UNIVERSIDAD ALFONSO X EL SABIO

FACULTAD DE BUSINESS & TECH

GRADO EN INGENIERÍA MATEMÁTICA

TRABAJO DE FIN DE GRADO

Seguridad en criptografía postcuántica:
Análisis de Learning With Errors e implementación de esquemas seguros

Alejandro Martínez Ronda

Junio de 2026

UNIVERSIDAD ALFONSO X EL SABIO

Facultad de Business & Tech

Grado en Ingeniería Matemática

Seguridad en criptografía postcuántica

Análisis de Learning With Errors e implementación de esquemas seguros

ALUMNO: Alejandro Martínez Ronda

N.P.: 141681

TUTOR ACADÉMICO: Jorge Calvo Martín

FECHA DE PRESENTACIÓN: Junio de 2026

BREVE DESCRIPCIÓN:

Este Trabajo de Fin de Grado analiza los fundamentos matemáticos y computacionales de la criptografía postcuántica, con especial atención a los problemas basados en retículos y al problema Learning With Errors. El trabajo parte de la vulnerabilidad de los sistemas clásicos de clave pública, como RSA y ECC, frente al modelo de computación cuántica y al algoritmo de Shor. A partir de ahí, se introducen los retículos como estructuras algebraico-geométricas, los problemas computacionales asociados y la formulación de LWE como base de esquemas criptográficos resistentes al paradigma cuántico. Finalmente, se desarrolla una implementación experimental simplificada para estudiar el comportamiento del descifrado bajo distintos parámetros y visualizar cómo afectan el módulo, la dimensión, el error y el número de muestras a la corrección del esquema.

Firma del director del trabajo

Jorge Calvo Martín

Firma del alumno

Índice

1. Introducción

1.1. Contexto: evolución de la criptografía

La historia de la criptografía es, en un sentido muy preciso, la historia de la lucha por el control de la información. Desde sus orígenes, la criptografía ha sido una herramienta para garantizar la confidencialidad de la información. Tradicionalmente, su uso se asociaba a contextos militares o diplomáticos, donde el objetivo principal era impedir que un tercero pudiera comprender un mensaje interceptado.

En estos primeros sistemas, la seguridad dependía en gran medida del secreto del propio método de cifrado, lo que se conoce como "seguridad por oscuridad". Dichos sistemas utilizaban una clave compartida previamente entre emisor y receptor, característica propia de la criptografía simétrica.

La situación cambió de forma significativa en la década de 1970 con la aparición de la criptografía de clave pública, que permitió establecer comunicaciones seguras sin necesidad de compartir una clave secreta de antemano. Este avance supuso un punto de inflexión, al hacer posible el desarrollo de sistemas de comunicación abiertos y escalables.

Con el desarrollo de la informática y, especialmente, con la expansión de internet, la criptografía ha pasado a desempeñar un papel central en la vida cotidiana. Actualmente se emplea de forma masiva en comunicaciones digitales, transacciones bancarias, almacenamiento de datos y sistemas de autenticación. En este nuevo contexto, ya no es suficiente con ocultar el funcionamiento del sistema: la seguridad debe fundamentarse en propiedades matemáticas verificables.

Este cambio dio lugar a la criptografía moderna, en la que los sistemas se diseñan de forma que su robustez no dependa del secreto del algoritmo, sino de la dificultad computacional de ciertos problemas matemáticos. Es decir, aunque un atacante conozca completamente el funcionamiento del sistema, no debería ser capaz de comprometerlo en un tiempo razonable.

Formalmente, esto se traduce en que un sistema criptográfico se considera seguro cuando el problema matemático subyacente es computacionalmente intratable, en el sentido de que no existe un algoritmo eficiente (en tiempo polinómico) capaz de resolverlo.

En consecuencia, la seguridad de gran parte de la infraestructura digital actual descansa sobre hipótesis de dificultad computacional: sobre suposiciones de complejidad cuya validez se asume en el modelo de computación clásico. Esta circunstancia constituye un punto crítico, ya que la aparición de nuevos modelos de computación, en particular la computación cuántica, podría alterar de forma sustancial dichas hipótesis. El impacto de esta posible ruptura no se limitaría a usuarios individuales, sino que afectaría también a infraestructuras críticas como sistemas financieros, redes de comunicación o servicios gubernamentales, cuya estabilidad depende directamente de estos mecanismos criptográficos.

En los siguientes apartados se analizará con mayor profundidad esta dependencia, así como las propuestas actuales orientadas a garantizar la seguridad de los sistemas criptográficos en este nuevo contexto.

1.2. Criptografía actual: RSA y ECC

Esta idea no es meramente abstracta, se concreta en problemas matemáticos muy específicos cuya resolución resulta sencilla en una dirección, pero extraordinariamente compleja en la inversa. Entre ellos destacan dos algoritmos que han marcado la criptografía de las últimas décadas: la factorización de enteros y el logaritmo discreto.

RSA y la factorización de enteros

El sistema RSA, propuesto por Rivest, Shamir y Adleman en 1978 [10], constituye quizá el ejemplo más emblemático de esta asimetría computacional. Su construcción se fundamenta en una propiedad elemental de la aritmética: multiplicar números primos es computacionalmente sencillo, mientras que deshacer ese producto puede no serlo en absoluto.

En su simplicidad, se eligen dos números primos grandes —típicamente de 1024 bits cada uno en implementaciones de 2048 bits— y se construye el entero

$$N = pq.$$

Obtener N a partir de p y q es inmediato incluso cuando los números son enormes. Sin embargo, el problema inverso —determinar los factores primos de N conociendo únicamente el número compuesto— es el denominado problema de *factorización entera*.

Formalmente, dado un entero compuesto N, el objetivo es encontrar una descomposición no trivial en factores primos,

$$N = p_1^{e_1} \cdots p_k^{e_k},$$

siendo los p_i números primos. En el contexto de RSA, N se construye específicamente como producto de dos primos grandes de tamaño similar, situación para la cual los algoritmos clásicos conocidos presentan una complejidad que crece rápidamente con el tamaño del módulo.

La seguridad de RSA no está demostrada en un sentido absoluto; más bien, se apoya en una hipótesis de complejidad ampliamente aceptada: que no existe, en el modelo clásico de computación, ningún algoritmo eficiente —esto es, de tiempo polinómico en $\log N$— capaz de factorizar enteros de tamaño criptográfico. Esa asimetría estructural entre la facilidad de realizar la operación directa y la dificultad computacional de invertirla constituye, en definitiva, el fundamento del esquema.

Criptografía de curva elíptica y el logaritmo discreto

En la criptografía de curva elíptica el escenario cambia, aunque el principio subyacente permanece. Ya no trabajamos con productos de enteros, sino con puntos que satisfacen una ecuación algebraica.

Una curva elíptica sobre un cuerpo finito $\mathbb{F}_q$ puede describirse, en términos elementales, como el conjunto de soluciones de una ecuación del tipo

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

donde las coordenadas se toman dentro de un conjunto finito de elementos y los coeficientes satisfacen ciertas condiciones que garantizan que la curva no presente singularidades. Esta construcción posee una propiedad decisiva: es posible definir sobre ella una operación de suma que convierte dichos puntos —junto con un punto especial denominado punto en el infinito— en un grupo abeliano finito.

Aquí la operación directa consiste en calcular múltiplos de un punto. Dado P y un entero k, obtener

$$Q = kP,$$

es decir, el resultado de sumar el punto P consigo mismo k veces mediante algoritmos de duplicación y suma cuya complejidad crece de manera polinómica con el tamaño del parámetro.

El interés criptográfico aparece al considerar el proceso inverso. Conocidos los puntos P y $Q = kP$, determinar el entero k constituye el denominado *problema del logaritmo discreto en curvas elípticas (ECDLP)*. Y, al igual que en el caso de la factorización, la seguridad no se basa en una imposibilidad demostrada, sino en una hipótesis de dureza: hasta la fecha, no se conoce ningún algoritmo clásico eficiente —esto es, de tiempo polinómico respecto a $\log q$— que permita resolver el ECDLP para parámetros criptográficamente adecuados.

Así, aunque el entorno algebraico sea muy distinto del de RSA, la idea rectora es la misma: una operación directa computacionalmente accesible frente a una inversión que, al menos en el modelo clásico, parece resistirse sistemáticamente a todo intento eficiente.

En última instancia, la factorización entera y el logaritmo discreto representan problemas cuya presumida intratabilidad en el modelo clásico sostiene la seguridad de la criptografía de clave pública actual.

1.3. Amenaza de la computación cuántica

La criptografía de clave pública moderna, tal como se utiliza en los sistemas introducidos anteriormente, presupone un modelo concreto de computación: el modelo clásico. En él, los algoritmos operan sobre bits que adoptan valores bien definidos —0 o 1— y cuya

evolución se describe mediante transformaciones deterministas o probabilísticas.

Sin embargo, este no es el único paradigma posible.

La computación cuántica define un modelo de computación distinto, apoyándose en los principios fundamentales de la mecánica cuántica. En lugar de bits clásicos, emplea *qubits*, sistemas físicos que pueden describirse como combinaciones lineales de los estados básicos $|0\rangle$ y $|1\rangle$. Un qubit puede encontrarse en un estado de la forma

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

donde α y β son números complejos que satisfacen $|\alpha|^2 + |\beta|^2 = 1$.

La superposición no debe entenderse únicamente como "estar en dos estados a la vez". Esa intuición es útil, pero superficial.

En un sistema clásico, n bits se encuentran en una única configuración entre 2^n posibles en cada instante. Un sistema de n qubits, en cambio, se describe mediante 2^n coeficientes que evolucionan conjuntamente. Este número crece de forma exponencial. Por ello, una operación cuántica no actúa sobre una configuración aislada, sino que transforma simultáneamente todos esos coeficientes, combinándolos entre sí.

No se manipulan estados individuales, sino una estructura global. Y es precisamente esta estructura la que permite transformaciones que no pueden reproducirse de forma eficiente en el modelo clásico.

A ello se añade el entrelazamiento cuántico. Dos qubits entrelazados no pueden describirse por separado, como estados independientes; ya que la información relevante reside en las correlaciones entre ellos. Estas correlaciones no clásicas permiten establecer dependencias globales entre variables, que algunos algoritmos cuánticos explotan para reducir el coste computacional de ciertos problemas.

El resultado es un modelo de computación que, para ciertos problemas, presenta órdenes de complejidad significativamente menores a las alcanzables en el paradigma clásico.

La dificultad sobre la que se sustenta RSA o la criptografía de curva elíptica no desaparece por arte de magia; lo que cambia es el modelo bajo el cual se analiza dicha dificultad. Es precisamente en este punto donde aparece el algoritmo de Shor, que transforma esta posibilidad teórica en una amenaza concreta.

1.4. Algoritmo de Shor

Propuesto por Peter Shor en 1994, este algoritmo constituye uno de los primeros indicios teóricos de que los ordenadores cuánticos pueden ofrecer ventajas exponenciales frente a los clásicos en problemas con estructura algebraica [11]. Como resultado, un ordenador cuántico ideal puede resolver en tiempo polinómico problemas como la factorización de enteros y el logaritmo discreto.

Para entender su impacto, no es necesario analizar los detalles del algoritmo, sino identificar el cambio de enfoque que introduce.

En el caso de la factorización, Shor no ataca la recuperación de p y q a partir del módulo $N = pq$ directamente. El punto de partida es una observación aritmética: si a es coprimo con N, la función

$$f(x) = a^x \text{ mód } N$$

es periódica. Conocer su período r permite (esto es, el número de pasos tras los cuales los valores de a^x mód N comienzan a repetirse; formalmente, el menor entero positivo r tal que $a^r \equiv 1 \pmod{N}$), con alta probabilidad, recuperar un factor no trivial de N. El problema central pasa a ser determinar el período de dicha función.

En el modelo clásico, este problema no admite algoritmos eficientes conocidos. En el modelo cuántico, en cambio, la estructura del problema puede explotarse: la Transformada de Fourier cuántica puede detectar la periodicidad oculta de f operando sobre una superposición de todos los valores de la función de forma simultánea. El resultado es un algoritmo cuya complejidad es polinómica en $\log N$, frente a la complejidad subexponencial del mejor algoritmo clásico conocido para factorización —el cribado del cuerpo de números—, que crece mucho más rápido con el tamaño del módulo.

Este mismo esquema se extiende al logaritmo discreto. En lugar de trabajar con enteros, se opera en grupos finitos —incluidos los definidos sobre curvas elípticas—, pero la estructura del problema permite una reformulación similar, en la que la información buscada queda codificada en la periodicidad de una función adecuada, la Transformada de Fourier cuántica resuelve igualmente ese problema en tiempo polinómico en $\log q$.

En este escenario, los problemas que sustentan RSA y la criptografía de curva elíptica dejan de ser computacionalmente intratables bajo este modelo. No se trata de una mejora incremental, sino de una pérdida del fundamento de seguridad.

Conviene señalar, sin embargo, que Shor no compromete toda la criptografía por igual. En los esquemas simétricos, el comportamiento se aproxima al de una función aleatoria, sin una estructura algebraica explotable, lo que impide aplicar técnicas basadas en la detección de periodicidad. El algoritmo de Grover proporciona en ese ámbito una aceleración cuadrática para la búsqueda exhaustiva de claves, que se compensa duplicando la longitud de clave; una degradación controlable, no una ruptura estructural. Ese análisis, no obstante, queda fuera del alcance de este trabajo, cuyo foco es la criptografía asimétrica.

En definitiva, el algoritmo de Shor muestra que las hipótesis de dureza sobre las que descansa la criptografía de clave pública dejan de ser propiedades intrínsecas de los problemas matemáticos y pasan a depender del modelo de computación considerado. La factorización de enteros no se volvió más fácil en 1994; lo que cambió fue qué clase de máquina podía atacarla. Y esa distinción, aparentemente técnica, tiene consecuencias que van mucho más allá de RSA o ECC.

1.5. Necesidad de criptografía post-cuántica

La pregunta que emerge es cuáles de esas hipótesis de dureza sobreviven al cambio de modelo; pero antes de buscar respuestas matemáticas, conviene precisar por qué esa búsqueda es urgente.

Un ordenador cuántico capaz de ejecutar Shor a escala criptográfica no existe hoy, sin embargo, la amenaza ya opera en el presente. Un adversario con recursos suficientes puede interceptar y almacenar tráfico cifrado ahora —por ejemplo, protegido mediante RSA, ECDH o esquemas clásicos basados en problemas de factorización y logaritmo discreto— y descifrarlo más tarde, cuando el hardware cuántico alcance escala suficiente.

Para datos cuya confidencialidad debe mantenerse durante años —registros médicos, comunicaciones diplomáticas, secretos industriales— ese desplazamiento es relevante. A eso se añade la inercia inherente a los sistemas criptográficos desplegados: actualizar protocolos, reemplazar hardware y recertificar infraestructuras lleva tiempo. Esperar a que la amenaza sea inminente es, en muchos contextos, esperar demasiado.

Esta presión llevó al NIST a iniciar en 2016 un proceso abierto de evaluación y estandarización de algoritmos resistentes al modelo cuántico. La convocatoria fue pública; la participación, global. Una primera fase de ese proceso concluyó en agosto de 2024 con la publicación de los estándares FIPS 203 y FIPS 204, basados en los esquemas Kyber y Dilithium respectivamente [6, 7]. No son propuestas experimentales, son el resultado de casi una década de análisis criptográfico internacional, con múltiples rondas de evaluación y criptoanálisis.

La fundamentación de la resistencia cuántica de esos estándares descansa en problemas geométricos sobre estructuras matemáticas conocidas como retículos, cuya forma de dureza parece cualitativamente distinta de la de RSA o ECC, y que, hasta la fecha, el modelo cuántico no logra comprometer.

Shor no compromete toda la dureza computacional, solo la que nace de estructura periódica explotable mediante transformadas globales. Los problemas sobre retículos no tienen esa arquitectura: no hay dominio frecuencial donde traducirlos, no hay periodicidad que localizar. Si el algoritmo de Shor muestra que la dificultad computacional puede depender del modelo de computación, los retículos muestran que no toda dificultad tiene la misma forma ni los mismos puntos débiles.

1.6. Introducción a la criptografía basada en retículos

Los retículos como objetos matemáticos son anteriores a cualquier aplicación criptográfica. Su estudio, motivado por problemas de geometría de números y aproximación diofántica, no perseguía ningún fin de seguridad. Lo que cambió con el trabajo de Ajtai en 1996 [1] fue la identificación de una propiedad inusual de ciertos problemas de retículos: la posibilidad de construir instancias promedio cuya resolución eficiente implicaría

resolver problemas de retículos en el peor caso. Esa equivalencia es poco común, ya que, la mayor parte de los problemas que sostienen sistemas criptográficos ofrecen garantías sobre instancias promedio; en retículos, esas garantías pueden heredar dureza del caso más adverso existente.

La estructura de estos problemas también difiere de la de RSA o ECC en un aspecto que resulta decisivo frente al modelo cuántico. El algoritmo de Shor actúa detectando periodicidad oculta: en la factorización, la función a^x mód N es periódica; en el logaritmo discreto, la estructura del grupo es explotable mediante una transformada de Fourier cuántica. Los problemas sobre retículos no presentan, al menos en las formulaciones utilizadas en criptografía, esa arquitectura: no hay simetría que localizar, ni dominio frecuencial donde traducirlos eficientemente. La barrera no es que los recursos cuánticos sean insuficientes, sino que la técnica sencillamente no aplica.

Esto no equivale a una prueba de seguridad incondicional. Afirmar que no se conoce un algoritmo cuántico eficiente para estos problemas es distinto de afirmar que no existe. Pero esa diferencia define el estado del arte en criptografía post-cuántica en general: toda hipótesis de dureza es, en último término, una hipótesis. Lo que distingue a los problemas sobre retículos frente a otras familias candidatas es la calidad de la evidencia acumulada y la solidez de las reducciones que conectan su dificultad con casos extremos bien estudiados. De entre las distintas formulaciones posibles, una ha concentrado la mayor parte de la actividad teórica y práctica desde 2005: el problema *Learning With Errors*.

1.7. Learning With Errors como problema central

Learning With Errors puede describirse, en primera aproximación, como un sistema de ecuaciones lineales al que se le ha añadido ruido. Fijado un módulo q y una dimensión n, cada muestra del problema consiste en un vector $a \in \mathbb{Z}_q^n$ elegido al azar y un escalar $b = \langle a, s \rangle + e \pmod{q}$, donde s es el secreto oculto y e un error pequeño. En su versión de búsqueda, dadas suficientes muestras, el objetivo es recuperar s. Sin el error, la tarea se reduce a eliminación gaussiana sobre $\mathbb{Z}_q$ —un problema de complejidad polinómica—. Con él, esa vía queda bloqueada, y no se conoce ningún algoritmo eficiente, clásico ni cuántico, que la sustituya.

El resultado que convierte esa observación en una garantía rigurosa es la reducción de Regev [8, 9], que conecta la dificultad de LWE con problemas geométricos sobre retículos. Regev demuestra que cualquier algoritmo eficiente para resolver LWE en instancias aleatorias implicaría la existencia de algoritmos eficientes para ciertos problemas geométricos sobre retículos formulados en el peor caso; problemas para los cuales, como se ha visto, no se conoce ninguna ventaja cuántica cualitativa. La cadena transfiere dureza desde el plano geométrico al algebraico, y desde el peor caso al promedio. Ese es el núcleo que distingue a LWE de una conjetura empírica: no solo ha resistido ataques, sino que su resistencia

está formalmente anclada.

Pero LWE no es únicamente una fuente de dureza. El mismo elemento que lo hace difícil —el ruido— es lo que permite construir criptografía sobre él. La perturbación controla la separación entre mensajes cifrados, la corrección del descifrado y, en última instancia, la viabilidad del esquema. Calibrar ese equilibrio es una de las preguntas centrales que este trabajo aborda, tanto desde el análisis teórico de los parámetros como desde la implementación experimental.

1.8. Metodología y objetivos del trabajo

El trabajo persigue cuatro objetivos concretos.

El primero es proporcionar una exposición matemáticamente rigurosa de las estructuras sobre las que descansa LWE: retículos como objetos geométricos, los problemas computacionales que sobre ellos se formulan —SVP y CVP, en sus versiones exactas y aproximadas— y los algoritmos de reducción que delimitan la dificultad práctica de resolverlos.

El segundo es analizar en profundidad el problema LWE: su definición formal, sus dos versiones algorítmicas —búsqueda y decisión—, la interpretación geométrica en términos de retículos q-arios, la reducción de Regev [8, 9], que vincula su dureza promedio con problemas de retículos en el peor caso, y el papel de los parámetros en el equilibrio entre seguridad y corrección del descifrado.

El tercero es describir las principales construcciones criptográficas derivadas de LWE —un esquema de cifrado elemental, un mecanismo de intercambio de claves y un esquema de firma digital—. En este punto se introducen también estándares reales como ML-KEM y ML-DSA, así como FrodoKEM como alternativa más conservadora basada en LWE estándar.

El cuarto es implementar una versión simplificada del esquema de cifrado de Regev y estudiar experimentalmente cómo los parámetros principales —ruido, módulo y dimensión— afectan a la corrección del descifrado, al coste computacional y al tamaño de las claves.

1.9. Estructura del trabajo

El trabajo se organiza en ocho secciones principales. Tras la introducción, las secciones 2, 3 y 4 construyen el sustrato matemático necesario para abordar LWE con rigor, siguiendo una progresión que va del objeto al problema y del problema al ataque. El Capítulo 2 introduce los retículos: su definición formal, propiedades geométricas fundamentales —volumen, determinante, norma— y el papel de la base en la representación del objeto. El Capítulo 3 formula los problemas computacionales centrales, SVP y CVP, junto con sus variantes aproximadas, y discute en qué sentido se entiende que son difíciles.

El Capítulo 4 describe los algoritmos de reducción LLL y BKZ, que representan el estado del arte en la práctica de ataques sobre esquemas basados en retículos y fijan el umbral mínimo de seguridad que cualquier esquema debe superar.

Estos cuatro capítulos no son un fin en sí mismos. Establecen el lenguaje, los problemas y los límites dentro de los cuales LWE adquiere su sentido criptográfico.

El Capítulo 5 es el núcleo teórico. Introduce LWE como sistema lineal perturbado, desarrolla su interpretación geométrica en términos de retículos q-arios, analiza la reducción de Regev y el papel de los parámetros, y presenta las variantes estructuradas Ring-LWE y Module-LWE que hacen el problema viable en la práctica. El Capítulo 6 muestra cómo esa teoría se convierte en criptografía: esquemas de cifrado, intercambio de claves, firmas digitales, y los estándares o construcciones derivadas de cada enfoque. El Capítulo 7 recoge la implementación experimental: cuatro experimentos que estudian el efecto de los parámetros sobre la corrección, el coste y la geometría del descifrado. El Capítulo 8 cierra con las conclusiones.

2. Fundamentos matemáticos de los retículos

La necesidad de replantear los fundamentos de la criptografía asimétrica frente al nuevo modelo de computación introducido en las secciones anteriores conduce de forma natural al estudio de nuevas estructuras matemáticas sobre las que sustentar hipótesis de dificultad robustas.

Entre las distintas propuestas desarrolladas en el ámbito de la criptografía post-cuántica, los retículos ocupan un lugar central. No se trata únicamente de una alternativa práctica a los sistemas clásicos, sino de un marco matemático especialmente adecuado para formular problemas cuya dificultad parece mantenerse incluso frente a adversarios cuánticos.

Antes de introducir construcciones criptográficas concretas, y en particular el problema Learning With Errors, resulta necesario fijar con precisión la estructura matemática sobre la que se apoyan estos esquemas.

2.1. Definición formal de retículo

Un retículo puede entenderse, de manera intuitiva, como un subgrupo aditivo discreto de puntos distribuido con una regularidad global en un espacio de dimensión finita $\mathbb{R}^m$. Esta caracterización captura dos propiedades esenciales de los retículos. La primera, que es un subgrupo, implica que el conjunto está cerrado bajo suma y bajo inversión; la segunda, que es discreto, implica que los puntos están aislados, separados entre sí por una distancia mínima positiva, geométricamente inducen una noción de distancia, proximidad y volumen. Ambas condiciones son necesarias para capturar la estructura característica de un retículo.

De forma operativa, la definición se construye a partir de una familia finita de vectores linealmente independientes. Dados n vectores linealmente independientes

$$b_1, b_2, \ldots, b_n \in \mathbb{R}^m,$$

el retículo generado por dicha familia es el conjunto de todas sus combinaciones lineales con coeficientes enteros:

$$\mathcal{L}(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} z_i \, b_i \;\middle|\; z_i \in \mathbb{Z} \right\}.$$

La restricción sobre los coeficientes es el núcleo de la definición. La imposición $z_i \in \mathbb{Z}$ selecciona, dentro de ese subespacio, una malla de puntos aislados. El salto de $\mathbb{R}$ a $\mathbb{Z}$ es el salto de la geometría continua a la geometría discreta.

Los vectores $b_1, \ldots, b_n$ reciben el nombre de *base* del retículo. El entero n es el *rango* y m la dimensión ambiente; cuando $n = m$, el retículo se dice de *rango pleno*. En criptografía,

el caso relevante es casi siempre este último.

Una base no es única. Por ejemplo, si $z \in \mathbb{Z}$, reemplazar b_1 por $b_1 + z \cdot b_2$ produce una base distinta que genera el mismo retículo. Este tipo de transformaciones no altera el conjunto de puntos, pero sí la forma en que estos se describen.

Más en general, cualquier transformación unimodular —una matriz de coeficientes enteros con determinante ± 1— transforma una base en otra base del mismo retículo.

Esta no unicidad tiene consecuencias que van más allá de lo formal: distintas bases pueden describir el mismo retículo y, aun así, ofrecer representaciones geométricas muy diferentes. Esta observación será importante más adelante, cuando se estudie cómo la base disponible puede facilitar u ocultar la estructura del retículo.

En este contexto, es relevante estudiar la existencia y la búsqueda de vectores de pequeña longitud dentro de un retículo. Aunque la definición del retículo es sencilla, determinar tales vectores —o incluso aproximarlos— es, en general, un problema computacionalmente difícil. Esta dificultad da lugar a problemas bien definidos cuya complejidad constituye uno de los pilares de la criptografía basada en retículos.

2.2. Bases y dimensión

La definición anterior introduce una familia de vectores generadores, pero todavía no aclara del todo qué información geométrica queda codificada en esa elección. En un retículo, la base no cumple únicamente la función de describir el conjunto; también fija la forma en que su estructura se representa y se manipula.

Sea la base

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

la matriz cuyas columnas son los vectores base del retículo. En notación matricial,

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

Los vectores de la base generan un subespacio vectorial de dimensión n, mientras que m describe únicamente el espacio en el que ese subespacio está inmerso. El entero n, denominado rango, puede interpretarse como la dimensión intrínseca del retículo. Cuando $n = m$, ocupa todo el espacio ambiente; cuando $n < m$, queda contenido en un subespacio de menor dimensión.

La base no es única. En particular, si $U \in \mathbb{Z}^{n \times n}$ es una matriz unimodular, entonces

$$\mathcal{L}(B) = \mathcal{L}(BU).$$

Estas transformaciones preservan $\mathbb{Z}^n$ y no alteran el conjunto de puntos del retículo.

Sin embargo, distintas bases pueden describir el mismo retículo con propiedades geométricas muy diferentes.

Esto refleja una diferencia esencial: el retículo, como conjunto de puntos, es independiente de la base elegida, pero los algoritmos operan sobre representaciones concretas, y su comportamiento sí depende de dicha elección. Esta dependencia —que los algoritmos operan sobre representaciones concretas aunque el retículo sea independiente de ellas— es lo que convierte la elección de base en algo criptográficamente relevante.

Más allá de esto, la dimensión no es suficiente para describir completamente la geometría del retículo. No solo importa cuántas direcciones linealmente independientes hay, sino también cómo se distribuyen los puntos en el espacio. Para capturar esta idea es necesario introducir una noción volumétrica asociada al retículo que permita medir la densidad de la malla generada por la base: su determinante.

2.3. Volumen y determinante

En el apartado anterior se ha puesto de manifiesto que distintas bases pueden generar exactamente el mismo retículo. Si se pretende trabajar con problemas cuya dificultad no dependa de una representación concreta, es necesario considerar magnitudes invariantes frente al cambio de base. El determinante cumple precisamente ese papel, al proporcionar una medida intrínseca del tamaño geométrico del retículo.

Dada una base

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

puede asociarse de forma natural una región elemental construida a partir de combinaciones lineales de los vectores base con coeficientes en el intervalo $[0, 1)$:

$$\mathcal{P}(B) = \left\{ \sum_{i=1}^{n} \alpha_i b_i \ : \ 0 \leq \alpha_i < 1 \right\}.$$

Esta región, denominada *celda fundamental*, actúa como una unidad geométrica básica.

El volumen de la celda fundamental asociada viene dado por

$$\mathrm{vol}(\mathcal{P}(B)) = \sqrt{\det(B^\top B)}.$$

En el caso de rango pleno ($n = m$), esta expresión se reduce a $|\det(B)|$, forma que utilizaremos en lo que sigue para simplificar la exposición.

Aunque la forma de la celda fundamental depende de la base elegida, su volumen no cambia al pasar a otra base del mismo retículo. Si B' es otra base de $\mathcal{L}$, existe una matriz unimodular $U \in \mathbb{Z}^{n \times n}$ tal que $B' = BU$, y por ello

$$|\det(B')| = |\det(B)|\,|\det(U)| = |\det(B)|,$$

porque $\det(U) = \pm 1$.

Este valor es un invariante del retículo, conocido como su *determinante* y denotado habitualmente por $\det(\mathcal{L})$.

Cuando $|\det(B)|$ es pequeño, la celda fundamental ocupa menos volumen y los puntos del retículo quedan, en conjunto, más concentrados; cuando es grande, la malla es más dispersa.

En este punto aparece una restricción que no era visible hasta ahora. No es posible transformar una base para hacer todos sus vectores arbitrariamente cortos sin afectar a su estructura: el volumen total debe preservarse. Acortar en una dirección obliga, de alguna forma, a compensar en otra.

En este sentido, el determinante no es solo una magnitud geométrica más, sino que introduce una limitación rígida sobre lo que es posible dentro del retículo. Esa rigidez, combinada con la naturaleza discreta de la estructura, permite formular problemas cuya dificultad parece mantenerse incluso cuando cambia el modelo de computación considerado.

Y es precisamente en este marco donde se formulan muchos de los problemas relevantes en criptografía: encontrar vectores excepcionalmente cortos o aproximar puntos con precisión bajo la restricción de que el volumen del retículo permanece inalterado.

2.4. Geometría de retículos

Hasta aquí han aparecido tres rasgos del retículo que conviene mantener juntos: su carácter discreto, la dependencia de la base y la existencia de un volumen invariante. Pero ninguno de ellos dice realmente cómo se comportan los puntos entre sí. La densidad global no distingue dos direcciones dentro del mismo retículo, y esa falta de estructura interna empieza a ser limitante.

Un retículo hereda de $\mathbb{R}^m$ las nociones habituales de norma, ángulo y ortogonalidad. Esto permite hablar de vectores cortos, de bases casi ortogonales o de puntos cercanos entre sí sin introducir estructuras adicionales. En este contexto, la norma euclídea fija la forma natural de medir longitudes. Dado un vector $x \in \mathbb{R}^m$, su longitud viene dada por

$$\|x\|_2 = \left(\sum_{i=1}^{m} x_i^2 \right)^{1/2},$$

y la distancia entre dos puntos se mide como $\|x - y\|_2$.

Esta elección fija la geometría en la que se formularán los problemas relevantes. A partir de aquí, preguntas como encontrar vectores especialmente cortos o aproximar puntos dejan de depender únicamente de la definición algebraica del retículo y pasan a depender de la distribución geométrica de sus puntos en el espacio.

Estas herramientas —norma, distancia, producto escalar— son las que dan contenido a preguntas como encontrar vectores especialmente cortos o localizar el punto del retículo

más próximo a un objetivo exterior. Son también las que hacen que esas preguntas sean, en alta dimensión, computacionalmente no triviales.

2.5. Bases buenas y malas

Una base será *buena* cuando sus vectores sean relativamente cortos y estén razonablemente cerca de ser ortogonales. No hace falta exigir ortogonalidad perfecta, basta con que la base no deforme en exceso la geometría del retículo. Una base será *mala*, en cambio, cuando sus vectores sean largos, estén muy sesgados o presenten dependencias angulares fuertes, de modo que la estructura quede descrita de forma algebraicamente correcta pero geométricamente opaca.

Con una base buena, los puntos cercanos en el espacio lo son también en términos de coordenadas enteras, y la representación resulta estable. En cambio, con una base mala, pequeñas variaciones en los coeficientes enteros pueden provocar desplazamientos grandes en el espacio.

Este fenómeno está directamente relacionado con la correlación entre los vectores base. Cuando dos vectores forman un ángulo pequeño, distintas combinaciones lineales pueden producir vectores de longitud moderada tras cancelar componentes grandes en direcciones casi alineadas. La dificultad no proviene de una mayor complejidad del retículo, sino de que su geometría queda escondida tras una parametrización poco adecuada.

Una descripción más fina se obtiene mediante la ortogonalización de Gram–Schmidt. Dada una base $B = [b_1, \dots, b_n]$, se definen

$$b_i^* = b_i - \sum_{j<i} \mu_{i,j} b_j^*, \qquad \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}.$$

Los vectores b_i^* capturan la estructura angular y las dependencias entre las direcciones de la base. En términos informales, una base es *buena* cuando los coeficientes $\mu_{i,j}$ permanecen controlados y las normas $\|b_i^*\|$ no decrecen bruscamente. En una base *mala*, algunas de estas normas se vuelven muy pequeñas, reflejando dependencias fuertes entre direcciones.

Una manera cuantitativa de medir globalmente la calidad de una base es el *defecto de ortogonalidad*:

$$\mathrm{od}(B) = \frac{\prod_{i=1}^{n} \|b_i\|}{\det(L)}.$$

La desigualdad de Hadamard garantiza $\mathrm{od}(B) \geq 1$, con igualdad si y solo si los vectores son mutuamente ortogonales entre sí. Cuanto mayor sea este cociente, más alejada está la base de la ortogonalidad: el producto de longitudes supera en mucho el volumen del retículo, lo que refleja la existencia de correlaciones angulares intensas. Una base buena tiene defecto próximo a 1; una base mala puede tenerlo exponencialmente grande en n.

Una visión local de la misma propiedad la ofrecen los cocientes de Gram–Schmidt consecutivos $\|b_{i+1}^*\|/\|b_i^*\|$. En una base bien reducida estos cocientes no pueden ser uniformemente pequeños; controlar su valor mínimo es, de hecho, el criterio central que emplean los algoritmos de reducción del Capítulo 4.

Esta asimetría es la que se explota en criptografía basada en retículos: se trabaja con un retículo que admite una base corta y casi ortogonal —que se mantiene privada— mientras que la representación pública se obtiene mediante transformaciones unimodulares que destruyen esa estructura favorable.

Sin acceso a una base adecuada, la estructura del retículo permanece oculta; con ella, se vuelve explotable.

En las secciones siguientes, estas ideas se formalizarán a través de problemas como SVP y CVP.

3. Problemas computacionales en retículos

Hasta ahora el retículo ha sido presentado como un objeto geométrico: un conjunto discreto de puntos, dotado de volumen, norma y una representación mediante bases. Sin embargo, en criptografía no interesa únicamente su definición, sino los problemas que pueden formularse sobre él y la dificultad de resolverlos.

Esta dificultad no aparece en la definición del objeto, sino en la interacción entre tres factores: la discreción, la alta dimensión y el hecho de que los algoritmos solo tienen acceso a una base, que puede ocultar la geometría relevante.

Para formalizar estos problemas es necesario fijar primero cómo se mide la longitud. En lo que sigue se utilizará la norma euclídea

$$\|x\|_2 = \left(\sum_{i=1}^{n} x_i^2 \right)^{1/2},$$

aunque en algunos contextos resulta útil considerar otras normas, como la norma infinita

$$\|x\|_\infty = \max_i |x_i|,$$

que permite controlar el tamaño de las coordenadas.

Dado un retículo $\mathcal{L} \subset \mathbb{R}^n$, se define su primer mínimo como

$$\lambda_1(\mathcal{L}) = \min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2,$$

es decir, la longitud del vector no nulo más corto del retículo. Esta cantidad es intrínseca al retículo y no depende de la base elegida. Determinarla —o incluso aproximarla— constituye uno de los problemas centrales de la teoría de retículos, y no se conoce ningún algoritmo eficiente que, dada una base arbitraria de dimensión alta, permita calcularla exactamente en tiempo polinómico.

A partir de aquí surgen problemas fundamentales como encontrar un vector de longitud mínima o, dado un punto exterior, determinar el punto del retículo más cercano.

En las subsecciones siguientes se formalizan estos problemas, comenzando por el problema del vector más corto (SVP) y el del punto más cercano (CVP), junto con sus variantes aproximadas.

3.1. Shortest Vector Problem (SVP)

Si la dificultad geométrica del retículo depende de cómo están distribuidos sus puntos cerca del origen, el primer objeto que conviene entender es el vector de menor longitud no trivial.

Definición (SVP). Dada una base $B \in \mathbb{R}^{n \times n}$ de un retículo $\mathcal{L} = \mathcal{L}(B)$, el *Shortest*

Vector Problem consiste en encontrar un vector $v \in \mathcal{L} \setminus \{0\}$ tal que

$$\|v\|_2 = \lambda_1(\mathcal{L}).$$

Dicho de otro modo, se trata de producir un vector no nulo cuya longitud sea mínima entre todas las longitudes posibles dentro del retículo.

Desde el punto de vista geométrico, $\lambda_1(\mathcal{L})$ mide la primera escala a la que el retículo se hace visible alrededor del origen. Si se dibuja una bola euclídea de radio creciente centrada en 0, el valor $\lambda_1(\mathcal{L})$ es el menor radio para el que esa bola contiene un punto del retículo distinto del origen. Esta interpretación deja de ser útil en dimensión alta y el problema pasa entonces a ser computacional.

Se distinguen dos versiones del problema: la exacta y las aproximadas. La versión exacta exige encontrar un vector de longitud exactamente $\lambda_1(\mathcal{L})$. La versión aproximada, más relevante en criptografía, fija un factor $\gamma(n) \geq 1$ y pide encontrar un vector no nulo $v \in \mathcal{L}$ tal que

$$\|v\|_2 \leq \gamma(n)\, \lambda_1(\mathcal{L}).$$

No se conoce ningún algoritmo clásico que resuelva SVP exacto en tiempo polinómico. Incluso sus versiones aproximadas resultan difíciles en general, y no se dispone de algoritmos cuánticos que alteren sustancialmente esta situación en alta dimensión. Estas cuestiones se precisarán más adelante al analizar la dificultad computacional de estos problemas.

Esta resistencia es una de las razones por las que los problemas sobre retículos se consideran candidatos naturales para construir esquemas criptográficos post-cuánticos.

En aplicaciones criptográficas rara vez se exige resolver SVP exacto. Lo relevante es la dificultad práctica de encontrar vectores suficientemente cortos en alta dimensión a partir de una base desfavorable. Esta distinción será clave al pasar a problemas más estructurados como LWE.

Antes de llegar a ellas, aparece un segundo problema igualmente fundamental. En SVP el punto de referencia es el origen, en CVP, ese punto será un vector arbitrario del espacio ambiente.

3.2. Closest Vector Problem (CVP)

El problema siguiente surge al modificar un único elemento de la formulación anterior. En SVP el punto de referencia era el origen. En el *Closest Vector Problem*, ese papel lo desempeña un vector arbitrario del espacio ambiente que, en principio, no tiene por qué pertenecer al retículo. Ya no se trata de detectar la primera escala interna del retículo, sino de decidir qué punto de la malla aproxima mejor un objetivo exterior, potencialmente perturbado.

Sea $\mathcal{L} \subset \mathbb{R}^n$ un retículo descrito mediante una base B, y sea $t \in \mathbb{R}^n$ un vector cualquiera del espacio ambiente. El problema consiste en determinar qué punto del retículo se encuentra más próximo a t con respecto a la norma fijada.

Definición (CVP). Dado un retículo $\mathcal{L}$ y un vector $t \in \mathbb{R}^n$, el *Closest Vector Problem* consiste en encontrar un vector $v \in \mathcal{L}$ tal que

$$\|t - v\|_2 = \min_{x \in \mathcal{L}} \|t - x\|_2.$$

Es decir, se busca el punto del retículo cuya distancia al objetivo sea mínima en norma euclídea.

Mientras que en SVP la bola euclídea crecía desde el origen hasta tocar el primer punto no trivial del retículo, en CVP esa bola se centra en un vector exterior t y se expande hasta encontrar el primer punto de la malla.

De forma equivalente, el problema también puede interpretarse como una partición del espacio. Cada punto del retículo induce una región formada por todos los puntos que están más cerca de él que de cualquier otro. Estas regiones —las celdas de Voronoi— recubren todo el espacio sin solaparse.

Resolver CVP equivale entonces a determinar en qué celda de Voronoi cae el vector t.

Si el vector objetivo t coincide con el origen, CVP se reduce esencialmente a SVP. En sentido inverso, SVP puede verse como un caso particular de CVP en un retículo adecuadamente construido. Esto no es exactamente así, ya que para $t = 0$, el punto del retículo más cercano al origen es, trivialmente, el propio 0; pero esta interdependencia ayuda a entender por qué ambos problemas comparten resultados de dureza similares —realmente demostrado mediante reducciones—.

Al igual que en SVP, se distinguen versiones exactas y aproximadas. La versión exacta exige encontrar el punto más cercano. En la versión aproximada, dado un factor $\gamma(n) \geq 1$, se pide producir un vector $v \in \mathcal{L}$ tal que

$$\|t - v\|_2 \leq \gamma(n) \cdot \min_{x \in \mathcal{L}} \|t - x\|_2.$$

CVP exacto es, en general, al menos tan difícil como SVP exacto, y no se conocen algoritmos en tiempo polinómico para dimensión arbitraria. Incluso las versiones aproximadas presentan una complejidad elevada cuando el factor de aproximación es pequeño.

Aun así, CVP introduce un elemento adicional que no estaba presente en SVP: el desplazamiento.

A diferencia de SVP, en CVP la dificultad no reside únicamente en la estructura del retículo, sino en la interacción entre este y el vector objetivo. Si t se encuentra en el interior de una celda de Voronoi, pequeñas perturbaciones no modifican el vector del retículo más cercano. En cambio, si t está próximo a una frontera entre celdas, variaciones arbitrariamente pequeñas pueden hacer que pase a una celda adyacente, provocando un

cambio discreto en la solución.

La inestabilidad observada refleja el efecto del ruido sobre una estructura discreta.

En este sentido, CVP puede verse como un problema de descodificación: dado un punto cercano a un vector del retículo, recuperar dicho vector. Esta misma estructura aparece en problemas como Learning With Errors, donde la información se presenta deliberadamente perturbada. Resolver el problema consistirá, en esencia, en identificar de qué punto estructurado procede esa observación desplazada.

3.3. Dificultad computacional

Llegados a este punto, conviene precisar en qué sentido se entiende aquí la dificultad de los problemas sobre retículos. En teoría de la complejidad hace falta distinguir entre varios niveles: la resolución exacta, la aproximación con un factor controlado y la naturaleza de las reducciones bajo las que se demuestra la dureza.

Resolver SVP o CVP de forma exacta significa encontrar la solución óptima: el vector no nulo más corto en un caso, o el punto del retículo más cercano al objetivo en el otro. Las versiones aproximadas relajan esta exigencia. Como ya se ha comentado, en lugar de pedir la solución óptima, se admite un margen multiplicativo $\gamma(n) \geq 1$.

La dificultad del problema depende de lo exigente que sea el factor de aproximación $\gamma(n)$: cuanto más cercano esté a 1, más próxima debe ser la solución al óptimo y más difícil resulta el problema.

Se conocen resultados de dureza que muestran que ciertas versiones de SVP y CVP son NP-hard bajo reducciones apropiadas [4], y esa dureza persiste incluso en algunos regímenes aproximados. Dicho de forma informal, no solo parece difícil encontrar la solución exacta; también puede seguir siendo difícil obtener soluciones suficientemente cercanas al óptimo cuando el factor de aproximación es pequeño.

Conviene, no obstante, no comprimir demasiado esta afirmación ya que la dificultad depende del tipo exacto de problema, de la norma considerada, del factor $\gamma(n)$ y del modelo de reducción utilizado. Para factores de aproximación suficientemente grandes existen algoritmos polinómicos, pero a medida que el margen de aproximación se reduce, la complejidad crece rápidamente.

Esto explica por qué los algoritmos de reducción de bases, que aparecerán en la siguiente sección, no resuelven en general SVP o CVP de forma exacta, sino que buscan compromisos controlados entre coste computacional y calidad de la aproximación. En dimensiones altas, la cuestión rara vez es si puede obtenerse la solución óptima, sino cuán buena puede ser la solución alcanzable con recursos computacionales realistas.

Desde la perspectiva criptográfica, no se exige demostrar que un problema sea imposible en sentido absoluto, sino contar con evidencia sólida de que no puede resolverse eficientemente. En este contexto se trabaja en el régimen donde los mejores algoritmos

conocidos requieren tiempo exponencial o subexponencial en la dimensión.

En particular, no se conoce ningún algoritmo clásico de tiempo polinómico que resuelva SVP o CVP exactos en dimensión arbitraria. Los algoritmos cuánticos actuales ofrecen, en el mejor de los casos, mejoras polinómicas o cuadráticas para determinadas subrutinas, pero no alteran de forma cualitativa la complejidad asintótica en alta dimensión.

Esta ausencia de una estructura algebraica explotable marca una diferencia fundamental con los problemas tratados en la primera parte del trabajo. Mientras que la factorización o el logaritmo discreto admiten reformulaciones periódicas que pueden atacarse mediante transformadas globales; los problemas sobre retículos combinan discreción, alta dimensión y geometría euclídea sin exhibir simetrías que permitan una reducción análoga.

Esa estabilidad relativa frente al cambio de modelo computacional, junto con los resultados formales de dureza disponibles, una de las razones por las que los retículos ocupan un lugar tan destacado en criptografía post-cuántica.

Existe además una distinción que resultará decisiva en el capítulo siguiente: la diferencia entre dificultad en el *peor caso* y dificultad en el *caso promedio*. Las garantías de NP-dureza mencionadas son del tipo peor caso: afirman que existen instancias concretas para las que ningún algoritmo eficiente conocido encuentra solución, pero no dicen nada sobre instancias generadas aleatoriamente. En muchos problemas NP, estas dos nociones divergen: las instancias difíciles pueden ser excepcionales y la mayoría de las instancias aleatorias pueden ser, en la práctica, relativamente accesibles. Lo que hace a ciertos problemas sobre retículos especialmente valiosos para criptografía es que admiten reducciones formales desde la dificultad en el peor caso a la dificultad en el caso promedio: resolver eficientemente instancias aleatorias implicaría resolver eficientemente las instancias más adversas posibles. Esta propiedad inusual es la que convierte a SVP y CVP en el fundamento de las construcciones que aparecerán en el Capítulo 5.

Con todo, los problemas que se utilizarán más adelante no serán simplemente SVP o CVP en su forma exacta. Lo que hace especialmente fértil a la teoría de retículos es que estas nociones de dureza pueden transferirse, mediante reducciones, a problemas algebraicamente más manejables. Ese será el punto de entrada natural hacia las técnicas de reducción de bases y, más adelante, hacia Learning With Errors.

4. Reducción de retículos

4.1. Concepto de reducción

Desde un punto de vista geométrico, reducir una base significa reemplazar la familia generadora por otra que describa el mismo conjunto de puntos de forma más regular. Formalmente, dado un retículo $\mathcal{L}(B)$, se busca una nueva base

$$B' = BU,$$

donde $U \in \mathbb{Z}^{n\times n}$ es unimodular, tal que los vectores de B' sean más cortos y limiten su dependencia angular.

La herramienta conceptual que permite medir esta mejora vuelve a ser la ortogonalización de Gram–Schmidt, que cuantifica hasta qué punto los vectores interfieren entre sí. Los algoritmos de reducción no actúan directamente sobre los vectores originales, sino que controlan la calidad de la base a través de sus vectores ortogonalizados; ya que es, en esta representación, donde puede detectarse si la base se degrada —cuando algunas normas $\|b_i^*\|$ colapsan o los coeficientes de proyección crecen—, y donde se formulan las condiciones que definen una base reducida.

Como se ha visto, el determinante del retículo es invariante. El producto de las longitudes de los vectores base queda acotado inferiormente por el determinante, lo que impone un límite rígido sobre cuánto pueden acortarse simultáneamente. Ganar en una dirección obliga a compensar en otra. Mejorar en una dirección implica, inevitablemente, pagar en otra.

Con esto, la palabra *reducción* no designa un estado absoluto, sino una familia de compromisos entre dos objetivos que tiran en direcciones distintas. Por un lado, interesa acercarse a una base corta y casi ortogonal. Por otro, el coste de encontrar una base así crece muy deprisa con la dimensión. Reducir una base significa, criptográficamente hablando, aceptar una mejora controlada de su geometría a un coste computacional asumible.

La reducción exacta —en el sentido de obtener vectores realmente cortos— es, en esencia, tan difícil como resolver SVP.

En la práctica, estos algoritmos no resuelven el problema en general, pero proporcionan las mejores aproximaciones conocidas con recursos finitos. Sin ellos, problemas como SVP o CVP permanecerían como objetos teóricos sin herramientas efectivas para aproximarlos. Desde el punto de vista algorítmico, permiten aproximar soluciones en dimensión alta cuando se dispone únicamente de una base arbitraria; desde el punto de vista criptográfico, representan el principal método conocido para atacar esquemas basados en retículos. La seguridad de los esquemas basados en retículos no depende únicamente de la dureza teórica de SVP o CVP, sino de hasta qué punto los algoritmos de reducción son capaces —o mejor dicho, hasta qué punto NO son capaces— de explotar la estructura de la base pública.

En las subsecciones siguientes se analizarán dos de los algoritmos más relevantes en este contexto: LLL, que proporciona una reducción en tiempo polinómico con garantías moderadas, y BKZ, que permite obtener bases de mayor calidad a costa de un incremento significativo del coste computacional.

En el cuerpo principal se presenta únicamente el papel conceptual y criptográfico de estos algoritmos. El pseudocódigo correspondiente se recoge en el Anexo A, ya que el objetivo del trabajo no es implementar técnicas de reducción de bases, sino entender cómo delimitan la seguridad práctica de los problemas reticulares.

4.2. Algoritmo LLL

El algoritmo de Lenstra–Lenstra–Lovász (LLL), propuesto en 1982, constituye el primer procedimiento eficiente que permite transformar una base arbitraria de un retículo en otra base con propiedades geométricas controladas.

La idea no consiste en buscar directamente vectores extremadamente cortos —lo que conduciría de nuevo a SVP—, sino en imponer condiciones locales que eviten que la base se degrade. LLL no produce una base óptima, pero sí garantiza una mejora controlada en tiempo polinómico.

Sea $B = [b_1, \ldots, b_n]$ una base de un retículo, b_i^* sus vectores ortogonalizados de Gram–Schmidt y los coeficientes

$$\mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}.$$

El algoritmo actúa sobre estos coeficientes y sobre las normas $\|b_i^*\|$, corrigiendo progresivamente la base mediante transformaciones unimodulares que satisface dos condiciones fundamentales.

(1) Reducción de tamaño. Para todo $i > j$,

$$|\mu_{i,j}| \leq \tfrac{1}{2}.$$

Esta condición evita que un vector contenga grandes proyecciones en direcciones anteriores, lo que reduciría la estabilidad de la representación.

(2) Condición de Lovász. Existe un parámetro $\delta \in (1/4, 1)$ tal que, para todo $i \geq 2$,

$$\|b_i^*\|^2 \geq (\delta - \mu_{i,i-1}^2)\, \|b_{i-1}^*\|^2.$$

Cuando la desigualdad no se cumple, el algoritmo intercambia los vectores b_{k-1} y b_k, y actualiza la ortogonalización. Esta restricción impide que las normas de los vectores ortogonalizados decrezcan bruscamente, lo que equivaldría a una base degenerada.

El parámetro δ controla el compromiso entre calidad de la base y coste computacional. En la práctica se suele tomar $\delta \approx 0{,}99$.

Estas dos operaciones —reducción de tamaño y posibles intercambios— se aplican iterativamente. El algoritmo recorre la base de izquierda a derecha y, en cada paso, modifica el vector actual restándole múltiplos enteros de los vectores anteriores para que no tenga demasiada componente en esas direcciones (es decir, que los coeficientes $\mu_{k,j}$, obtenidos de proyectar b_k sobre b_j^*,no sean grandes).

A continuación, verifica la condición de Lovász para el par de vectores consecutivos. Si la condición se satisface, el algoritmo avanza al siguiente índice. Si se viola, intercambia los vectores correspondientes y retrocede una posición para restaurar la estructura. Este mecanismo de avance y retroceso, basado en intercambios locales, se repite hasta que todas las condiciones se cumplen simultáneamente.

El resultado es una base *LLL-reducida*. Bajo estas condiciones, el algoritmo garantiza que el primer vector de la base reducida satisface

$$\|b_1\| \leq 2^{\frac{n-1}{2}} \lambda_1(\mathcal{L}),$$

es decir, proporciona una aproximación exponencial del vector más corto.

Aunque LLL mejora de forma significativa una base arbitraria, la calidad de la aproximación degrada rápidamente con la dimensión. En alta dimensión, el factor $2^{(n-1)/2}$ es demasiado grande como para resolver SVP de forma efectiva.

En este sentido, LLL no rompe los sistemas basados en retículos, pero sí define el nivel mínimo de seguridad que estos deben superar. Todo lo que LLL puede explotar debe considerarse, a efectos prácticos, información accesible para un adversario.

Por tanto, LLL actúa como el nivel base de ataque: cualquier esquema que no resista una reducción mediante LLL debe considerarse inseguro. Aunque por sí solo no rompe las construcciones modernas, establece un umbral mínimo; todo lo que LLL puede explotar debe asumirse accesible para un adversario.

Ese límite es lo que marca la frontera entre lo que puede hacerse con una reducción polinómica y lo que exigirá técnicas más costosas; donde entra BKZ —sacrificando eficiencia a cambio de una reducción más precisa—.

4.3. Algoritmo BKZ

El algoritmo Block Korkine–Zolotarev (BKZ) puede entenderse como una generalización del algoritmo LLL que permite obtener bases de mayor calidad a costa de un incremento significativo del coste computacional.

LLL opera mediante correcciones: coeficientes de proyección demasiado grandes, desequilibrios entre vectores consecutivos, desajustes visibles en la ortogonalización de Gram–Schmidt. BKZ parte de esa misma lógica, pero introduce una búsqueda más profunda dentro de subretículos generados por subconjuntos de la base.

Sea $B = [b_1, \ldots, b_n]$ una base de un retículo previamente reducida por LLL. BKZ

fija un parámetro $\beta \in \{2, \ldots, n\}$, denominado *tamaño de bloque*. Para cada posición $k \in \{1, \ldots, n - \beta + 1\}$ de la base, se considera el subretículo generado por los vectores

$$b_k, b_{k+1}, \ldots, b_{k+\beta-1}.$$

Sobre este subretículo se aplica un algoritmo de resolución (exacta o aproximadamente) de SVP (SVP-oracle) en dimensión β, implementado mediante enumeración o técnicas de *sieving*.

El vector corto obtenido se inserta en la base global mediante transformaciones unimodulares, sustituyendo a uno de los vectores del bloque. A continuación, se aplica un paso de reducción tipo LLL para reordenar los vectores y restaurar condiciones de reducción locales, el equilibrio de la base. Este proceso se repite en sucesivas pasadas sobre la base hasta que no se producen mejoras en ninguno de los bloques. En ese punto, se dice que la base es *BKZ-reducida*.

Todo depende entonces del parámetro β. Si β es pequeño, BKZ se comporta de manera cercana a LLL: rápido, robusto y con mejoras moderadas. A medida que β crece, la calidad de la base mejora, pero el coste computacional aumenta rápidamente, llegando al límite, cuando $\beta = n$, el algoritmo se aproxima a resolver SVP exacto. Cada incremento en β compra calidad geométrica a costa de tiempo y memoria.

La calidad de la reducción suele evaluarse mediante el *Root Hermite Factor* (δ_0), que mide la longitud del primer vector respecto al determinante del retículo que disminuye a medida que β aumenta, acercándose al valor óptimo.

A diferencia de LLL, que establece un umbral mínimo de seguridad, BKZ representa el mejor ataque genérico conocido. La seguridad práctica de muchos esquemas se estima precisamente como el tamaño de bloque β necesario para que BKZ recupere información relevante con recursos computacionales inasumibles. Entender por tanto su comportamiento equivale, en gran medida, a entender la seguridad efectiva de los esquemas basados en retículos.

La dificultad del problema deja de depender únicamente de su forma algebraica y pasa a estar determinada por los límites de los algoritmos de reducción aplicables a las instancias asociadas. Por ello, los esquemas criptográficos no se construyen directamente sobre SVP o CVP, sino sobre problemas más estructurados cuya dureza puede relacionarse con ellos mediante reducciones.

En este punto, la dificultad pasa a manifestarse en problemas donde la información aparece oculta, perturbada o incompleta, pero sigue estando gobernada por esa misma estructura subyacente.

El ejemplo canónico de este enfoque es el problema Learning With Errors, cuya dificultad refleja de manera operativa los límites de los algoritmos de reducción analizados anteriormente y permite trasladar esa dureza al diseño de construcciones criptográficas

concretas.

5. Learning With Errors

5.1. Motivación y definición formal

Los algoritmos de reducción introducidos en la sección anterior establecen, en la práctica, el límite hasta dónde llega hoy la capacidad de ataque sobre problemas geométricos en retículos. Por debajo de ese umbral se encuentran estructuras que todavía pueden ser manipuladas mediante técnicas como LLL o BKZ; por encima, la geometría se vuelve progresivamente inaccesible. El problema Learning With Errors se sitúa precisamente en esa frontera: conserva una estructura algebraica simple, pero introduce una perturbación suficiente como para romper los métodos clásicos de resolución. No toma la forma desnuda de SVP o CVP, pero tampoco se separa de ellos.

La construcción de LWE parte de muestras aleatorias que combinan una parte lineal con un término de ruido.

Formalmente, se procede de la siguiente manera. Fijados un módulo entero $q \geq 2$ y una dimensión n. Una muestra de LWE se genera eligiendo primero un vector

$$a \xleftarrow{\$} \mathbb{Z}_q^n,$$

uniformemente al azar en el espacio modular. A continuación, se selecciona un vector secreto fijo $s \in \mathbb{Z}_q^n$, y un error

$$e \leftarrow \chi,$$

donde χ es una distribución de probabilidad centrada en cero. La observación que se entrega es

$$b = \langle a, s \rangle + e \pmod{q}.$$

Cada muestra consiste, por tanto, en un par

$$(a, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q,$$

que oculta una relación lineal exacta bajo una perturbación controlada. La tarea algorítmica consistirá, de una forma u otra, en extraer información sobre el secreto s a partir de muchas muestras independientes construidas de este modo.

Sin el término e, el sistema sería resoluble mediante eliminación gaussiana sobre $\mathbb{Z}_q$. Con él, las ecuaciones dejan de ser exactas y la información aparece desplazada de forma controlada, pero no eliminada.

El comportamiento del sistema está determinado por tres parámetros fundamentales. El primero es la dimensión n, que controla el tamaño del secreto y, en la práctica, el nivel de seguridad del esquema. A medida que n crece, también lo hace el espacio de búsqueda y la dimensión de los espacios de posibles soluciones consistentes con las observaciones.

El segundo parámetro es el módulo q, que define el espacio aritmético en el que se trabaja. Todas las operaciones se realizan en $\mathbb{Z}_q$, lo que introduce una periodicidad discreta en las ecuaciones. Su elección afecta tanto a la estabilidad del sistema como a la separación entre posibles valores válidos.

La distribución χ, por su parte, determina el comportamiento del error. En la práctica, ese error suele modelarse mediante una distribución discreta sobre $\mathbb{Z}$, concentrada en torno a cero. Dado un parámetro $\sigma > 0$, la distribución discreta gaussiana $\mathcal{D}_{\mathbb{Z},\sigma}$ asigna a cada entero $x \in \mathbb{Z}$ una probabilidad proporcional a

$$\Pr[e = x] \propto e^{-\pi x^2/\sigma^2}, \qquad x \in \mathbb{Z}.$$

Esta distribución asigna mayor probabilidad a valores pequeños y reduce progresivamente la probabilidad de errores grandes. El parámetro σ controla la intensidad de la perturbación: si es demasiado pequeño, la estructura lineal se vuelve casi visible; si es demasiado grande, la información queda completamente difuminada. En la practica, el parámetro σ representa la desviación típica del error.

Si en lugar de una sola muestra se consideran m muestras independientes (a_i, b_i), resulta natural agruparlas en forma matricial. Sean

$$A \in \mathbb{Z}_q^{m\times n}, \qquad s \in \mathbb{Z}_q^n, \qquad e \in \mathbb{Z}^m, \qquad b \in \mathbb{Z}_q^m,$$

donde las filas de A son los vectores $a_i^\top$ y las componentes de e son los errores correspondientes. Entonces la colección completa de muestras puede escribirse como

$$b = As + e \pmod q.$$

Parece un sistema lineal, pero la presencia de e impide leerlo como una igualdad exacta en $\mathbb{Z}_q^m$, y esa pequeña desviación es precisamente la fuente de la dificultad que se explotará más adelante.

Un ejemplo elemental ayuda a fijar la intuición sin anticipar todavía la interpretación geométrica. Tomemos $q = 11$ y $n = 2$, y supongamos que el secreto es

$$s = \begin{pmatrix} 3 \\ 2 \end{pmatrix}.$$

Si elegimos

$$a = \begin{pmatrix} 4 \\ 7 \end{pmatrix}, \qquad e = 1,$$

entonces

$$\langle a, s\rangle = 4 \cdot 3 + 7 \cdot 2 = 26 \equiv 4 \pmod{11},$$

y por tanto

$$b = 4 + 1 \equiv 5 \pmod{11}.$$

La muestra obtenida es, en consecuencia,

$$(a, b) = \left(\begin{pmatrix} 4 \\ 7 \end{pmatrix}, 5 \right).$$

Nada en esta pareja revela de forma inmediata el secreto. Tampoco parece una perturbación caótica. Contiene estructura, pero esa estructura llega mezclada con un ruido pequeño y controlado.

Eso basta, de momento, para situar el problema. LWE no nace como una rareza algebraica, sino como una manera muy precisa de esconder información lineal dentro de ecuaciones casi correctas. En la subsección siguiente hará falta distinguir dos formulaciones naturales del problema. Después podrá verse por qué esa perturbación aparentemente modesta cambia por completo su naturaleza computacional.

5.2. Versiones del problema

La formulación anterior describe cómo se generan las muestras, pero todavía no especifica qué significa exactamente *resolver* LWE. La respuesta admite dos lecturas algorítmicas distintas que, aunque equivalentes bajo ciertos supuestos, capturan dificultades de tipo diferente. La primera se centra en la recuperación del secreto; la segunda en la imposibilidad de distinguir estructura de aleatoriedad.

Search-LWE. El problema consiste en, dado acceso a suficientes muestras LWE asociadas a un secreto fijo s, recuperar ese secreto.

En forma matricial, el problema puede enunciarse así: dados

$$A \in \mathbb{Z}_q^{m \times n}, \qquad b = As + e \pmod{q},$$

donde A es pública, e es un vector de error cuyas componentes siguen una distribución χ y $s \in \mathbb{Z}_q^n$ es desconocido, se pide encontrar s.

Esta formulación refleja con mayor claridad la intuición de recuperación: se observan muchas ecuaciones lineales ruidosas y se pide reconstruir la variable común que las genera. Si el error no estuviera presente, bastaría con resolver el sistema. Con error, esa estrategia deja de ser válida, pero la meta sigue siendo la misma: reconstruir el secreto.

Decision-LWE. La segunda versión elimina explícitamente la idea de recuperación y se centra en la capacidad de distinguir distribuciones.

Se considera una distribución sobre pares $(a, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ generados de dos formas posibles:

- En el primer caso, se elige $a \xleftarrow{\$} \mathbb{Z}_q^n$, $s \in \mathbb{Z}_q^n$ fijo, $e \leftarrow \chi$, y se define

$$b = \langle a, s \rangle + e \pmod{q}.$$

- En el segundo caso, se elige completamente al azar

$$a \xleftarrow{\$} \mathbb{Z}_q^n, \quad b \xleftarrow{\$} \mathbb{Z}_q.$$

El problema *Decision-LWE* consiste en determinar si una colección de muestras proviene del primer proceso o del segundo. Es decir, distinguir entre estructura con ruido o pura aleatoriedad.

A primera vista, aunque esta versión parece más débil, desde el punto de vista criptográfico, esta es la formulación decisiva. Un sistema de cifrado necesita, en primera instancia, impedir que distinga un texto cifrado real de un objeto sin información útil. En particular, incluso una ventaja no despreciable en esta tarea de distinción implicaría la existencia de estructura explotable en las muestras, lo que contradice la hipótesis de dureza del problema.

Search-LWE modela un escenario de recuperación: existe una estructura oculta y el objetivo es reconstruirla. Decision-LWE, en cambio, modela un escenario de indistinguibilidad: no se intenta reconstruir nada, sino determinar si hay estructura alguna.

No obstante, para LWE esa diferencia no da lugar a dos problemas independientes: bajo hipótesis estándar —en particular cuando el módulo q es primo y el número de muestras es suficientemente grande en función de n— se puede demostrar que Search-LWE y Decision-LWE son equivalentes.

Proposición. Sea q un número primo y supóngase que se dispone de un número suficiente de muestras. Bajo hipótesis estándar sobre los parámetros, las versiones de búsqueda y de decisión de LWE son equivalentes mediante reducciones probabilísticas de tiempo polinómico [8].

Por ello, aunque la intuición inicial de LWE suele formularse como recuperación de s, gran parte de las pruebas de seguridad criptográfica se apoyan en Decision-LWE. La equivalencia entre ambas variantes permite trasladar resultados de dureza y trabajar con la formulación más conveniente en cada contexto sin pérdidas sustanciales de eficiencia [8, 9]

Estas dos maneras complementarias de mirar el mismo fenómeno —recuperación e indistinguibilidad— es la que convierte a LWE en un objeto fundamental dentro de la criptografía post-cuántica, y la que justifica que LWE puede servir, al mismo tiempo, como problema matemático duro y como base de construcciones criptográficas concretas.

Fijadas estas dos perspectivas, el siguiente paso consiste en reinterpretar las ecuaciones ruidosas no ya como objetos algebraicos aislados, sino como puntos cercanos a ciertas

estructuras geométricas de alta dimensión ya explicadas anteriormente: los retículos.

5.3. Interpretación geométrica

Hasta aquí LWE ha aparecido como una familia de ecuaciones lineales con ruido. Esa descripción es correcta, pero todavía no dice con precisión qué tipo de dificultad encierra el problema. Ya no conviene pensar solo en coeficientes y congruencias, sino en cercanía. Ahí es donde LWE empieza a conectarse de forma directa con la geometría de retículos desarrollada anteriormente.

5.3.1. LWE como sistema lineal perturbado

Partamos de la forma matricial ya introducida:

$$b = As + e \pmod{q},$$

donde $A \in \mathbb{Z}_q^{m \times n}$ es pública, $s \in \mathbb{Z}_q^n$ es el secreto y e recoge los errores.

Si el vector de error fuese nulo, es decir, si $e = 0$, se obtendría un sistema lineal exacto sobre $\mathbb{Z}_q$

$$b = As \pmod{q}.$$

Dadas suficientes muestras y siempre que la matriz A tuviera rango suficiente, para recuperar s bastaría con aplicar eliminación gaussiana, o cualquier otro procedimiento polinómico equivalente, para reconstruir el secreto. En términos computacionales, la resolución sería polinómica en el tamaño de la instancia.

Sin embargo, lo que realmente se observa en LWE viene dado por

$$b_i = \langle a_i, s \rangle + e_i \pmod{q}.$$

con un vector e pequeño, pero no nulo. Ese pequeño desplazamiento implica que, aunque existe un secreto que ha generado todas las muestras, las igualdades no se satisfacen exactamente en $\mathbb{Z}_q^m$ mediante una única elección de s. Cada muestra arrastra su propia perturbación, y esa perturbación impide alinear todas las observaciones dentro de una misma identidad lineal perfecta.

Esto implica que, aunque existe un secreto s que ha generado todas las muestras, las igualdades no se satisfacen exactamente en $\mathbb{Z}_q^m$. En particular, si se interpreta el sistema como un conjunto de ecuaciones lineales exactas, no existe ningún vector s que verifique simultáneamente todas ellas.

Esta reformulación equivale a que, para resolver LWE, ya no hay que encontrar un

secreto que haga $As = b$, a encontrar un vector s tal que la cantidad

$$As - b \pmod{q}$$

sea pequeña en norma, en un sentido coherente con la magnitud del error introducido

Por ello, LWE no debe entenderse como un problema de resolución exacta de sistemas lineales, sino como un problema de aproximación en presencia de ruido estructurado.

Las soluciones exactas del sistema $b = As \pmod{q}$ forman una estructura discreta; el error desplaza el punto observado fuera de ella. Recuperar el secreto equivale entonces a determinar de qué punto de esa estructura proviene la observación perturbada.

5.3.2. Retículos q-arios y estructura geométrica

La reformulación anterior da paso a la siguiente pregunta: ¿cerca de qué objeto se mide esa proximidad?

Las ecuaciones exactas $b = As \pmod{q}$ no describen solo soluciones aisladas. Al considerar todos los posibles valores de s, generan una familia estructurada de vectores en $\mathbb{Z}_q^m$. El soporte que permite expresar las congruencias modulares definidas por la matriz A como objetos geométricos en $\mathbb{Z}^m$ lo proporcionan los llamados retículos q-arios.

El primero de ellos es el **retículo primal** asociado a A. Se define por

$$\mathcal{L}_q(A) = \{x \in \mathbb{Z}^m : x \equiv As \pmod{q} \text{ para algún } s \in \mathbb{Z}_q^n\}.$$

Esta expresión recoge todos los vectores enteros que, módulo q, pueden obtenerse como combinación lineal de las columnas de A. Estos vectores se expanden como una malla discreta de puntos en $\mathbb{Z}^m$ con periodicidad controlada por q.

Aquí aparece la lectura geométrica de LWE. Si las muestras se escriben como

$$b = As + e \pmod{q},$$

entonces existe un vector $v \in \mathcal{L}_q(A)$ tal que

$$b = v + e \pmod{q}.$$

La parte As describe, módulo q, un punto perteneciente a $\mathcal{L}_q(A)$, mientras que el vector de error e lo desplaza de su posición exacta. Dicho de forma más directa: LWE consiste en recibir un punto desplazado de un retículo q-ario.

Entender entonces que el error no es una simple perturbación, sino un desplazamiento geométrico de una estructura discreta de alta dimensión, permite ver la relación de LWE con los problemas de retículos.

Existe además una segunda estructura asociada a A, el **retículo dual**, definido por

$$\mathcal{L}_q^{\perp}(A) = \{y \in \mathbb{Z}^m : A^{\top} y \equiv 0 \pmod{q}\}.$$

Este conjunto reúne los vectores enteros ortogonales, módulo q, a todas las columnas de A. O dicho de otra manera, las direcciones que anulan el producto escalar con A.

En efecto, si $y \in \mathcal{L}_q^{\perp}(A)$ y se parte de una muestra LWE escrita como $b = As + e \pmod{q}$, entonces

$$\langle y, b \rangle = \langle y, As \rangle + \langle y, e \rangle \pmod{q}.$$

Pero, por definición de $\mathcal{L}_q^{\perp}(A)$,

$$\langle y, As \rangle = y^{\top} As = (A^{\top} y)^{\top} s \equiv 0 \pmod{q},$$

de modo que se obtiene la identidad

$$\langle y, b \rangle \equiv \langle y, e \rangle \pmod{q}.$$

Al proyectar sobre una dirección del retículo dual, la contribución de s se cancela y solo permanece el efecto del error.

Esto permite construir combinaciones lineales donde la contribución de s se anula y toda la información queda concentrada en el ruido. Si el error no está bien calibrado, estas combinaciones introducen correlaciones explotables. Si lo está, esa vía de ataque queda bloqueada.

Primal y dual son dos lecturas complementarias de mirar la misma instancia: la cercanía geométrica a puntos válidos y las relaciones lineales que anulan la estructura oculta y dejan expuesto únicamente el ruido. Esta segunda lectura será especialmente útil al analizar los ataques duales.

A partir de aquí, el problema queda completamente reformulado: LWE ya no consiste en resolver ecuaciones exactas, sino de encontrar el punto más cercano a una observación perturbada en retículos q-arios. Esa conexión con el problema del punto más cercano —estudiado en el Capítulo 3.2 — es la que ancla la dificultad de LWE en problemas geométricos con garantías formales de dureza, y que se precisarán en la sección siguiente.

5.4. Dificultad computacional

Hasta ahora la dificultad de LWE se ha descrito de forma operativa: un sistema lineal perturbado, una interpretación geométrica en términos de cercanía y una conexión con problemas de descodificación sobre retículos. Esa lectura explica por qué el ruido transforma un sistema lineal en un problema geométrico, pero todavía no justifica por qué esa dificultad debería sostenerse frente a adversarios generales, y menos aún frente a modelos

de computación distintos del clásico, como el cuántico introducido en la primera parte del trabajo.

5.4.1. Reducción de Regev

La respuesta aparece en uno de los resultados más influyentes de la teoría de retículos aplicada a criptografía. Regev demuestra que LWE no es difícil de manera aislada, sino que su dificultad se hereda de problemas geométricos para los que existe una evidencia sólida de dureza en alta dimensión.

El resultado de Regev puede enunciarse en términos muy generales de la siguiente manera:

Resultado (Regev, 2005). Existe una reducción probabilística en tiempo polinómico desde versiones aproximadas de problemas de retículos en el peor caso —en particular *GapSVP* y *SIVP*— al problema *Decision-LWE*, bajo parámetros apropiados [9].

Este enunciado contrasta dos niveles distintos: problemas de retículos formulados en el peor caso; y LWE como problema promedio generado aleatoriamente. La fuerza de la reducción está precisamente en unir ambos planos.

Para entender el alcance de esta afirmación conviene detenerse en la cadena de ideas que subyace a la reducción, sin entrar en su demostración técnica.

Aunque el resultado suele enunciarse mencionando tanto *GapSVP* como *SIVP*, aquí conviene seguir la intuición de *SIVP*, porque conecta de forma más natural con la geometría desarrollada antes.

El punto de partida, por tanto, es el problema *SIVP* (*Shortest Independent Vectors Problem*), que generaliza SVP al exigir no un único vector corto, sino una familia de vectores linealmente independientes todos ellos de longitud controlada. Se trata de un problema geométrico en el que hay que capturar estructura en varias direcciones simultáneamente.

Bastará con recorrer la serie de conexiones a alto nivel:

- Se parte de una instancia de *SIVP* en el peor caso: dado un retículo arbitrario, el objetivo es encontrar una familia de vectores cortos linealmente independientes. No se busca solo un accidente geométrico, sino información global sobre varias direcciones del retículo al mismo tiempo.

- Esa dificultad se transforma en un problema de **muestreo gaussiano discreto**: generar vectores del retículo según una distribución que favorece los vectores cortos, pero sin perder la estructura global del retículo.

- Finalmente, ese muestreo se conecta con muestras de LWE. La información geométrica queda codificada en ecuaciones lineales con ruido, de modo que resolver *Decision-LWE* permitiría recuperar información suficiente para resolver el problema reticular inicial.

La reducción requiere además que el ruido tenga una magnitud suficiente. Concretamente, debe superar lo que se conoce como el **parámetro de suavizado** $\eta_\varepsilon(\mathcal{L})$ del retículo dual asociado. Este parámetro determina a partir de qué escala la distribución gaussiana sobre el retículo dual induce una distribución casi uniforme al reducir módulo q. Esta propiedad es esencial para que las muestras resultantes sean indistinguibles de ruido uniforme, que es precisamente la hipótesis subyacente en Decision-LWE.

La cadena completa puede resumirse, a nivel conceptual, como

$$\text{SIVP (peor caso)} \longrightarrow \text{muestreo gaussiano discreto} \longrightarrow \text{Decision-LWE (caso promedio)}.$$

La primera flecha traduce información geométrica sobre vectores cortos en una distribución probabilística sobre el retículo. La segunda convierte esa distribución en muestras algebraicas con ruido.

Esto permite formular una idea que atraviesa toda la criptografía basada en retículos.

Si existiera un algoritmo eficiente capaz de resolver *Decision-LWE* en caso promedio (es decir, una instancia aleatoria) con ventaja no despreciable, entonces ese mismo algoritmo podría utilizarse —mediante la reducción— para resolver instancias arbitrarias de *GapSVP* o *SIVP* en el peor caso.

Y aquí es donde encaja con todo lo anterior. Por un lado, no se conocen algoritmos clásicos eficientes para resolver *Decision-LWE* bajo parámetros criptográficos; por otro, estos problemas de retículos no presentan la estructura algebraica que permitió a Shor romper la factorización o el logaritmo discreto, por lo que tampoco algoritmos cuánticos conocidos que alteren cualitativamente su complejidad. Su dificultad descansa en una transferencia rigurosa de dificultad desde problemas geométricos ampliamente considerados intratables.

No hace falta reproducir aquí la prueba de Regev para que esa conclusión sea útil, sino entender qué tipo de garantía ofrece. No es una demostración absoluta de seguridad, porque sigue dependiendo de hipótesis de dureza sobre problemas de retículos. Pero sí proporciona una base teórica mucho más robusta que la disponible en muchos otros sistemas criptográficos.

Esta robustez es la que explica por qué LWE se ha convertido en el núcleo de muchas de las propuestas de criptografía post-cuántica. Aun así, la reducción no fija los parámetros concretos: solo garantiza que, si existen, son seguros. Determinar qué valores de n, q y α satisfacen simultáneamente seguridad y corrección del descifrado es la pregunta que aborda el apartado siguiente, y cuya respuesta empírica se observará en los experimentos del Capítulo 7.

Con esta pieza en su sitio, la discusión puede bajar de nuevo al nivel de los parámetros concretos. Los siguientes apartados se basarán en entender cómo esa dureza depende del tamaño del ruido, del módulo y de la dimensión, y cómo esos parámetros se traducen en niveles efectivos de seguridad frente a los mejores ataques conocidos.

5.5. Parámetros y nivel de seguridad

Hasta ahora la discusión de LWE ha sido cualitativa: existe un error, existe un secreto, existe una estructura oculta. Para construir esquemas criptográficos reales es imprescindible ser cuantitativo. Los parámetros no pueden elegirse libremente; deben seleccionarse con cuidado para equilibrar seguridad y eficiencia, y esa elección tiene consecuencias directas sobre lo que puede o no puede hacer un adversario.

El papel central lo desempeña la relación entre tres magnitudes: la desviación estándar del error σ, el módulo q y la dimensión n. Una forma habitual de expresar la intensidad relativa del ruido es introducir el cociente

$$\alpha = \frac{\sigma}{q},$$

donde σ mide la escala del error y q fija la escala modular. Esta notación es especialmente útil en el análisis teórico de LWE, aunque los esquemas prácticos suelen emplear distribuciones discretas específicas y parámetros ajustados al diseño concreto. Es α el que finalmente determina el nivel de dificultad: q fija la escala aritmética global y σ fija la amplitud típica de la perturbación. El cociente entre ambos, codificado en α, indica cuánta estructura lineal sigue siendo visible tras el ruido.

Para entender por qué, conviene considerar tres regímenes extremos.

Régimen de error pequeño: $\alpha \to 0$**.** Si el error es muy pequeño en relación a q, las ecuaciones $b = As + e \pmod q$ se aproximan a $b \approx As \pmod q$. La perturbación es casi invisible: mediante técnicas de álgebra lineal sobre $\mathbb{Z}_q$ es posible recuperar s con un número polinómico de muestras. No hace falta que el ruido desaparezca por completo; basta con que sea suficientemente pequeño en comparación con q para que la instancia caiga en una zona peligrosa. Formalmente, esto corresponde al régimen

$$\alpha \ll \frac{1}{\sqrt{n}},$$

donde la estructura lineal del sistema sigue siendo explotable.

Régimen de error grande: $\alpha \to 1$**.** Si el error es casi tan grande como q, entonces $b = As + e \pmod q$ se aproxima a una distribución uniforme en $\mathbb{Z}_q$: el ruido ahoga el sistema. Aunque esto maximiza la indistinguibilidad, también destruye la utilidad criptográfica; las muestras no contienen información sobre el secreto y son indistinguibles de ruido puro.

Régimen criptográfico: α **intermedio.** Aquí el error es lo suficientemente grande para destruir la resolubilidad lineal directa, pero lo suficientemente pequeño para que el secreto siga dejando rastros en las ecuaciones. Es en este régimen donde se sitúan todos los

esquemas criptográficos prácticos, y corresponde a valores

$$\alpha \gtrsim \frac{1}{\sqrt{n}}.$$

El error está calibrado para que su magnitud sea comparable a la distancia entre puntos del retículo dual —lo que formalmente se expresa exigiendo que αq supere el parámetro de suavizado $\eta_\varepsilon(L^*)$ del retículo dual asociado, condición que garantiza que las muestras resulten indistinguibles de ruido uniforme sin conocer el secreto—. Al combinar las ecuaciones no se puede eliminar el ruido sin perder también la información del secreto; no se puede extraer el secreto sin que el ruido crezca hasta volverse dominante. Aquí se concentra el compromiso entre corrección, eficiencia y resistencia frente a los ataques conocidos.

En términos geométricos, el parámetro α determina hasta qué punto la perturbación puede desplazar una observación dentro de la estructura de celdas del retículo dual: si es demasiado pequeño, la observación no cambia de celda y la estructura sigue siendo visible; si es demasiado grande, puede cruzar múltiples celdas y la información original se pierde.

Los parámetros seleccionados en la práctica para esquemas como ML-KEM y ML-DSA se encuentran siempre en este régimen intermedio. Se elige q polinómico en n y el error sigue una distribución discreta gaussiana con desviación típica pequeña respecto de q, pero no despreciable. En implementaciones modernas es habitual trabajar con dimensiones del orden de $n \in \{512, 1024\}$, aunque el valor concreto depende del nivel de seguridad buscado y de si se trabaja con LWE estándar, Module-LWE o Ring-LWE.

La seguridad está directamente vinculada a n. A medida que n crece, aumenta el espacio del secreto, crece la dimensión de los retículos asociados y los mejores ataques conocidos —principalmente BKZ con tamaño de bloque variable— encarecen su tiempo exponencialmente.

Cuando se afirma que una instancia ofrece 128 bits de seguridad, lo que se quiere decir es que el mejor ataque conocido exige del orden de 2^{128} operaciones elementales. Como referencia aproximada, en instancias de LWE estándar, los niveles de seguridad dependen de la dimensión, del módulo, de la distribución del error y del estimador de ataque utilizado; por ello, no existe una correspondencia universal entre un valor de n y un nivel de seguridad concreto. En construcciones conservadoras como FrodoKEM, los parámetros utilizados son $n = 640$, $n = 976$ y $n = 1344$ para niveles asociados a AES128, AES192 y AES256, respectivamente.

En variantes estructuradas como Module-LWE, la situación cambia de forma sustancial. En ML-KEM, por ejemplo, el grado polinomial permanece fijo en $n = 256$, y la seguridad se obtiene ajustando el rango k, la distribución de error y otros parámetros del esquema. Por tanto, cualquier afirmación que relacione directamente dimensión y nivel de seguridad debe entenderse como una estimación dependiente del esquema y del modelo

de ataque.

Sin embargo, aumentar el nivel de seguridad no es gratuito. Incrementar n encarece tanto los ataques como la ejecución del propio esquema, elevando el coste computacional, el tamaño de las claves y la comunicación. Esa relación entre dimensión y seguridad es robusta frente a mejoras algorítmicas moderadas, pero vulnerable a cambios cualitativos: un algoritmo sustancialmente mejor que BKZ desplazaría los umbrales y obligaría a recalibrar los parámetros. Eso es precisamente lo que hace que el supuesto de dureza de LWE sea menos demostrable en sentido absoluto, pero mucho más creíble que la dureza del logaritmo discreto o la factorización, para los cuales sí existe demostración cuántica de vulnerabilidad.

5.5.1. Ataques conocidos

El régimen criptográfico descrito en la sección anterior —con α suficientemente grande para destruir la resolubilidad lineal directa, pero suficientemente pequeño para preservar la información necesaria para el descifrado— es precisamente aquel en que los mejores ataques conocidos alcanzan costes exponenciales en función de n.

Aunque se asume que LWE es difícil, existen varios enfoques algorítmicos que, bajo parámetros especiales o instancias débiles, pueden explotar su estructura. Conocer estos ataques es esencial para calibrar los parámetros y para entender dónde reside la dureza del problema. En la práctica, tres familias concentran la mayor parte del análisis: los ataques primales, los duales y los combinatorios de tipo BKW.

Ataque primal. El ataque más directo consiste en reducir la instancia de LWE mediante técnicas de reducción de bases, como BKZ, para obtener una base más *informativa* y, sobre esa base reducida, se intenta resolver una instancia de CVP o BDD que permita recuperar el secreto s o una información equivalente.

De modo inteligible, el adversario construye, a partir de la matriz pública A y las observaciones b, un retículo cuyo problema de descodificación reproduce la recuperación del secreto, donde s aparece como un vector corto.

El coste de esta estrategia depende de forma muy sensible de la dimensión y del tamaño del error: cuando el ruido aumenta, la instancia se aleja del régimen en el que esa reconstrucción geométrica puede explotarse directamente. Por eso la complejidad del ataque primal no viene dada por un único parámetro aislado: depende de la interacción entre n, q, σ y la calidad de reducción que BKZ pueda alcanzar con recursos realistas.

Ataque dual. Un segundo enfoque explota el retículo dual $\mathcal{L}_q^{\perp}(A)$. En lugar de intentar recuperar directamente s, se buscan vectores cortos en el dual que permitan proyectar las muestras sobre direcciones donde la contribución del secreto desaparece.

Recordemos que para cualquier $y \in \mathcal{L}_q^{\perp}(A)$, se satisface

$$\langle y, b \rangle \equiv \langle y, e \rangle \pmod{q}.$$

El secreto desaparece. Si se encuentra un vector dual corto y, entonces la cantidad $\langle y, b \rangle$ (mód q) contiene solo información sobre el error.

El ataque procede proyectando todas las muestras sobre vectores duales cortos y acumulando información sobre el error. Si el error no está bien calibrado —si, por ejemplo, es demasiado pequeño— estas proyecciones revelan una pequeña desviación respecto de lo que sería ruido completamente aleatorio —como se ha comentado en el apartado Subsección 5.2. Con eso ya es suficiente para romper la versión decisional del problema; y, como ambas versiones son equivalentes, también termina afectando a la recuperación del secreto.

Con parámetros criptográficos bien elegidos, este ataque requiere también resolver BDD o SVP en el retículo dual, lo que es exponencial en n.

Algoritmo BKW. Existe un tercer ataque de naturaleza muy distinta, el algoritmo Blum–Kalai–Wasserman (BKW), que resulta conceptualmente interesante porque muestra que LWE no solo puede atacarse desde la geometría de retículos, sino también de técnicas combinatorias.

BKW procede combinando ecuaciones LWE para eliminar progresivamente variables. Aunque es elegante teóricamente, en la práctica el crecimiento del ruido tras muchas combinaciones impide que el ataque se vuelva competitivo de manera general frente a las estrategias basadas en reducción de bases. BKW resulta relevante solo en regímenes donde el módulo q es muy pequeño o donde el número de muestras es extraordinariamente alto — situaciones que no ocurren en criptografía post-cuántica estándar. En los parámetros de Kyber o Dilithium, los ataques basados en reducción de retículos —especialmente los que se estiman mediante BKZ— suelen marcar el es el ataque dominante.

En conjunto, todos estos ataques desembocan, para parámetros criptográficos estándar, en costes exponenciales —o subexponenciales muy elevados en algunos regímenes— en función de la dimensión y de la calibración del error. Esa constatación no equivale a una prueba absoluta de seguridad (nunca lo hace). Sí proporciona, sin embargo, el tipo de evidencia que realmente importa en este contexto: ataques de naturaleza muy distinta han sido estudiados durante años y todos tropiezan con la misma barrera paramétrica.

En conclusión, no se conocen algoritmos clásicos que resuelvan LWE en tiempo polinómico; tampoco se conoce algoritmo cuántico que altere sustancialmente este panorama. Esa convergencia es lo que sitúa a LWE como candidato tan robusto para criptografía post-cuántica.

5.6. Variantes estructuradas

A pesar de su solidez teórica, LWE presenta un problema importante cuando se traslada a construcciones criptográficas reales: el coste. Las matrices involucradas tienen tamaño creciente con la dimensión y, en consecuencia, tanto las claves como las operaciones básicas terminan siendo relativamente pesadas.

Las operaciones requeridas para cifrar, descifrar y generar claves crecen en complejidad cuadrática con la dimensión n. En LWE estándar, trabajar con matrices de tamaño $m \times n$, que implica que para dimensión $n = 1024$, una clave pública requiere almacenar una matriz de $\sim 1024 \times 1024$ elementos en $\mathbb{Z}_q$, lo que representa cientos de kilobytes. Eso es prohibitivo en muchas aplicaciones prácticas.

La solución natural es introducir estructura algebraica. En lugar de trabajar con vectores arbitrarios sobre $\mathbb{Z}_q$, se trabaja con elementos de un anillo cociente. Esta especialización, aunque introduce supuestos adicionales sobre la dureza del problema, permite mejoras de eficiencia dramáticas.

5.6.1. Ring-LWE

La idea de *Ring-LWE* es reemplazar los vectores en $\mathbb{Z}_q^n$ y matrices en $\mathbb{Z}_q^{m \times n}$ de LWE estándar por objetos algebraicos definidos sobre anillos de polinomios. En lugar de trabajar en espacios del tipo $\mathbb{Z}_q^n$, se trabaja en el anillo cociente

$$R_q = \mathbb{Z}_q[x]/(f(x)),$$

es decir, el conjunto de polinomios con coeficientes en $\mathbb{Z}_q$, donde las operaciones se realizan módulo q y donde dos polinomios se consideran equivalentes si difieren en un múltiplo de $f(x)$. El polinomio $f(x)$ suele elegirse como un polinomio ciclotómico, típicamente

$$f(x) = x^n + 1,$$

para valores de n potencia de dos.

Por tanto, los vectores pasan a interpretarse como polinomios y las operaciones lineales se convierten en multiplicaciones polinómicas módulo $f(x)$ y módulo q, que sigue pudiendo interpretarse, si se quiere, un vector de coeficientes

La ecuación básica de LWE,

$$b = As + e \pmod{q},$$

adopta entonces una forma más compacta:

$$b = a \cdot s + e \in R_q,$$

donde a, s, e y b son ahora polinomios en R_q.

Conceptualmente, el cambio no elimina el ruido ni altera la intuición geométrica del problema. Sigue existiendo una relación algebraica perturbada por un error pequeño. Lo que cambia es la estructura sobre la que esa relación se construye.

La ventaja práctica es enorme. Las multiplicaciones de polinomios pueden acelerarse mediante técnicas como la *Number Theoretic Transform* (NTT), análoga discreta de la Transformada Rápida de Fourier. Gracias a ello, operaciones que en LWE estándar tendrían un coste aproximadamente cuadrático pasan a ejecutarse en complejidad

$$O(n \log n).$$

Esta mejora afecta directamente a los tres elementos críticos en criptografía práctica: tamaño de claves, velocidad de cifrado y coste computacional de descifrado y generación de firmas.

Ring-LWE asume que la estructura algebraica del anillo no proporciona ataques más rápidos que los disponibles para LWE general y, hasta la fecha, no se han encontrado ataques que exploten significativamente esa estructura. Aun así, tampoco existe una prueba de que Ring-LWE sea tan duro como LWE. Es una conjetura, afortunadamente, una conjetura que ha resistido años de escrutinio y análisis por criptógrafos.

Ring-LWE es el paso conceptual a Module-LWE, donde se forma la base de esquemas modernos como Kyber (ahora FIPS 203 tras la estandarización del NIST bajo el nombre ML-KEM) y de la firma Dilithium (ML-DSA). Su adopción en estándares es un claro indicador de que la comunidad criptográfica considera que la mejora de eficiencia compensa el supuesto adicional.

5.6.2. Module-LWE

Ring-LWE resuelve gran parte de los problemas prácticos de eficiencia presentes en LWE estándar, pero lo hace introduciendo una estructura algebraica considerablemente más rígida. Las instancias del problema dejan de construirse sobre espacios vectoriales arbitrarios y pasan a depender de un único anillo polinómico cociente. Esa regularidad adicional ha motivado históricamente el estudio de posibles ataques capaces de explotar propiedades específicas de dicha estructura.

Module-LWE aparece precisamente como un punto intermedio entre ambos extremos. Mantiene gran parte de la eficiencia algebraica de Ring-LWE, pero reduce el nivel de estructura impuesto sobre el problema.

La idea consiste en trabajar no sobre un único anillo, sino sobre módulos de dimensión pequeña definidos sobre dicho anillo. En lugar de que los secretos y muestras sean elementos individuales de

$$R_q,$$

pasan a ser vectores de longitud k con entradas en R_q:

$$R_q^k.$$

La ecuación de LWE recupera entonces una forma matricial similar a la original,

$$b = As + e,$$

pero ahora

$$A \in R_q^{m \times k}, \quad s \in R_q^k,$$

Por lo tanto, cuando $k = 1$, Module-LWE coincide esencialmente con Ring-LWE; a medida que k aumenta, el problema se aproxima progresivamente a LWE estándar. Dicho de otra forma, Module-LWE permite interpolar entre máxima eficiencia y mínima estructura algebraica adicional.

Las operaciones continúan beneficiándose de la NTT y de las multiplicaciones rápidas de polinomios, manteniendo costes cercanos a

$$O(n \log n).$$

Los estándares actuales seleccionados por el NIST utilizan precisamente esta variante: ML-KEM (Kyber) para intercambio de claves y ML-DSA (Dilithium) para firmas digitales se construyen sobre problemas de tipo Module-LWE y Module-SIS. Aunque la dureza de Module-LWE sigue siendo una hipótesis criptográfica y no una demostración absoluta de seguridad, los parámetros seleccionados actualmente proporcionan niveles estimados de 128, 192 y 256 bits de seguridad clásica frente a los mejores ataques conocidos —principalmente variantes de BKZ combinadas con técnicas híbridas y duales— sobre los parámetros concretos del esquema.

Module-LWE añade una segunda idea sobre Ring-LWE: esa estructura puede replicarse varias veces y organizarse en forma matricial sin volver al coste bruto de LWE estándar. Una vez entendida la dureza abstracta del problema y las variantes estructuradas que permiten hacerlo viable en la práctica, el siguiente paso consiste en estudiar cómo toda esta teoría se convierte en esquemas criptográficos e implementaciones reales.

5.6.3. Nota sobre supuestos de seguridad

Las tres variantes — LWE, Module-LWE, Ring-LWE — viven en un espectro:

Variante	**Supuesto de seguridad**	**Eficiencia**	**Estándar**
LWE	Más conservador	$O(n^2)$	–
Ring-LWE	Más estructurado	$O(n \log n)$	–
Module-LWE	Intermedio	$O(nk \log n)$	NIST

Nota. En los estándares actuales, el parámetro k es pequeño y constante respecto del grado polinomial. En ML-KEM, por ejemplo, se toman valores $k \in \{2, 3, 4\}$. Por ello, el factor k no cambia el orden asintótico respecto de n, aunque sí afecta a las constantes y al tamaño final de claves y criptogramas.

La elección entre variantes no es una cuestión matemática, sino una cuestión de ingeniería criptográfica: qué nivel de estructura algebraica adicional estamos dispuestos a asumir y qué coste computacional estamos dispuestos a evitar. En términos generales, cuanto mayor es la estructura disponible, mayores son también las oportunidades para obtener implementaciones eficientes, aunque a costa de formular hipótesis de seguridad más específicas.

En la práctica moderna, Module-LWE representa la alternativa predominante, asumiendo una estructura algebraica adicional respecto a LWE estándar —aunque manteniendo un supuesto considerado suficientemente robusto— a cambio de una ganancia de eficiencia muy significativa.

6. Criptografía basada en LWE

Los resultados de dureza estudiados anteriormente explican por qué bajo ciertos parámetros, un algoritmo eficiente para resolver LWE en caso promedio permitiría resolver problemas difíciles de retículos en el peor caso. Pero aún no especifica cómo usar esa dificultad para proteger información, generar claves compartidas o autenticar mensajes mediante firmas digitales. Entre la dureza del problema y un esquema criptográfico funcional hay un abismo de decisiones: qué parámetros elegir, cómo estructurar la información, cómo hacer que el sistema sea eficiente sin perder seguridad.

Esta sección recorre ese camino. Se comenzará con construcciones conceptualmente simples —un esquema de cifrado básico— que ilustra cómo LWE se convierte en confidencialidad. De ahí se pasará a problemas más sofisticados: acordar claves con un adversario escuchando, firmar mensajes sin revelar el secreto. Finalmente, se presentarán los esquemas reales y estandarizados: ML-KEM [6] —basado en Kyber— para intercambio de claves, ML-DSA [7] —basado en Dilithium— para firmas digitales, y FrodoKEM [2] como alternativa más conservadora.

No se trata de reproducir las construcciones en detalle completo —eso corresponde a documentación técnica—, sino de mostrar cómo la teoría anterior se convierte en herramientas que protegen datos en sistemas reales.

6.1. Esquema de cifrado básico

El caso más sencillo es el de Alice y Bob: Alicia quiere enviar un bit de información a Bob de forma que ningún adversario puede saber qué bit es, incluso si escucha todo lo que se transmite. En criptografía clásica, esto se resolvería con RSA o ECC. Ahora, queremos hacerlo con LWE.

La construcción es la siguiente.

Generación de claves

Bob elige un secreto $s \in \mathbb{Z}_q^n$ que será su clave privada. A partir de él, genera una clave pública mediante LWE: elige una matriz pública $A \in \mathbb{Z}_q^{m \times n}$ tomada al azar de manera uniforme, y computa

$$b = As + e \pmod{q},$$

donde $e \in \mathbb{Z}^m$ es un vector de errores pequeños según la distribución elegida. La clave pública es el par (A, b); la clave secreta es s.

La distribución de la información es, por tanto, asimétrica. Bob conserva s como clave privada. Alicia recibe únicamente la clave pública (A, b), suficiente para cifrar. El adversario puede observar esa misma clave pública y también el texto cifrado final, pero no dispone del secreto que permite eliminar la parte lineal del cálculo.

La seguridad de esta construcción ya reposa en la dificultad de LWE: aunque Alicia (o un adversario) conocen A y b, no pueden recuperar s sin resolver una instancia del problema, cuya dificultad ha sido explicada anteriormente.

Cifrado

Alicia quiere enviar un bit $\mu \in \{0, 1\}$. Realiza lo siguiente:

Sea, por ejemplo,

$$r \in \{0, 1\}^m$$

un vector efímero que permite combinar varias muestras públicas en un único texto cifrado. A partir de él se construyen

$$u = A^\top r \pmod{q}$$

y

$$v = b^\top r + \mu \left\lfloor \frac{q}{2} \right\rfloor \pmod{q}.$$

El texto cifrado es el par $(u, v) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. La única diferencia entre cifrar 0 y cifrar 1 es el desplazamiento por aproximadamente $q/2$. Eso basta, siempre que el ruido total permanezca bastante por debajo de esa separación.

Descifrado

Bob, que conoce s, computa

$$v - \langle u, s \rangle \pmod{q}.$$

La razón se ve al expandir la expresión de v. Como

$$b = As + e,$$

trabajando siempre módulo q, se obtiene

$$v = (As + e)^\top r + \mu \left\lfloor \frac{q}{2} \right\rfloor = s^\top A^\top r + e^\top r + \mu \left\lfloor \frac{q}{2} \right\rfloor.$$

Pero $A^\top r = u$, de modo que el receptor, que conoce s, puede calcular

$$v - s^\top u \equiv e^\top r + \mu \left\lfloor \frac{q}{2} \right\rfloor \pmod{q}.$$

Si $\mu = 0$, esta cantidad queda cerca de 0, salvo por el error acumulado. Si $\mu = 1$, esta

cantidad es aproximadamente $q/2$ más el error. Así:

$$v - \langle u, s \rangle \approx \begin{cases} \text{pequeño} & \text{si } \mu = 0 \\ q/2 + \text{pequeño} & \text{si } \mu = 1 \end{cases}$$

Bob simplemente comprueba si el resultado está más cerca de 0 o de $q/2$. Si está más cerca de 0, el mensaje es 0; si está más cerca de $q/2$, es 1.

El descifrado consiste entonces en una decisión por umbral: no se intenta eliminar por completo el ruido, sino verificar que sigue siendo lo bastante pequeño como para no confundir ambas regiones. Para ello, el parámetro q debe ser suficientemente grande y la distribución del error suficientemente estrecha como para que el término

$$e^\top r$$

no desplace la observación a la región equivocada.

¿Por qué funciona?

El adversario conoce el procedimiento de descifrado. Sabe que, si pudiera calcular

$$v - \langle u, s \rangle,$$

bastaría con comprobar si el resultado queda cerca de 0 o de $q/2$ para recuperar el bit. El problema es que esa operación requiere conocer s. Sin ese secreto, solo observa u y v como valores modulares aparentemente mezclados con ruido.

Sin conocer s, un adversario observa (A, b, u, v). Para recuperar el mensaje, tendría que:

- O bien resolver LWE sobre (A, b) para obtener s,
- O bien distinguir entre (u, v) cifrado de 0 y cifrado de 1.

La primera opción es computacionalmente intratable bajo nuestras suposiciones. La segunda opción no es exactamente Decision-LWE escrito en la misma forma, pero se formaliza mediante un argumento de reducción: distinguir entre una instancia de LWE y una distribución uniforme es precisamente lo que se asume difícil.

Parámetros típicos para este esquema:

Los parámetros no pueden elegirse de forma independiente. La dimensión n controla el tamaño del secreto y la dificultad de los ataques conocidos; el número de muestras m determina cuánta información pública se ofrece sobre ese secreto; el módulo q fija la escala en la que se separan los mensajes codificados; y la distribución de error χ debe situarse en un margen estrecho: suficientemente grande para ocultar la estructura lineal, pero suficientemente pequeña para permitir el descifrado correcto.

Fijar valores numéricos en este punto sería prematuro: los parámetros solo adquieren sentido una vez fijado el esquema concreto y el nivel de seguridad que se pretende alcanzar.

Conexión con lo anterior:

La seguridad, en definitiva, no proviene del procedimiento de cifrado en sí, sino de que recuperar s exige resolver una instancia LWE cuya dificultad está anclada en lo del 5.4. Lo que sigue es mostrar cómo esa misma estructura se convierte en un protocolo de establecimiento de claves.

Este es el punto donde se justifica la teoría abstracta introducida en las secciones anteriores.

Aún así, este esquema cumple una función principalmente explicativa. En la práctica moderna, la aplicación dominante no es cifrar directamente mensajes largos con LWE, sino establecer claves compartidas mediante mecanismos de encapsulación y utilizar después criptografía simétrica.

6.2. Intercambio de claves

El esquema anterior resuelve confidencialidad de un único bit. Pero entre eso y la criptografía real hay un salto: Alice y Bob necesitan acordar una clave simétrica compartida —típicamente 256 bits para AES— que después usarán para cifrar el tráfico de verdadero volumen. El cifrado asimétrico es costoso; se usa para proteger la negociación, no los datos.

En criptografía clásica, Diffie-Hellman lo resuelve mediante exponenciación en grupos cíclicos: una operación fácil de calcular en una dirección, pero difícil de invertir sin resolver el logaritmo discreto. Esa es precisamente la parte que Shor compromete en el modelo cuántico.

Escenario: Bob publica, Alice encapsula

El protocolo no es simétrico entre Alice y Bob. Bob actúa primero, como custodio de una instancia LWE pública.

Bob elige una matriz pública

$$A \in \mathbb{Z}_q^{m \times n},$$

un secreto privado

$$s \in \mathbb{Z}_q^n,$$

un vector de error pequeño

$$e \in \mathbb{Z}^m,$$

y publica

$$b = As + e \pmod{q}.$$

Esto es exactamente lo de la subsección anterior: una instancia LWE. El secreto s

permanece privado. Todo el mundo conoce (A, b).

Cuando Alice quiere establecer una clave compartida con Bob, toma la instancia pública de Bob y la *tienta* con aleatoriedad propia. Elige un vector corto (típicamente binario)

$$r \in \{0,1\}^m$$

y errores pequeños adicionales

$$e_1 \in \mathbb{Z}^n, \qquad e_2 \in \mathbb{Z}.$$

Con esto, calcula

$$u = A^\top r + e_1 \pmod q,$$
$$v = b^\top r + e_2 \pmod q,$$

y envía (u, v) a Bob.

Aquí no hay secreto nuevo de Alice que comunique, sino una aleatoriedad que codifica indirectamente su participación. Lo importante ahora es que (u, v) codifica información sobre r de forma ruidosa, el secreto de Alice.

¿Cómo derivan la clave?

Bob, que conoce s, computa

$$\text{Bob's candidate} = s^\top u = s^\top (A^\top r + e_1) = s^\top A^\top r + s^\top e_1.$$

Alice no conoce s, pero puede usar el valor v que ella misma construyó:

$$\text{Alice's candidate} = v = b^\top r + e_2 = (As + e)^\top r + e_2 = s^\top A^\top r + e^\top r + e_2.$$

Expandamos la diferencia:

$$\text{Alice's} - \text{Bob's} = (e^\top r + e_2) - s^\top e_1.$$

Con parámetros correctamente calibrados —es decir, con el ruido ajustado al régimen criptográfico estudiado en la Sección anterior—, esa diferencia permanece acotada y pequeña. No es cero, pero está controlada, así que ambos valores viven en una vecindad bien definida alrededor de $s^\top A^\top r$.

Y aquí es donde aparece lo distinto respecto a Diffie-Hellman: en ese protocolo clásico, ambas partes derivan exactamente el mismo valor algebraico; en LWE no necesitan ser idénticos, solo necesitan estar lo bastante cerca.

El mecanismo de reconciliación

Alice y Bob no extraen la clave directamente de esos valores cercanos. Hacerlo sería

arriesgado: si Alice redondea hacia abajo y Bob redondea hacia arriba (o al revés), obtienen claves distintas, y el protocolo colapsa.

Lo que hacen es una operación de *reconciliación* o *cross-rounding*.

Para que Alice y Bob se sincronicen, necesitan un mecanismo de reconciliación. Bob publica una ayuda muy pequeña: h, un vector donde cada coordenada codifica dónde redondear (típicamente, un solo bit por coordenada de $s^\top A^\top r$). Esa información es pública pero, sin conocer el verdadero valor, carece de significado. En esquemas prácticos como Kyber, esta ayuda se optimiza al formar parte del propio encapsulado —enviado por Alice— o al incorporarse en el procedimiento de redondeo y compresión.

Con eso, redondean hacia la misma región y ambos terminan con la misma cadena de bits. La ayuda es pública y parece información sin valor, pero su razón de ser es coordinar dos observaciones ruidosas desde posiciones distintas del protocolo.

Esa cadena de bits actúa como material de clave: normalmente se procesa mediante una función de derivación para obtener una clave simétrica de sesión, que después se usará con algoritmos como AES. Así, la parte basada en LWE solo resuelve el establecimiento inicial del secreto compartido; el cifrado masivo de datos queda en manos de la criptografía simétrica.

El protocolo completo —(A, b, h) publicado por Bob, (u, v) enviado por Alice— no es un intercambio simétrico, es una encapsulación: Alice encapsula un secreto en la instancia pública de Bob, y solo Bob puede desencapsularlo.

Ejemplo. Para concretar la necesidad de reconciliación, considérese $q = 17$, con frontera de decisión entre los valores 8 y 9, y tomemos el caso escalar $n = m = 1$. Sea $s = 3$ el secreto de Bob, $A = 3$ y $e = 0$, de modo que

$$b = As + e = 9 \pmod{17}.$$

Alice elige $r = 1$, $e_1 = 0$ y $e_2 = -1$. Entonces

$$u = A^\top r + e_1 = 3, \qquad v = b^\top r + e_2 = 9 - 1 = 8.$$

El valor algebraico común alrededor del cual deberían situarse ambas partes es

$$k = s^\top A^\top r = 9.$$

Bob calcula

$$s^\top u = 3 \cdot 3 = 9,$$

que cae en la región superior $[9, 16]$, y por tanto redondearía al bit 1. Alice, en cambio, dispone de

$$v = 8,$$

que cae en la región inferior $[0, 8]$, y redondearía al bit 0. Una diferencia de una sola unidad, producida por el ruido, ha situado ambas observaciones en lados distintos de la frontera de decisión. Sin reconciliación, las claves derivadas serían distintas.

Este ejemplo muestra por qué no basta con que los valores de Alice y Bob sean cercanos: deben redondearse de forma coherente. La información auxiliar h no debe entenderse como una revelación directa del bit de clave, sino como una ayuda pública de redondeo que permite corregir discrepancias pequeñas entre dos observaciones ruidosas del mismo valor subyacente.

¿Qué ven los adversarios?

Eve escucha la clave pública (A, b), el encapsulado (u, v) y h, es decir, los valores públicos que intervienen en la comunicación; pero no conoce el secreto privado s.

Para derivar la clave, Eve tendría que:

- Recuperar el secreto privado s a partir de la clave pública (A, b) (lo que equivale a resolver la instancia LWE subyacente), o
- Distinguir la clave encapsulada de una clave aleatoria (que es precisamente la propiedad de indistinguibilidad que se apoya en la hipótesis *Decision-LWE*), o
- Aprovechar (u, v) para computar información sobre r sin resolver LWE en (A, b). Pero esto equivale a distinguir $b^\top r$ de un valor uniforme cuando se cubre de ruido moderado (nuevamente, Decision-LWE).

Las tres vías son, bajo los supuestos considerados y con parámetros adecuados, computacionalmente inasumibles. La seguridad nace nuevamente de la dificultad de separar la parte algebraica del ruido sin conocer las variables ocultas correctas.

Alice conoce r, Bob conoce s, y cada uno utiliza su propia información privada para llegar a valores cercanos. Eve solo observa (A, b, u, v); sin r ni s, no puede reproducir ninguno de los dos cálculos que permiten derivar la clave compartida.

6.3. Firmas digitales

Mientras que cifrado e intercambio de claves protegen confidencialidad, las firmas digitales resuelven un problema distinto: autenticidad. Ya no se trata de esconder nada, sino de demostrar que un mensaje procede realmente de quien dice haberlo emitido y que no ha sido modificado en tránsito.

En RSA, se cifra un hash del mensaje con la clave privada. Aquí, la estructura es más sofisticada porque LWE no tiene una operación "inversa" directa como la exponenciación, aunque tampoco es puramente LWE. Para firmas, los esquemas modernos se construyen mejor sobre problemas emparentados como Module-SIS: encontrar relaciones cortas que satisfacen una condición lineal pública. En otras palabras, el firmante demuestra conocer

un testigo pequeño compatible con ecuaciones públicas; el verificador solo comprueba que dichas ecuaciones se satisfacen y que los valores permanecen dentro de rangos previstos.

La primera intuición sería un esquema de tipo compromiso–reto–respuesta: Alicia publica primero un compromiso aleatorio, Bob le envía un reto y Alicia responde combinando ese reto con su secreto. Si la respuesta tiene la forma correcta, Bob queda convencido de que Alicia conoce algo que un impostor no podría producir. El problema es que eso describe un protocolo interactivo. Una firma digital necesita eliminar la interacción y convertir ese diálogo en un objeto autónomo verificable más tarde, sin la presencia del firmante.

Ahí entra la heurística de Fiat–Shamir. En lugar de recibir el reto de un verificador externo, el firmante lo deriva mediante una función hash —una huella digital del mensaje— aplicada al mensaje y al compromiso. El reto deja de venir del exterior y pasa a generarse de forma reproducible a partir de datos públicos. La firma queda compuesta por un compromiso, un reto implícito y una respuesta. Verificar consiste en recomputar ese reto, comprobar que la ecuación pública se satisface y asegurarse de que la respuesta es suficientemente pequeña. Esa pequeñez no es un detalle técnico secundario; es lo que liga la corrección del esquema con la dureza geométrica del problema subyacente.

Construcción via Fiat-Shamir

Dilithium (ahora ML-DSA) sigue esta estructura:

1. Alice tiene $s \in \mathbb{Z}_q^n$ (clave privada).
2. Para firmar un mensaje m, genera un vector aleatorio y y computa un compromiso

$$w = Ay \pmod{q}.$$

3. Por un lado, el desafío obtenido como

$$c = H(m, w),$$

   mediante una función hash criptográfica, siguiendo la transformación de Fiat–Shamir.

4. Por otro lado, una respuesta $z = y + c \cdot s$ (con ajustes para controlar magnitud y evitar que muchas firmas revelen información sobre s).
5. La firma, es decir, lo publicado, es (c, z).

Verificación:

Bob conoce la clave pública, formada por A y por un valor público t asociado al secreto, que puede pensarse de forma simplificada como

$$t = As \pmod{q}.$$

Al recibir una firma (c, z) sobre el mensaje m, verifica dos cosas:

1. Que $\|z\|$ sea suficientemente pequeño. Esta cota impide aceptar respuestas modulares arbitrarias y mantiene la conexión con la dificultad geométrica del problema.

2. Que el reto pueda reconstruirse correctamente. Como $z = y + cs$, se tiene

$$Az - ct = A(y + cs) - cAs = Ay.$$

   Por tanto, el verificador reconstruye el compromiso $w' = Az - ct$ y comprueba que

$$c = H(m, w')$$

   sin conocer ni y ni s.

La firma es válida cuando el reto recibido coincide con el reto que se obtiene al reconstruir el compromiso, cuando ambas condiciones se cumplen.

¿Por qué es seguro?
Falsificar una firma requeriría:

- Encontrar un z corto tal que satisfaga las ecuaciones públicas, sin conocer s (se relaciona con resolver SIS, un problema de aproximación sobre retículos).
- Calcular dos preimágenes diferentes del hash (colisión criptográfica).
- Explorar estadísticamente múltiples firmas para aproximar s (esto se evita mediante rejection sampling: el firmante reinicia las operaciones hasta comprobar que la respuesta no filtra información).

Los últimos dos son resistentes bajo supuestos estándar. El primero es computacionalmente intratable en alta dimensión.

Comparación de firmas digitales con RSA:

Aspecto	**RSA**	**Dilithium/ML-DSA**
Clave privada	Factores p, q	Vector $s \in \mathbb{Z}_q^n$
Firma	Cifrado del hash	Respuesta Fiat–Shamir
Tamaño firma	≈256 bytes	≈2.5 KB
Operación	Exponenciación	Mult. matriz-vector
Vulnerable a cuántica	Sí (Shor)	No (conjetura)

Las firmas de Dilithium son más grandes, pero ese es un costo aceptable por resistencia cuántica y por las garantías que proporciona el trasfondo geométrico de retículos.

Este cambio de objetivo –certificación en lugar de confidencialidad– es también el que justifica por qué firmas y encapsulación, aunque comparten el trasfondo reticular, requieren arquitecturas distintas. Mientras que en intercambio de claves basta sincronizar dos observaciones cercanas, en firma todo es público, salvo el testigo privado. El testigo privado es el vector secreto s: la información que permite generar firmas válidas, pero que nunca debe aparecer explícitamente en la verificación. Eso obliga a que la aritmética modular y el control de magnitud se organicen alrededor de una evidencia pública auditable sin ambigüedad. Es una exigencia más rígida.

6.4. Ejemplos reales

Llegados aquí, los nombres propios importan. No porque sustituyan a la teoría, sino porque muestran qué parte de toda la discusión anterior ha sobrevivido al contacto con restricciones reales: latencia, ancho de banda, tamaño de claves, simplicidad de implementación y margen de seguridad frente a ataques que siguen evolucionando. La criptografía post-cuántica no se decide solo en el plano de los supuestos. Se decide también en el terreno, menos elegante pero más decisivo, de qué puede desplegarse de verdad.

Tres ejemplos permiten ver bien ese paisaje. Dos de ellos —ML-KEM y ML-DSA— representan la línea que ha acabado cristalizando en los estándares actuales. El tercero —FrodoKEM— no terminó siendo estándar, pero sigue siendo una referencia importante porque encarna la alternativa más conservadora dentro de la familia basada en LWE.

6.4.1. Kyber / ML-KEM (FIPS 203)

ML-KEM, derivado de Kyber, es el estándar seleccionado por el NIST para mecanismos de encapsulación de claves basados en retículos. Su función no es cifrar directamente grandes cantidades de datos, sino establecer una clave compartida que después será utilizada por criptografía simétrica.

Desde el punto de vista de este trabajo, ML-KEM representa la materialización práctica de Module-LWE. La idea de fondo es la misma que se ha visto en el intercambio de claves: una parte publica una estructura algebraica ruidosa, la otra encapsula un secreto utilizando esa información, y solo quien posee la clave privada puede recuperar el material necesario para derivar la clave compartida. La diferencia está en el refinamiento. ML-KEM incorpora compresión, redondeo, selección cuidadosa de parámetros y transformaciones adicionales para alcanzar seguridad frente a ataques más fuertes que los considerados en el esquema básico.

Parámetros y costes:

Nivel de seguridad	Variante	Clave pública	Secreto compartido
128 bits (Cat. 1)	ML-KEM-512 (k=2)	800 bytes	32 bytes
192 bits (Cat. 3)	ML-KEM-768 (k=3)	1.184 bytes	32 bytes
256 bits (Cat. 5)	ML-KEM-1024 (k=4)	1.568 bytes	32 bytes

En ML-KEM, el grado polinomial n permanece fijo en 256. Lo que varía entre los tres niveles es principalmente el rango k del módulo, junto con otros parámetros de compresión y ruido.

Las claves públicas son comprimidas (gracias a la estructura algebraica y técnicas de cuantización); el secreto compartido es siempre 32 bytes (tamaño de una clave AES-256).

Complejidad computacional:

Las operaciones dominantes son multiplicaciones en R_q (anillo de polinomios), que se aceleran mediante NTT a coste $O(n \log n)$. En práctica, Kyber128 requiere del orden de cientos de microsegundos en procesadores modernos.

Estado:

Fue seleccionado por el NIST en agosto de 2024 como parte del estándar FIPS 203, lo que significa que se espera que sea ampliamente adoptado en criptografía post-cuántica. Su adopción se encuentra en una fase progresiva, especialmente en despliegues híbridos donde ML-KEM se combina con mecanismos clásicos para proteger las comunicaciones frente a ataques de tipo *harvest now, decrypt later*.

6.4.2. Dilithium / ML-DSA (FIPS 204)

ML-DSA, derivado de Dilithium, cumple otra función: firmas digitales. Su objetivo no es establecer un secreto compartido, sino permitir que cualquiera pueda verificar la autenticidad e integridad de un mensaje.

A diferencia de ML-KEM, que se apoya principalmente en Module-LWE, ML-DSA se entiende de forma más natural a partir de problemas de tipo Module-SIS y de técnicas de firma basadas en Fiat–Shamir.

Parámetros y costes:

Nivel de seguridad	Variante	Clave privada	Clave pública	Tamaño firma
128 bits	ML-DSA-44	2,5 KB	1,3 KB	2,4 KB
192 bits	ML-DSA-65	3,9 KB	1,9 KB	3,2 KB
256 bits	ML-DSA-87	4,8 KB	2,5 KB	4,5 KB

ML-DSA-44, ML-DSA-65 y ML-DSA-87 corresponden, respectivamente, a las categorías NIST 2, 3 y 5.

Las firmas son más grandes que RSA-2048 (256 bytes), pero sigue siendo práctico para la mayoría de aplicaciones.

La contrapartida práctica es evidente: las firmas y claves son mayores que en muchos esquemas clásicos —RSA-2048 (256 bytes)—, pero sigue siendo práctico para la mayoría de aplicaciones. Eso sí, no dependen de factorización ni de logaritmo discreto, que son precisamente los problemas comprometidos por el algoritmo de Shor.

Velocidad:

Firma: milisegundos. Verificación: cientos de microsegundos. Las diferencias respecto a RSA son pequeñas en práctica.

Estado:

Fue seleccionado por el NIST en agosto de 2024 como FIPS 204, estandarizándose para firmas digitales post-cuánticas. Aún en fase inicial de adopción

6.4.3. FrodoKEM

FrodoKEM ocupa una posición distinta. También es un mecanismo de encapsulación de claves, pero evita deliberadamente la estructura algebraica adicional de Ring-LWE o Module-LWE. Su construcción se basa en LWE estándar sobre matrices, sin organizar las operaciones sobre anillos o módulos de polinomios. Paga más para asumir menos estructura

Parámetros y costes:

Nivel seguridad	n	**Clave pública**	**Secreto compartido**
128 bits	640	9,4 KB	16 bytes
192 bits	976	15,3 KB	24 bytes
256 bits	1.344	21,0 KB	32 bytes

Las claves públicas son 10–40 veces más grandes que Kyber para el mismo nivel de seguridad.

¿Por qué existe si es menos eficiente?

Kyber depende de Module-LWE, que es un supuesto criptográfico más fuerte que LWE estándar. Si alguien descubriera un ataque específico a Ring-LWE (explotando estructura polinómica), Kyber sería vulnerable. FrodoKEM, en cambio, solo confía en LWE, que tiene menos estructura y ha sido más extensamente estudiado.

Esta es la opción para organizaciones que priorizan conservadurismo teórico sobre eficiencia práctica.

Estado:

No fue seleccionado por el NIST en la ronda principal, pero sigue siendo un candidato relevante y es usada en algunos contextos donde la máxima robustez es necesaria.

Comparación

Esquema	**Función**	**Problema base**	**Idea principal**
ML-KEM / Kyber	Encapsulación de claves	Module-LWE	Eficiencia práctica
ML-DSA / Dilithium	Firmas digitales	Module-SIS /Module-LWE	Autenticidad verificable
FrodoKEM	Encapsulación de claves	LWE estándar	Enfoque conservador

En la práctica moderna, Kyber y Dilithium representan el punto óptimo: estructura algebraica suficiente para ser eficientes, pero no tanta como para introducir riesgos especulativos. Sus tamaños son contenidos, sus operaciones modulares pueden implementarse con gran eficiencia y su aritmética encaja bien con hardware y software generalistas. FrodoKEM es la opción cuando se busca máxima robustez teórica a costa de eficiencia.

Con esto queda cerrada la transición entre teoría y práctica. LWE proporciona el problema de base; sus variantes estructuradas permiten eficiencia; y los esquemas reales muestran cómo esa combinación se convierte en mecanismos concretos de confidencialidad, establecimiento de claves y autenticación. El siguiente paso será implementar una versión simplificada de estas ideas para observar, aunque sea en un entorno controlado, cómo influyen los parámetros en la corrección y el comportamiento del sistema.

7. Implementación práctica

La dificultad de LWE no depende únicamente de la dimensión del problema o del tamaño del módulo. Precisamente por eso, el objetivo no es reproducir Kyber ni competir con implementaciones reales. De hecho, muchas de las optimizaciones algebraicas y consideraciones de seguridad presentes en los estándares modernos han sido deliberadamente omitidas.

Esta sección cierra el espacio introducido durante todo el proyecto; plasmando aquí algo más básico y, al mismo tiempo, más ilustrativo: observar experimentalmente cómo aparece la separación entre seguridad y corrección a medida que el ruido entra en juego.

7.1. Descripción del esquema implementado

El esquema sigue directamente la construcción introducida en la Sección 6.1: una versión simplificada del sistema de Regev que cifra bits individuales. Consta de tres operaciones.

Generación de claves. Se elige una matriz pública $A \in \mathbb{Z}_q^{m\times n}$ de forma uniforme sobre $\mathbb{Z}_q$, y un secreto $s \in \mathbb{Z}_q^n$ también uniforme. El vector de error

$$e \in \mathbb{Z}^m$$

se muestrea componente a componente desde una distribución gaussiana discreta con desviación típica σ: cada e_i se obtiene redondeando una muestra de $\mathcal{N}(0,\sigma^2)$ al entero más próximo. La clave pública es el par (A, b) donde

$$b = As + e \pmod q,$$

y la clave privada es s.

Cifrado. Para cifrar un bit $\mu \in \{0,1\}$, se genera un vector binario efímero $r \in \{0,1\}^m$ uniformemente al azar y se calculan

$$u = A^\top r \pmod q, \qquad v = b^\top r + \mu \left\lfloor \frac{q}{2} \right\rfloor \pmod q.$$

El criptograma es el par $(u, v) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

El bit 0 se representa mediante una cantidad cercana a 0, mientras que el bit 1 se representa mediante una cantidad cercana a $q/2$.

Descifrado. A partir del secreto s y del criptograma (u, v), se calcula

$$d = v - \langle u,\, s\rangle \bmod q$$

y se decide por umbral: $\mu = 0$ si d está más próximo a 0 que a $\lfloor q/2 \rfloor$; $\mu = 1$ en caso contrario.

La razón por la que el descifrado funciona se obtiene expandiendo d en función de $b = As + e$:

$$d = \left(b^\top r + \mu\lfloor q/2 \rfloor\right) - s^\top A^\top r = \left(s^\top A^\top r + e^\top r + \mu\lfloor q/2 \rfloor\right) - s^\top A^\top r = e^\top r + \mu \left\lfloor \frac{q}{2} \right\rfloor \quad (\text{mód } q).$$

El secreto se cancela y queda únicamente el error acumulado $e^\top r$ más el desplazamiento del bit. Si el ruido acumulado permanece suficientemente acotado, el resultado queda más próximo a uno de los dos centros esperados y el bit puede recuperarse mediante una decisión por umbral.

7.2. Detalles de implementación en Python

Este esquema difiere de los sistemas desplegados en producción en varios aspectos, todos intencionales.

La primera simplificación es trabajar con LWE matricial básico, sin pasar a Ring-LWE ni Module-LWE. En esquemas reales, como Kyber, la estructura modular sobre anillos de polinomios permite reducir tamaños y acelerar operaciones —pero también introduce detalles adicionales: productos polinomiales, transformadas NTT, compresión y redondeos— innecesarios para estudiar el papel del ruido.

La omisión de ruido adicional en u —presente en algunas formulaciones del esquema de Regev— simplifica el análisis del error de descifrado. Con $u = A^\top r$ (sin perturbación), la identidad $d = e^\top r + \mu\lfloor q/2 \rfloor$ es exacta, lo que permite aislar el efecto de σ sin interferencias de otras fuentes de error. Añadir ruido a u introduce un término $s^\top e_1$ que, para secretos uniformes en $\mathbb{Z}_q^n$, tiene magnitud del orden de $\sigma\sqrt{n}\, q/\sqrt{3}$, mucho mayor que q, y dificulta la interpretación directa de los experimentos.

El cifrado está limitado a bits individuales. Esta restricción no afecta al comportamiento de los parámetros bajo estudio; la extensión a mensajes más largos requeriría cifrar bit a bit o introducir técnicas de codificación que no añaden información sobre el papel del ruido.

No se aplica ningún mecanismo de compresión ni cuantización. En Kyber y esquemas similares, estas técnicas reducen el tamaño de claves y criptograma, pero introducen errores adicionales que desacoplan el efecto de σ del comportamiento observable. Mantenerlas fuera del esquema conserva la interpretabilidad directa de los resultados.

También se omiten transformaciones de seguridad como Fujisaki–Okamoto, técnicas de reconciliación, compresión de claves o protección frente a canales laterales relevantes en implementaciones industriales.

Replicar un esquema industrial completo habría desplazado el foco del trabajo hacia

detalles de implementación y optimización que, aunque importantes desde el punto de vista práctico, dificultan observar con claridad el fenómeno principal que se pretende estudiar: el papel criptográfico del ruido.

El código completo de la implementación y de los experimentos se encuentra disponible en el Anexo B.

7.3. Parámetros utilizados

Los experimentos se organizan alrededor de tres parámetros principales:

$$n, \qquad q, \qquad \sigma.$$

Parámetro	**Valor base**	**Descripción**
n	64	Dimensión del secreto $s \in \mathbb{Z}_q^n$.
q	257	Módulo primo; fija el espacio modular y la separación entre las regiones de descifrado.
σ	3	Desviación típica del error gaussiano discreto.
m	$2n = 128$	Número de muestras públicas, es decir, filas de $A \in \mathbb{Z}_q^{m \times n}$.

Aun así, los experimentos se han realizado variando progresivamente los parámetros principales del esquema para observar cómo afectan tanto a la corrección del descifrado como al coste computacional. Las métricas analizadas han sido principalmente:

- tasa de fallo de descifrado;
- distribución de los valores de descifrado;
- crecimiento temporal de las operaciones;
- tamaño aproximado de la clave pública.

Para los experimentos de barrido se varían:

- $\sigma \in \{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 16, 20, 24\}$, con $n = 64$ y $q = 257$ fijos.
- $q \in \{97, 199, 257, 401, 521, 769\}$, con $n = 64$ y σ fijo en el rango de transición.
- $n \in \{16, 32, 64, 96, 128\}$, con $q = 257$ y $\sigma = 3$ fijos, para el análisis de coste computacional.

La elección de estos valores busca un equilibrio entre claridad experimental y coste computacional. El módulo $q = 257$ (9 bits) se toma porque es un primo pequeño y cercano

a 2^8, lo que permite trabajar con números manejables y mantener una separación suficiente entre las dos regiones de descifrado.

La dimensión $n = 64$ permite que los experimentos sean rápidos, pero sin quedar reducidos a un caso demasiado pequeño. Como se toma $m = 2n = 128$, el error acumulado $e^\top r$ tiene una desviación típica aproximada

$$\sigma\sqrt{m/2} = 8\sigma.$$

Para el valor base $\sigma = 3$, esto da un ruido típico de aproximadamente 24, mientras que el umbral de corrección es

$$q/4 \simeq 64.$$

Por tanto, el ruido está presente, pero sigue siendo lo bastante pequeño como para que el descifrado sea fiable. Finalmente, la elección $m = 2n$ permite usar un número de muestras proporcional a la dimensión sin hacer que la matriz pública sea innecesariamente grande.

Cada punto experimental agrega resultados sobre múltiples claves independientes y múltiples cifrados por clave. Los intervalos de confianza al 95 % se calculan mediante la aproximación normal $\bar{x} \pm 1{,}96\, s/\sqrt{k}$, donde k es el número de repeticiones independientes y s la desviación típica muestral de la tasa de error entre repeticiones; lo que permite separar la variabilidad propia de una sola clave del comportamiento medio del esquema.

7.4. Experimentos

Se presentan cuatro experimentos, cada uno orientado a una dimensión distinta del comportamiento del esquema.

Experimento 1: fallo de descifrado frente al ruido.

La primera serie de experimentos se centró en estudiar cómo evoluciona la tasa de fallo de descifrado al variar el parámetro de ruido σ, manteniendo fijos n y q. Este análisis resulta especialmente representativo porque refleja el equilibrio central de LWE: el error debe ser suficientemente grande como para ocultar la estructura algebraica subyacente, pero no tanto como para impedir la recuperación correcta del mensaje.

El objetivo es observar empíricamente los tres regímenes descritos en la Sección 7.3. Sin ruido, la relación lineal entre las variables permanece prácticamente expuesta y el sistema pierde gran parte de su interés criptográfico. Introducir cantidades moderadas de error altera significativamente esa estructura observable, mientras que un ruido excesivo termina degradando la capacidad de descifrado.

Experimento 2: fallo de descifrado frente al módulo q.

A continuación se analizó el efecto del módulo q. Para ello, se fijó σ dentro de la zona de transición observada en el experimento anterior y se estudió cómo varía la probabilidad de error al modificar el tamaño del módulo.

Aumentar q incrementa la separación entre los centros esperados asociados a los bits 0 y 1, ampliando el margen de corrección aproximadamente hasta $q/4$. Intuitivamente, esto mejora la robustez frente al ruido, aunque también implica trabajar con representaciones numéricas mayores y operaciones más costosas. El experimento permite verificar empíricamente la relación teórica

$$\sigma \ll \frac{q}{4\sqrt{m/2}},$$

que describe el régimen donde el descifrado continúa siendo fiable.

Experimento 3: coste temporal y tamaño de clave.

Otro bloque experimental se centró en estudiar el efecto de la dimensión n sobre el coste temporal y el tamaño de la clave pública. Se midió el tiempo de cada operación principal del esquema (*key generation*, cifrado y descifrado) y el crecimiento del tamaño de la clave pública al aumentar la dimensión.

Aunque la implementación desarrollada no busca eficiencia industrial, sí permite observar claramente la tendencia de crecimiento asociada al modelo matricial clásico de LWE. El objetivo es confirmar empíricamente el crecimiento aproximadamente cuadrático del coste computacional y del tamaño de las claves, lo que motiva el uso de variantes estructuradas como Ring-LWE o Module-LWE en los esquemas postcuánticos modernos.

Experimento 4: distribución del valor de descifrado.

Finalmente, se estudió la distribución del valor de descifrado

$$d = (v - \langle u, s \rangle) \bmod q$$

mediante histogramas generados tanto para una clave fija como promediando sobre múltiples claves independientes.

Estos histogramas permiten visualizar directamente cómo el ruido modifica la distribución de los valores descifrados y cómo se separan o solapan las regiones de decisión asociadas a $\mu = 0$ y $\mu = 1$. Parte del interés de este análisis reside en que transforma una propiedad abstracta del esquema en un comportamiento geométricamente observable.

Para este experimento se fijó $\sigma = 6$, un valor situado dentro de la zona de transición detectada en el Experimento 1, donde la tasa de fallo ya resulta visible pero el sistema conserva todavía cierta capacidad de descifrado correcto. Esta elección es deliberada: con valores de ruido demasiado pequeños las distribuciones quedan completamente separadas, mientras que un ruido excesivo provoca un solapamiento casi total y elimina información útil del histograma.

7.5. Análisis de resultados

7.5.1. Fallo de descifrado frente al ruido

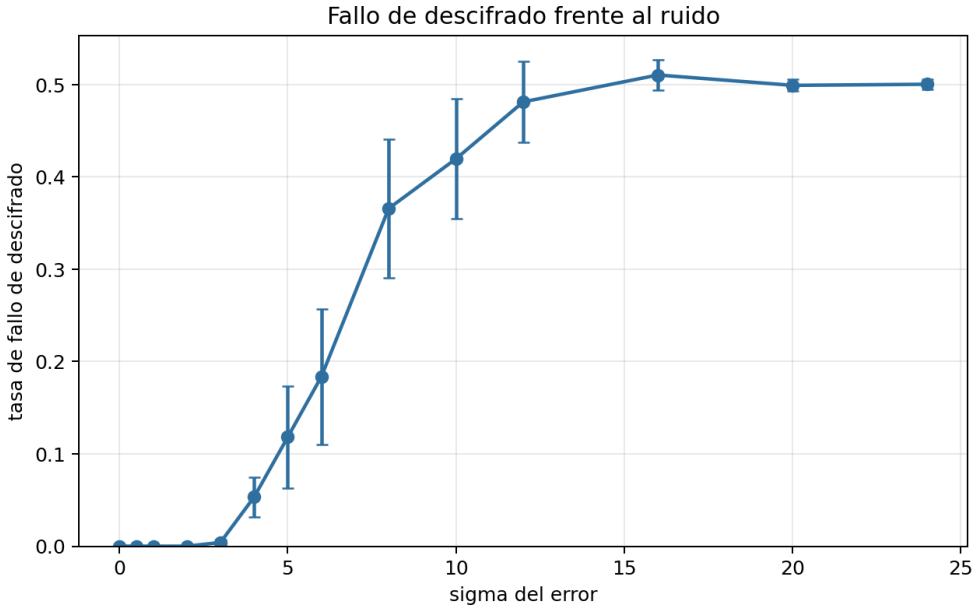

Figura 1: Tasa de fallo de descifrado en función de σ ($n = 64$, $q = 257$). Barras de error: IC bootstrap al 95 %.

La Figura 1 muestra la curva empírica de fallo de descifrado al variar σ. El comportamiento es sigmoidal: parte de cero para σ pequeño y converge a 0,5 para σ grande, el valor esperado cuando el descifrador no tiene ninguna ventaja sobre el azar.

Para $\sigma \leq 3$, la tasa de error es prácticamente nula. El error acumulado $e^{\top}r$ tiene una magnitud típica que, para estos parámetros, sigue quedando claramente por debajo del umbral de corrección $q/4 = 64$. Este umbral es, aproximadamente, el régimen donde el esquema sigue siendo funcional: el ruido es suficientemente grande para que dificulte significativamente la recuperación directa de la estructura lineal, pero suficientemente pequeño para que el descifrado sea fiable.

A partir de $\sigma \approx 4$ la curva empieza a crecer. La desviación típica del error sube a 32, acercandose al umbral de corrección, y la probabilidad de superar el umbral $q/4 = 64$ ya no es despreciable. Para $\sigma = 6$, la tasa de error supera el 15 %; para $\sigma = 8$–10, supera el 35 %.

Dos detalles del gráfico merecen atención. El primero es que los intervalos de confianza se ensanchan precisamente en la zona de transición ($\sigma \in [4, 8]$): en ese régimen, el comportamiento del esquema es más sensible a la instancia concreta —qué A y qué s se eligieron—, lo que traduce en más variabilidad entre claves. El segundo es que la convergencia a 0,5 para σ grande es consistente con la predicción teórica: cuando el ruido supera con creces la separación $q/4$, el ruido termina ocultando la información útil del bit

y el descifrado es indistinguible del azar.

La curva observada encaja con el comportamiento esperado teóricamente y permite identificar aproximadamente la zona donde el esquema sigue siendo funcional: para los valores ensayados, $\sigma \in [1, 3]$ presenta tasa de fallo inferior al 1 %, mientras que $\sigma = 0$ coincide con un sistema lineal exacto resoluble en tiempo polinómico. Cualquier $\sigma > 0$ ya rompe la resolubilidad exacta, pero la corrección del descifrado se mantiene mientras el error acumulado no supere el umbral $q/4$.

7.5.2. Fallo de descifrado frente al módulo q

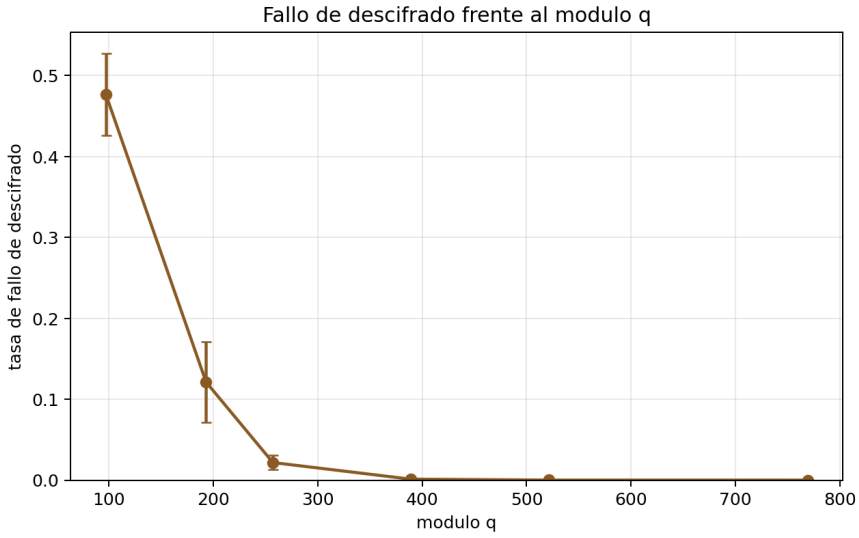

Figura 2: Tasa de fallo de descifrado en función del módulo q ($n = 64$, $\sigma = 4$, $k = 10$ repeticiones independientes). Barras de error: IC al 95 %.

La Figura 2 fija $\sigma = 4$ —el valor donde la curva del experimento anterior comenzaba a crecer— y varía q. El resultado es una caída pronunciada de la tasa de error: para $q = 97$, la tasa ronda 0,5 (el esquema prácticamente no descifra); para $q = 257$, ya es baja; para $q \geq 400$, cae esencialmente a cero.

Geométricamente, de forma bastante intuitiva, el error acumulado $e^\top r$ tiene magnitud fija, determinada por σ y m. Lo que cambia con q es el umbral de corrección $q/4$: cuanto mayor es q, más margen hay entre la región de decisión de $\mu = 0$ y la de $\mu = 1$, y menor es la probabilidad de que el error cruce esa frontera. Para $q = 97$, el umbral es $q/4 \approx 24$; con $\sigma = 4$ y $m = 128$, la desviación típica del error es 32, que supera ese umbral aproximadamente en el 45 % de los casos, lo que es coherente con la tasa de fallo observada cercana a 0,5. Para $q = 257$, $q/4 = 64$, que queda por encima de la desviación típica del error para σ en ese rango. A medida que q aumenta, la separación entre regiones de decisión se vuelve mucho mayor y el descifrado se estabiliza.

Este comportamiento confirma la condición de corrección del descifrado derivada de la identidad $d = e^\top r + \mu\lfloor q/2 \rfloor$ introducida en la Sección 7.1: el descifrado es fiable cuando $\sigma\sqrt{m/2} \ll q/4$, es decir, $\sigma \ll q/(4\sqrt{m/2})$. Para los parámetros del experimento, esa condición se satisface claramente a partir de $q \approx 300$. Reordenando, esa condición puede leerse como $q \gg 4\sigma\sqrt{m/2}$: para un nivel de ruido dado, el módulo debe ser suficientemente grande como para que el umbral de corrección domine el error típico. Los resultados observados de la Figura 2 son coherentes con la relación teórica.

Subir q no es gratuito: el tamaño de cada elemento crece con $\lceil \log_2 q \rceil$ bits, con el consiguiente impacto en claves y comunicación; a partir del punto en que $q/4 \gg \sigma\sqrt{m/2}$, la mejora en corrección es marginal mientras el coste sigue aumentando. En esquemas reales como ML-KEM se trabaja con $q = 3329$, pero con dimensiones n mucho mayores donde el equilibrio entre ruido, módulo y coste se recalibra por completo. El experimento solo cubre un rango estrecho donde la penalización de subir q es pequeña; en dimensiones criptográficamente relevantes, esa penalización se vuelve determinante.

7.5.3. Coste computacional y tamaño de clave

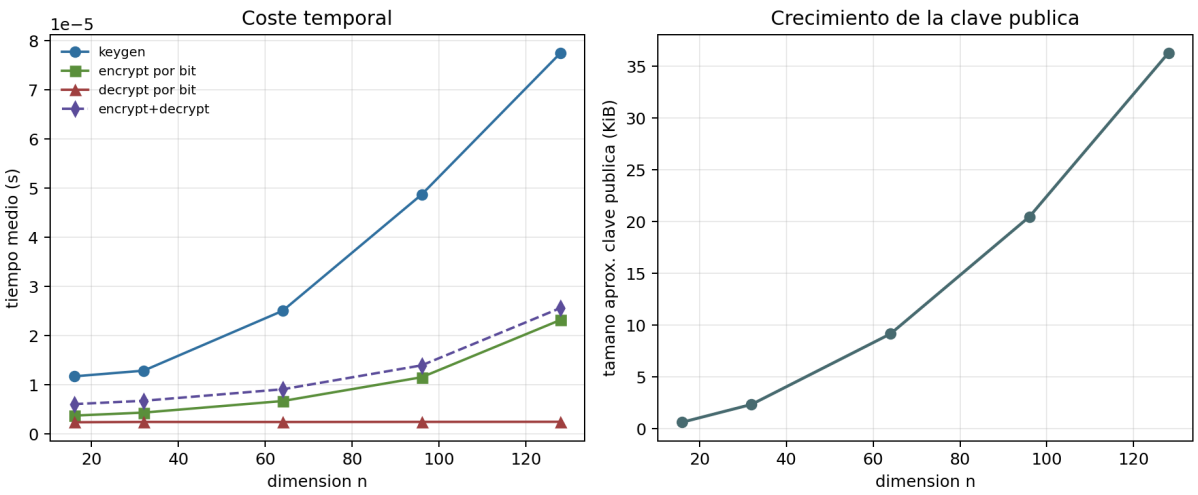

Figura 3: Izquierda: coste temporal de las operaciones del esquema en función de la dimensión n ($q = 257$, $\sigma = 2$). Derecha: tamaño aproximado de la clave pública.

La Figura 3 muestra el resultado del tercer experimento. El panel derecho es el más directo: el tamaño de la clave pública (A, b) crece cuadráticamente con n. La matriz $A \in \mathbb{Z}_q^{2n \times n}$ ocupa, en representación ideal,

$$2n^2 \lceil \log_2 q \rceil$$

bits, sin contar el vector b. Para $q = 257$, cada elemento requiere $\lceil \log_2 257 \rceil = 9$ bits. Así, para $n = 64$, la matriz A ocupa aproximadamente 9 KiB; para $n = 128$, ocupa aproximadamente 36 KiB. En LWE estándar, el coste de almacenamiento y transmisión crece cuadráticamente con la dimensión, lo que en la práctica hace inviable trabajar en dimensiones criptográficamente relevantes si no se introduce estructura algebraica adicional.

El panel izquierdo muestra el coste temporal. La generación de claves es la operación más costosa: involucra el producto matricial As de coste $O(mn) = O(n^2)$, y su curva crece claramente más rápido que las demás. El cifrado por bit involucra el mismo orden asintótico ($A^\top r$), pero en la práctica es más rápido porque A ya está generada, r es binario —las multiplicaciones se reducen a sumas condicionales— y la operación $b^\top r$ es $O(m)$, no $O(mn)$. El descifrado, en cambio, permanece casi constante en todo el rango de n estudiado: se reduce al producto escalar $\langle u, s \rangle$ de coste $O(n)$, que para $n \leq 128$ queda por debajo del umbral de resolución del rango estudiado.

Aunque este resultado era esperable desde el punto de vista teórico, el experimento permite observarlo de forma directa. El crecimiento cuadrático de LWE estándar no es un artefacto de la implementación concreta: es una consecuencia directa de trabajar con matrices densas. Las variantes estructuradas estudiadas en la Sección 5.6 resuelven exactamente este problema mediante estructura algebraica sobre anillos de polinomios, reduciendo el coste a $O(n \log n)$ mediante NTT. Lo que este experimento ilustra es el punto de partida: la barrera que las variantes modulares superan, y que explica por qué los estándares actuales no utilizan LWE estándar.

7.5.4. Distribución del valor de descifrado

Los tres experimentos anteriores cuantifican la tasa de fallo como un número. Lo que no muestran es la geometría subyacente: por qué el descifrador acierta o falla, y cuál es la forma del error que separa ambos casos. Este experimento examina la distribución completa de $d = (v - \langle u, s \rangle) \bmod q$ —la cantidad sobre la que opera el umbral de decisión— y permite ver directamente las dos regiones y el margen que las separa.

Se realizan dos variantes con $\sigma = 6$, dentro de la zona de transición, donde el solapamiento entre distribuciones es visible sin que ninguna quede completamente sumergida.

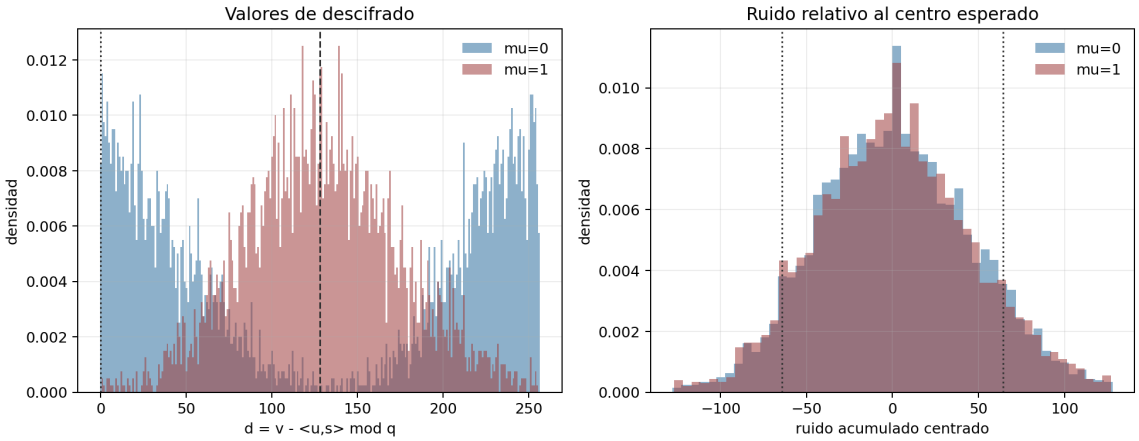

Figura 4: Distribución del valor d promediando sobre múltiples claves independientes. La mayor dispersión respecto a la figura anterior refleja la variabilidad entre instancias.

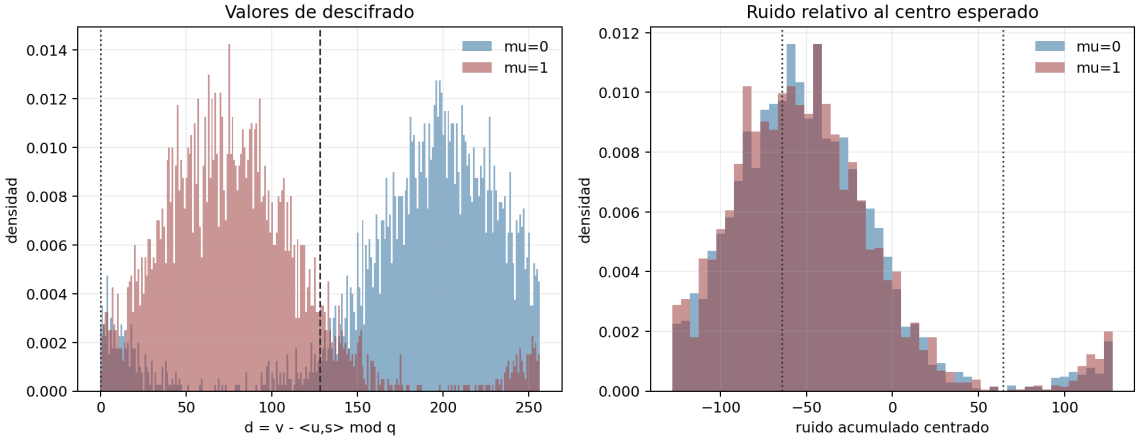

Figura 5: Distribución del valor $d = (v - \langle u, s\rangle)$ mód q para un par de clave fijo. Izquierda: distribución en $[0, q)$. Derecha: ruido acumulado centrado respecto al valor esperado de cada bit.

Los paneles izquierdos de las Figuras 5 y 4 muestran que las dos distribuciones están desplazadas entre sí por $\lfloor q/2 \rfloor$, con solapamiento en los bordes de las regiones de decisión. El umbral a $q/2 = 128$ (línea discontinua) separa correctamente la mayor parte de la masa de ambas distribuciones.

Los paneles derechos representan el error acumulado centrado: $e_{\text{acum}} = d - \mu\lfloor q/2 \rfloor$ (con representación simétrica módulo q alrededor de cero). Corresponde directamente al término $e^\top r$. Su distribución muestra un perfil aproximadamente gaussiano, compatible con $e^\top r \sim \mathcal{N}(0,\, \sigma^2 m/2)$, un comportamiento aproximadamente gaussiano del término de error; las líneas de puntos marcan el umbral de corrección $\pm q/4$. Los valores que caen fuera de ese intervalo corresponden, de forma determinista, a descifrados incorrectos para ese criptograma concreto —dado un d con error fuera del umbral, el descifrador toma la decisión equivocada con independencia de cualquier otro parámetro.

El histograma multiclave (Figura 4) corrige este sesgo al promediar sobre muchos vectores de error independientes con media cero, produciendo una distribución del ruido centrada alrededor de cero como predice la teoría. La Figura 5 muestra el mismo experimento condicionado a una única clave. La distribución del ruido puede aparecer desplazada porque, al reutilizar siempre el mismo vector de error e, el valor medio de $e^\top r$ sobre muchos cifrados no es exactamente cero: depende de los valores concretos de e_i de esa clave. Al usar muchas claves distintas (Figura 4), esos desplazamientos se promedian y la distribución recupera la simetría que predice la teoría.

Al comparar ambas figuras, se observa que con clave fija la distribución del error es más estrecha: para una instancia concreta (A, s), la variabilidad proviene únicamente de diferentes muestras de r y e. Al promediar sobre múltiples claves, la distribución se ensancha: distintas matrices A y distintos secretos s implican distintas realizaciones del producto $e^\top r$, añadiendo variabilidad adicional. Esta diferencia es directamente el fenómeno descrito en el análisis estadístico de los experimentos anteriores: la tasa de error no es una propiedad fija del esquema, sino una propiedad media sobre la distribución de instancias, con diferencias apreciables entre distintas instancias del esquema.

7.6. Síntesis

En conjunto, los cuatro experimentos describen desde ángulos distintos la misma realidad: el ruido es el elemento que da sentido criptográfico al esquema.

Sin ruido ($\sigma = 0$), el sistema $b = As$ es un sistema lineal exacto sobre $\mathbb{Z}_q$. Con ruido excesivo, el descifrado falla con probabilidad cercana a la mitad —visible en la Figura 1 para $\sigma \geq 16$—, y el esquema deja de ser funcional. Entre ambos extremos existe una ventana de parámetros —visible empíricamente en la curva sigmoidal del Experimento 1 y verificable analíticamente a través de la cota $\sigma \ll q/(4\sqrt{m/2})$— donde el ruido es suficiente para impedir la resolución directa pero insuficiente para destruir la corrección.

El módulo q amplía o estrecha esa ventana sin desplazarla cualitativamente: aumentar q mejora el margen de corrección, aunque también incrementa el tamaño de las claves y el coste asociado al esquema. La dimensión n no modifica de forma cualitativa el comportamiento observado, aunque sí escala el coste computacional cuadráticamente, lo que ayuda a entender por qué LWE estándar resulta poco práctico en dimensiones criptográficamente relevantes sin estructura algebraica adicional.

En ese sentido, los experimentos no son solo una validación de la teoría anterior: son una justificación empírica simplificada de las decisiones de diseño que explican por qué los esquemas modernos como ML-KEM trabajan sobre Module-LWE y no sobre LWE estándar.

8. Conclusiones

Este trabajo ha partido de una idea concreta: la seguridad criptográfica no depende solo del problema matemático elegido, sino también del modelo de computación frente al que se analiza. RSA y ECC siguen siendo construcciones sólidas dentro del modelo clásico, pero el algoritmo de Shor cambia el escenario al ofrecer un procedimiento eficiente para problemas como la factorización y el logaritmo discreto. Por eso, la cuestión no es simplemente aumentar tamaños de clave, sino buscar fundamentos matemáticos que no presenten la misma estructura explotable en el modelo cuántico.

En ese contexto, los retículos proporcionan un marco especialmente adecuado. Su interés no reside únicamente en que no se conozcan algoritmos cuánticos eficientes para los problemas que plantean, sino también en que permiten relacionar instancias promedio con problemas difíciles en el peor caso. Esta conexión es una de las razones por las que los problemas reticulares ocupan un lugar central dentro de la criptografía post-cuántica.

Dentro de esta familia, Learning With Errors concentra el hilo principal del trabajo. Su formulación, una relación lineal perturbada por un error pequeño, cambia por completo la naturaleza del problema. Sin ruido, el sistema se reduce a álgebra lineal sobre $\mathbb{Z}_q$; con ruido correctamente calibrado, la estructura queda oculta sin que el receptor legítimo pierda la capacidad de descifrar. Esta dicotomía entre ocultar y conservar información es la idea que atraviesa tanto la teoría como la parte experimental.

8.1. Resumen de resultados

El resultado principal del trabajo no es un resultado aislado, sino la construcción de un recorrido coherente desde la amenaza cuántica hasta una implementación experimental basada en LWE. En primer lugar, se ha mostrado que la seguridad de RSA y ECC depende de hipótesis de dificultad que dejan de ser adecuadas cuando el adversario dispone del modelo cuántico descrito por Shor. A partir de ahí, el trabajo ha ido desplazando el foco hacia problemas cuya dificultad no procede de la misma estructura algebraica.

Ese recorrido ha exigido introducir primero los retículos como objetos geométricos, los problemas computacionales que se formulan sobre ellos, y por qué esa dificultad resulta útil desde el punto de vista criptográfico.

Sobre esa base se ha situado LWE. La interpretación mediante retículos q-arios permite entender sus muestras como observaciones desplazadas de una estructura discreta. La reducción de Regev añade la pieza teórica esencial: resolver LWE en promedio tendría consecuencias sobre problemas reticulares difíciles en el peor caso. Esta relación no constituye una prueba absoluta de seguridad, pero sí explica por qué LWE no se apoya únicamente en una confianza empírica.

El último paso teórico ha sido pasar del problema matemático a la construcción criptográfica –esquema de cifrado básico, los mecanismos de encapsulación y las firmas digitales–

para mostrar cómo esa dificultad puede convertirse en confidencialidad, establecimiento de claves y autenticación. A su vez, el estudio de Ring-LWE y Module-LWE permite entender por qué los estándares actuales no utilizan LWE estándar en su forma matricial pura: la estructura algebraica adicional reduce costes y hace viable la implementación práctica, aunque obliga a aceptar supuestos algo más específicos.

La parte experimental ha servido para comprobar ese mismo equilibrio en un entorno controlado. El esquema implementado no pretende reproducir un estándar real, sino aislar el papel de los parámetros en un cifrado básico basado en LWE. Al variar la desviación típica del error, aparece una zona clara de transición: con ruido pequeño el descifrado se mantiene estable, mientras que con ruido excesivo la tasa de fallo se aproxima al comportamiento aleatorio. Esto confirma que el ruido es necesario para ocultar la estructura, pero también que su tamaño condiciona directamente la corrección.

El módulo q actúa como margen de separación entre las regiones de decisión. Al aumentarlo, el descifrado resiste mejor el mismo nivel de ruido, aunque cada elemento modular exige más bits. Por su parte, el estudio de la dimensión n muestra el límite práctico de LWE estándar: el uso de matrices densas provoca un crecimiento cuadrático del tamaño de la clave pública y del coste asociado. Este resultado conecta directamente con la motivación de las variantes estructuradas.

Finalmente, los histogramas del valor de descifrado permiten visualizar lo que las tasas de error solo resumen numéricamente. El error acumulado desplaza los valores respecto a sus centros esperados y, cuando cruza el margen de decisión, aparece el fallo de descifrado. La conclusión experimental es la misma que se ha ido construyendo en la parte teórica: en LWE, seguridad y corrección dependen de los mismos parámetros.

8.2. Limitaciones

La principal limitación del trabajo es que la implementación es deliberadamente simplificada. Se ha trabajado con LWE matricial básico, cifrado de bits individuales y sin incorporar elementos propios de esquemas desplegables, como compresión, reconciliación completa, transformación de Fujisaki–Okamoto o protección frente a canales laterales. Esta simplificación permite estudiar con claridad el papel del ruido, pero impide interpretar el código como un sistema criptográfico real.

También ha quedado fuera un desarrollo más detallado de las técnicas algorítmicas que hacen eficientes a las variantes estructuradas, en particular la NTT y la aritmética rápida sobre polinomios. Se han mencionado por su papel en Module-LWE, pero no se han estudiado formalmente, ya que eso habría desplazado el foco desde LWE como problema criptográfico hacia la optimización algebraica de los estándares.

Los parámetros utilizados tampoco pretenden ofrecer seguridad industrial. Valores como $n = 64$, $q = 257$ o las desviaciones típicas empleadas sirven para obtener experimentos

rápidos y visuales, no para alcanzar niveles comparables a los estándares post-cuánticos. Por tanto, los resultados deben entenderse como una validación cualitativa del fenómeno, no como una recomendación de parámetros.

Otra limitación es que el análisis de ataques se ha mantenido en un nivel conceptual. Se han explicado las familias principales, como los ataques primales, duales y BKW, y su relación con la reducción de bases, pero no se ha realizado una estimación cuantitativa completa mediante modelos como Core-SVP. Incluir ese análisis habría ampliado el trabajo hacia una criptoanalítica más específica y menos centrada en el papel del ruido.

Por último, el estudio experimental se ha limitado a un conjunto acotado de barridos. Se han considerado varias claves, varios cifrados por clave e intervalos de confianza, pero no se ha explorado exhaustivamente todo el espacio de parámetros. Aun así, los experimentos realizados son suficientes para mostrar el fenómeno central que se buscaba estudiar: la existencia de una ventana de funcionamiento entre la resolución lineal trivial y el fallo de descifrado.

8.3. Trabajo futuro

Una continuación natural sería implementar una variante simplificada de Module-LWE con NTT. Esto permitiría comparar directamente el crecimiento cuadrático observado en LWE estándar con el comportamiento más eficiente de las operaciones sobre módulos de polinomios. Sería una forma de completar experimentalmente la conexión con los estándares actuales.

También sería interesante ampliar el esquema hacia un mecanismo de encapsulación de claves más completo. Para ello habría que introducir ruido adicional, reconciliación, compresión y una transformación de seguridad adecuada. Esa extensión permitiría estudiar hasta qué punto las conclusiones obtenidas en el modelo simplificado se mantienen cuando la construcción se aproxima más a un esquema real.

Otra línea posible sería incorporar una estimación cuantitativa de seguridad. A partir de los parámetros n, q y σ, podrían calcularse costes aproximados de ataques basados en BKZ y compararlos con la corrección observada experimentalmente. Incluso podría plantearse la implementación de ataques concretos, como el primal o el dual, para unir en una misma discusión el funcionamiento del descifrado y el coste esperado para un adversario.

Por último, podría ampliarse el análisis estadístico del descifrado, estudiando con más detalle la variabilidad entre claves, distintas distribuciones de error y el efecto del número de muestras. Esa ampliación permitiría pasar de una observación clara del fenómeno a una caracterización probabilística más completa. De forma más amplia, también podría compararse este enfoque con otras familias post-cuánticas, como códigos correctores o firmas basadas en hash, aunque esa línea quedaría fuera del núcleo de este trabajo.

Más allá de los resultados concretos obtenidos, el recorrido realizado permite conectar conceptos que inicialmente parecen independientes —computación cuántica, geometría de retículos, complejidad computacional y diseño criptográfico— y entender cómo forman parte de una misma estructura conceptual. Desde la amenaza planteada por Shor hasta el comportamiento observado en las simulaciones, cada elemento contribuye a explicar por qué una pequeña perturbación introducida en una relación algebraica exacta puede convertirse en el fundamento de una construcción criptográfica completa.

En su forma actual, el trabajo no pretende cerrar todos estos problemas. Su aportación es más concreta: mostrar cómo una idea aparentemente sencilla —introducir una pequeña cantidad de incertidumbre en una relación matemática exacta— puede transformarse en una de las propuestas más sólidas para afrontar la transición hacia la criptografía post-cuántica.

A. Pseudocódigo de los algoritmos de reducción

Este anexo recoge el pseudocódigo de los algoritmos LLL y BKZ introducidos en la Sección 4. En el cuerpo principal del trabajo se ha mantenido únicamente la explicación conceptual de estos algoritmos, ya que su función dentro del hilo del trabajo es servir como marco para entender los ataques y la estimación de seguridad en problemas basados en retículos.

A.1. Pseudocódigo del algoritmo LLL

Algorithm 1 Algoritmo LLL (Lenstra-Lenstra-Lovász)

Input: Base $B = [b_1, \ldots, b_n]$, parámetro $\delta \in (1/4, 1)$
$k \leftarrow 2$

while $k \leq n$ **do**
    `# Paso 1: reducción de tamaño`
    `# Se elimina la componente de` b_k `en dirección de los vectores anteriores`
    for $j = k - 1$ **downto** 1 **do**
        size-reduce b_k respecto a b_j
        `# equivale a:` $b_k \leftarrow b_k - \text{round}(\mu_{k,j})b_j$
    end for

    `# Paso 2: comprobar condición de Lovász`
    if $\delta\|b_{k-1}^*\|^2 \leq \|b_k^*\|^2 + \mu_{k,k-1}^2\|b_{k-1}^*\|^2$ **then**
        `# la base está equilibrada localmente →avanzar`
        $k \leftarrow k + 1$
    else
        `# la base está desbalanceada →intercambiar vectores`
        intercambiar b_{k-1}, b_k
        `# retroceder para re-ajustar la base`
        $k \leftarrow \text{máx}(k - 1, 2)$
    end if
end while

Output: base LLL-reducida

A.2. Pseudocódigo del algoritmo BKZ

Algorithm 2 Algoritmo BKZ (Block Korkine–Zolotarev)

Input: Base $B = [b_1, \ldots, b_n]$, tamaño de bloque β

`# Paso inicial: reducción global`
aplicar LLL a B

repeat
  `# Recorrido por bloques`
  for $k = 1$ **to** $n - \beta + 1$ **do**

    `# Paso 0: Definir límites del bloque actual`
    $h \leftarrow \min(k + \beta - 1, n)$

    `# Paso 1: proyección del bloque`
    considerar el subretículo generado por $[b_k, \ldots, b_{k+\beta-1}]$
    proyectar ortogonalmente respecto a $b_1^*, \ldots, b_{k-1}^*$

    `# Paso 2: llamada al SVP-oracle`
    $v \leftarrow \text{SVP-Oracle}(b_{k:h}^*)$
    encontrar un vector corto v en el subretículo proyectado

    `# Paso 3: lifting a la base original`
    expresar v en función de $b_k, \ldots, b_{k+\beta-1}$

    `# Paso 4: inserción en la base`
    if $\|v\| < \|b_k^*\|$ **then**
      insertar v en la base en posición k

      `# reequilibrar tras la inserción`
      aplicar LLL sobre $[b_1, \ldots, b_{k+\beta-1}]$
    end if
  end for
until no se producen inserciones en toda la pasada

Output: base BKZ-reducida

B. Repositorio de código

El código utilizado para la implementación experimental descrita en la Sección 7 se encuentra disponible en el siguiente repositorio:

`https://github.com/alejandromtnz/LWE_implementation`

El repositorio contiene los módulos empleados para la generación de claves, cifrado, descifrado, ejecución de experimentos y generación de gráficas. La implementación no pretende reproducir un estándar post-cuántico completo, sino proporcionar una versión simplificada del esquema basado en LWE que permita estudiar de forma controlada el papel del ruido y de los parámetros.

Referencias

[1] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing (STOC)*, pages 99–108. ACM, 1996.

[2] Erdem Alkim, Joppe W. Bos, Léo Ducas, Patrick Longa, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Chris Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM: Learning with errors key encapsulation algorithm specifications and supporting documentation, June 2021. Version of June 4, 2021.

[3] William V. Jenkins. *Criptografía postcuántica: la guía esencial para entender la ciberseguridad del futuro*. 2023.

[4] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: A Cryptographic Perspective*. Springer, 2002.

[5] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen, editors, *Post-Quantum Cryptography*. Springer, 2009.

[6] National Institute of Standards and Technology. FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM), 2024.

[7] National Institute of Standards and Technology. FIPS 204: Module-Lattice-Based Digital Signature Standard (ML-DSA), 2024.

[8] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*. ACM, 2005.

[9] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. arXiv:2401.03703, 2024.

[10] Ronald L. Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[11] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS)*, 1994.