Análisis comparativo de las firmas digitales postcuánticas basadas en retículos

Eva Iglesias Hernández
Instituto de Tecnologías Físicas y de la Información
Consejo Superior de Investigaciones Científicas
C/ Serrano 144, 28006, Madrid
eva.iglesias@csic.es

Luis Hernández-Álvarez
Instituto de Tecnologías Físicas y de la Información
Consejo Superior de Investigaciones Científicas
C/ Serrano 144, 28006, Madrid
luis.hernandez@csic.es

Luis Hernández Encinas
Instituto de Tecnologías Físicas y de la Información
Consejo Superior de Investigaciones Científicas
C/ Serrano 144, 28006, Madrid
luis.h.encinas@csic.es

José Ignacio Sánchez García
Instituto de Tecnologías Físicas y de la Información
Consejo Superior de Investigaciones Científicas
C/ Serrano 144, 28006, Madrid
ji.sanchez@csic.es

***Resumen*—Después de la publicación de los algoritmos de Shor, la comunidad criptográfica ha aceptado que una vez que se desarrolle un ordenador cuántico con la suficiente capacidad de cómputo, la criptografía asimétrica actual será vulnerada de modo eficiente. Por ello, el NIST lanzó una convocatoria internacional para la elección de nuevos estándares criptográficos resistentes a la computación cuántica. En 2022, el NIST publicó un mecanismo de encapsulamiento de claves (CRYSTALS-Kyber) y tres esquemas de firma electrónica (CRYSTALS-Dilithium, FALCON y SPHINCS$^+$). En este trabajo se presenta un análisis de los dos primeros esquemas de firma, los basados en retículos, con el fin de homogeneizar sus características en cuanto a tamaño de sus parámetros y su rendimiento, habida cuenta que las propuestas originales fueron presentadas utilizando diferentes arquitecturas, lo que dificulta su comparación en cuanto a los dos aspectos señalados anteriormente.**

***Index Terms*—Criptografía postcuántica, Firmas digitales, Problema del aprendizaje con errores, Rendimiento computacional.**

Tipo de contribución: *Investigación original y en desarrollo*

I. Introducción

Como es bien sabido, Peter Shor publicó en [1] dos algoritmos capaces de resolver en tiempo polinómico, caso de existir un ordenador cuántico con la suficiente capacidad de cómputo, los dos problemas de la Teoría de Números más utilizados en la criptografía asimétrica actual: el problema de la factorización de números enteros y el problema del logaritmo discreto. La dificultad computacional de ambos problemas con ordenadores no cuánticos es lo que garantiza la seguridad de los dos criptosistemas asimétricos más empleados a día de hoy: el RSA [2] y los basados en curvas elípticas [3], [4], además de otros muchos protocolos basados en ellos, como los de firma digital o los de acuerdo de clave, como los de Diffie-Helmman [5].

No es fácil aventurar cuándo todos estos protocolos criptográficos dejarán de ser seguros, porque no sabemos con certeza cuándo existirá un ordenador cuántico criptoanalíticamente relevante o CRQC (*(Cryptanalytically Relevant Quantum Computer*); pero lo que sí es seguro es que la criptografía actual o precuántica tiene los días contados.

A la vista de esta situación, el Instituto de estándares y tecnología norteamericano o NIST (*National Institute for Standards and Technology*) realizó, en 2016, una convocatoria internacional para la propuesta y posterior elección de nuevos estándares criptográficos asimétricos, resistentes a la computación cuántica. Esta convocatoria, conocida como criptografía postcuántica o PQC (*Post-Quantum Cryptography*), se restringía a los mecanismos de encapsulamiento de claves o KEM (*Key Encapsulation Mechanism*) y a los esquemas de firma digital.

Después de varias rondas, en 2022, el NIST publicó los nuevos estándares, que pueden considerarse como los ganadores de esta PQC [6]. Esta lista contiene un único KEM, a saber, CRYSTALS-Kyber [7], y tres esquemas de firma digital: CRYSTALS-Dilithium [8], FALCON [9] y SPHINCS$^+$ [10]. Los tres primeros basan su seguridad en determinados problemas matemáticos definidos sobre la estructura de retículo; mientras que el último lo hace sobre determinadas funciones *hash*.

En este trabajo vamos a llevar a cabo un análisis comparativo de los dos esquemas de firma digital cuya seguridad se fundamenta en retículos, esto es, CRYSTALS-Dilithium y FALCON. Este análisis tiene como objetivo utilizar una misma arquitectura y un mismo marco de referencia para las correspondientes implementaciones, que permita comparar de un modo claro y coherente las características de ambos esquemas. Así, el estudio se centrará en analizar los tamaños de las claves públicas y privadas utilizadas y de las firmas obtenidas por cada uno de los esquemas, como de su rendimiento, tanto en tiempo de ejecución como en el uso de la CPU.

De este modo, el resto de esta comunicación se estructura de la siguiente manera. En la sección II se presentan, de forma muy resumida, los fundamentos matemáticos en los que se basa la seguridad de los esquemas de firma a estudiar. Las características y los algoritmos que definen ambos esquemas se incluyen en la sección III. La sección IV contiene los análisis comparativos que hemos llevado a cabo y los resultados correspondientes. Finalmente, en la sección V se incluyen las principales conclusiones de este trabajo.

ISBN:978-84-09-62140-8

II. Fundamentos y notación

En esta sección se presentarán, de forma muy elemental, los fundamentos en los que se basan los dos esquemas de firma digital que serán analizados.

Dado un conjunto de vectores linealmente independientes del espacio vectorial real de dimensión n, $\mathbb{R}^n$, sea $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$, con $m \leq n$. Un retículo (*lattice*), es un subgrupo aditivo discreto de $\mathbb{R}^n$ formado por las combinaciones lineales enteras de $\mathbf{B}$. Esta estructura suele denotarse por $\mathcal{L} \subset \mathbb{R}^n$:

$$\begin{aligned}\mathcal{L} &= \left\{ \sum_{i=1}^{m} a_i \cdot \mathbf{b}_i : a_i \in \mathbb{Z} \right\} \\ &= \{\mathbf{B} \cdot \mathbf{a} : \mathbf{a} = \{a_1, \ldots, a_m\} \in \mathbb{Z}^m\}.\end{aligned}$$

Dicho de otro modo, un retículo no es más que una malla de puntos en $\mathbb{R}^n$ de modo que tales puntos se obtienen como múltiplos enteros de determinado número de vectores linealmente independientes.

Los dos problemas, considerados computacionalmente difíciles, clásicos definidos en un retículo son el problema del vector más corto o SVP (*Shortest Vector Problem*) y el del vector más cercano o CVP (*Closest Vector Problem*).

El SVP consiste en determinar un vector no nulo, $\mathbf{x} \in \mathcal{L}$, que sea el vector más corto de $\mathcal{L}$, es decir, de modo que $||\mathbf{x}|| \leq ||\mathbf{y}||$, $\forall \mathbf{y} \in \mathcal{L}$, siendo $||\cdot||$ la norma definida en $\mathcal{L}$.

Por otra parte, dado un vector $\mathbf{x} \in \mathbb{R}^n$ de modo que $\mathbf{x} \notin \mathcal{L}$, el CVP consiste en encontrar un vector $\mathbf{v} \in \mathcal{L}$ de modo que $||\mathbf{x} - \mathbf{v}|| \leq ||\mathbf{x} - \mathbf{w}||$, $\forall \mathbf{w} \in \mathcal{L}$.

Al margen de estos dos problemas clásicos existen otros problemas, también considerados computacionalmente difíciles, que son la base de la seguridad de la mayor parte de los estándares considerados por el NIST, hasta la fecha.

Uno de estos problemas es el problema de la solución entera corta o SIS (*Short Integer Solution*), cuyo enunciado es: sean n y q dos números enteros y sea $\beta > 0$. Dada, además, una matriz uniformemente aleatoria $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ para algún $m = \text{poly}(n)$, se trata de encontrar un vector entero no nulo $\mathbf{z} \in \mathbb{Z}^m$, tal que $\mathbf{Az} = \mathbf{0} \pmod q$ y $||\mathbf{z}|| \leq \beta$, usando la norma estándar.

Existe dos variantes del problema anterior, definidas, respectivamente, sobre la estructura de anillo, llamada RSIS, o de módulo, denotada como MSIS. Sea R un anillo (resp. módulo) y sean m, q enteros positivos. Dados los coeficientes de una matriz, $\mathbf{A}, = a_{i,j} \leftarrow_{\$} R_q^m$, se trata de encontrar un vector del anillo (resp. módulo) distinto de cero $z \in R^m$ de norma $||z|| \leq \beta$ tal que $\mathbf{A}z = 0 \pmod q$.

Otro problema, de gran importancia criptográfica es el problema del aprendizaje con errores o LWE (*Learning With Errors*, que ha dado lugar a dos versiones: una definida sobre anillos, RLWE (*Ring-Learning With Errors*), y otra sobre módulos, MLWE (*Module-Learning With Errors*).

El problema LWE, del que existen dos versiones: de búsqueda y de decisión, se parametriza por un entero n, un número primo $q \geq 2$ y una distribución de probabilidad χ sobre $\mathbb{Z}_q$.

El problema LWE de búsqueda es el siguiente: dadas m muestras independientes $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ siguiendo la distribución $A_{\mathbf{s},\chi}$ con $b_i = \langle \mathbf{s}, \mathbf{a}_i \rangle + e_i \pmod q \in \mathbb{Z}_q$, se trata de encontrar el vector secreto $\mathbf{s} \in \mathbb{Z}_q^n$, siendo $m \geq n$.

El problema LWE de decisión tiene como objetivo es distinguir entre dos conjuntos de $m \geq n$ pares de vectores $(\mathbf{a}_i, b_i)$ y $(\bar{\mathbf{a}}_i, \bar{b}_i)$, donde $\mathbf{a}_i, \bar{\mathbf{a}}_i \leftarrow_\chi \mathbb{Z}_q^n$, $b_i = \langle \mathbf{s}, \mathbf{a}_i \rangle + e_i \pmod q$ y $\bar{b}_i \in \mathbb{Z}_q$. Es decir, el problema es decidir, para cada par de vectores dados, si el segundo vector, b_i, es el producto escalar del primer vector, $\mathbf{a}_i$, por algún vector secreto, $\mathbf{s}$, sumado con algún error, e_i, o si es un vector uniformemente aleatorio.

En el caso del RLWE, la estructura que se considera es la del anillo finito de los polinomios enteros módulo el ideal generado por $\Phi(x)$: $R_q = \mathbb{Z}_q[x]/\langle \Phi(x) \rangle$, siendo $\Phi(x) = x^n + 1 \in \mathbb{Z}[x]$, el m-ésimo polinomio ciclotómico de grado $\varphi(m) = 2^{k-1} = n$. También para el problema RLWE existen las versiones de búsqueda y de decisión.

El problema MLWE es otra variante del LWE que trata de ser un caso intermedio entre el LWE y el RLWE. El módulo considerado en el MLWE es $R_q^k = (\mathbb{Z}_q[x]/\langle x^n + 1 \rangle)^k$ y puede entenderse como un retículo de dimensión $k \cdot n$, siendo k la dimensión del módulo R_q^k sobre el anillo R_q.

En general, el valor considerado para k en las propuestas criptográficas es un número pequeño. De hecho, en CRYSTALS-Kyber se considera $k = 2, 3, 4$, según el nivel de seguridad. Esto es, a mayor k, mayor nivel de seguridad y, por tanto, mayor costo computacional.

La idea subyacente en los problemas RLWE y MLWE es considerar que cada polinomio de R_q se puede identificar con un vector cuyas coordenadas son sus n coeficientes y los elementos de R_q^k son vectores de k polinomios.

Por otra parte, a la hora de comparar la seguridad de los problemas LWE, RLWE y MLWE, la idea más extendida es que cuanto más estructurado sea un retículo, menos seguridad ofrecerá, precisamente porque la existencia de tal estructura permitirá ataques más eficientes.

Finalmente, es digno de destacar que tanto en el caso del RLWE como del MLWE, dado que se trabaja con polinomios, la operación básica en las propuestas postcuánticas basadas en retículos es el producto ordinario de polinomios. Para cada uno de los n coeficientes afectados en este producto son necesarias n multiplicaciones. Por ello, su complejidad es $O\left(n^2\right)$ y para valores grandes de n este cómputo es muy costoso.

Debido a esta complejidad computacional, es fundamental disponer de un algoritmo que lleve a cabo tal multiplicación de modo eficiente. Por este motivo se suele recurrir a la transformada de teoría de números o NTT (*Number Theoretic Transform*). Esta transformada es un algoritmo eficiente que lleva a cabo el producto de polinomios y es, esencialmente, una transformada discreta de Fourier (*Discrete Fourier Transform*) en un cuerpo finito $\mathbb{F}_q$.

Si se considera q primo, la dimensión del retículo, n, sea una potencia de 2 y $q \equiv 1 \pmod{2n}$, la complejidad de la multiplicación de polinomios haciendo uso de la NTT se reduce a $O(n \log n)$.

III. Firmas postcuánticas basadas en retículos: Dilithium y FALCON

En esta sección se incluyen las principales características de los dos esquemas de firma, basadas en retículos, que se han analizado: CRYSTALS-Dilithium y FALCON.

III-A. CRYSTALS-Dilithium

CRYSTALS-Dilithium, o simplemente Dilithium, es uno de los dos esquemas de firma basados en retículos incluidos en la cuarta ronda del NIST [11]. Puede considerarse como uno de los tres estándares de firma electrónica, dado que el NIST ya ha publicado el borrador de este esquema [12], denotándolo como ML-DSA, y se espera que a lo largo de 2024 se publique la versión definitiva del mismo.

Dilithium, es un esquema de firma digital que se basa en la firma de Fiat-Shamir [13] y fue propuesto por Ducas *et al.* en [14]. Su seguridad se fundamenta en la fortaleza del problema MLWE y del problema MSIS. Dado que Dilithium usa el mismo módulo y anillo para todos los conjuntos posibles de parámetros, es relativamente fácil de implementar y, como se verá más adelante, presenta un rendimiento equilibrado entre los tamaños de las claves y el de la firma y es eficiente en la ejecución de los tres algoritmos de los que consta: generación de claves, elaboración de la firma y verificación de la firma.

Presentaremos, de forma esquemática, cada uno de estos tres algoritmos de modo que se aprecien las diferentes operaciones y cómputos que cada uno de ellos lleva a cabo. No entraremos a detallar las diferentes subrutinas de estos algoritmos por razones de espacio. No obstante, el lector interesado puede recurrir a la propuesta original de Dilithium [8] para obtener más detalles.

El Algoritmo 1 lleva a cabo la generación de las claves, tanto de la pública, pk, como de la privada, sk. Para ello, se genera una matriz $\mathbf{A}$ de tamaño $k \times l$ cuyas entradas pertenecen al anillo de polinomios $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$, con $q = 2^2 3 - 2^1 3 + 1$ y $n = 256$. Luego de calcular dos vectores aleatorios, s_1 y s_2, de modo que cada coeficiente de tales vectores es un elemento de R_q con coeficientes pequeños y de tamaño, a lo sumo, η. Por último, se determinan la clave pública, pk, y la clave privada, sk.

Algoritmo 1 Dilithium: Generación de claves

1. $\mathbf{A} \leftarrow_R R_q^{k \times l}$
2. $(s_1, s_2) \leftarrow_R S_\eta^l \times S_\eta^k$
3. $t := \mathbf{A}s_1 + s_2$
4. $pk = (\mathbf{A}, t)$
5. $sk = (\mathbf{A}, t, s_1, s_2)$
6. **return** (pk, sk)

Para la elaboración de la firma (véase el Algoritmo 2), se genera un vector de enmascaramiento del polinomio, y, con coeficientes menores que γ_1. El firmante determina $\mathbf{A}y$ y define los bits de orden superior como w_1. Cada coeficiente α en $\mathbf{A}y$ se puede escribir como $\alpha = w_1 \cdot 2\gamma_2 + w_0$ donde $|w_0| \leq \gamma_2$. El desafío c se crea como el hash del mensaje y w_1. La salida c es un polinomio en R_q con exactamente τ coeficientes iguales a ± 1 y el resto igual a 0. La firma se calcula como $z = y + cs_1$. Como entrada al algoritmo se consideran la clave privada y el mensaje a firmar: sk, M.

Por último, el Algoritmo 3 muestra el proceso para la verificación de la firma. Para ello se calcula w_1' como los bits de orden superior de $\mathbf{A}z - ct$ y se acepta la firma como válida si todos los coeficientes de z son menores que $\gamma_1 - \beta$ y si c es el hash del mensaje y w_1'. Nótese que $\mathbf{A}z - ct = \mathbf{A}y - cs_2$,

Algoritmo 2 Dilithium: Elaboración de la firma digital

1. $z := \perp$.
2. **while** $z = \perp$
   a) $y \leftarrow S_{\gamma_1 - 1}^l$
   b) $w_1 := HighBits(\mathbf{A}y, 2\gamma_2)$
   c) $c \in B_\tau := H(M||w_1)$
   d) $z := y + cs_1$
3. **if** $||z||_\infty \geq \gamma_1 - \beta$
4. **or** $||\text{LowBits}(\mathbf{A}y - cs_2, 2\gamma_2)|| \geq \gamma_2 - \beta$
5. **then** $z := \perp$
6. **return** $\sigma = (z, c)$.

por lo que basta con probar que

$$\text{HighBits}(\mathbf{A}y, 2\gamma_2) = \text{HighBits}(\mathbf{A} - cs_2, 2\gamma_2).$$

Toda firma válida cumple que

$$||\text{LowBits}(\mathbf{A}y - cs_2, 2\gamma_2)||_\infty < \gamma_2 - \beta.$$

Además, como los coeficientes de cs_2 son más pequeños que β, es claro que sumar cs_2 no es suficiente para aumentar cualquier coeficiente de orden inferior para que tenga una magnitud de al menos γ_2.

Algoritmo 3 Dilithium: Verificación de la firma digital

1. $w_1' := HighBits(Az - ct, 2\gamma_2)$
2. **if** $c = H(M||w_1')$ **and** coeffs$(z) < \gamma_1 - \beta$
3. **then return** "Firma válida"
4. **else return** "Firma inválida"

III-B. FALCON

FALCON (*FAst-Fourier Lattice-based COmpact signatures over NTRU*) es uno de los tres esquemas de firma digital seleccionado por el NIST después de la tercera ronda [6]; si bien, al igual que Dilithium, su seguridad se basa en un tipo de problema definido sobre retículos, en este caso, el problema es el SIS sobre retículos NTRU [9].

El diseño de FALCON sigue el enfoque de "resume-y-firma". Además, hace uso de la reducción de seguridad de NTRU [15] y de una implementación práctica de la propuesta GPV (Gentry-Peikert-Vaikuntanathan) sobre retículos NTRU [16].

FALCON, como la mayor parte de los esquemas de firma digital, consta de tres algoritmos: generación de claves, elaboración de la firma y verificación de la misma. A continuación se presenta cada uno de ellos de forma esquemática. Además, la estructura subyacente a FALCON está definida por el anillo $R_q = \mathbb{Z}[x]/\langle\phi\rangle$, donde $\phi = x^n + 1$ el polinomio ciclotómico con $n = 2^k$.

El Algoritmo 4 muestra cómo generar tanto la clave pública, pk, como privada, sk. El algoritmo utiliza tres subrutinas que, por razones de espacio, no entramos a detallar pero que el lector puede consultar en la documentación de referencia [9]. Estas son: NTRUGen, FFT y ffLDL*. NTRUGen genera los cuatro polinomios NTRU que cumplen la siguiente ecuación:

$$fG - gF = 0 \pmod{\phi}, \quad (1)$$

donde $f, g, F, G \in R_q$. Es de destacar que los polinomios g y f obtenidos por NTRUGen se generan aleatoriamente y deben verificar determinadas condiciones. Una de ellas es que se pueda calcular la clave pública $h = gf^{-1} \pmod{\phi, q}$. Además, hay otra condición que afecta a los polinomios g y f que está relacionada con la longitud de las firmas.

NTRUGen incluye la función NTRUSolve que permite calcular otros dos polinomios, cortos, G y F de modo que verifiquen la ecuación (1).

Algoritmo 4 Falcon: Generación de claves (ϕ, q)

1. $f, g, F, G \leftarrow$ NTRUGen(ϕ, q)
2. $\mathbf{B} \leftarrow \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}$
3. $\hat{\mathbf{B}} \leftarrow$ FFT(B)
4. $\mathbf{G} \leftarrow \hat{\mathbf{B}} \times \hat{\mathbf{B}}^*$
5. $T \leftarrow$ ffLDL$^*(\mathbf{G})$
6. **for** each leaf of T **do**
7. leaf value $\leftarrow \sigma/\sqrt{\text{leaf value}}$
8. $sk \leftarrow (\hat{\mathbf{B}}, T)$
9. $pk \leftarrow gf^{-1} \pmod{q}$
10. **return** (pk, sk)

La elaboración de la firma se lleva a cabo tal y como se presenta en el Algoritmo 5. Este algoritmo emplea como entrada el mensaje a firmar, M, la clave privada, sk, y el valor $\lfloor \beta^2 \rfloor$. Además, hace uso de las funciones HashToPoint, ffSampling, invFF y Compress.

Algoritmo 5 Falcon: Elaboración de la firma $(M, sk, \lfloor \beta^2 \rfloor)$

1. $r \leftarrow \{0,1\}^{320}$
2. $c \leftarrow$ HashToPoint$(r||M, q, n)$
3. $z \leftarrow$ ffSampling$_n(f, T)$
4. $s' = (t - z)\hat{\mathbf{B}}$
5. **while** $||s'|| > \lfloor \beta^2 \rfloor$ **do**
6. $(s_1, s_2) \leftarrow$ invFF(s')
7. $s \leftarrow$ Compress$(s_2, 8 \cdot sbytelen - 328)$
8. **return** $\sigma = (r, s)$

El Algoritmo 6 lleva a cabo la verificación de la firma digital y utiliza las funciones HashToPoint y Decompress. Sus entradas son el mensaje, M, la firma digital correspondiente, σ, la clave pública, pk, y el valor $\lfloor \beta^2 \rfloor$.

Algoritmo 6 Falcon: Verificación de la firma $(M, \sigma, pk, \lfloor \beta^2 \rfloor)$

1. $c \leftarrow$ HashToPoint$(r||M, q, n)$
2. $s_2 \leftarrow$ Decompress$(s, 8 \cdot sbytelen - 328)$
3. **if** $s_2 = \perp$
4. **then return** "Rechazo"
5. $s_1 \leftarrow c - s_2 h \pmod{q}$
6. **if** $||(s_1, s_2)||^2 \leq \lfloor \beta^2 \rfloor$
7. **then return** "Aceptación"
8. **else return** "Rechazo"

IV. Análisis y resultados obtenidos

Aunque el NIST ya haya publicado el borrador del estándar para ML-DSA (Dilithium), no vamos a comparar esta versión con la de FALCON, dado que el borrador de este último no ha sido publicado. Así pues, en este trabajo vamos a utilizar la propuesta de Dilithium y la de FALCON que pasaron a la cuarta ronda del NIST y cuyas implementaciones han sido publicadas.

Como ya hemos mencionado, Dilithium se basa en los problemas MSIS y MLWE para prevenir la falsificación de las firmas y la recuperación de claves, respectivamente. Por su parte, FALCON basa su seguridad en la versión no modular de SIS para su diseño esquemático y en el uso de retículos NTRU, que también le confieren una gran compacidad.

Por otra parte, Dilithium proporciona conjuntos de parámetros que responden a los niveles de seguridad de fuerza bruta 2, 3 y 5, y FALCON solo presenta dos versiones para alcanzar los niveles 1 y 5, que a veces se denominan FALCON-512 y FALCON-1024. Por lo tanto, para extraer conclusiones más generales, la comparación que hemos realizado considera todos los conjuntos de parámetros de cada esquema, dado que sólo considerar solo el nivel más alto de seguridad (el único común de ambas propuestas) no sería tan representativo.

IV-A. Software y arquitectura empleados

Los resultados del rendimiento que se dan para cada propuesta, en su documentación original, se obtienen utilizando los criterios y la arquitectura que cada grupo desarrollador ha considerado oportuno. Este hecho es muy destacable y a tener en cuenta dado que, por ejemplo, FALCON utiliza muestreos sobre distribuciones no uniformes y también una aritmética de coma flotante cuyo rendimiento está influenciado, en gran medida, por el procesador utilizado.

En cualquier caso, el código de referencia utilizado para esta comparación pertenece a la librería de software LIBOQS (*LIbrary Open-Quantum-Safe*), que es una biblioteca de código abierto escrita en lenguaje C y que incluye los principales algoritmos criptográficos poscuánticos, en particular los dos esquemas de firma considerados en este trabajo.

LIBOQS puede ser utilizada en los sistemas operativos Linux, Windows y macOS y es compatible con varias arquitecturas y compiladores. Todas las implementaciones en esta librería se derivan de los códigos de referencia optimizados y enviados por los autores en la convocatoria de estandarización del NIST. Por lo tanto, hemos utilizado en esta comparativa el código estándar optimizado contenido en esta biblioteca.

Además de emplear la misma librería, con el fin de obtener una comparación significativa, hemos considerado la misma arquitectura para los dos esquemas de firma y hemos utilizado la prueba disponible en el código dado para medir los tamaños (en bytes) de las claves y la firma y el tiempo medio de ejecución de cada procedimiento (en ciclos de reloj y microsegundos μ-sec). Todos los resultados que se muestran en este análisis se han obtenido empleando un procesador Intel Core i7-4790 a 3,60 GHz y con sistema operativo Linux.

IV-B. Comparación en tamaños de datos

Los tamaños de la clave pública, la clave privada (secreta) y la firma de cada algoritmo permiten una comparación de seguridad, ya que los rendimientos muestran la cantidad de datos (y el tamaño del secreto) necesarios para alcanzar un determinado nivel de seguridad. Estos datos se obtuvieron de la documentación de respaldo enviada al NIST por los

desarrolladores de Dilithium [17] y FALCON [18]. Los resultados que hemos obtenido acerca de los tamaños de los datos mencionados, para cada esquema, se muestran en la Tabla I. Dicha tabla incluye todos los conjuntos de parámetros proporcionados en las propuestas enviadas al NIST y, por lo tanto, contiene todos los niveles de seguridad disponibles para ambos esquemas.

Tabla I
TAMAÑO EN BYTES DE LAS CLAVES Y FIRMA SEGÚN LOS NIVELES DE SEGURIDAD DE CADA ESQUEMA DE FIRMA.

	Dilithium			FALCON	
Nivel de seguridad	2	3	5	1	5
Clave pública	1312	1952	2592	897	1793
Clave privada	2528	4000	4864	7552	13953
Firma	2420	3293	4595	666	1280
Total	6260	9245	12051	9115	17026

Los tamaños de las claves pública y privada y de la firma para cada uno de los niveles de seguridad de cada esquema analizado se representan gráficamente en la Figura 1, para una mejor visualización de los resultados.

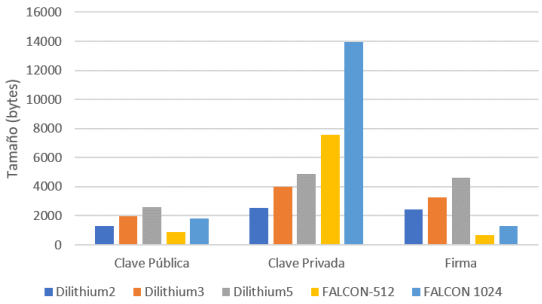

Figura 1. Tamaño de las claves y firma por niveles de seguridad (bytes).

En general, todos los tamaños asociados al esquema Dilithium y mostrados en la Tabla I son manejables para los tres niveles. Además, la clave pública es la que tiene menor tamaño, lo cual es especialmente interesante, dado que es un valor compartido. Los incrementos en tamaño para los datos considerados de Dilithium se comportan de manera proporcional a medida que aumenta el nivel de seguridad. La razón de este comportamiento está en los parámetros utilizados al modificar la seguridad, dado que dependen, fundamentalmente, de las dimensiones de la matriz $\mathbf{A}$, al margen de la adecuación correspondiente del resto de valores a considerar. Este diseño proporciona un intercambio de tamaños consistente a la hora de seleccionar un nivel de seguridad u otro.

Las diferencias de estos tamaños entre FALCON y Dilithium son notables. Debe recordarse que el uso de retículo de tipo NTRU en FALCON tiene como objetivo lograr que los datos sean más compactos. Como se puede ver en la Tabla I, los tamaños de la clave pública y de la firma son significativamente más pequeños que los correspondientes de Dilithium. De hecho, estos tamaños son notablemente más pequeños incluso para el caso del nivel 5 en FALCON (FALCON-1024), cuya firma es casi la mitad del tamaño de la firma para la versión menos segura de Dilithium.

No ocurre lo mismo para el caso de la clave privada, donde es FALCON quien sale perjudicado con relación a Dilithium. Este hecho es el que permite afirmar que la suma de las longitudes de los tres valores para cada esquema hace que Dilithium salga beneficiado en los tres niveles de seguridad. En particular el total del tamaño requerido para el FALCON de nivel de seguridad 1 es semejante al del nivel de seguridad 3 de Dilithium.

En todo caso, la notable diferencia en los tamaños de las claves privadas podría ser asumible a la vista de la reducción en los tamaños de la clave pública y la firma, dado que los valores que se transmiten son precisamente estos más pequeños; mientras que la clave privada se mantiene en secreto por su propietario.

IV-C. Comparación en rendimiento

Dado que los resultados de rendimiento de los algoritmos considerados, FALCON [18] y Dilithium [17], son proporcionados por los equipos desarrolladores utilizando distintos criterios, medidas y arquitecturas, el principal objetivo de esta sección es mostrar algunos datos que permitan realizar una comparación más adecuada, para lo que se emplearán criterios, medidas y arquitecturas análogos.

Analizamos, así, el rendimiento de cada uno de los tres algoritmos de que consta cada uno de los dos esquemas de firma basados en retículos, esto es, generación de claves, elaboración de la firma y verificación de la misma, para todos sus niveles de seguridad.

Se han realizado dos análisis diferentes en cuanto al rendimiento. El primero de ellos relativo a los tiempos de computación empleados y el segundo relacionado con el consumo de la CPU. En ambos casos se ha empleado la prueba proporcionada por cada uno de los equipos desarrolladores y que fueron enviadas a la convocatoria del NIST. Como ya se ha indicado, haremos uso de las implementaciones optimizadas que se basan en las propuestas originales y que están incluidas en LIBOQS.

En la Tabla II se presentan los resultados obtenidos para el análisis del rendimiento de tiempos medidos en microsegundos (μsec) para cada algoritmo de cada esquema de firma y cada nivel; mientras que la Tabla III muestra los resultados correspondientes para el rendimiento de uso de la CPU medidos en ciclos de reloj.

Para proporcionar una comparación gráfica más visual de los resultados obtenidos relativos al uso de CPU, se incluyen varias figuras. Los tiempos de rendimiento totales para cada propuesta (en ciclos de reloj) se presentan en la Figura 2.

Por otra parte, debido a la diferencia en el número de ciclos para cada uno de los algoritmos considerados para cada uno de los dos esquemas de firma, hemos separado el algoritmo de generación de claves de los de firma y verificación. Así, la Figura 3 muestra los resultados para el primero de los algoritmos y la Figura 2 presenta los valores obtenidos para los otros dos algoritmos, para los diferentes niveles de seguridad, tanto para Dilithium como para FALCON.

Como se puede ver por los resultados generales mostrados en la Figura 2, FALCON es mucho menos eficiente que Dilithium. El primero tiene un tiempo de desempeño mayor, hasta tal punto que la escala necesaria para medirlo no permite apreciar adecuadamente los tiempos empleados de Dilithium.

Tabla II
RENDIMIENTO DE TIEMPO DE LOS ESQUEMAS DE FIRMA DILITHIUM Y FALCON (μSEC).

	CRYSTALS-Dilithium			FALCON	
	Tiempo medio de ejecución (μsec)				
Nivel de seguridad	2	3	5	1	5
Generación de claves	24,811	46,271	72,938	6790,418	19318,732
Firma	61,026	113,801	140,293	274,317	559,497
Verificación	24,837	42,864	69,170	43,884	87,906
Total	106,177	173,062	264,140	6996,455	19779,455

Tabla III
RENDIMIENTO DE CPU DE LOS ESQUEMAS DE FIRMA DILITHIUM Y FALCON (EN CICLOS DE RELOJ).

	CRYSTALS-Dilithium			FALCON	
	Tiempo medio de ejecución (ciclos de reloj)				
Nivel de seguridad	2	3	5	1	5
Generación de claves	57143	106605	168047	15652763	44532312
Firma	140615	262256	323315	632273	1289647
Verificación	57204	98751	159387	101105	202576
Total	244699	398871	608812	16126491	45553057

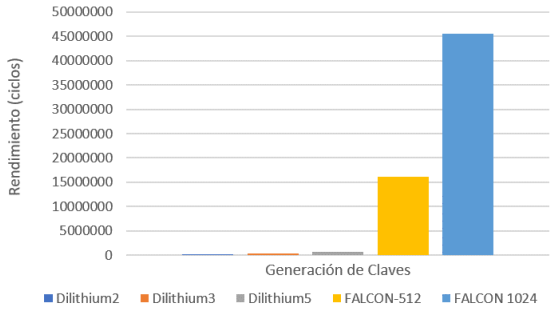

Figura 2. Rendimiento total de CPU de Dilithium y FALCON por niveles.

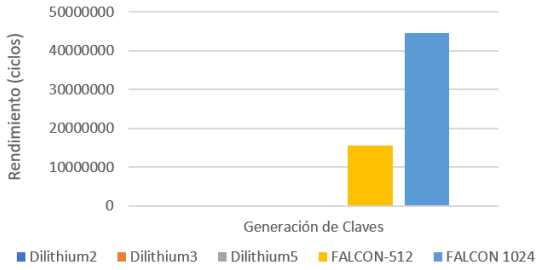

Figura 3. Rendimiento CPU para generar claves de Dilithium y FALCON.

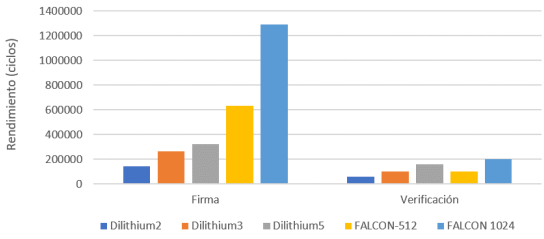

Figura 4. Rendimiento de CPU para la firma y la verificación.

Por esa razón, hemos separado cada uno de los algoritmos particulares (ver Figuras 3 y 4), de modo que pueda observarse la influencia de cada uno de ellos en el incremento en el tiempo para FALCON y obtener escalas más apropiadas.

La principal causa de esta diferencia se encuentra en el algoritmo de generación de claves (Figura 3). Como puede verse, este algoritmo provoca un impacto crítico en los tiempos de rendimiento de FALCON. Por su parte, Dilithium sólo realiza operaciones simples para generar sus claves, siendo la multiplicación en el entorno de la NTT la más costosa; mientras que FALCON requiere procedimientos mucho más complejos.

Adentrándonos con más detalle en las causas de esta gran diferencia en el rendimiento, hemos de señalar, en primer lugar, que FALCON utiliza la solución (secreta) de una ecuación NTRU y, por tanto, el algoritmo de generación de claves debe resolver esta ecuación para poder obtener su solución. Este proceso de resolución supone realizar varias operaciones internas que contribuyen, en gran medida, a incrementar el tiempo de ejecución. Posteriormente, se calcula la clave privada (secreta) correspondiente y esto significa obtener el árbol FALCON. También este procedimiento recursivo es largo y complejo y, por tanto, provoca el consiguiente aumento en el tiempo de ejecución.

Como resultado, se tiene que la complejidad del esquema de firma FALCON acarrea un aumento muy significativo de eficiencia en términos de rendimiento.

Los resultados de rendimiento del algoritmo de elaboración de firmas son más fácilmente comparables, dado que no hay tanta diferencia como en el caso de la generación de claves. Se puede observar (Figura 4) que aunque FALCON es significativamente más lento que Dilithium, el número de ciclos de reloj de ambas son más cercanos. El origen de estas diferencias se deben, principalmente, a dos causas. La primera es que el tamaño de la clave privada de FALCON es significativamente mayor que la de Dilithium y en el proceso de elaboración de la firma se cifra el hash del mensaje haciendo uso de esta clave, por lo que FALCON requiere más tiempo de ejecución. La segunda es que, además, FALCON utiliza un muestreador de trampilla (de hecho lleva a cabo un muestreo rápido de Fourier) que involucra el árbol FALCON y un redondeo aleatorio según una distribución gaussiana discreta. Esta operación es, en realidad, el núcleo del proceso

de la elaboración de las firmas y es, claramente, más compleja que las utilizadas para Dilithium, cuya operación más costosa es la multiplicación en el entorno de la NTT. Estas son las razones que explican la diferencia en cuanto al rendimiento de ambos esquemas, en perjuicio de FALCON, en el algoritmo de elaboración de firmas.

En los algoritmos de verificación de firma, los rendimientos de ambos esquemas son mejores que en los de su elaboración, lo que es habitual porque este algoritmo suele ser más sencillo que el de la firma en sí. Puede apreciarse (ver Figura 4) que en este caso, la mejora en el rendimiento de FALCON es muy notable, y los resultados son similares a los de Dilithium. Las versiones de mayor nivel de seguridad de Dilithium tienen un tiempo de verificación incluso mayor que el FALCON de menor nivel. De manera similar a lo que se observó anteriormente para los tamaños, esta característica puede ser digna de destacar, dado que el proceso de verificación de una firma digital se puede llegar a realizar varias veces, mientras que la generación de claves solo se ejecuta una vez en un largo período de tiempo y la elaboración de la firma solo en contadas ocasiones.

La explicación de estos resultados en la verificación de la firma es que ambos esquemas realizan operaciones muy similares: sumas, multiplicaciones y comparaciones entre variables. Por tanto, el tamaño de estas variables es lo que determina, en gran medida, el tiempo de ejecución. Debe tenerse en cuenta que, en este caso, no se emplea la clave privada y, aunque las claves públicas y la firma de FALCON son menores que las de Dilithium (ver Tabla I), sus tiempos en la verificación no difieren mucho de los de Dilithium, como cabría de esperar. La razón de que no exista una diferencia mayor debida a los tamaños de las claves públicas y las firmas de ambos esquemas se debe a la falta de compensación en la aritmética a realizar en FALCON, que es más costosa que la de Dilithium.

V. Conclusiones

El análisis presentado en esta investigación sugiere que Dilithium es, en circunstancias generales y en los casos regulares de uso, un esquema de firma más seguro y eficiente que FALCON. Señalamos dos razones para esta conclusión.

La primera es que el rendimiento es significativamente mejor, ya que la generación de claves en FALCON es muy costosa. Incluso considerando que esta generación se realiza sólo de vez en cuando, los resultados de Dilithium para la firma y la verificación también son más eficientes. El diseño conceptual simplificado de Dilithium lo hace más adecuado en términos de rendimiento.

La segunda razón es que, aunque los tamaños de los valores de Dilithium no siempre son menores que los de FALCON, su rango es más manejable y los valores de su rendimiento son proporcionales tanto a los tamaños de los parámetros empleados como a los diferentes niveles de seguridad establecidos.

Todo ello da a entender que el diseño de Dilithium es más compacto y su seguridad es más fácil de ajustar a las necesidades requeridas. Por su parte, FALCON ofrece otras ventajas interesantes para ciertos casos de uso particulares, donde puede ser la mejor opción; a pesar de ser más complejo y ofrezca menos niveles de seguridad que Dilithium. Estos casos de uso son aquellos en los que la generación de las claves pública y privada no sea un factor determinante en cuanto al rendimiento del esquema, dado que las mismas no se generan con demasiada frecuencia y, además, son menores que las claves generadas por Dilithium.

Como trabajo futuro se pretende extender este tipo de análisis a otras plataformas, en particular a aquellas que tengan menor capacidad de cómputo que un ordenador estándar. Es de esperar que los resultados sean similares, en términos generales, de modo que confirmen las conclusiones obtenidas en este estudio. A la vez, este estudio permitiría tener una idea más clara sobre las posibles dificultades a la hora de implementar estos esquemas de firma en dispositivos con recursos computacionales limitados, más cercanos a las implementaciones prácticas, como puede ser en los dispositivos empleados en la Internet de las cosas (Internet of Things o IoT).

Otra cuestión a considerar en el futuro sería la de comparar los resultados obtenidos en este estudio con los obtenidos al realizar un análisis similar sobre los mismos aspectos para otros esquemas de firma postcuánticos, no basados en retículos, como es el caso de SPHINCS$^+$.

Agradecimientos

Este trabajo ha sido parcialmente financiado por la Agencia Estatal de Investigación (AEI) del Ministerio de Ciencia e Innovación (MCIN) y por la Unión Europea NextGerationUE/PRTR en los proyectos P2QProMeTe (PID2020-112586RBI00/AEI/10.13039/501100011033, cofinanciado por el Fondo Europeo de Desarrollo Regional: FSE, FEDER y FEDER, UE); y QURSA (TED2021-130369BC33, MCIN/AEI/10.13039/501100011033); en parte por el Proyecto ORACLE (PCI2020-120691-2, MCIN/AEI/10.13039/501100011033) y en parte por el programa de investigación e innovación Horizonte 2020 de la UE, proyecto SPIRS (Grant Agreement N° 952622). E.I.H y L.H.A. desean agradecer su apoyo a los proyectos del CSIC EFiDiP y SAIACAP, respectivamente.

Referencias

[1] P. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997, https://doi.org/10.1137/S0097539795293172.

[2] R. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978, https://doi.org/10.1145/359340.359342.

[3] N. Koblitz, "Elliptic curve cryptosytems," *Math. Comp.*, vol. 48, no. 177, pp. 203–209, 1987, https://doi.org/10.1090/S0025-5718-1987-0866109-5.

[4] V. S. Miller, "Use of elliptic curves in cryptography," *Lecture Notes Comput. Sci.*, vol. 218, pp. 417–426, 1986, https://doi.org/10.1007/3-540-39799-X_31.

[5] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory*, vol. 22, pp. 644–654, 1976, htpps://doi.org/10.1109/TIT.1976.1055638.

[6] NIST, "Post-quantum cryptography. Selected algorithms 2022," On-line publication, 2022, https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[7] P. Schwabe, R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, G. Seiler, and D. Stehle, "CRYSTALS-KYBER," Online publication, 2020, https://pq-crystals.org/kyber/index.shtml.

[8] V. Lyubashevsky, L. Ducas, E. Kiltz, T. Lepoint, P. Schwabe, G. Seiler, and D. Stehle, "CRYSTALS-DILITHIUM," Online publication, 2020, https://pq-crystals.org/dilithium/index.shtml.

[9] T. Prest, P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "FALCON," Online publication, 2020, https://falcon-sign.info/.

[10] A. Hulsing, D. J. Bernstein, C. Dobraunig, M. Eichlseder, S. Fluhrer, S.-L. Gazdag, P. Kampanakis, S. Kolbl, T. Lange, M. M. Lauridsen, F. Mendel, R. Niederhagen, C. Rechberger, J. Rijneveld, P. Schwabe, and J.-P. Aumasson, "SPHINCS+," Online publication, 2020, https://sphincs.org/.

[11] NIST, "PQC standardization process: Announcing four candidates to be standardized, plus fourth round candidates," Online publication, 2022, https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4.

[12] ——, *Module-Lattice-Based Digital Signature Standard*, National Institute of Standard and Technology, NIST FIPS 204, 2023, https://doi.org/10.6028/NIST.FIPS.204.ipd.

[13] V. Lyubashevsly, "Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures," *Lecture Notes Comput. Sci.*, vol. 5912, pp. 598–616, 2009, https://doi.org/10.1007/978-3-642-10366-7_35.

[14] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Dilithium: A lattice-based digital signature scheme," in *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018, pp. 238–268, https://doi.org/10.13154/tches.v2018.i1.238-268.

[15] D. Stehlé and R. Steinfeld, "Making NTRU as secure as worst-case problems over ideal lattices," in *Proc. Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology - EUROCRYPT 2011, Lecture Notes Comput. Sci.*, vol. 6632, 2011, pp. 27–47, https://doi.org/10.1007/978-3-642-20465-4_4.

[16] L. Ducas, V. Lyubashevsky, and T. Prest, "Efficient identity-based encryption over NTRU lattices," in *Proc. International Conference on the Theory and Application of Cryptology and Information Security, Advances in Cryptology - ASIACRYPT 2014, Lecture Notes Comput. Sci.*, vol. 8874, 2014, pp. 22–41, https://doi.org/10.1007/978-3-662-45608-8_2.

[17] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevesky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Dilithium: Algorithm specifications and supporting documentation," NIST, Tech. Rep., 2020, https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf.

[18] P. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "FALCON: Fast-Fourier lattice-based compact signatures over NTRU," NIST, Tech. Rep., 2018, https://falcon-sign.info/falcon.pdf.