^T \cdot \vec{s} \mod q, \text{ for some } \vec{s} \in \mathbb{Z}^m\right\}$;
2. $\Lambda_q^{\perp}(X) = \left\{\vec{y} \in \mathbb{Z}^n : X \cdot \vec{y} = \vec{0} \mod q\right\}$.

For the first case, we generate a specific q-ary lattice structure from the rows of matrix X. In the second case, a q-ary lattice structure contains all vectors being orthogonal modulo q to the rows of matrix X. In other words, the q-ary lattice structure in the first case corresponds to the linear code generated by the rows of the matrix X. The one from the second case corresponds to the linear code whose parity check matrix is X. It follows since these lattice structures are dual to each other, up to normalization. Namely, we have the following equalities:

1. $\Lambda_q(X) = q \cdot \Lambda_q^{\perp}(X)^*$;
2. $\Lambda_q^{\perp}(X) = q \cdot \Lambda_q(X)^*$.

There are several ways to use lattice structures to build cryptographic primitives for (Classical) Post-Quantum Cryptography that are not always obvious. One first milestone in this line of research is a paper from Miklós Ajtai in 1996, which defined the SIS (Short Integer Solution) problem, and related its average case complexity to the worst-case hardness of finding short vectors in every integer lattice structure, giving novel cryptographic directions, proposing lattice-based one-way functions and trapdoor functions. Then, in the same year, Miklós Ajtai, jointly with Cynthia Dwork, presented a probabilistic public-key cryptosystem based on the hardness of the SVP (Shortest Vector Problem). The hardness assumption they used in their cryptosystem is that it is secure unless it is possible to solve the worst case of a well-known lattice problem in polynomial time.

But before discussing this type of (Classical) Post-Quantum Cryptography in more detail, we first need to describe some mathematical problems that are computationally hard to solve, involving lattice structures at their core.

2.2 Lattice-based Problems

The constructions of cryptographic primitives for Lattice-based Cryptography use the presumed computational hardness of mathematical problems related to lattice structures, on which the Shortest Vector Problem (SVP) is the most known one and was the inspiration for other similar lattice-based problems. In this case, we start with a lattice structure represented by an arbitrary basis given as an input, and the main goal is to output the shortest non-zero vector in it.

Before entering the details of these problems, we need to introduce the notion of the shortest non-zero vector in a lattice structure $\mathcal{L}$ for a metric $\mathcal{M}$, given as:

$$\lambda(\mathcal{L}) = \min_{\vec{v} \,\in\, \mathcal{L} \setminus \{\vec{0}\}} ||\vec{v}||_{\mathcal{M}} \tag{2.3}$$

Another important constant associated with a lattice structure $\mathcal{L}$ is the covering radius. We define a covering radius as the smallest radius r for a family of closed spheres (or circles) with a center c_i, where $r > 0$, such that any $x \in R^n$ belong at least to one of those spheres (or circles), covering the entire vector space $\mathcal{V}$. In other words, we can define the covering radius r of a lattice structure $\mathcal{L}$ generated from a basis B as the maximum distance $||(\vec{x} - \mathcal{L}(B))||_{\mathcal{M}}$, for a given metric $\mathcal{M}$, where the vector $\vec{x}$ ranges over the linear span of the basis B. Thus, we can define the notion of covering radius r of a lattice structure $\mathcal{L}$ as follows:

$$r = \rho_M(\mathcal{L}(B)) = \max_{\vec{x} \,\in\, \mathbb{R}^n} ||(\vec{x} - \mathcal{L}(B))||_{\mathcal{M}} \tag{2.4}$$

Now, we can finally enumerate and briefly describe all the most currently known lattice-based mathematical and computational problems, in their *exact* forms:

1. ***Shortest Vector Problem*** (***SVP***):

   In the SVP instance, a basis of a vector space $\mathcal{V}$ and a metric $\mathcal{M}$, usually the Euclidean norm L^2, are given for a lattice structure $\mathcal{L}$ of size n as input. Then, the goal is to find the shortest non-zero vector in the space vector $\mathcal{V}$, as measured by $\mathcal{M}$, in the lattice structure $\mathcal{L}$. In other words, the solving algorithm should output a non-zero vector $\vec{v}$, such that $||\vec{v}||_{\mathcal{M}} = \lambda(\mathcal{L})$. This mathematical problem in its *exact* form, i.e., without any approximation (considering $\gamma(n) = 1$), is only known to be NP-Hard for randomized reductions. By contrast, the corresponding problem, when considering the uniform norm for the metric $\mathcal{M}$, is also known to be an NP-Hard problem.

2. ***Closest Vector Problem*** (***CVP***):

   In the CVP instance, a basis of a vector space $\mathcal{V}$ and a metric $\mathcal{M}$, usually the Euclidean norm L^2, are given for a lattice structure $\mathcal{L}$ of size n as input, together with a vector $\vec{v}$ in the vector space $\mathcal{V}$ but not necessarily in the lattice structure $\mathcal{L}$. The goal of the solving algorithm is to find the vector $\vec{v'}$ in the lattice structure $\mathcal{L}$ closest to the given vector $\vec{v}$, as measured by the given metric $\mathcal{M}$. It is also known that any computational hardness of an SVP instance implies the same computational hardness for a CVP instance. On the other hand, CVP instances are widely regarded, both in theory and in practice, as considerably harder computational problems than SVP instances.

3. ***Shortest Independent Vector Problem*** (***SIVP***):

   In the SIVP instance, a basis of a vector space $\mathcal{V}$ and a metric $\mathcal{M}$, usually the Euclidean norm L^2, are given for a lattice structure $\mathcal{L}$ of size n as input. The goal of the solving algorithm is to find a set of n linearly independent and short non-zero vectors $B' = \{\vec{b'}_1, \vec{b'}_2, ..., \vec{b'}_n\}$, so that $\max_i ||\vec{b'}_i||_{\mathcal{M}} \leq \max_B ||\vec{b_i}||_{\mathcal{M}}$, where $1 \leq i \leq n$ and $B = \{\vec{b_1}, \vec{b_2}, ..., \vec{b_n}\}$. In other words, in the best scenario, we seek to minimize the length of the longest vector in the original basis B and find a new basis B' that yields the same lattice structure $\mathcal{L}$. As far as we know, SIVP instances are also NP-Hard. However, they should generally be harder than SVP instances.

4. ***Shortest Basis Problem*** (***SBP***):

   In the SBP instance, a basis of a vector space $\mathcal{V}$ and a metric $\mathcal{M}$, usually the Euclidean norm L^2, are given for a lattice structure $\mathcal{L}$ of size n as input. The goal is to find an equivalent basis such that the length of the longest vector in that basis is as short as possible.

5. ***Covering Radius Problem*** (***CRP***):

   In the CRP instance, a basis of a vector space $\mathcal{V}$, and a metric $\mathcal{M}$, usually the Euclidean norm L^2, are given for a lattice structure $\mathcal{L}$ of size n as input. The goal is to find the smallest radius r', defined as a rational number, such each sphere (or circle) of radius r' centered around each point of the lattice structure $\mathcal{L}$ together can cover the entire space of the lattice structure. In other words, the goal of the solving algorithm is to find r' such that $r = \rho_{\mathcal{M}}(\mathcal{L}(B)) \leq r$, where $r, r' \in \mathbb{Q}$. Oppositely to the previous problems, we still do not know if CRP instances are in the NP-Hard complexity class. TODO - ver este problema melhor e com mais calma, confirmar se ja se sabe se este problema e NP-Hard ou nao. Wikipedia esta errado???

However, practical lattice-based public-key cryptosystems usually involve other approximate versions of the previously mentioned problems, which are outside the regime known to be NP-Hard. In addition, these NP-Hardness results only describe the worst-case asymptotic complexity of the problem, and we do not know how to apply this computational hardness directly to algebraically structured lattices. We can describe these approximate instances as follows:

1. γ-***Approximate Shortest Vector Problem*** (γ-***Approximate SVP***):
   In the γ-Approximate SVP instance, we consider the setup of the original SVP instance, but with an additional small approximation factor γ that works as a *relaxation* to the original *exact* problem. Namely, we also consider a function defined as $\gamma = \gamma(n) \geq 1$ that depends on the dimension n of the lattice structure $\mathcal{L}$. The goal of the solving algorithm is to find a non-zero lattice vector of a scaled length at most $\gamma \cdot \lambda(L)$, setting a lower bound for its shortest size that is greater than the one from the original *exact* problem.

2. γ-***Approximate Closest Vector Problem*** (γ-***Approximate CVP***):
   In the γ-Approximate CVP instance, we consider the setup of the original CVP instance, but with an additional small approximation factor γ that works as a *relaxation* to the original *exact* problem. Namely, we also consider a function defined as $\gamma = \gamma(n) \geq 1$ that depends on the dimension n of the lattice structure $\mathcal{L}$. The goal of the solving algorithm is to find the vector $\vec{v'}$ in the lattice structure $\mathcal{L}$ closest to the given vector $\vec{v}$ at distance at most γ, as measured by the given metric $\mathcal{M}$, setting a lower bound for the distance to the closest vector that is greater than the one from the *exact* problem.

3. γ-***Approximate Shortest Independent Vector Problem*** (γ-***Approximate SIVP***):
   In the γ-Approximate SIVP instance, we consider the setup of the original SIVP instance, but with an additional small approximation factor γ that works as a *relaxation* to the original *exact* problem. Namely, we also consider a function defined as $\gamma = \gamma(n) \geq 1$ that depends on the dimension n of the lattice structure $\mathcal{L}$. The goal of the solving algorithm is to find a set of n linearly independent and short non-zero vectors $B' = \{\vec{b'}_1, \vec{b'}_2, ..., \vec{b'}_n\}$, so that $\max_i ||\vec{b'}_i||_{\mathcal{M}} \leq \gamma \cdot \max_B ||\vec{b_i}||_{\mathcal{M}}$, where $1 \leq i \leq n$ and $B = \{\vec{b_1}, \vec{b_2}, ..., \vec{b_n}\}$. In other words, in the best scenario, we seek to minimize the length of the longest vector in the original basis B and find a new basis B', scaled by the given approximation factor γ, that yields the same lattice structure $\mathcal{L}$.

4. γ-***Approximate Shortest Basis Problem*** (γ-***Approximate SBP***):
   TODO

2.3 Algorithms for Lattice-based Problems

3 CRYSTALS (CRYptographic SuiTe for Algebraic LatticeS)

The CRYptographic SuiTe for Algebraic LatticeS (CRYSTALS) is a (classical) post-quantum cryptographic suite encompassing two cryptographic primitives, named Kyber and Dilithium. The former is a secure INDistinguishabile under Chosen Plaintext Attack (IND-CPA) asymmetric encryption algorithm and a secure INDistinguishable under adaptive Chosen Ciphertext Attack (IND-CCA2) Key Encapsulation Method (KEM), which we can also use as a Key Exchange protocol. The latter is a secure digital signature scheme and is strongly Existential UnForgeability under adaptive Chosen-Message Attack (EUF-CMA).

These cryptographic primitives have their security assumptions based on computational hardness from mathematical problems over module lattice groups, which are considered, until the current date, secure against attacks made from both classical and quantum computers. Both Kyber and Dilithium have been submitted to the contest of the Post-Quantum Cryptography Standardization project, being finalists of the 3^{rd} round of the standardization contest and also selected as some of the first-ever standard cryptographic primitives for the post-quantum era. Each of these cryptographic primitives has three variants offering different levels of security, which we will address in detail in this report.

3.1 CRYSTALS-Kyber

CRYSTALS-Kyber is a (classical) post-quantum asymmetric cryptosystem designed to be quantum-resistant to future cryptanalytic attacks performed by future powerful quantum computers, ensuring security in the classical contexts as well. This cryptographic primitive uses a variant of the already mentioned LWE problem on lattice groups as its primary trapdoor function. Namely, this asymmetric cryptosystem won the first spot in the NIST Post-Quantum Cryptography Standardization project and is already selected as a new standard to replace the currently used (classical) pre-quantum asymmetric cryptosystems that are vulnerable to attacks performed by quantum computers in the future.

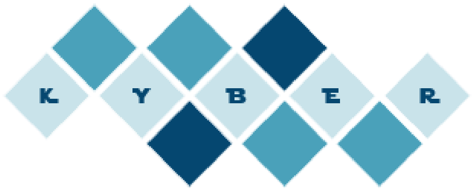

Fig. 2: Logotype of CRYSTALS-Kyber cryptographic primitive.

The design of the primitive CRYSTALS-Kyber has its roots inspired by the LWE-based asymmetric cryptosystem previously proposed by Oded Regev in 2005. Namely, we can improve the practical efficiency of such cryptosystems by employing the same probability distribution used for the noise to build a secret (i.e., a private key) and using a square matrix rather than a rectangular one as the public key. Another improvement is to use polynomial rings rather than integer numbers, as originally used in the NTRU cryptosystem, to define the RLWE and MLWE mathematical problems. From these two main ideas, we build CRYSTALS-Kyber as having its computational hardness assumption based on the MLWE problem. Since from the CRYSTALS-Kyber cryptosystem, we construct the IND-CCA KEM on top of the CPA asymmetric encryption algorithm, the formal definition of the latter will come in the first place.

3.1.1 CRYSTALS-Kyber IND-CPA Asymmetric Encryption Algorithm

Let k, d_t, d_u, and d_v be positive integer parameters, and recall that $n = 256$. Additionally, let $\mathcal{M} = \{0,1\}^{256}$ denote the messages space, where every message $m \in \mathcal{M}$ can be viewed as a polynomial in the algebraic ring $\mathcal{R}$ with binary coefficients (i.e., in $\{0,1\}$). Now, consider the public-key asymmetric encryption algorithm denoted by $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc} = \big(\texttt{Key\_Gen}, \texttt{Enc}, \texttt{Dec}\big)$. Here, the ciphertexts are of the form $ct = (u, v) \in \Big(\{0,1\}^{(256 \cdot k \cdot d_u)} \times \{0,1\}^{(256 \cdot d_v)}\Big)$. Then, in order to properly define the $\texttt{CRYSTALS}_{\texttt{Kyber}}$ asymmetric encryption algorithm $\texttt{Asym\_Enc}$, the sub-routines $\texttt{Key\_Gen}$, $\texttt{Enc}$, and $\texttt{Dec}$ are defined as:

Sub-routine 1 $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc.Key\_Gen}()$: Key Generation

```
Input: $(n, q, k, \eta)$
Output: $(k_{pub}, k_{priv})$
Require: $n = 256$, $q = 7681$, $k > 0$, $\eta > 0$
Ensure: $n \in \mathbb{Z}$, $q \in \mathbb{Z}$, $k \in \mathbb{Z}$, $\eta \in \mathbb{Z}$

1: $\rho, \sigma \leftarrow \{0,1\}^n = \{0,1\}^{256}$
2: $A \sim \mathcal{R}_q^{(k \times k)} = \mathcal{R}_{7681}^{(k \times k)} := \texttt{Sample}(\rho)$
3: $(s, e) \sim \big(\beta_\eta^k \times \beta_\eta^k\big) := \texttt{Sample}(\sigma)$
4: $t := \texttt{Compress}_q(A \cdot s + e, d_t)$

5: $k_{pub} := (t, \rho)$
6: $k_{priv} := s$

7: return $(k_{pub}, k_{priv})$
```

Sub-routine 2 $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc.Enc}(k_{pub} = (t, \rho),\ m)$: Asymmetric Encryption

```
Input: $(n, q, k, \eta, k_{pub}, m)$
Output: $ct$
Require: $n = 256,\ q = 7681,\ k > 0,\ \eta > 0$
Ensure: $n \in \mathbb{Z},\ q \in \mathbb{Z},\ k \in \mathbb{Z},\ \eta \in \mathbb{Z},\ m \in \mathcal{M}$

1: $r \leftarrow \{0,1\}^n = \{0,1\}^{256}$
2: $t := \texttt{Decompress}_q(t, d_t)$
3: $A \sim \mathcal{R}_q^{(k \times k)} = \mathcal{R}_{7681}^{(k \times k)} := \texttt{Sample}(\rho)$
4: $(r, e_1, e_2) \sim \left(\beta_\eta^k \times \beta_\eta^k \times \beta_\eta\right) := \texttt{Sample}(r)$
5: $u := \texttt{Decompress}_q(A^T \cdot r + e_1, d_u)$
6: $u := \texttt{Decompress}_q(t^T \cdot r + e_2 + \lceil \frac{q}{2} \rfloor \cdot m, d_v)$
7: $ct := (u, v)$
8: return $ct$
```

Sub-routine 3 $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ $\texttt{.Dec}(k_{priv} = s,\ ct = (u, v))$: Asymmetric Decryption

```
Input: $(n, q, k, \eta, k_{priv}, ct)$
Output: $pt$
Require: $n = 256,\ q = 7681,\ k > 0,\ \eta > 0$
Ensure: $n \in \mathbb{Z},\ q \in \mathbb{Z},\ k \in \mathbb{Z},\ \eta \in \mathbb{Z}$

1: $u := \texttt{Decompress}_q(u, d_u)$
2: $v := \texttt{Decompress}_q(v, d_v)$
3: $pt := \texttt{Compress}_q(v - s^T \cdot u, 1)$
4: return $pt$
```

Correctness: In order to show the correctness of the $\texttt{CRYSTALS}_{\texttt{Kyber}}$ asymmetric encryption algorithm `Asym_Enc`, we need to demonstrate that the same has a negligible decryption error δ, concluding that the algorithm is $(1 - \delta)$-correct.

Theorem 1: Let k be a positive integer parameter. Also, let s, e, r, e_1, e_2 be random variables that have the same probability distribution as in the previously defined sub-routines `Key_Gen` and `Enc` from $\texttt{CRYSTALS}_{\texttt{Kyber}}$ asymmetric encryption algorithm `Asym_Enc`. Then, let $c_t \leftarrow \psi_{d_t}^k$, $c_u \leftarrow \psi_{d_u}^k$, and $c_v \leftarrow \psi_{d_v}$, be distributed according to the probability distribution ψ defined as follows:

– Let ψ_d^k be the following probability distribution over polynomial ring $\mathcal{R}$:
  1. Choose uniformly-random $y \leftarrow \mathcal{R}^k$;
  2. **return** $(\ y - \texttt{Decompress}_{\text{q}}(\texttt{Compress}_{\text{q}}(y, d), d)\ )\mod \pm q$.
     TODO - confirmar mod com superscript +/- na pag 5 do paper ???

– Denote the decryption error δ as follows:
  - $\delta = \Pr\Big[\ ||\ e^T \cdot r + e_2 + c_v - s^T \cdot e_1 + c_t^T \cdot r - s^T \cdot c_u\ ||_\infty \geq \lceil \frac{q}{4} \rfloor\ \Big]$

– Then, we can conclude that $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ is $(1 - \delta)$-correct.

3.1.2 CRYSTALS-Kyber IND-CCA Key Encapsulation Method

First, let $G : \{0,1\}^* \mapsto \{0,1\}^{(2\times256)}$ and $H : \{0,1\}^* \mapsto \{0,1\}^{256}$ be cryptographic hash functions. Then, consider the public-key asymmetric KEM denoted by $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM} = \big(\texttt{Key\_Gen}, \texttt{Encaps}, \texttt{Decaps}\big)$, where `Key_Gen` is very similar to the sub-routine defined previously in $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ with the difference that k_{priv} also contains $k_{pub} = (t, \rho)$ and a secret random value z with 256 bits as well. We obtain this asymmetric KEM $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM}$ algorithm by applying a KEM variant with "implicit rejection" of the Fujisaki-Okamoto transform to the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm. The sub-routine $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM.Decaps}$ never returns $\perp$ (i.e., *absurdum*). Instead, in the case of re-encryption failure, a pseudo-random key $K := H(z, ct)$ is returned, where z is a random secret seed. Then, in order to properly define the $\texttt{CRYSTALS}_{\texttt{Kyber}}$ KEM algorithm `KEM`, the sub-routines `Key_Gen`, `Encaps`, and `Decaps` are defined as follows:

Sub-routine 4 $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM.Encaps}(k_{pub} = (t, \rho))$: Key Encapsulation

```
Input: $(n, q, k, \eta, k_{priv}, m)$
Output: $encaps$
Require: $n = 256$, $q = 7681$, $k > 0$
Ensure: $n \in \mathbb{Z},\ q \in \mathbb{Z},\ k \in \mathbb{Z},\ \eta \in \mathbb{Z}$

1: $m \leftarrow \{0,1\}^n = \{0,1\}^{256}$
2: $\big(\hat{K}, r\big) := G(H(k_{pub}), m)$
3: $(u, v) := \texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc.Enc}(k_{pub},\ m,\ r)$
4: $ct := (u, v)$
5: $K := H(\hat{K}, H(ct))$
6: $encaps := (ct, K)$
7: return $encaps$
```

Sub-routine 5 `CRYSTALS`$_{\texttt{Kyber}}$`.KEM.Decaps`$(k_{priv} = (s, z, t, \rho),\ ct = (u, v))$: Key Decapsulation

Input: $(n, q, k, \eta, k_{priv}, m)$
Output: *encapsulation*
Require: $n = 256$, $q = 7681$, $k > 0$
Ensure: $n \in \mathbb{Z},\ q \in \mathbb{Z},\ k \in \mathbb{Z},\ \eta \in \mathbb{Z}$

```
1: $m' :=$ CRYSTALS$_{\texttt{Kyber}}$.Asym_Enc.Dec$(s,\ ct)$
2: $\left(\hat{K}', r'\right) := G(H(k_{pub}), m')$
3: $(u', v') :=$ CRYSTALS$_{\texttt{Kyber}}$.Asym_Enc.Enc$((t, \rho),\ m';\ r')$
4: if $(u', v') = (u, v)$ then
5:     $K := H(\hat{K}', H(ct))$
6: else
7:     $K := H(z, H(ct))$
8: end if
9: return $K$
```

Correctness: If the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm is $(1-\delta)$-correct and the cryptographic hash function G is a random oracle, then the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM}$ algorithm is also $(1-\delta)$-correct.

Security: The following concrete security statement proves the security of the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm when we model the cryptographic hash functions G and H as random oracles. Namely, for the concrete security bounds, the KEM variant of the previously mentioned Fujisaki-Okomoto transform is considered, and a non-zero correctness (or decryption) error δ is also assumed.

Theorem 3: For any classical adversary $\texttt{A}$ that makes at most q_{RO} many queries to the random oracles G and H, and q_{DO} queries to the decryption oracle, there is another classical adversary $\texttt{B}$ such that:

$$\mathbf{Adv}^{\mathrm{CCA}}_{\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM}}(\texttt{A}) \leq 3 \cdot \mathbf{Adv}^{\mathrm{CPA}}_{\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}}(\texttt{B}) + q_{RO} \cdot \delta + \frac{(3 \cdot q_{RO})}{2^{256}}$$

As for security in the Quantum Random Oracle Model (QROM), we can use the fact that $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm is ONE-WAY CPA secure and *sparse pseudo-random* to prove that $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM}$ algorithm is INC-CCA secure. Namely, sparse pseudo-randomness is a slightly stronger security notation than the IND-CPA security and essentially states the following properties:

1. A properly generated ciphertext is *pseudo-random* (i.e., it is computationally indistinguishable from a random high-entropy binary string).
2. A random binary string is, with high probability, not a properly generated ciphertext.

The proof of Theorem 2 shows that the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$' algorithm (i.e., the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm variant without compressing the vector t) is tightly pseudo-random under the Module-LWE computational hardness assumption. Concretely, the pseudo-randomness advantage is bounded by:

$$\mathbf{Adv}^{\mathrm{PR}}_{\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}'}(\texttt{A}) \leq 2 \cdot \mathbf{Adv}^{\mathrm{MLWE}}_{(k+1,k,\eta)}(\texttt{B})$$

Now, we can argue again that the same mathematical bound holds for the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm. Namely, the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.Asym\_Enc}$ algorithm has its sparseness property trivially fulfilled since the set of properly generated ciphertexts is a sparse subset of the original ciphertext space $\left(\{0,1\}^{(256 \cdot k \cdot d_u)} \times \{0,1\}^{d_v}\right)$.

Then, we can also provide the following concrete security statement in the QROM, considering a quantum adversary, for a correctness (decryption) error δ.

Theorem 4: For any quantum adversary `A` that makes at most q_{RO} many queries to the quantum random oracles G and H, and at most q_{DO} (classical) queries to the decryption oracle, there exists a quantum adversary `B` such that:

$$\mathbf{Adv}^{\mathrm{CCA}}_{\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.KEM}}(\mathtt{A}) \leq 8 \cdot q_{RO}^2 \cdot \delta + 4 \cdot q_{RO} \cdot \sqrt{\mathbf{Adv}^{\mathrm{PR}}_{\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.Asym\_Enc}}(\mathtt{B})}$$

Unfortunately, the above security bound is non-tight, and we can only use it as an asymptotic indication of the CCA security of the $\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.KEM}$ algorithm in QROM. However, we can derive a tight security bound in the QROM from a non-standard security assumption, namely that a deterministic version of $\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.Asym\_Enc}$ algorithm is sparse pseudo-random in the QROM. The $\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.Asym\_Enc}$ algorithm is already deterministic, but the randomness vector r used in the encryption process is derived deterministically from the message m by performing $r := G(m)$. In the Classical Random Oracle Model (CROM), this assumption is tight and implied by the IND-CPA security of $\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.Asym\_Enc}$ algorithm, but the security reduction in the QROM is non-tight. For this reason, we need to have the term $q_{RO} \cdot \sqrt{\mathbf{Adv}^{\mathrm{PR}}_{\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.Asym\_Enc}}(\mathtt{B})}$ in Theorem 4).

Hashing k_{pub} ***into*** $\hat{K}$***:*** The transformation used in the $\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.Asym\_Enc}$ algorithm is essentially the Fujisaki–Okamoto transform with a small adjustment. Most specifically, we need to hash the public key k_{pub} by computing $\hat{K} := H(k_{pub})$. Namely, this adjustment has two effects. First, this adjustment makes the $\mathtt{CRYSTALS}_{\mathtt{Kyber}}\mathtt{.KEM}$ algorithm contributory since the final shared key K does not depend only on the input of one of the two parties involved. Second, this adjustment also offers multi-target protection. In particular, consider an attacker who searches through many m to find one particular message that produces a decryption failure with a considerable probability. Such decryption failure of a legitimate ciphertext would leak some information about the secret key. In the pre-quantum context, the negligible decryption error delta guarantees that it is not a practical attack. In the post-quantum context, an attacker can use Grover's algorithm to speed up the search for that message m. However, the attacker will have difficulty encoding such message m, which may produce a decryption failure with a considerable probability in Grover's quantum oracle. In particular, this problem is equivalent to identifying noise vectors that likely have a large inner product with the tuple of vectors (s, e). Eventually, the best strategy is to search for a message m that produces noise vectors with a large norm. This attack probably does not offer any better performance than a brute-force attack performed by Grover's algorithm for searching the shared key K with 256 bits. However, applying a cryptographic hash function and transforming k_{pub} into $\hat{K}$ ensures that an attacker would not be able to use pre-computed values (i.e., pre-images) of messages m against multiple targets.

3.1.3 CRYSTALS-Kyber Key Exchange Protocol

Let $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM} = \big(\texttt{Key\_Gen}, \texttt{Encaps}, \texttt{Decaps}\big)$ be the IND-CCA secure KEM algorithm described before. Now, we denote the Key Exchange protocol as $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KE}$, and we obtain it as a direct cryptographic application of the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM}$ algorithm. In Key Exchange protocols using a KEM algorithm, a cryptographic hash function is commonly applied to all messages each participant sends and receives in order to compose the final secret key. Namely, we compute a cryptographic hash function on the public key k_{pub} to obtain the pre-shared key $\hat{K}$, and the ciphertext is then cryptographically hashed into the final secret key K. Therefore, the shared secret key K obtained during the Key Exchange protocol already includes these exchanged messages of each participant. We illustrate the steps of the $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KE}$ algorithm in Fig. 3:

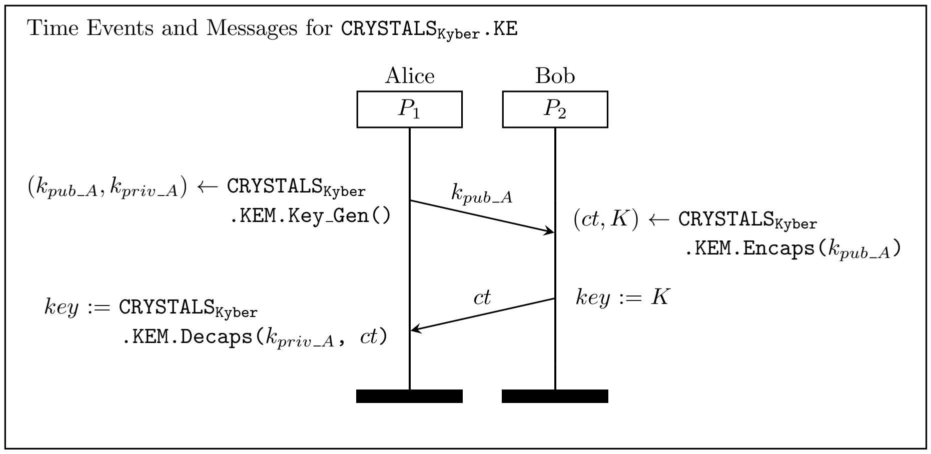
Time Events and Messages for $\texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KE}$
Alice
Bob
P_1
P_2
$(k_{pub\_A}, k_{priv\_A}) \leftarrow \texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM.Key\_Gen()}$
k_{pub_A}
$(ct, K) \leftarrow \texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM.Encaps(}k_{pub\_A}\texttt{)}$
ct
$key := K$
$key := \texttt{CRYSTALS}_{\texttt{Kyber}}\texttt{.KEM.Decaps(}k_{priv\_A}\texttt{,}\ ct\texttt{)}$

Fig. 3: Schematic of the CRYSTALS-Kyber Key Exchange protocol, with the respective Time Events and Messages.

3.2 CRYSTALS-Dilithium

CRYSTALS-Dilithium is a (classical) post-quantum digital signature scheme developed to be closely related and complementary to the CRYSTALS-Kyber asymmetric cryptosystem previously presented, being another component of CRYSTALS. Therefore, this digital signature scheme is also quantum-resistant and intended to be computationally secure against future cryptanalytic attacks performed by future quantum computers with considerable processing power, also ensuring security in the classical contexts. This cryptographic primitive also uses the computational hardness of lattice mathematical problems over module lattices as its trapdoor function and has most of its security notions based on it. For this reason, it is considered strongly secure under a Chosen-Message Attack (CMA). Its security notion also means that an adversary with access to a signing oracle cannot produce a signature of a message whose signature he has not yet seen nor produce a different signature of a specific message he already saw signed. Moreover, CRYSTALS-Dilithium is one of the first selected standards for digital signatures in the NIST Post-Quantum Cryptography Standardization project, together with FAst-fourier Lattice-based Compact signatures Over Ntru (FALCON) and Stateless Practical Hash-based Incredibly Nice Cryptographic Signatures$^+$ (SPHINCS$^+$) to replace the currently used (classical) pre-quantum digital signatures that are insecure against to attacks from quantum computers.

Fig. 4: Logotype of CRYSTALS-Dilithium cryptographic primitive.

The core design of the primitive CRYSTALS-Dilithium is based mainly on a cryptographic technique known as “Fiat-Shamir with Aborts”, proposed by Vadim Lyubashevsky in 2009. This technique uses rejection sampling in order to make lattice-based Fiat-Shamir digital signature schemes compact and secure. It is possible to create digital signature schemes applying this approach, and with small signature sizes, based on the hardness assumption of the mathematical problem used in the NTRU lattice-based asymmetric cryptosystem that crucially uses random samplings from Gaussian probability distributions. However, since it is hard to implement samplings from the Gaussian probability distribution, the CRYSTALS-Dilithium digital signature scheme uses only the uniform probability distribution to obtain those random samplings. This cryptographic primitive improved the previous most efficient digital signature scheme that only uses the uniform probability distribution, proposed by Shi Bai and Steven Galbraith in 2013. Namely, CRYSTALS-Dilithium digital signature scheme uses a new technique that shrinks the size of the public key by more than half. This digital signature scheme has the smallest public key and signature sizes of any known lattice-based digital signature scheme that only uses a uniform probability distribution for the random samplings. However, it may have larger sizes for the signature, public key, and private key when we compare it to the other digital signature schemes NIST also selected as new standards in the NIST Post-Quantum Cryptography Standardization project. Namely, it has larger public and private keys than SPHINCS$^+$ hash-based digital signature scheme in spite of having shorter signatures. Additionally, the FALCON lattice-based digital signature scheme has smaller sizes for the signature, public key, and private key than the CRYSTALS-Dilithium digital signature scheme. However, the former uses a Gaussian probability distribution for the random samplings.

3.2.1 CRYSTALS-Dilithium Digital Signature

The CRYSTALS-Dilithium digital signature scheme uses a pseudo-randomness in the signing procedure generated using the well-known cryptographic hash function Secure Hash Algorithm and KECCAK - 256 (SHAKE-256) from the Secure Hash Algorithm - 3 (SHA-3) family as a deterministic function of the message to be signed and a small secret key. Since most of the signing procedure may need to be repeated several times in a computational loop until we build a final signature, a counter is also appended to the input of the SHAKE-256 cryptographic hash function to make its output differ with each signing attempt of the same message m. Because each (possibly long) message m may require several iterations to be signed, we compute an initial hash digest of the message m using a collision-resistant hash function. Then, we use the respective hash digest as the input of the cryptographic hash function in the computational loop throughout the signing procedure (see steps 2 and 10 of the sub-routine 7).

Additionally, one of the primary design improvements offered by the CRYSTALS-Dilithium digital signature scheme over the previous ones is the fact that the size of the public key is approximately halved, at the expense of little longer signatures with fewer than a hundred extra bytes. In order to perform this size reduction, the key generation sub-routine outputs $t_1 := \texttt{Power2Round}_q(t, d_t)$ as the public key. Additionally, one of the primary design improvements of the CRYSTALS-Dilithium digital signature scheme over the previous ones is the fact that the size of the public key is approximately halved, at the expense of little longer signatures with fewer than a hundred extra bytes. In order to perform this size reduction, the key generation sub-routine outputs $t_1 := \texttt{Power2Round}_q(t, d_t)$ as the public key. From this size reduction, the public key requires $\lceil \log(q) \rceil - d_t$ bits per coefficient of the ring polynomial. In the instantiations proposed by the authors, we assume $q \approx 2^{23}$ and $d_t = 13$, which means that instead of 23 bits in each coefficient of the public key, there is instead only 10 bits (i.e., $\lceil \log(q) \rceil - d_t = \lceil \log(2^{23}) \rceil - 13 = 23 - 13 = 10$), what results in a compression of the size of the public key generated (see the step 6 of the sub-routine 6).

The main complication of not having the entire vector t in the public key is that it is no longer possible to exactly compute w_1 in the signature verification sub-routine. The signature verification will need the high-order bits of $A \cdot z - c \cdot t$ to compute this, but it can only compute $A \cdot z - c \cdot t_1 \cdot 2^{d_t} = A \cdot z - c \cdot t + c \cdot t_0$. Even though the high-order bits of $c \cdot t_0$ are 0, its presence in the sum creates "carry" bits which may affect the higher bits. The signer of the message thus sends these "carries" as a hint to the verifier of the signature. Heuristically, based on the parameter choices proposed by the authors, there should not be more than ω positions in which a "carry" is caused. Therefore, the signer of the message simply sends the positions in which these "carries" occur, which allows the verifier to compute the high order bits of $A \cdot z - c \cdot t$ (these are the extra bytes in CRYSTALS-Dilithium compared to other previous similar digital signatures).

In order to keep the size of the public and private keys small, both the Sign and Verify procedures begin with extracting the matrix A (or more accurately, its NTT domain representation $\hat{A}$) from the pseudo-random seed ρ. If storage space is not a factor, then the NTT domain representation matrix $\hat{A}$ can be pre-computed and be part of the public/private key. The signer of the message m can additionally pre-compute the NTT domain representations of s, e, and t_0 to slightly speed up the signing sub-routine. On the other hand, if the signer of the message m wants to store a private key small as possible, he only needs to store a secret seed ζ with 32 bytes, which is used to generate the randomness to create ρ, K, s, and e, in the key generation sub-routine. Furthermore, we can also keep the memory for intermediate computations low by only keeping the parts of the NTT domain representation that we are currently working with.

We can use the CRYSTALS-Dilithium cryptographic primitive to produce deterministic or randomized digital signatures. The former always gives the same digital signature output σ for a particular message since we derive the random seed ρ' from the hashed message μ and a key K. The latter does not produce the same digital signature output σ for a particular message since we choose the random seed ρ' completely at random (see the step 5 of the sub-routine 7).

In situations where some side channels that exploit determinism are possible, we may use the randomized variant of this digital signature scheme. Another case where we may want to avoid determinism is when the signer party does not wish to reveal the content of the message which that party is signing. When the digital signature scheme is deterministic, despite not happening timing leakage of the secret key, there is a timing leakage of the hashed message μ. For this case, and since we derive the randomness of this digital signature from the hashed message μ, the number of aborts for a particular message will always be the same.
To prove the correctness and security of this digital signature scheme, we need to define a set of lemmas that highlight the properties of its supporting algorithms.

Lemma 1: Suppose that q and α are positive integer numbers, where $q > 2 \cdot \alpha$, $q \equiv 1 \pmod{\alpha}$, and α is an even number. Let r and z be vectors containing elements in R_q where $||z||_\infty \leq \frac{\alpha}{2}$, and let h, h' be vectors of bits. Then, the $\texttt{HighBits}_q$, $\texttt{MakeHint}_q$, and $\texttt{UseHint}_q$ algorithms satisfy the following properties:

1. $\texttt{UseHint}_q(\texttt{MakeHint}_q(z,\, r,\, \alpha),\, r,\, \alpha) = \texttt{HighBits}_q(r + z,\, \alpha)$;
2. Let $v_1 = \texttt{UseHint}_q(h,\, r,\, \alpha)$. Then, we also verify that $||r - v_1 \cdot \alpha||_\infty \leq \alpha + 1$. Furthermore, if the Hamming Weight of the vector h is equal to ω, then all except at most w coefficients of the result $r - v_1 \cdot \alpha$ will have magnitude at most $\frac{\alpha}{2}$ after centered reduction modulo q using modular multiplications;
3. For any vectors h and h', if we verify $\texttt{UseHint}_q(h, r, \alpha) = \texttt{UseHint}_q(h', r, \alpha)$, then we can conclude the binary vectors h and h' are equal (i.e., $h = h'$).

Lemma 2: If we verify that $||s||_\infty \leq \beta$ and $||\texttt{LowBits}_q(r, \alpha)||_\infty < \frac{\alpha}{2} - \beta$, then we can also verify that the equality $\texttt{HighBits}_q(r, \alpha) = \texttt{HighBits}_q(r + s, \alpha)$ holds.

Lemma 3: Let $(r_1, r_0) = \texttt{Decompose}_q(r, \alpha)$, $(w_1, w_0) = \texttt{Decompose}_q(r + s, \alpha)$, and $||s||_\infty \leq \beta$. Then, $||s + r_0||_\infty < \frac{\alpha}{2} - \beta \iff w_1 = r_1 \wedge ||w_0||_\infty < \frac{\alpha}{2} - \beta$ holds.

Regarding the details of the correctness property of the CRYSTALS-Dilithium digital signature scheme, we have the following enumerated mathematical proof:

- If $\left|\left|c \cdot t_0\right|\right|_\infty < \gamma_2$, then by Lemma 1 introduced before, we know that $\texttt{UseHint}_q(h, w - c \cdot e + c \cdot t_0, 2 \cdot \gamma_2) = \texttt{HighBits}_q(w - c \cdot e, 2 \cdot \gamma_2)$;
- Since we verify that the results $w = A \cdot y$ and $t = A \cdot s + e$ hold, we have that $w - c \cdot e = A \cdot y - c \cdot e = A \cdot (z - c \cdot s) - c \cdot e = A \cdot z - c \cdot t$ holds, and we also verify that the equality $w - c \cdot e + c \cdot t_0 = A \cdot z - c \cdot t_1 \cdot 2^{d_t}$ is valid.
- Then, the party that verifies if the digital signature is valid, is also able to compute $\texttt{UseHint}_q(h, A \cdot z - c \cdot t_1 \cdot 2^{d_t}, 2 \cdot \gamma_2) = \texttt{HighBits}_q(w - c \cdot e, 2 \cdot \gamma_2)$;
- Furthermore, because we set β such we verify that $||c \cdot e||_\infty \leq \beta$ and the signer party checks that $\texttt{LowBits}_q(w - c \cdot e, 2 \cdot \gamma_2) < \gamma_2 - \beta$ (see the steps 13 and 14 of the sub-routine 7). Then, Lemma 2 implies the following equalities:

$$\begin{aligned} \texttt{HighBits}_q(w - c \cdot e, 2 \cdot \gamma_2) &= \texttt{HighBits}_q(w - c \cdot e + c \cdot e, 2 \cdot \gamma_2) \\ &= \texttt{HighBits}_q(w, 2 \cdot \gamma_2) = w_1 \end{aligned}$$

- Therefore, the vector w_1' computed by the verifier party to be the input of the hash function is equal to the vector w_1 calculated by the signer party (see the step 4 and 5 of the sub-routine 8, as well as the step 9 of the sub-routine 7). And thus, the verification procedure will always accept the digital signature in this case, verifying the correctness property.

Sub-routine 6 `CRYSTALS`$_{\texttt{Dilithium}}$`.Key_Gen()`: Key Generation

Input: $(n, q, k, \eta, k_{priv}, m)$
Output: (k_{pub}, k_{priv})
Require: $n = 256$, $q = 7681$, $k > 0$
Ensure: $n \in \mathbb{Z}$, $q \in \mathbb{Z}$, $k \in \mathbb{Z}$, $\eta \in \mathbb{Z}$

```
1: $\zeta \leftarrow \{0,1\}^n = \{0,1\}^{256}$
2: $(\rho, \rho', K) \sim \Big( \{0,1\}^n \times \{0,1\}^{(2\cdot n)} \times \{0,1\}^n \Big) =$
       $= \big( \{0,1\}^{256} \times \{0,1\}^{512} \times \{0,1\}^{256} \big) := H(\zeta)$
3: $A \sim \mathcal{R}_q^{(k\times\ell)} = \mathcal{R}_{7681}^{(k\times\ell)} := \texttt{Sample}(\rho)$
4: $(s, e) \sim \big(\upsilon_\eta^\ell \times \upsilon_\eta^k\big) := \texttt{Sample}(\rho')$
5: $t := A \cdot s + e$
6: $(t_1, t_0) := \texttt{Compress}_q(t, d_t) = \texttt{Power2Round}_q(t, d_t)$
7: $t_r \sim \{0,1\}^n = \{0,1\}^{256} := H\,(\rho \,||\, t_1)$
8: $k_{pub} := (\rho, t_1)$
9: $k_{priv} := (\rho, K, t_r, s, e, t_0)$
10: return $(k_{pub}, k_{priv})$
```

Sub-routine 7 $\texttt{CRYSTALS}_{\texttt{Dilithium}}.\texttt{Sign}(k_{priv}, M)$: Message Signing

Input: $(n, q, k, \eta, k_{priv}, m)$
Output: (k_{pub}, k_{priv})
Require: $n = 256,\ q = 7681,\ k > 0$
Ensure: $n \in \mathbb{Z},\ q \in \mathbb{Z},\ k \in \mathbb{Z},\ \eta \in \mathbb{Z}$

```
1:  $A \sim \mathcal{R}_q^{(k \times \ell)} = \mathcal{R}_{7681}^{(k \times \ell)} := \texttt{Sample}(\rho)$
2:  $\mu \sim \{0,1\}^{(2 \cdot n)} = \{0,1\}^{512} := H\left(t_r \mid\mid M\right)$
3:  $\kappa := 0$
4:  $(z, h) := \perp$
5:  $\rho' \sim \{0,1\}^{(2 \cdot n)} = \{0,1\}^{512} := H\left(K \mid\mid \mu\right)$ (or $\sigma \leftarrow \{0,1\}^{(2 \cdot n)} = \{0,1\}^{512}$
        for randomized signing)
6:  while $(z, h) = \perp$ do
7:      $y \sim \tilde{v}_{\gamma_1}^{\ell} := \texttt{SampleMask}(\sigma, \kappa)$
8:      $w := A \cdot y$
9:      $w_1 := \texttt{HighBits}_q(w, 2 \cdot \gamma_2)$
10:     $\tilde{c} \sim \{0,1\}^{n} = \{0,1\}^{256} := H\left(\mu \mid\mid w_1\right)$
11:     $c \sim B_\tau := \texttt{SampleInBall}(\tilde{c})$
12:     $z := y + c \cdot s$
13:     $r_0 := \texttt{LowBits}_q(w - c \cdot e, 2 \cdot \gamma_2)$
14:     if $\left( [\![\, ||z||_\infty \geq \gamma_1 - \beta \,]\!] \text{ or } [\![\, ||r_0||_\infty \geq \gamma_2 - \beta \,]\!] \right)$ then
15:         $(z, h) := \perp$
16:     else
17:         $h := \texttt{MakeHint}_q(-c \cdot t_0, w - c \cdot e + c \cdot t_0, 2 \cdot \gamma_2)$
18:         if $\left( [\![\, ||c \cdot t_0||_\infty \geq \gamma_2 \,]\!] \text{ or } [\![\, weight(h) > \omega \,]\!] \right)$ then
19:             $(z, h) := \perp$
20:         end if
21:     end if
22:     $\kappa := \kappa + \ell$
23: end while
24: return $\sigma = (\tilde{c}, z, h)$
```

Sub-routine 8 $\texttt{CRYSTALS}_{\texttt{Dilithium}}.\texttt{Verify}(k_{priv}, M, \sigma = (\tilde{c}, z, h))$: Signature Verification

Input: $(n, q, k, \eta, k_{priv}, m)$
Output: (k_{pub}, k_{priv})
Require: $n = 256,\ q = 7681,\ k > 0$
Ensure: $n \in \mathbb{Z},\ q \in \mathbb{Z},\ k \in \mathbb{Z},\ \eta \in \mathbb{Z}$

```
1: $A \sim \mathcal{R}_q^{(k \times \ell)} = \mathcal{R}_{7681}^{(k \times \ell)} := \texttt{Sample}(\rho)$
2: $\mu \sim \{0,1\}^{(2 \cdot n)} = \{0,1\}^{512} := H\left(H(\rho, t_1) \mid\mid M\right)$
3: $c := \texttt{SampleInBall}(\tilde{c})$
4: $w_1' := \texttt{UseHint}_q(h, A \cdot z - c \cdot t_1 \cdot 2^{d_t}, 2 \cdot \gamma_2)$
5: $is\_valid := \left( [\![\, ||z||_\infty < \gamma_1 - \beta \,]\!] \text{ and } [\![\, \tilde{c} = H(\mu \mid\mid w_1') \,]\!] \text{ and } [\![\, weight(h) \leq \omega \,]\!] \right)$
6: return $is\_valid$
```