Universidad de Valladolid

FACULTAD DE CIENCIAS

TRABAJO FIN DE GRADO

Grado en Matemáticas

KYBER, una aproximación a la criptografía postcuántica basada en retículos

Autora: Lucía Olmos Vela
Tutor: José Ignacio Farrán Martín
Año 2024

Resumen

El reciente concurso público del NIST para establecer un nuevo estándar en cifrado de clave pública ha dado como ganador el cifrado KYBER. El concurso tenía como objetivo ofrecer una alternativa a los cifrados actualmente en uso (RSA y logaritmo discreto) que sea resistente a la computación cuántica. El cifrado ganador del concurso se basa en la complejidad de ciertos problemas sobre retículos, y que hasta ahora no se han podido resolver en tiempo polinomial con algoritmos cuánticos. Este trabajo analiza las debilidades de la criptografía vigente frente a la computación cuántica, los problemas complejos relacionados con la teoría de retículos, y presenta el funcionamiento del cifrado KYBER, actualmente en proceso de estandarización.

Palabras claves: Criptografía postcuántica, Retículos, CRYSTALS-KYBER.

Abstract

The recent public contest by NIST to establish a new standard in public-key encryption declared KYBER the winner. The contest aimed to provide an alternative to the currently used encryptions (RSA and discrete logarithm) that would be resistant to quantum computing. The winning encryption is based on the complexity of certain lattice problems, which so far could not be solved in polynomial time using quantum algorithms. This work analyzes the weaknesses of current cryptography in view of quantum computing, the complex problems related to lattice theory, and presents the performance of the KYBER encryption, which is currently being standarized.

Keywords: Post-Quantum Cryptography, Lattices, CRYSTALS-KYBER.

Índice general

Capítulo 1

Introducción

Con la llegada de los ordenadores cuánticos se ha vulnerado la seguridad de la criptografía de clave pública existente. Se considera que los físicos Paul Benioff y Richard Feynman fueron los primeros en relacionar los principios de la mecánica cuántica con la computación a principios de la década de 1980. Durante esta década y la siguiente continúan los avances teóricos: principio de superposición, entrelazamiento, teleportación cuántica, algoritmo de Shor, algoritmo de Grover, ... entre otros. No es hasta 1998 cuando se consiguen los primeros ordenadores cuánticos compuestos por un *qubit* (por el laboratorio Los Álamos), dos (por la Universidad de Berkeley) y tres *qubits* (gracias a los laboratorios IBM-Almaden). Actualmente se cuenta con ordenadores cuánticos de 433 *qubits*.

Volviendo al campo de interés para este trabajo, la mayoría de los sistemas criptográficos que poseen complejidad exponencial son reducidos a complejidad polinomial con un ordenador cuántico que posea una cantidad adecuada de *qubits*. Como consecuencia se consigue que el sistema se pueda romper en tiempo finito.

Por este motivo surge la necesidad de buscar sistemas criptográficos seguros ante el ataque de un ordenador cuántico que posea un número elevado de *qubits*. Este área de estudio se conoce como criptografía postcuántica. Dentro de ésta se distinguen distintas vertientes:

- Criptografía basada en retículos: existen una serie cuestiones teóricas dentro del ámbito de los retículos, denominados problemas difíciles. La complejidad de éstos se define como la dificultad de resolución del problema en el peor escenario posible. Este tipo de problemas resultan atractivos para la criptografía, tomándose como base para construir los sistemas criptográficos.

- Criptografía basada en códigos correctores de errores: en 1978 Robert McEliece introdujo un sistema de cifrado basado en un problema de decodificación. Se basa en la idea de cifrar cada mensaje como un conjunto de palabras de un determinado código lineal, que a su vez posea capacidad de corrección de errores. Este es el fundamento de este tipo de criptografía. Además tiene como objetivo intentar conseguir tamaños de clave pública asumibles en la práctica.

- Criptografía basada en funciones *hash*: principalmente se emplea para elaborar algoritmos de firma digital, y se basa en la seguridad que poseen las funciones *hash*. Estas funciones son de uso limitado para la firma, puesto que se debe emplear una clave distinta cada vez.

- Criptografía multivariante: se fundamenta en la dificultad de resolución de determinados sistemas de ecuaciones polinomiales en varias variables sobre cuerpos finitos.
- Criptografía basada en isogenias: aprovecha las ventajas de las isogenias, es decir, aplicaciones algebraicas entre curvas elípticas. El punto fundamental de los sistemas basados en ellas es encontrar una función que sea capaz de mapear una curva elíptica en otra.
- Criptografía basada en grupos de trenzas: la seguridad de sus sistemas criptográficos radica en problemas basados en el grupo de trenzas, que es una generalización del grupo simétrico.

El Instituto Nacional de Estándares y Tecnología de Estados Unidos (NIST, National Institute of Standards and Technology) con el fin de promover la investigación en este ámbito de la criptografía postcuántica, convocó en 2016 el concurso *Post-Quantum Cryptography Standardization Process*. El propósito buscado era el de conseguir la estandarización uno o más algoritmos post-cuánticos de clave pública tanto para el cifrado (PKE) e intercambio de claves (KEM) como para la firma digital.

Fueron numerosas las propuestas recibidas, de entre los que se seleccionaron sesenta y nueve candidatos. En la siguiente ronda se redujo el número de sistemas seleccionados, quedándose con diecisiete para el intercambio de clave y nueve para firma digital. En la última fase se seleccionaron siete finalistas, cuatro para cifrado e intercambio y tres para firma digital.

Finalmente, en 2022 se seleccionaron los ganadores, y por tanto los sistemas que se han convertido en estándar. CRYSTALS-KYBER se ha consolidado como el estándar en el ámbito de cifrado e intercambio de claves. Se fundamenta en la criptografía basada en retículos y desarrollarlo será el objetivo principal de este Trabajo de Fin de Grado. Posee tres variantes en función del nivel de seguridad, KYBER-512, KYBER-768, KYBER-1024. CRYSTALS-DILITHIUM, FALCON y SPHINCS+ son los estándares en el ámbito de la firma digital. Los dos primeros emplean teoría de retículos, mientras que el último emplea criptografía basada en funciones *hash*. DILITHIUM posee igualmente tres variantes en función de su nivel de seguridad Dilithium2, Dilithium3 y Dilithium5. SPHINCS+ por su parte se basa en *hashes* sin estado y posee tres variedades SPHINCS+-SHAKE256, SPHINCS+-SHA-256 y SPHINCS+-Haraka. FALCON se fundamenta en el uso de funciones de una vía para resolver el problema SIS (se detalla en la Subsección 4.3.3).

En 2023 el NIST inició una nueva ronda para encontrar nuevos estándares de firma digital que no se encuentren fundamentados en la criptografía de retículos. Esto aún se encuentra en fase de evaluación.

Este trabajo trata el problema de reducción de la complejidad computacional al que están sometidos los sistemas criptográficos de cifrado e intercambio de claves de criptografía de clave pública actuales con la llegada de los ordenadores cuánticos. Además, estudia la seguridad y ventajas del algoritmo ganador en el concurso del NIST en este ámbito, CRYSTALS-KYBER.

Para ello primeramente es necesario introducir ciertos conceptos fundamentales de la mecánica cuántica. De ello se encarga el Capítulo 2. Posteriormente se introducirán conceptos concernientes a la criptografía, centrándose en la criptografía de clave pública. Además se explicarán los principales algoritmos de cifrado e intercambio de claves

asimétricos, RSA y ElGamal. Se desarrolla su funcionamiento y el foco de su vulnerabilidad, detallando los algoritmos cuánticos que consiguen reducir la complejidad de estos algoritmos a complejidad polinomial. Para finalizar el capítulo se realiza una breve explicación del sistema criptográfico de curvas elípticas, que aunque se postulaba seguro, también se puede romper mediante un ataque cuántico. Todo esto lo recoge el Capítulo 3. El Capítulo 4 abarca todo lo referente a la teoría de retículos útil para este trabajo. Comienza con definiciones básicas para posteriormente explicar el proceso de los mínimos sucesivos. Finaliza con la explicación de los considerados problemas difíciles, que constituyen la base para el desarrollo de un sistema criptográfico seguro. El Capítulo 5 detalla el funcionamiento del algoritmo KYBER y dónde radica su seguridad frente a ataques cuánticos. Por último se desarrolla el Apéndice A donde queda recogido el pseudocódigo de los principales algoritmos empleados en KYBER.

Capítulo 2

Introducción a la computación cuántica

Richard P. Feynman, en su artículo publicado en 1982 establece un estudio sobre la eficiencia de cálculo computacional que poseen los ordenadores cuánticos, frente a los ordenadores clásicos. Aprovechando las ventajas de la mecánica cuántica como la superposición y el entrelazamiento se consigue, entre otras características, un paralelismo cuántico en el que un *qubit* puede encontrarse en un estado intermedio entre los estados 0 y 1. Este concepto no es accesible para un ordenador clásico en el que los bits poseen valor 0 o 1. Este capítulo acerca e introduce nociones cuánticas con el fin de facilitar el entendimiento del posterior desarrollo acerca del algoritmo de Shor, algoritmo que vulnera la seguridad de los sistemas criptográficos.

2.1. Qubits

El *qubit* es la mínima unidad de información de un sistema cuántico. Por hacer la analogía, un *qubit* es a un sistema cuántico, lo que un bit a un sistema clásico. Un *qubit* posee un estado, siendo los estados base el estado cero, $|0\rangle$, y el estado 1, $|1\rangle$. La notación estándar en mecánica cuántica para representar un estado es la de Dirac $|x\rangle$, indicando que el *qubit* se encuentra en el estado x. Además se pueden realizar combinaciones lineales de estos estados, obteniendo todos los estados posibles. Esto se denomina superposición,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \ ,$$

α y β son números complejos. Se cumple $|\alpha|^2 + |\beta|^2 = 1$, entendiéndose por $|\alpha|^2$ la probabilidad de que el *qubit* se encuentre en el estado cero y $|\beta|^2$ la probabilidad de que se encuentre en el estado uno.

Otra representación válida de un *qubit* es mediante un vector en un espacio vectorial complejo de dos dimensiones. Así los estados $|0\rangle$ y $|1\rangle$ forman una base ortonormal, denominada base computacional

$$\mathcal{B} = \left\{ |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\}$$

y de igual manera, todos los posibles estados son una combinación lineal de los elementos de la base

$$|\psi\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} \ ,$$

α, β pertenecientes a $\mathbb{C}$. Por simplicidad para exponer esta teoría se está empleando el caso en el cual el sistema posee un único *qubit*. Si se generaliza para n *qubits*, el espacio vectorial poseerá dimensión 2^n.

Esfera de Bloch

La expresión $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ puede ser reescrita como

$$|\psi\rangle = cos\frac{\theta}{2}|0\rangle + e^{i\phi}|1\rangle$$

con θ, ϕ ambos números reales. θ y ϕ conforman las coordenadas esféricas de un vector, tomando la esfera de radio uno. De esta forma, los estados de un *qubit* se pueden representar en esta esfera denominada esfera de Bloch (Figura 2.1).

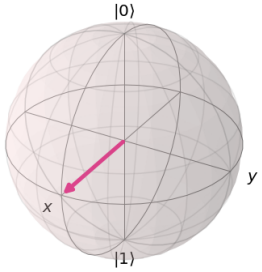

Figura 2.1: Esfera de Bloch

Proceso de medida

Se crea un estado en superposición o superpuesto, cuando el *qubit* posee de forma simultánea dos o más valores, existiendo la incapacidad de conocer con exactitud en qué estado se encuentra. El proceso de medición de varios estados en superposición obliga a que se produzca un colapso, es decir, el *qubit* que se está midiendo colapsa en un único estado $|0\rangle$ o $|1\rangle$ de forma probabilística. La superposición a su vez arroja la ventaja de obtener paralelismo cuántico, dota al sistema de la capacidad para evaluar una función dada sobre un conjunto de valores de forma simultánea.

2.2. Puertas cuánticas

De forma análoga a los circuitos eléctricos a nivel de bits con los giros y las puertas lógicas existentes, la computación cuántica también posee una herramienta para reflejar los cambios que va sufriendo en este caso un estado cuántico. El circuito cuántico está formado también por giros que transportan información a través del circuito y por puertas cuánticas que realizan manipulaciones la información. A continuación se muestran las principales puertas que se emplean en los circuitos cuánticos.

Puertas de Pauli

Las puertas o matrices de Pauli deben su nombre a Wolfgang Ernst Pauli. Son un total de tres puertas y es sencillo comprobar que forman una base en el espacio de matrices

$\mathcal{M}_{2\times 2}$. La condición de normalización sobre un *qubit* $|\alpha|^2 + |\beta|^2 = 1$ se traduce en que la matriz definida debe ser unitaria $U^\dagger U = I$, entendiendo por $U^\dagger$ la matriz adjunta de U. En realidad, cualquier matriz unitaria define una puerta cuántica válida.

Puerta X

Tiene su analogía con la puerta NOT clásica. Esta puerta envía el estado $|0\rangle$ al estado $|1\rangle$ y viceversa. De tal forma que se experimenta una rotación de π radianes alrededor del eje X en la esfera de Bloch. Se representa con la matriz

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = |0\rangle\langle 1| + |1\rangle\langle 0|$$

entendiendo por $|x\rangle\langle y|$ el producto externo. Al aplicar esta puerta sobre el estado de un *qubit* éste se ve modificado de la forma que sigue

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

$$X|1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle \ .$$

Puerta Y

Esta puerta realiza una rotación de π radianes alrededor del eje Z en la esfera de Bloch. La matriz que representa esta puerta es

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = -i|0\rangle\langle 1| + i|1\rangle\langle 0| \ .$$

Puerta Z

Esta puerta realiza una rotación de π radianes alrededor del eje Y en la esfera de Bloch. La matriz que representa esta puerta es

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = |0\rangle\langle 0| + |1\rangle\langle 1| \ .$$

Puerta de Hadamard

Esta puerta es una de las más empleadas en la construcción de circuitos cuánticos, puesto que permite a partir de cualquiera de los estados de la base computacional generar una superposición de los estados. Si se aplica el cuadrado de la matriz de Hadamard a un estado, éste queda invariante puesto que $H^2 = I$. La matriz que representa esta transformación es

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \ ,$$

que aplicada a los estados $|0\rangle$ y $|1\rangle$ se obtiene

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle$$

$$H|1\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\-1\end{bmatrix} = |-\rangle$$

considerando que los estados $|+\rangle$ y $|-\rangle$ son respectivamente

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \ .$$

La transformación realizada en la esfera de Bloch consiste en una rotación de $\frac{\pi}{2}$ radianes alrededor del eje Y, seguido de una rotación de π radianes alrededor del eje X. Por ejemplificarlo, la Figura 2.2 muestra el resultado de aplicar la puerta de Hadamard al estado $|0\rangle$

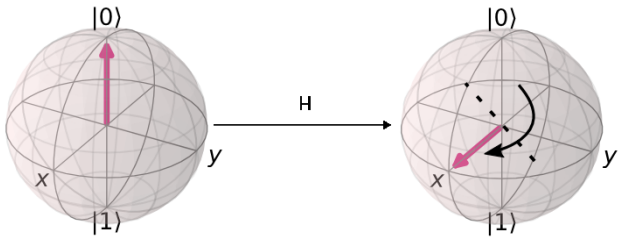

Figura 2.2: Hadamard aplicada al estado $|0\rangle$

Puertas R_ϕ

Esta puerta efectúa una rotación de ϕ radianes alrededor del eje Z. También se denomina R_z,

$$R_\phi = \begin{bmatrix}1 & 0\\ 0 & e^{i\phi}\end{bmatrix} \ .$$

Existe la puerta P que actúa de igual forma que R_ϕ y se representa con la misma matriz. Por otra parte existen algunas puertas R_ϕ para un ángulo de rotación concreto que poseen nombre por si mismas debidas a su importancia y asiduidad en la participación de los circuitos.

Puerta I

La matriz identidad constituye a su vez una puerta, dejando invariante el estado sobre el que actúa,

$$I = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \ .$$

Puerta S

Esta puerta realiza un giro de $\phi = \frac{\pi}{2}$ radianes sobre el eje Z en la esfera de Bloch. Su matriz es

$$S = \begin{bmatrix}1 & 0\\ 0 & e^{\frac{i\pi}{2}}\end{bmatrix} \ .$$

De todas las puertas relatadas hasta este punto, la puerta S es la primera que no es su propia inversa, de esta forma la puerta $S^\dagger$ realiza una rotación de $\phi = -\frac{\pi}{2}$ radianes, tomando como matriz

$$S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-\frac{i\pi}{2}} \end{bmatrix} .$$

A menudo toma el nombre de puerta $\sqrt{Z}$ puesto que al aplicar dos veces consecutivas una puerta S sobre un *qubit*, se obtiene una puerta Z:

$$SS|q\rangle = Z|q\rangle .$$

Puerta T

Esta puerta realiza una rotación de $\phi = \frac{\pi}{4}$ radianes sobre la esfera de Bloch. De nuevo T no es su propia inversa.

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix}, \quad T^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-\frac{i\pi}{4}} \end{bmatrix} .$$

Puerta U_3

Esta es la puerta más general que se puede aplicar sobre un *qubit*. Recibe como parámetros tres ángulos, el primero realiza una rotación sobre el eje Y, el segundo sobre el eje X y el último sobre el eje Z. Su matriz es

$$U_3(\theta, \phi, \lambda) = \begin{bmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i\lambda+i\phi}\cos(\theta/2) \end{bmatrix} .$$

Puerta C-NOT

La puerta C-NOT y la puerta SWAP son puertas que actúan sobre múltiples *qubits*. C-NOT es un tipo de puerta controlada o condicionada, posee uno o varios *qubits* de control y otro *qubit* objetivo. Se aplica una puerta NOT sobre el *qubit* objetivo si el *qubit* de control se encuentra en el estado $|1\rangle$ y permanecerá sin aplicarse la puerta en el objetivo si el estado del *qubit* de control es $|0\rangle$. Se puede expresar como una generalización de la puerta clásica XOR,

$$CNOT(|a, b\rangle) = |a, b \oplus a\rangle .$$

Para dos *qubits*, esta puerta se puede representar con dos matrices dependiendo cuál sea el *qubit* de control y cuál el objetivo

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad \text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} .$$

Por ejemplificar, si se toma como *qubit* de control el primer *qubit* y como objetivo el segundo, se obtendrá la siguiente transformación:

$$|00\rangle \longrightarrow |00\rangle$$

$$|01\rangle \longrightarrow |01\rangle$$
$$|10\rangle \longrightarrow |11\rangle$$
$$|11\rangle \longrightarrow |10\rangle \ .$$

De forma más general

$$\text{CNOT}|q\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} q_{00} \\ q_{01} \\ q_{10} \\ q_{11} \end{bmatrix} = \begin{bmatrix} q_{00} \\ q_{11} \\ q_{10} \\ q_{01} \end{bmatrix} \ .$$

Por lo tanto aplicando a dos *qubits* cualquiera lo que se consigue en este caso es un intercambio en las amplitudes $|01\rangle$ y $|11\rangle$.

Puerta SWAP

Realiza un intercambio entre los dos *qubits* a los que afecta. Se puede descomponer empleando rotaciones sobre un *qubit* y puertas CNOT:

$$|ab\rangle \longrightarrow |ba\rangle \ \ b, a \in \{0, 1\} \ ,$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ .$$

A su vez, también se puede expresar con puertas XOR clásicas

$$SWAP(|a, b\rangle) = |a \oplus (a \oplus b), (a \oplus b) \oplus b\rangle = |b, a\rangle \ .$$

Capítulo 3

Criptografía

La criptología es la ciencia que estudia los sistemas criptográficos. Se bifurca en dos ramas, la criptografía y el criptoanálisis. La criptografía abarca el ámbito de la implementación de los sistemas criptográficos, distinguiendo entre métodos de clave pública y clave privada. El criptoanálisis, por su parte, establece los procedimientos para romper estos sistemas. Estos pueden ser activos o pasivos, entre estos últimos cabe destacar los ataques a texto cifrado conocido, ataques a texto claro conocido y ataques a texto claro elegido.

Este capítulo pretende sentar las bases de la criptografía prestando especial interés a la criptografía de clave pública (Sección 3.2), junto con los algoritmos más importantes que se emplean en los protocolos de intercambios de claves, RSA (Sección 3.3) y ElGamal (Sección 3.4). Se fundamentan en el problema de factorización de números enteros y en el problema del logaritmo discreto. En la Sección 3.5 se abordará el algoritmo de Shor. Éste se desglosa en una fase clásica y otra cuántica. La importancia que adquiere en este trabajo es debida a la capacidad de factorizar números primos de gran tamaño, poniendo en jaque los sistemas detallados de clave pública. Finalmente, se concluirá con una breve exposición sobre las versiones basadas en curvas elípticas (Sección 3.6) de los sistemas criptográficos tratados.

3.1. Introducción a la criptografía

Es necesario primeramente establecer el contexto sobre el cual se va a trabajar. En el caso más simplificado, la criptografía permite que un emisor pueda transmitir un mensaje a un receptor de forma segura, es decir, sin que una persona ajena sea capaz de comprenderlo. De esta forma el emisor posee un mensaje en texto en claro, éste lo cifra mediante un algoritmo de cifrado y una clave obteniendo un mensaje cifrado. El mensaje cifrado es ininteligible para aquellos no involucrados en la comunicación. El emisor envía el mensaje cifrado al receptor. El receptor mediante un algoritmo de descrifrado y una clave consigue obtener el mensaje en claro correspondiente. Por lo tanto, un sistema criptográfico es un conjunto de estos elementos.

Definición 3.1.1. Un *sistema criptográfico* o *criptosistema* es una quíntupla (M, C, K, e, d), donde:

- M es el conjunto finito de mensajes o textos en claro.
- C es el conjunto finito de mensajes cifrados.

- K es el conjunto finito de claves.
- $e_k(m)$ es la función de encriptado o cifrado, que envía un elemento $m \in M$ a un elemento c del conjunto C, para un cierto $k \in K$.

$$e_k : M \longrightarrow C \ .$$

- $d_k(c)$ es la función de descifrado, que envía un elemento $c \in C$ a un elemento $m \in M$ para $k \in K$.

$$d_k : C \longrightarrow M \ .$$

Además, es necesario que se cumpla la igualdad

$$d_k(e_k(m)) = m \quad \forall m \in M \ \ \forall k \in K \ .$$

Es necesario mencionar que un elemento $m \in M$ es una sucesión finita de letras de un alfabeto A. De igual manera ocurre con los elementos $c \in C$, tratándose también de sucesiones finitas de un alfabeto B, que no necesariamente debe coincidir con A. Es decir, para cualquier $m = (m_1, ..., m_m)$, $m_i \in A$ ocurre $c = e(m)$ siendo $c = (c_1, ...c_n)$, $c_i \in B$.

El propósito de los sistemas criptográficos, y por ende, el de la criptografía es conseguir que las funciones de cifrado y descifrado, conociendo la clave k, resulten sencillas de efectuar, pero sin el conocimiento de ésta, suponga su descifrado un problema computacionalmente muy costoso. Al encontrarse dentro del marco de la compartición de secretos en las comunicaciones se plantea un escenario básico. Éste consiste en la transmisión por parte de un emisor de un mensaje $m \in M$ cifrado con una cierta clave $k_1 \in K$, $c = e_{k_1}(m) \in C$. El receptor o receptores que conocen la clave necesaria para descifrarlo $k_2 \in K$, descifran c obteniendo el mensaje original.

Los criptosistemas se clasifican en dos grupos, en función de si comparten la misma clave $k_1 = k_2$ o por el contrario difiere. En los *sistemas de clave privada*, tanto emisor como los receptores poseen la misma clave. Aunque el escenario más común ocurre cuando en la comunicación interviene un solo emisor y un solo receptor. Esto se debe a que el proceso de compartición de una clave única por un canal aparentemente seguro aumenta su vulnerabilidad cuanto mayor sea el número de personas involucradas. Además las funciones de cifrado y descrifrado son inversas entre sí $d_k = e_k^{-1}$ luego a partir de una de ellas es posible el cálculo de la otra. Esto supone un inconveniente para la seguridad, quedando el sistema más vulnerable frente a ataques. Solo sería necesario el conocimiento de una de las funciones, de la clave y del texto cifrado para obtener el mensaje en claro. Este tipo de sistemas de clave privada también se denominan *sistemas simétricos* por estar basados en funciones fácilmente inversibles.

Ejemplo 3.1.1. Por evidenciar esta situación, se presenta un tipo de sistema criptográfico de clave privada basado en códigos matriciales. Es considerado un cifrado de bloque, debido a que el mensaje es un vector fila de una longitud n dada.

$$\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{M} \ .$$

Cada x_i se considera una letra de un determinado alfabeto $\mathcal{A}$, $|\mathcal{A}| = N$. Cada una de las letras es por lo tanto un elemento del anillo $\mathbb{Z}_N$. La clave es una matriz inversible $A \in \mathcal{M}_{n\times n}(\mathbb{Z}_N)$, luego el cifrado se obtiene como

$$e(\mathbf{x}) = \mathbf{y} = \mathbf{x}A \ .$$

Y el descifrado se obtiene de la forma

$$d(\mathbf{y}) = \mathbf{x} = \mathbf{y}A^{-1} \ .$$

Numéricamente, sea $A = \begin{bmatrix} 2 & 3 \\ 7 & 8 \end{bmatrix} \in \mathcal{M}_{2\times 2}(\mathbb{Z}_{26})$. El mensaje que se pretende cifrar es "HI"$= \mathbf{x} = (8,9) \in \mathbb{Z}_{26}$. Luego

$$e(\mathbf{x}) = \mathbf{x}A = (8,9)\begin{bmatrix} 2 & 3 \\ 7 & 8 \end{bmatrix} = (1,18) \ ,$$

de forma que el mensaje cifrado se corresponde con "SQ". Para el descifrado se actua como se ha descrito:

$$d(\mathbf{y}) = \mathbf{y}A^{-1} = (1,18)\begin{bmatrix} 14 & 11 \\ 17 & 10 \end{bmatrix} = (8,9) \ .$$

Por el contrario, en los *sistemas de clave pública* cada usuario i posee dos claves, una clave pública (conocida por los restantes usuarios) y otra privada (solo el usuario propietario es conocedor de ella) (pk_i, sk_i). De esta manera un emisor j que envía un mensaje m al receptor i cifrará el mensaje con la clave pública de i, $c = e_{pk_i}(m)$. Para el descifrado es necesario el uso de la clave privada, puesto que ésta arroja información adicional para obtener el mensaje. El nivel de seguridad del sistema radica en el grado de dificultad para conseguir computacionalmente la función de descifrado a partir de la de cifrado, sin el conocimiento previo de la clave privada. Por este motivo también se denomina a los sistemas de clave pública *criptografía asimétrica*. La siguiente sección profundiza en este tipo de sistema criptográfico.

3.2. Criptografía de clave pública

En 1976 Whitfield Diffie y Martin E. Hellman en [8] introducen la noción de clave pública y sientan los pilares de la criptografía de clave pública. Pero para ello primeramente definen el concepto de intercambio de claves, estableciendo a su vez una serie de requisitos que debe cumplir todo sistema de clave pública. Éstos toman el nombre de *Condiciones de Diffie-Hellman*.

3.2.1. Intercambio de claves

Es un método por el cual se genera una clave para la comunicación, sin necesidad de un canal seguro para el intercambio de la misma. Sea $G = \langle g \rangle_n = \{g^q : q \in \mathbb{Z}_n\}$ el grupo cíclico de orden n generado por g, con g y n coprimos entre sí. A cada elemento p perteneciente al conjunto G se le puede asignar una de las claves. Es decir, existe una aplicación biyectiva $h : G \longrightarrow K_S$ del grupo cíclico en el conjunto de claves existentes para un sistema concreto S. En términos criptográficos, esta aplicación consigue codificar las claves con elementos del grupo G. En el ámbito de la comunicación entre dos usuarios A y B, cada uno de ellos escoge un número entero, del que solo cada uno tiene consciencia, $\alpha, \beta \in \mathbb{Z}$ respectivamente, obteniendo así $k_A = g^\alpha \ mod \ n$ y $k_B = g^\beta \ mod \ n$. El paso consecutivo se basa en el intercambio de los valores k_A y k_B entre los usuarios A y B. El

usuario B, calcula $k_1 = k_A^{\beta}\ mod\ n$. Por su parte, A genera $k_2 = k_B^{\alpha}\ mod\ n$. Lo ventajoso de este resultado se deduce de:

$$k_1 = k_B^{\alpha}\ mod\ n = (g^{\beta})^{\alpha}\ mod\ n = (g^{\alpha})^{\beta}\ mod\ n = k_A^{\beta}\ mod\ n = k_2\ .$$

Frente a esta igualdad, resulta seguro definir como clave de la comunicación entre ambos $k_1 = k_2 = k$. Como queda evidenciado la seguridad radica en la no resolución en tiempo polinomial del problema del logaritmo discreto. Luego esto muestran Diffie y Hellman sientan las bases teóricas y las condensan en un conjunto de postulados:

1. El algoritmo que permite obtener la clave pública y privada debe poseer complejidad polinómica.
2. El algoritmo de cifrado debe ser computacionalmente factible.
3. El algoritmo de descifrado, conociendo la clave privada, debe ser a su vez computacionalmente resoluble.
4. El algoritmo de descifrado debe poseer complejidad exponencial.
5. Aún siendo conocedor del mensaje cifrado y la clave pública, su resolución debe poseer complejidad exponencial.

Esta concepción de Diffie-Hellman admite una generalización, se puede expresar en términos de una aplicación de intercambio de claves. Para ello, previamente es necesario definir el concepto de función de una vía.

Definición 3.2.1. Sea $f : A \longrightarrow B$, se verifica que:

a. Es posible calcular en tiempo polinomial $f(a)$, para todo $a \in A$.

b. Para casi todo elemento $b \in Im(f)$, es imposible encontrar en tiempo polinómico el elemento a perteneciente a A de forma que $f(a) = b$.

Una aplicación que cumple estas características se le denomina *función de una vía.*

Dos funciones de una vía sobre las que se fundamentan los sistemas criptográficos posteriores son la factorización de un número natural y el logaritmo discreto.

Definición 3.2.2. Una función de una vía $f : A \longrightarrow B$ se denomina *función trampa*, si mediante el conocimiento de alguna información suplementaria es computacionalmente viable el cálculo de la función inversa $f^{-1} : Im(f) \longrightarrow A$.

Definición 3.2.3. Se define *protocolo de intercambio de claves* a la aplicación

$$f : K \times C \longrightarrow K$$

siendo K y C conjuntos finitos, tal que:

a. Para todo $k \in K$ y todo $c_1, c_2 \in C$, se verifica

$$f(f(k, c_1), c_2) = f(f(k, c_2), c_1)\ .$$

b. La función $k \longrightarrow f(k, c)$ es de una vía para casi todo $c \in C$.

Con estas definiciones, la clave empleada en la comunicación se obtiene como se procede a describir. Los usuarios implicados, A y B deben acordar una clave inicial (no hace falta que se comunique por un medio seguro) $k_i \in K$, y cada uno debe escoger un elemento de C, es decir c_A, $c_B \in C$ respectivamente. De esta forma A calcula $k_A = f(k_i, c_A)$, siendo $f : K \times C \longrightarrow K$ la aplicación de intercambio de claves, y trasmite el resultado a B. Éste actúa de forma semejante, calculando $k_B = f(k_i, c_B)$ y transmitiéndoselo a A. Puesto que $f(K, C)$ es una función de intercambio de claves, del cálculo de $k_1 = f(k_B, c_A) \in K$ y $k_2 = f(k_A, c_B) \in K$ y de la condición *a)* de la definición, se obtiene:

$$k_1 = f(k_B, c_A) = f(f(k_i, c_B), c_A) = f(f(k_i, c_A), c_B) = f(k_A, c_B) = k_2 \ .$$

La consecuencia principal es que $k = k_1 = k_2$ puede ser empleada como clave en la comunicación entre los usuarios A y B. Esta forma de actuar es segura, puesto que si una persona ajena quisiera interceptar la comunicación podría ser conocedora únicamente de k_A, k_B, f y k_i, sin tener oportunidad de disponer de los valores c_A y c_B. Además, debido a la segunda condición de la definición de f, ésta es una función de una vía y no es computacionalmente posible obtener su inversa sin poseer información adicional.

3.2.2. Sistema criptográfico de clave pública

Tras lo abordado, nos encontramos en condiciones de definir formalmente el concepto de sistema criptográfico de clave pública.

Definición 3.2.4. Sea M el conjunto finito de mensajes, C el conjunto de textos cifrados, K y K' conjuntos que albergan claves. Un *sistema criptográfico de clave pública* o *sistema criptográfico asimétrico* es una tupla (K, K', M, C, c, e_k, $d_{k'}$) donde:

- $c : K' \longrightarrow K$ es una función de una vía, que envía una clave $k' \in K'$ en un elemento del conjunto K. Esta aplicación c se denomina *algoritmo de generación de claves.*
- e_k es una aplicación de cifrado,

  $$e_k : M_k \longrightarrow C_k \quad \forall k \in K$$

  siendo para casi todo $k \in K$ una función de una vía.
- $d_{k'}$ es una aplicación de descifrado,

  $$d_{k'} : C_{k'} \longrightarrow M_{k'} \quad \forall k \in K \ .$$

  Su aplicación debe ser computacionalmente factible si se posee la clave k' y de complejidad exponencial si no se está en disposición de ella.

Además, debe satisfacer si $k = c(k')$ $\forall m \in M_k$, entonces:

- $M_k = M_{k'}$.
- $C_k = C_{k'}$.
- $d_{k'}(e_k(m)) = m$.
- $d_k^{-1} \circ e_k = 1_M$.

k' se denomina *clave privada* y k *clave pública* de la comunicación. La Figura 3.1 muestra un esquema de lo descrito.

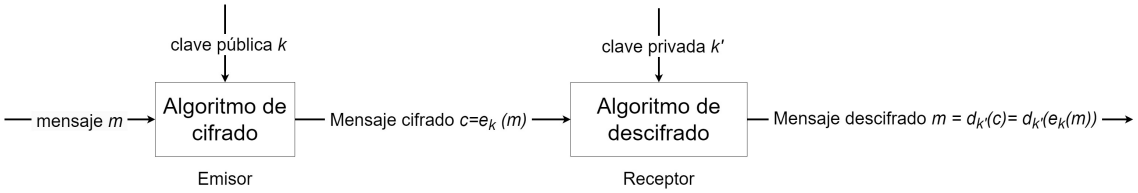

Figura 3.1: Esquema criptografía clave pública

3.2.3. Seguridad semántica

En 1984 Shafi Goldwasser y Silvio Micali introducen la noción de seguridad semántica equivalente al concepto de indistinguibilidad, mientras que Danny Dolev, Cynthia Dwork y Moni Naor tratan el concepto de no maleabilidad como una extensión a la noción de seguridad semántica.

La seguridad semántica es un concepto que abarca el hecho de que los mensajes cifrados no arrojen ninguna información sobre los mensajes en claro correspondientes. Se puede dar tanto en sistemas de clave privada como de clave pública, adquiriendo especial relevancia en estos últimos. El estudio detallado a continuación contempla exclusivamente el ámbito de los criptosistemas asimétricos. Este tipo de seguridad se plantea en términos de un modelo objetivo-ataque, donde el ataque consta de un adversario que posee acceso al resultado que arroja un simulador del algoritmo de cifrado. Un simulador del algoritmo de cifrado es una caja negra en la que al introducir un mensaje en claro devuelve ese mensaje cifrado con el algoritmo de cifrado escogido. El propósito de este adversario es traspasar los objetivos de indistinguibilidad (Definición 3.2.5) y de no maleabilidad (Definición 3.2.6) de un mensaje cifrado.

Definición 3.2.5. Dado un sistema criptográfico de clave pública $\alpha = (K, K', M, C, c, e_k, d_{k'})$ y dado un adversario $a \in A$ se sigue el procedimiento:

- Tras la aplicación del algoritmo de generación de claves se obtiene p_k que se proporciona al adversario.
- El adversario escoge dos mensajes m_1 y m_2 para ser cifrados y se lo envía al simulador.
- El simulador elige un bit $b \in \{0, 1\}$ y calcula $c_b = e_{p_k}(m_b)$. De esta forma el adversario posee c_b, pero no el bit b.

Bajo estas condiciones, se dice que el sistema α salvaguarda la *indistinguibilidad del cifrado*, IND, si para cualquier adversario $a \in A$ éste no puede adivinar, con probabilidad superior a $\frac{1}{2}$ cual ha sido el bit escogido, es decir, si el texto cifrado c_b se corresponde con m_1 o con m_2. También se dice que α es *semánticamente seguro*.

Definición 3.2.6. Dado un sistema criptográfico de clave pública $\alpha = (K, K', M, C, c, e_k, d_{k'})$, dado un adversario $a \in A$ y un algoritmo de muestreo encargado de generar conjuntos de mensajes $\{m \in M\}$, se sigue el procedimiento:

- Tras la aplicación del algoritmo de generación de claves se obtiene p_k que se proporciona al adversario.

- Se elige un mensaje del conjunto arrojado por el algoritmo de muestreo y se cifra con el algoritmo correspondiente $c = e_{p_k}(m)$.

- El adversario, con el conocimiento de la clave pública y el mensaje cifrado c debe conseguir encontrar n mensajes cifrados $c_1, \ldots, c_n$ con $c_i \neq c \quad \forall i = 1..n$, de tal manera que los mensajes correspondientes a esos mensajes cifrados guarden una relación conocida con m.

Se dice que el sistema criptográfico es *no maleable*, NM, si la probabilidad de que cualquier adversario sea capaz de obtener los n mensajes cifrados y sus correspondientes mensajes en claro no sea mayor que la probabilidad de esta misma situación pero con una cadena de bits elegida al azar dentro del espacio muestral del conjunto de mensajes generados por el algoritmo de muestreo.

Tal y como establecieron Goldwasser y Micali, estas definiciones han sido dadas en función de un ataque en el cual el adversario actúa de forma pasiva. En la actualidad, ha surgido la necesidad de ampliar el espectro, de forma que nos encontramos con distintos tipos de ataques:

- *CPA seguridad (Chosen-plaintext attack)*: el adversario únicamente posee un simulador del algoritmo de cifrado. Se trata de usuarios pasivos por definición.

- *CCA seguridad (Chosen-ciphertext attack)*: se provee al adversario de un sistema que descifra cualquier mensaje cifrado, aunque solo puede hacer uso de él antes de recibir el reto. Este sistema actúa como una caja negra, ocultando la lógica subyacente y arrojando el mensaje en claro correspondiente al texto cifrado introducido. Este sistema con estas propiedades se denomina *oráculo*.

- *CCA2 seguridad (Adaptative chosen-ciphertext attack)*: en este caso, el adversario es poseedor de un oráculo que tiene la capacidad de descifrar cualquier mensaje cifrado, a excepción del dado en el reto, por tiempo ilimitado.

Tras sentar estas definiciones, lo que Goldwasser y Micali trataron fue IND-CPA, es decir, la indistinguibilidad frente a adversarios pasivos. Siguiendo la misma nomenclatura, se establecen esquemas en función del objetivo tratado y del adversario existente: IND-CCA, IND-CCA2, NM-CPA, NM-CCA, NM-CCA2. Las distintas implicaciones entre estos esquemas se muestran en la Figura 3.2, obtenida de [3] .

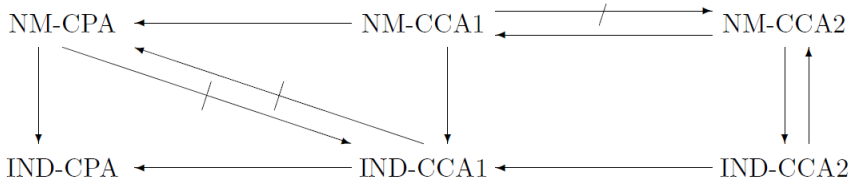

Figura 3.2: Implicaciones

3.3. Algoritmo RSA

En 1978 R.L. Rivest, A. Shamir y L. Adleman, integrantes del Instituto Tecnológico de Massachusetts (MIT) presentan un algoritmo de cifrado de clave pública [17], RSA. Anteriormente, en 1973 Clifford Cocks elaboró un sistema equivalente, pero este resultado no trascendió debido al elevado coste de las computadoras que requería en su implementación. La seguridad del RSA reside en el hecho de que cualquier función de factorización de un número natural es una función de una vía. Además aprovecha el hecho de que que computar su función inversa posee complejidad exponencial.

3.3.1. Procedimiento

Cada usuario $i \in \mathbb{N}$, $i \neq \infty$, del sistema debe escoger dos números primos p_i y q_i, con $p_i \neq q_i$. Sea $n_i = p_i q_i$, sea la función de Euler $\varphi(n_i) = \varphi(p_i)\varphi(q_i) = (p_i - 1)(q_i - 1)$. Un usuario elige un número arbitrario e_i tal que $0 < e_i < \varphi(n_i)$ y que sea coprimo con $\varphi(n_i)$, es decir $mcd(e_i, \varphi(n_i)) = 1$. Además debe cumplirse $d_i = e_i^{-1} \ mod \ \varphi(n_i)$. Ante estas suposiciones, se está en condiciones de establecer las claves pública y privada que rigen la comunicación:

- Clave pública: $(n_i,\ e_i)$.
- Clave privada: d_i.

Y las funciones de cifrado y descifrado se definen como sigue:

- La función de cifrado
$$f_i : \mathbb{Z}_{n_i} \longrightarrow \mathbb{Z}_{n_i}$$
definida como $m \longrightarrow c \equiv m^{e_i} \ \ mod \ n_i$.
- La función de descifrado
$$g_i : \mathbb{Z}_{n_i} \longrightarrow \mathbb{Z}_{n_i}$$
definida como $c \longrightarrow m \equiv c^{d_i} \ \ mod \ n_i$.

La función de descrifrado permite que el receptor pueda obtener el mensaje en claro, es decir, está bien definida como muestra el Teorema 3.3.1.

Teorema 3.3.1. Con la notación anterior, se verifica que

$$g_i(f_i(m)) = m \ \ \forall m \in \mathbb{Z}_{n_i} \ .$$

Demostración. Es necesario comprobar que $m \equiv m^{e_i d_i} \ \ mod \ n_i$, $\forall i \in \mathbb{N}$ y $\forall m \in \mathbb{Z}, m \neq 0$. De esta forma se daría la igualdad requerida

$$g_i(f_i(m)) = g_i(m^{e_i}) = m^{e_i d_i} = m \ .$$

Se distinguen tres casos:

- *Caso 1.* Si p_i y q_i no dividen a m, $p_i \nmid m$ y $q_i \nmid m$. Puesto que
$$d_i = e_i^{-1} \ mod \ \varphi(n_i)$$

es decir, $e_i d_i = 1 + z\varphi(n_i)$ para algún elemento $z \in \mathbb{Z}$, para todo i se verifica la cadena de igualdades:

$$\begin{aligned} g_i(f_i(m)) &\equiv m^{e_i d_i} \ \ mod\ n \\ &\equiv m^{(1+z\varphi(n_i))} \ \ mod\ n_i \\ &\equiv m m^{z\varphi(n_i)} \ \ mod\ n_i \\ &\equiv m(m^{\varphi(n_i)})^z \ \ mod\ n_i\ . \end{aligned}$$

Puesto que p y q no dividen a m, $n_i = pq$ tampoco divide a m. Luego $mcd(m, n_i) = 1$ y aplicando el Teorema de Euler

$$m^{\varphi(n_i)} = 1$$

y

$$g_i(f_i(m)) \equiv m \ \ mod\ n_i \ \ \forall i \in \mathbb{N}\ ,$$

como se quería probar.

- *Caso 2.* Si p_i divide a m y q_i no divide a m, $p \mid m$ y $q \nmid m$. Teniendo en cuenta que $e_i d_i = 1 + z\varphi(n_i)$ para algún elemento $z \in \mathbb{Z}$ y haciendo uso de las propiedades de la función de Euler $\varphi(n_i) = \varphi(p_i)\varphi(q_i)$, se verifica la cadena de igualdades:

$$\begin{aligned} g_i(f_i(m)) &\equiv m^{e_i d_i} \ \ mod\ n_i \\ &\equiv m^{(1+z\varphi(n_i))} \ \ mod\ n_i \\ &\equiv m m^{(z\varphi(p_i)\varphi(q_i)} \ \ mod\ n_i \\ &\equiv m(m^{\varphi(q_i)})^{z\varphi(p_i)} \ \ mod\ n_i\ . \end{aligned}$$

De forma análoga al caso anterior, puesto que $mcd(m, q_i) = 1$, por el teorema de Euler

$$m^{\varphi(q_i)} = 1$$

y

$$g_i(f_i(m)) \equiv m \ \ mod\ n_i \ \ \forall i \in \mathbb{N}\ ,$$

como se pretendía probar.

- *Caso 3.* Si q_i divide a m y p_i no divide a m, $q_i \mid m$ y $p_i \nmid m$. Se razona de forma análoga al *Caso 2.*

□

Ejemplo 3.3.1. Este ejemplo muestra el procedimiento relatado empleando números primos pequeños para mayor claridad. Sea $p = 1553$ y $q = 1661$. De tal manera que $n = pq = 2579533$, $\varphi(n) = 2576320$. Se elige arbitrariamente $0 < e = 65537 < \varphi$ y coprimo con φ. De esta forma

$$d \equiv e^{-1} \ \ mod\ \varphi(n)$$

$$2511233 \equiv 65537^{-1}\ mod\ \ 2576320\ .$$

En estas condiciones para cifrar un mensaje se ha identificando previamente cada letra del alfabeto con el orden lexicográfico con un número en su orden natural. El mensaje HOLA se correspondería con:

$$8 \cdot 26^3 + 15 \cdot 26^2 + 12 \cdot 26 + 1 = 151061 \ ,$$

luego el mensaje cifrado es

$$c \equiv 151061^{65537} \ \ mod\ 2579533$$

$$1327561 \equiv 151061^{65537} \ \ mod\ 2579533$$

$$1327561 = 2 \cdot 26^4 + 23 \cdot 26^3 + 13 \cdot 26^2 + 22 \cdot 26 + 1$$

y alfabeticamente se corresponde con BWMVA. El descifrado se realiza como se ha descrito

$$m \equiv 1327561^{2511233} \ \ mod\ 2579533$$

$$151061 \equiv 1327561^{2511233} \ \ mod\ 2579533 \ .$$

3.3.2. Consideraciones

Se puede comprobar que el sistema RSA cumple las condiciones de Diffie-Hellman. Las funciones de cifrado y descifrado se basan en el cálculo de una potencia en $\mathbb{Z}_{n_i}$, de forma que son sencillas de computar. Además, al elegir de forma aleatoria el número e, es necesario comprobar que sea coprimo con $\varphi(n)$. Para esta labor se hace uso del algoritmo de Euclides. Mientras, el algoritmo de Euclides extendido $xe + y\varphi(n) = 1$ arroja los coeficientes de Bezout $x, y \in \mathbb{Z}$. Por tanto se puede escoger $d \equiv x \quad mod\ \varphi(n)$. La complejidad polinómica viene dada en ambos casos como consecuencia del empleo del algoritmo de Euclides, algoritmo que posee dicha complejidad. Esto verifica las condiciones 1), 2) y 3). Por otro lado, conociendo la clave pública (n_i, e_i), para conseguir descifrar el mensaje es necesario el conocimiento de d_i y por ende el conocimiento de $\varphi(n_i)$. Esto conlleva que sea necesario conocer la factorización de n_i, para conseguir los elementos p_i y q_i. Esta factorización no es posible realizarla en tiempo polinomial. Esto verifica las condiciones 4) y 5).

Es necesario remarcar, que para que esta factorización posea complejidad exponencial, los números primos escogidos p y q deben ser lo suficientemente grandes. Con la capacidad computacional actual, se considera que deben poseer más de cien cifras. Por otra parte, a la hora de escoger los primos p y q hay que tener en consideración que sean lo más resistentes posibles a los algoritmos existentes de factorización de números enteros, como el algoritmo de Pollard o el de Fermat. Para evitar la factorización por el método de Pollard, es necesario que $p - 1$ y $q - 1$ no tengan todos sus factores primos pequeños. Mientras que es necesario que los números p y q no se encuentren próximos, para evitar su factorización por el método de Fermat. Existe una elección ventajosa de los mismos, se escogen dos números primos auxiliares n y m con un número elevado de cifras, ambos de longitudes similares (pero no la misma). Con esta elección se consigue que

$$p = 2n + 1 \qquad q = 2m + 1$$

sean a su vez primos. En este caso, p y q se denominan primos seguros.

Además para la elección de los primos p y q, éstos se someten a un test de primalidad. Existen test tanto deterministas como probabilísticos. Los test probabilísticos arrojan con una probabilidad alta si un número es primo o no. Existen numerosas de estas pruebas de primalidad entre las que destacan el Teorema de Wilson, el test de Fermat, el test de Miller-Rabin o el test de Solovay-Strassen. Como test determinista de complejidad polinomial se encuentra el algoritmo AKS.

3.4. Algoritmo ElGamal

Taher ElGamal introduce en 1985 en [9] un nuevo algoritmo empleado en el criptosistema de clave pública que lleva su mismo nombre. El hecho de que la función de descifrado posea complejidad exponencial está fundamentado en el problema del logaritmo discreto.

Proposición 3.4.1. Sea $n \in \mathbb{N}$, sea $G = \langle g \rangle$ un grupo cíclico de orden n. La función $f : \mathbb{Z}_n \longrightarrow G$ definida como $f(x) = g^x$ es una función biyectiva.

Definición 3.4.1. En las condiciones de la Proposición 3.4.1, se denomina *logaritmo discreto* a la función inversa $f^{-1}(y) = \log_g(y)$, siendo $g^{(f(y)^{-1})} = y$.

ElGamal hace uso de esta formulación, aprovechando que si se define n lo suficientemente grande, entonces f se considera como función de una vía. De esta forma brinda seguridad al sistema criptográfico.

3.4.1. Procedimiento

Un usuario A elige un número primo p, de forma que sea lo suficientemente grande, evitando así que se pueda resolver el problema del logaritmo discreto. Sea g un generador del grupo cíclico $\langle g \rangle = \mathbb{Z}_p^*$. Este usuario elige n, de forma que $1 \leq n \leq p-2$. De esta forma, la clave pública se conforma con p, g y $q = g^a$, (p, g, q), mientras que la clave privada se corresponde con n. De esta forma, el usuario B que desea enviar un mensaje m, elije un número natural arbitrario $k_b \in \mathbb{N}$. En estas condiciones, la función de cifrado arroja el par $(c_1, c_2) \in \mathbb{Z}_p \times \mathbb{Z}_p$ definido como $c_1 \equiv g^{k_b} \mod p$, $c_2 \equiv m\, q^{k_b} \mod p$. La función de descifrado

$$(c_1, c_2) \longrightarrow c_2 (c_1^n)^{-1} \ .$$

Se comprueba de forma sencilla que $c_2(c_1^n)^{-1}$ coincide con el mensaje m enviado por el usuario B.

$$\begin{aligned} c_2(c_1^n)^{-1} &\equiv (mq^{k_b})(g^{nk_b})^{-1} \mod p \\ &\equiv (mq^{k_b})(q^{k_b})^{-1} \mod p \\ &\equiv m \mod p \ . \end{aligned}$$

3.4.2. Consideraciones

En el problema descrito si es posible resolver en tiempo polinomial el logaritmo discreto, sería posible obtener k_b de la equivalencia $c_1 \equiv g^{k_b} \mod p$ y por consiguiente el mensaje en claro de $c_2 \equiv m\, q^{k_b} \mod p$. Para evitar esta situación es necesario tomar en cuenta una serie de consideraciones, cuyo procedimiento detallado no es objeto de este estudio particular.

Como observación, como grupo cíclico se puede considerar el grupo multiplicativo de un cuerpo finito $\mathbb{K}$, los puntos de una curva elíptica, puesto que conforman un grupo abeliano, una variedad de Jacobi de una curva hiperelíptica. En el caso $\mathbb{Z}_p^*$, una correcta elección de p es un primo seguro grande $p = 2q + 1$, con q un número primo. Métodos concretos de ataque a este sistema criptográfico explotando la debilidad del logaritmo discreto se pueden encontrar en [10]. Ligado al problema del logaritmo discreto, también se debe mencionar el sistema de intercambio de claves de Diffie y Hellman. Debido a que conociendo g^α y g^β el conocimiento de la clave se fundamenta en resolver el problema tratado y así extraerla $g^{\alpha\beta}$

3.5. Amenaza al RSA

Como se ha evidenciado la seguridad de RSA radica en la no resolución del problema de factorización, mientras que la de ElGamal se fundamenta en la no resolución del problema del logaritmo discreto. Ambas cuestiones poseen por el momento complejidad exponencial. Pero con la llegada de los ordenadores cuánticos y el empleo de un número elevado de *qubits*, ambos problemas serán reducidos a una complejidad polinomial $\mathcal{O}(n^2 \log n \log\log n)$.

Esta sección va a tratar el procedimiento teórico de factorización de un número N suficientemente grande como producto de dos primos. Como consecuencia de esto se consigue obtener la clave privada de una comunicación en la cual se ha empleado como método de cifrado de clave pública el algoritmo RSA. Este método posee una parte clásica y otra cuántica. La parte cuántica se centra en conseguir establecer el orden de un elemento perteneciente a un grupo cíclico (Algoritmo de Shor), mientras que la clásica aprovecha este resultado para realizar la descomposición de un número en factores primos. Se presentan seguidamente los pasos que es necesario seguir para factorizar un número n como producto de dos primos. Los detalles y justificaciones de su funcionamiento se encuentran en las Subsecciones 3.5.1, 3.5.2 siguientes.

1. Elegir un número a, $a \in (2, n-2)$, coprimo con n.

2. Calcular el máximo común divisor entre a y n. En este paso es útil el empleo del Algoritmo de Euclides.

3. Si $mcd(a, n) \neq 1$, entonces se finaliza debido a que es un factor de n.

4. Si $mcd(a, n) = 1$, se define la función $f(i) = a^i \mod n$ y se halla su periodo, coincidiendo como se ha visto con el orden r del elemento a. En esta etapa entra en juego la parte cuántica con el Algoritmo de Shor (calcular el estado superpuesto, aplicar la transformada de Fourier, aplicar la transformación inversa y medir para calcular el periodo).

5. Si el periodo obtenido es impar, se escoge un nuevo número y se vuelve a repetir el procedimiento.

6. Si $a^{r/2} \equiv -1 \mod n$ se repite de nuevo el procedimiento con otro elemento a.

7. Finalmente los factores primos de n buscados son

$$mcd(a^{r/2} + 1, n) = p \;\; y \;\; mcd(a^{r/2} - 1, n) = q \; .$$

3.5.1. Algoritmo factorización, parte clásica

La parte clásica abarca todos los pasos anteriores a excepción del cuarto. El cálculo del periodo de la función dada si se calcula de forma clásica posee complejidad exponencial. Este es el motivo por el cual es preferible desarrollar las bases teóricas del cálculo del periodo de una función de forma cuántica. Para las etapas que abarcan la parte clásica, se muestran una serie de definiciones y proposiciones empleadas en este procedimiento y que justifican las sucesivas elecciones y generalizaciones.

Definición 3.5.1. Sean a, b enteros tal que $mcd(a, b) = 1$. Se define el *orden* de a módulo b como el menor entero positivo $r \in \mathbb{Z}^+$ tal que

$$a^r \equiv 1 \ \ mod\, b \ .$$

b es un número primo con más de cien cifras para partir de un sistema criptográfico lo más seguro posible. Por lo tanto, el proceso para encontrar el orden clásicamente siguiendo esta definición posee complejidad exponencial.

Definición 3.5.2. Sea a un número natural $a \in \mathbb{N}$, $b \in \mathbb{Z}$. a es una *raíz cuadrada módulo* b si $a \in (2, b-2)$ y

$$a^2 \equiv 1 \ \ mod\, b \ .$$

Proposición 3.5.1. Sean p, q dos números primos, $p \neq q$, tales que $n = p \cdot q$. Hallar la raíz cuadrada no trivial módulo n implica conseguir los factores p y q.

Demostración. Sea a una raíz cuadrada módulo n, $a \neq 0$. Es decir, se tiene

$$a^2 \equiv 1 \ \ mod\, n \ , \quad a \in (2, n-2) \ .$$

Luego se obtiene la cadena de equivalencias siguiente

$$a^2 \equiv 1 \ \ mod\, n \iff$$

$$a^2 - 1 \equiv 0 \ \ mod\, n \iff$$

$$(a+1)(a-1) \equiv 1 \ \ mod\, n \ ,$$

esto implica que el producto $(a+1)(a-1)$ es múltiplo de n, $(a+1)(a-1) = kn$ para cierto $k \in \mathbb{N}$. Esto implica que, puesto que ocurre por definición que n es producto de dos primos $n = p \cdot q$, es necesario constatar que p es un factor de $(a+1)$ y q un factor de $(a-1)$, o viceversa. Puesto que a pertenece al intervalo $(2, n-2)$, tanto $a+1$ como $a-1$ son menores que n. Luego ninguno de los factores puede ser múltiplo de n. De esta forma, la única opción aceptada es que $x+1$ tenga como factor a p y $x-1$ posea entre sus factores a q. Además,

$$mcd(x+1, n) = p, \quad mcd(x-1, n) = q \ .$$

Y empleando el Teorema Fundamental de la Aritmética, queda demostrado lo que se pretendía probar, $n = p \cdot q$. □

De esto se deduce que para realizar la factorización de un número n basta con conseguir hallar a, la raíz cuadrada no trivial modulo n. Ahora la cuestión fundamental radica en la búsqueda de a. Para ello se define la función

$$f(i) = a^i \mod n$$

con $a \in (2, n-2)$. Es sencillo comprobar que esta función es periódica y su periodo t es precisamente el orden de a modulo n. Por otra parte, para calcular la raíz cuadrada no trivial modulo n se realiza un proceso iterativo, escogiendo $a \in (2, n-2)$ tal que $mcd(a, n) = 1$, se construye la función $f(i) = a^i \mod n$ y se calcula su periodo. Aunque a priori, por lo establecido lo que se requiere encontrar es $i = 2$ para que a sea una raíz cuadrada modular, basta con que el periodo sea par, puesto que

$$a^p \mod n = (a^{p/2})^2 \mod n$$

con p par, siendo $a^{p/2}$ la raíz cuadrada no trivial buscada. Por otra parte, el siguiente Teorema 3.5.1 junto con el Lema 3.5.1 previo permite arrojar información sobre una elección ventajosa de $a \in (2, n-1)$.

Lema 3.5.1. Sea $p \in \mathbb{N}$ un número primo impar, sea α un entero positivo . Sea 2^d la mayor potencia de 2 tal que divide $\varphi(p^\alpha)$. Entonces 2^d divide al orden r módulo p^α de un elemento aleatorio del grupo $\mathbb{Z}_{p\alpha}^\times$ con probabilidad de $\frac{1}{2}$.

Demostración. Puesto que p es impar, $\varphi(p^\alpha) = p^{\alpha-1}(p-1)$ es par, luego d es mayor o igual que 1. Puesto que p es impar y α es un entero positivo, entonces $\mathbb{Z}_{p^\alpha}^\times$ es cíclico y existe g elemento generador del grupo. Sea r el orden de g^k módulo p^α, $k = 1, \ldots, \varphi(p^\alpha)$. Se consideran entonces dos casos

- Si k es impar, entonces puesto que $g^{rk} = 1 \mod p^\alpha$, ocurre que $\varphi(p^\alpha)$ divide al producto de k por r, $\varphi(p^\alpha)|kr$. Entonces $2^d|r$.
- Si k es par. Entonces

$$\begin{aligned} & g^{k\varphi(p^\alpha)/2} \mod p^\alpha \\ = & (g^{\varphi(p^\alpha)})^{\frac{k}{2}} \mod p^\alpha \\ = & 1^{\frac{k}{2}} \mod p^\alpha \\ = & 1 \mod p^\alpha . \end{aligned}$$

  Luego $r|\frac{\varphi(p^\alpha)}{2}$ de donde se deduce que 2^d no divide a r, $2^d \nmid r$.

Tras esto, el grupo $\mathbb{Z}_{p\alpha}^\times$ queda particionado en dos subconjunto de igual cardinal. Luego con probabilidad $\frac{1}{2}$ 2^d divide al orden r de un elemento aleatorio del grupo $\mathbb{Z}_{p\alpha}^\times$, como se pretendía probar. □

Teorema 3.5.1. Sea n un número natural impar tal que se puede descomponer en factores primos como $n = p_1^{\alpha_1} \ldots p_m^{\alpha_m}$. Sea $a \in (2, n-2)$, con $mcd(a, n) = 1$ y sea r el orden del elemento a. Entonces, si se mide la probabilidad de que r sea par y $x^{r/2} \neq -1 \mod n$ es mayor o igual que $1 - \frac{1}{2^m}$.

Demostración. Lo que se pretende comprobar es que

$$p(\text{r par y } x^{r/2} \neq -1 \mod n) \geq 1 - \frac{1}{2^m} ,$$

lo que equivale a

$$p(\text{r impar o } x^{r/2} = -1 \mod n) \leq \frac{1}{2^m} .$$

Se va a probar esta última implicación. Haciendo uso del Teorema Chino de los Restos, la elección de $x \in \mathbb{Z}_n^\times$ equivale a escoger x_j independientes y uniformemente distribuidos pertenecientes a $\mathbb{Z}^\times_{p_j^{\alpha^j}}$ tal que

$$x = x_j \mod p_j^{\alpha^j} \quad j = 1, \ldots, m .$$

De esta forma, sea r_j el orden de cada elemento x_j correspondiente módulo $p_j^{\alpha^j}$. Además, sea 2^{d_j} la mayor potencia de 2 tal que divide al orden r_j para $j = 1, \ldots, m$, y sea de igual forma 2^d la mayor potencia de 2 que divide al orden r. Para que se cumpla una de las dos condiciones pedidas, es decir que o r es impar o en su defecto $x^{r/2} = -1 \mod n$, es necesario que todos los elementos $d_j \ \forall j$ posean el mismo valor. Por el Lema 3.5.1 la probabilidad de que esto ocurra es al menos $\frac{1}{2^m}$.

Falta comprobar que el valor que toman los d_j es exactamente d o 0. Se distinguen dos casos

- Si r es impar, entonces cada $r_j | r$, $r = 1, .., m$. De esta forma r_j es impar y $d_i = 0$, $i = 1, \ldots, s$.
- Si r es par y $x^{r/2} = -1 \mod n$, entonces $x^{r/2} = -1 \mod p_j^{\alpha^j} \quad j = 1, \ldots, m$. De esta forma $r_j \nmid \frac{r}{2}$, y puesto que $r_j | r$ se tiene $d_j = d$, $\forall j$.

□

Aclarar que n se establece impar puesto que se puede suponer sin pérdida de generalidad que n no es primo y por tanto existen factores no triviales. Esta suposición se realiza gracias a los test de primalidad existentes.

3.5.2. Algoritmo factorización, parte cuántica

En 1994 Peter Shor en un artículo del ACM [18] presenta un procedimiento capaz de capturar el periodo de una función. Aprovecha las características de la mecánica cuántica, como el principio de superposición o la transformada de Fourier discreta. Se realiza un procesamiento en paralelo del algoritmo, para posteriormente colapsar la función y medir. Tras esto, el resultado que arroja es probabilístico. La importancia de Shor radica en que comprueba, que a pesar de tener que realizar varias repeticiones de este procedimiento de medida para obtener una distribución, es posible realizarlo con una complejidad computacional significativamente menor que la obtenida empleando una metodología clásica.

Transformada cuántica de Fourier

La transformada cuántica de Fourier (QFT por sus siglas en inglés *Quantum Fourier Transformation*) es la implementación de la transformada de Fourier discreta en un circuito cuántico. Se basa en aplicar esta transformada discreta de Fourier sobre las 2^n amplitudes de una función de onda en un circuito cuántico. Con este propósito efectúa la descomposición como producto de matrices unitarias.

Definición 3.5.3. Sea $N = 2^n$ y $\omega = e^{2\pi i/N}$ una raíz N-ésima de la unidad. La matriz QFT es

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega_N^{jk} |k\rangle\langle j| =$$

$$= \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & w_N & w_N^2 & w_N^3 & \dots & w_N^{N-1} \\ 1 & w_N^2 & w_N^4 & w_N^6 & \dots & w_N^{2(N-1)} \\ 1 & w_N^3 & w_N^6 & w_N^9 & \dots & w_N^{3(N-1)} \\ : & : & : & : & \dots & : \\ 1 & w_N^{N-1} & w_N^{2(N-1)} & w_N^{3(N-1)} & \dots & w_N^{(N-1)(N-1)} \end{bmatrix}$$

y posee dimensión $N \times N$.

La transformada de Fourier es capaz de, dada una base computacional, obtener la base de Fourier, entendida esta como aquella en la que los estados se encuentran superpuestos.

Proposición 3.5.2. La matriz U_{QFT} es unitaria.

Demostración. Es necesario comprobar que ella misma por su transpuesta conjugada es la matriz identidad,

$$U_{QFT} \cdot U_{QFT}^* = U_{QFT}^* \cdot U_{QFT} = I \ .$$

Un elemento de la matriz producto $U_{QFT} \cdot U_{QFT}^*$ es de la forma genérica:

$$U_{QFT_{ij}} = 1/N \sum_{k=0}^{N-1} \omega^{jk} \omega^{-ik} \ .$$

Es necesario distinguir dos casos, en función de si los elementos pertenecen o no a la diagonal. Si $i = j$, entonces es un elemento diagonal, luego $\forall i \in 0, \dots, N-1$:

$$U_{QFT_{ii}} = \frac{1}{N} \sum_{k=0}^{N-1} \omega^{jk} \omega^{-jk} = \frac{1}{N} \sum_{k=0}^{N-1} 1 = \frac{1}{N} N = 1 \ .$$

Si el elemento no pertenece a la diagonal, entonces $i \neq j$ y

$$U_{QFT_{ij}} = \frac{1}{N} \sum_{k=0}^{N-1} \omega^{jk} \omega^{-ik} = \frac{1}{N} \sum_{k=0}^{N-1} \omega^{(j-i)^k}$$

siendo ω^{j-i} es una raíz N-ésima de la unidad, y denominando $\alpha = \omega^{j-i}$, luego

$$U_{QFT_{ij}} = \frac{1}{N} \sum_{k=0}^{N-1} \alpha^k = \frac{1-\alpha^k}{1-\alpha} = 0 \ .$$

De esta forma, la matriz resultante es la matriz diagonal como se pretendía probar. La demostración para el caso $U_{QFT}^* \cdot U_{QFT}$ es análoga. □

El circuito de n *qubits* que implementa la transformada cuántica de Fourier emplea dos puertas, una puerta de Hadamard aplicada a un *qubit* y una rotación controlada sobre dos *qubits*, es decir, un desplazamiento de fase. Además, añade la puerta SWAP empleada para invertir el estado de los *qubits*. La puerta de Hadamard actúa como

$$H|x_k\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + \exp\left(\frac{2\pi i}{2}x_k\right)|1\rangle\right)$$

para cualquier *qubit* $|x_k\rangle$, $k \in 0, \ldots, n$. Por otro lado, la rotación controlada es aplicada en dos *qubits* $|x_k x_j\rangle$ siendo el primero el *qubit* de control. Esto da como resultado:

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix} ,$$

es decir,

$$R_k|0x_j\rangle = |0x_j\rangle$$
$$R_k|1x_j\rangle = \exp\left(\frac{2\pi i}{2^k}x_j\right)|1x_j\rangle \ .$$

La puerta SWAP se conforma aplicando tres puertas C-NOT, produciendo el intercambio de estados entre los *qubits* implicados:

$$SWAP(|x_i, x_j\rangle) = |x_i \oplus (x_i \oplus x_j), (x_i \oplus x_j) \oplus x_j\rangle = |x_j, x_i\rangle \ .$$

El uso combinado de las puertas explicadas configuran el circuito buscado que se presenta en la Figura 3.3, obtenida de [12], para n *qubits*. Para mayor información sobre como se construye de manera precisa este circuito mostrado, se puede consultar [15].

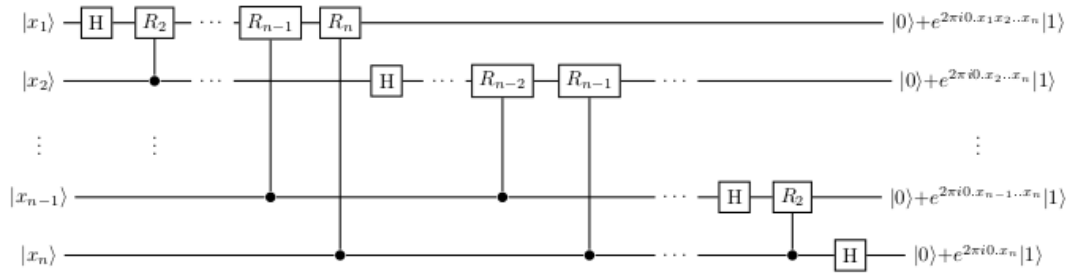

Figura 3.3: Circuito QFT

Ejemplo 3.5.1. Por evidenciar lo explicado para un circuito con tres *qubits*, la definición de la transformada cuántica de Fourier posee la forma:

$$U_{QFT} = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & w & w^2 & w^3 & w^4 & w^5 & w^6 & w^7 \\ 1 & w^2 & w^4 & w^6 & 1 & w^2 & w^4 & w^6 \\ 1 & w^3 & w^6 & w & w^4 & w^7 & w^2 & w^5 \\ 1 & w^4 & 1 & w^4 & 1 & w^4 & 1 & w^4 \\ 1 & w^5 & w^2 & w^7 & w^4 & w & w^6 & w^3 \\ 1 & w^6 & w^4 & w^2 & 1 & w^6 & w^4 & w^2 \\ 1 & w^7 & w^6 & w^5 & w^4 & w^3 & w^2 & w \end{bmatrix}$$

con $\omega = e^{2\pi i/8}$. El circuito se muestra a continuación (Figura 3.4). Las rotaciones controladas en este caso se pueden construir con puertas S que generan una rotación de fase de $\frac{\pi}{2}$, y puertas T que añaden una rotación de $\frac{\pi}{4}$.

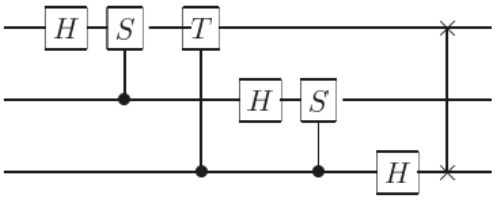

Figura 3.4: Circuito con 3 *qubits*

Para el cálculo de la transformada discreta de Fourier se empleaba anteriormente la transformada rápida de Fourier. La aparición de la transformada cuántica de Fourier consigue disminuir exponencialmente el tiempo de cómputo necesario. Para realizar la transformada cuántica de Fourier en un circuito de n *qubits* se comienza empleando una puerta de Hadamard, seguida de $n-1$ rotaciones controladas en el primer *qubit*. Esto supone un total de n puertas. El circuito continua con la aplicación de una nueva puerta de Hadamard y $n-2$ rotaciones condicionadas en el segundo *qubit*. Esto supone aplicar $n-1$ puertas. Continuando este proceso para los n *qubits*, se emplean un total de $n+(n-1)+\ldots+1 = \frac{n(n+1)}{2}$ puertas. Este cálculo es necesario completarlo con el empleo de las $\frac{n}{2}$ puertas SWAP, teniendo en cuenta que cada una de ellas hace uso de tres puertas C-NOT. Por lo descrito, la complejidad que posee la transformada cuántica de Fourier es $\mathcal{O}(n^2)$. Esto supone una importante ventaja frente a la complejidad $\mathcal{O}(n2^n)$ de la transformada rápida de Fourier.

Por último, cabe mencionar que la transformada cuántica de Fourier inversa se conforma con las operaciones en orden inverso y las rotaciones existentes realizándose en sentido contrario.

Estimación cuántica de la fase

La estimación cuántica de la fase es una herramienta empleada en numerosos algoritmos cuánticos. Este algoritmo supone la existencia de un operador unitario U junto con un autovector $|u\rangle$ y el autovalor correspondiente $e^{2\pi i\varphi}$. El objetivo que persigue es conseguir estimar φ. Para ello se emplean dos registros, el primero está formado por un número $t \in \mathbb{N}$ de *qubits* que se encuentran inicialmente en el estado $|0\rangle$. La correcta elección de t depende tanto del número de dígitos en la precisión con la que se desee estimar φ como de la probabilidad con la que se pretenda obtener la estimación de la fase. Por otra parte, el segundo registro parte del estado $|u\rangle$, luego necesitará tantos *qubits* como sean necesarios para representar $|u\rangle$.

El procedimiento general para la estimación de φ consiste en, partiendo del estado inicial $|0\rangle|u\rangle$, aplicar puertas de Hadamard para conseguir un estado superpuesto

$$\frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|u\rangle \ .$$

Seguidamente se aplican los operadores controlados U^j con $j = 0 \dots 2^{t-1}$

$$\frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle U^j |u\rangle = \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} e^{2\pi i j \varphi u} |j\rangle |u\rangle \ .$$

En último lugar, se aplica la transformada de Fourier inversa y se mide en el primer registro, obteniendo así la estimación de la fase φ buscada. El circuito de la Figura 3.5, obtenido de [12], muestra el proceso descrito.

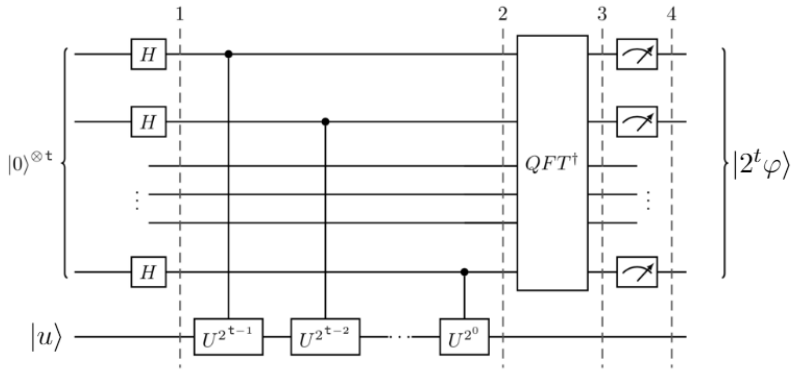

Figura 3.5: Estimación cuántica de la fase

Algoritmo de Shor

Retomando el problema planteado de la obtención del periodo dada una función periódica $f(x)$, el algoritmo de Shor ilustra un procedimiento que gana en eficiencia al algoritmo clásico. Puesto que un circuito cuántico posee un número finito de *qubits*, será posible representar únicamente números del 0 al $2^n - 1$. Para encontrar el periodo de la función se va a realizar la búsqueda sobre esos valores. Se denota por $\mathcal{A}$ el cardinal del conjunto de los elementos que se pueden representar. Tal como se ha definido, $\mathcal{A}$ es múltiplo del periodo. La esencia del algoritmo de Shor es aprovechar la ventaja de calcular una medición parcial. De forma que si se emplean por ejemplo dos *qubits*, midiendo únicamente en uno de ellos se puede obtener información adicional del otro *qubit*, puesto que éste no ha colapsado.

A continuación se analiza paso a paso. Sea $f(x)$ una función periódica, es decir $f(x + r) = f(x)$ para un $r \in (0, 2^n - 1)$. Para conseguir hallar r, se construye el circuito con t *qubits* en el primer registro y n *qubits* en el segundo registro. Se parte del estado inicial $|0\rangle|0\rangle$. Para conseguir un estado superpuesto, se aplican puertas de Hadamard en cada uno de los elementos del primer registro

$$\frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} |x\rangle |0\rangle \ ,$$

haciendo uso de los operadores definidos en la estimación cuántica de fase U, éstos actúan de la forma $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$. Luego, al aplicarlo al sistema que concierne se obtiene

$$\frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} |x\rangle |f(x)\rangle = \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{x=0}^{2^t-1} |x\rangle |\hat{f}(s)\rangle \ .$$

Posteriormente, es necesario aplicar la transformada de Fourier inversa en el primer registro

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\widehat{s/r}\rangle |\hat{f}(s)\rangle \ .$$

Siendo $\widehat{s/r}$ una estimación de la fase, donde s es escogido aleatoriamente. Finalmente, midiendo en el primer registro se obtiene $\widehat{s/r}$. Para obtener el periodo r, se hace uso de su expansión en fracciones continuas. Este proceso se muestra en la Figura 3.6.

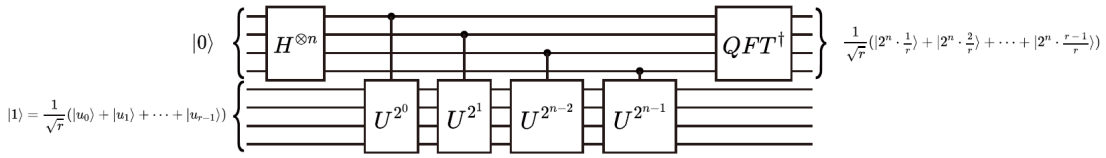

Figura 3.6: Algoritmo de Shor

La complejidad de este algoritmo es polinomial, $\mathcal{O}(n^2)$ con n el número de *qubits*. Lo que supera en eficiencia a cualquier algoritmo clásico de búsqueda del periodo de una función.

3.6. Curvas elípticas

Las curvas elípticas surgen como alternativa a los sistemas criptográficos basados en el problema del logaritmo discreto. El cuerpo finito empleado en este caso será el conjunto de puntos de una curva elíptica escogida. Esta sección muestra de una forma somera, sin entrar en detalles debido a no ser el objetivo de este trabajo, los fundamentos de la criptografía basada en curvas elípticas. Las demostraciones no realizadas se pueden consultar en [19].

3.6.1. Definiciones

Definición 3.6.1. Sea $\mathbb{K}$ un cuerpo con característica distinta de dos y de tres. Sea $a, b \in \mathbb{R}$, $x^3 + ax + b$ un polinomio cúbico que carece de raíces múltiples. Una *curva elíptica* sobre $\mathbb{K}$ es el conjunto de puntos $(x, y) \in \mathbb{K}$ que satisfacen la ecuación

$$y^2 = x^3 + ax + b \ ,$$

considerando a su vez el punto del infinito p_∞.

La ecuación $y^2 = x^3 + ax + b$ se denomina *forma normal de Weierstrass*.

Observaciones:

- Si el cuerpo $\mathbb{K}$ posee característica 2, la curva elíptica la conforma el conjunto de puntos que satisfacen la ecuación

$$y^2 + cy = x^3 + ax + b$$

o

$$y^2 + xy = x^3 + ax^2 + b$$

junto con el punto del infinito, $a, b \in \mathbb{R}$.

- Si el cuerpo $\mathbb{K}$ posee característica 3, la curva elíptica sobre $\mathbb{K}$ la conforma el conjunto de puntos que satisfacen la ecuación

$$y^2 = x^3 + ax^2 + bx + c$$

junto con el punto del infinito, $a, b \in \mathbb{R}$.

El requerimiento de que no tenga raíces múltiples, es equivalente a exigir que todos los puntos sobre la curva sean no singulares. Además, los puntos de una curva elíptica forman un grupo abeliano.

Teorema 3.6.1. Sea una curva elíptica $E : y^2 = x^3 + ax + b$ sobre un cuerpo $\mathbb{K}$. Entonces el conjunto

$$E(\mathbb{K}) = \{(x, y)|x, y \in \mathbb{K}, y^2 = x^3 + ax + b\} \cup \{p_\infty\}$$

posee estructura de grupo abeliano con la suma.

La operación se define como sigue: sean P y Q dos elementos de $E(\mathbb{K})$, para definir P+Q y el elemento inverso −P se distinguen los siguientes casos:

- Si P es el punto del infinito, $P = p_\infty$, se define $-P = p_\infty$ y P+Q=Q. Esto implica que el punto del infinito actúa como elemento neutro del grupo. Análogamente, si Q fuera el punto del infinito, P+Q=P.

- Si $P \neq p_\infty$, $P = (x, y)$ y $Q \neq p_\infty$. El elemento −P se define como $-P := -(x, y) = (x, -y)$. Se verifica que −P pertenece al conjunto $E(\mathbb{K})$ siempre que P pertenezca a él.

  Si $P \neq Q$, la recta $l = \overline{PQ}$ intersecta a la curva E exactamente en un único punto $R \in E(\mathbb{K})$. Si esta recta fuera tangente a P, entonces $R = P$ y si es tangente a Q, entonces $R = Q$. Se define $P + Q = -R$ (Figura 3.7).

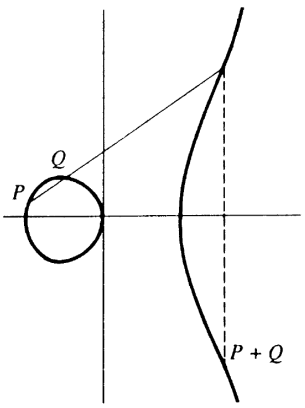

Figura 3.7: Ejemplo operación P+Q para cierta curva elíptica.
[Imagen perteneciente a [10]]

- Si $Q = -P$, $P \neq p_\infty$ y $Q \neq p_\infty$. Entonces la operación $P + Q = p_\infty$ da como resultado el punto del infinito.
- Si $Q = P$, $P \neq p_\infty$ y $Q \neq p_\infty$. Entonces la recta l es la recta tangente a P y se toma R como el otro punto de intersección con la curva E, definiendo de igual manera $P + Q = -R$.

En criptografía se trabaja con un cuerpo finito $\mathbb{F}_q$. Una curva elíptica posee a los sumo $2q + 10$ puntos, es decir el punto del infinito y $2q$ puntos de la forma $(x, y) \in \mathbb{F}_q$ Si se desea emplear en un sistema criptográfico $E(\mathbb{K})$, de forma que la seguridad del mismo radique en el problema del logaritmo discreto, es necesario que un elemento posea orden lo suficientemente grande. Para conocer la magnitud de una curva elíptica se emplea el Teorema de Hasse (Teorema 3.6.2).

Teorema 3.6.2. Teorema de Hasse. Si $E(\mathbb{K})$ es el conjunto de puntos de una curva elíptica E sobre un cuerpo $\mathbb{F}_q$, entonces

$$q + 1 - 2\sqrt{q} \leq |E(\mathbb{K})| \leq q + 1 + 2\sqrt{2}$$

La demostración se puede consultar en [20].

3.6.2. Criptografía de curvas elípticas

Puesto que la criptografía de curvas elípticas está basada en el problema del logaritmo discreto, tanto los sistemas de intercambio de claves, los sistemas de cifrado y la firma digital, los cuales su seguridad radica en este problema, poseen su versión elíptica. Este apartado muestra de forma breve y concisa como se enuncian estos sistemas en el contexto de las curvas elípticas.

Sea F_q un cuerpo finito, por lo expuesto anteriormente, el grupo $E(F_q)$ es cíclico o producto de dos grupos cíclicos. Además, gracias al teorema de Hasse es posible obtener en tiempo polinomial una buena acotación del cardinal de este grupo. Tal como se define la suma dentro del grupo $E(F_q)$, ésta está basada en la realización de una serie de operaciones elementales sobre el cuerpo F_q, luego es posible realizarlo en tiempo polinomial. Sumado a todo lo mencionado, el empleo de este grupo posee a su vez la ventaja de que plantear un ataque a un sistema basado en el problema del logaritmo discreto sobre él, posee mayor complejidad que para un sistema basado en el mismo problema sobre un cuerpo finito del mismo tamaño. La consecuencia inmediata es que se permite el uso de grupos de un tamaño significativamente menor para alcanzar el mismo nivel de seguridad. Además, permitirá el empleo de claves de menor longitud, disminuyendo los problemas de cómputo.

Definición 3.6.2. Sea F_q un cuerpo finito, E una curva elíptica definida sobre F_q, $P \in E$ un punto de la curva de orden n y $Q \in \langle P \rangle$. El *problema del logaritmo discreto para curvas elípticas* (ECDLP) consiste en encontrar el número entero $k \in [0, n-1]$ tal que

$$Q = kP$$

k se denomina logaritmo discreto de Q respecto a la base P,

$$k = \log_p Q$$

Intercambio de claves elíptico

Los dos usuarios que van a realizar el intercambio de claves acuerdan el orden del cuerpo finito q, los coeficientes a, b que definen la curva elíptica E sobre F_q, el punto $P \in E$ y el orden de este elemento n. De esta forma, el usuario A selecciona de forma aleatoria $d_A \in [1, n-1]$, calcula $Q_A = d_A P$ y por tanto obtiene el par de claves (Q_A, d_A). El usuario B realiza la misma operación, escogiendo aleatoriamente $d_B \in [1, n-1]$, calcula $Q_B = d_B P$ y obtiene el par (Q_B, d_B). Ambos usuarios intercambian sus claves Q_A y Q_B. Si ahora el usuario A calcula $d_A Q_B$ y el usuario B calcula $d_B Q_A$ ambos obtienen:

$$d_B Q_A = d_B(d_A P) = (d_A d_B)P = d_A(d_B P) = d_A Q_B \ .$$

La ventaja del intercambio de claves, es que posteriormente $(d_A d_B)P$ puede ser empleado, por ejemplificar un caso, como clave de un sistema de cifrado simétrico, sobre todo en el caso que ésta se comparta por un canal inseguro.

ElGamal elíptico

Puesto que se trata de un sistema asimétrico es necesaria la definición de las claves pública y privada de la comunicación. Sea F_q un cuerpo finito, la clave pública estará formada por el par (E, P), donde E es una curva elíptica y $P = (x_P, y_P) \in E$ es un punto perteneciente a la misma. Si $E(F_q)$ es cíclico, entonces sería conveniente pero no necesario que el punto P fuera un elemento generador. Para salvaguardar la seguridad del sistema es conveniente que $\langle P \rangle$ sea lo suficientemente grande.

Concretando detalles, cada usuario del sistema elige un número d_i como clave secreta y muestra de forma pública el producto $d_i P$, su clave pública, $\forall i$. Si el usuario i desea enviar un mensaje P_m al usuario j, elegirá un $k \in \mathbb{Z}$ y enviará el par $(kP, P_m + k(d_j P))$. El receptor j para recuperar el mensaje original calcula $kd_j P$ empleando el valor kP y su clave secreta.

3.6.3. Amenaza al logaritmo discreto para curvas elípticas

Dependiendo de la elección de la curva elíptica, en algunos casos el problema del logaritmo discreto para curvas elípticas puede ser vulnerado de forma clásica. Entre los ataques existentes se encuentra el algoritmo de Pohlig-Hellman y el algoritmo rho de Pollard. Aunque la mejor solución lo arroja una combinación de ambos, reduciendo el problema a complejidad exponencial.

Dentro de la computación cuántica es posible romper el problema del logaritmo discreto mediante el uso de la transformada cuántica de Fourier. Sea $G = \mathbb{Z}^*/(n)$ un grupo cíclico multiplicativo de orden n. Generalmente n es un número primo. Sea g el generador del grupo G. El logaritmo discreto del elemento $x \in G$ es el entero $0 \leq a < n-1$ tal que

$$g^a \equiv x \mod n \ .$$

Para calcularlo, se define la función

$$f(\alpha, \beta) = \frac{g^\alpha}{x^\beta} \mod n = g^{\alpha - a\beta} \mod n$$

que es periódica con periodo $\tau = (a, 1)$

$$f(\alpha + a, \beta + 1) = g^{\alpha + a - a\beta - a} \mod n = g^{\alpha - a\beta} \cdot g^0 \mod n = f(\alpha, \beta) \ .$$

De esta forma, la transformada cuántica de Fourier es capaz de calcular el periodo τ y en consecuencia el logaritmo discreto a. La descripción detallada del procedimiento se encuentra en [18]. De hecho, en este artículo se cita el trabajo de Boneh y Lipton en esta área [5], que generaliza el método anterior para cualquier grupo abeliano que contenga un subgrupo cíclico, en particular el grupo multiplicativo de cualquier cuerpo finito $\mathbb{F}_q$. Si el cuerpo finito no es primo $\mathbb{F}_q$, la función empleada en la transformada cuántica de Fourier toma la forma

$$f(\alpha, \beta) = \frac{g^\alpha}{x^\beta} = g^{\alpha - a\beta}$$

siendo g un elemento primitivo del cuerpo. Por otro lado, en el caso de una curva elíptica, sea P un punto fijado previamente, entonces la función es

$$f(\alpha, \beta) = \alpha P - \beta Q = (\alpha - a\beta)P$$

donde $Q = aP$, es decir, a es el "logaritmo discreto" del punto Q.

Para poder aplicar la transformada cuántica de Fourier es indispensable, en cualquiera de los casos, que se puedan multiplicar y calcular inversos de los elementos en tiempo polinomial. Estas operaciones en ambos casos se realizan de forma eficiente. Por un lado, la multiplicación en cuerpos finitos está basada en la exponenciación modular. La operación $a^m \ mod \ n$, $a, a^m \in \mathbb{F}_q$, $m \in \mathbb{N}$, se realiza de forma eficiente considerando la expresión binaria de m:

$$m = s_0 + 2s_1 + \ldots + 2^{i-1}s_{i-1} \ .$$

El procedimiento de calculo es iterativo, mostrando el pseudocódigo en los Algoritmos 1 y 2.

Algoritmo 1: Exponenciación modular para cuerpos finitos

```
Data: a, m, n
Result: a^m ≡ b mod n
b = 1
for i = k − 1 hasta 0 do
    if s_i = 1 then
        b = b · a mod n
    if i > 0 then
        b = b^2 mod n
```

Para realizar la multiplicación de manera eficiente de un punto $P \in E$ por un entero m dentro de una curva elíptica E se trabaja con la expresión binaria del entero m, reduciendo el cálculo mP a realizar multiplicaciones por dos y sumar puntos una cantidad $\log_2(m)$. Es decir, $m = s_0 + 2s_1 + \ldots + 2^{i-1}s_{i-1}$.

Algoritmo 2: Exponenciación para curvas elípticas

```
Data: a, m, n
Result: b = mP
b = 1
for i = 0 hasta k − 1 do
    if s_i = 1 then
        b = b · 2 + P
    if s_i = 0 then
        b = b · 2
```

Por otra parte, para el cálculo de inversos dentro de un cuerpo finito se emplea el teorema de Bézout, tanto para el caso modular como para polinomios para extensiones de cuerpos. El cálculo de inversos en curvas elípticas ya ha sido explicado en la correspondiente sección.

Capítulo 4

Retículos

La criptografía basada en retículos constituye un extenso área dentro de la criptografía postcuántica. Dentro del ámbito de los retículos se encuentran los llamados problemas difíciles. Los algoritmos principales poseen complejidad exponencial, y los que son resolubles en tiempo polinomial arrojan soluciones pobres y alejadas de una buena aproximación. Hasta la fecha, no existe ningún algoritmo cuántico sólido capaz de resolver alguno de estos problemas. El método de encontrar la periodicidad de una función que propone el algoritmo de Shor no es aplicable en este contexto. Esta es la razón por la cual los problemas difíciles en retículos se postulan como candidatos idóneos sobre los cuales fundamentar sistemas criptográficos.

Lo expuesto en este capítulo abarca los aspectos teóricos sobre la teoría de retículos y los problemas difíciles asociados a ellos. El siguiente capítulo aborda un sistema criptográfico basado en estos problemas. Por otra parte, es necesario establecer que los vectores que aparecen en las secciones sucesivas son vectores columna.

4.1. Definiciones

Definición 4.1.1. Un *retículo $\mathcal{L}$ de dimensión n* es cualquier subconjunto contenido en $\mathbb{R}^n$ que es:

- Un subgrupo con la operación suma. Esto implica que el elemento neutro pertenezca al retículo, $0 \in \mathcal{L}$. Si x es un elemento del retículo, entonces $-x$ también pertenece, $-x \in \mathcal{L}\ \forall x \in \mathcal{L}$. Si $x, y \in \mathcal{L}$, entonces se cumple que la suma pertenece al retículo, $x + y \in \mathcal{L}\ \forall x, y \in \mathcal{L}$.

- Un subgrupo discreto, lo que deriva en que todo elemento x del retículo $\mathcal{L}$ posee un entorno en $\mathbb{R}^n$ en el cual x es el único elemento que pertenece a $\mathcal{L}$.

Ejemplo 4.1.1.

- El conjunto de los enteros $\mathbb{Z}^n$ conforma un retículo.

- Es un retículo el conjunto $c\mathcal{L}$, siendo c un número real.

- El conjunto $\{x : x \in \mathbb{Z}^n, \sum_i x_i \text{ es par}\}$ conforma un retículo.

Las comprobaciones de que los ejemplos anteriores constituyen un retículo son inmediatas.

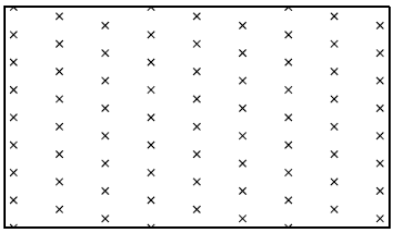

Figura 4.1: Ejemplo de retículo en $\mathbb{R}^2$

Definición 4.1.2. Sea $\mathcal{L}$ un retículo de dimensión n. Se denomina *conjunto generador de un retículo* $\mathcal{L}$ a un conjunto ordenado de vectores $\mathcal{G} = (\mathbf{b_1}, \ldots, \mathbf{b_m})$, $\mathbf{b_i} \subset \mathbb{R}^n$ tal que

$$\mathcal{L} = \mathcal{L}(\mathcal{G}) = \mathcal{G}\mathbb{Z}^m = \left\{ \sum_{i=1}^{m} a_i \mathbf{b_i} : a_i \in \mathbb{Z} \right\} .$$

m se denomina *rango del retículo*. Cuando $m = n$ se dice que el retículo posee rango completo.

Definición 4.1.3. Sea $\mathcal{L}$ un retículo de dimensión n. $\mathcal{G}$ un conjunto generador del retículo $\mathcal{L}$. Se denomina *base de un retículo* si este conjunto es minimal, es decir

$$\mathcal{L} = \mathcal{L}(\mathcal{B}) = \mathcal{B}\mathbb{Z}^m = \left\{ \sum_{i=1}^{n} a_i \mathbf{b_i} : a_i \in \mathbb{Z} \right\} .$$

Puede ser representada a su vez de forma matricial $\mathcal{B} = [\mathbf{b_1}, \ldots, \mathbf{b_n}] \in \mathcal{M}(\mathbb{R})_{n \times n}$.

Definición 4.1.4. Se define *subespacio generado* como el conjunto de todas las combinaciones lineales de un conjunto de vectores $\mathcal{B} = (\mathbf{b_1}, \ldots, \mathbf{b_n})$.

$$span(\{\mathbf{b_1}, \ldots, \mathbf{b_n}\}) = \{B\mathbf{y} : \mathbf{y} \in \mathbb{R}\} .$$

De esta forma, en el ámbito de los retículos, $span(\mathcal{L}(\mathcal{B}))$ es el espacio lineal generado por los vectores de la base dada. Por otra parte, para un retículo $\mathcal{L}$, escogiendo una base $\mathcal{B}$, es usual emplear como dominio fundamental aquel centrado en el origen y denominado paralelepípedo fundamental.

Definición 4.1.5. El *paralelepípedo fundamental* de una base dada $\mathcal{B}$, es el conjunto

$$P(\mathcal{B}) = \mathcal{B}\left[-\frac{1}{2}, \frac{1}{2}\right)^n = \left\{ \sum_{i=1}^{n} c_i \mathbf{b_i} : -\frac{1}{2} \leq c_i < \frac{1}{2} \right\} .$$

Definición 4.1.6. Sea $\mathcal{L}$ un retículo. Su *retículo dual* $\mathcal{L}^* \subset \mathbb{R}^n$ se define como

$$\mathcal{L}^* = \{\mathbf{w} : \langle \mathbf{w}, \mathcal{L} \rangle \subset \mathbb{Z}\} .$$

Entendiendo por $\langle , \rangle$ el producto interno. De esta forma el espacio dual de un retículo dado está formado por el conjunto de puntos cuyo producto interno con los vectores presentes en el retículo $\mathcal{L}$ sea entero. Es inmediata la comprobación de que $\mathcal{L}^*$ es a su vez un retículo. Además si $\mathcal{B}$ es una base de $\mathcal{L}$, $\mathcal{B}^{-t}$ lo es de su dual $\mathcal{L}^*$. La siguiente proposición está encaminada a arrojar información sobre si un conjunto de vectores forman una base de un retículo.

Proposición 4.1.1. Sea $\mathcal{L}$ un retículo de dimensión n. Sean $\mathbf{b_1}, \ldots, \mathbf{b_n} \in \mathcal{L}$ un conjunto de n vectores linealmente independientes pertenecientes al retículo. Entonces, $\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}$ constituyen una base de $\mathcal{L}$ si y solo si

$$P(\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}) \cap \mathcal{L} = 0 \ .$$

Demostración. Se asume primeramente que los vectores $\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}$ forman una base $\mathcal{B}$ del retículo $\mathcal{L}$. Por definición, el retículo $\mathcal{L}$ lo conforma el conjunto de todas las posibles combinaciones de números enteros. El paralelepípedo fundamental $P(\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n})$ está formado por el conjunto de todas las combinaciones lineales de los vectores $\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}$ con coeficientes en $[0, 1)$. Luego la intersección de ambos conjuntos es el conjunto formado por el elemento cero, $\{0\}$, $P(\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}) \cap \mathcal{L} = 0$

Recíprocamente, se supone que se cumple que $P(\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}) \cap \mathcal{L} = 0$. Puesto que el retículo posee dimensión n y los vectores son linealmente independientes, se puede expresar cualquier vector perteneciente al retículo de la forma:

$$\mathbf{x} = \sum_i a_i \mathbf{b_i} \quad \text{para ciertos} \quad a_i \in \mathbb{R} \ .$$

Puesto que un retículo es un subgrupo aditivo, el vector

$$\mathbf{x}' = \sum_i (a_i - \lfloor a_i \rfloor) \mathbf{b_i}$$

también pertenece al retículo $\mathcal{L}$, siendo $\lfloor a_i \rfloor$ la parte entera de a_i, $\forall i$. La hipótesis inicial implica que $\mathbf{x}' = 0$, luego todos los valores a_i son enteros, y en consecuencia $\mathbf{x}$ es una combinación entera de los vectores $\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}$. □

Los siguientes resultados arrojan información sobre cuando dos bases dadas son equivalentes, es decir generan el mismo retículo.

Definición 4.1.7. Una matriz $A \in \mathcal{M}(\mathbb{Z})_{n \times n}$ se dice que es *unimodular* si

$$det(A) = \pm 1$$

Esta definición establece que si una matriz $A \in \mathcal{M}(\mathbb{Z})_{n \times n}$ es unimodular, es porque constituye un elemento del grupo lineal especial *SL(n,ℤ)*. Cabe destacar y es sencilla su comprobación que si A es unimodular, también lo es su inversa $A^{-1} \in \mathcal{M}(\mathbb{Z})_{n \times n}$.

Proposición 4.1.2. Dadas dos bases $\mathcal{B}_1$ y $\mathcal{B}_2$ pertenecientes a $\mathbb{R}$, son equivalentes si y solo si existe una matriz unimodular U tal que $\mathcal{B}_2 = \mathcal{B}_1 U$.

Demostración. Se supone en primer lugar que ambas bases son equivalentes, es decir $\mathcal{L}(\mathcal{B}_1) = \mathcal{L}(\mathcal{B}_2)$. Luego cada uno de los vectores pertenecientes a $\mathcal{B}_2$, pertenecen a su vez al retículo generado por $\mathcal{B}_1$,

$$\mathbf{b_i} \in \mathcal{B}_2 \implies \mathbf{b_i} \in \mathcal{L}(\mathcal{B}_2) \implies \mathbf{b_i} \in \mathcal{L}(\mathcal{B}_1) \ \ \forall i \ .$$

De esta forma, existe por tanto una matriz $U \in \mathcal{M}(\mathbb{Z})_{n\times n}$ de forma que $\mathcal{B}_2 = \mathcal{B}_1 U$. Análogamente existe una matriz $V \in \mathcal{M}(\mathbb{Z})_{n\times n}$ tal que $\mathcal{B}_1 = \mathcal{B}_2 V$. Se obtiene por tanto la cadena de igualdades

$$\mathcal{B}_2 = \mathcal{B}_1 U = \mathcal{B}_2 V U$$

$$\mathcal{B}_2^T \mathcal{B}_2 = (VU)^T \mathcal{B}_2^T \mathcal{B}_2 (VU) \ ,$$

tras aplicar determinantes a ambos lados

$$det(\mathcal{B}_2^T \mathcal{B}_2) = det((VU)^T \mathcal{B}_2^T \mathcal{B}_2 (VU))$$

$$det(\mathcal{B}_2^T \mathcal{B}_2) = det((VU))^2 det(\mathcal{B}_2^T \mathcal{B}_2) \ .$$

Entonces de aquí se obtiene que $det(V)det(U) = \pm 1$, en consecuencia esto implica que $det(U) = \pm 1$, la matriz U es unimodular como se pretendía comprobar. Recíprocamente, si $\mathcal{B}_2 = \mathcal{B}_1 U$, siendo $U \in \mathcal{M}(\mathbb{Z})_{n\times n}$ una matriz unimodular, entonces cada uno de los vectores $\mathbf{b}_i$ que conforman la base $\mathcal{B}_2$ están contenidos en el retículo $\mathcal{L}(\mathcal{B}_1)$. Luego

$$\mathcal{L}(\mathcal{B}_2) \subset \mathcal{L}(\mathcal{B}_1) \ .$$

Además, $\mathcal{B}_1 = \mathcal{B}_2 U^{-1}$, y por lo visto anteriormente, la matriz U^{-1} es a su vez unimodular por serlo U. Esto permite obtener la contención contraria

$$\mathcal{L}(\mathcal{B}_1) \subset \mathcal{L}(\mathcal{B}_2)$$

y en consecuencia la equivalencia como se pretendía probar

$$\mathcal{L}(\mathcal{B}_2) = \mathcal{L}(\mathcal{B}_1) \ .$$

□

Esta proposición arroja un corolario inmediato:

Corolario 4.1.1. $\mathcal{B}$ es una base de $\mathbb{Z}^n$ si la matriz que conforman sus vectores es unimodular.

Definición 4.1.8. Sea $\mathcal{B}$ una base cualquiera del retículo $\mathcal{L}$, sea $\mathcal{L} = \mathcal{L}(\mathcal{B})$ un retículo de dimensión n. El *determinante* de $\mathcal{L}$ se define como

$$det(\mathcal{L}) = \sqrt{det(\mathcal{B}^T \mathcal{B})} \ .$$

En el caso en el cual el retículo posea rango completo, la matriz que conforman los vectores pertenecientes a la base $\mathcal{B}$ es una matriz cuadrada y por tanto el determinante es

$$det(\mathcal{L}) = |det(\mathcal{B})| \ .$$

Además, el determinante de un retículo es equivalente al volumen n dimensional de su paralelepípedo fundamental $P(\mathcal{B})$.

Proposición 4.1.3. El valor absoluto del determinante es independiente de los vectores de la base escogidos.

Demostración. La demostración es inmediata de la Proposición 4.1.2. Sean $\mathcal{B}_1$ y $\mathcal{B}_2$ dos bases equivalentes del retículo $\mathcal{L}$. Luego por la proposición 4.1.2 existe una matriz unimodular $U \in SL(n, \mathbb{Z})$ tal que $\mathcal{B}_2 = \mathcal{B}_1 U$. Luego $det(\mathcal{B}_2) = det(\mathcal{B}_1)det(U) = det(\mathcal{B}_1)$. En consecuencia el valor absoluto del determinante no depende de la base, como se pretendía demostrar. □

Este desarrollo permite concluir que el determinante de un retículo es inversamente proporcional a su densidad. Es decir, cuanto menor sea el determinante, mayor es la concentración de puntos en el retículo. El cálculo de este determinante se puede efectuar de forma eficiente, haciendo uso de la base de Gram-Schmidt. Se aprovecha la ventaja de que ésta puede ser calculada en tiempo polinomial dada una base cualquiera de un retículo.

Definición 4.1.9. Sean $\mathbf{b_1}, \dots \mathbf{b_n}$ un conjunto de n vectores linealmente independientes. El proceso de *ortogonalización de Gram-Schmidt* es el conjunto de vectores $\mathbf{b_1^*}, \dots \mathbf{b_n^*}$ definidos como

$$\mathbf{b_i^*} = \mathbf{b_i} - \sum_{j=1}^{i-1} \mu_{ij} \mathbf{b_j^*}, \text{ donde } \mu_{ij} = \frac{\langle \mathbf{b_i}, \mathbf{b_j^*} \rangle}{\langle \mathbf{b_j^*}, \mathbf{b_j^*} \rangle}$$

siendo $\mathbf{b_i^*}$ la componente de $\mathbf{b_i}$ ortogonal a $\mathbf{b_1^*}, \dots, \mathbf{b_{i-1}^*}$.

Proposición 4.1.4. Sea $\mathcal{L}$ un retículo de dimensión n, sea $\mathcal{B}$ una base del retículo. Entonces

$$det(\mathcal{L}) = \prod_{i=0}^{n} \mathbf{b_i^*}$$

donde $B^* = (\mathbf{b_1^*}, \dots, \mathbf{b_n^*})$ es la base de Gram-Schmidt.

La demostración se puede encontrar en [14]

4.2. Mínimos sucesivos

Esta sección está estrechamente relacionada con la Sección 4.3 posterior, presenta resultados encaminados a la obtención de una cota para la norma del vector más corto no nulo. Algunos problemas difíciles están basados en la dificultad de hallar el vector con norma mínima.

Definición 4.2.1. Sea $\mathcal{L}$ un retículo. Se define la *distancia mínima* de este retículo como

$$\lambda_1(\mathcal{L}) = \min_{\mathbf{v} \in \mathcal{L} \setminus \{0\}} ||\mathbf{v}|| = \min_{\mathbf{x}, \mathbf{y} \in \mathcal{L}, \mathbf{x} \neq \mathbf{y}} ||\mathbf{x} - \mathbf{y}|| \ .$$

Tal y como se ha definido, λ_1 es el menor r para el cual los puntos del retículo dentro de la bola de centro cero y radio uno generan un espacio de dimensión uno. De esta forma, se pueden ir definiendo los mínimos sucesivos de los vectores de un retículo dado.

Definición 4.2.2. Sea $\mathcal{L}$ un retículo de rango n. Se definen los *mínimos sucesivos* de la forma

$$\lambda_i(\mathcal{L}) = \text{ínf} \left\{ r : \dim(span(\mathcal{L} \cap \overline{B}(0, r))) \geq i \right\} \ .$$

$\overline{B}(0,r)$ es la bola cerrada centrada en cero y de radio r. Los mínimos sucesivos son el menor radio r tal que la intersección entre la bola centrada en cero y radio r y el retículo, contenga i vectores linealmente independientes. Con esta definición, se obtiene la siguiente cadena de desigualdades

$$\lambda_1(\mathcal{L}) \leq \lambda_2(\mathcal{L}) \leq \ldots \leq \lambda_n(\mathcal{L}) \ .$$

Los siguientes resultados arrojan información sobre las cotas superiores e inferiores de los mínimos sucesivos, especialmente de λ_1, que permitirá esclarecer la obtención la norma del vector no nulo más corto para un retículo dado.

4.2.1. Cota inferior

Se demuestra que cualquier retículo contiene vectores no nulos de longitud mínima. Este resultado será generalizable a los mínimos sucesivos.

Teorema 4.2.1. Sea $\mathcal{L}$ un retículo, $\mathcal{B}$ una base de ese retículo, $\mathcal{B}^*$ la base que se forma tras aplicar la ortogonalización de Gram-Schmidt. Entonces

$$\lambda_1(\mathcal{L}(\mathcal{B})) \geq \min_{i=1,\ldots,n} ||\mathbf{b_i}^*|| > 0 \ .$$

Demostración. Sea $\mathbf{x} \in \mathbb{Z}^n$, $\mathbf{x} \neq 0$. Sea j el mayor entero tal que $x_j \neq 0$, $j \in \{1,\ldots,n\}$. Por lo tanto, puesto que para $i < j$, $\langle \mathbf{b_i}, \mathbf{b_j}^* \rangle = 0$ y además $\langle \mathbf{b_j}, \mathbf{b_j}^* \rangle = \langle \mathbf{b_j}^*, \mathbf{b_j}^* \rangle$ por ser una base de Gram-Schmidt. Además, para todo $i > j$, $x_i = 0$. Tras estas consideraciones, se obtiene

$$|\langle B\mathbf{x}, \mathbf{b_j}^* \rangle| = |\langle \sum_{i=1}^{j} x_i \mathbf{b_i}, \mathbf{b_j}^* \rangle| = |\sum_{i=1}^{j} x_i \langle \mathbf{b_i}, \mathbf{b_j}^* \rangle| = |x_j| \langle \mathbf{b_j}^*, \mathbf{b_j}^* \rangle = |x_j| \ ||\mathbf{b_j}^*||^2 \ .$$

Teniendo en cuenta la desigualdad de Cauchy-Schwarz $|\langle B\mathbf{x}, \mathbf{b_j}^* \rangle| \leq ||B\mathbf{x}|| \ ||\mathbf{b_j}^*||$, se obtiene

$$|\langle B\mathbf{x}, \mathbf{b_j}^* \rangle| = |x_j| \ ||\mathbf{b_j}^*||^2 \leq ||B\mathbf{x}|| \ ||\mathbf{b_j}^*|| \iff$$
$$||B\mathbf{x}|| \geq |x_j| \ ||\mathbf{b_j}^*|| \geq ||\mathbf{b_j}^*|| \geq \min_{i \in \{1,\ldots,n\}} ||\mathbf{b_i}^*|| \ .$$

De esta forma, puesto que se ha probado que $||B\mathbf{x}|| \geq \min ||\mathbf{b_i}^*||$ queda demostrado el teorema, ya que es equivalente a lo que se pretendía probar. □

Corolario 4.2.1. Sea $\mathcal{L}$ un retículo. Entonces existe un $\varepsilon > 0$ tal que la norma de la diferencia entre dos puntos del retículo es mayor que épsilon

$$||\mathbf{x} - \mathbf{y}|| > \varepsilon \qquad \mathbf{x}, \mathbf{y} \in \mathcal{L}, \ \ \mathbf{x} \neq \mathbf{y} \ .$$

Demostración. Puesto que $\mathbf{x}, \mathbf{y} \in \mathcal{L}$, $\mathbf{x} \neq \mathbf{y}$, aplicando el teorema 4.2.1,

$$||\mathbf{x} - \mathbf{y}|| \geq \lambda_1(\mathcal{L}) > 0 \ .$$

□

Corolario 4.2.2. Sea $\mathcal{L}$ un retículo. Entonces existe para todo $i \in [1,n]$ un punto del retículo $\mathbf{x_i} \in \mathcal{L}$ con norma

$$||\mathbf{x_i}|| = \lambda_i(\mathcal{L}) \ .$$

Demostración. Aplicando el Corolario 4.2.1, la bola $B(0, 2\lambda_i(\mathcal{L}))$ contiene un conjunto finito de puntos del retículo. Luego, haciendo uso de la definición de mínimo sucesivo, alguno de esos vectores poseerá norma igual a $\lambda_i(\mathcal{L})$, como se pretendía probar. □

4.2.2. Cota superior

Esta sección pretende refinar la cota superior de $\lambda_1(\mathcal{L})$, y que a su vez no dependa de la base escogida. Una cota trivial lo constituye

$$\lambda_1(\mathcal{L}) \geq \min_i ||\mathbf{b_i}||$$

siendo $\mathbf{b_i}$ un vector no nulo de una base dada del retículo $\mathcal{L}$. Los teoremas que se presentan son en su mayor parte debidos a Minkowski. En la exposición posterior se emplea la norma ℓ_2, pero generalizarlo a otras normas no supone mayor problema.

Teorema 4.2.2. *Blichfeld.* Sea $\mathcal{L}$ un retículo contenido en $\mathbb{R}^n$. Entonces, para cualquier conjunto medible $A \subset \mathbb{R}^n$ con volumen $vol(A) > det(\mathcal{L})$ existen dos puntos $a_1, a_2 \in A$, $a_1 \neq a_2$ tal que $a_1 - a_2 \in \mathcal{L}$.

Demostración. Se toma una base cualquiera $\mathcal{B}$ del retículo. Se escoge $\mathbf{x} \in \mathcal{L}$, de esta forma se definen los conjuntos

$$\mathbf{x} + P(\mathcal{B}) = \{\mathbf{x} + \mathbf{y} : \mathbf{y} \in P(\mathcal{B})\}$$

que conforman una partición del espacio $\mathbb{R}^n$. Se define a su vez para cada punto $\mathbf{x} \in \mathcal{L}$

$$A_{\mathbf{x}} = A \cap (\mathbf{x} + P(\mathcal{B})) \ .$$

Es importante notar que tal como están definidos los conjuntos $A_{\mathbf{x}}$, el conjunto A queda expresado como unión de ellos $A = \bigcup_{\mathbf{x}\in\mathcal{L}} A_{\mathbf{x}}$. Luego

$$vol(A) = \sum_{\mathbf{x}\in\mathcal{L}} vol(A_{\mathbf{x}}) \ .$$

Ahora se definen los conjuntos trasladados $A'_{\mathbf{x}} = A_{\mathbf{x}} - \mathbf{x}$, de forma que $A'_{\mathbf{x}} \subset P(\mathcal{B})$ y $vol(A'_{\mathbf{x}}) = vol(A_{\mathbf{x}})$. Se obtiene

$$\sum_{\mathbf{x}\in\mathcal{L}} vol(A'_{\mathbf{x}}) = \sum_{\mathbf{x}\in\mathcal{L}} vol(A_{\mathbf{x}}) = vol(A) > vol(P(\mathcal{B})) \ .$$

Por lo tanto, existen dos vectores del retículo, $\mathbf{a_1}, \mathbf{a_2} \in \mathcal{L}$, $\mathbf{a_1} \neq \mathbf{a_2}$ para los cuales $A'_{\mathbf{x}} \cup A'_{\mathbf{y}} \neq \emptyset$. Luego existe un vector a perteneciente a este conjunto intersección $\mathbf{a} \in A'_{\mathbf{x}} \cup A'_{\mathbf{y}}$. De esta forma

$$\mathbf{a} + \mathbf{a_1} \in A_{\mathbf{a_1}} \subset A$$

$$\mathbf{a} + \mathbf{a_2} \in A_{\mathbf{a_2}} \subset A$$

$$(\mathbf{a} + \mathbf{a_1}) + (\mathbf{a} + \mathbf{a_2}) = \mathbf{a_1} - \mathbf{a_2} \in \mathcal{L} \ ,$$

como se pretendía demostrar. □

Teorema 4.2.3. *Teorema del cuerpo convexo de Minkowski.* Sea $\mathcal{L}$ un retículo de dimensión n, sea A un subconjunto acotado, simétrico con respecto al origen y convexo. Si el volumen del conjunto $vol(A) > 2^n det(\mathcal{L})$ entonces A contiene al menos un vector no nulo perteneciente al retículo $\mathcal{L}$.

Demostración. Se parte de conjunto A acotado, simétrico y convexo. Se define el conjunto

$$A' = \frac{1}{2}A = \{\mathbf{x} : 2\mathbf{x} \in A\} \ .$$

Luego el volumen del conjunto A' es

$$vol(A') = 2^{-n} vol(A) > det(\mathcal{L}) \ .$$

Por el Teorema de Blichfeld 4.2.2, existen dos vectores $\mathbf{a_1}$ y $\mathbf{a_2}$, $\mathbf{a_1} \neq \mathbf{a_2}$, pertenecientes al conjunto A' tal que su diferencia pertenece al retículo $\mathbf{a_1} - \mathbf{a_2} \in \mathcal{L}$. Si este vector perteneciera a su vez a A, el teorema quedaría probado. Por lo tanto por como se ha definido A', $2\mathbf{a_1}$ y $2\mathbf{a_2}$ pertenecen al conjunto A.

$$\frac{2\mathbf{a_1} - 2\mathbf{a_2}}{2} = \mathbf{a_1} - \mathbf{a_2}$$

pertenece entonces al conjunto A como se quería probar, puesto que por ser simétrico con respecto al origen $-2\mathbf{a_2} \in A$ y por ser convexo $\frac{2\mathbf{a_1} - 2\mathbf{a_2}}{2} \in A$. □

Corolario 4.2.3. El volumen de una bola de centro cero y radio r $B(0, r) \in \mathbb{R}^n$ es

$$vol(B(0, r)) \geq \left(\frac{2r}{\sqrt{n}}\right)^n \ .$$

Demostración. Esta prueba se basa en el hecho de que la bola de centro cero y radio r contiene un cubo cuya longitud de cada lado es $\frac{2r}{\sqrt{n}}$, es decir

$$\left\{\mathbf{x} : \mathbf{x} \in \mathbb{R}^n, |x_i| < \frac{r}{\sqrt{n}}\right\} \subset B(0, r) \ .$$

□

Teorema 4.2.4. *Primer teorema de Minkowski.* Para cualquier retículo $\mathcal{L}$ de dimensión n se verifica

$$\lambda_1(\mathcal{L}) \leq \sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}} \ .$$

Demostración. Haciendo uso del Corolario 4.2.3 precedente puesto que se verifican las hipótesis, es decir el cuerpo A posee un volumen estrictamente mayor a $2^n det(\mathcal{L})$ debido a que contiene un hipercubo de n dimensiones de lado igual a $2det(\mathcal{L})^{\frac{1}{n}}$. Por el Teorema del cuerpo convexo de Minkowski 4.2.3 existe un vector no nulo perteneciente al retículo, tal que también pertenece al conjunto A, es decir, $\mathbf{a} \in \mathcal{L}$, $\mathbf{a} \in A$, $||\mathbf{a}|| < \sqrt{n}det(\mathcal{L})^{\frac{1}{n}}$. Luego se obtiene

$$\left(\frac{2\lambda_1(\mathcal{L})}{\sqrt{n}}\right)^n \leq vol(B(0, \lambda_1(\mathcal{L}))) \leq 2^n det(\mathcal{L})$$

y por tanto la longitud del vector no nulo más corto verifica

$$\lambda_1(\mathcal{L}) \leq \sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}} \ .$$

□

Cabe destacar como se ha mencionado anteriormente, que esta cota no depende de la base $\mathcal{B}$ del retículo escogida. Por otra parte, la cota se modifica de forma lineal tras una transformación lineal de retículo, es decir si se escoge el retículo $k\mathcal{L}$, con $k \in \mathbb{R}$, ocurre que

$$\lambda_1(k\mathcal{L}) = k\lambda_1(\mathcal{L})$$
$$\sqrt{n}(det(k\mathcal{L}))^{\frac{1}{n}} = k\sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}} \ .$$

Teorema 4.2.5. *Segundo teorema de Minkowski.* Para cualquier retículo $\mathcal{L}$ de dimensión n, se cumple

$$\left(\prod_{i=1}^{n}\lambda_i(\mathcal{L})\right)^{\frac{1}{n}} \leq \sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}} \ .$$

Demostración. Sea $\mathbf{x_1}, \ldots, \mathbf{x_n} \in \mathcal{L}$ un conjunto de vectores linealmente independientes que conforman una base $\mathcal{B}$ y pertenecen al retículo dado $\mathcal{L}$ de forma que constituyen los mínimos sucesivos

$$||\mathbf{x_i}|| = \lambda_i(\mathcal{L}) \ .$$

Sean $\mathbf{x_1^*}, \ldots, \mathbf{x_n^*}$ los vectores que conforman la ortogonalización de Gram-Schmidt. Sea E la elipsoide cuyos ejes son este conjunto de vectores $\{\mathbf{x_1^*}, \ldots, \mathbf{x_n^*}\}$ y de longitudes $\lambda_1, \ldots, \lambda_n$

$$E = \left\{\mathbf{y} : \mathbf{y} \in \mathbb{R}^n, \sum_{i=1}^{n}\left(\frac{\langle \mathbf{y}, \mathbf{x_i^*}\rangle}{||\mathbf{x_i^*}||\lambda_i}\right)^2 < 1\right\} \ .$$

Tal como ha quedado definido, el elipsoide E no contiene ningún vector no nulo del retículo. De esta forma tomando un vector no nulo perteneciente al retículo $\mathbf{y} \in \mathcal{L}$, se define $k \in [1, n]$ como el índice maximal tal que

$$||\mathbf{y}|| \geq \lambda_k(\mathcal{L}) \ .$$

Es necesario que el vector $\mathbf{y}$ pertenezca al espacio generado por los vectores de ambas bases, $\mathbf{y} \in span(\mathbf{x_1^*}, \ldots, \mathbf{x_k^*}) = span(\mathbf{x_1}, \ldots, \mathbf{x_k})$. De ocurrir lo contrario se llegaría a una contradicción, puesto que $\{\mathbf{x_1}, \ldots, \mathbf{x_k}, \mathbf{y}\}$ son $k+1$ vectores linealmente independientes de longitud inferior a $\lambda_k(\mathcal{L})$. Por lo tanto ocurre

$$\sum_{i=1}^{n}\left(\frac{\langle \mathbf{y}, \mathbf{x_i^*}\rangle}{||\mathbf{x_i^*}||\lambda_i}\right)^2 = \sum_{i=1}^{k}\left(\frac{\langle \mathbf{y}, \mathbf{x_i^*}\rangle}{||\mathbf{x_i^*}||\lambda_i}\right)^2 \geq \frac{1}{\lambda_k^2}\sum_{i=1}^{k}\left(\frac{\langle \mathbf{y}, \mathbf{x_i^*}\rangle}{||\mathbf{x_i^*}||}\right)^2 = \frac{||\mathbf{y}||^2}{\lambda_k^2} \geq 1 \ .$$

Luego el vector $\mathbf{y}$ no pertenece al elipsoide E, $\mathbf{y} \notin E$. Ahora, del Teorema del cuerpo convexo de Minkowski 4.2.3, se obtiene

$$vol(E) \leq 2^n det(\mathcal{L}) \ .$$

Y además

$$vol(E) = \left(\prod_{i=1}^{n}\lambda_i(\mathcal{L})\right) vol(B(0,1)) \geq \left(\prod_{i=1}^{n}\lambda_i(\mathcal{L})\right)\left(\frac{2}{\sqrt{n}}\right)^n \ .$$

Combinando ambas acotaciones se obtiene el resultado que se deseaba probar

$$\left(\prod_{i=1}^{n}\lambda_i(\mathcal{L})\right)^{\frac{1}{n}} \leq \sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}} \ .$$

□

4.3. Problemas difíciles

El primer teorema de Minkowski 4.2.4 arroja información acerca de una cota superior para la longitud de los vectores para cualquier retículo $\mathcal{L}$. Es decir, cualquier retículo $\mathcal{L}$ posee un vector no nulo con longitud menor o igual que $\lambda_1(\mathcal{L}) \leq \sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}}$. Contemplando la demostración de este primer teorema, queda evidenciado que ésta no es constructiva, luego no existe algoritmo que construya el vector más corto de un retículo dado de forma eficiente. Generalmente la longitud del vector más corto es extremadamente inferior a la cota dada. Por ejemplificar esta afirmación, si se considera un retículo $\mathcal{L}$ de dimensión dos, la base

$$\mathcal{L} = \left\{ \mathbf{b_1} = \varepsilon \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \mathbf{b_2} = \frac{1}{\varepsilon} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\} .$$

Luego, $\sqrt{n}(det(\mathcal{L}))^{\frac{1}{n}} = \sqrt{2} \cdot 1^{\frac{1}{2}} = \sqrt{2}$, pero la longitud del vector más corto es $\lambda_1(\mathcal{L}) = \varepsilon$, que puede ser tan pequeño como se defina.

Lo enunciado referente al hecho de determinar el vector de menor longitud de un retículo forma parte de un problema computacional denominado *Problema del vector más corto*, SVP (*Shortest Vector Problem*). Dentro del ámbito de los retículos existen una serie de problemas computacionales que no pueden ser resueltos en tiempo polinomial. De esta forma, son candidatos a formar parte de criptosistemas cuya seguridad radique en los mismos. A continuación se introducen los problemas más comunes. Se reducen estas definiciones al ámbito de los sistemas criptográficos. Por lo tanto se emplean retículos cuyas bases son enteras, en contraposición con las vectores reales con las que se venía trabajando. El motivo principal es que estos problemas deben ser representados con un número finito de bits.

Se puede hacer una analogía entre los retículos y los códigos correctores. En estos problemas de retículos se emplea generalmente la norma Euclídea, mientras que en los códigos correctores se emplea la distancia de Hamming. La definición del problema CVP es semejante al problema de decodificar con errores: dado un vector $\mathbf{x}$ que no está en el retículo (o código) hallar el vector del retículo (o código) más próximo a $\mathbf{x}$. Además, la definición de $\lambda_1(\mathcal{L})$ es similar a la distancia mínima de un código corrector de errores. Mientras $\lambda_1(\mathcal{L})$ se generaliza a los mínimos sucesivos y los valores son crecientes, la distancia mínima se generaliza a las llamadas "distancias generalizadas", que también son crecientes.

4.3.1. Shortest Vector Problem, SVP

Definición 4.3.1. El problema *SVP (Shortest Vector Problem)* se define como, dado un retículo $\mathcal{L}$ y una base $\mathcal{B} \in \mathbb{Z}^{m \times n}$ de ese retículo, encontrar un vector no nulo $\mathbf{x} \in \mathcal{L}(\mathcal{B})$ tal que

$$||\mathbf{x}|| = \lambda_1(\mathcal{L}) .$$

Existen versiones derivadas de esta definición, en función de si se desea encontrar la longitud exacta del vector o hallar un vector menor que una constante establecida. Se puede comprobar que las Definiciones 4.3.1, 4.3.2 y 4.3.3 son equivalentes.

Definición 4.3.2. *Optimización SVP.* Dado un retículo $\mathcal{L}$ y una base $\mathcal{B} \in \mathbb{Z}^{m \times n}$ de ese retículo, se define el problema de *Optimización SVP* como encontrar el valor $\lambda_1(\mathcal{L}(\mathcal{B}))$.

Definición 4.3.3. *Decisión SVP.* Dado un retículo $\mathcal{L}$, una base $\mathcal{B} \in \mathbb{Z}^{m \times n}$ de ese retículo y un número racional $d \in \mathbb{Q}$, el problema de *Decisión SVP* se define como determinar cuando es cierta la desigualdad $\lambda_1(\mathcal{L}(\mathcal{B})) \leq d$ y para que valores no es cierta.

Un problema de decisión se plantea cuando es necesario determinar bajo qué condiciones se satisface una propiedad específica, y bajo qué condiciones no se cumple. Debido a la falta de algoritmos eficientes para resolver cualquiera de las cuestiones concernientes a los problemas SVP, se consideran soluciones aproximadas de los mismos. El factor de aproximación viene dado por un parámetro o una función $\gamma(n) > 1$, siendo n la dimensión del retículo empleado del cual depende. Ocurre que la dificultad de resolución se incrementa, si se incrementa el valor de $\gamma(n)$.

Definición 4.3.4. *$SVP_{\gamma(n)}$*. Dado un retículo $\mathcal{L}$ y una base $\mathcal{B} \in \mathbb{Z}^{m\times n}$ de ese retículo, encontrar un vector no nulo $\mathbf{x} \in \mathcal{L}(\mathcal{B})$ tal que

$$||\mathbf{x}|| \leq \gamma(n) \cdot \lambda_1(\mathcal{L}) \ .$$

Definición 4.3.5. *Optimización $SVP_{\gamma(n)}$*. Dado un retículo $\mathcal{L}$ y una base $\mathcal{B} \in \mathbb{Z}^{m\times n}$ de ese retículo, encontrar el valor d de forma que se cumpla

$$d \leq \lambda_1(\mathcal{L}) \leq \gamma(n) \cdot d \ .$$

Definición 4.3.6. *Promesa $SVP_{\gamma(n)}$*. Dado un retículo $\mathcal{L}$, una base $\mathcal{B} \in \mathbb{Z}^{m\times n}$ de ese retículo y un número racional $d \in \mathbb{Q}$, determinar cuando es cierta la desigualdad $\lambda_1(\mathcal{L}(\mathcal{B})) \leq d$ y para que valores se cumple $\lambda_1(\mathcal{L}(\mathcal{B})) > \gamma(n) \cdot d$.

Un problema de promesa conforma la generalización de un problema de decisión. Una promesa es un par de elementos disjuntos (π_{SI}, π_{NO}). De esta forma, un algoritmo resuelve esta promesa si para cada elemento e distingue correctamente si pertenece a $e \in \pi_{SI}$ o en su defecto pertenece a $e \in \pi_{NO}$. Este problema generalmente también se denomina *$GapSVP_{\gamma(n)}$*.

El algoritmo que mayor precisión alcanza para resolver el problema aproximado $\mathrm{SVP}_{\gamma(n)}$ es el algoritmo de Lenstra-Lenstra-Lovász (LLL). Este trabajo no se entra en detalles técnicos sobre este algoritmo, para una extensa comprensión por parte del lector se recomienda consultar [11]. El algoritmo LLL tiene por objetivo a partir de una base de un retículo conseguir una base ortogonal y reducida.

Definición 4.3.7. Una base $\mathcal{B} = \{\mathbf{b_1}, \ldots, \mathbf{b_n}\}$ es *LLL-reducida* con parámetro ρ si se verifica:

- $|\mu_{i,j}| \leq \frac{1}{2}$ para todo $i > j$, siendo $|\mu_{i,j}| = \frac{\langle \mathbf{b_i}, \mathbf{b_j} \rangle}{\langle \mathbf{b_j}, \mathbf{b_j} \rangle}$ los coeficientes de Gram-Schmidt.
- para cada par de vectores consecutivos $\mathbf{b}_i, \mathbf{b}_{i+1}$ se cumple

$$\frac{3}{4}||\pi_i(\mathbf{b}_i)||^2 \leq ||\pi_i(\mathbf{b}_{i+1}) + \mu_{ii}\pi_i(\mathbf{b}_i)||^2 \quad \forall i = 1, \ldots, n$$

  siendo $\pi_i(\mathbf{b}_i)$ la componente de $\mathbf{b}_i$ ortogonal al espacio formado por las combinaciones lineales de los elementos de la base (a excepción de $\mathbf{b}_i$). También se denota como $\pi_i(\mathbf{b}_i) = \mathbf{b}_i^*$.

Este algoritmo no es capaz de encontrar un vector de dimensión exactamente λ_1, pero consigue que la función $\gamma(n)$ tome la forma $\gamma(n) = (2/\sqrt{3})^n$ siendo n el rango del retículo, es decir encuentra un vector en el retículo de al menos dimensión $\gamma(n) \cdot \lambda_1 = (2/\sqrt{3})^n \cdot \lambda_1$. Se puede distinguir como caso particular cuando el retículo posee dimensión dos. En este

caso, es asequible encontrar con exactitud un vector del retículo con longitud $||\mathbf{a}|| = \lambda_1$. Esto es posible gracias a que se puede encontrar una base $\{\mathbf{b_1}, \mathbf{b_2}\}$ con $||\mathbf{b1}|| = ||\mathbf{a}|| = \lambda_1$ y $\mathbf{b_2} = \lambda_2$. De esta forma, el algoritmo se encarga de ir determinando los mínimos sucesivos del retículo. Supone una generalización del algoritmo de Gauss para cualquier norma.

A pesar de esto, los problemas originales siguen siendo difíciles de resolver, puesto que el factor $\gamma(n)$ que se consigue es de orden exponencial, luego las soluciones obtenidas se encuentran muy alejadas de la solución exacta para $n \gg 0$. Para dimensiones bajas el algoritmo LLL, como se ha visto, suele obtener soluciones exactas o muy próximas a estas.

4.3.2. Closest Vector Problem, CVP

Otro problema computacional relevante en el ámbito de los retículos es el *Problema del vector más cercano*. Como su propio nombre indica, su expresión general consiste en encontrar el vector, perteneciente al retículo escogido, más cercano a un vector dado. Este vector puede o no pertenecer al retículo.

Definición 4.3.8. El problema *Closest Vector Problem (CVP)* se define como, sea $\mathcal{L}$ un retículo, $\mathcal{B} \in \mathbb{Z}^{m\times n}$ una base de ese retículo, $\mathbf{v} \in \mathbb{Z}^m$ un vector, que generalmente no pertenece al retículo, encontrar un elemento perteneciente al retículo $\mathbf{x} \in \mathcal{L}(\mathcal{B})$ tal que se minimice la norma $||\mathbf{x} - \mathbf{v}||$.

Definición 4.3.9. *Optimización CVP*. Sea $\mathcal{L}$ un retículo, $\mathcal{B} \in \mathbb{Z}^{m\times n}$ una base de ese retículo, $\mathbf{v} \in \mathbb{Z}^m$ un vector, encontrar el valor de la norma mínima $||\mathbf{x} - \mathbf{v}||$.

Definición 4.3.10. *Decisión CVP*. Sea $\mathcal{L}$ un retículo, $\mathcal{B} \in \mathbb{Z}^{m\times n}$ una base de ese retículo, $\mathbf{v} \in \mathbb{Z}^m$ un vector, $r \in \mathbb{Q}$, decidir cuando se verifica $||\mathbf{x} - \mathbf{v}|| \leq r$.

El problema de decisión es un problema NP-completo, luego no existe ningún algoritmo capaz de resolverlo en tiempo polinomial. Una posible demostración de esta afirmación se puede encontrar en [14]. Por otra parte, siguiendo la misma dinámica que en el caso del problema del vector más corto, las versiones aproximadas a estos problemas se formulan como se describe a continuación, con $\gamma(n) > 1$.

Definición 4.3.11. El problema $\text{CVP}_{\gamma(n)}$ se define como, sea $\mathcal{L}$ un retículo, $\mathcal{B} \in \mathbb{Z}^{m\times n}$ una base de ese retículo, $\mathbf{v} \in \mathbb{Z}^m$ un vector, encontrar un elemento perteneciente al retículo $\mathbf{x} \in \mathcal{L}(\mathcal{B})$ tal que se verifique

$$||\mathbf{x} - \mathbf{v}|| \leq \gamma(n) \cdot dist(\mathbf{v}, \mathcal{L}(\mathcal{B})) \ .$$

Definición 4.3.12. *Optimización* $CVP_{\gamma(n)}$. Sea $\mathcal{L}$ un retículo, $\mathcal{B} \in \mathbb{Z}^{m\times n}$ una base de ese retículo, $\mathbf{v} \in \mathbb{Z}^m$ un vector, encontrar d de forma que se cumpla

$$d \leq dist(\mathbf{v}, \mathcal{L}(\mathcal{B})) \leq \gamma(n) \cdot d \ .$$

Definición 4.3.13. *Promesa* $CVP_{\gamma(n)}$. Sea $\mathcal{L}$ un retículo, $\mathcal{B} \in \mathbb{Z}^{m\times n}$ una base de ese retículo, $\mathbf{v} \in \mathbb{Z}^m$ un vector, $r \in \mathbb{Q}$. La condición π_{SI} abarca las situaciones en las que se cumple $dist(\mathbf{v}, \mathcal{L}(\mathcal{B})) \leq r$, mientras que π_{NO} engloba las situaciones donde se verifica $dist(\mathbf{v}, \mathcal{L}(\mathcal{B})) > \gamma(n) \cdot r$. Este problema generalmente también se denomina $GapCVP_{\gamma(n)}$.

Es factible comprobar que para cualquier factor $\gamma(n)$, encontrar soluciones aproximadas para el problema del vector más cercano resulta más sencillo que encontrarlas para el problema original del vector más cercano. De igual manera que ocurría con la aproximación del problema del vector más corto, también es posible emplear el algoritmo LLL para resolver en tiempo polinomial la versión aproximada del problema del vector más cercano $\text{CVP}_{\gamma(n)}$. De esta manera, la función $\gamma(n)$ toma la forma $\gamma(n) = 2(2/\sqrt{3})^n$. Gracias al algoritmo LLL, se parte de una base reducida $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ y de un vector $\mathbf{v}$. Posteriormente se aplica de nuevo el algoritmo LLL para el conjunto de vectores $\{\mathcal{B}, \mathbf{v}\}$. Esto permite obtener un vector $\mathbf{x}$ tal que la diferencia $\mathbf{x} - \mathbf{v}$ se exprese como

$$\mathbf{x} - \mathbf{v} = \mathbf{v}^* + \sum_{i=1}^{n} c_i \mathbf{b}_i^* \qquad |c_i| \leq \frac{1}{2} \; \forall i = 1, .., n$$

siendo $\mathbf{v}^*$ la componente de $\mathbf{v}$ ortogonal al conjunto de combinaciones lineales de la base dada $\mathcal{B}$. Actuando de esta manera, quedaría por tanto comprobar que la distancia entre $\mathbf{v}$ y $\mathbf{x}$ verifica

$$||\mathbf{x} - \mathbf{v}|| \leq 2 \cdot (2/\sqrt{3})^n \cdot dist(\mathbf{v}, \mathcal{L}(\mathcal{B})) \; .$$

4.3.3. Short Integer Solution, SIS y SIVP

Existen variantes de los problemas anteriores, como es el caso del *Problema de los vectores independientes más cortos*, relacionado con el mínimo sucesivo de orden n.

Definición 4.3.14. *SIVP.* Dado $\mathcal{L}$ un retículo y $\mathcal{B} \in \mathbb{Z}^{m \times n}$ una base de ese retículo, encontrar un conjunto de n vectores linealmente independientes $\{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$ tal que

$$||\mathbf{v_i}|| \leq \lambda_n(\mathcal{L}) \quad i \in \{1, \ldots, n\} \; .$$

Definición 4.3.15. *SIVP*$_\gamma(n)$. Dado $\mathcal{L}$ un retículo y $\mathcal{B} \in \mathbb{Z}^{m \times n}$ una base de ese retículo, $\gamma > 1$, encontrar un conjunto de n vectores linealmente independientes $\{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$, tales que

$$||\mathbf{v_i}|| \leq \gamma(n)\lambda_n(\mathcal{L}) \quad i \in \{1, \ldots, n\} \; .$$

Definición 4.3.16. *Short Integer Solution (SIS).* Sea $A \in \mathcal{M}(\mathbb{Z}_q^{m \times n})$ una matriz, sea $\mathcal{L}(A)^\perp\{\mathbf{v} : \mathbf{v} \in \mathbb{Z}^n, \; A\mathbf{v} = 0 \mod q\}$, luego es necesario encontrar un vector $\mathbf{x} \in \mathbb{Z}^m$ perteneciente al retículo $\mathcal{L}(A)^\perp$ de forma que

$$||\mathbf{x}|| < d \quad \text{con} \quad \sqrt{n \log q} \leq d < q \; .$$

La acotación de d asegura que la ecuación $A\mathbf{v} = 0$ posea soluciones no triviales.

4.3.4. Learning With Errors, LWE

LWE

El problema de *Aprendizaje con Errores, LWE*, fue introducido en el año 2005 por Oded Regev [16]. El propósito buscado era conseguir que el caso peor de este problema fuera difícil de resolver.

Definición 4.3.17. *Learning With Errors (LWE).* Dada $A \in \mathcal{M}(\mathbb{Z}_q^{m\times n})$ una matriz, se considera el retículo constituido por las combinaciones lineales de las columnas de la matriz $A \ mod\ q$

$$\mathcal{L}(A) = \{\mathbf{z} : \mathbf{z} \in \mathbb{Z}_q^m,\ \mathbf{z} \equiv A\mathbf{s}\ mod\ q \quad \text{para algún vector } \ \mathbf{s} \in \mathbb{Z}_q^n\}\ .$$

El problema LWE trata de encontrar un vector $\mathbf{s}$ tal que se verifica

$$\mathbf{b} = A\mathbf{s} + \varepsilon\ \ mod\ q$$

para un error $\varepsilon \in \mathbb{Z}_q^m$ "pequeño"que sigue una cierta distribución de probabilidad a partir del conocimiento de A y $\mathbf{b}$.

Existe la versión *LWE de búsqueda* de este problema, planteada en términos de encontrar el vector $\mathbf{b}$ que se encuentre más próximo a un vector del retículo, mientras que la versión *LWE de decisión* plantea distinguir cuando $\mathbf{b}$ proviene de una distribución uniforme.

Si no existiera el término del error, el cálculo del vector $\mathbf{s}$ resultaría extremadamente sencillo. Esto es debido a que tras n ecuaciones, realizando un proceso de eliminación Gaussiana se conseguiría recuperar el término buscado $\mathbf{s}$. En cambio, si se añade al sistema el vector del error, cuando se emplea el algoritmo de eliminación Gaussiana, éste se amplifica de forma que las ecuaciones resultantes no arrojan ningún tipo de información valiosa. El peor caso del problema LWE posee complejidad exponencial, incluso haciendo uso de las herramientas que proporciona la computación cuántica. Por otra parte, LWE se considera una extensión del problema LPN (*Learning Parity with Noise*), estando este problema considerado como un problema difícil de resolver. Además, LWE constituye la base de problemas tales como SIVP y el problema de decisión SVP.

Ring LWE, RLWE

El problema LWE original emplea vectores enteros módulo un número primo q, por ejemplo, $\mathbf{z} \in \mathbb{Z}_q^n$. La principal diferencia de Ring LWE con respecto LWE radica en la sustitución del grupo $\mathbb{Z}_q^n$ de la Definición 4.3.17 por el anillo de polinomios $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n+1)$. Por ejemplo, $r = r_0 + r_1x + \ldots + r_{n-1}x^{n-1} \in \mathbb{Z}_q[x]/(x^n+1)$ r posee coeficientes enteros módulo q. Se presentan unas definiciones previas antes de presentar el problema *Ring LWE*.

Proposición 4.3.1. Sea P un anillo conmutativo. Se considera el conjunto de polinomios sobre P

$$P[x] = \left\{ \sum_{i=0}^{n} p_i x^i : p_i \in P,\ \ 1 \leq i \leq n \right\}$$

con las operaciones suma + y producto · definidas como:

- Suma

$$q(x) + s(x) = \sum_{i=0}^{n} q_i x^i + \sum_{j=0}^{m} s_j x^j = \sum_{k=0}^{n} (q_k + s_k) x^k$$

suponiendo que $m \leq n$, es decir, $s_j = 0\ \ \forall j \in \{m+1, \ldots, n\}$, $q(x)$, $s(x) \in P[x]$.

- Producto

$$q(x) \cdot s(x) = \left(\sum_{i=0}^{n} q_i x^i\right)\left(\sum_{j=0}^{m} s_j x^j\right) = \sum_{k=0}^{m+n} c_k x^k \ .$$

siendo $c_k = \sum_{i+j=k} q_i s_i$, $q(x)$, $s(x) \in P[x]$.

Entonces este conjunto de polinomios $P[x]$ junto con las operaciones definidas constituye un anillo conmutativo, denominado *anillo de polinomios con coeficientes en P*, $(P[x], +, \cdot)$.

Se define ahora el concepto de polinomio ciclotómico, que es aquel cuyas raíces son todas las raíces primitivas de orden n de la unidad.

Definición 4.3.18. Sea $n \in \mathbb{N}$. El *n-ésimo polinomio ciclotómico* $\phi_n(x)$ se define de forma recursiva, partiendo de $\phi_1(x) = x - 1$:

$$\phi_n(x) = \frac{x^n - 1}{\prod_{\substack{d|n \\ d<n}} \phi_d(x)} \ .$$

Proposición 4.3.2. Sea $(A, +, \cdot)$ un anillo conmutativo y unitario. Sea I un ideal de A, entonces el conjunto cociente A/I es un anillo conmutativo y unitario con las operaciones

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

Este anillo A/I se denomina *anillo cociente de A sobre I.*

Estas nociones son de utilidad puesto que el anillo con el que se va a trabajar es $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ siendo q un número primo y $n = 2^{m-1}$ tal que $x^n + 1$ es el 2^m-ésimo polinomio ciclotómico. Además es inmediata la comprobación de que es correcto definir este anillo como el anillo cociente de $\mathbb{Z}_q[x]$ sobre el ideal (x^n+1). Esto se deriva de que, por la Proposición 4.3.2 $\mathbb{Z}_q$ es el anillo cociente de $\mathbb{Z}$ sobre el ideal (q). Y por la Proposición 4.3.1 se puede considerar el anillo de polinomios $\mathbb{Z}_q[x]$. Por otra parte, generalmente n es una potencia de 2, para que se garantice que $x^n + 1$ sea un polinomio irreducible en $\mathbb{Q}$. Además, trabajando con este anillo los problemas de decisión LWE y el problema de búsqueda son equivalentes. Tras estas indicaciones es correcto formular el problema *Ring LWE* como sigue:

Definición 4.3.19. *Ring Learning With Errors (RLWE).* Sea $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ un anillo y sea $\mathbf{a} \in \mathcal{R}_q$. El problema RLWE trata de encontrar un vector $\mathbf{s} \in \mathcal{R}_q$ tal que verifique

$$\mathbf{b} = \mathbf{as} + \varepsilon \ \ mod\ q \in \mathcal{R}_q$$

para un error $\varepsilon \in \mathbb{Z}_q$ "pequeño"que sigue una cierta distribución de probabilidad a partir del conocimiento de $\mathbf{a}$ y $\mathbf{b}$.

Module LWE, MLWE

Por realizar una analogía, un M-módulo es a un anillo lo que un espacio vectorial es a un cuerpo. El problema *Module-LWE (MLWE)* va a trabajar con el módulo de anillos de polinomios. MLWE es una generalización de RLWE, más concretamente RLWE es un MLWE de rango uno. Por esclarecer este concepto, se define lo que es un $M-$Módulo.

Definición 4.3.20. Sea $\mathcal{R}$ un anillo, un $\mathcal{R}$*-módulo* es un conjunto M que posee las operaciones de suma $a+b \in M$, $a, b \in M$ y producto por escalares $k \cdot a \in M$, $a \in M$, $k \in \mathcal{R}$. Además, cumple las siguientes propiedades

- Asociativa:
$$a + (b + c) = (a + b) + c$$
$$r \cdot (s \cdot a) = (r \cdot s) \cdot a \ .$$
- Conmutativa: $a + b = b + a$.
- Distributiva:
$$r \cdot (a + b) = r \cdot a + r \cdot b$$
$$(r + s) \cdot a = r \cdot a + s \cdot a \ .$$
- Elemento neutro:
$$0 + a = a$$
$$1a = a \ .$$
- Elemento opuesto: $a + (-a) = 0$.

Definición 4.3.21. *Module Learning With Errors (RLWE).* Sea $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ un anillo. Dada $A \in \mathcal{M}(\mathcal{R}_q)$ una matriz, se considera el retículo constituido por las combinaciones lineales de las columnas de la matriz $A \ mod\ q$

$$\mathcal{L}(A) = \{\mathbf{z} : \mathbf{z} \in \mathbb{Z}_q^m,\ \mathbf{z} \equiv A\mathbf{s}\ mod\ q \quad \text{para algún vector } \ \mathbf{s} \in \mathcal{R}_q\} \ .$$

Luego MLWE trata de encontrar un vector $\mathbf{s}$ tal que se verifica

$$\mathbf{b} = A\mathbf{s} + \varepsilon \ \ mod\ q \in \mathcal{R}_q$$

para un error $\varepsilon \in \mathbb{Z}_q$ que sigue una cierta distribución de probabilidad a partir del conocimiento de A y $\mathbf{b}$.

Capítulo 5

Algoritmo CRYSTALS-KYBER

Este capítulo detalla CRYSTALS-KYBER [2], finalista del concurso del NIST tanto en el ámbito de la criptografía de clave pública (PKE, *Public Key Encription Scheme*) como en el de intercambio de claves (KEM, *Key Encapsulation Method*). La seguridad que posee el algoritmo criptográfico es IND-CPA, es decir indistinguibilidad frente ataques de texto plano escogido, mientras que el intercambio de claves posee indistinguibilidad frente ataques de texto cifrado escogido adaptativo IND-CCA2. La seguridad de KYBER se analizará en detalle en la Subsección 5.2.3, ésta se basa en la dificultad de resolver el problema de Aprendizaje con Errores LWE en retículos modulares. Trabajar con módulos permite realizar ciertas operaciones de forma más eficiente pero, a su vez, trabajar con estructuras de mayor estructura algebraica puede facilitar algunos tipos de ataque. Es por tanto necesario establecer un equilibrio entre la seguridad y la eficiencia. Por otra parte, se abordará en primer lugar (Sección 5.1) una breve descripción sobre el algoritmo LWE, algoritmo sobre el que se fundamenta CRYSTALS-KYBER.

5.1. Algoritmo LWE

Emplea un conjunto de parámetros $n, m, l, t, r, q \in \mathbb{Z}$ y $\alpha \in \mathbb{R}$. El parámetro n es el principal parámetro de seguridad y se corresponde con la dimensión de los retículos que constituyen el peor caso posible. Se emplea además una función auxiliar f y su inversa f^{-1}. f se encarga de asociar el espacio de mensajes $\mathbb{Z}_t^l$ con $\mathbb{Z}_q^l$ multiplicando cada coordenada por q/t y redondeando al entero más cercano. Inversamente, f^{-1} para cada elemento de $\mathbb{Z}_q^l$ lo convierte en un elemento del espacio $\mathbb{Z}_t^l$ dividiendo cada coordenada entre q/t y redondeando.

La clave privada está compuesta por una matriz $\mathbf{S} \in \mathcal{M}_{n\times l}(\mathbb{Z}_q)$ uniformemente aleatoria siguiendo una distribución normal. Por su parte, se escoge $\mathbf{A} \in \mathcal{M}_{m\times n}(\mathbb{Z}_q)$ uniformemente aleatorio y una matriz de error $\mathbf{E} \in \mathbb{Z}_q^{m\times l}$. De esta forma la clave pública está formada por el par

$$(\mathbf{A}, \mathbf{B} = \mathbf{AS} + \mathbf{E}) \in \mathcal{M}_{m\times n}(\mathbb{Z}_q) \times \mathcal{M}_{m\times l}(\mathbb{Z}_q) \ .$$

Una vez generadas ambas claves, el algoritmo de cifrado toma como entrada un mensaje $\mathbf{m} \in \mathbb{Z}_t^l$ y la clave pública generada anteriormente $(\mathbf{A}, \mathbf{B})$. Se escoge un vector $\mathbf{a} \in \{-r, -r+1, \ldots, r\}^m$ uniformemente aleatorio de forma que el output de este algoritmo es

$$(\mathbf{u} = \mathbf{A}^T\mathbf{a}, \mathbf{v} = \mathbf{B}^T\mathbf{a} + f(\mathbf{m})) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^l \ .$$

Por su parte, el algoritmo de descifrado toma como entrada el texto cifrado $(\mathbf{u}, \mathbf{v}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^l$ y la clave privada $\mathbf{s} \in \mathcal{M}_{n\times l}(\mathbb{Z}_q)$ produciendo como salida el texto en claro $f^{-1}(\mathbf{v} - \mathbf{s}^T\mathbf{u})$.

5.2. Algoritmo CRYSTALS-KYBER

5.2.1. Notación y conceptos previos

KYBER emplea el anillo de polinomios $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1) = \mathbb{Z}[x]/(x^{256} + 1)$ y el anillo de polinomios $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1) = \mathbb{Z}_{7681}[x]/(x^{256} + 1)$ con $n = 256$ y $q = 7681$ valores constantes. Se cumple que q es un número primo y $n = 2^{m-1}$ es tal que $x^n + 1$ es el 2^m-ésimo polinomio ciclotómico. El algoritmo KYBER trabaja con vectores de bytes, siendo cada byte un vector formado por 8 bits. Se denota por $\mathcal{B}$ al conjunto de enteros $\{0, \ldots, 255\}$, por $\mathcal{B}^k$ al conjunto de vectores de bytes de longitud k y por $\mathcal{B}^*$ al conjunto de vectores de bytes de longitud arbitraria. Por otra parte, se denota $a||b$ a la concatenación de los vectores de bytes $a, b \in \mathcal{B}^*$. Además $\lceil x \rfloor$ es el entero más cercano a un elemento $x \in \mathbb{R}$. Además es necesario contar con un conjunto de funciones auxiliares:

- Una función pseudoaleatoria, PRF (*Pseudo Random Function*)

$$\text{PRF} : \mathcal{B}^{32} \times \mathcal{B} \longrightarrow \mathcal{B}^* \ .$$

- Dos funciones *hash* de 256 y 512 bits respectivamente

$$H : \mathcal{B}^* \longrightarrow \mathcal{B}^{32}$$

$$G : \mathcal{B}^* \longrightarrow \mathcal{B}^{32} \times \mathcal{B}^{32} \ .$$

- Una función de salida extensible, XOF (*Extendable-Output Function*), es decir, una función *hash* cuya salida puede tener la longitud deseada

$$\text{XOF} : \mathcal{B}^* \times \mathcal{B} \times \mathcal{B} \longrightarrow \mathcal{B}^* \ .$$

- Una función que genera una clave privada a partir de un valor $b \in \mathcal{B}^*$ que es también secreto, KDF (*Key Derivation Function*)

$$\text{KDF} : \mathcal{B}^* \longrightarrow \mathcal{B}^* \ .$$

Las implementaciones de estas funciones son algoritmos conocidos y empleados en el ámbito de la criptografía. Para instanciar PRF se emplea SHAKE-256, al igual que para KDF. Para la función XOF se emplea el algoritmo SHAKE-128, mientras que para las funciones *hash* H y G se emplea SHA3-256 y SHA3-512 respectivamente.

Reducción modular

Definición 5.2.1. Sea q un entero par (respectivamente impar) positivo, $q \in \mathbb{Z}^+$. Se define $r' \equiv r \ \ mod^{\pm} \ q$ como el único entero $r' \in \left[-\frac{q}{2}, \frac{q}{2}\right]$ (respectivamente $r' \in \left[-\frac{q-1}{2}, \frac{q-1}{2}\right]$) tal que $r' \equiv r \ \ mod \ q$.

Definición 5.2.2. Sea q un entero positivo, $q \in \mathbb{Z}^+$. Se define $r' \equiv r \ \ mod^{+} \ q$ como el único entero $r' \in [0, q]$ tal que $r' \equiv r \ \ mod \ q$.

Se observa que la Definición 5.2.2 se trata del resto positivo usual en la división entera.

Compresión y descompresión

KYBER emplea dos funciones auxiliares de compresión y descompresión que tienen el propósito de descartar bits del texto cifrado que tienen una baja influencia sobre la probabilidad de corrección del proceso, consiguiendo así reducir el tamaño del mismo.

Definición 5.2.3. Una *función de compresión* $\texttt{Compress}_q(x, d)$ es aquella que para cada elemento $x \in \mathbb{Z}_q$ lo hace corresponder con un entero en el rango $\{1, \ldots, 2^d - 1\}$ siendo $d < \lceil \log_2(q) \rceil$.

Definición 5.2.4. Una *función de descompresión* es aquella función tal que cada elemento

$$x' = \texttt{Decompress}_q(\texttt{Compress}_q(x, d), d)$$

es un elemento próximo a x, es decir

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

Las funciones de compresión y descompresión particulares que emplea KYBER son

$$\texttt{Compress}_q(x, d) = \left\lceil \frac{2^d}{q} x \right\rfloor \mod^{+} 2^d$$

$$\texttt{Decompress}_q(x, d) = \left\lceil \frac{2^d}{q} x \right\rfloor$$

y satisfacen las condiciones de las Definiciones 5.2.3, 5.2.4. En el caso particular de trabajar con elementos pertenecientes a un polinomio o un vector de polinomios, estas funciones se aplican a cada coeficiente de cada polinomio.

Number Theoretic Transform (NTT)

La Transformada Teórica de Números (NTT) es un caso particular de la Transformada Rápida de Fourier cuando se trabaja en un anillo de números enteros módulo un número primo. Este concepto permite basándose en el teorema de convolución, realizar la multiplicación de polinomios de forma eficiente obteniendo una complejidad de $\mathcal{O}(n \log n)$ en lugar de $\mathcal{O}(n^2)$.

Definición 5.2.5. Sea $q \in \mathbb{Z}$, sean $a(x), b(x) \in \mathbb{Z}_q[x]$ dos polinomios de grado $n - 1$. La *multiplicación de polinomios* entre $a(x)$ y $b(x)$ se define como

$$c(x) = a(x) \cdot b(x) = \sum_{k=0}^{2(n-1)} c_k x^k \ ,$$

con $c_k = \sum_{i=0}^{k} a_i b_{k-i} \mod q$, siendo $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1})$ y $\mathbf{b} = (b_0, b_1, \ldots, b_{n-1})$ el vector de coeficientes de los polinomios $a(x)$ y $b(x)$ respectivamente.

Dada esta definición, se puede demostrar que es equivalente a una convolución lineal discreta entre los vectores de coeficientes de cada uno de los polinomios, es decir

$$c[k] = (a * b)[k] = \sum_{i=0}^{k} a[i] b[k - i] \ .$$

Para el caso concreto de KYBER interesa calcular el *Negative Wrapped Convolution* por medio de NTT para conseguir una forma eficiente de multiplicación de polinomios (Proposición 5.2.1). Estos conceptos se describen en las Definiciones 5.2.6, 5.2.7, 5.2.8 sucesivas.

Definición 5.2.6. Sean $a(x), b(x) \in \mathcal{R}_q$ dos polinomios de grado $n-1$. En caso de no poseer el grado indicado se completan los coeficientes con ceros. Sean $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1}) \in \mathbb{Z}_q^n$ y $\mathbf{b} = (b_0, b_1, \ldots, b_{n-1}) \in \mathbb{Z}_q^n$ el vector de coeficientes de cada uno de los polinomios. El *Negative Wrapped Convolution (NWC)* entre $a(x)$ y $b(x)$ se define como el producto de ambos polinomios:

$$c(x) = \sum_{i=0}^{n-1} c_i x^i$$

siendo cada $c_i = \sum_{j=0}^{i} a_j b_{i-j} - \sum_{j=i+1}^{n-1} a_j b_{i+n-j}$.

Definición 5.2.7. Sea $\mathbf{a} = (a_0, \ldots, a_{n-1})$ el vector de coeficientes del polinomio $a(x) \in \mathcal{R}_q$ de grado $n-1$. Sea $q \in \mathbb{N}$ un número primo de forma que se verifique que $q \equiv 1 \mod 2n$. Además es necesario que exista la $2n$-ésima raíz de la unidad en $\mathbb{Z}^n$ denotada por ξ. El *Negative Wrapped Number Teoretic Transform* (NTT) de un vector de coeficientes se define como

$$\mathrm{NTT}_{\xi}(\mathbf{a}) = \hat{\mathbf{a}}$$

siendo $\forall i \in [0, n-1]$

$$\hat{a}_i = \sum_{j=0}^{n-1} \xi^{2ij+i} a_j \mod q \ .$$

Definición 5.2.8. Sea $\hat{\mathbf{a}} = (\hat{a}_0, \ldots, \hat{a}_{n-1})$ el vector de coeficientes del polinomio $\hat{a}(x) \in \mathcal{R}_q$ de grado $n-1$. Bajo las mismas hipótesis de la Definición 5.2.7, se define el *Negative Wrapped Inverse of Number Teoretic Transform* (INTT)

$$\mathrm{INTT}_{\xi^{-1}}(\hat{\mathbf{a}}) = \mathbf{a}$$

siendo $\forall i \in [0, n-1]$

$$a_i = n^{-1} \sum_{j=0}^{n-1} \xi^{-2ij-j} \hat{a}_j \mod q \ .$$

Proposición 5.2.1. Sean $a(x), b(x) \in \mathcal{R}_q$ dos polinomios de grado $n-1$. Sean $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1}) \in \mathbb{Z}_q^n$ y $\mathbf{b} = (b_0, b_1, \ldots, b_{n-1}) \in \mathbb{Z}_q^n$ el vector de coeficientes de cada uno de los polinomios. El *Negative Wrapped Convolution (NWC)* entre $a(x)$ y $b(x)$ puede ser expresado como

$$\mathbf{c} = \mathrm{INTT}_{\xi_{-1}}(\mathrm{NTT}_{\xi}(\mathbf{a}) \circ \mathrm{NTT}_{\xi}(\mathbf{b})) \ ,$$

siendo $\circ$ la multiplicación elemento a elemento.

La demostración se puede consultar en [1]. Por otra parte para el valor de $q = 7681$ esta definición es válida. Pero en 2019 se publicó una tercera versión del algoritmo KYBER en la que se reducía el parámetro q, adquiriendo el valor $q = 3329$. Para esta última versión es necesario emplear una definición alternativa para la multiplicación eficiente de polinomios puesto que no existe la $2n$-ésima raíz de la unidad ξ en $\mathbb{Z}^n$. Los detalles técnicos de esta implementación se pueden consultar en [13].

Distribuciones de probabilidad

Para un conjunto S, la notación $s \leftarrow S$ implica que el elemento s se conforma aleatoriamente tomando una distribución uniforme de los elementos de S. En el caso de que S sea una distribución de probabilidad, entonces $s \leftarrow S$ implica que s se escoge siguiendo la distribución establecida por S.

Por otra parte, se define la función

$$\text{Parse: } \mathcal{B}^* \longrightarrow \mathcal{R}_q \ ,$$

tomando como entrada un vector de bytes $\mathbf{B} = (b_0, b_1, \ldots) \in \mathcal{B}^*$ y generando la NTT-representación $\hat{a} = \hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{n-1} x^{n-1} \in \mathcal{R}_q$. De esta forma se asegura que si el vector de entrada es uniformemente aleatorio, entonces los coeficientes del polinomio resultante también se distribuyen de forma uniformemente aleatoria en el anillo $\mathcal{R}_q$. La demostración se basa en el hecho de que la función NTT es biyectiva y conserva la uniformidad de la aleatoriedad de los coeficientes.

Dentro de esta sección, es necesario mencionar también la forma que emplea KYBER para generar ruido. Hace uso de una distribución binomial centrada $B_\eta = \{0,1\}^{2\eta}$, $\eta \in \mathbb{Z}^+$ para asegurar que los coeficientes de los polinomios involucrados se generen de forma pseudoaleatoria y uniforme. Se define de esta forma una función que toma como entrada $(a_1, \ldots, a_n, b_1, \ldots, b_n) \longleftarrow \{0,1\}^{2\eta}$ y produce la salida $\sum_{i=1}^{\eta}(a_i - b_i)$. Esto es generalizable a un vector de polinomios $\mathbf{b} \in \mathcal{R}_q$, de forma que sea generado siguiendo la distribución de $\mathcal{B}_\eta$. Es decir,

$$\text{CBD}_\eta : \mathcal{B}^{64\eta} \longrightarrow \mathcal{R}_q$$

Toma como entrada $\mathbf{b} = (b_0, b_1, \ldots, b_{64\eta-1}) \in \mathcal{B}^{64\eta}$ y produce la salida $f_0 + f_1 x + f_2 x^2 + \ldots + f_{n-1} x^{n-1}$. La descripción precisa tanto de Parse como de CBD se encuentra en el Apéndice A.

Codificación y decodificación

Es necesario serializar y deserializar vectores de polinomios. Para ello se definen las funciones de codificación y decodificación, siendo inversas entre ellas. Se encarga de codificar o decodificar (según se emplee Encode o Decode respectivamente) cada uno de los polinomios de forma individual del vector.La descripción detallada de la función Decode se encuentra en el Apéndice A. Ambas dependen de un parámetro ℓ que determina el número de bits que se van agrupando para obtener los coeficientes enteros correspondientes:

$$\begin{aligned} \text{Decode}_\ell : \mathcal{B}^{32\ell} &\longrightarrow \mathcal{R}_q \\ B &\mapsto f_0 + f_1 x + \ldots + f_{n-1} x^{n-1} \\ \text{Encode}_\ell : \mathcal{R}_q &\longrightarrow \mathcal{B}^{32\ell} \\ f_0 + f_1 x + \ldots + f_{n-1} x^{n-1} &\mapsto B \ . \end{aligned}$$

5.2.2. Algoritmo CRYSTALS-KYBER

Para el correcto entendimiento de KYBER se describen cada uno de los sistemas de forma esquemática, el sistema de criptografía simétrica y el sistema de intercambio de

claves. Existen distintas variantes de KYBER en función de los valores que tome un conjunto de parámetros, esto se muestra en la Tabla 5.1.

	n	q	k	η_1	η_2	d_u	d_v	δ	ℓ
KYBER	256	7681	3	4	4	11	3	2^{-142}	13
KYBER512	256	3329	2	3	2	10	4	2^{-139}	12
KYBER768	256	3329	3	2	2	10	4	2^{-164}	12
KYBER1024	256	3329	4	2	2	11	5	2^{-174}	12

Cuadro 5.1: Parámetros de KYBER

$k \in \mathbb{Z}$ es un parámetro que se ajusta para conseguir que la dimensión del retículo involucrado sea múltiplo de n. k es por lo tanto un parámetro para aumentar la seguridad y eficiencia del algoritmo. η_1 y η_2 son enteros empleados por las funciones CBD y que permiten definir el ruido existente entre dos vectores. d_u, d_v son enteros positivos. η_1, η_2, d_u y d_v se definen de forma que se mantenga un equilibrio entre la seguridad, el tamaño del texto cifrado y la probabilidad de error. δ es la probabilidad de error en el descifrado (definida en el Teorema 5.2.1) y ℓ es el parámetro que determina el número de bits que se agrupan en las funciones Encode y Decode.

Criptografía de clave pública

Algoritmo 3: Generación de claves. KYBER.CPAPKE.KeyGen()

Salida: Clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$
Salida: Clave privada $\mathbf{s} \in \mathcal{B}^{12 \cdot k \cdot n/8}$
$d \leftarrow \mathcal{B}^{32}$ ▷ 32 bytes aleatorios para generar ρ y σ
Generar ρ y σ ▷ Función *hash* G para generar ρ y σ de 32 bytes cada uno
Generar $\hat{\mathbf{A}}$ ▷ Matriz A aleatoria a partir de ρ usando XOF
Generar $\mathbf{s}$ ▷ Vector de polinomios $\mathbf{s}$ aleatorio a partir de σ usando PRF y CBD
Generar $\mathbf{e}$ ▷ Vector de polinomios $\mathbf{e}$ aleatorio a partir de σ usando PRF y CBD
$\hat{\mathbf{s}} := \text{NTT}(\mathbf{s})$ ▷ Calcular $\hat{\mathbf{s}}$ mediante NTT
$\hat{\mathbf{e}} := \text{NTT}(\mathbf{e})$ ▷ Calcular $\hat{\mathbf{e}}$ mediante NTT
$\hat{\mathbf{b}} := \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \hat{\mathbf{e}}$
$\mathbf{b} := \text{Encode}_\ell(\hat{\mathbf{b}} \bmod^+ q) \,||\, \rho$ ▷ Clave pública
$\mathbf{s} := \text{Encode}_\ell(\hat{\mathbf{s}} \bmod^+ q)$ ▷ Clave privada
return $\mathbf{b}$, $\mathbf{s}$

El Algoritmo 3 muestra como generar la clave pública y privada. Primeramente se escogen dos elementos aleatorios ρ, σ como resultado de aplicar una función *hash* al elemento d, que sigue la distribución de $\mathcal{B}^{32}$. La matriz $\hat{\mathbf{A}} \in \mathcal{R}_q^{k \times k}$ se genera a partir de ρ y constituye el módulo que define el retículo sobre el que se trabaja. Por su parte, los vectores $\hat{\mathbf{s}} \in \mathcal{R}_q^k$ y $\hat{\mathbf{e}} \in \mathcal{R}_q^k$ se generan a partir del elemento σ constituyendo ambos muestras de $\mathcal{B}_{\eta_1}$. Se calcula además un elemento próximo $\hat{\mathbf{b}} = \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \hat{\mathbf{e}}$ a un elemento del retículo, haciendo uso de la operación de convolución descrita para realizar la multiplicación de polinomios de forma eficiente. De esta forma la clave pública está formada por $\mathbf{b}$ y ρ y la privada es $\mathbf{s}$. Se emplea ρ como parte de la clave pública a partir de la cual se genera $\hat{\mathbf{A}}$ mediante funciones conocidas, en lugar de usar $\hat{\mathbf{A}}$, para poseer así menor tamaño de la clave.

Se observan las similitudes que comparte este algoritmo con el algoritmo LWE descrito en la Sección 5.1. Las funciones Encode y Decode de KYBER tienen su análogo en la función f y su inversa f^{-1} de LWE. Ambas se encarga de reducir y aumentar respectivamente la dimensión del espacio en el cual se trabaja. Por otra parte, las operaciones dentro de cada uno de los retículos para la obtención de la clave pública y privada son semejantes a excepción de que LWE trabaja con vectores enteros y KYBER con polinomios.

Algoritmo 4: Cifrado. KYBER.CPAPKE.Enc(b, m, r)

Entrada: Clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$
Entrada: Mensaje $m \in \mathcal{B}^{32}$
Entrada: Número aleatorio $r \in \mathcal{B}^{32}$
Salida: Texto cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$
$\hat{\mathbf{b}} := \text{Decode}_\ell(\mathbf{b})$ ▷ Clave pública como polinomio
Recuperar ρ ▷ A partir de la posición $\frac{12kn}{8}$ de la clave pública
Generar $\hat{\mathbf{A}}$ ▷ Matriz A aleatoria a partir de ρ usando XOF
Generar $\mathbf{r}$ ▷ Vector $\mathbf{r}$ aleatorio a partir de r usando PRF y CBD
Generar $\mathbf{e}_1$ ▷ Vector $\mathbf{e}_1$ aleatorio a partir de r usando PRF y CBD
Generar e_2 ▷ Escalar e_2 aleatorio a partir de r usando PRF y CBD
$\hat{\mathbf{r}} := \text{NTT}(\mathbf{r})$
$\mathbf{u} := \text{NTT}^{-1}(\hat{\mathbf{A}}^T \circ \hat{\mathbf{r}}) + \mathbf{e_1}$ ▷ Calcular $\mathbf{u}$ mediante convolución
$v := \text{NTT}^{-1}(\hat{\mathbf{b}}^T \circ \hat{\mathbf{r}}) + e_2 + \text{Decompress}_q(\text{Decode}_1(m), 1)$
$c_1 := \text{Encode}_{d_u}(\text{Compress}_q(\mathbf{u}, d_u))$ ▷ Convertir $\mathbf{u}$ en cadena de bytes
$c_2 := \text{Encode}_{d_v}(\text{Compress}_q(v, d_v))$ ▷ Convertir v en cadena de bytes
return $c = (c_1 \| c_2)$ ▷ Concatenar c_1 y c_2 para obtener el texto cifrado

El Algoritmo 4 se encarga de cifrar un mensaje $m \in \mathcal{B}^{32}$ dado a partir de la clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$ y un número aleatorio $r \in \mathcal{B}^{32}$. Se calcula nuevamente la matriz $\hat{\mathbf{A}}$ a partir de ρ. $\mathbf{r} \in \mathcal{R}_q^k$ es un vector aleatorio formado siguiendo la misma distribución de $\mathcal{B}_{\eta_1}$, el vector $\mathbf{e}_1 \in \mathcal{R}_q^k$ sigue la misma distribución de $\mathcal{B}_{\eta_2}$ y el escalar $e_2 \in \mathcal{R}_q$ también es escogiendo siguiendo $\mathcal{B}_{\eta_2}$. Finalmente se genera el vector $\mathbf{u} = \hat{\mathbf{A}}^T\mathbf{r} + \mathbf{e}_1$ para introducir los errores del problema LWE en el cifrado y el mensaje cifrado $v = \mathbf{b}^T\mathbf{r} + e_2 + m$. Concatenando ambos se obtiene el cifrado

$$c = (\text{Compress}_q(\mathbf{u}, d_u), \text{Compress}_q(\mathbf{v}, d_v))$$

Nuevamente, se puede observar que la generación del vector $\mathbf{u}$ en KYBER se equipara con la creación de ese mismo vector para LWE $\mathbf{u} = \mathbf{A}^T\mathbf{a}$ y v con $\mathbf{v} = \mathbf{B}^T\mathbf{a} + f(\mathbf{m})$ de LWE, añadiendo en el caso de KYBER vectores $\mathbf{e}_1$ y $\mathbf{e}_2$ de error para añadir mayor aleatoriedad.

Algoritmo 5: Descifrado. KYBER.CPAPKE.Dec($\mathbf{s}$, c)

Entrada: Clave privada $\mathbf{s} \in \mathcal{B}^{12 \cdot k \cdot n/8}$
Entrada: Texto cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$
Salida: Mensaje $m \in \mathcal{B}^{32}$
Recuperar $\mathbf{u}$ ▷ Calcular $\mathbf{u}$ mediante las funciones Decompress y Decode
Recuperar v ▷ Calcular v mediante las funciones Decompress y Decode
$\mathbf{s} := \text{Decode}_\ell(\mathbf{s})$
$m := \text{Encode}_1(\text{Compress}_q(v - \text{NTT}^{-1}(\mathbf{s}^T \circ \text{NTT}(\mathbf{u})), 1))$ ▷ Recuperar el mensaje
return m

El Algoritmo 5 muestra el proceso de descifrado de un mensaje cifrado c para obtener el mensaje original $m \in \mathcal{B}^{32}$. Es necesario además el empleo de la clave privada $\mathbf{s} \in \mathcal{B}^{12 \cdot k \cdot n/8}$. Se emplea la función Decompress para recuperar el mensaje cifrado v. Posteriormente el mensaje en claro se obtiene a partir de la función Compress, $\text{Compress}_q(v - \mathbf{s}^T\mathbf{u}, 1)) = m$.

Comparándolo con LWE, el descifrado consiste igualmente en recuperar los elementos $(\mathbf{u}, \mathbf{v})$. De forma semejante, $\text{Compress}_q(v - \mathbf{s}^T\mathbf{u}, 1))$ en KYBER es semejante a la función de LWE $f^{-1}(\mathbf{v} - \mathbf{s}^T\mathbf{u})$. Con estas definiciones de los algoritmos, este sistema es $(1-\delta)$-correcto como muestra el Teorema 5.2.1 y cuya demostración se puede encontrar en [6].

Teorema 5.2.1. Sea $k \in \mathbb{Z}^+$. Sean $\mathbf{s}, \mathbf{e}, \mathbf{r}, \mathbf{e}_1$ y e_2 variables aleatorias que tienen la misma distribución que la que tienen esas variables en los Algoritmos 3 y 4. Sea $\mathbf{c}_t \leftarrow \Psi_{d_t}^k$, $\mathbf{c}_u \leftarrow \Psi_{d_u}^k$ y $\mathbf{c}_v \leftarrow \Psi_{d_v}$. Ψ_d^k es la distribución sobre $\mathcal{R}$ que es aquella que a partir de un elemento $\mathbf{y} \in \mathcal{R}$ uniformemente aleatorio se devuelve $(\mathbf{y} - \text{Decompress}_q(\text{Compress}_q(\mathbf{y}, d), d))\ \ mod^{\pm}\ q$. Por otra parte se denota

$$\delta = Pr\left[||\mathbf{e}^T\mathbf{r} + e_2 + \mathbf{c}_v - \mathbf{s}^T\mathbf{e}_1 + \mathbf{c}_t^T\mathbf{r} - \mathbf{s}^T\mathbf{c}_u||_\infty \geq \left\lceil \frac{q}{4} \right\rfloor \right] .$$

Entonces KYBER.CPAPKE es (1-δ)-correcto.

Ejemplo 5.2.1. Se muestra un ejemplo simplificado de este problema empleando polinomios sencillos. Sirve para evidenciar cómo funciona este sistema. El parámetro q toma el valor $q = 17$ y $n = 4$. Se trabaja en el anillo de polinomios $\mathcal{R}_{17}[x]/(x^4 + 1)$. La matriz $\hat{\mathbf{A}}$ se genera de forma aleatoria, por ejemplo

$$\hat{\mathbf{A}} = \begin{bmatrix} 2x^3 + 15x^2 + 7x + 1 & 5x^3 + 5x^2 + 10 \\ 3x^3 + 8x^2 + 5x + 4 & 9x^3 + 8x^2 + 2x + 4 \end{bmatrix} ,$$

mientras que los vectores de polinomios $\mathbf{s}$ y $\mathbf{e}$ aleatorios son

$$\mathbf{s} = \begin{bmatrix} x^2 - x - 1 \\ x^3 + x + 1 \end{bmatrix} \quad \mathbf{e} = \begin{bmatrix} x + 1 \\ -x^3 + x \end{bmatrix} .$$

Al efectuar la operación $\hat{\mathbf{A}} \circ \mathbf{s} + \mathbf{e}$ se obtiene el vector

$$\hat{\mathbf{b}} = \begin{bmatrix} 12x^3 + 13x^2 + 15x + 7 \\ 14x^3 + 9x^2 + 4x + 1 \end{bmatrix}$$

que forma parte de la clave pública junto con la matriz $\hat{\mathbf{A}}$, $(\hat{\mathbf{b}}, \hat{\mathbf{A}})$. Para ser precisos, la clave pública está formada por $(\hat{\mathbf{b}}, \rho)$ para reducir el tamaño de la clave, puesto que $\hat{\mathbf{A}}$ se deriva de ρ. Pero para evitar añadir complejidad al ejemplo se asume ese abuso de notación. Por otro lado, la clave privada está compuesta por $\mathbf{s}$. De esta forma la seguridad se basa en la dificultad de obtener $\mathbf{s}$ a partir de $(\hat{\mathbf{b}}, \hat{\mathbf{A}})$. Sería necesario resolver el problema MLWE. Para el algoritmo de cifrado, se desea cifrar el mensaje cuya representación binaria es (1100) y por tanto la representación polinomial es $m = x^3 + x^2$. Para llevar a cabo el proceso de cifrado es necesario contar con los vectores de polinomios

$$\mathbf{r} = \begin{bmatrix} x^3 - x^2 + x \\ x^3 - 1 \end{bmatrix} \quad \mathbf{e}_1 = \begin{bmatrix} -x^3 - 1 \\ x^3 - x \end{bmatrix}$$

y el escalar $e_2 = -x^3 - x^2 - x$. Con esto se calcula por tanto

$$\mathbf{u} = \hat{\mathbf{A}}^T\mathbf{r} + \mathbf{e}_1 = \begin{bmatrix} 9x^3 + 12x^2 + 7x + 15 \\ 11x^3 + 2x^2 - x + 11 \end{bmatrix}$$

y $v = \mathbf{b}^T\mathbf{r} + e_2 + m = 16x^3 + 14x^2 + 9x + 15$. De esta forma el mensaje cifrado es el par $(\mathbf{u}, v)$. Ahora bien, si se quiere realizar el proceso contrario y obtener el mensaje en claro a partir del mensaje cifrado, es necesario aplicar el algoritmo de descifrado. El mensaje se calcula como

$$m = v - \mathbf{s}^T\mathbf{u} = 9x^3 + 9x^2 = x^3 + x^2 \ ,$$

coincidiendo efectivamente con el mensaje inicial.

Intercambio de claves

Un sistema de intercambio de claves (KEM) es un mecanismo que permite la transmisión de una clave simétrica haciendo uso de un sistema de criptografía de clave pública. De esta forma se conserva la seguridad propia de la criptografía de clave pública y se aprovecha la rapidez que proporciona la criptografía simétrica al poseer longitudes de claves menores. El algoritmo KYBER de intercambio de claves (KYBER.CCAKEM) se construye a partir de los algoritmos de criptografía simétrica explicados en la Subsección 5.2.2 tras aplicar la transformada de Fujisaki-Okamoto ligeramente modificada. A continuación se detalla cada uno de los procesos involucrados.

Algoritmo 6: Generación de claves. KYBER.CCAKEM.KeyGen()

Salida: Clave privada $\mathbf{s} \in \mathcal{B}^{24\cdot k\cdot n/8+96}$
Salida: Clave pública $\mathbf{b} \in \mathcal{B}^{12\cdot k\cdot n/8+32}$
$z \leftarrow \mathcal{B}^{32}$ ▷ 32 bytes aleatorios
$(\mathbf{b}, \mathbf{s}) \leftarrow$ KYBER.CPAPKE.KeyGen() ▷ Generar **b** y **s**
return **b**, **s**, H(**b**), z

El Algoritmo 6 genera las claves pública y privada. Muestra como salida las claves obtenidas en el Algoritmo 3 con el *hash* de la clave pública $H(\mathbf{b})$ y un valor aleatorio $z \in \mathcal{B}^{32}$. De esta forma otorga al sistema seguridad y robustez adicional.

Algoritmo 7: Cifrado. KYBER.CCAKEM.Enc(**s**)

Entrada: Clave pública $\mathbf{b} \in \mathcal{B}^{12\cdot k\cdot n/8+32}$
Salida: Cifrado $c \in \mathcal{B}^{d_u\cdot k\cdot n/8+d_v\cdot n/8}$
Salida: Clave compartida simétrica $K \in \mathcal{B}^*$
Generar m ▷ *Hash* de un mensaje de 32 bytes aleatorio
Generar (K', r) ▷ Calcular K' y r mediante una función *hash* G
$c \leftarrow$ KYBER.CPAPKE.Enc(**b**, m, r) ▷ Obtener mensaje cifrado
Generar clave K ▷ Derivar la clave compartida empleando KDF
return c, K

Los Algoritmos 7, 8 emplean como base los algoritmos de cifrado y descifrado para encapsular y desencapsular las claves. El algoritmo de cifrado genera un vector aleatorio m siguiendo la distribución $\mathcal{B}^{32}$ y lo cifra empleando la clave pública $\mathbf{b} \in \mathcal{B}^{12\cdot k\cdot n/8+32}$ para generar una clave compartida $K \in \mathcal{B}^*$. Mientras que el algoritmo de descifrado obtiene m' a partir del cifrado anterior c. Posteriormente genera un cifrado de comprobación c' para en función de si coincide con el cifrado c obtener la clave compartida K o una clave alternativa. Esta última comprobación de desencapsular la clave debe ser realizada tanto por el remitente como por el destinatario, para que ambos posean la misma clave para la comunicación.

De esta forma, si se desea establecer una comunicación con un usuario A, el Algoritmo

Algoritmo 8: Descifrado. KYBER.CCAKEM.Dec(c, **s**, **b**, H(**b**), z)

Entrada: Cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$
Entrada: Clave privada **s**
Entrada: Clave pública **b**
Entrada: *Hash* de 32 bytes de la clave pública $h = H(\mathbf{b}) \in \mathcal{B}^{32}$
Entrada: Polinomio aleatorio z
Salida: Clave compartida $K \in \mathcal{B}^*$
$m' \leftarrow$ KYBER.CPAPKE.Dec($\mathbf{s}, c$) ▷ Descifrar el mensaje
Generar (K'', r) ▷ Calcular K" y r mediante una función *hash* G
$c' \leftarrow$ KYBER.CPAPKE.Enc($\mathbf{b}, m', r'$) ▷ Cifrar el mensaje obtenido
si $c = c'$ **entonces**
  Generar clave K ▷ Derivar la clave compartida empleando KDF si c coincide con c'
si no
  Generar clave K ▷ Derivar la clave compartida alternativa empleando KDF y el polinomio aleatorio z si c no coincide con c'
fin si
  return K

6 de generación de claves genera una clave pública y una clave privada. La clave privada únicamente la conoce el usuario A, mientras que la clave pública es conocida por cualquier usuario. Con la clave pública, un usuario B que desee comunicarse con A, cifrará con el Algoritmo 7 la clave simétrica K y se la transmitirá a este usuario. Por su parte, el usuario A, en posesión de la clave privada aplicará el Algoritmo 8 de descifrado para conseguir la clave simétrica compartida y así emplear un sistema simétrico de comunicación con el usuario B.

5.2.3. Seguridad CRYSTALS-KYBER

La seguridad de KYBER radica en la estructura MLWE empleada. Existe una cadena de reducciones de forma que si un problema A se reduce a B ($A \implies B$), significa que B es un problema más duro que A, puesto que si B se resuelve, se resuelve también A. De esta forma se encuentra la cadena

$$\text{RLWE} \implies \text{MLWE} \implies \text{LWE} \implies \text{SVP} \implies \text{CVP}$$

RLWE $\implies$ MLWE es cierto puesto que el problema RLWE es un caso particular de módulo-LWE, es decir es un MLWE de rango 1. MLWE posee mayor estructura algebraica que LWE, luego es más sencillo vulnerar MLWE que LWE, es decir MLWE $\implies$ LWE. Por otra parte existe un algoritmo denominado BZK que reduce LWE a resolver una cantidad polinomial de problemas de tipo CVP (LWE $\implies$ SVP). Por último SVP $\implies$ CVP es cierto puesto que SVP se reduce a la resolución de una serie de problemas CVP.

5.3. Conclusiones

Tras examinar detalladamente el funcionamiento de KYBER se puede concluir que es semejante al del sistema criptográfico basado en LWE. La principal diferencia radica en la eficiencia. KYBER al estar basado en MLWE posee mayor estructura algebraica, así que

en consecuencia posee mayor eficiencia. Pero poseer mayor estructura algebraica está a su vez ligado a una menor seguridad del sistema.

Actualmente la criptografía basada en retículos se conforma como candidata para emplearse en los algoritmos criptográficos frente a la llegada de los ordenadores cuánticos. Ambos campos, tanto la computación cuántica como la criptografía basada en retículos, son campos en constante avance. Es por eso que la seguridad de los sistemas criptográficos de clave pública está en continua revisión, ya que en cualquier momento se puede encontrar un nuevo algoritmo que haga vulnerable un cifrado que hasta entonces se consideraba seguro. En Abril de 2024 Yilei Chen [7] muestra un posible algoritmo cuántico mediante el cual se reduciría el problema LWE a complejidad polinomial. Este resultado se encuentra aún en proceso de revisión, pero sirve para evidenciar la creciente necesidad de estandarizar sistemas criptográficos seguros frente a un ordenador cuántico.

Apéndice A

Pseudocódigo CRYSTALS-KYBER

Este Apéndice muestra los algoritmos principales involucrados en el algoritmo KYBER con mayor nivel de detalle. Por una parte, se encuentran los algoritmos auxiliares Parse, CBD y Decode. Seguidamente se detallan los algoritmos de criptografía de clave pública y de intercambio de claves.

A.1. Conceptos previos

A.1.1. Distribuciones de probabilidad

Algoritmo 9: Parse: $\mathcal{B} \longrightarrow \mathcal{R}_q$

Data: Vector de bytes $\mathbf{B} = (b_0, b_1, \ldots) \in \mathcal{B}^*$
Result: Representación NTT $\hat{a} \in \mathcal{R}_q$ de $a \in \mathcal{R}_q$

$i := 0$
$j = 0$
while $j < n$ **do**
  $d_1 := b_i + 256 \cdot (b_{i+1} \ \ mod^+ \ 16)$
  $d_2 := \lfloor b_{i+1}/16 \rfloor + 16 \cdot b_{i+2}$
  if $d_1 < q$ **then**
    $\hat{a}_j := d_1$
    $j := j + 1$
  if $d_2 < q$ **then**
    $\hat{a}_j := d_2$
    $j := j + 1$
  $i := i + 3$

return $\hat{a} = \hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{n-1} x^{n-1}$

Algoritmo 10: CBD: $\mathcal{B}^{64\eta} \longrightarrow \mathcal{R}_q$

Data: Vector $\mathbf{b} = (b_0, b_1, \ldots, b_{64\eta-1}) \in \mathcal{B}^{64\eta}$
Result: Polinomio $f \in \mathcal{R}_q$

$(\beta_0, \ldots, \beta_{512\eta-1} :=$ BytesToBits(B)
for *i from* 0 *to* $n-1$ **do**
  $a := \sum_{j=0}^{\eta-1} \beta_{2i\eta+j}$
  $b := \sum_{j=0}^{\eta-1} \beta_{2i\eta+\eta+j}$
  $f_i := a - b$

return $f = f_0 + f_1 x + \ldots + f_{n-1} x^{n-1}$

A.1.2. Codificación y decodificación

Algoritmo 11: Decode: $\mathcal{B}^{64\eta} \longrightarrow \mathcal{R}_q$

Data: $B \in \mathcal{B}^{32\ell}$
Result: Polinomio $f \in \mathcal{R}_q$

$(\beta_0, \ldots, \beta_{256\ell-1} :=$ BytesToBits(B)
for *i from* 0 *to* $n-1$ **do**
  $f_i := \sum_{j=0}^{\ell-1} \beta_{i\ell+j} 2^j$

return $f = f_0 + f_1 x + \ldots + f_{n-1} x^{n-1}$

A.2. Algoritmo CRYSTALS-KYBER

A.2.1. Criptografía de clave pública

Algoritmo 12: Generación de claves. KYBER.CPAPKE.KeyGen()

Result: Clave privada $\mathbf{s} \in \mathcal{B}^{12 \cdot k \cdot n/8}$
Clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$

d $\longleftarrow \mathcal{B}^{32}$
$(\rho, \sigma) := G(d)$
$N := 0$
for *i from 0 to k-1* **do**
  for *j from 0 to* $k-1$ **do**
    $\hat{\mathbf{A}}[i][j] := Parse(\text{XOF}(\rho, j, i))$
for *i from 0 to k-1* **do**
  $\mathbf{s}[i] := \text{CBD}_{n1}(\text{PRF}(\sigma, N))$
  $N := N + 1$
for *i from 0 to k-1* **do**
  $\mathbf{e}[i] := \text{CBD}_{n1}(\text{PRF}(\sigma, N))$
  $N := N + 1$

$\hat{\mathbf{s}} := NTT(\mathbf{s})$
$\hat{\mathbf{e}} := NTT(\mathbf{e})$
$\hat{\mathbf{b}} := \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \hat{\mathbf{e}}$

$\mathbf{b} := \text{Encode}_{12}(\hat{\mathbf{b}} \; mod^+ \; q) \; || \; \rho$
$\mathbf{s} := \text{Encode}_{12}(\hat{\mathbf{s}} \; mod^+ \; q)$

return **b**, **s**

Algoritmo 13: Cifrado. KYBER.CPAPKE.Enc(**b**, m, r)

Data: Clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$
Mensaje $m \in \mathcal{B}^{32}$
Número aleatorio $r \in \mathcal{B}^{32}$
Result: Texto cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$

$\hat{\mathbf{b}} := \text{Decode}_{12}(\mathbf{b})$
$\rho := \mathbf{b} + 12 \cdot k \cdot n/8$
$N := 0$
for *i from 0 to k-1* **do**
  for *j from 0 to* $k-1$ **do**
    $\hat{\mathbf{A}}^T[i][j] := Parse(\text{XOF}(\rho, i, j))$

for *i from 0 to k-1* **do**
  $\mathbf{r}[i] := \text{CBD}_{n1}(\text{PRF}(r, N))$
  $N := N + 1$

for *i from 0 to k-1* **do**
  $\mathbf{e_1}[i] := \text{CBD}_{n2}(\text{PRF}(r, N))$
  $N := N + 1$

$e_2 := \text{CBD}_{n2}(\text{PRF}(r, N))$
$\hat{\mathbf{r}} := \text{NTT}(\mathbf{r})$
$\mathbf{u} := \text{NTT}^{-1}(\hat{\mathbf{A}}^T \circ \hat{\mathbf{r}}) + \hat{\mathbf{e_1}}$
$v := \text{NTT}^{-1}(\hat{\mathbf{b}}^T \circ \hat{\mathbf{r}}) + e_2 + \text{Decompress}_q(\text{Decode}_1(m), 1)$

$c\_1 := \text{Encode}_{d_u}(\text{Compress}_q(\mathbf{u}, d_u))$
$c\_2 := \text{Encode}_{d_v}(\text{Compress}_q(\mathbf{v}, d_v))$

return c $= (c_1 || c_2)$

Algoritmo 14: Descifrado. KYBER.CPAPKE.Dec(**s**, c)

Data: Clave privada $\mathbf{s} \in \mathcal{B}^{12 \cdot k \cdot n/8}$
Texto cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$
Result: Mensaje $m \in \mathcal{B}^{32}$

$\boldsymbol{u} := \text{Decompress}_q(\text{Decode}_{d_u}(c), d_u)$
$v := \text{Decompress}_q(\text{Decode}_{d_v}(c + d_u \cdot k \cdot n/8), d_v)$

$\hat{\mathbf{s}} := \text{Decode}_{12}(\mathbf{s})$
$m := \text{Encode}_1(\text{Compress}_q(v - \text{NTT}^{-1}(\hat{\mathbf{s}}^T \circ \text{NTT}(\mathbf{u})), 1))$

return m

A.2.2. Intercambio de claves

Algoritmo 15: Generación de claves. KYBER.CCAKEM.KeyGen()

Result: Clave privada $\mathbf{s} \in \mathcal{B}^{24 \cdot k \cdot n/8+96}$
Clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$

$z \longleftarrow \mathcal{B}^{32}$
$(\mathbf{b}, \mathbf{s}') :=$ KYBER.CPAPKE.KeyGen()
$\mathbf{s} := (\mathbf{s}' \,||\, \mathbf{b} \,||\, H(\mathbf{b}) \,||\, z)$

return b, **s**

Algoritmo 16: Cifrado. KYBER.CCAKEM.Enc($\mathbf{b}$)

Data: Clave pública $\mathbf{b} \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$
Result: Cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8+d_v \cdot n/8}$
Clave compartida $K \in \mathcal{B}^*$

$m \longleftarrow \mathcal{B}^{32}$
$m \longleftarrow H(m)$
$(K', r) := G(m\ H(\mathbf{b}))$
$c :=$ KYBER.CPAPKE.Enc(**b**,m,r)
$K := \text{KDF}(K'||H(c))$

return c, K

Algoritmo 17: Descifrado. KYBER.CCAKEM.Dec(c, $\mathbf{s}$)

Data: Cifrado $c \in \mathcal{B}^{d_u \cdot k \cdot n/8+d_v \cdot n/8}$
Clave privada $\mathbf{s} \in \mathcal{B}^{24 \cdot k \cdot n/8+96}$
Result: Clave compartida $K \in \mathcal{B}^*$

$\mathbf{b} := \mathbf{s} + 12 \cdot k \cdot n/8$
$h := \mathbf{s} + 24 \cdot k \cdot n/8 + 32 \in \mathcal{B}^{32}$
$z := \mathbf{s} + 24 \cdot k \cdot n/8 + 64$
$m' :=$ KYBER.CPAPKE.Dec($\mathbf{s}, c$)
$(K'', r') := G(m' \,||\, h)$
$c' :=$ KYBER.CPAPKE.Enc($\mathbf{b}, m', r'$)
if $c = c'$ **then**
  return $K := \text{KDF}(K'' \,||\, H(c))$
else
  return $K := \text{KDF}(z \,||\, H(c))$

return K

Bibliografía

[1] R.C. Agarwal y C.S. Burrus. «Number theoretic transforms to implement fast digital convolution». En: *Proceedings of the IEEE* 63.4 (1975), págs. 550-560.

[2] Roberto Avanzi et al. *CRYSTALS-Kyber. Algorithm Specifications And Supporting Documentation*. Inf. téc. 3.01. NIST, 2021.

[3] M. Bellare et al. «Relations Among Notions of Security for Public-Key Encryption Schemes». En: *Advances in Cryptology. CRYPTO* 1462 (1998), págs. 26-46.

[4] Daniel J. Bernstein, Johannes Buchmann y Erik Dahmen. *Post-Quantum Cryptography*. 1st. Springer, 2009.

[5] R. J. Lipton Boneh. «Quantum cryptanalysis of hidden linear functions». En: *Advances in Cryptology—CRYPTO '95, 15th Annual International Cryptology Conference* (1995), págs. 424-437.

[6] Joppe Bos et al. «CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM». En: *Cryptology ePrint Archive* (abr. de 2018).

[7] Yilei Chen. «Quantum Algorithms for Lattice Problems». En: *Cryptology ePrint Archive, Paper 2024/555* (2024).

[8] Whitfield Diffie y Martin E. Hellman. «New Directions in Cryptography». En: *IEEE Transactions on information theory* IT-22, NO. 6 (November 1976), págs. 644-654.

[9] Taher ElGamal. «A public key cryptosystem and a signature scheme based on discrete logarithms». En: *IEEE Transactions on Information Theory* 31 (1985), págs. 469-472.

[10] Neal Koblitz. *A Course in Number Theory and Cryptography*. 2nd. Springer-Verlag, 1994.

[11] A. K. Lenstra, H. W. Lenstra Jr. y L. Lovász. «Factoring polynomials with rational coefficients». En: *Mathematische Annalen* 261 (1982), págs. 515-534.

[12] Michael Loceff. *Una introducción a la criptografía de clave pública*. 2.ª ed. Ediciones Uninorte, 2010.

[13] Patrick Longa y Michael Naehrig. «Speeding up the Number Theoretic Transform for Faster Ideal Lattice-Based Cryptography». En: *Cryptology and Network Security*. Springer International Publishing, 2016, págs. 124-139.

[14] Daniele Micciancio y Shafi Goldwasser. *Complexity of Lattice Problems*. 1st. Springer Science + Business Media, 2002.

[15] M. Nielsen e I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[16] Oded Regev. «On lattices, learning with errors, random linear codes, and cryptography». En: *Journal of the ACM* 56 (2009), págs. 1-40.

[17] R.L. Rivest, A. Shamir y L. Adleman. «A Method for Obtaining Digital Signatures and Public-Key Cryptosystems». En: *Communications of the ACM* 21 (February 1978), págs. 120-126.

[18] Peter W. Shor. «Algorithms for Quantum Computation: Discrete Logarithms and Factoring». En: *Communications of the ACM* 46 (1994), págs. 124-134.

[19] Josheph H. Silverman. *The Arithmetic of Elliptic Curves*. 2nd. Springer, 1986.

[20] Lawrence C. Washington. *Elliptic curves: Number theory and Cryptography*. 2nd. Chapman y Hall, 2008.