Sección de Matemáticas
Universidad de La Laguna

Claudia Tarajano Afonso

Esquema de cifrado CRYSTALS-Kyber

CRYSTALS-Kyber encryption scheme

Trabajo Fin de Grado
Grado en Matemáticas
La Laguna, Mayo de 2023

Dirigido por
Pino Teresa Caballero Gil

Pino Teresa Caballero Gil
Departamento de Ingeniería
Informática y de Sistemas
Universidad de La Laguna
38200 La Laguna, Tenerife

Agradecimientos

Quiero agradecer a todas las personas que han estado conmigo a lo largo de esta etapa, en particular, muchas gracias a mi tutora, Pino, ya que me ha abierto puertas a un mundo, para mí, desconocido.

Agradecer a mis padres y mi hermano por su apoyo y confianza constantes, por estar ahí siempre que lo he necesitado y por aguantarme día tras día.

Agradecer a toda mi familia por estar conmigo una etapa más, y a mis amigos, en especial, los que me ha regalado esta facultad, por hacer de esta etapa un recuerdo inolvidable.

Además, quiero agradecer a Ester, mi profesora de bachiller, por enseñarme a querer las matemáticas, pues si no fuera por ella igual nunca hubiera seguido este camino.

Claudia Tarajano Afonso
La Laguna, 17 de mayo de 2023

Resumen · Abstract

Resumen

En la actualidad, la ciberseguridad se basa en lo complejo que es con un ordenador clásico, de forma eficiente, resolver ciertos problemas matemáticos. Sin embargo, con la aparición de los ordenadores cuánticos todo puede cambiar, pues estos, basándose en propiedades como el principio de superposición cuántica, tendrán una mayor capacidad de cálculo y gestión de la información. Por ello, la mayoría de los esquemas usados en la criptografía actual no serán seguros en la era post-cuántica.

A lo largo de este trabajo se introduce la Criptografía Post-Cuántica, y en particular el esquema de cifrado CRYSTALS-Kyber, que es un criptosistema de clave pública basado en retículos. Para ello, inicialmente se introduce el contexto en el que surge este esquema, pues actualmente se está intentando escoger el estándar de criptosistema resistente a los ordenadores cuánticos. Concretamente, el NIST (National Institute of Standards and Technology) inició hace unos años, en 2016, un proceso de estandarización de criptografía postcuántica. Inicialmente en la primera ronda del proceso fueron aceptadas 69 propuestas. En la actualidad, tras cuatro rondas, han quedado 4 finalistas de los cuales 3 son firmas digitales y 1 es un criptosistema de clave pública, que es CRYSTALS-Kyber.

Este esquema de cifrado, como otros participantes del concurso, está basado en retículos. Por ello, esta memoria comienza con los conceptos matemáticos fundamentales para comprender este concepto y así poder entender el esquema. Posteriormente, se describen los problemas en los que se basa el esquema: el problema del vector más corto (SVP, Shortest Vector Problem) y el Module-LWE (Learning With Error). A continuación, se detallan los algoritmos de generación de clave, cifrado y descifrado que conforman el criptosistema. Finalmente, se presenta CRYPTRIS, que es un videojuego similar al TETRIS, desarrollado en el INRIA (Institut National de Recherche en Informatique) bajo licencia abierta para introducir la criptografía basada en retículos.

Palabras clave: *Criptosistema – Clave pública – Anillo de polinomios – Retículo – SVP – Module-LWE – Algoritmo.*

Abstract

Nowadays, cybersecurity is based in how complex it is to solve certain mathematical problems efficiently using a standard computer. However, everything could change with the emergence of quantum computers, since these, based on properties such as quantum superposition, would have a greater capacity to perform calculations and manage information. Therefore, the schemes used in our current cryptography would not be secure in the post-quantum era.

This work introduces Post-Quantum Cryptography and, in particular, the CRYSTALS-Kyber encryption scheme, which is a public key cryptographic system based on lattices. In order to do so, the context in which this scheme arises is introduced, because currently the standard cryptosystem resistant to quantum computers is being chosen. Specifically, the NIST (National Institute of Standards and Technology) began a few years ago, in 2016, a process of standardization of post-quantum cryptography. Currently, after four rounds, there are 4 finalists, 3 of which are digital signatures while only 1 is a public-key cryptosystem, which is CRYSTALS-Kyber.

This encryption scheme is based on lattices, just as other participants of the contest. Therefore, this report begins with the underlying mathematical concepts in order to comprehend this concept and then be able to understand the scheme. Afterwards, this report describes the problems in which this scheme is based on: the Shortest Vector Problem (SVP) and the Module-LWE (Learning With Error). After that, we will elaborate on the key generating algorithms, the encryption algorithms and the decryption algorithms that shape the cryptographic system. Lastly, This report presents CRYPTRIS, which is a computer game similar to TETRIS, developed at INRIA (Institut National de Recherche en Informatique) under an open license to introduce the lattice-based cryptography.

Keywords: *Cryptographic system – Public key – Polynomial ring – Lattice – SVP – Module-LWE – Algorithm*

Contenido

Agradecimientos ... III

Resumen/Abstract ... VI

1. Fundamentos matemáticos ... 1
- 1.1. Anillo de polinomios ... 1
- 1.2. Anillo cociente ... 3
- 1.3. Espacios vectoriales ... 5
- 1.4. Retículos ... 7
  - 1.4.1. Caracterización geométrica ... 9
  - 1.4.2. Determinante de un retículo ... 10
  - 1.4.3. Retículo dual y q-ario ... 10

2. Descripción del esquema de cifrado CRYSTALS-Kyber ... 11
- 2.1. Problemas en los que se basa CRYSTALS-Kyber ... 11
  - 2.1.1. Problema del vector más corto (SVP) ... 12
  - 2.1.2. Problema de aprendizaje con errores (LWE) ... 12
- 2.2. Descripción ... 13
  - 2.2.1. Funciones ... 14
  - 2.2.2. Algoritmos ... 16

3. CRYPTRIS ... 21
- 3.1. Criptografía de clave pública en CRYPTRIS ... 21
- 3.2. Criptografía basada en retículos en CRYPTRIS ... 26

Bibliografía ... 33

Lista de símbolos y abreviaciones ... 35

Poster ... 37

1

Fundamentos matemáticos

A lo largo de este capítulo se definen los conceptos matemáticos necesarios para poder comprender el esquema de cifrado CRYSTALS-Kyber. Gracias a estos conceptos es posible definir el problema en el que está basado el esquema de cifrado.

1.1. Anillo de polinomios

Las siguientes definiciones se utilizan posteriormente en el esquema de cifrado.

Definición 1.1. *Sea A un conjunto no vacío dotado de dos leyes de composición interna que denotaremos $+$ y $\cdot$ se dice que A es un anillo si verifica:*

1. *$(A, +)$ es asociativa, conmutativa, existe elemento neutro y todo elemento admite simétrico.*
2. *$(A, \cdot)$ es asociativa.*
3. *$(A, \cdot)$ es distributiva respecto de $+$.*

Ejemplo 1.2. Claros ejemplos de anillos son: $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ y $\mathbb{C}$. Además también es un anillo el conjunto $M_n(\mathbb{R})$ formado por las matrices cuadradas de orden n y coeficientes en $\mathbb{R}$ con la suma y la multiplicación usual de matrices .

Definición 1.3. *Se dice que un anillo $(A, +, \cdot)$ es conmutativo o abeliano si $(A, \cdot)$ verifica la propiedad conmutativa. Por otro lado, si $(A, \cdot)$ admite elemento neutro, denotado 1_A, se dice que es unitario.*

Ejemplo 1.4. El anillo $M_n(\mathbb{R})$ no es conmutativo.

Si A es un anillo, un elemento $a \in A \setminus \{0_A\}$ es una unidad de A si admite inverso, esto es, $\exists\, b \in A \setminus \{0_A\}$ tal que $a \cdot b = b \cdot a = 1_A$. Si A es anillo unitario, se denota A^* al conjunto de las unidades de A.

Definición 1.5. *Sea A un anillo conmutativo y unitario, se dice que A es un cuerpo si $A^* = A \setminus \{0_A\}$.*

Proposición 1.6. *Sea A un anillo conmutativo, se tiene el conjunto de polinomios sobre A:*

$$A[x] = \left\{ \sum_{i=0}^{n} a_i x^i : n \in \mathbb{N} \cup \{0\}, a_i \in A, \forall i \in \{1, \dots, n\} \right\}$$

en el que se pueden definir las operaciones $+$ y $\cdot$ tales que si $f(x) = \sum_{i=0}^{n} a_i x^i, g(x) = \sum_{j=0}^{m} b_j x^j \in A[x]$ entonces:

1. *Suponiendo sin pérdida de generalidad que $n \geq m$:*
   $(b_j = 0, \forall j \in \{m+1, \dots, n\})$

$$f(x) + g(x) = \sum_{i=0}^{n} a_i x^i + \sum_{j=0}^{m} b_j x^j = \sum_{k=0}^{n} (a_k + b_k) x^k$$

2. $$f(x)g(x) \left(\sum_{i=0}^{n} a_i x^i \right) \left(\sum_{j=0}^{m} b_j x^j \right) = \sum_{k=0}^{m+n} c_k x^k, \text{ siendo } c_k = \sum_{i+j=k} a_i b_j$$

Dado el conjunto de polinomios $A[x]$ y las operaciones anteriores, se puede definir un anillo conmutativo $(A[x], +, \cdot)$ al que se llama anillo de polinomios con coeficientes en A e indeterminada x.

Definición 1.7. *Sea $a \in A$ y $p(x) \in A[x]$, se dice que a es una raíz de $p(x)$ si $p(a) = 0$.*

Ejemplo 1.8. En $\mathbb{C}$ el polinomio $f(x) = x^2 + 1$ tiene por raíces $\pm i$. En cambio en $\mathbb{R}$ este polinomio no tendría raíces, pues no existen números reales $a \in \mathbb{R}$ tales que $f(a) = 0$.

Ahora se define un tipo concreto de polinomios, llamados polinomios ciclotómicos, pues serán de gran utilidad en el esquema de cifrado CRYSTALS-Kyber se trabaja con el anillo $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$, con q primo y $n = 2^{n'-1}$ tal que $x^n + 1$ es el $2^{n'}$-ésimo polinomio ciclotómico.

Definición 1.9. *Para cada n número natural, se puede definir el n-ésimo polinomio ciclotómico $\Phi_n(x)$ de forma recursiva partiendo de $\Phi_1(x) = x - 1$, como:*

$$\Phi_n(x) = \frac{x^n - 1}{\prod_{\substack{d|n \\ d<n}} \Phi_d(x)}$$

Estos polinomios son aquellos cuyas raíces son todas las raíces primitivas de orden n de la unidad:

$$U_n = \{w \in \mathbb{C}/w \text{ es raíz de } x^n - 1\} = \left\{e^{\frac{2\pi i}{n}k}/k \in \{0, \ldots, n-1\}\right\}$$

Ejemplo 1.10. Los primeros conjuntos de raíces primitivas son los siguientes:

1. $U_1 = \{w \in \mathbb{C}/w \text{ es raíz de } x - 1\} = \{1\}$
2. $U_2 = \{w \in \mathbb{C}/w \text{ es raíz de } x^2 - 1\} = \{\pm 1\}$
3. $U_3 = \{w \in \mathbb{C}/w \text{ es raíz de } x^3 - 1\} = \left\{1, e^{\frac{2\pi i}{3}}, e^{\frac{2\pi i}{3}\cdot 2}\right\} = \{1, \tau_3, \tau_3^2\}$
4. $U_4 = \{w \in \mathbb{C}/w \text{ es raíz de } x^4 - 1\} = \{1, i, -1, -i\} = \{1, \tau_4, \tau_4^2, \tau_4^3\}$

De este modo podemos obtener las raíces de $x^n - 1$ para cualquier n, y con esto, es fácil deducir el polinomio ciclotómico correspondiente.

1.2. Anillo cociente

Se trabajará con el anillo $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$, luego, a continuación se introduce el concepto de anillo cociente. Para ello, en primer lugar se ve qué es el ideal de un anillo A.

Definición 1.11. *Sean $(A, +, \cdot)$ un anillo conmutativo y unitario e $I \subseteq A$, con $I \neq \emptyset$. Se dice que I es un ideal de A si verifica:*

1. $0_A \in I$
2. $\forall a, b \in I, a - b \in I$
3. $\forall a \in A, \forall b \in I, a \cdot b \in I$

Ejemplo 1.12. El conjunto formado por los múltiplos de n es un ideal de $\mathbb{Z}$, luego $\mathbb{Z}$ tiene infinitos ideales.

Con la definición anterior, es fácil ver que se cumple la siguiente proposición:

Proposición 1.13. *Sean $(A, +, \cdot)$ un anillo conmutativo y unitario, y elementos $a_1, \ldots, a_n \in A$. Entonces el conjunto $I = \{r_1 \cdot a_1 + \ldots + r_n \cdot a_n : r_i \in A\}$ es un ideal de A. A este ideal se le denomina ideal generado por $a_1, \ldots, a_n$ y se denota por $(a_1, \ldots, a_n)$.*

Sean A un anillo conmutativo y unitario e I un ideal de A. En A definimos la siguiente relación binaria:

$$\forall a, b \in A : a \sim_I b \Leftrightarrow a - b \in I$$

Se puede probar que, efectivamente, la relación $\sim_I$ es una relación de equivalencia, pues cumple la propiedad reflexiva, simétrica y transitiva. Como consecuencia, se puede obtener la clase de equivalencia, clase de a módulo I, de todo $a \in A$:

$$\{b \in A : b \sim_I a\} = \{b \in A : \exists z \in I, b - a = z\} = \{b = a + z : z \in I\} \underset{\text{not.}}{\equiv} a + I$$

se denota por A/I al conjunto cociente asociado a la relación $\sim_I$, esto es, $A/I = \{a + I : a \in A\}$ que es una partición de A.

Ejemplo 1.14. Tomando el conjunto $A = \mathbb{Z}$ e $I = (2)$, se tiene que $\mathbb{Z}/(2) = \{a + (2) : a \in \mathbb{Z}\}$, luego se observa que si se toma la clase de equivalencia de 0, se tiene que $0 + I = \{0 + z : z \in (2)\} = (2)$, en cambio, si se toma la clase de equivalencia de 1 , se tiene $1 + I = \{1 + z : z \in (2)\}$ por tanto, $0 + I$ está formado por los números enteros pares, y $1 + I$ por los impares, luego se tiene una partición de $\mathbb{Z}$ y como consecuencia se concluye que $\mathbb{Z}/(2) = \{0 + I, 1 + I\}$.

En general, si se toma el conjunto $A = \mathbb{Z}$ e $I = (n)$ con $n \in \mathbb{N}, n > 1$ entonces en A/I hay n clases de equivalencia que serán:

$$A/I = \{0 + I, 1 + I, \ldots, (n - 1) + I\}$$

y en tal caso, se denota al cociente A/I como $\mathbb{Z}_n$ y a las clases de equivalencia se denotan como $[a]_n$ en lugar de $a + I$. Además a la relación $a \sim_I b$ se le denota como $a \equiv b \mod n$ y se lee: a convergente con b módulo n.

Proposición 1.15. *Sean A un anillo conmutativo y unitario e I un ideal de A, entonces A/I es un anillo conmutativo y unitario con las siguientes operaciones:*

$$(a + I) + (b + I) = (a + b) + I$$

$$(a + I) \cdot (b + I) = (a \cdot b) + I$$

Al anillo A/I se le llama anillo cociente de A sobre I.

El anillo con el que se trabaja en el esquema es: $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$. Por tanto, conociendo las definiciones anteriores, se tiene que:

1. $\mathbb{Z}_q$ es el anillo cociente de $\mathbb{Z}$ sobre el ideal (q) que es un anillo conmutativo y unitario por la proposición 1.15.
2. Por la proposición 1.6, se puede tomar el anillo de polinomios $\mathbb{Z}_q[x]$.
3. Tiene sentido hablar del anillo $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$, pues se puede definir el anillo cociente de $\mathbb{Z}_q[x]$ sobre el ideal $(x^n + 1)$.

A la hora de trabajar con este anillo se trabaja de la siguiente manera, con vectores columna de la forma:

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_i \\ \vdots \\ v_n \end{bmatrix} \in \mathcal{R}_q^n$$

con el elemento i-ésimo $v_i \in \mathcal{R}_q, \forall i \in \{1, \ldots, n\}$.

Del mismo modo, se trabaja con matrices como la siguiente:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \underset{\text{not.}}{\equiv} (a_{ij}) \in \mathcal{R}_q^{n \times m}$$

con el elemento de la i-ésima fila y j-ésima columna, $a_{ij} \in \mathcal{R}_q, \forall i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}$.

1.3. Espacios vectoriales

Definición 1.16. *Un espacio vectorial V sobre un cuerpo K es un conjunto no vacío dotado de operaciones:*

1. *Una operación interna $+ : V \times V \longrightarrow V$ que cumple:*
   - *Propiedad conmutativa: $u + v = v + u,\ \forall u, v \in V$*
   - *Propiedad asociativa: $u + (v + w) = (u + v) + w,\ \forall u, v, w \in V$*
   - *Existencia de elemento neutro: $\exists 0_V \in V : u + 0_V = 0_V + u = u,\ \forall u \in V$*
   - *Existencia de elemento opuesto: $\forall u \in V, \exists -u \in V : u + (-u) = (-u) + u = e$*
2. *Una operación externa $\cdot : K \times V \longrightarrow V$ que cumple:*
   - *Propiedad asociativa: $a \cdot (b \cdot u) = (a \cdot b) \cdot u,\ \forall a, b \in K, \forall u \in V$*
   - *Existencia de elemento neutro: $\exists e \in K : e \cdot u = u \cdot e = u, \forall u \in V$*
   - *Propiedad distributiva respecto a la suma vectorial: $a \cdot (u + v) = a \cdot u + a \cdot v, \forall a \in K, \forall u, v \in V$*
   - *Propiedad distributiva respecto a la suma escalar: $(a + b) \cdot u = a \cdot u + b \cdot u, \forall a, b \in K, \forall u \in V$*

Definición 1.17. *Se llama subespacio generado al conjunto de todas las combinaciones lineales de un conjunto de vectores $v_1, \ldots, v_n$ y se denota $span(\{v_1, \ldots, v_n\})$*

Definición 1.18. *Sea V un espacio vectorial sobre un cuerpo $\mathbb{K} = \mathbb{R}$ o $\mathbb{C}$. Una aplicación $||\cdot||: V \longrightarrow \mathbb{K}$, se dice que es una norma si cumple:*

1. $|| v || \geq 0, \forall v \in V$
2. $|| v ||= 0 \Leftrightarrow v = 0 \in V$
3. $|| k \cdot v ||=| k | \cdot || v ||, \forall v \in V, \forall k \in \mathbb{K}$
4. *Desigualdad triangular:* $|| v + u ||\leq || v || + || v ||, \forall u, v \in V$

Al par $(V, || \, . \, ||)$ *se le denomina espacio vectorial normado.*

Ejemplo 1.19. Ejemplos de espacios vectoriales normados podrían ser los siguientes:

1. $(\mathbb{R}, | \, . \, |)$, siendo $| \, . \, |$ el valor absoluto
2. $(\mathbb{R}^2, || \, . \, ||_2)$, donde $|| x ||_2= \sqrt{\sum_{k=1}^{n} x_i^2}$
3. $(\mathbb{R}^p, || \, . \, ||_p)$, con $p \geq 1$ donde $|| x ||_p= (\sum_{k=1}^{n} x_i^p)^{1/p}$

Definición 1.20. *Sea* V *un espacio vectorial sobre un cuerpo* $\mathbb{K} = \mathbb{R}$ *o* $\mathbb{C}$*. Una transformación* $\langle \cdot, \cdot \rangle : V \times V \longrightarrow \mathbb{K}$ *se dice que es un producto interno o producto escalar si* $\forall u, v, w \in V, \forall a \in \mathbb{K}$ *se cumple:*

1. *Linealidad:* $\langle u + v, w \rangle = \langle u, w \rangle + \langle v, w \rangle$
2. *Homogeneidad:* $\langle au, v \rangle = a\langle u, v \rangle$
3. *Simetría* $\langle u, v \rangle = \langle v, u \rangle$*. En el caso de que el cuerpo de escalares sea* $\mathbb{C}$ *se considera simetría hermitiana:* $\langle u, v \rangle = \overline{\langle v, u \rangle}$
4. $\langle u, u \rangle \geq 0$
5. $\langle u, u \rangle = 0 \Leftrightarrow u = 0 \in V$

Uno de los productos más comunes es el denominado producto escalar usual en el espacio euclídeo $\mathbb{R}^n$ que dados dos vectores $u = [u_1, \ldots, u_n], v = [v_1, \ldots, v_n]$ se define como:

$$\langle u, v \rangle = \sum_{i=1}^{n} u_i v_i = u^T v$$

Proposición 1.21. *Todo producto escalar induce una norma tomando:*

$$|| v ||= \sqrt{\langle v, v \rangle}$$

Definición 1.22. *Dos elementos* $u, v \in V$ *se dice que son ortogonales, y se denota* $u \perp v$*, si* $\langle u, v \rangle = 0$

Definición 1.23. *Sean* $v_1, \ldots, v_n$ *vectores, se dice que son linealmente independientes si la única forma de obtener*

$$\alpha_1 v_1 + \ldots + \alpha_n v_n = 0$$

es con $\alpha_1 = \ldots = \alpha_n = 0$

Definición 1.24. *Sean $v_1, \ldots, v_n$ vectores linealmente independientes en $\mathbb{R}^n$, entonces, se definen los vectores ortogonales de Gram-Schmidt $\tilde{v}_1, \ldots, \tilde{v}_n$ como*

$$\begin{cases} \tilde{v}_1 = v_1 \\ \tilde{v}_j = v_j - \sum_{i<j} \mu_{i,j} \tilde{v}_i \end{cases}$$

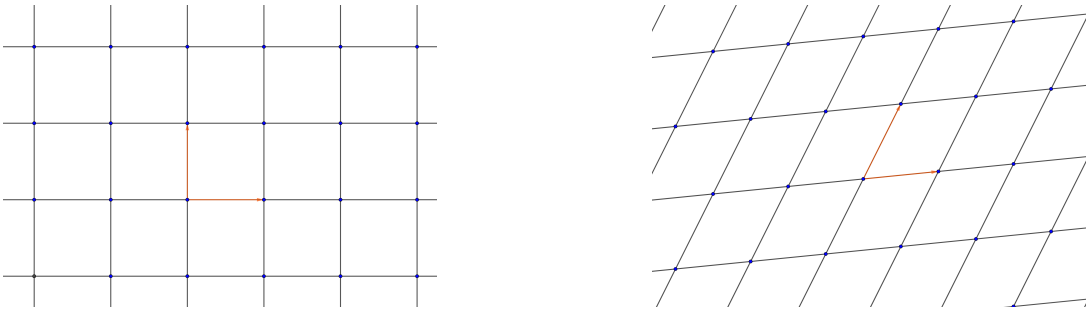
siendo $\mu_{i,j} = \frac{\langle v_j, \tilde{v}_i \rangle}{||v_i||^2}$, $\forall j \in \{2, \ldots, n\}$. A este proceso se le llama proceso de ortogonalización de Gram-Schmidt.

1.4. Retículos

Definición 1.25. *Sean $\mathbb{K}$ un cuerpo y $\{b_1, \ldots, b_n\} \subseteq \mathbb{K}^m$ un conjunto de vectores linealmente independientes $(m \geq n)$. Se define un retículo (lattice) $\mathcal{L}$ generado por $\{b_1, \ldots, b_n\}$ como el conjunto:*

$$\mathcal{L} \equiv \mathcal{L}(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z} \right\}$$

de todas las combinaciones lineales de $\{b_1, \ldots, b_n\}$ con coeficientes enteros.



(a) Retículo $\mathbb{Z}^2$ (b) Retículo con base no ortogonal

Figura 1.1: Retículos 2-dimensionales

Ejemplo 1.26. El conjunto de los números pares es un retículo definido a partir del conjunto formado por el elemento $\{2\}$. Un retículo trivial es el formado idénticamente por el vector nulo para cualquier dimensión del mismo.

Los vectores $b_1, \ldots, b_n$ forman una base del retículo $\mathcal{L}$ que representaremos de forma matricial como

$$B = [b_1, \ldots, b_n] \in \mathbb{K}^{m \times n}$$

lo que nos daría una representación equivalente de retículo:

$$\mathcal{L} \equiv \mathcal{L}(B) = \{Bx : x \in \mathbb{Z}^n\}$$

Se dice que n es el rango del retículo y m su dimensión. Cuando $n = m$, diremos que el retículo tiene rango máximo.

Proposición 1.27. *Dada $B \in \mathbb{K}^{m\times n}$ con columnas linealmente independientes. Luego, el retículo será de rango máximo, si y solo si,*

$$span(B) = \{Bx : x \in \mathbb{K}^n\} = \mathbb{K}^m$$

Veamos ahora cual es la relación entre dos bases del retículo:

Supongamos que $B = \{b_1, b_2, \ldots, b_n\}$ es una base del retículo $\mathcal{L}(B)$ y sea $B' = \{w_1, \ldots, w_n\}$ un conjunto de vectores en $\mathcal{L}$, por lo tanto, podremos escribir:

$$\begin{aligned} w_1 &= a_{11}b_1 + a_{12}b_2 + \cdots + a_{1n}b_n \\ w_2 &= a_{21}b_1 + a_{22}b_2 + \cdots + a_{2n}b_n \\ &\vdots \\ w_n &= a_{n1}b_1 + a_{n2}b_2 + \cdots + a_{nn}b_n \end{aligned} \tag{1.1}$$

con $a_{ij} \in \mathbb{Z}$ para $i, j \in \{1, \ldots, n\}$. Luego podemos definir la matriz A como:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

y reescribir el sistema anterior 1.1 como:

$$\begin{pmatrix} w_1 & w_2 & \cdots & w_n \end{pmatrix} = \begin{pmatrix} b_1 & b_2 & \cdots & b_n \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

Entonces, si queremos expresar los v_i en términos de los w_j debemos poder invertir la matriz A, ya que en ese caso podríamos escribir:

$$\begin{pmatrix} w_1 & w_2 & \cdots & w_n \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}^{-1} = \begin{pmatrix} b_1 & b_2 & \cdots & b_n \end{pmatrix}$$

Ahora bien, como $w_i \in \mathcal{L}$, entonces los coeficientes de la matriz A son enteros, luego, podemos garantizar la existencia de A^{-1} y además, asegurar que también será entera. Luego,

$$1 = \det(I) = \det(A)\det(A^{-1})$$

Proposición 1.28. *Dada una matriz entera B, su inversa B^{-1}, es también una matriz enteras si, y solo si,* $\det(B) = \pm 1$.

Luego, con esto, podemos asegurar que $\det(A) = \pm 1$. Además, diremos que una matriz $U \in \mathbb{Z}^{n\times n}$ es unimodular si $\det(U) = \pm 1$.

Por tanto, con esto, podemos concluir la siguiente proposición:

Proposición 1.29. *Dadas dos bases $B, B' \in \mathbb{K}^{m\times n}$ para un retículo $\mathcal{L} \subset \mathbb{K}^m$, existe una matriz unimodular $U \in \mathbb{Z}^{n\times n}$ tal que $B' = BU$.*

Es fácil ver que si el retículo es de rango máximo (y por tanto invertible), entonces de este resultado podemos deducir que basta verificar que $B^{-1}B' = U$ sea unimodular.

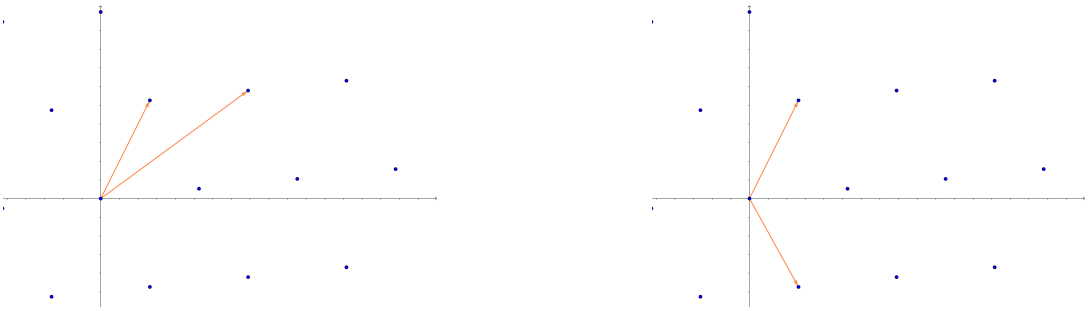


Figura 1.2: Mismo retículo generado por dos bases distintas

1.4.1. Caracterización geométrica

Definición 1.30. *El paralelepípedo fundamental de n vectores $b_1, \ldots, b_n \in \mathbb{R}^m$ linealmente independientes viene dado por:*

$$\mathcal{P}(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i : x_i \in [0, 1) \right\}$$

Observamos que los vectores que conforman la base de un retículo van a tener asociado un paralelepípedo fundamental y que además dos bases distintas generan paralelepípedos fundamentales distintos.

Proposición 1.31. *Sea $\mathcal{P}(B)$ un paralelepípedo fundamental asociado a un retículo $\mathcal{L}(B) \subseteq \mathbb{R}^n$ de rango n. Entonces, para todo $w \in \mathbb{R}^n$, existe un único $t \in \mathcal{P}(B)$ y un único $v \in \mathcal{L}$ tales que $w = t + v$. En otras palabras, si*

consideramos traslaciones de $\mathcal{P}(B)$ sobre los vectores del retículo cubrimos el espacio $\mathbb{R}^n$ sin solapamientos, esto es,

$$\bigcup_{v \in \mathcal{L}} \mathcal{P}(B) + v = \mathbb{R}^n$$

Teorema 1.32. *Sea $\mathcal{L} \subset \mathbb{R}^n$ un retículo de rango máximo, n dimensional, y sea $B = \{b_1, \ldots, b_n\} \subset \mathcal{L}$ un conjunto de vectores linealmente independientes. Entonces B es una base de L si, y solo si, $\mathcal{P}(b_1, \ldots, b_n) \cap \mathcal{L} = \{0\}$.*

Este concepto nos va a ayudar a poder definir el determinante de un retículo.

1.4.2. Determinante de un retículo

Definición 1.33. *Dado un retículo $\mathcal{L} \subset \mathbb{R}^m$ definimos su determinante, que denotaremos por* $\det(\mathcal{L})$*, por el volumen n dimensional de su paralelepípedo fundamental $\mathcal{P}(B)$.*

Para el caso en el que el retículo es de rango máximo, se da que $\det(\mathcal{L}) = |\det(B)|$, pues B es cuadrada.

Para que esta definición tenga sentido se debe cumplir para cualquier base del retículo. Luego, veamos que los paralelepípedos asociados a distintas bases tienen el mismo volumen. Por la proposición 1.29 sabemos que para dos bases B y B' de $\mathcal{L}$ se da que $B' = BU$, con U unimodular. Luego,

$$|\det(B')| = |\det(BU)| = |\det(B)| \cdot |\det(U)| = |\det(B)|$$

por ser U unimodular.

1.4.3. Retículo dual y q-ario

Definición 1.34. *Sea $\mathcal{L}$ un retículo en $\mathbb{K}^n$ se llama dual de $\mathcal{L}$ y se denota como $\mathcal{L}^*$ al conjunto formado por los vectores $y \in \mathbb{F}^n$ tales que $\langle z, y \rangle \in \mathbb{Z}, \forall z \in \mathcal{L}$.*

Proposición 1.35. *Para todo retículo $\mathcal{L}$ se tiene que:*

- $(\mathcal{L}^*)^* = \mathcal{L}$
- $\det(\mathcal{L}^*) = \frac{1}{\det(\mathcal{L})}$

Definición 1.36. *Sean $\mathcal{L}$ un retículo en $\mathbb{R}^n$ y q un número entero primo. Si tomamos una matriz $A \in \mathbb{Z}_q^{n \times m}$, con $n \leq m$, se llama retículo q-ario a*

$$\Lambda_q^{\perp}(A) = \{y \in \mathbb{Z}^m : Ay = 0 \ mod \ q\}$$

tal que $\mathbb{Z}_q^m \subseteq \Lambda_q^{\perp}(A) \subseteq \mathbb{Z}^m$

Se tiene que

$$\left(\Lambda_q^{\perp}(A)\right)^* = \frac{1}{q}\Lambda_q(A)$$

2

Descripción del esquema de cifrado CRYSTALS-Kyber

En este capítulo, en primer lugar se introducen los problemas computacionales sobre los que se basa el esquema de cifrado CRYSTALS-Kyber, y a continuación se describen los algoritmos necesarios para su comprensión y desarrollo.

2.1. Problemas en los que se basa CRYSTALS-Kyber

CRYSTALS-Kyber es un mecanismo de encapsulación de clave (KEM, *Key-Encapsulation Mechanism*) con la propiedad IND-CCA2, cuya seguridad se basa en la dificultad de resolver los problemas del vector más corto (SVP, *Shortest Vector Problem*) y Module-LWE.

Un KEM es un mecanismo que permite que cualquier participante en posesión de la clave pública de otro participante pueda transmitir de forma segura una clave secreta a esa otra parte. Por otra parte, se dice que un criptosistema posee la propiedad de indistinguibilidad (IND) si un atacante no puede distinguir pares de textos cifrados en función del texto original al que corresponden. Existen varias versiones de indistinguibilidad en función de las capacidades del atacante.

En particular, en la indistinguibilidad bajo ataque de texto cifrado elegido adaptativo (IND-CCA2, *INDistinguishability under adaptive Chosen Ciphertext Attack*), además de a la clave pública, el adversario tiene acceso a un oráculo de descifrado que le permite descifrar textos cifrados arbitrarios que hayan sido generados utilizando la clave pública de cifrado, recuperando los correspondientes textos originales. Además, el adversario puede continuar consultando el oráculo de descifrado incluso tras haber recibido el texto cifrado en cuestión, con la advertencia de que no puede pasar justo ese texto cifrado al oráculo para conseguir su descifrado. Actualmente esta definición de seguridad es la más sólida para un sistema criptográfico de clave pública.

2.1.1. Problema del vector más corto (SVP)

El problema del vector más corto se basa en encontrar el vector no nulo más corto de $\mathcal{L}(B)$, siendo $\mathcal{L}$ un retículo y B la matriz base de $\mathcal{L}$. Se puede observar una idea visual sobre el problema en la siguiente imagen:

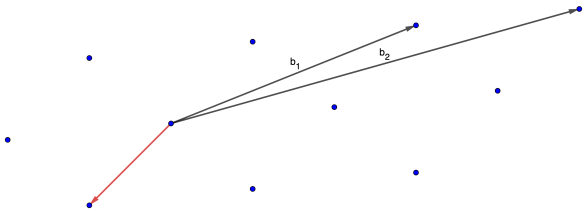


Figura 2.1: Idea del SVP

(a) Los vectores de la base son los negros y el vector rojo es el más corto

Para ello, se define $\lambda_1(\mathcal{L})$ como la distancia mínima de un retículo $\mathcal{L}$, siendo esta la longitud más corta de un vector del retículo, esto es,

$$\lambda_1(\mathcal{L}) = \min_{b \in \mathcal{L}(B)} || \, b \, ||$$

Veamos un resultado que acota esta distancia:

Teorema 2.1 (Teorema de Minkowski). *Sea $\mathcal{L} \subseteq \mathbb{R}^n$ un retículo de rango máximo, entonces:*

$$\lambda_1(\mathcal{L}) \leq \sqrt{n} \cdot \det(\mathcal{L})^{1/n}$$

2.1.2. Problema de aprendizaje con errores (LWE)

Sean $m, n, q \in \mathbb{Z}$, se consideran $a_i \in \mathbb{Z}_q^n$ uniformes y $b_i \in \mathbb{Z}_q$ para todo $i \in \{1, \ldots, m\}$. Además se considera un vector $s \in \mathbb{Z}_q^n$ secreto. Con esto, se crea un sistema de ecuaciones con error, formado por m ecuaciones cuya incógnita es s,

$$\begin{aligned} \langle s, a_1 \rangle &\approx_\chi b_1 \quad \text{mód } q \\ &\vdots \\ \langle s, a_i \rangle &\approx_\chi b_i \quad \text{mód } q \\ &\vdots \\ \langle s, a_m \rangle &\approx_\chi b_m \quad \text{mód } q \end{aligned}$$

donde $\chi : \mathbb{Z}_q \to \mathbb{R}^+$ es una distribución de probabilidad sobre $\mathbb{Z}_q$ que determina el error en las ecuaciones. De forma compacta se puede escribir que $As + e = b$ mód q, siendo $e \in \mathbb{Z}_q^m$ el error. El problema consiste en ser capaz de recuperar el vector s de las ecuaciones anteriores que podemos escribir como: (siendo e_i las componentes de e)

$$\begin{aligned} \langle s, a_1 \rangle + e_1 &= b_1 \\ &\vdots \\ \langle s, a_i \rangle + e_i &= b_i \\ &\vdots \\ \langle s, a_m \rangle + e_m &= b_m \end{aligned}$$

El Module-LWE utiliza el problema Ring-LWE, por tanto, veamos en que consiste dicho problema. La diferencia es que en lugar de trabajar con $\mathbb{Z}_q$, ahora se utiliza un anillo, normalmente $\mathbb{Z}_q[X]/(X^n + 1)$, por tanto todas las variables que se utilizan son polinomios de algún anillo. Por otro lado, en CRYSTALS-Kyber, las variables están definidas sobre el mismo anillo polinómico $\mathcal{R}_q$ de tamaño constante.

Recordamos que el anillo con el que se trabaja es $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$, siendo q un número primo y con n siendo $2^{n'-1}$ tal que $x^n + 1$ es el $2^{n'}$-ésimo polinomio ciclotómico.

2.2. Descripción

Para poder comprender los algoritmos es importante fijar la notación que se emplea, teniendo en cuenta que las entradas y salidas de todos los algoritmos de Kyber son arrays de bytes:

- Denotamos por, $\mathcal{B}$ al conjunto $\{0, \ldots, 255\}$, $\mathcal{B}^k$ al conjunto de arrays de bytes de longitud k y $\mathcal{B}^*$ el conjunto de arrays de bytes de longitud arbitraria.
- $(a \mid\mid b)$: Denotamos por $(a \mid\mid b)$ a la concatenación de a y b, siendo a y b arrays de bytes.
- $a + k$: Supongamos un array de bytes a, denotamos por $a + k$ al nuevo array de bytes que comienza a partir del byte k de a y continua hasta completar a.
- `BytesToBits`: Habrá momentos en los que será preferible trabajar con un array de bits en lugar de un array de bytes. Luego vamos a definir una función

que haga esta conversión, esta función es `BytesToBits`. Toma como entrada un array de l bytes y devuelve un array de $8l$ bits (ya que un byte son 8 bits).

Nota 2.2. Con la función anterior, si se tiene un bit β_i en la posición i del array de bits que se obtiene como salida de la función, para obtener de que posición del array de entrada se obtiene, se toma del byte $b_{i/8}$ la posición $i/8$. Luego, se obtiene la siguiente relación entre la posición de un bit en el array de salida y de un byte en el de entrada (bajo la función `BytesToBits`):

$$\beta_i = \left(\left(b_{i/8}/2^{(i \mod 8)}\right) \mod 2\right)$$

- $r' = r \mod^{\pm} \alpha$: Para un entero positivo par α, respectivamente impar, se define $r' = r \mod^{\pm} \alpha$ como el único elemento r' tal que $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$, respectivamente $-\frac{\alpha-1}{2} < r' \leq \frac{\alpha-1}{2}$, tal que $r' = r \mod \alpha$. Para un entero positivo α, se define $r' = r \mod^{+} \alpha$ como la definición usual de módulo, esto es, el único elemento r' en el rango $0 \leq r' < \alpha$ tal que $r' = r \mod \alpha$.
- $\lceil x \rfloor$: Para un elemento $x \in \mathbb{Q}$ se denota por $\lceil x \rfloor$ al redondeo de x al número entero más próximo.
- $|| w ||_\infty$: Sea $w \in \mathbb{Z}_q$, y se denota por $|| w ||_\infty$ para referirnos a $| w \mod^{\pm} q |$. Con esto, se definen las normas l_∞ y l_2 para un cierto elemento $w = w_0 + w_1 X + \ldots + w_{n-1} X^{n-1} \in \mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$:

$$|| w ||_\infty = \max_i || w_i ||_\infty, \quad || w || = \sqrt{|| w_0 ||_\infty^2 + \ldots + || w_{n-1} ||_\infty^2}$$

- $s \leftarrow S$: Se denota por $s \leftarrow S$ para hacer referencia a que la s se elige uniformemente al azar del conjunto S (que puede ser una distribución de probabilidad).

A continuación, se fijan los valores $n = 256$, $n' = 9$ y un número primo $q = 3329$ para lo que queda de memoria.

2.2.1. Funciones

En primer lugar se definen dos funciones, $\texttt{Compress}_q$ y $\texttt{Decompress}_q$ de tal forma que:

$$x' = \texttt{Decompress}_q\left(\texttt{Compress}_q(x, d), d\right) \tag{2.1}$$

es un elemento cercano a x, específicamente:

$$|x' - x \mod^{\pm} q| \leq B_q := \left\lceil \frac{q}{2^{d+1}} \right\rfloor$$

La función $\texttt{Compress}_q(x, d)$ toma como entrada un elemento $x \in \mathbb{Z}_q$ y devuelve un entero en el rango $\{0, \ldots, 2^d - 1\}$, siendo $d < \lceil \log_2(q) \rceil$. Por otro lado, la función $\texttt{Decompress}_q$ es aquella que cumple la propiedad anterior 2.1. Luego, se define como:

$$\begin{aligned} \texttt{Compress}_q(x,d) &= \left\lceil \left(2^d/q\right) \cdot x \right\rfloor \mod {}^+2^d \\ \texttt{Decompress}_q(x,d) &= \left\lceil \left(q/2^d\right) \cdot x \right\rfloor \end{aligned} \tag{2.2}$$

Cuando estas funciones son usadas para un cierto $x \in \mathcal{R}_q$ o $x \in \mathcal{R}_q^k$ se aplica el procedimiento a cada coeficiente de forma individual.

Otras funciones que son necesarias para comprender los algoritmos son:

- $\texttt{PRF} : \mathcal{B}^{32} \times \mathcal{B} \longrightarrow \mathcal{B}^*$ una función pseudoaleatoria.
- $\texttt{XOF} : \mathcal{B}^* \times \mathcal{B} \times \mathcal{B} \longrightarrow \mathcal{B}^*$ una función extendable-output, esto es, una función hash que genera un gran número arbitrario de bits aleatorios (aparentemente).
- $\texttt{H} : \mathcal{B}^* \longrightarrow \mathcal{B}^{32}$ y $\texttt{G} : \mathcal{B}^* \longrightarrow \mathcal{B}^{32} \times \mathcal{B}^{32}$ dos funciones hash.
- $\texttt{KDF} : \mathcal{B}^* \longrightarrow \mathcal{B}^*$ una función key-derivation, es decir, un algoritmo que deriva claves secretas de un valor secreto mediante una función pseudoaleatoria.

A continuación se define una función que será de gran utilidad para hacer operaciones en nuestro anillo $\mathcal{R}_q$. Para ello, se emplea la transformada teórica de números (NTT) que trabaja con la primera $256-$raíz primitiva de la unidad modulo q que se denota ω y $\{\omega, \omega^3, \omega^5, \ldots, \omega^{255}\}$ el conjunto formado por todas las $256-$raices de la unidad. En nuestro caso particular, por los valores que se toman de n, n', q se tiene que $\omega = 17$. Luego, se define la aplicación biyectiva $\texttt{NTT} : \mathcal{R}_q \longrightarrow \mathcal{R}_q$ tal que, para un cierto $f = f_0 + f_1 X + \ldots + f_{255} X^{255} \in \mathcal{R}_q$ devuelve:

$$\texttt{NTT}(f) = \hat{f}_0 + \hat{f}_1 X + \ldots + \hat{f}_{255} X^{255} \equiv \hat{f}$$

siendo:

$$\begin{aligned} \hat{f}_{2i} &= \textstyle\sum_{j=0}^{127} f_{2j} \omega^{(2\texttt{br}_7(i)+1)j} \\ \hat{f}_{2i+1} &= \textstyle\sum_{j=0}^{127} f_{2j+1} \omega^{(2\texttt{br}_7(i)+1)j} \end{aligned} \tag{2.3}$$

con ω tal y como la definimos antes y donde $\texttt{br}_7(i)$ con $i \in \{1, \ldots, 127\}$ es la inversión de bits del entero i de 7 bits sin signo. Realmente, la representación algebraica de $\texttt{NTT}(f) = \hat{f}$ son los siguientes 128 polinomios de grado 1:

$$\hat{f} = \left(\hat{f}_0 + \hat{f}_1 X, \hat{f}_2 + \hat{f}_3 X, \ldots, \hat{f}_{254} + \hat{f}_{255} X\right)$$

Por otro lado, usando $\texttt{NTT}$ y su inversa $\texttt{NTT}^{-1}$ se puede computar $f \cdot g$ donde $f, g \in \mathcal{R}_q$ como $\texttt{NTT}^{-1}\left(\texttt{NTT}(f) \circ \texttt{NTT}(g)\right)$ donde $\texttt{NTT}(f) \circ \texttt{NTT}(g) = \hat{f} \circ \hat{g} = \hat{h}$ denota los 128 productos:

$$\hat{h}_{2i} + \hat{h}_{2i+1} X = (\hat{f}_{2i} + \hat{f}_{2i+1} X)(\hat{g}_{2i} + \hat{g}_{2i+1} X) \mod X^2 - \omega^{2\texttt{br}_7(i)+1}$$

En caso de aplicar $\circ$ a matrices (o vectores) esto implica que se hace el producto usual de matrices, pero los productos de las entradas se hacen como acabamos de definir.

Además, cuando se aplica $\texttt{NTT}$ ($\texttt{NTT}^{-1}$) a una matriz o un vector de elementos de $\mathcal{R}_q$, esto implica que se aplica a cada entrada de forma individual.

2.2.2. Algoritmos

En primer lugar se define una función $\texttt{Parse} : \mathcal{B}^* \longrightarrow \mathcal{R}_q$ que recibe como entrada un conjunto de bytes $B = b_0, b_1, \ldots$ y calcula la representación $\hat{a} = \hat{a}_0 + \hat{a}_1 X + \ldots + \hat{a}_{n-1} X^{n-1} \in \mathcal{R}_q$ para cierto $a \in \mathcal{R}_q$.

Algorithm 1 $\texttt{Parse} : \mathcal{B}^* \longrightarrow \mathcal{R}_q^n$

Input: Conjunto de bytes $B = b_0, b_1, \ldots \in \mathcal{B}^*$
Output: Representación $\texttt{NTT}$: $\hat{a} \in \mathcal{R}_q$ de cierto $a \in \mathcal{R}_q$

```
i := 0
j := 0
while j < n do
    d1 := b_i + 256 · (b_{i+1} mód^+ 16)
    d2 := ⌊b_{i+1}/16⌋ + 16 · b_{i+2}
    if d1 < q then
        â_j := d1
        j := j + 1
    end if
    if d2 < q and j < n then
        â_j := d1
        j := j + 1
    end if
    i := i + 3
end while
```

La intuición detrás de esta función es que si la matriz de bytes de entrada está estadísticamente cerca de una matriz de bytes uniformemente aleatoria, entonces el polinomio de salida está estadísticamente cerca de un elemento uniformemente aleatorio de $\mathcal{R}_q$.

Seguidamente, se define una nueva función. Para ello primero se define una distribución binomial centrada B_η para $\eta = 2$ o $\eta = 3$ de la siguiente manera:

$$\text{Teniendo: } (a_1, \ldots, a_\eta, b_1, \ldots, b_\eta) \leftarrow \{0,1\}^{2\eta}$$

$$\text{devuelve: } \sum_{i=1}^{\eta} (a_i - b_i)$$

Con esto, se define una función $\texttt{CBD}$ que emplee el procedimiento anterior a los coeficientes de un polinomio $f \in \mathcal{R}_q$.

Algorithm 2 $\texttt{CBD} : \mathcal{B}^{64\eta} \longrightarrow \mathcal{R}_q$

Input: Array de Bytes $B = (b_0, b_1, \ldots, b_{64\eta-1}) \in \mathcal{B}^{64\eta}$
Output: Polinomio $f = f_0 + f_1X + \ldots + f_{255}X^{255}$
$(\beta_1, \ldots, \beta_{512\eta-1}) := \texttt{BytesToBits}(B)$ ▷ Usamos la función que se define anteriormente `BytesToBits` para trabajar con un array de bits.
for i **from** 0 **to** 255 **do** ▷ Para cubrir todos los coeficientes del polinomio
  $a := \sum_{j=0}^{\eta-1} \beta_{2i\eta+j}$
  $b := \sum_{j=0}^{\eta-1} \beta_{2i\eta+\eta+j}$
  $f_i := a - b$
end for

A continuación, se describe una función $\texttt{Decode}_l$ que sirve para pasar de un array de $32l$ bytes a un polinomio $f = f_0 + +f_1X + \ldots + f_{255}X^{255}$ con cada coeficiente de $f_i \in \{0, \ldots, 2^l - 1\}$.

Algorithm 3 $\texttt{Decode}_l : \mathcal{B}^{32l} \longrightarrow \mathcal{R}_q$

Input: Array de Bytes $B \in \mathcal{B}^{32l}$
Output: Polinomio $f = f_0 + f_1X + \ldots + f_{255}X^{255}$
$(\beta_1, \ldots, \beta_{256l-1}) := \texttt{BytesToBits}(B)$
for i **from** 0 **to** 255 **do**
  $f_i := \sum_{j=0}^{l-1} \beta_{il+j} 2^j$ ▷ Supongamos $l = 2$ luego tenemos que $f_i = \beta_{i2+0}2^0 + \beta_{i2+1}2^1 = \beta_{i2} + 2\beta_{i2+1}$, y como $\beta_t \in \{0, 1\}, \forall t$ entonces se cumple que $f_i \in \{0, \ldots, 2^l - 1\} = \{0, 1, 2, 3\}$. En base a este ejemplo se puede deducir que esto se cumple para todo l y por tanto la función cumple lo que se busca.
end for

Además se define $\texttt{Encode}_l$ como su inversa, que cuando se aplique a un vector de polinomios, codifica cada uno de ellos individualmente y los concatena obteniendo un array de bytes de salida.

Un esquema de cifrado se basa en 3 algoritmos: generador de claves, cifrado y descifrado, luego vamos a ver como se describen dichos algoritmos en el esquema Kyber.

En primer lugar, se define el algoritmo de generación de claves (Algoritmo 4), donde cabe tener en cuenta que dicho algoritmo nos devuelve dos claves, una pública pk y una privada sk.

Ahora, vamos con la función de cifrado (Algoritmo 5). En este caso, se tiene como entrada la clave pública y el mensaje, y se busca que nos devuelva el texto cifrado.

Y por último se tiene el algoritmo descifrador (Algoritmo 6), que toma un texto cifrado, y devuelve un mensaje claro, por tanto toma como entradas la clave privada, el texto cifrado y devuelve el mensaje claro.

Para acabar de dejar claro el esquema de cifrado, vamos a hacer un esquema para describirlo (Figura 2.2). Para ello, supongamos que Bob quiere enviarle un mensaje a Alice, luego, en primer lugar, usando la función de generación de

Algorithm 4 Kyber: KeyGen()

Output: Clave privada $sk \in \mathcal{B}^{12 \cdot k \cdot n/8}$
Output: Clave pública $pk \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$
$d \leftarrow \mathcal{B}^{32}$
$(\rho, \sigma) := G(d)$ ▷ G es una función hash. Además tenemos que $(\rho, \sigma) \in \mathcal{B}^{32} \times B^{32}$.
$N := 0$
for i **from** 0 **to** $k-1$ **do**
  for j **from** 0 **to** $k-1$ **do**
    $\hat{A}[i][j] := \texttt{Parse}(\texttt{XOF}(\rho, j, i))$ ▷ Recordamos que `XOF` es una extendable-output que genera un número arbitrario de bits que toma la función `Parse` como entrada y devuelve la representación NTT, luego con esto generamos una matriz $\hat{A} \in \mathcal{R}_q^{k \times k}$ cuyos coeficientes son polinomios en representación `NTT`
  end for
end for
for i **from** 0 **to** $k-1$ **do**
  $s[i] := \texttt{CBD}_{\eta_1}(\texttt{PRF}(\sigma, N))$
  $N := N + 1$ ▷ Al haber aplicado $G(d)$ se tiene que $(\rho, \sigma) \in \mathcal{B}^{32} \times \mathcal{B}^{32}$ luego, en particular $\sigma \in \mathcal{B}^{32}$ luego, tiene sentido aplicar $\texttt{PRF}(\sigma, N)$ y tomando la salida como entrada a $\texttt{CBD}_{\eta_1}$ se obtiene para cada i un polinomio $f \in \mathcal{R}_q$, por tanto, en general $s \in \mathcal{R}_q^k$ bajo la distribución B_{η_1}.
end for
for i **from** 0 **to** $k-1$ **do**
  $e[i] := \texttt{CBD}_{\eta_1}(\texttt{PRF}(\sigma, N))$
  $N := N + 1$ ▷ Aplicamos el mismo razonamiento que antes, luego tenemos que $e \in \mathcal{R}_q^k$
end for
$\hat{s} := \texttt{NTT}(s)$ ▷ $\hat{s} \in \mathcal{R}_q^k$
$\hat{e} := \texttt{NTT}(e)$ ▷ $\hat{e} \in \mathcal{R}_q^k$
$\hat{t} := \hat{A} \circ \hat{s} + \hat{e}$ ▷ Con lo anterior tenemos que $\hat{t} \in \mathcal{R}_q^k$
$pk = \left(\texttt{Encode}_{12}(\hat{t} \mod^+ q) \,||\, \rho\right)$ ▷ $pk := As + e : pk \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$ (sumamos 32 de concatenar ρ)
$sk = \texttt{Encode}_{12}(\hat{s} \mod^+ q)$ ▷ Del mismo modo $sk := s : sk \in \mathcal{B}^{12 \cdot k \cdot n/8}$
return (pk, sk)

claves, Alice obtiene sk, pk. Con esto, Alice, le pasa su clave pública a Bob, con la que él, empleando la función de cifrado, consigue cifrar el mensaje que quiere enviarle a Alice. Para acabar el ciclo, Alice usando la función de descifrado y su clave privada que obtuvo al principio, consigue descifrar el mensaje que Bob le quiere enviar. De esta manera intuitiva se consigue comprender la criptografía de clave pública, y en particular, este esquema de cifrado, pues después las "cajas" no son más que funciones que a pesar de ser complejas su función es obtener como producto claves, mensajes cifrados y mensajes claros de manera segura.

Algorithm 5 Kyber: Encryption(pk,m,r)

Input: Clave pública $pk \in \mathcal{B}^{12\cdot k\cdot n/8+32}$
Input: Mensaje $m \in \mathcal{B}^{32}$
Input: Bytes aleatorios $r \in \mathcal{B}^{32}$
Output: Texto cifrado $c \in \mathcal{B}^{d_u\cdot k\cdot n/8+d_v\cdot n/8}$

$N := 0$
$\hat{t} := \texttt{Decode}_{12}(pk)$
$\rho := pk + 12 \cdot k \cdot n/8$ ▷ Tomamos de pk los bytes a partir de $12 \cdot k \cdot n/8$ que de lo anterior son los que corresponden a ρ
for i **from** 0 **to** $k-1$ **do**
  for j **from** 0 **to** $k-1$ **do**
    $\hat{A}^T[i][j] := \texttt{Parse}(\texttt{XOF}(\rho, i, j))$ ▷ $\hat{A} \in \mathcal{R}_q^{k\times k}$
  end for
end for
for i **from** 0 **to** $k-1$ **do**
  $r[i] := \texttt{CBD}_{\eta_1}(\texttt{PRF}(r, N))$
  $N := N + 1$ ▷ Siguiendo la idea anterior obtenemos $r \in \mathcal{R}_q^k$ de la distribución B_{η_1}
end for
for i **from** 0 **to** $k-1$ **do**
  $e_1[i] := \texttt{CBD}_{\eta_2}(\texttt{PRF}(r, N))$
  $N := N + 1$ ▷ En este caso tenemos la distribución B_{η_2} y además $e_1 \in \mathcal{R}_q^k$
end for
$e_2[i] := \texttt{CBD}_{\eta_2}(\texttt{PRF}(r, N))$ ▷ Obtenemos $e_2 \in \mathcal{R}_q$ por la distribución B_{η_2}
$\hat{r} := \texttt{NTT}(r)$
$u := \texttt{NTT}^{-1}(\hat{A}^T \circ \hat{r}) + e_1$ ▷ $u := A^T r + e_1$, tal que $u \in \mathcal{R}_q^k$
$v := \texttt{NTT}^{-1}(\hat{t}^T \circ \hat{r}) + e_2 + \texttt{Decompress}_q(\texttt{Decode}_1(m), 1)$ ▷ Por las dimensiones de los elementos que intevienen tenemos que $v \in \mathcal{R}_q$
$c_1 := \texttt{Encode}_{d_u}\left(\texttt{Compress}_q(u, d_u)\right)$ ▷ Aplicamos $\texttt{Compress}_q$ a $u \in \mathcal{R}_q^k$ (coeficiente a coeficiente de los polinomios) y a este nuevo vector de polinomios aplicamos la función $\texttt{Encode}_l$ devolviendonos un array de bytes tal que $c_1 \in \mathcal{B}^{d_u\cdot k\cdot n/8}$
$c_2 := \texttt{Encode}_{d_v}\left(\texttt{Compress}_q(v, d_v)\right)$ ▷ De aquí obtenemos $c_2 \in \mathcal{B}^{d_v\cdot n/8}$
return $c = (c_1 \,||\, c_2)$ ▷ Concatenamos c_1 y c_2 obteniendo $c \in \mathcal{B}^{d_u\cdot k\cdot n/8+d_v\cdot n/8}$

Algorithm 6 Kyber: Decryption(sk,c)

Input: Clave privada $sk \in \mathcal{B}^{12\cdot k\cdot n/8}$
Input: Texto cifrado $c \in \mathcal{B}^{d_u\cdot k\cdot n/8+d_v\cdot n/8}$
Output: Mensaje $m \in \mathcal{B}^{32}$

$u := \texttt{Decompress}_q(\texttt{Decode}_{d_u}(c), d_u)$ ▷ $u \in \mathcal{R}_q^k$
$v := \texttt{Decompress}_q(\texttt{Decode}_{d_v}(c + d_u \cdot k \cdot n/8), d_v)$ ▷ Recordamos que $c + d_u \cdot k \cdot n/8$ implica que tomamos de c a partir del byte $d_u \cdot k \cdot n/8$, luego tenemos que $v \in \mathcal{R}_q$
$\hat{s} := \texttt{Decode}_{12}(sk)$ ▷ $s \in \mathcal{R}_q^k$
$m := \texttt{Encode}_1(\texttt{Compress}_q(v - \texttt{NTT}^{-1}(\hat{s}^T \circ \texttt{NTT}(u)), 1))$ ▷ Con las dimensiones anteriores tenemos que $v - \texttt{NTT}^{-1}(\hat{s}^T \circ \texttt{NTT}(u)) \in \mathcal{R}_q$, luego $\texttt{Compress}_q(v - \texttt{NTT}^{-1}(\hat{s}^T \circ \texttt{NTT}(u)), 1) \in \mathcal{R}_q$, entonces tenemos que $\texttt{Encode}_1(\texttt{Compress}_q(v - \texttt{NTT}^{-1}(\hat{s}^T \circ \texttt{NTT}(u)), 1)) \in \mathcal{B}^{32}$
return m

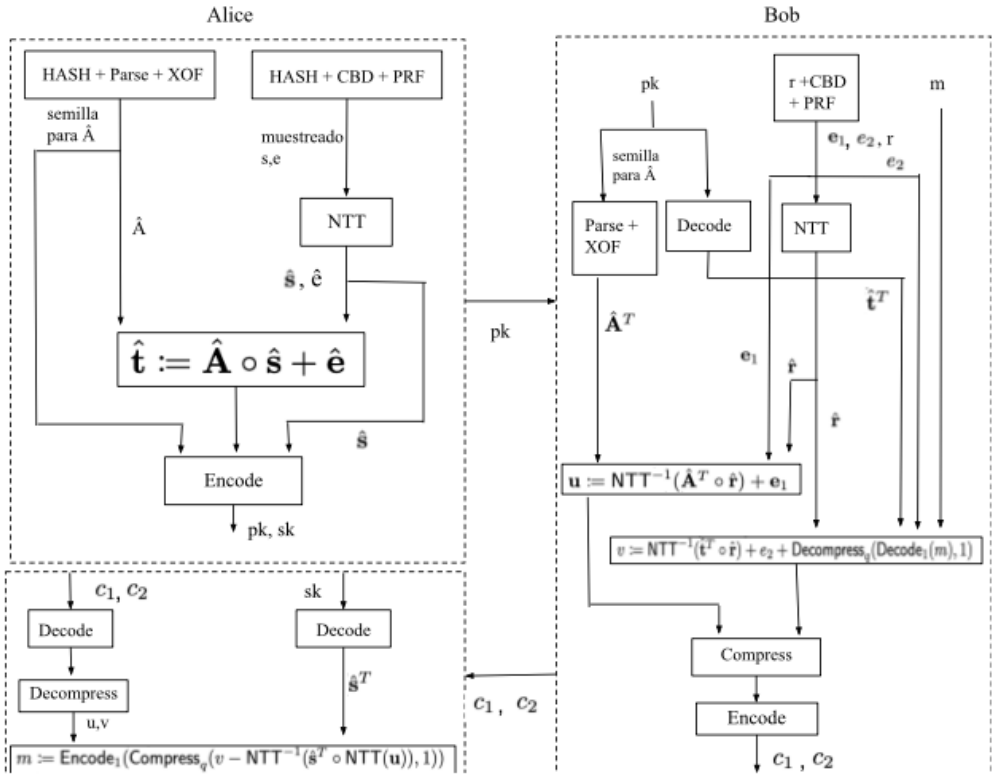


Figura 2.2: Esquema general de CRYSTALS-Kyber

3

CRYPTRIS

CRYPTRIS [21] es un videojuego creado en el INRIA (*Institut National de Recherche en Informatique et en Automatique*) y desarrollado por *Digitalcuisine*, para explorar los mecanismos de la criptografía de una manera divertida, planteando el descifrado de un mensaje como una partida de TETRIS. De hecho, las matemáticas detrás de este juego son las matemáticas de los retículos, objeto de esta memoria y del nuevo estándar de criptografía post-cuántica.

3.1. Criptografía de clave pública en CRYPTRIS

Paso 1. Codificar el mensaje.

El primer paso en el juego es convertir los mensajes en números o secuencias de números. Por ejemplo, supóngase un mensaje de dos letras m =“OK”, se codifican estos caracteres en ternario según la Tabla 3.1. Así, en lugar de bits en el juego se usan trits que pueden tomar los valores -1, 0 y +1, de forma que:

- -1 se representa con un ladrillo azul
- 0 se representa con un ladrillo azul oscuro
- +1 se representa con un ladrillo azul claro

A continuación se agrupan los trits en grupos de 4, de forma que con esos grupos se logran codificar $3^4 = 81$ símbolos. En un ordenador, que trabaja en bits, se suelen usan códigos de longitud 8, que permiten codificar $2^8 = 256$ caracteres.

Carácter a codificar	Código ternario	Carácter a codificar	Código ternario
	0 0 0 0	G	1 -1 1 1
0	1 0 0 0	H	-1 -1 1 1
1	-1 0 0 0	I	0 0 -1 1
2	0 1 0 0	J	1 0 -1 1
3	1 1 0 0	K	-1 0 -1 1
4	-1 1 0 0	L	0 1 -1 1
5	0 -1 0 0	M	1 1 -1 1
6	1 -1 0 0	N	-1 1 -1 1
7	-1 -1 0 0	O	0 -1 -1 1
8	0 0 1 0	P	1 -1 -1 1
9	1 0 1 0	Q	-1 -1 -1 1
etc...		etc...	

Con esta tabla, se ve que por ejemplo, el texto "OK" se codifica como $[0, -1, -1, 1, -1, 0, -1, 1]$.

Paso 2. Cifrar el mensaje.

Para ocultar el mensaje original y añadir confusión, se agrega ruido aleatorio varias veces. El objetivo es que un atacante que intercepte un mensaje cifrado y conozca la clave pública, no pueda realizar la operación inversa para recuperar el mensaje original.

El juego representa este procedimiento de adición de ruido de la forma siguiente. Dado que la clave es una secuencia de números, como por ejemplo, $p = [-1, 6, 8, 0, -2, -1, 7, 7]$, la codificación ternaria mencionada hace corresponder en el juego esa clave con 1 ladrillo azul, 6 ladrillos azul claro, 8 ladrillos azul claro, 1 ladrillo azul oscuro,... Así, se tienen columnas de ladrillos de distintos tamaños y colores en función de si representan números positivos, negativos o nulos. En consecuencia, se entiende que añadir ruido a la clave en el juego es manipular las columnas de ladrillos rotándolas debido a sumas, o cambiándoles el color de los ladrillos por un cambio de signo del correspondiente número. Además, tal y como está planteado el juego, se observa que el ruido es un valor pseudoaleatorio generado por el ordenador.

Estas operaciones se representan mediante un TETRIS, de forma que se observa la clave pública p en la parte superior y el mensaje original m en la parte inferior, y ambos se combinan a medida que caen las piezas. De hecho, esa combinación representa la suma de forma que cuando "cae" la clave pública sobre el mensaje, lo que sucede es que en cada columna los ladrillos se suman:

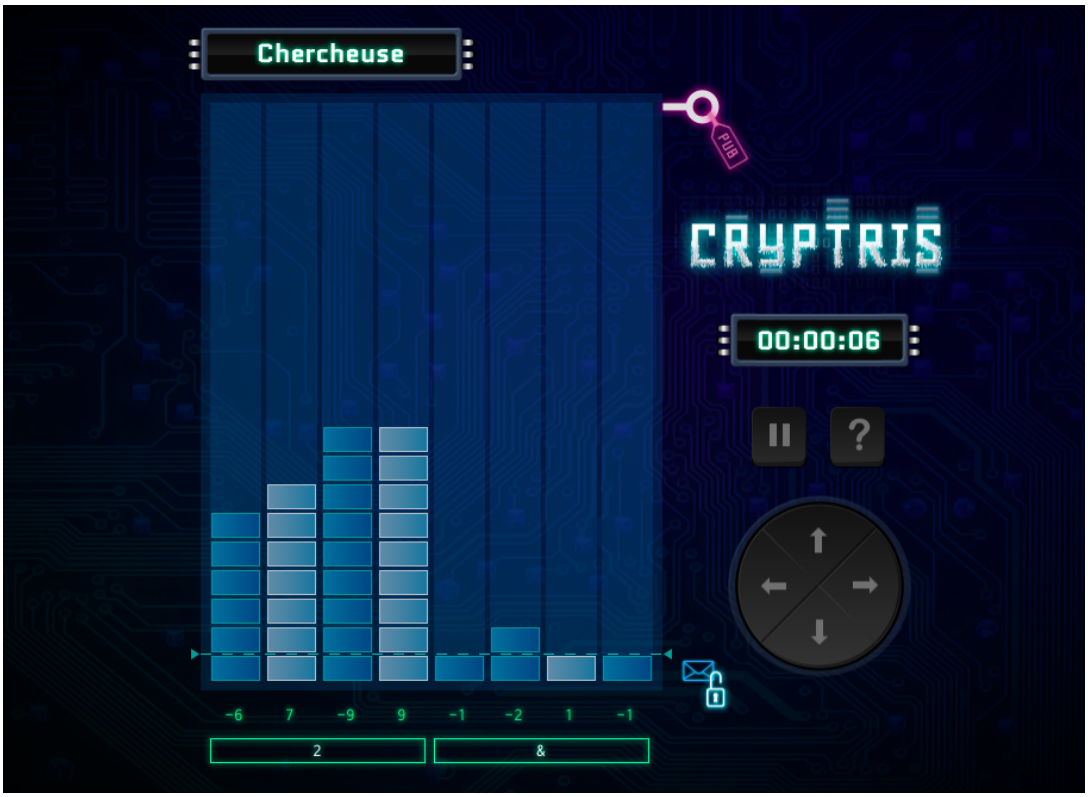


Figura 3.1: Panel de control. Aplicación de clave pública sobre mensaje original

si son del mismo color se apilan y se mantienen en ese color, y si son de distinto color se opera con los valores y podrían cambiar de color. Es decir, si un ladrillo azul claro "cae" sobre uno azul se obtiene un ladrillo azul oscuro que representa un 0. En cambio, si por ejemplo, un ladrillo azul cae sobre otro ladrillo azul, se obtienen dos ladrillos azules apilados que representan un -2.

Obsérvese que aplicando la idea anterior reiteradamente se logra añadir más confusión al mensaje original. Concretamente, al jugador se le permiten las siguientes operaciones con la clave:

- Adición: Sumar la clave dejándola caer mediante la flecha hacia abajo [↓], obteniendo así $p+m$. Además, si se repite varias veces se obtiene $2p+m, 3p+m, 4p+m, \ldots$
- Sustracción: Restar la clave, cambiándole el signo antes de sumarla, mediante la flecha hacia arriba [↑]. Así, se observa que cambian los colores de la clave debido al cambio de signo de los números. Además, si se repite varias veces se puede obtener $-p+m, -2p+m, -3p+m, \ldots$
- Rotación: Desplazar la clave mediante las flechas hacia la derecha [→] y la izquierda [←], pudiendo obtener $(x * p) + m$. Obsérvese que se trata de un desplazamiento circular de las columnas.

Por lo tanto, si p es la clave pública, el cifrado con esa clave equivale a hacer una mezcla aleatoria calculando

$$c = (a * p) + m$$

Cabe destacar que a no es un número, sino una fórmula que combina adiciones y multiplicaciones, es decir, un polinomio. Se representa con "$*$" a la mezcla y mediante "$+$" la combinación. De esta manera, el mensaje m se oculta de forma segura en el resultado c de la operación anterior, ya que para que un atacante pueda separar el mensaje m de c, tendría que enumerar todos los posibles valores de ruido a y aplicar su inversa sobre p. El número de tales combinaciones posibles es enorme, lo cual garantiza la seguridad del esquema.

Paso 3. Descifrar el mensaje

Quien ha generado una clave pública p, lo ha hecho a partir de la correspondiente clave privada s (pues, si se observa el algoritmo 4 se obtiene $\hat{s}$ y $\hat{e}$, luego operando obtenemos $\hat{t}$ y es gracias a esta variable y ρ que se obtiene pk, en cambio sk se obtiene solo a partir de $\hat{s}$) . Con la clave pública cualquiera puede cifrar mensajes, pero solo quien tiene la correspondiente clave privada puede descifrar los mensajes cifrados obtenidos.

Para generar p se realizan varias combinaciones de s consigo misma, lo que equivale en el juego a dejar caer la clave privada s sobre sí misma varias veces. Matemáticamente, esto significa que para cualquier valor g, $p = g * s$ o bien

$$p = g_0 * s + g_1 * s * X + g_2 * s * X^2 + \dots$$

siendo $X * s, X^2 * s$ las rotaciones de la clave s.

El truco está en que s tiene una forma simple, tanto en el juego como matemáticamente hablando (en el algoritmo 4 se percibe la complejidad de ambos, y se tiene que s tiene una forma más simple que p). En el juego, se compone de columnas pequeñas excepto una que es grande y es esta la forma que permite al destinatario descifrar el mensaje cifrado (Figura 3.2). El objetivo es eliminar los múltiplos de s en un mensaje cifrado $c = s * (g * a) + m$. Posteriormente se ve que, matemáticamente, para descifrar un mensaje, se reduce el cifrado c módulo s.

Esto es lo que el jugador de CRYPTRIS hace naturalmente mientras juega, reducir las columnas del mensaje, apuntar una por una a las columnas a derribar anulándolas con la columna grande de la clave. Como las otras columnas de la clave son pequeñas, afectarán poco a las otras columnas del mensaje, reduciendo así poco a poco el mensaje hasta encontrar el mensaje inicial que es solo una línea.

La clave pública no tiene esta ventaja, cada vez que un atacante intenta reducir una columna del mensaje, aumenta otra y nunca logra reducir el mensaje a una sola línea.

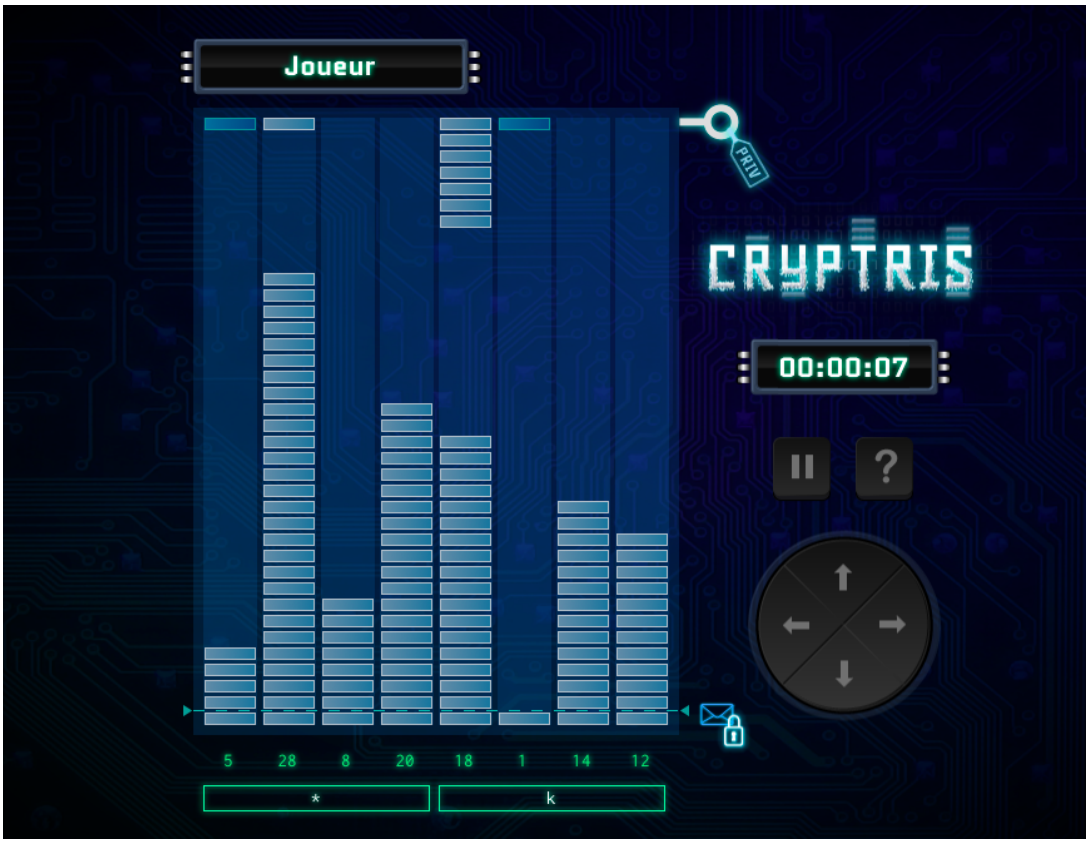


Figura 3.2: Situación descrita para descifrar el mensaje

Por tanto, la clave privada funciona bien para descifrar a diferencia de la clave pública.

Para cifrar, se tiene un mensaje m, y la clave p, luego se hace un número c agregando a m una mezcla de p, $c = (a*p)+m$ y aunque el remitente no conoce s, quien posee la clave privada sabe que esto se puede escribir:

$$c = (a * p) + m = (a * g * s) + m$$

luego, c es igual al mensaje más un múltiplo de s, y como s tiene una forma práctica, resulta que se pueden eliminar fácilmente los múltiplos de s. De hecho, se construye a propósito s con una forma práctica para ser capaces de descifrar. Quien no conoce s y solo conoce p está obligado a buscar múltiplos de p que es mucho más complicado.

La idea intuitiva que relaciona lo anterior con los retículos es la siguiente. En la clave privada, cada componente es independiente, perpendicular a las otras direcciones por lo que moverte en una dirección no modifica mucho las demás, mientras que para la clave pública las direcciones están torcidas por lo que moverse en una dirección también desplaza en las otras.

Aunque un atacante podría encontrar los factores s y g conociendo p, eso conllevaría un cálculo muy largo, pues realmente s y g no son números sino polinomios.

3.2. Criptografía basada en retículos en CRYPTRIS

Subyacentes a las operaciones descritas en la sección anterior están los retículos y problemas difíciles de resolver que hacen que romper el criptosistema sea casi imposible.

(a) s, (5,-1)

(b) X*s, (-1,5)

Se tiene la clave $s = (5, -1)$ y $X * s = (-1, 5)$, luego se pueden representar ambos en el plano coordenado y uniendo el origen con ambos puntos obtenemos dos vectores:

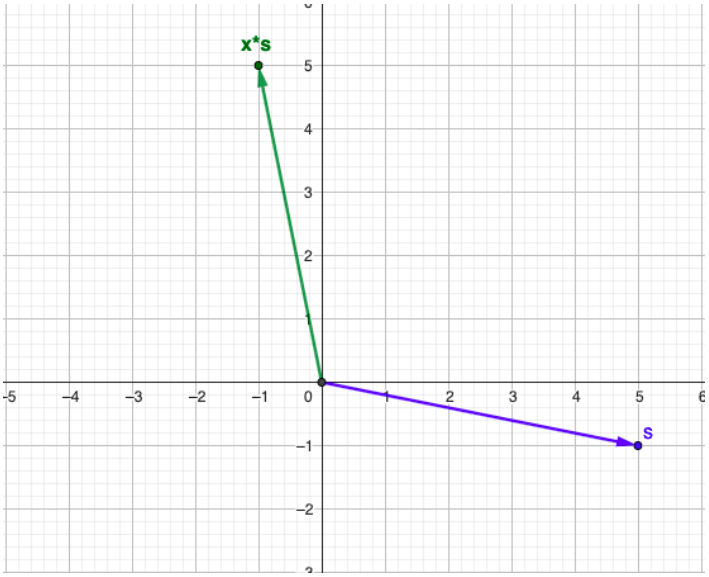


Figura 3.4: Representación en el plano

Se observa que el vector verde se obtiene al girar las coordenadas de s. En general se pueden realizar las siguientes operaciones:

- Intercambiar las coordenadas para obtener $X * s = (-1, 5)$.
- Cambiar todos los signos al mismo tiempo, para obtener $-s = (-5, 1)$.
- Sumar o restar varias veces este vector a si mismo o a uno de los transformados

Si se hacen todas las combinaciones (un número infinito de combinaciones posibles) y dibujamos estos puntos se obtiene lo que llamamos retículo:

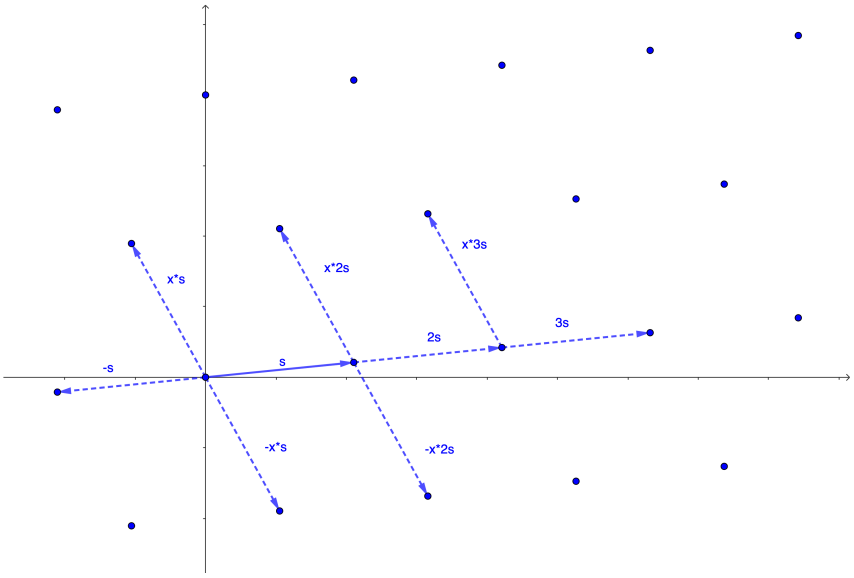


Figura 3.5: Obtención del retículo a partir de las operaciones anteriores

Todos los puntos de esta red que se extiende hasta el infinito son, por tanto, todos los códigos de la forma $r*s$, teniendo en cuenta que r no es un número sino una combinación de operaciones que se puede representar mediante el polinomio:

$$r = r_0 + r_1 * X + r_2 * X^2 + \dots$$

Por otro lado, la clave pública es otra base del mismo retículo, y esta tiene una forma "mala" (posteriormente veremos que significa mala en este contexto. Figura 3.6).

Para cifrar un mensaje usando la clave pública en primer lugar hay que tener en cuenta que el mensaje m se ve como un vector, pues gracias al código ternario un mensaje es un vector de coordenadas en el conjunto $\{-1, 0, 1\}$, por tanto, geométricamente se encuentra en el cuadrado rosa siguiente (Figura 3.7). Luego, para cifrar el mensaje m se le agrega un punto de la red elegido al azar con ayuda de la base que conforma la clave pública. Por tanto, se elige un r

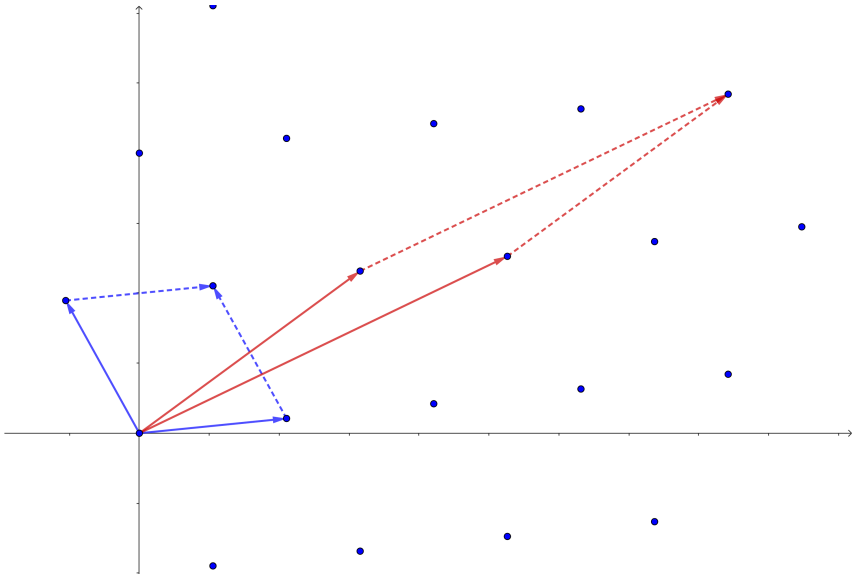


Figura 3.6: Azul: clave privada. Rojo: clave pública

aleatorio para construir el punto naranja $r * p$ en la red. Finalmente, se agrega el mensaje m a este vector obteniendo así el punto violeta $m + r * p$.

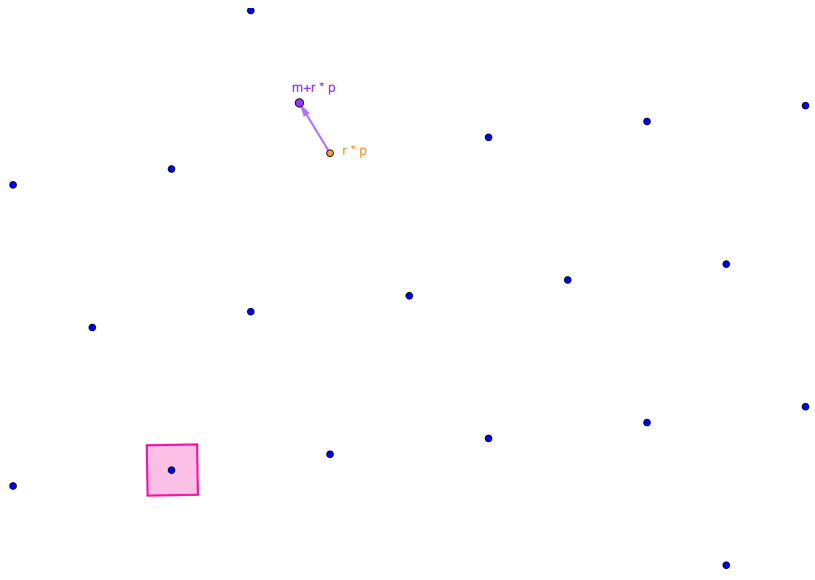


Figura 3.7: Cifrando un mensaje

Ahora falta ver cómo descifrar con la clave privada s. Se busca utilizar la base correcta s para separar el mensaje m del punto de la red que lo esconde $a * p$, para ello se utiliza una base para cortar el espacio en cajas: se dibuja

un pavimento del espacio con paralelepípedos. Se dice que la clave privada es una "buena base" por que corresponde a una referencia casi ortogonal de la red, esto es, las baldosas de la pavimentación son casi cuadradas. En este caso, se puede determinar en qué casilla de la red nos encontramos y, por lo tanto, eliminar el ruido del mensaje. Si por el contrario tomamos la clave pública para descifrar, las baldosas son muy largas y delgadas, luego si se toma el centro ya no se encuentra $r * p$ sino un valor $t * p$, por tanto se consigue lo buscado y es que cualquiera que sepa p puede cifrar pero solo aquellos que saben s pueden descifrar.

Obsérvese que las cajas asociadas a s son casi cuadradas pero las de p no lo son. La razón es que en forma de ladrillo, s tenia una columna grande y las otras pequeñas, lo que geométricamente significa que s es un vector casi horizontal cuya coordenada X es grande y su coordenada y es pequeña , lo que hace que su rotación sea casi vertical, mientras que p es una mezcla aleatoria de s, luego rompe completamente la armonía de s.

Además, no se puede usar el algoritmo de Gram-Schmidt para hacer las "cajas" cuadradas, ya que en las redes, al igual que en los espacios vectoriales, no se puede hacer una división, ya que sino nos salimos de la red y tanto el algoritmo de Gram-Schmidt como cualquier otro algoritmo de ortogonalización requiere dividir.

Cabe destacar que para encontrar el centro de las cajas en dimensión 2 basta con dibujarlas. Para un juego con más columnas, László Babai fue el matemático que dio con la mejor solución. El algoritmo requiere de ortogonalización de Gram-Schmidt e inversión del sistema triangular lo que complica los cálculos. En CRYPTRIS se usa una versión adaptada más simple, el algoritmo "glotón", cuando la clave tiene una forma práctica, con la columna grande se reduce la columna más alta del mensaje encriptado y repitiendo este proceso se reduce el mensaje hasta hacerlo pequeño. Geométricamente, cada paso nos acerca al origen hasta encontrar el mensaje m.

Para acabar y comprender mejor la multiplicación y rotación, falta por ver que sucede si multiplicamos un polinomio por X. Supongamos $p = 5X^3 - 4X^2 + 5X + 4$, luego siguiendo el procedimiento usual se tiene que:

$$X * p = 5X^4 - 4X^3 + 5X^2 + 4X$$

por tanto, se cambian todos los coeficientes un punto a la izquierda (pues los coeficientes de p recordamos que son las columnas de CRYPTRIS, esto es, en el caso de p tendríamos las columnas 5,-4,5,4). Ahora bien, como X no tiene un valor concreto, se le pueden agregar restricciones, y se va a agregar la siguiente restricción: $X^4 = 1$, es decir, hacer cálculos "módulo $X^4 - 1$". Esta restricción hace que si se piensa en la pantalla del juego, por ejemplo como en la Figura 3.2, y se quieren rotar las columnas a la izquierda, entonces el ladrillo azul que está a

la izquierda pasaría a estar a la derecha, y de este modo se obtienen coeficientes de rotación reales. Luego,

$$X * p = -4X^3 + 5X^2 + 4X + 5 \quad \text{mód } X^4 - 1$$

Con lo anterior se rota hacia la izquierda, pero para rotar hacia la derecha basta con pensar que girar a la derecha es girar tres veces a la izquierda, es decir, multiplicar por X^3.

Conclusiones

La finalidad principal de este trabajo ha sido servir de acercamiento a la criptografía post-cuántica y, en particular, al criptosistema CRYSTALS-Kyber.

El desarrollo de ordenadores cuánticos que pondrán en peligro la mayoría de los criptosistemas actuales parece cada vez más cercano. Por ello, en breve será urgente la transición hacia un nuevo estándar de cifrado post-cuántico en los productos y servicios de ciberseguridad. Precisamente la importancia del cifrado CRYSTALS-Kyber resulta de su reciente elección como estándar tras el proceso de estandarización de criptografía post-cuántica llevado a cabo por el NIST.

Para esta investigación ha sido necesario en primer lugar profundizar en varios conceptos matemáticos, conocidos, como los anillos y espacios vectoriales, y desconocidos, como son los retículos. Además ha sido necesario introducir los problemas en los que está basado el esquema de CRYSTALS-Kyber. De esa manera, ha sido posible definir el esquema de cifrado, incluyendo los algoritmos de generación de claves, cifrado y descifrado.

El trabajo se ha completado con el estudio de de un videojuego, CRYPTRIS, que simula la criptografía basada en retículos mediante una especie de TETRIS. Este juego es de especial interés ya que con unas normas básicas y la idea intuitiva de un TETRIS se consigue entender la criptografía de clave pública, y concretamente la basada en retículos. Este juego, desarrollado por autores del CRYSTALS-Kyber es de gran utilidad pues permite acercarse a conceptos que a priori parecen bastante complejos, a través de un simple juego que simplifica su comprensión. De esa manera se facilita el acercamiento a un cifrado que está llamado a ser fundamental en la era post-cuántica.

Bibliografía

[1] Langlois, A., & Stehlé, D. (2015). *Worst-case to average-case reductions for module lattices. Designs, Codes and Cryptography*, 75(3), 565-599.

[2] Ahsan Z. Zahid.(2017) *Lattices, Cryptography, and NTRU. An introduction to lattice theory and the NTRU cryptosystem*[St. Mary's College or California] [Consulta: 20-10-2022]

[3] Antonio Viruel (2020). Mirando hacia el futuro. *LA GACETA DE LA RSME*, 23, 187-204 [Consulta: 31-03-2023]

[4] BARRÓN VIDALES, J. (2008). Diseño e implementación de un esquema de cifrado híbrido basado en DHIES. *Director: Debrup Chakraborty. Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Méjico.*

[5] Micciancio, D., & Regev, O. (2009). Lattice-based cryptography. *Post-quantum cryptography*, 147-191.

[6] Chi, D. P., Choi, J. W., San Kim, J., & Kim, T. (2015). Lattice based cryptography for beginners. *Cryptology ePrint Archive.*

[7] Escribano Pablos, J. I. (2022). Criptografía segura frente a adversarios cuánticos. Análisis y variantes de propuestas para estandarización.

[8] Goldreich, O., Goldwasser, S., & Halevi, S. (1997). Public-key cryptosystems from lattice reduction problems. *In Advances in Cryptology-CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17-21, 1997 Proceedings 17* (pp. 112-131). Springer Berlin Heidelberg.

[9] Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J. M., Schwabe, P., Seiler, G., & Stehlé, D. (2018, April). CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. In 2018 IEEE *European Symposium on Security and Privacy* (EuroS&P) (pp. 353-367). IEEE.

[10] Chen, L., Chen, L., Jordan, S., Liu, Y. K., Moody, D., Peralta, R., Perlner, R., & Smith-Tone, D. (2016). *Report on post-quantum cryptography*(Vol. 12). Gaithersburg, MD, USA: US Department of Commerce, National Institute of Standards and Technology.

[11] Miguel Salgado, A. (2021). Criptografía postcuántica.
[12] Reyes Rosado, Á. R. (2018). Estado de la criptografía post-cuántica y simulaciones de algoritmos post-cuánticos.
[13] Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Shwabe, P., Seiler, G., & Stehlé, D. (2019). CRYSTALS-Kyber algorithm specifications and supporting documentation. (version 3.02) *NIST PQC Round*, 2(4), 1-43.
[14] S. González y C. Martínez (2007). Las Matemáticas de la seguridad. *ARBOR*, 725, 419-425. [Consulta: 31-03-2023]
[15] Harrigan, S. Lattice-Based Cryptography and the Learning with Errors Problem.
[16] Scarone Etchamendi, B. (2018). Criptografía post cuántica basada en reticulados: fundamentos teóricos y sistemas de clave pública.
[17] Lyubashevsky, V., Peikert, C., & Regev, O. (2013). On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6), 1-35.
[18] *Lattices, Learning with Errors and Post-Quantum Cryptography* . Notas tomadas en el curso CS294: Lattices,Learning with Errors and Post-Quantum Cryptography at UC Berkeley in Spring 2020 [Consulta: 27-10-2022]. Disponible en: https://people.csail.mit.edu/vinodv/CS294/lecturenotes.pdf
[19] Anthony Teston, Léo Ducas-Binda, Mathieu Jouhet, Thierry Viéville. *Cryptris 1/2. Comprendre une des techniques les plus sophistiquées de cryptographie en... jouant à Tetris.* Images des Mathématiques, CNRS, 2014
[20] Léo Ducas-Binda. *Cryptris 2/2. Les dessous géométriques de Cryptris : la cryptographie sur les réseaux euclidiens.* Images des Mathématiques, CNRS, 2014
[21] Link a CRYPTRIS: https://inriamecsci.github.io/cryptris/

Lista de símbolos y abreviaciones

$\mathcal{M}$	Conjunto formado por los posibles mensajes claros
$\mathcal{C}$	Conjunto formado por los posibles mensajes cifrados
$\mathcal{K}$	Conjunto formado por las posibles claves
$\mathcal{E}$	Conjunto formado por las funciones de cifrado
$\mathcal{D}$	Conjunto formado por las funciones de descifrado
$\Phi_n(x)$	n-ésimo polinomio ciclotómico
$\mathbb{Z}_n$	Anillo cociente de $A = \mathbb{Z}$ sobre el ideal $I = (n)$
SVP	Problema del vector más corto
LWE	Problema de aprendizaje con error

CRYSTALS-Kyber encryption scheme

Claudia Tarajano Afonso
Facultad de Ciencias • Sección de Matemáticas
Universidad de La Laguna
alu0101361413@ull.edu.es

Abstract

Nowadays, cybersecurity is based in how complex it is to solve certain mathematical problems efficiently using a standard computer. However, everything could change with the emergence of quantum computers, since these, based on properties such as quantum superposition, would have a greater capacity to perform calculations and manage information. Therefore, the schemes used in our current cryptography would not be secure in the post-quantum era.
This work introduces Post-Quantum Cryptography and, in particular, the CRYSTALS-Kyber encryption scheme, which is a public key cryptographic system based on lattices. In order to do so, the context in which this scheme arises is introduced, because currently the standard cryptosystem resistant to quantum computers is being chosen. Specifically, the NIST (National Institute of Standards and Technology) began a few years ago, in 2016, a process of standardization of post-quantum cryptography. Currently, after four rounds, there are 4 finalists, 3 of which are digital signatures while only 1 is a public-key cryptosystem, which is CRYSTALS-Kyber.
This encryption scheme is based on lattices, just as other participants of the contest. Therefore, this report begins with the underlying mathematical concepts in order to comprehend this concept and then be able to understand the scheme. Afterwards, this report describes the problems in which this scheme is based on: the Shortest Vector Problem (SVP) and the Module-LWE (Learning With Error). After that, we will elaborate on the key generating algorithms, the encryption algorithms and the decryption algorithms that shape the cryptographic system. Lastly, This report presents CRYPTRIS, which is a computer game similar to TETRIS, developed at INRIA (Institut National de Recherche en Informatique) under an open license to introduce the lattice-based cryptography.

1. Lattice

DEFINITION: Let $\mathbb{K}$ be a field and $\{b_1, \ldots, b_n\} \subseteq \mathbb{K}^m$ a set of linearly independent vectors $(m \geq n)$. A lattice $\mathcal{L}$ generated by $\{b_1, \ldots, b_n\}$ is defined as the set:

$$\mathcal{L} \equiv \mathcal{L}(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z} \right\}$$

of all linear combinations of $\{b_1, \ldots, b_n\}$ with integer coefficients.

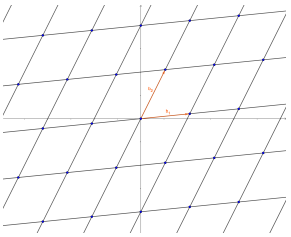


Figure 1: *Example of a lattice.*

2. Problem on which the encryption scheme is based

SVP: The shortest vector problem (SVP) is based on finding the shortest non-null vector of $\mathcal{L}(B)$, if $\mathcal{L}$ is a lattice and B is the base matrix of $\mathcal{L}$.

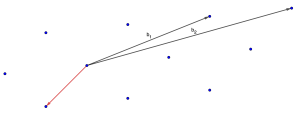


Figure 2: *SVP*

MODULE-LWE: The problem consists in being able to recover the vector s, secret from the equations: $As + e = b \mod q$, where $e \in \mathbb{Z}_q^m$ is the error and working on the ring $\mathbb{Z}_q[X]/(X^n + 1)$

3. Algorithms

First, we are going to fix the notation that is going to be used in the algorithms and then the necessary functions are defined to be able to carry out the encryption, decryption and key generation algorithms in this encryption scheme.

4. CRYPTRIS

CRYPTRIS is a video game that proposes to explore the mechanisms of cryptography in a fun way, deciphering a message is like playing a kind of tetris. Also, the math behind this game is lattice. The operation of the game, its rules and the relationship with the lattices and public key cryptography are developed.

References

[1] Anthony Teston, Léo Ducas-Binda, Mathieu Jouhet, Thierry Viéville. *Cryptris 1/2. Comprendre une des techniques les plus sophistiquées de cryptographie en... jouant à Tetris.* Images des Mathématiques, CNRS, 2014

[2] Léo Ducas-Binda. *Cryptris 2/2. Les dessous géométriques de Cryptris : la cryptographie sur les réseaux euclidiens.* Images des Mathématiques, CNRS, 2014

[3] Miguel Salgado, A. (2021). Criptografía postcuántica.

[4] Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Shwabe, P., Seiler, G., & Stehlé, D. (2019). CRYSTALS-Kyber algorithm specifications and supporting documentation. (version 3.02) *NIST PQC Round*, 2(4), 1-43.

[5] Scarone Etchamendi, B. (2018). Criptografía post cuántica basada en reticulados: fundamentos teóricos y sistemas de clave pública.