UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y DISEÑO INDUSTRIAL

Grado en Ingeniería Electrónica Industrial y Automática

TRABAJO FIN DE GRADO

Estudio comparativo de diferentes algoritmos de encriptación para comunicaciones industriales

Bogurad Barañski Barañska

Cotutor: Basil Mohammed Al-Hadithi
Departamento: ingeniería eléctrica, electrónica, automática y física aplicada.

Tutor: Roberto Gonzalez Herranz
Departamento: ingeniería eléctrica, electrónica, automática y física aplicada.

Madrid, Septiembre, 2025

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y DISEÑO INDUSTRIAL

Grado en Ingeniería Electrónica Industrial y Automática

TRABAJO FIN DE GRADO

Título del trabajo

Firma Autor

Firma Tutor

Título: Estudio comparativo de diferentes algoritmos de encriptación para comunicaciones industriales
Autor: Bogurad Barañski Barañska
Tutor: Roberto Gonzalez Herranz

EL TRIBUNAL

Presidente:

Vocal:

Secretario:

Realizado el acto de defensa y lectura del Trabajo Fin de Grado el día ..... de ..... de ... en ....., en la Escuela Técnica Superior de Ingeniería y Diseño Industrial de la Universidad Politécnica de Madrid, acuerda otorgarle la CALIFICACIÓN de:

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

Agradezco a .....

Resumen

Este proyecto se resume en.....

Palabras clave: palabraclave1, palabraclave2, palabraclave3.

Abstract

In this project...

Keywords: keyword1, keyword2, keyword3.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación del proyecto

Cuando en una instalación industrial se actúa o se mide un proceso, el autómata que envía las señales puede estar situado a gran distancia de dicho proceso. Por esta razón, las comunicaciones industriales precisan el uso de buses de longitudes considerables o realizar comunicaciones a distancia.

Aunque las comunicaciones a distancia pudieran parecer una solución más económica de implementar, tienen el problema de ser vulnerables a ataques de intermediario (alguien ajeno al proceso intercepta los mensajes enviados), lo cual pone en peligro la confidencialidad de la información. Por esta misma razón, en este trabajo se estudiarán distintos algoritmos propuestos para la encriptación de los mensajes.

1.2. Objetivos

Para realizar el proyecto, se proponen los siguientes objetivos:

- Implementar los siguientes algoritmos en C/C++
  - RSA
  - Curvas elípticas
  - AES 256
  - Celosías
  - Algoritmo de Shore
  - Algoritmos post-cuánticos
- Estudiar la eficacia de cifrado
  - Estudiar velocidad de ejecución del algoritmo
  - Estudiar recursos requeridos por el microprocesador
  - Estudiar la robustez del cifrado
  - Estudiar la posibilidad de ataques de canal lateral
- Posibilidad de ejecución en sistemas basados en FPGAs

- Qué y cómo medir en las ECC Intercambio de claves pública privado mediante RSA/ leif-haunman
  - Capacidad de memoria
  - Tiempo de CPU
  - Estudio de entropía

1.3. Herramientas utilizadas

1.3.1. LaTex [1]

Se ha preferido el uso de LaTeX debido a la facilidad que ofrece para el maquetado de textos, superando a otras herramientas de elaboración de documentos. Además, LaTeX permite crear figuras vectorizadas, representar correctamente ecuaciones y ubicar adecuadamente figuras, tablas y bibliografía.

1.3.2. TikzMaker [2]

Esta herramienta permite crear figuras vectorizadas de LaTeX mediante el paquete de circuitikz. Su principal ventaja radica en la interfaz gráfica que proporciona y en la facilidad para elaborar figuras.

1.3.3. C [3] y C++ [4]

1.3.4. Microprocesador CY8CPROTO-063-BLE [5]

1.4. Estructura del documento

A continuación y para facilitar la lectura del documento, se detalla el contenido de cada capítulo:

- En el capítulo 1 se realiza una introducción.
- En el capítulo 2 se hace un repaso de desarrollos anteriores .
- En el capítulo 3 se desarrollan los fundamentos matemáticos del proyecto.
- En el capítulo 4 se describe la implementación de los algoritmos.
- En el capítulo 5 se exponen los resultados obtenidos en el capítulo anterior.
- En el capítulo 6 se comparan los resultados de los distintos algoritmos.

Capítulo 2

Estado del arte

Capítulo 3

Fundamentos generales

En este capítulo se desarrollan las bases matemáticas de los distintos algoritmos a implementar.

3.1. Introducción

3.2. Algoritmos de Hashing y Funciones de Salida Extendida[6]

3.3. Métodos clásicos de cifrado asimétrico

3.3.1. RSA

3.3.2. ECC

3.3.3. Algoritmo de Shore

3.4. Funcionamiento básico de los algoritmos postcuánticos

En esta sección se describe el funcionamiento de los algoritmos postcuánticos analizados en este trabajo. Dado que no se desarrollaron implementaciones propias, sino que se utilizó el código proporcionado por el NIST en la tercera [7] y cuarta [8] ronda del proceso de estandarización, resulta apropiado presentar su funcionamiento aquí en lugar de en la sección de desarrollo.

3.4.1. CRYSTALS-Kyber

Se utilizará la misma notación empleada en el artículo [9]. El conjunto de los enteros sin signo de 8 bits se denota por $\mathcal{B} = \{0, \ldots, 255\}$. Para representar vectores de tamaño k, se utiliza la notación $\mathcal{B}^k$, mientras que para vectores de tamaño arbitrario se emplea $\mathcal{B}^*$.

Para trabajar con estos vectores, se utiliza el símbolo $||$ para denotar la concatenación de dos cadenas, y la notación $+k$ para indicar el desplazamiento de k bytes desde el inicio de una cadena. Por ejemplo, si se tiene una cadena a de longitud l y se concatena con una cadena b, se obtiene:

$$c = a||b \tag{3.1}$$

Entonces:

$$b = c + l \tag{3.2}$$

Para denotar vectores, se utiliza la notación $v[i]$, donde v es un vector columna e i indica la posición del elemento (empezando desde 0, si no se indica lo contrario). Para las matrices, se emplea la notación $A[i][j]$, donde i representa la fila y j la columna. La transpuesta de una matriz A se denota como A^T.

Se denota mediante $\lfloor x \rceil$ el redondeo de x al entero más cercano. Por ejemplo: $\lfloor 2{,}3 \rceil = 2$, $\lfloor 2{,}5 \rceil = 3$ y $\lfloor 2{,}8 \rceil = 3$.

Para las reducciones modulares se emplean dos tipos: una centrada en cero y otra correspondiente a la reducción modular estándar. Para la reducción modular centrada en cero, sea α un entero par. Esta operación se define como:

$$r' = r \bmod^{\pm} \alpha \quad \Longrightarrow \quad -\frac{\alpha}{2} < r' \leq \frac{\alpha}{2} \tag{3.3}$$

Mientras que la reducción modular estándar se denota como:

$$r' = r \bmod^{+} \alpha \quad \Longrightarrow \quad 0 \leq r' < \alpha \tag{3.4}$$

Finalmente, se denota mediante $s \leftarrow S$ la selección de s de manera uniformemente aleatoria del conjunto S. Si S representa una distribución de probabilidad, entonces s se selecciona de acuerdo con dicha distribución.

3.4.1.1. Transformada Teórica de Números o Number Theoretic Transform (NTT)

Para acelerar las operaciones de multiplicación en el esquema basado en retículas, se utiliza la Transformada Teórica de Números (NTT, por sus siglas en inglés), la cual permite reducir la complejidad temporal de la multiplicación de polinomios desde $\mathcal{O}(n^2)$, correspondiente al método tradicional, hasta $\mathcal{O}(n\log(n))$. Para más detalles, consúltese [10].

Antes de pasar a explicar el funcionamiento de este método es relevante aclarar que se trabaja sobre el siguiente anillo de polinomios para realizar las operaciones, denotado mediante R_q:

$$R_q := \frac{\mathbb{Z}_q[X]}{X^n + 1} \tag{3.5}$$

En la implementación especificada de Kyber, según el artículo [9], se utiliza un valor de $q = 3329$ y $n = 256$. Esta elección es esencial para permitir el uso de la multiplicación mediante la Transformada Teórica de Números (NTT), la cual requiere que $n|(q-1)$, es decir, que n divida a $(q-1)$. Esta condición garantiza la existencia de n raíces enésimas de la unidad en Z_q, lo cual es necesario para definir la NTT. La validez de esta afirmación se fundamenta en el siguiente teorema [11]:

Teorema 1 *Para n, $q > 1$, el cuerpo Z_q tiene una raíz enésima de la unidad si y solo sí $n|(q-1)$*

Demostración 1 *Si ω es una raíz enésima de la unidad en el conjunto $\mathbb{Z}_q$, entonces el conjunto:*

$$\Omega = \left\{1, \omega, \omega^2, \ldots, \omega^{n-1}\right\} \tag{3.6}$$

forma un subgrupo cíclico H del grupo multiplicativo G_{q-1}. Por el Teorema de Lagrange, se concluye que el orden de H divide al orden de G_{q-1}, es decir, $n \mid (q-1)$.

Dado que G_{q-1} es también un grupo cíclico, existe un generador α tal que, por el pequeño teorema de Fermat, se cumple:

$$\alpha^{q-1} = 1 \tag{3.7}$$

Por lo tanto, el grupo G_{q-1} puede escribirse como:

$$G_{q-1} = \left\{1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\right\} \tag{3.8}$$

Si se define ω como:

$$\omega = \alpha^{\frac{q-1}{n}}, \tag{3.9}$$

entonces:

$$\omega^n = \alpha^{q-1} = 1, \tag{3.10}$$

y además, para todo $0 < k < n$, se cumple:

$$k \cdot \frac{q-1}{n} < q-1 \quad \Rightarrow \quad \omega^k \neq 1. \tag{3.11}$$

Por lo tanto, ω es una raíz enésima de la unidad en $\mathbb{Z}_q$.

Por tanto, el polinomio $X^{256}+1$ se puede factorizar sobre el cuerpo $\mathbb{Z}_q$. En la implementación concreta del esquema Kyber, este polinomio se descompone en 128 factores cuadráticos.

A este polinomio se le aplica la transformada NTT a sus coeficientes, la cual no es más que una variación de la Transformada Discreta de Fourier (DFT) aplicada a cuerpos finitos $\mathbb{Z}_q$ y aplicada a polinomios de grado n. Para ello, se definen dos operaciones fundamentales:

1. La transformada directa:

$$\hat{a}_j = \sum_{i=0}^{n-1} \phi^{i(2j+1)} a_j \bmod q \tag{3.12}$$

2. La transformada inversa:

$$a_i = n^{-1} \sum_{j=0}^{n-1} \phi^{-i(2j+1)} \hat{a}_j \bmod q \tag{3.13}$$

Donde ϕ es un valor tal que $\phi^2 = \omega$, con ω una raíz enésima de la unidad en $\mathbb{Z}_q$ y n^{-1} es la inversa multiplicativa de n en $\mathbb{Z}_q$.

A continuación, se presenta un ejemplo extraído de [10] para ilustrar su funcionamiento. Sea el polinomio $G(x) = 5 + 6x + 7x^2 + 8x^3$, cuyo vector de coeficientes es $g = [5, 6, 7, 8]$. Trabajando en el anillo $\mathbb{Z}_{7681}$, y tomando $\phi = 1925$, se puede calcular la transformada NTT $\hat{g}$. Aplicando luego la transformada inversa, es posible recuperar el vector original g.

$$\hat{g} = \begin{bmatrix} \phi^0 & \phi^1 & \phi^2 & \phi^3 \\ \phi^0 & \phi^3 & \phi^6 & \phi^1 \\ \phi^0 & \phi^5 & \phi^2 & \phi^7 \\ \phi^0 & \phi^7 & \phi^6 & \phi^5 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 1 & 1925 & 3383 & 6468 \\ 1 & 6468 & 4298 & 1925 \\ 1 & 5756 & 3383 & 1213 \\ 1 & 1213 & 4298 & 5756 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 2489 \\ 7489 \\ 6478 \\ 6607 \end{bmatrix} \tag{3.14}$$

Aplicando la transformada inversa, donde la inversa de $\phi = 1925$ en $\mathbb{Z}_{7681}$ es $\phi^{-1} = 1213$ y el inverso del orden del polinomio $n = 4$ es $n^{-1} = 5761$:

$$g = n^{-1} \begin{bmatrix} \phi^0 & \phi^0 & \phi^0 & \phi^0 \\ \phi^{-1} & \phi^{-3} & \phi^{-5} & \phi^{-7} \\ \phi^{-2} & \phi^{-6} & \phi^{-2} & \phi^{-6} \\ \phi^{-3} & \phi^{-1} & \phi^{-7} & \phi^{-5} \end{bmatrix} \cdot \hat{g} = 5761 \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1213 & 5756 & 6468 & 1925 \\ 4298 & 3383 & 4298 & 3383 \\ 5756 & 1213 & 1925 & 6468 \end{bmatrix} \cdot \begin{bmatrix} 2489 \\ 7489 \\ 6478 \\ 6607 \end{bmatrix} \tag{3.15}$$

Con estas transformadas definidas se define el producto o convolución negativa en el anillo $R_q \coloneqq \dfrac{\mathbb{Z}_q[X]}{X^n+1}$ entre dos polinomios g y h como:

$$g \cdot h = \mathrm{NTT}^{-1}(\mathrm{NTT}(g) \circ \mathrm{NTT}(h)) \tag{3.16}$$

Donde la operación $\circ$ denota la multiplicación elemento a elemento (o punto a punto) entre los vectores en $Z_q[X]$.

Ahora, queda demostrar que el tiempo de ejecución de la NTT es de $\mathcal{O}(n\log(n))$. Para lograr este cometido, se aprovechan dos propiedades que cumplen estas raíces calculadas:

1. Peridicidad:
$$\phi^{k+2n} = \phi^k \tag{3.17}$$
2. Simetría:
$$\phi^{k+n} = \phi^{-k} \tag{3.18}$$

Con estas propiedades, se puede implementar el algoritmo de Cooley-Tukey [12] que consiste en ir descomponiendo el problema en mitades de manera recursiva para reducir al máximo la cantidad de cálculos realizados. Partiendo de la transformada directa y desarrollando:

$$\hat{a}_j = \sum_{i=0}^{n-1} \phi^{i(2j+1)} a_j \bmod q = \sum_{i=0}^{n/2-1} \phi^{4ij+2i} a_{2i} + \phi^{2j+1} \sum_{i=0}^{n/2-1} \phi^{4ij+2i} a_{2i+1} \bmod q \tag{3.19}$$

Si se sustituye $A_j = \sum_{i=0}^{n/2-1} \phi^{4ij+2i} a_{2i}$ y $B_j = \sum_{i=0}^{n/2-1} \phi^{4ij+2i} a_{2i+1}$. Ahora aplicando simetría en ϕ:

$$\hat{a}_j = A_j + \phi^{4ij+2i} B_j \bmod q \tag{3.20}$$
$$\hat{a}_{j+n/2} = A_j - \phi^{4ij+2i} B_j \bmod q \tag{3.21}$$

Donde las matrices A_j y $B_j$pueden obtenerse como el resultado de aplicar la NTT sobre la mitad de los puntos, gracias a la estructura recursiva del algoritmo. Esto implica que, si n es una potencia de 2, el proceso puede repetirse recursivamente sobre subproblemas de tamaño cada vez menor, hasta alcanzar el caso base.

De manera similar, se puede mostrar esta propiedad para la transformada inversa. Por tanto, se demuestra que este algoritmo tiene complejidad $\mathcal{O}(n\log(n))$.

En el caso concreto de kyber [9], la Transformada (NTT) de un polinomio en el anillo R_q se representa como un vector de 128 polinomios de grado 1.

Sean las 256 raíces enésimas de la unidad $\{\xi, \xi^3, \ldots, \xi^{255}\}$, con $\xi = 17$ como primera raíz primitiva. Entonces, se cumple que:

$$X^{256} + 1 = \prod_{i=0}^{127} \left(X^2 - \xi^{2i+1}\right) \tag{3.22}$$

Aplicando la NTT propuesta en [9] a un polinomio $f \in R_q$ se obtiene:

$$NTT(f) = \hat{f} = \left(\hat{f}_0 + \hat{f}_1 X, \hat{f}_2 + \hat{f}_3 X, ..., \hat{f}_{254} + \hat{f}_{255} X\right) \tag{3.23}$$

Con

$$\begin{aligned} \hat{f}_{2i} &= \sum_{j=0}^{127} f_{2j} \xi^{(2i+1)j} \\ \hat{f}_{2i+1} &= \sum_{j=0}^{127} f_{2j+1} \xi^{(2i+1)j} \end{aligned} \tag{3.24}$$

Donde mediante la transformada directa NTT e inversa NTT^{-1} se puede realizar el producto de f,$g \in R_q$ de manera eficiente de la siguiente manera:

$$h = f \cdot g = \text{NTT}^{-1}\left[\text{NTT}(f) \circ \text{NTT}(g)\right] \tag{3.25}$$

Siendo $\hat{h} = \hat{f} \circ \hat{g} = \text{NTT}(f) \circ \text{NTT}(g)$ la multiplicación base definida como:

$$\hat{h}_{2i} + \hat{h}_{2i+1}X = \left(\hat{f}_{2i} + \hat{f}_{2i+1}\right)\left(\hat{g}_{2i} + \hat{g}_{2i+1}\right) \mod \left(X^2 - \xi^{2i+1}\right) \tag{3.26}$$

En la figura 3.4.1.1 se puede ver una representación gráfica de como funciona este proceso.

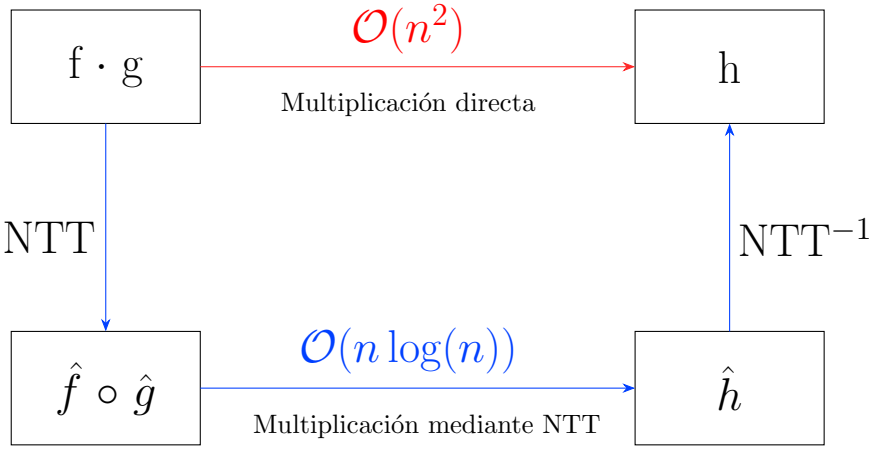

Figura 3.1: Representación del cálculo de un producto de polinomios mediante NTT en comparación con los algoritmos clásicos.

3.4.1.2. Aprendizaje Con Errores o Learning With Errors (LWE)

[13]

3.4.2. Transformadas Fujisaki-Okamoto

[14]

3.4.2.1. Algoritmos principales

3.4.3. SABER

Para [15]

3.4.4. Hamming Quasi-Cyclic (HQC)

Para [16]

3.4.5. Bit Flipping Key Encapsulation (Bike)

Para [17]

3.5. Fundamentos de seguridad de los algoritmos

3.5.1. CRYSTALS-Kyber

Para [9]

3.5.2. SABER

Para [15]

3.5.3. Hamming Quasi-Cyclic (HQC)

Para [16]

3.5.4. Bit Flipping Key Encapsulation (Bike)

Para [17]

Capítulo 4

Desarrollo

4.1. Implementación comunicación serie

4.1.1. Parámetros generales y formato mensajes

4.1.2. Implementación en el ordenador

4.1.3. Implementación en el microprocesador

4.2. Implementación algoritmos de cifrado asimétrico

4.2.1. Kyber

4.2.2. Saber

4.2.3. Bike

4.2.4. HQC

4.3. Implementación del intercambio de claves. Creación del secreto compartido

Capítulo 5

Resultados y discusión

En este capítulo se muestran los resultados obtenidos de aplicar las rutinas desarrolladas con anterioridad.

5.1. Resultados

5.2. Discusión

Capítulo 6

Conclusiones

Se presentan a continuación las conclusiones del proyecto y desarrollos futuros para mejorar la implementación.

6.1. Conclusión

Una vez finalizado el proyecto...

6.2. Desarrollos futuros

Un posible desarrollo...

Apéndice A

Definiciones básicas

Bibliografía

[1] Leslie Lamport et al. The latex project, 2024.

[2] Tikzmaker. https://tikzmaker.com/editor, 2024.

[3] ISO/IEC 9899:2018 Information technology ? Programming languages ? C. https://www.iso.org/standard/82075.html, 2024. International Organization for Standardization, Geneva, Switzerland.

[4] ISO/IEC 14882:2025 Information technology ? Programming languages ? C++. https://www.iso.org/standard/83626.html, 2024. International Organization for Standardization, Geneva, Switzerland.

[5] Infineon Technologies AG. CY8CPROTO-063-BLE PSoC 6 BLE Prototyping Kit. https://www.infineon.com/cms/en/product/evaluation-boards/cy8cproto-063-ble/, 2020. Consultado: 2025-05-05.

[6] National Institute of Standards, Technology (NIST), and Morris J. Dworkin. Sha-3 standard: Permutation-based hash and extendable-output functions, 2015-08-04 00:08:00 2015.

[7] National Institute of Standards and Technology. Post-quantum cryptography - round 3 submissions. https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions, 2020. Consultado: 2025-05-05.

[8] National Institute of Standards and Technology. Post-quantum cryptography - round 4 submissions. https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions, 2022. Consultado: 2025-05-05.

[9] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Kyber: Algorithm Specifications and Supporting Documentation (version 3.01). Technical report, CRYSTALS Project, January 2021. NIST PQC Round 3 submission.

[10] Ardianto Satriawan, Rella Mareta, and Hanho Lee. A complete beginner guide to the number theoretic transform (NTT). Cryptology ePrint Archive, Paper 2024/585, 2024.

[11] Miguel A. Moreno. Primitive n-th roots of unity of finite fields. https://www.csd.uwo.ca/~mmorenom/CS874/Lectures/Newton2Hensel.html/node9.html, n.d. Consultado: 2025-05-05.

[12] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301, 1965.

[13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6), November 2013.

[14] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554, Berlin, Heidelberg, 1999. Springer.

[15] Andrea Basso, José María Bermudo Mera, Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Michiel Van Beirendonck, and Frederik Vercauteren. SABER: Mod-LWR based KEM (Round 3 Submission). Technical report, Katholieke Universiteit Leuven and University of Birmingham, 2020. NIST PQC Round 3 submission.

[16] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jurjen Bos, Jean-Christophe Deneuville, Arnaud Dion, Philippe Gaborit, Jérôme Lacan, Edoardo Persichetti, Jean-Marc Robert, Pascal Véron, and Gilles Zémor. HQC: Hamming Quasi-Cyclic (Fourth Round Submission). Technical report, HQC Team, October 2022. NIST PQC Round 4 submission.

[17] Nicolas Aragon, Paulo S. L. M. Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Santosh Ghosh, Shay Gueron, Tim Güneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Jan Richter-Brockmann, Nicolas Sendrier, Jean-Pierre Tillich, Valentin Vasseur, and Gilles Zémor. BIKE: Bit Flipping Key Encapsulation (Round 4 Submission). Technical report, BIKE Team, October 2022. NIST PQC Round 4 submission.