Escuela Politécnica Superior de Jaén

Grado en Ingeniería Informática

UNIVERSIDAD DE JAÉN

Nombre del Centro

Trabajo Fin de Grado

CRIPTOGRAFÍA POST-CUÁNTICA

Alumno: Sergio Sánchez López

Tutor: Manuel José Lucena López

Dpto: Departamento de Informática

Septiembre, 2022

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Manuel José Lucena López, tutor del Proyecto Fin de Carrera titulado: Criptografía Post-Cuántica, que presenta Sergio Sánchez López, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

En Jaén, a 8 de Septiembre de 2022

El alumno:

Sergio Sánchez López

Los tutores:

Firmado digitalmente por LUCENA LOPEZ, MANUEL JOSE.
Fecha: 09.09.2022 - 12:05:49.

Manuel José Lucena López

Índice

1. Introducción

La Computación Cuántica[1] es un tema que llama la atención de las personas. No se si será porque tiene nombre de película de ficción, porque los medios le hayan dado fama o porque estamos entrando en una etapa en la que las personas se preocupan cada vez más por los cambios tecnológicos, ya que hacemos uso de ellos a diario, y de no hacerlo, tenemos la extraña sensación de quedarnos atrás.

Pero lo que más me despierta curiosidad sobre ésto es, que personas que no tienen conocimientos en este tipo de campos (con las que he hablado sobre ello), lejos de preocuparse por los avances que se puedan obtener utilizando esta herramienta en diferentes sectores como la economía y servicios financieros, química, medicina y salud, logística y cadena de suministro, energía y agricultura, etc. Lo que más me preguntan es sobre “algoritmos que se rompen” y “quien tiene esos ordenadores” y si “el que lo tenga lo va a controlar todo”.

Es decir una preocupación directa por su seguridad y privacidad, porque, aunque sin conocimientos técnicos, ya sea por ayuda de las noticias o el aumento de concienciación de la ciberseguridad, sienten que esos “ordenadores tan poderosos” podrían repercutir en sus vidas sin que puedan hacer nada.

Por lo tanto, la Criptografía Post-Cuántica es un tema al que tenemos que prestar mucha atención en los próximos años, pues la criptografía es el núcleo de la seguridad de la información.

[1] Computación Cuántica: https://www.ibm.com/es-es/topics/quantum-computing

1.1. Objetivos

Con el presente trabajo los objetivos que se pretenden conseguir son los siguientes.

Realizar un breve análisis de por qué la computación cuántica permitirá romper algunos de los algoritmos de cifrado actuales. Realizar una revisión del estado del arte en Criptografía post-cuántica. Evaluar alguna de las propuestas actuales en el campo de la Criptografía post-cuántica, incluyendo sus fundamentos teóricos. Implementar una aplicación capaz de usar alguna técnica criptográfica post-cuántica.

1.2. Temas a tratar

Los siguientes puntos citados serán los temas que vamos a desarrollar en este trabajo:

1. Introducción: Explicación sobre qué es la Computación Cuántica.
    - Nacimiento
    - Características
    - Capacidad

2. Problema al que nos enfrentamos: Descripción del problema que supone la llegada de los ordenadores cuánticos.
    - Problema desde un punto de vista técnico y práctico.

3. Análisis de las propuestas: Descripción de los algoritmos que podrían llegar a convertirse en estándares para la Criptografía Post-Cuántica.
    - Proyecto Post-Quantum Cryptography del NIST.
        - Situación actual del concurso.
        - Candidatos.
4. Programa de ejemplo.
    - Caso de uso de un algoritmo para las dos categorías de estandarización existentes.

5. Conclusión

1.3. Computación Cuántica

A principios del siglo XX surge la Mecánica Cuántica[2], definida como un campo de la física que se encarga de describir cómo se comporta la naturaleza a niveles subatómicos, como el de los fotones, comportamiento para el que la mecánica clásica era incapaz de encontrar resultados satisfactorios.

Las propiedades de esta nueva mecánica son:

- **Superposición**: un sistema no solo puede estar en un estado A o B, sino que puede representar una mezcla de ambos.
- **Colapso**: tras haber medido un sistema y obtener uno de sus resultados posibles de forma probabilística, cualquier otra medición alcanzará el mismo resultado.
- **Entrelazamiento**: independientemente de la distancia que los separe, cualquier cambio que sufra un sistema A afectará simultáneamente a otro sistema B.
- **Incertidumbre**: no se puede determinar a la vez la posición de una partícula y su movimiento.

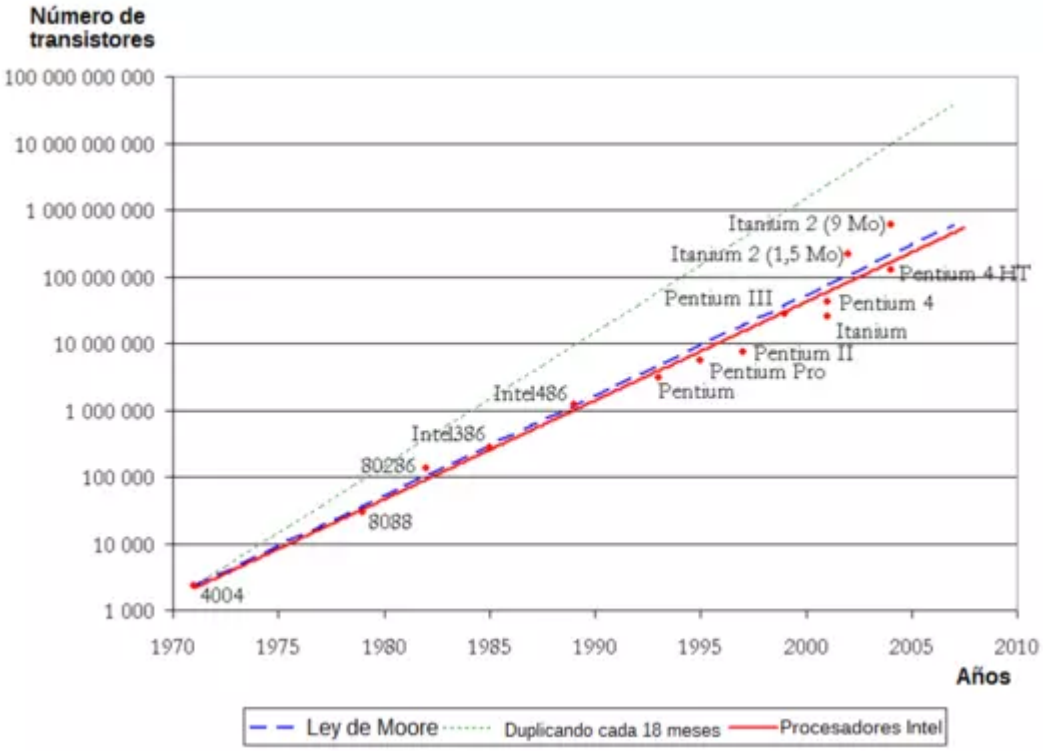
Casi dos siglos después de este nacimiento, al principio de los años ochenta, Richard Feynman ideó la construcción de los ordenadores que poseían variables cuánticas como estados internos.

Feynman, junto con el físico estadounidense Paul Benioff y el matemático ruso Yuri Manin, consiguieron iniciar las bases de la Computación Cuántica sobre las que hoy día se basan todos los prototipos más funciones tales como:

- Eagle (IBM) [Eagle (IBM)]
- Borealis (Xanadu Quantum Technologies) [Borealis (Xanadu Quantum Technologies)]

[2] Mecánica Cuántica: Feynman, R. P., Leighton, R. B., & Sands, M. (2000). Física: Mecánica cuántica. III. Pearson Educación.

Hasta ahora, la mejora en los procesadores de los computadores clásicos venía determinada por la Ley de Moore[3]:



Esta ley, escrita en 1965 por Gordon Moore decía en un principio que el número de transistores por unidad de superficie en circuitos integrados se duplicaría cada año, pero más tarde en 1975, la modificó para aumentar el tiempo a dos años.

Actualmente, y esto es un hecho anunciado por los mismos fabricantes, estamos cerca de que dicha ley se rompa, ya que nuestra capacidad de reducir el tamaño de transistores para aumentar su cantidad en nuestros procesadores es físicamente más difícil. Esto implicaría que estaríamos llegando a un estancamiento en nuestra capacidad de cómputo, por lo que podemos ver la Computación Cuántica como una solución a dicha limitación y una posibilidad de continuar desarrollando avances.

[3] Ley de Moore: https://www.revista.unam.mx/vol.6/num7/art65/jul_art65.pdf

Como se conoce, la informática clásica está basada en el concepto de bit, el cual toma valores 0 o 1, a diferencia de la informática cuántica que trabaja con quantum bit o también llamados qubits. En ambos casos, representan la unidad mínima de información en sus sistemas.

Pero lo principal no es que se llamen de manera diferente sino que el qubit puede hallarse simultáneamente en los dos estados 0 y 1, por lo que con 10 qubits manejaremos 1.024 estados simultáneos. Visto de una forma más matemática, estaríamos hablando de que los qubits toman valores exponenciales, por lo tanto, un sistema con N qubits tendría la capacidad de realizar 2^N operaciones simultáneas.

Por ahora este tipo de procesadores que aplican estas leyes mecánicas y trabajan siguiendo estas teorías no tienen fines comerciales por limitaciones como que algunos necesitan alcanzar temperaturas cercanas al cero absoluto para poder funcionar. Pero no estamos lejos de encontrar alguna alternativa, ni tampoco tiene que haber la posibilidad de que cualquiera pueda acceder a uno de estos computadores como para que su existencia represente un problema real.

2. Problema al que nos enfrentamos

El principal problema al que nos enfrentamos es simple:

- La seguridad de nuestra información se consigue con algoritmos que tienen una muy consolidada y demostrada robustez computacional, y están aprobados por entidades oficiales dedicadas a ello.
- Los ordenadores cuánticos serían capaces de romper dichos algoritmos por su capacidad.
- Hay que crear nuevos algoritmos criptográficos con sistemas basados en problemas matemáticos lo suficientemente complejos para que un ordenador cuántico no pueda resolverlos de forma práctica.

Quizás leyendo lo anterior se podría llegar a pensar: Y… ¿En qué me afecta esto a mí?

2.1. Visión Técnica

Viéndolo desde un punto de vista más técnico podríamos comenzar nombrando dos de los algoritmos más sonados a la hora de la “Destrucción de la Criptografía tradicional”.

2.1.1. Algoritmo de Shor [Shor94]

Este algoritmo fue pensado por el profesor de matemáticas del MIT Peter Shor Williston en 1994, y luego fue mejorado en 1996.

Se teorizaba con que, aplicado en un ordenador cuántico, este algoritmo lograría encontrar factores de un número de forma eficiente. Esto implicaría que de una forma sencilla, podríamos resolver los problemas sobre los que se basan algoritmos de clave pública estandarizados como RSA.

2.1.2. Algoritmo de Grover [Grover]

Para la criptografía simétrica nos encontramos con este otro, creado por Lov Grover en 1996, también llamado de búsqueda cuántica.

Un Grover aplicado sería capaz de encontrar con alta probabilidad la entrada única de un resultado de salida para una función de caja negra.

Esto lo hace en $O(\sqrt{N})$ evaluaciones de la función, cuando en la computación clásica se necesitan al menos $O(N)$. Esto obligaría a duplicar la longitud de las claves para estar a salvo por el momento.

2.2.Visión de hecho

Que seamos capaces de aplicar algoritmos como los anteriormente nombrados implica que protocolos muy implantados ha día de hoy tales como:

Protocolo	Uso	Implicaciones
WPA2	Conexiones Wi-Fi	- No tendríamos conexiones inalámbricas seguras.

SSL/TLS	Conexiones Web	- Navegaciones cotidianas como:
		  - Leer un periódico digital.
		  - Acceder a la aplicación del Banco.

SSH	Conexión Cliente - Servidor	- Conexiones remotas desprotegidas.

Simplemente con estos tres puntos ya se podría hablar de que la mayoría de las acciones que realizamos diariamente estarían expuestas.

Y ya no solo hablar de lo que es responsabilidad de cada uno, si no de todos los datos que las empresas poseen de los usuarios serían vulnerables desde las redes sociales, las cuales poseen datos privados, hasta contextos más críticos como datos bancarios o incluso médicos.

Hay que añadir que la creación de un ordenador cuántico con la capacidad de ejecutar este tipo de algoritmos capaces de romper nuestra seguridad todavía necesitan madurar un poco y no estará al alcance como uno doméstico. Y con esto quiero decir que con las investigaciones que se están realizando, los expertos hablan de un intervalo de tiempo de entre diez y treinta años para la elaboración de un producto funcional.

Pero no sería la primera vez que un avance tecnológico surge de un día para otro de forma repentina y tenemos que adaptarnos a base de parches, que básicamente, es lo que llevamos haciendo todo este tiempo.

Por lo tanto no hay que perder de vista, ya no tanto el cuándo saldrán computadores cuánticos capaces, sino el hecho de que esta situación puede llegar a ser un gran problema y anticiparnos con los cambios y desarrollos oportunos.

3. Análisis de las propuestas

Tras ponernos en situación, nos podemos llegar a preguntar sobre qué podemos hacer ante esto y cómo lo haríamos. Pues bien, al igual que hubo personas capaces para idear algoritmos capaces de destruir la criptografía clásica, tenemos la suerte de que también las hay que encuentran la forma de volvernos a proteger ante las características de este nuevo escenario.

Alrededor del mundo hay repartidos muchos grupos de investigación dedicados al descubrimiento, desarrollo y avance de problemas matemáticos que puedan ser aplicados a la criptografía post-cuántica. Pero en este trabajo nos vamos a centrar en los candidatos del NIST.

3.1. Post-Quantum Cryptography Project [PQC Project]

El Instituto Nacional de Estándares y Tecnología (National Institute of Standards and Technology, NIST[4]) es una agencia de administración tecnológica que pertenece al Departamento de comercio de los Estados Unidos, y es la encargada de promover la innovación y la competencia industrial en Estados Unidos mediante avances en metrología, normas y tecnología.

Hablando de la parte criptográfica, lleva creando competiciones abiertas bastante tiempo con el fin de encontrar algoritmos criptográficos para ser tomados como estándares en todo el mundo. Por poner algunos ejemplos, de estas competiciones han salido algunos como AES [AES] y SHA-3 [SHA-3].

En 2016 el NIST anunció el proyecto “Post-Quantum Cryptography”, el cual sería un concurso para encontrar un estándar que sea capaz de coexistir con las capas que actualmente tenemos pero que además nos proteja de una muy posible aparición de computadores cuánticos.

[4] NIST: https://www.nist.gov

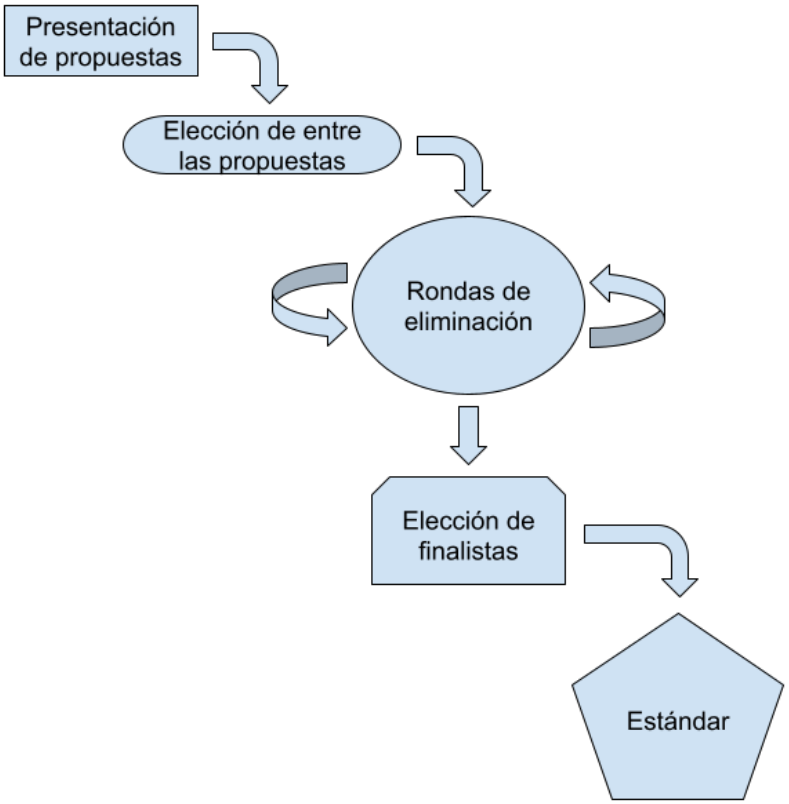
Cuando esto fue anunciado, causó bastante revuelo y a las empresas les entraron un poco las prisas al no saber a lo que se estaban enfrentando y no tener control de ello todavía. Pero los expertos afirmaban que teníamos la suerte de que se había visto el problema a tiempo, y que no era momento de correr, sino de asentar bien todas estas bases y estándares que nos protegerían de los progresos en computación cuántica.

3.1.1. Concurso

Para esta competición había dos categorías:

Categoría	Finalidad
PKE(Public Key Encryption) / KEM(Key Encapsulation Mechanism)	Para cifrado asimétrico e intercambio de claves.
Digital Signature	Para el proceso de firma digital.

Una vez sabiendo las dos categorías se ha seguido un proceso:



3.1.2. Situación Actual

Actualmente nos encontramos en la ronda cuatro pero en la ronda tres ya se eligieron algunos finalistas para su estandarización en las categorías:

Categoría	Algoritmos Seleccionados	Ronda 4
PKE(Public Key Encryption) / KEM(Key Encapsulation Mechanism)	- CRYSTALS-Kyber	- BIKE, - Classic McEliece - HQC - SIKE
Digital Signature	- CRYSTALS-Dilithium - Falcon - SPHINCS+	

Esto significa que los algoritmos seleccionados en la ronda tres ya pueden realizar el último proceso para presentarse como candidatos a estandarización. Esto no significa que ya lo sean, incluso pueden ser eliminados o sustituidos por otros candidatos de otras rondas, o por suplentes.

El NIST ha dicho que si a final de año no hay nada claro, podrían convocar una quinta.

3.1.3. Candidatos

Dado a que actualmente tenemos seleccionados posibles algoritmos que servirán como estándar vamos a verlos un poquito más en detalle cada uno de ellos.

3.1.3.1. Familia CRYSTALS-(Kyber y Dilithium)

La "Cryptographic Suite for Algebraic Lattices" (CRYSTALS) abarca dos primitivas criptográficas: Kyber, un mecanismo de encapsulación de clave (KEM) seguro de Indistinguibilidad bajo el Ataque de texto Cifrado Elegido Adaptativo [IND-CCA2]; y Dilithium, un algoritmo de firma digital Imposibilidad de Falsificar bajo Ataque de Mensaje Elegido [EUF-CMA]. Ambos algoritmos se basan en problemas complejos sobre retículos modulares.

Retículos modulares

Se pueden considerar como retículos que se encuentran entre los que se usan en las definiciones del problema LWE(Aprendizaje Con Error) y los que se usan para el problema Ring-LWE [Ring-LWE]. Si el anillo subyacente al módulo tiene un grado suficientemente alto (como 256), entonces estas retículas heredan toda la eficiencia de las utilizadas en el problema Ring-LWE y, además, tienen las siguientes ventajas cuando se usan en nuestros algoritmos criptográficos:

Las únicas operaciones requeridas para Kyber y Dilithium para todos los niveles de seguridad son variantes de Keccak, sumas/multiplicaciones en Zq para un q fijo y NTT (number theoretic transform) para el anillo $Zq[X]/(X^{256} + 1)$.

Esto significa que aumentar/disminuir el nivel de seguridad prácticamente no implica implementar de nuevo los esquemas software o hardware. Simplemente, cambiando algunos parámetros se consigue convertir una implementación preparada para un cierto nivel de seguridad en otro diferente. Esto da una gran versatilidad.

Los retículos utilizados en Kyber y Dilithium tienen una estructura menos algebraica que las utilizadas para Ring-LWE y están más cerca de ser no estructuradas, como las utilizadas en LWE. Por lo tanto, es concebible que si aparecen ataques algebraicos contra Ring-LWE (no se conoce ninguno actualmente), entonces pueden ser menos efectivos contra esquemas como Kyber y Dilithium.

3.1.3.2. FALCON

FALCON (**FA**st Fourier **L**attice-based **CO**mpact signatures over **N**TRU) se basa en el marco teórico de Gentry, Peikert y Vaikuntanathan para esquemas de firma basados en retículos. Dicho marco se instancia sobre retículos NTRU [NTRU], otro criptosistema de clave pública basado en retículos, con un muestreador de trampilla llamado "fast Fourier sampling". El problema complejo sobre el que se sustenta es el problema de solución de enteros cortos (SIS) sobre retículos NTRU, para el cual actualmente no se conoce ningún algoritmo de resolución eficiente en el caso general, incluso con la ayuda de computadoras cuánticas.

Falcon ofrece las siguientes características:

- Seguridad: internamente se utiliza un muestreo gaussiano, lo que garantiza una fuga despreciable de información sobre la clave secreta hasta un número prácticamente infinito de firmas (más de 264).
- Compacidad: gracias al uso de retículos NTRU, las firmas son sustancialmente más cortas que en cualquier esquema de firma de la misma familia con las mismas garantías de seguridad, mientras que las claves públicas tienen aproximadamente el mismo tamaño.
- Velocidad: el uso del muestreo rápido de Fourier permite implementaciones muy rápidas (miles de firmas por segundo en computadora no cuántica); la verificación es de cinco a diez veces más rápida.
- Escalabilidad: las operaciones tienen un coste $O(n \log n)$ para el grado n, lo que permite utilizar parámetros de seguridad de muy largo plazo a un coste moderado.
- Economía de RAM: el algoritmo mejorado de generación de claves de FALCON utiliza menos de 30 kilobytes de RAM, una mejora cien veces superior a los diseños anteriores, como NTRUSign. FALCON es compatible con dispositivos integrados pequeños con limitaciones de memoria.

3.1.3.3. SPHINCS+

SPHINCS+ es un esquema de firma basado en hash. Esta es la elección conservadora, cuya seguridad no tiene nada que ver con las retículos, sino que se basa en propiedades bastante básicas de las funciones hash, lo que da es plus de entendimiento y de saber que no corren el riesgo de romperse en el futuro cercano (aunque, para ser justos, no sabemos realmente, matemáticamente hablando, si las funciones hash seguras pueden existir en absoluto).

El rendimiento de SPHINCS+ no es tan bueno, como es habitual en los esquemas de firma basados en hash: las claves públicas son muy pequeñas (32 bytes en el nivel de seguridad base), pero las firmas son bastante grandes (al menos 7856 bytes). Cabe señalar que SPHINCS+ es un esquema sin estado; existen otros esquemas estandarizados basados en hash con estado (por ejemplo, XMSS y LMS) que ofrecen firmas algo más pequeñas, pero requieren que el firmante mantenga algún estado que cambie para cada firma producida. En general, estos esquemas basados en hash son adecuados en situaciones como un sistema integrado que verifica una firma criptográfica en su imagen de firmware cada vez que se inicia.

4. Programa de ejemplo

Cómo último punto se redactará como ha sido el proceso de elaboración del caso práctico.

4.1.Investigación

Cuando me puse a buscar las herramientas con las que iba a proceder a realizar dicha prueba me decanté por buscar implementaciones de los algoritmos seleccionados y los que estaban en la cuarta ronda del concurso del NIST, porque supuse que serían los que más maduros a nivel de desarrollo estarían ya que son los que se encuentran en el punto de mira.

Comencé buscando en sus páginas oficiales, accesibles desde el sitio web del mismo concurso, pero, la mayoría presentaban códigos de prueba, o implementaciones tediosas y poco usables si no tienes un conocimiento profundo de los respectivos algoritmos.

Las bibliotecas más accesibles estaban implementadas por terceros, distribuidas por distintos repositorios y en casi todas ponía un aviso como que no estaban seguros de su implementación, que no se usasen para software criptográfico.

Yo buscaba algo que encapsularse varías implementaciones de distintos algoritmo en una misma interfaz homogénea para poder realizar pruebas más compactas y no tener que estar compilando de un sitio y de otro, sino que sigan un mismo estilo de implementación para que las diferencias se encuentren en el algoritmo y no en el desarrollo. Y encontré **libpqcrypto[5].**

[5] libpqcrypto: https://libpqcrypto.org

4.2.libpqcrypto

Esta biblioteca fue implementada por una entidad europea dedicada a la investigación con algoritmos criptográficos post-cuánticos a largo plazo llamada PQCRYPTO ICT-645622[6].

Implementa los siguientes algoritmos:

- BIG QUAKE: crypto_kem_bigquake{1,3,5}
- Classic McEliece: crypto_kem_mceliece{6960119,8192128}
- CRYSTALS-DILITHIUM: crypto_sign_dilithium{2,3,4}
- CRYSTALS-KYBER: crypto_kem_kyber{512,768,1024}
- DAGS: crypto_kem_dags{3,5}
- FrodoKEM: crypto_kem_frodokem{640,976}
- Gui: crypto_sign_gui{184,312,448}
- KINDI: crypto_kem_kindi{256342,256522,512222,512241,512321}
- LUOV: crypto_sign_luov{863256,890351,8117404,4849242,6468330,8086399}
- MQDSS: crypto_sign_mqdss{48,64}
- NewHope: crypto_kem_newhope{512,1024}cca
- NTRU-HRSS-KEM: crypto_kem_ntruhrss701
- NTRU Prime: crypto_kem_{ntrulpr,sntrup}4591761
- Picnic: crypto_sign_picnicl{1,3,5}{fs,ur}
- qTESLA: crypto_sign_qtesla{128,192,256}
- Rainbow: crypto_sign_rainbow{1a,1b,1c,3b,3c,4a,5c,6a,6b}
- Ramstake: crypto_kem_ramstakers{216091,756839}
- SABER: crypto_kem_{firesaber,lightsaber,saber}
- SPHINCS+: crypto_sign_sphincs{f,s}{128,192,256}{haraka,sha256,shake256}

En un principio me gustó bastante porque, aparte de implementar bastantes algoritmos tanto de clave pública como de firma digital presentaba una interfaz en Python, lenguaje con el que estoy bastante familiarizado.

[6] PQCRYPTO ICT-645622: https://pqcrypto.eu.org

Esta biblioteca presentaba 3 interfaces distintas para el uso de todos los algoritmos que hemos visto anteriormente:

- Línea de Comandos (CLI)
- Python API
- C API

4.3.Preparación de entorno de pruebas

Como he dicho anteriormente, me siento cómo con Python, por lo que en un principio opté por usar dicha interfaz.

4.3.1. Instalación

Otra de las cosas por las que elegí esta biblioteca fue que las instrucciones de instalación parecían bastante sencillas, así que comencé a realizar el proceso sobre mi ordenador personal con las siguientes características:

Odenador Personal	
Sistema Operativo	Ubuntu 18.04
Hardware	- Procesador: Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz - RAM: 16Gb - Almacenamiento: CT500MX500SSD1

4.3.1.1. Prerrequisitos

- Sistema operativo Debian/Ubuntu
- gcc and other compiler tools: apt install build-essential
- OpenSSL header files: apt install libssl-dev
- GMP header files: apt install libgmp-dev
- Python 3: apt install python3

4.3.1.2. Pasos a seguir

Una vez cumplimos con los requisitos procedí a la instalación con:

1. Abrimos un terminal root y creamos el usuario “libpqcrypto”:

```
TERMINAL: sudo -i
          adduser --disabled-password --gecos libpqcrypto libpqcrypto
```

2. Abrimos el terminal con ese usuario:

```
TERMINAL: su - libpqcrypto
```

3. Descargamos y descomprimimos la última versión de “libpqcrypto”

```
TERMINAL: wget -m https://libpqcrypto.org/libpqcrypto-latest-version.txt
          version=$(cat libpqcrypto.org/libpqcrypto-latest-version.txt)
          wget -m https://libpqcrypto.org/libpqcrypto-$version.tar.gz
          tar -xzf libpqcrypto.org/libpqcrypto-$version.tar.gz
          cd libpqcrypto-$version
          ln -s $HOME link-build
          ln -s $HOME link-install
```

4. Compilamos, realizamos los test e instalamos:

```
TERMINAL: ./do
```

Llegados a este punto a mi me dió un error mientras ejecutaba los test en el que me indicaba que no tenía “clang” instalado, por lo que tuve que hacer un paso adicional que no está indicado en las instrucciones:

```
TERMINAL: apt install clang
```

Tras esto volví a compilar y ejecutar los tests, los cuales ahora ya sí funcionaron.

En la plataforma, indican que este cuarto paso puede tardar un tiempo, lo que me hizo pensar que tardaría unos minutos o como mucho media hora, pero no. El proceso entero de compilación, realización de test e instalación tardó más de tres horas, incluso había veces que el proceso se había parado o estaba congelado. Finalmente y tras esperar semejante cantidad de tiempo la librería estaba instalada.

4.4.Prueba con Interfaz de Python

Antes de empezar a usarla se tenía que hacer una configuración previa indicada en el sitio web

4.4.1. Configuración de Interfaz Python

Para poder usar esta herramienta te indican que tienes que exportar la ruta de Python donde está instalada la librería, cosa que tenías que hacer cada vez que iniciabas el ordenador:

```
TERMINAL: export
PYTHONPATH="/home/libpqcrypto/python${PYTHONPATH+:$PYTHONPATH}"
```

Una vez teniendo ya todo listo creé mi proyecto en PyCharm para empezar a realizar las pruebas, el cual genera un entorno virtual para cada proyecto. Intenté importar la librería para empezar a utilizarla pero no la reconocía. Investigando por la web encontré que tenía que importar la biblioteca como recurso raíz al proyecto en el que estaba trabajando.

Ese fue el primer fallo, pero el segundo y con el que decidí no realizar el programa con la interfaz de Python fue que cuando quise hacer procesos básicos de estos algoritmos como cifrados, no encontraba ninguna función que me lo permitiera, y no solo eso, sino que no hay ningún tipo de documentación de las llamadas que implementa (tuve que mirarlas en el código).

Por los problemas anteriormente mencionados dejé de lado esta opción.

4.5.Prueba con Interfaz de Línea de Comandos

Echando un vistazo sobre esta otra herramienta vi, en el código de ejemplo, que si estaban implementadas las funciones que yo necesitaba para hacer las pruebas que quería por lo que realicé la configuración necesaria, muy parecida a la de Python.

4.5.1. Configuración de Interfaz de Línea de Comandos

```
TERMINAL: export PATH=$PATH:/home/libpqcrypto/command
```

Esta también era necesaria realizarla en cada inicio.

Realicé unas pruebas iniciales y al ver que dichas funciones que necesitaría para las pruebas eran satisfactorias comencé el caso práctico.

4.6.Caso Práctico

Para esta prueba de concepto se han elegido a los algoritmos de la familia CRYSTALS:

Características	
Algoritmos	- CRYSTALS-Kyber - CRYSTALS-Dilithium

Y para cada categoría del concurso realizaremos las siguientes acciones:

- **PKE/KEM**
  - Generación de Claves
  - Cifrado y Descifrado
- **Digital Signature**
  - Generación de Claves
  - Firma y Verificación

4.6.1. CRYSTALS-Kyber (PKE/KEM)

1. **Creación del mensaje**

Se creará un fichero de texto con el mensaje:

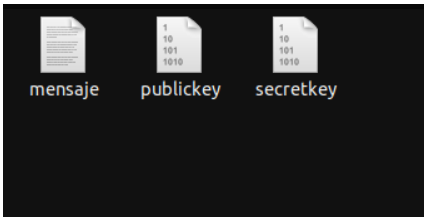
- “Vamos a cifrar y descifrar este mensaje.”

```
TERMINAL: echo "Vamos a cifrar y descifrar este mensaje." > mensaje
```

2. **Generación de Claves**

Generamos un par de claves de CRYSTALS-Kayber 1024 al que le pasamos como parámetros de salida el nombre del fichero que será la clave pública(`publickey`) y la privada(`secretkey`).

```
TERMINAL: pq-keypair-kyber1024 5>publickey 9>secretkey
```

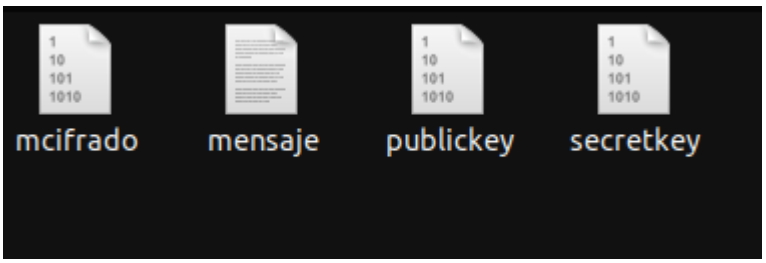


3. **Obtener mensaje cifrado**

Para el cifrado, se necesita el fichero a cifrar(`mensaje`) y la clave pública(`publickey`) como entrada.

Como salida se dará el nombre al fichero en el que estará el mensaje cifrado (`mcifrado`).

```
TERMINAL: pq-encrypt-kyber1024 <mensaje 4<publickey >mcifrado
```



Al mostrar el contenido del mensaje cifrado veremos que ya no será legible.

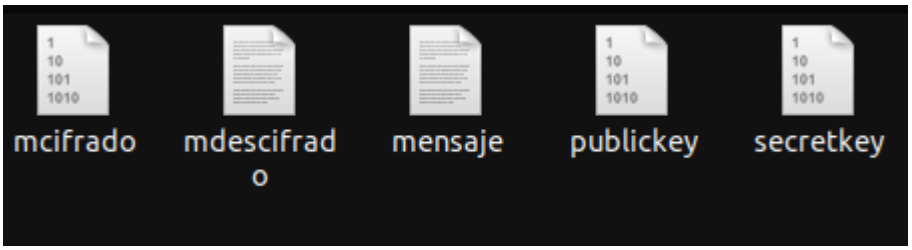
```
TERMINAL: cat mensaje
Vamos a cifrar y descifrar este mensaje.

TERMINAL: cat mcifrado
j}v<3◆+mTf_X:_S§3A5MY`D◆F◆kJPT`^1b6A◆◆^F#FwJ+44¹◆(x◆*-◆◆:◆\y◆5◆FT`"t,]1|◆mdbh◆
<◆◆◆Z/T◆1B◆◆ ᴋMa_ ◆◆JsXD◆sv◆&Y%◆$Ģ◆@◆g=V◆◆H p◆◆.'◆◆s◆Z{lq◆#◆7◆}RyW M/N}◆◆]
◆ʃX- <^j@l+◆…
```

4. **Descifrar mensaje**

Ahora, al descifrar, necesitamos pasar como parámetros de entrada, el fichero a descifrar(`mcifrado`) y la clave privada(`secretkey`). No arrojará un fichero de salida descifrado (`mdescifrado`).

```
TERMINAL: pq-decrypt-kyber1024 <mcifrado 8<secretkey >mdescifrado
```



Se puede observar que efectivamente, el mensaje es como el original.

```
TERMINAL: cat mdescifrado
Vamos a cifrar y descifrar este mensaje.
```

Resultados	Promedio de ns en mil ejecuciones	Tamaño
Creación del mensaje	N/A	- mensaje: 41 Bytes
Generación de Claves	2609708 ns	- publickey: 1440 Bytes - secretkey: 3168 Bytes
Cifrado	2548854 ns	- mcifrado: 1561 Bytes
Descifrado	2503110 ns	- mdescifrado: 41 Bytes

4.6.2. CRYSTALS-Dilithium (Digital Signature)

1. **Creación del mensaje**

Se creará un fichero de texto con el mensaje:

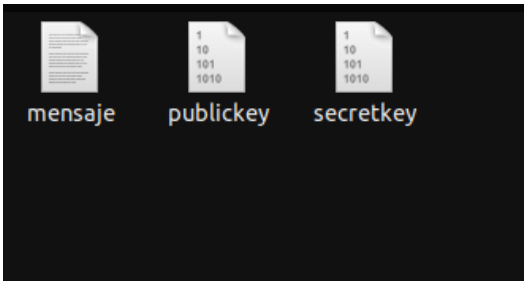
- "Vamos a firmar y verificar este mensaje."

TERMINAL: `echo "Vamos a firmar y verificar este mensaje." > mensaje`

2. **Generación de Claves**

Generamos un par de claves de CRYSTALS-Dilithium 4 al que le pasamos como parámetros de salida el nombre del fichero que será la clave pública(`publickey`) y la privada(`secretkey`).

TERMINAL: `pq-keypair-dilithium4 5>publickey 9>secretkey`



3. **Firma**

Para firmar, se necesita el fichero a firmar(`mensaje`) y la clave pública(`publickey`) como entrada.

Como salida se dará el nombre al fichero en el que estará el mensaje con la firma.

TERMINAL: `pq-sign-dilithium4 <mensaje 8<secretkey >mfirmado`

Se puede ver cómo está la firma, y seguidamente el mensaje.

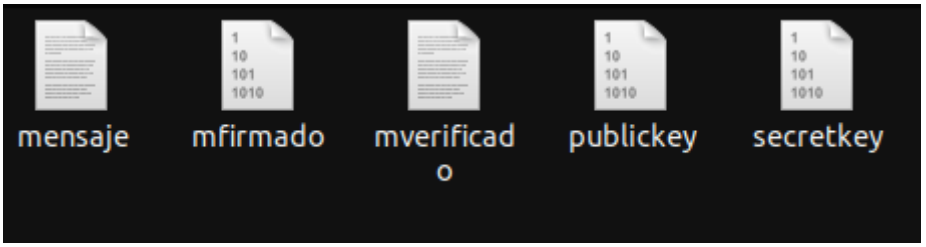
```
TERMINAL: cat mfirmado

…Q(◆◆◆◆◆◆R◆1`[.◆B.◆◆G$y)`◆!◆◆l◆f◆=σL◆F~%Y◆ -1367=qrt◆EPUVimo◆◆
4MPu}◆◆'*-28:;JUd◆◆ "◆◆;k◆$3GQXh $    P,A! sY_@"QVamos a firmar y verificar
este mensaje.
```

4. **Verificación**

Por último, para verificar la firma se necesita pasar como parámetros de entrada, el fichero a descifrar(`mfirmado`) y la clave privada(`secretkey`). No arrojará un fichero de salida descifrado (`mverificado`).

```
TERMINAL: pq-open-dilithium4 <mfirmado 4<publickey >mverificado
```



El mensaje vuelve a aparecer sin la firma.

```
TERMINAL: cat mverificado

Vamos a firmar y verificar este mensaje.
```

Resultados	Promedio de ns en mil ejecuciones	Tamaño
Creación del mensaje	N/A	- mensaje: 41 Bytes
Generación de Claves	2823847 ns	- publickey: 1760 Bytes - secretkey: 3856 Bytes
Cifrado	3037447 ns	- mfirmado: 3407 Bytes
Descifrado	2806884 ns	- mverificado: 41 Bytes

5. Conclusión

Una vez hecho este repaso sobre el estado de la Criptografía Post-Cuántica y volviendo a los objetivos de este trabajo he podido confirmar que:

- ☑ ~~Realizar una revisión del estado del arte en Criptografía post-cuántica.~~

Estamos muy próximos a la incorporación de ordenadores cuánticos a nuestras vidas, ya sea por nuestra parte o por la de otros; por lo que si queremos seguir haciendo uso de todos los sistemas de información que utilizamos a diario debemos poner atención y prevenir de la mejor forma posible un cambio tecnológico tan grande.

- ☑ ~~Realizar un breve análisis de por qué la computación cuántica permitirá romper algunos de los algoritmos de cifrado actuales.~~

Es evidente que estamos ante un problema inminente que afecta a demasiadas cosas de forma global como para no prestar mucha atención. Si bien es cierto que quedan años para que esto sea un problema real, no podemos relajarnos y hay que hacerle un seguimiento exhaustivo para estar lo mejor preparados posibles.

- ☑ ~~Evaluar alguna de las propuestas actuales en el campo de la Criptografía post-cuántica, incluyendo sus fundamentos teóricos.~~

También he comprendido que la certificación o estandarización de algoritmos criptográficos es un proceso tan meticuloso como necesario y que necesita de su tiempo y esfuerzo. He podido ver como en el concurso del NIST llevan años haciendo revisiones sobre complejos sistemas criptográficos, y aún a día de hoy, habiendo pasado ya 6 años desde que empezó el proyecto surgen ataques que destronan a candidatos que se encontraban en la 4 ronda del concurso y que ya habían pasado sobre muchas revisiones como es el caso de SIKE.

- ☑ ~~Implementar una aplicación capaz de usar alguna técnica criptográfica post-cuántica.~~

Con el programa de prueba me he dado cuenta que todavía no hay demasiados recursos como para que todo el mundo comience a realizar los cambios preparatorios necesarios. Esto es algo que tanto las grandes entidades, tanto públicas como privadas, como las diferentes comunidades, deben de tomar parte. Porque si tenemos todos los algoritmos, estándares y herramientas necesarias pero los cambios no se desarrollan correctamente, todo caerá en saco roto.

Para finalizar me gustaría comentar una inquietud que me surge sobre el rendimiento.
Tenemos la suerte de haber detectado el problema a tiempo e incluso seguramente, estaremos en disposición de usar los nuevos estándares post-cuánticos antes de la llegada de dichos computadores. Pero que éstos sean funcionales no implica que demos un salto tecnológico de manera global, puesto que a pesar de no poder hacer uso de ellos, si vamos a tener que estar protegidos, lo que implica que habrá una disminución del rendimiento en la comunicación y manejo de información segura al estar tratando con algoritmos más pesados en procesamiento y memoria.

Bibliografía

Velthuis, M. P. (2021, 3 abril). Computación cuántica: un salto tan grande como el que hubo entre el ábaco y la informática actual. El País. Recuperado 8 de septiembre de 2022, de https://elpais.com/tecnologia/2021-04-03/computacion-cuantica-un-salto-tan-grande-como-el-que-hubo-entre-el-abaco-y-la-informatica-actual.html

Molina, A. (2020, 11 noviembre). La computación cuántica no matará a la criptografía. Santander Global Tech. Recuperado 8 de septiembre de 2022, de https://santandergto.com/la-computacion-cuantica-no-matara-a-la-criptografia/#anexos

[Eagle (IBM)]: Pastor, J. (2021, 15 noviembre). *IBM presenta Eagle, su procesador cuántico de 127 qubits: "es imposible simularlo con cualquier otra. . .* Xataka. Recuperado 8 de septiembre de 2022, de https://www.xataka.com/investigacion/ibm-presenta-eagle-su-procesador-cuantico-127-qubits-imposible-simularlo-cualquier-otra-cosa

[Borealis (Xanadu Quantum Technologies)]: Borealis. (s. f.). Xanadu. Recuperado 8 de septiembre de 2022, de https://www.xanadu.ai/products/borealis/

La ley de Moore llegará a su fin en 2021, según admiten los fabricantes de chips. (2017b, agosto 17). MIT Technology Review. Recuperado 8 de septiembre de 2022, de https://www.technologyreview.es/s/6128/la-ley-de-moore-llegara-su-fin-en-2021-segun-admiten-los-fabricantes-de-chips

[PQC Project]: Post-Quantum Cryptography | CSRC. (s. f.). Recuperado 8 de septiembre de 2022, de https://csrc.nist.gov/projects/post-quantum-cryptography

Pornin, T., & Pornin, V. A. P. B. T. (2022, 13 julio). NIST Selects Post-Quantum Algorithms for Standardization. NCC Group Research. Recuperado 8 de septiembre de 2022, de https://research.nccgroup.com/2022/07/13/nist-selects-post-quantum-algorithms-for-standardization/

[AES]: A. Hamza and B. Kumar, "A Review Paper on DES, AES, RSA Encryption Standards," *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*, 2020, pp. 333-338, doi: 10.1109/SMART50582.2020.9336800

[SHA-3]: Dworkin, M. (2015), SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, Federal Inf. Process. Stds. (NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD, [online], https://doi.org/10.6028/NIST.FIPS.202

[Shor94]: P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," Proceedings 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 124-134, doi: 10.1109/SFCS.1994.365700.

[Grover]: Lov K. Grover. 1996. A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing

(STOC '96). Association for Computing Machinery, New York, NY, USA, 212–219. https://doi.org/10.1145/237814.237866

[NTRU]: Blum, M., & Goldwasser, S. (1984). An Efficient Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information. *CRYPTO*.

[Ring-LWE]: Lyubashevsky, V., Peikert, C., & Regev, O. (2013). On Ideal Lattices and Learning with Errors over Rings. *JACM*.

[IND-CCA2]: Indistinguibilidad del texto cifrado. (s. f.). hmn.wiki. Recuperado 8 de septiembre de 2022, de https://hmn.wiki/es/IND-CCA2

[EUF-CMA]: EUF-CMA and SUF-CMA. (2018, 6 abril). A Few Thoughts on Cryptographic Engineering. Recuperado 8 de septiembre de 2022, de https://blog.cryptographyengineering.com/euf-cma-and-suf-cma

libpqcrypto: Intro. (s. f.). Recuperado 8 de septiembre de 2022, de https://libpqcrypto.org/