Moroccan Journal of Algebra and Geometry with Applications
Supported by Sidi Mohamed Ben Abdellah University, Fez, Morocco
Volume 4, Issue 1 (2025), pp 109-137
ISSN: 2820-7114

Title :

Survey on Side-Channel Attacks on Code-Based Key Encapsulation Mechanism

Author(s):

Pierre-Louis Cayrel, Mohamed Fall, Laila Mesmoudi, and Cheikh Tiecoumba Gueye

Moroccan Journal of Algebra and Geometry with Applications
Vol. 4(1) (2025), 109–137

Supported by Sidi Mohamed Ben Abdellah University, Fez, Morocco **ISSN: 2820-7114**

Survey on Side-Channel Attacks on Code-Based Key Encapsulation Mechanism

Pierre-Louis Cayrel[1], Mohamed Fall[2], Laila Mesmoudi[2] and Cheikh Tiecoumba Gueye[2]

[1] Universitan Monnet Saint-Etienne, CNRS, Institut d Optique Graduate School,
Laboratoire Hubert Curien UMR 5516, F-42023,SAINT-ETIENNE, France.
email: pierre.louis.cayrel@univ-st-etienne.fr
[2] Universiteikh Anta Diop, Laboratoire d'Alge de Cryptologie de Gie algique et Application,
Dakar, Sgal.
email: fall13.edu@gmail.com
email: laila.mesmoudi@ucad.edu.sn
email: cheikht.gueye@ucad.edu.sn

Communicated by Abderrahmane Nitaj
(Received 21 October 2024, Revised 25 November 2024, Accepted 31 December 2024)

Abstract. Code-based Key Encapsulation Mechanisms (KEM) represent a promising solution in the field of post-quantum cryptography, providing security guarantees rooted in well-studied mathematical problems, such as decoding linear codes with or without errors. This survey offers an in-depth analysis of the main code-based KEM constructions, including Classic McEliece, BIKE, and HQC. We describe their theoretical foundations, key algorithms (key generation, encapsulation, and decapsulation), as well as their strengths and weaknesses in terms of security and efficiency.

Particular attention is given to the practical vulnerabilities of these schemes against cryptanalytic attacks, such as side-channel attacks, fault injection attacks, and specific techniques for key and message recovery. We also highlight the countermeasures proposed in the literature to strengthen their security, including constant-time algorithms, hardware protections, and improvements to decoding mechanisms.

Furthermore, we discuss the challenges related to the implementation and adoption of these systems in constrained environments, such as the Internet of Things (IoT), while evaluating their potential in contexts where quantum security becomes a critical necessity. This work aims to provide a clear and comprehensive overview of recent advances and future directions for the development of robust and efficient code-based KEMs, contributing to the transition toward post-quantum cryptographic standards.

Key Words: Post-quantum cryptography, Code-based cryptography, Key encapsulation mechanisms (KEMs), Classic McEliece, BIKE, HQC, Cryptanalysis, Countermeasures.
2020 MSC: Primary 94A60; Secondary 14G50.

Dedicated to our Professor David E. Dobbs for his 80th Birthday.

1 Introduction

In the era of quantum computers, the foundations of modern cryptography, which have long been considered secure against classical computing attacks, are being profoundly challenged. Algorithms such as RSA, DSA, or elliptic curve-based cryptographic systems, which have underpinned the security of digital communications for several decades, face a new and formidable threat from quantum computing. Specifically, quantum algorithms, especially Shor's algorithm [79], can efficiently solve

mathematical problems based on these systems, such as integer factorization and the discrete logarithm problem. These problems, which are infeasible to solve within a reasonable timeframe using classical computers, can be solved in polynomial time with a sufficiently large quantum computer. As a result, the security of many widely deployed cryptosystems is severely undermined by the advent of quantum computing technologies. This imminent risk has galvanized the cryptographic community into searching for new cryptographic algorithms that can resist both classical and quantum attacks, giving rise to the burgeoning field of post-quantum cryptography.

Among the various candidates proposed for post-quantum cryptography, cryptosystems based on error-correcting codes stand out as particularly promising. These systems derive their security from the intrinsic difficulty of decoding linear codes, a problem that has been studied for many years and is believed to be hard, even for quantum computers. Error-correcting codes have a long history in both cryptography and communication theory, where they are used to correct transmission errors. However, their cryptographic strength lies in the complexity of the decoding problem. Problems like the syndrome decoding problem, which asks one to find the closest codeword to a given vector, are believed to be NP-hard, making them resistant to both classical and quantum attacks. Unlike systems based on number-theoretic problems (such as RSA), no efficient quantum algorithm has yet been found to solve the hardest problems in coding theory. This natural resistance to quantum attacks makes code-based cryptosystems an attractive option for post-quantum security.

One of the most well-known and extensively studied code-based cryptosystems is the McEliece cryptosystem [87], which was proposed as early as 1978 by Robert McEliece. Despite its age, the McEliece scheme remains one of the most secure post-quantum cryptosystems, largely due to its reliance on the hardness of decoding a general linear code. The scheme uses a family of algebraic codes known as Goppa codes, which are well suited to cryptographic applications due to their structure and resistance to known attacks. Over the years, various attempts have been made to break the McEliece cryptosystem, both from a classical and a quantum perspective, but it has consistently demonstrated remarkable resilience. This durability, combined with the performance advantages that come with well-understood decoding algorithms, ensures that the McEliece cryptosystem continues to be a central pillar in post-quantum cryptography research. Its proven security and efficiency in resisting attacks, even with the potential capabilities of quantum computers, have positioned it as one of the leading candidates for long-term cryptographic security.

In the ongoing efforts to standardize post-quantum cryptographic algorithms, several code-based key encapsulation mechanisms (KEMs) have emerged as front-runners. The National Institute of Standards and Technology (NIST), which is leading the global effort to define post-quantum cryptography standards, has recognized several code-based KEMs as highly promising candidates. These include Classic McEliece [87], BIKE (Bit Flipping Key Encapsulation) [82], and HQC (Hamming Quasi-Cyclic) [81]. Each of these schemes is grounded in variants of hard problems in coding theory, and they have been designed with a focus on achieving an optimal balance between security, efficiency, and practicality for real-world deployment.

Classic McEliece, in particular, is notable for its long-standing reputation as a highly secure cryptosystem. The scheme is based on the difficulty of decoding random linear codes, and it employs Goppa codes to ensure both robustness and efficiency. However, a common critique of the McEliece cryptosystem is the relatively large size of its public keys, which can present challenges in terms of storage and transmission, especially in resource-constrained environments. Nonetheless, ongoing research efforts are focused on mitigating this issue by exploring alternative code families and optimization techniques to reduce key sizes without compromising security.

The BIKE and HQC schemes are more recent developments in the landscape of code-based cryptography, and they introduce innovative approaches to key encapsulation. BIKE, for instance, utilizes quasi-cyclic codes and employs a bit-flipping decoding algorithm, which enables it to achieve competitive performance in both encryption and decryption operations. Its construction leverages the

cyclic structure of the underlying codes to reduce the size of public keys, while maintaining strong security guarantees based on the hardness of decoding quasi-cyclic codes. Similarly, HQC combines the use of quasi-cyclic codes with the syndrome decoding problem, a well-established hard problem in coding theory. This combination allows HQC to offer strong security against quantum attacks, while also achieving high efficiency in key generation and message encryption processes.

Despite the theoretical strengths of these code-based KEMs, cryptography in practice is rarely immune to real-world challenges. One of the most significant concerns for any cryptosystem is its vulnerability to side-channel attacks [84, 5, 85, 77, 74, 76, 83, 86]. These attacks exploit indirect information leaked during the execution of cryptographic algorithms, such as variations in power consumption, electromagnetic emissions, or timing variations, to recover secret information like private keys. For example, recent studies have demonstrated that timing attacks on the decoding process of code-based KEMs, including Classic McEliece, BIKE, and HQC, can reveal sensitive information that compromises their security. Such findings emphasize the importance of implementing cryptographic algorithms in a way that minimizes or eliminates these side-channel leakages, through techniques such as constant-time operations, randomization of processing steps, and masking.

In addition to side-channel attacks, fault injection attacks (FIA) [56, 14] pose another serious threat to the practical security of code-based cryptosystems. These attacks involve deliberately inducing faults in the hardware or software implementation of a cryptosystem, with the goal of causing errors that can be exploited to recover private keys or other sensitive data. Fault injection attacks can be highly effective against code-based KEMs if countermeasures are not carefully implemented. For example, by causing specific errors in the decoding process, an attacker might be able to gather enough information to perform a key-recovery attack, which would otherwise be impossible through conventional means.

In the case of Classic McEliece, BIKE, and HQC, several cryptanalytic studies have highlighted potential vulnerabilities related to the decapsulation process or the key generation routines. In particular, some recent attacks have identified timing leaks in the decoding algorithms used by these schemes, which could potentially allow an attacker to recover the secret key by analyzing the time it takes to process different inputs. These findings illustrate that while code-based cryptosystems are theoretically resistant to quantum attacks, they must still contend with the practical vulnerabilities that arise from their implementation in real-world devices and systems.

The ongoing research into these attacks underscores the need for robust countermeasures that address both theoretical and practical vulnerabilities. Cryptographic engineers must develop and deploy secure implementations that defend against side-channel and fault injection attacks, ensuring that these post-quantum cryptosystems can withstand not only the computational power of future quantum computers but also the more immediate threats posed by real-world attackers. By understanding and addressing these challenges, we can ensure that code-based KEMs like Classic McEliece, BIKE, and HQC will provide the long-term security required for the post-quantum era.

Organization. The remaining of the survey is described as follows. In **Section 2**, we provide a general presentation of the Classic McEliece, BIKE, and HQC key encapsulation mechanisms. In **Section 3**, we present various cryptanalysis on these different candidates. **Section 4** discusses the main results. Finally, in **Section 5**, we conclude the document.

2 General presentation of candidates

2.1 Classic McEliece

McEliece is one of the oldest and most established code-based cryptosystems, originally proposed by Robert McEliece in 1978. It is a public-key cryptosystem that relies on the hardness of decoding

random linear codes, a problem widely believed to be resistant not only to classical attacks but also to quantum attacks [80]. At the heart of the McEliece cryptosystem is its use of Goppa codes, a type of algebraic error-correcting linear code that is particularly suited to cryptographic purposes due to its efficient decoding algorithms. The core idea behind the McEliece scheme is that while decoding a randomly generated linear code is NP-hard, the problem becomes tractable when using structured codes like Goppa codes, which can be efficiently decoded by the legitimate user. This clever distinction between the public key, which appears to be a random linear code, and the private key, which allows for efficient decoding, is what provides the security and functionality of the scheme.

The security of the McEliece cryptosystem is derived from the inherent complexity of the decoding problem. In more technical terms, the challenge is to recover the original message from the received codeword when only the public key is known. This problem is computationally infeasible for an attacker because it involves finding the closest codeword to a given vector in a large-dimensional vector space, a problem known as the *syndrome decoding problem*. Unlike cryptosystems based on number-theoretic problems such as RSA, the McEliece cryptosystem is considered quantum-resistant. This is primarily due to the fact that no efficient quantum algorithm, including Shors algorithm, has been found that can solve the hard problems underlying code-based cryptography in polynomial time.

One of the strengths of the McEliece scheme is its resilience to a broad range of attacks. Over the past several decades, McEliece has resisted cryptanalytic efforts, including both classical and quantum attacks. This remarkable durability has earned McEliece a prominent place in the NIST post-quantum cryptography standardization process, where it is considered one of the leading candidates for future cryptographic standards in the quantum era. However, McEliece is not without its challenges. The size of its public key is relatively large, often on the order of several megabytes, which presents challenges for its implementation, particularly in memory-constrained environments such as embedded systems, IoT devices, and mobile platforms. Despite these practical limitations, its robustness against quantum threats makes McEliece one of the most reliable and trusted options for long-term secure communication.

The McEliece key encapsulation mechanism (KEM) is designed to provide secure encryption in post-quantum environments. It consists of three distinct phases: *key generation*, *encapsulation*, and *decapsulation*. Each phase plays a crucial role in ensuring the overall security of the system, with interdependent operations that collectively enable secure key exchange. The structure of these phases ensures that even if an attacker intercepts the encapsulated key, they will not be able to reverse the encapsulation without access to the private key.

2.1.1 Classic McEliece - key generation

The **key generation** process creates a public-private key pair based on the security parameters (m, n, t). During this phase, a random set of elements in $\mathbb{F}_{2^m}$ is generated, along with an irreducible monic polynomial of degree t. This enables the calculation of the parity-check matrix, which is later used to encapsulate data. Once the parity-check matrix is converted into a binary matrix and put into standard form, the public key is defined by the public matrix $\boldsymbol{T}$, while the private key consists of the polynomial g and the set $\mathcal{L}$.

Algorithm 1 The key-generation algorithm of the *Classic McEliece* KEM.

Input: The *Classic McEliece* security parameters (m, n, t),
Output: The private key $\mathsf{sk} = (g, \mathcal{L})$ and the public key $\mathsf{pk} = \boldsymbol{T}$

1: Generate a set $\mathcal{L} = \{\alpha_0, \ldots, \alpha_{n-1}\}$ of random elements of $\mathbb{F}_{2^m}$ with $\#\mathcal{L} = n$
2: Generate an irreducible monic polynomial $g \in \mathbb{F}_{2^m}[x]$ of degree t
3: Compute the $t \times n$ parity-check matrix $\boldsymbol{H}$

$$\boldsymbol{H} = \begin{pmatrix} (g(\alpha_0))^{-1} & \cdots & (g(\alpha_{n-1}))^{-1} \\ \vdots & \ddots & \vdots \\ \alpha_0^{t-1}(g(\alpha_0))^{-1} & \cdots & \alpha_{n-1}^{t-1}(g(\alpha_{n-1}))^{-1} \end{pmatrix}$$

4: Transform $\boldsymbol{H}$ to an $mt \times n$ binary matrix $\boldsymbol{H}'$ ▷ $\mathbb{F}_{2^m} \to \mathbb{F}_2^m$ mapping
5: Transform $\boldsymbol{H}'$ in standard-form $\boldsymbol{H}_{\text{pub}} = (\boldsymbol{I}_{mt} \mid \boldsymbol{T})$
6: Return the private key $\mathsf{sk} = (g, \mathcal{L})$ and the public key $\mathsf{pk} = \boldsymbol{T}$

2.1.2 Classic McEliece - encapsulation

During the **encapsulation** phase, a random vector $\boldsymbol{e}$ is generated with a Hamming weight of t, corresponding to the number of errors the code can correct. This error vector is encapsulated into a ciphertext $\boldsymbol{z}$, obtained by multiplying the public matrix $\boldsymbol{T}$ with $\boldsymbol{e}$. Finally, a session key K is derived by hashing $\boldsymbol{e}$ and $\boldsymbol{z}$, ensuring the confidentiality of the transmitted data

Algorithm 2 The encapsulation algorithm of the *Classic McEliece* KEM.

Input: The public key $\mathsf{pk} = \boldsymbol{T}$
Output: The ciphertext $\boldsymbol{z}$ and the session key K

1: Generate a random vector $\boldsymbol{e} \in \mathbb{F}_2^n$ with $\mathsf{wt}(\boldsymbol{e}) = t$
2: Compute $\boldsymbol{z} = (\boldsymbol{I}_{mt} \mid \boldsymbol{T})\, \boldsymbol{e}^T$
3: Compute $K = \mathsf{hash}(1|\boldsymbol{e}|\boldsymbol{z})$
4: Return the ciphertext $\boldsymbol{z}$ and the session key K

2.1.3 Classic McEliece - decapsulation

To the **decapsulation** phase, the receiver uses the private key to recover the encapsulated error. By computing the syndrome from the received ciphertext $\boldsymbol{z}$ and employing the Berlekamp-Massey algorithm, the error is located and corrected. This allows for the recovery of the original vector $\boldsymbol{e}$, and consequently, the same session key K as the sender.

Algorithm 3 The decapsulation algorithm of the *Classic McEliece* KEM.

Input: The ciphertext $\boldsymbol{z}$ and the private key $\mathsf{sk} = (g, \mathcal{L})$
Output: The session key K

1: Compute the vector $\boldsymbol{v} = (\boldsymbol{z}, 0, \ldots, 0)$ by padding $\boldsymbol{z}$ with $n - mt$ zeros
2: Compute the parity-check matrix

$$\boldsymbol{H}_{\text{priv}} = \begin{pmatrix} (g(\alpha_0))^{-2} & \cdots & (g(\alpha_{n-1}))^{-2} \\ \vdots & \ddots & \vdots \\ \alpha_0^{2t-1}(g(\alpha_0))^{-2} & \cdots & \alpha_{n-1}^{2t-1}(g(\alpha_{n-1}))^{-2} \end{pmatrix}$$

3: Compute the syndrome $\boldsymbol{s} = \boldsymbol{H}_{\text{priv}}\boldsymbol{v}^T$
4: Compute the error locator pol. $\sigma(x)$ with Berlekamp-Massey algorithm
5: Compute $(\sigma(\alpha_0), \ldots, \sigma(\alpha_{n-1}))$ and recover the error vector $\boldsymbol{e}$
6: Compute $K = \mathsf{hash}(1|\boldsymbol{e}|\boldsymbol{z})$
7: Return the session key K

2.2 BIKE (Bit Flipping Key Encapsulation)

BIKE is a KEM scheme based on binary quasicyclic codes and uses a bit-flipping decoding algorithm. Proposed within the NIST standardization process. BIKE relies on the hardness of decoding QC-MDPC (Quasi-Cyclic Moderate Density Parity Check) codes, a variant of LDPC (Low-Density Parity Check) codes, to encapsulate a session key. Errors are corrected using bit-flipping algorithms, which flip the erroneous bits in the received codeword [82]. The keys and ciphertexts are relatively compact compared to systems such as McEliece. BIKE benefits from fast and efficient implementations. The scheme is still vulnerable to practical attacks, such as side-channel attacks. Due to its efficiency and relatively small key sizes, BIKE is suitable for environments where memory and time constraints are critical, such as the Internet of Things (IoT). In terms of its structure, BIKE uses parameters based on QC-MDPC codes, denoted as $[n, r, w]$, where:
r is prime, $n = 2r$, $w = 2d$, with $w = \mathcal{O}\left(\sqrt{n}\right)$ and d odd.

To elaborate the algebraic framework of BIKE , the system operates in the cyclic polynomial ring $\mathcal{R} = \mathbb{F}_2[X]/(X^r - 1)$, where each element h in $\mathcal{R}$ has a Hamming weight $|h|$. Furthermore, $\mathcal{H}_w$ defines the set of elements in $\mathcal{R}^2$ with specific weight properties, expressed as follows:

$$\mathcal{H}_w = \{ (h_0, h_1) \in \mathcal{R}^2 \text{ s.t } |h_0| = |h_1| = d \}$$

In cryptographic operations, BIKE uses hash functions L and K to generate and encapsulate keys. These functions produce outputs of appropriate lengths, and another hash function, H, produces a weight vector t, where t represents the error correction capability of the decoding process [77].

2.2.1 BIKE (Bit Flipping Key Encapsulation) - key generation

The key generation process in BIKE is straightforward, but it plays a critical role in ensuring the security of the scheme. First, a random secret σ is generated. Then, two polynomials, h_0 and h_1, are randomly selected from the set $\mathcal{H}_w$. The public key h is calculated as the product of h_1 and the inverse of h_0. At the end of this process, the private key consists of (h_0, h_1, σ), while the public key is the polynomial h.

Algorithm 4 The key-generation algorithm of the BIKE KEM.

Input: The BIKE security parameters (ℓ),
Output: The private key $\mathsf{sk} = (h_0, h_1, \sigma)$ and the public key $\mathsf{pk} = h$

1: Pick $\sigma \xleftarrow{s} \{0,1\}^{\ell}$
2: Pick $(h_0, h_1) \xleftarrow{s} \mathcal{H}_w$
3: Compute $h \leftarrow h_1 {h_0}^{-1}$
4: Return the private key $\mathsf{sk} = (h_0, h_1, \sigma)$ and the public key $\mathsf{pk} = h$

2.2.2 BIKE - encapsulation

Once the keys are established, encapsulation involves using the public key to generate a ciphertext and a shared session key. The process starts by picking a random message m. From m, an error vector (e_0, e_1) is computed using the hash function H. Then, the ciphertext is formed by combining the error vector and the public key, producing two components (c_0, c_1). The shared secret $\mathcal{K}$ is derived from the message m and the ciphertext through the hash function K, which ensures the secure encapsulation of the session key.

Algorithm 5 The encapsulation algorithm of the BIKE KEM.

Input: The public key $\mathsf{pk} = h$ and ℓ
Output: The ciphertext $c = (c_0, c_1)$ and the session key $\mathcal{K}$

1: Pick $m \xleftarrow{s} \{0,1\}^{\ell}$
2: Compute $(e_0, e_1) \leftarrow \mathsf{H}(m)$
3: Compute $(c_0, c_1) \leftarrow (e_0 + e_1 h,\ m \oplus \mathsf{L}(e_0, e_1))$
4: Compute the shared secret $\mathcal{K} \leftarrow \mathsf{K}(m, c_0, c_1)$
5: Return the ciphertext $c = (c_0, c_1)$ and the session key $\mathcal{K}$

2.2.3 BIKE - decapsulation

The decapsulation process reverses the encapsulation using the private key. The bit-flip decoding algorithm is applied to recover the error vector e' from the ciphertext component c_0 using the private polynomials h_0 and h_1. Once e' is recovered, the original message m' is derived by XORing c_1 with the hash of e'. If the recovered error vector e' matches the one derived from m', the session key $\mathcal{K}$ is reconstructed. If decoding fails, an alternative session key is computed using the private random value σ.

Algorithm 6 The decapsulation algorithm of the BIKE KEM.

Input: The ciphertext c and the private key $\mathsf{sk} = (h_0, h_1, \sigma)$
Output: The session key $\mathcal{K}$

1: Compute $e' \leftarrow$ **Decoder**$(c_0 h_0, h_0, h_1)$
2: Compute $m' \leftarrow c_1 \oplus \mathsf{L}(e')$
3: If $e' = \mathsf{H}(m')$ Then Compute $\mathcal{K} = \mathsf{K}(m', c)$
4: Else Compute $\mathcal{K} = \mathsf{K}(\sigma, c)$
5: Return the session key $\mathcal{K}$

2.3 HQC (Hamming Quasi-Cyclic)

HQC is a scheme that combines the principles of linear codes and quasicyclic structures to ensure its security. Similarly to the Learning with Errors (LWE) problem, HQC is based on the difficulty of

decoding linear codes with added noise. Noise makes decoding difficult for an attacker, while the legitimate receiver, possessing the private key, can efficiently recover the message. This mechanism encapsulates a key by introducing noise into the encoded vectors, ensuring that the legitimate recipient can recover the original data while an attacker faces an intractable problem.
The HQC scheme operates with a decodable $[n,k]$ code C, generated by a matrix $\mathbf{G} \in \mathbb{F}_2{}^{k\times n}$, which can correct at least δ errors via an efficient decoding algorithm, denoted as $C.Decode(\cdot)$. Like other post-quantum schemes, HQC employs hash functions $\mathcal{G}$, $\mathcal{H}$, and $\mathcal{K}$ for secure key generation and encapsulation [81]. Compared to the McEliece scheme, HQC offers more reasonable key sizes while maintaining strong security guarantees. It is also highly resistant to quantum attacks. However, as with BIKE , HQC faces challenges from practical security attacks, such as side-channel attacks. HQC is particularly appealing in applications requiring a balance between security, efficiency, and key size, such as secure communication infrastructures.

2.3.1 HQC (Hamming Quasi-Cyclic) - key generation

The key generation process in HQC involves selecting random elements from a cyclic polynomial ring. First, a polynomial h is randomly chosen from $\mathcal{R} = \mathbb{F}_2[X]/(X^n-1)$. Then, two polynomials, x and y, are selected such that their Hamming weights are equal to the security parameter w. The public key is computed as the pair (h,s), where $s = x + h\cdot y$, and the private key is (x,y). This process ensures that the legitimate user has the required parameters for both encryption and decryption, while the structure of the public key maintains the security of the scheme.

Algorithm 7 The key-generation algorithm of the HQC KEM.

Input: The HQC security parameters (w)
Output: The private key $\mathsf{sk} = (x,y)$ and the public key $\mathsf{pk} = (h, s = x + h.y)$

1: $h \xleftarrow{s} \mathcal{R}$ ▷ $\mathcal{R} = \mathbb{F}_2[X]/(X^n-1)$: Cyclic polynomial ring
2: $(x,y) \xleftarrow{s} \mathcal{R}^2$ such that $w(x) = w(y) = w$
3: Return the private key $\mathsf{sk} = (x,y)$ and the public key $\mathsf{pk} = (h, s = x + h.y)$

In addition to the key generation, encapsulation and decapsulation algorithms, we will present the encryption and decryption algorithms that will be used in the encapsulation and decapsulation phases.

2.3.2 HQC - Encryption

During encryption, the sender generates random error and randomness vectors. Specifically, a random error vector e and randomness vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ are chosen, where the Hamming weights of these vectors match the predefined security parameters. The ciphertext is then formed as two components, $\mathbf{u} = \mathbf{r}_1 + h\cdot\mathbf{r}_2$ and $\mathbf{v} = \mathbf{m}\cdot\mathbf{G} + \mathbf{s}\cdot\mathbf{r}_2 + \mathbf{e}$, where $\mathbf{m}$ is the encoded message. This process ensures that the legitimate recipient, who knows the private key, can correctly decrypt the message, while an attacker without the private key will struggle to do so.

Algorithm 8 The encryption algorithm of the HQC.

Input: The public key $\mathsf{pk}, \mathbf{m}, w_e, w_r$
Output: The ciphertext c

1: generates $e \xleftarrow{s} \mathcal{R}$, $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2) \xleftarrow{s} \mathcal{R}^2$ such that $w(e) = w_e$, $w(\mathbf{r}_1) = w(\mathbf{r}_2) = w_r$
2: Sets $\mathbf{u} = \mathbf{r}_1 + h.\mathbf{r}_2$ and $\mathbf{v} = \mathbf{mG} + \mathbf{s}.\mathbf{r}_2 + \mathbf{e}$
3: Return the ciphertext $c = (\mathbf{u}, \mathbf{v})$

2.3.3 HQC - **Decryption**

The decryption process reverses the encryption operation using the private key (x, y). The receiver computes the plaintext by subtracting the product of $\mathbf{u}$ and y from $\mathbf{v}$, followed by decoding the resulting value using the algorithm $C.Decode$. This allows the legitimate recipient to recover the original message $\mathbf{m}$ efficiently. The decoding algorithm ensures that even with the added noise, the message can be correctly recovered.

Algorithm 9 The decryption algorithm of the HQC.

Input: The ciphertext $(\mathbf{u}, \mathbf{v})$ and the private key $\mathsf{sk} = (x, y)$
Output: The plaintext $\mathbf{m}$
1: Return the plaintext $\mathbf{m} = C.Decode(\mathbf{v} - \mathbf{u}.\mathbf{y})$

2.3.4 HQC - **encapsulation**

Encapsulation in HQC follows a structure similar to other KEM schemes. The sender first generates a random message $\mathbf{m}$, which is then used to derive randomness θ through the hash function $\mathcal{G}$. The ciphertext c is generated by encrypting the message $\mathbf{m}$ using the derived randomness and the public key. Simultaneously, the shared key $\mathcal{K}$ is derived using the hash function $\mathcal{K}$, which takes the message and the ciphertext as inputs. To ensure the integrity of the message, a hash value $d = \mathcal{H}(\mathbf{m})$ is computed and returned alongside the ciphertext.

Algorithm 10 The encapsulation algorithm of the HQC KEM.

Input: The public key $\mathsf{pk} = (h, s = x + h.y)$
Output: The ciphertext (c, d) and shared key $\mathcal{K}$
1: Generate a random message $\mathbf{m} \in {\mathbb{F}_2}^k$
2: Derive the randomness $\theta \leftarrow \mathcal{G}(\mathbf{m})$
3: Generate the ciphertext $c = Encrypt(pk, m, \theta)$ ▷ θ uses to generate $\mathbf{r}$ and e
4: Derive the symmetric key $K \leftarrow \mathcal{K}(m, c)$
5: Let $d \leftarrow \mathcal{H}(m)$
6: Return (c, d) and shared key $\mathcal{K}$

2.3.5 HQC - **decapsulation**

The decapsulation process mirrors the encapsulation in reverse. The recipient uses the private key to decrypt the ciphertext and recover the original message m'. Using this message, the randomness θ' is recomputed, and a new ciphertext c' is generated. The recipient then checks the integrity of the ciphertext by comparing c' with the received ciphertext c and verifying the hash value d. If the integrity check passes, the shared key $\mathcal{K}$ is derived from the decrypted message and the ciphertext, completing the decapsulation process.

Algorithm 11 The decapsulation algorithm of the HQC KEM.

Input: (c, d) and the private key $\mathsf{sk} = (x, y)$
Output: The Shared key $\mathcal{K}$
1: Decrypt $m' \leftarrow Decrypt(sk, c)$
2: $\theta' \leftarrow \mathcal{G}(\mathbf{m}')$
3: $c' \leftarrow Encrypt(pk, m', \theta')$
4: If $c \neq c'$ or $d \neq \mathcal{H}(m')$ then abort. Otherwise, derive the shared key $K \leftarrow \mathcal{K}(m, c)$
5: Return the Shared key $\mathcal{K}$

2.4 Comparison of the three KEMs

KEM	Quantum Security	Key Sizes	Efficiency	Vulnerabilities
Classic McEliece	Very high	Very large	Slow	SCA, FIA, Algebraic Attack
BIKE	High	Medium	Fast	SCA, FIA
HQC	High	Medium	Balanced	SCA, FIA

Table 1: Comparison of Classic McEliece, BIKE , and HQC

3 Cryptanalysis of the different candidates

3.1 Cryptanalysis on Classic McEliece

In this section, we examine the key-recovery and message-recovery attacks on *Classic McEliece*, one of the most prominent candidates in the field of code-based post-quantum cryptography. Over the years, various side-channel attacks have been developed, targeting both the decapsulation and key generation phases of this cryptosystem. These attacks highlight the ongoing challenges of ensuring the security of McEliece-based systems in real-world implementations.

3.1.1 Key-recovery

Several key-recovery side-channel attacks have been proposed, demonstrating the continuous evolution of cryptographic vulnerabilities and the sophisticated techniques used to exploit these weaknesses in *Classic McEliece*.

Guo et al. presented a highly impactful key-recovery side-channel attack focusing on the decapsulation step of *Classic McEliece* [9]. This attack is significant due to its ability to exploit the decapsulation of n invalid ciphertexts, where n represents the security parameter of *Classic McEliece*. By carefully observing the decapsulation process, attackers can deduce secret information, such as the private key. This attack emphasizes the necessity of implementing robust countermeasures, including the use of constant-time operations and secure memory management techniques, to prevent such vulnerabilities from being exploited.

Similarly, Seck et al. introduced a partial key-recovery attack during the decapsulation phase [5]. This attack focuses on recovering the private polynomial g by leveraging side-channel information while the polynomial's coefficients are loaded into the system. Although this attack does not fully recover the private key, it exposes critical weaknesses in how certain cryptographic operations are implemented, highlighting the potential for future improvements in the protection of side-channel information.

Expanding the scope of side-channel vulnerabilities, Brinkmann et al. described a key-recovery side-channel attack that occurs during the key generation process [24]. Their method relies on simulated leakages arising during Gaussian elimination, introducing a new class of vulnerabilities in *Classic McEliece*. This attack shows that not only the decapsulation process but also the key generation stage is susceptible to side-channel attacks, stressing the importance of securing all stages of the cryptographic pipeline.

In addition to these practical attacks, theoretical approaches have been explored by Kirshanova and May, who proposed algorithms for full key recovery assuming an attacker has access to "hints"partial information about the secret key [18]. While their work is theoretical, it underscores the potential vulnerabilities in McEliece-based cryptosystems if any secret information is leaked, reinforcing the need for careful design and secure implementation practices.

In summary, these studies illustrate the diverse range of key-recovery attacks that target different phases of the *Classic McEliece* cryptosystem. The complexity and effectiveness of these attacks highlight the importance of continuously developing countermeasures to protect against both classical and quantum threats.

Table 2: Comparison of key-recovery attacks on the *Classic McEliece* KEM.

Article	Step (Algorithm)	Scenario	Attack Type	Implementation and Hardware
[24]	Gaussian Elimination (KeyGen)	1 trace	Side-channel (sim.) Full key recovery	Ref. & Botan N/A
[9]	FFT $\sigma(x)$ (Decaps.)	n traces $\mathrm{wt}(\boldsymbol{e}) = 1$	Side-channel Full key recovery	Ref. & optimized FPGA & ARM
[5] (Decaps.)	load(g)	1 trace	Side-channel Partial key recovery	Optimized ARM
[12]	$\sigma(x)$ / Validation (Decaps.)	No CCA2 Faults in σ	Fault injection (sim.) Alternate key recovery	Reference RISC-V RTL

3.1.2 Message-recovery

Message-recovery side-channel attacks target the decryption process of cryptographic systems like *Classic McEliece*, often leveraging physical vulnerabilities to recover secret information. These attacks are particularly concerning as they can recover entire messages without directly attacking the private key.

Lahr et al. demonstrated a notable message-recovery attack that enhances the syndrome by adding random columns from the parity-check matrix [10]. This enhanced syndrome is then used in a decoding oracle, where side-channel analysis is performed to detect changes in the weight of the input message. This method effectively reveals whether the bits corresponding to the modified columns are predominantly 0 or 1, allowing the attacker to recover the message efficiently. Additionally, the use of Information Set Decoding (ISD) further reduces the number of oracle calls needed, enhancing the attack's practicality.

Cayrel et al. expanded on this approach with a laser-induced fault injection attack during the encryption process [14]. By manipulating the instructions within the cryptographic operation, they converted an XOR operation into an ADD operation, causing the matrix-vector multiplication to be performed over $\mathbb{N}$ (the set of natural numbers) instead of $\mathbb{F}_2$ (the binary field). This alteration modified the syndrome decoding problem and allowed for message recovery within minutes, demonstrating the practical feasibility of such attacks.

Another approach described by Danner et al. [44] proposes a framework for fault injection attacks that specifically targets the secret key, rather than the message. This attack, although theoretical, highlights how fault injection techniques can be extended to compromise the cryptographic key itself, revealing vulnerabilities in the protection of private key material during decryption.

Xagawa et al. presented a generic attack on the Fujisaki-Okamoto transformation, widely used in various post-quantum cryptographic schemes [56]. This attack exploits an instruction-skipping vulnerability during decapsulation, allowing the attacker to recover the message with a single fault. The attack's effectiveness was experimentally demonstrated, but it has not yet been applied to *Classic McEliece*, making it a future research avenue for potential vulnerabilities.

Finally, the concept of augmenting the Syndrome Decoding Problem (SDP) with additional information obtained through physical attacks has been further explored [73]. The introduction of the Integer Syndrome Decoding Problem ($\mathbb{N}$–SDP) [68], and the use of integer linear programming techniques, represents a promising new approach to solving SDP-based cryptosystems. However, while this method significantly reduces the problem's complexity, it still faces challenges related to error tolerance and physical attack repeatability, limiting its practical applicability.

Table 3: Comparison of message-recovery attacks on the *Classic McEliece* KEM.

Article	**Step (Algorithm)**	**Scenario**	**Attack Type**	**Implem. and Hardware**
[14]	Compute Syndrome (Encrypt)	1 trace	Laser fault / ILP Full message recovery	ARM Cortex-M3 core
[10]	Modified syndrome (Decrypt)	$\leq k$ traces Iterative Chunking	Side-channel / ISD Full message recovery	FPGA implementation EM side channel
[44]	d-th coefficient of $\sigma_p(x)$ (Decrypt)	Fault injection ($d = 0$ and $d = 2$)	Fault Injection Alternate key recovery	FPGA implementation CoCoA-5

3.1.3 Countermeasures

To address countermeasures against cryptanalytic attacks on the Classic McEliece scheme, it is essential to introduce protective techniques while highlighting their relevance in the context of past attacks. Although the scheme has proven robust against classical attacks, particularly due to the difficulty of decoding linear codes, recent attack vectors require new strategies.

The first set of countermeasures targets side-channel attacks, which exploit information leakage during the decoding process. The use of hardware countermeasures, such as adding noise to critical operations, or software techniques like masking, helps limit these leaks. Masking transforms sensitive data into protected data using random operations, so that an attacker cannot extract any useful information.

Another threat comes from algebraic and structural attacks, which aim to exploit the particular structure of the Goppa codes used in Classic McEliece. To counter these attacks, several researchers have proposed masking the schemes internal parameters, notably by using generator or parity-check matrices that do not reveal information about the code structure. This significantly complicates the attacker's task in identifying algebraic vulnerabilities.

Furthermore, performance optimizations, especially through specific hardware implementations, must be accompanied by precautions to prevent timing attacks. Here, the objective is to ensure that sensitive operations, such as error decoding, are performed in constant time to prevent the observation of exploitable timing variations.

Finally, recent countermeasure proposals include diversifying the scheme instances, such as using

alternative code families or introducing random perturbations in the public key. These strategies aim to make structural cryptanalysis, specific to Goppa codes, more challenging.

In conclusion, while Classic McEliece has withstood numerous theoretical and practical attacks, these countermeasures reinforce security against evolving cryptanalytic techniques. The effectiveness of these protection techniques depends on their proper implementation, and it is crucial to remain vigilant about emerging vulnerabilities, particularly in the context of quantum computing.

3.2 Cryptanalysis on BIKE

The Bit Flipping Key Encapsulation (BIKE) mechanism is an important candidate in the field of post-quantum cryptography, designed to provide secure key encapsulation in quantum-resistant environments. Despite its theoretical robustness, BIKE has been shown to exhibit practical vulnerabilities, particularly through key-recovery and message-recovery attacks that exploit decryption failures, timing leaks, and side-channel information.

3.2.1 Key-recovery

Key-recovery attacks against BIKE have highlighted critical weaknesses in its decryption mechanisms, particularly those involving rejection-sampling routines and timing information leaks. These vulnerabilities expose the system to significant risks, enabling attackers to recover the private key in certain scenarios.

In [77], a significant vulnerability is identified in the rejection-sampling routines used during BIKE's decapsulation. These routines, which rely on secret-dependent information, can inadvertently leak timing data, allowing an attacker to determine whether the decoding step in BIKE was successful. By analyzing the timing differences, the authors were able to infer critical information about the secret keys distance spectrum, leading to a key-recovery attack using approximately 5.8×10^7 oracle queries. This underscores the importance of implementing robust countermeasures to protect sampling routines from timing analysis.

Another notable key-recovery attack, combining power analysis with information set decoding (ISD), is described in [74]. This attack targets the decapsulation process of BIKE and leverages power consumption traces. By analyzing a single trace of power consumption, the authors were able to recover the complete private key. The attack exploits the manipulation of coordinates in the private key and uses clustering techniques to extract partial information, which is then expanded using ISD to recover the full key. Both C and assembly implementations of BIKE were found to be vulnerable to this attack. For the C version, the complete key is recoverable, while for the assembly version, a large proportion of the keys could still be retrieved.

Further attacks exploiting decryption failures in QC-MDPC (Quasi-Cyclic Moderate Density Parity-Check) schemes are presented in [78]. The authors demonstrate that these failures can reveal whether a target's secret key satisfies a specific property known as "gathering." This information is used to perform a modified decoding algorithm that recovers the secret key. For BIKE with 128-bit security parameters, the complexity of the attack is estimated to be $2^{116.61}$ when ciphertext reuse is not allowed, and it decreases to $2^{98.77}$ when reuse is permitted. This illustrates how decryption failures, particularly in QC-MDPC-based schemes, can significantly weaken security under specific conditions.

In conclusion, these key-recovery attacks reveal the practical limitations of BIKE despite its theoretical strength. To mitigate these vulnerabilities, it is crucial to implement robust countermeasures that minimize information leakage and protect the decryption process from timing, power, and side-channel analysis.

Table 4: Comparison of key-recovery attacks on the BIKE KEM.

Article	Step (Algorithm)	Scenario	Attack Type	Impl. and Hardware
[77]	Compute $\mathsf{H}(m')$ (Decapsulation)	5.8×10^7 queries	Side-channel (sim.) Full key recovery	BIKE-L1, Intel x86
[74]	**Decoder**(c_0, h_0, h_1) (Decapsulation)	1 trace	Power analysis / ISD Full key recovery	Optimized, ARM Cortex-M4
[78]	**Decoder**(c_0, h_0, h_1) (Decapsulation)	Decryption failure	Side-channel Full key recovery	N/A

3.2.2 Message-recovery

Message-recovery attacks exploit both algorithmic and hardware vulnerabilities in BIKE, typically through side-channel techniques such as timing analysis. These attacks target the cryptographic systems ability to reveal partial or complete messages during the decryption process.

One such attack targets a vulnerability introduced by the weight verification step in recent implementations of BIKE, including those found in the liboqs library and the new GitHub implementation. The study [77] describes a timing attack that exploits variations in execution time caused by the rejection-sampling process. In this modified version of BIKE, an additional weight check is performed after the decoding function is called. If the returned error vectors weight does not match the expected value, a random value is assigned. Although this step was introduced to improve security, it inadvertently introduced a new side-channel vulnerability. By repeatedly submitting the same ciphertext and measuring the timing differences in the weight check, attackers can determine whether the weight was correctly verified or not.

This timing attack allows attackers to construct a distinguishing tool that forms the basis of a message-recovery attack. The process involves flipping certain bits in the ciphertext and observing their impact on the weight verification. This information, combined with iterative decoding techniques like ISD, enables the attacker to reconstruct the original message with a high probability of success. Although millions of oracle queries may be required, this attack remains computationally less complex than traditional key-recovery attacks.

Optimizations in BIKE implementations have been proposed to reduce the risk of such attacks. These include binding the public key to the message during encapsulation to mitigate certain forms of manipulation, as well as adjusting encryption parameters to make it more difficult for attackers to exploit these vulnerabilities. However, as highlighted by [77], even countermeasures aimed at improving security can unintentionally introduce new weaknesses, underscoring the need for a holistic approach to cryptographic security.

Table 5: Message-recovery attack on the BIKE KEM.

Article	Step (Algorithm)	Scenario	Attack Type	Impl. and Hardware
[77]	Weight verification (Decapsulation)	Timing variations	Timing attack Message recovery	BIKE-L1, GitHub implementation

In conclusion, these results demonstrate that even with additional security measures, BIKE is still susceptible to message-recovery attacks. A comprehensive approach addressing both hardware vulnerabilities and algorithmic weaknesses is essential to ensure the security of modern cryptographic systems, particularly in the face of side-channel threats.

3.2.3 Countermeasures

Despite the robustness of BIKE, several vulnerabilities have been identified, particularly concerning decoding failures, side-channel attacks, and fault injections. Addressing these challenges requires a combination of techniques to enhance security and resilience.

One primary concern is decoding failures. BIKE's reliance on bit-flipping algorithms for decoding makes it vulnerable when the number of errors approaches the algorithm's correction capacity. When decoding fails, it may inadvertently leak information about the private key. To mitigate this risk, it is possible introducing randomness into the decoding process. By randomizing certain steps, the decoding behavior becomes less predictable, making it harder for attackers to exploit failure patterns. Additionally, improvements to the bit-flipping algorithm, such as incorporating adaptive flipping or increasing the number of decoding rounds, can further enhance error tolerance without significantly impacting performance.

Another significant area of vulnerability is side-channel attacks, where an attacker leverages physical signals, such as timing or power consumption, to gather information about the private key. BIKEs iterative and data-dependent decoding can expose it to these attacks. A key countermeasure here is the use of constant-time implementations, ensuring that operations like syndrome computation and bit-flipping decoding are executed with no variation in time, thus preventing attackers from deducing sensitive information based on timing discrepancies. Furthermore, masking techniques, which involve randomizing intermediate values during computations, can be employed to reduce leakage. By masking key-dependent operations, such as bit-flipping, BIKE can defend against power analysis attacks by obscuring the values an attacker might observe.

In addition to side-channel vulnerabilities, fault injection attacks pose a significant risk. These attacks involve deliberately introducing faults into the system to expose sensitive information. BIKEs error-correction mechanisms are particularly susceptible to this form of attack. To counter this, it is important to introduce redundant checks during the decoding process. Verifying the decoded message multiple times before using it ensures that malicious faults are detected and mitigated. Another effective approach involves incorporating error-detection mechanisms, such as checksums, to identify and handle injected faults. If discrepancies are detected, the decryption process can abort, preventing the leakage of critical information.

Though BIKE is designed to resist quantum attacks, the evolving nature of quantum cryptanalysis, including attacks leveraging Grover's algorithm, still presents a potential threat. Strengthening resistance to quantum-based attacks can be achieved by adjusting the key size and parameters used in BIKE. Larger keys provide greater protection against quantum search algorithms by making brute-force attacks less feasible. Additionally, the use of hybrid schemes, which combine BIKE with other quantum-resistant KEMs, can further bolster security. This hybrid approach ensures that even if one scheme becomes vulnerable, the other can still provide protection.

Finally, additional considerations include regularly analyzing and hardening BIKEs parameters to keep up with evolving cryptanalytic techniques. This involves adjusting code lengths and bit-flipping thresholds in response to new attacks, ensuring that BIKE remains secure in the face of emerging threats. Equally important is ensuring that BIKE is implemented in secure environments. Protecting the hardware and using robust random number generators can mitigate many attack vectors that rely on exploiting weaknesses in implementation rather than the cryptographic scheme itself.

3.3 Cryptanalysis on HQC

The Hamming Quasi-Cyclic (HQC) Key Encapsulation Mechanism (KEM) is a post-quantum cryptographic scheme known for its theoretical robustness. However, key-recovery and message-recovery attacks have exposed significant vulnerabilities in its practical implementations. These attacks ex-

ploit flaws in how HQC handles decoding operations, timing information, and side-channel leakage, underscoring the importance of securing both the algorithm and the hardware implementations.

3.3.1 Key-recovery

Key-recovery attacks on HQC have highlighted critical weaknesses in its decoding process and side-channel resistance, especially when implemented on hardware platforms like microcontrollers. Below, we explore several key-recovery attacks that reveal the practical vulnerabilities of this cryptosystem.

The first key-recovery attack, described in [75], exploits timing variations during the BCH decoding step in HQC. The authors observed a correlation between the weight of the error vector and the time taken to decode using BCH codes. By exploiting this timing side-channel, attackers can infer information about the secret key with a high degree of accuracy. The attack requires 5441 decoding queries and has a success probability of 93%, making it highly efficient. This demonstrates the need for implementing constant-time operations in cryptographic routines to mitigate timing-based side channels.

In [76], a power side-channel attack is presented, marking the first such attack on HQC. The authors created an oracle using power consumption traces, which allowed them to determine whether an error was corrected during BCH decoding. Using a pattern matching approach based on the sum of squared differences, the attack requires fewer than 10000 measurements to recover most of the secret key. This attack highlights the vulnerability of HQC when implemented on embedded devices like microcontrollers, where power analysis can be used to extract sensitive information.

A third key-recovery attack is discussed in [77]. This attack exploits timing leaks during deterministic re-encryption in the decapsulation process. The rejection-sampling mechanism used in HQC leads to timing variations that can be observed by an attacker, allowing for the recovery of the secret key after around 866000 queries. Despite the high number of queries, this attack demonstrates that even advanced security mechanisms like deterministic re-encryption are not immune to timing biases, making it critical to address these issues in future implementations.

In summary, while HQC provides strong theoretical security, these key-recovery attacks reveal significant practical challenges. Effective countermeasures, such as constant-time implementations, masking techniques, and protection against power side-channel attacks, are crucial to securing HQC in real-world environments.

Table 6: Comparison of key-recovery attacks on the HQC KEM.

Article	Step (Algorithm)	Scenario	Attack Type	Impl. and Hardware
[75]	C.Decode($\mathbf{v}-\mathbf{u}\cdot\mathbf{y}$) (Decryption)	$\mathcal{O}(n^{5/2})$	Timing attack Full key recovery	Optimized impl.
[76]	C.Decode($\mathbf{v}-\mathbf{u}\cdot\mathbf{y}$) (Decryption)	<10000 measurements	Power side-channel Full key recovery	ARM Cortex-M4
[77]	Encrypt(pk, m', θ') (Decapsulation)	Around 866000 queries	Timing attack Full key recovery	Ref. impl.

3.3.2 Message-recovery

Message-recovery attacks exploit the vulnerabilities in the HQC decoding process, specifically targeting weaknesses in the RS (Reed-Solomon) decoder. These attacks often rely on side-channel informa-

tion, such as the Hamming Weight (HW) leakage model, which can reveal the contents of exchanged messages during the decapsulation phase.

In [83], the authors present the first horizontal Access Correlation Side-channel (ACS) attack against the HQC RMRS KEM. This attack targets the RS decoder and uses the HW leakage model to infer the message being exchanged. During the decapsulation process, each byte of the codeword is manipulated multiple times, allowing correlation analysis to be performed on the side-channel data. By analyzing the correlation between the manipulated bytes and their Hamming weights, the attacker can infer the message contents.

A key advantage for the attacker in this scenario is that the HQC decoder is public and can be used to correct erroneous side-channel inferences. While the decoder has limited error correction capability, the authors enhance the attack by using more sophisticated RS decoders, which improve the success rate of message recovery. The attack is first simulated using controlled noise to analyze the impact of signal-to-noise ratio (SNR) on the attack's success. The results from simulated traces are then compared with practical results, showing that a real-world attack is feasible with fewer than 296 Galois field operations.

In conclusion, this message-recovery attack highlights the importance of securing side-channel leakages, especially in decoding routines. Implementing noise-resilient decoders and incorporating hardware countermeasures, such as obfuscation techniques, can significantly reduce the risk of successful message-recovery attacks.

Table 7: Message-recovery attack on the HQC KEM.

Article	**Step (Algorithm)**	**Scenario**	**Attack Type**	**Impl. and Hardware**
[83]	RS decoding (Decapsulation)	Hamming Weight (HW)	ACS attack Message recovery	Simulated traces and practical impl.

3.3.3 Countermeasures

While HQC is theoretically strong, offering resistance to quantum attacks and leveraging hard decoding problems, practical implementations of the scheme remain vulnerable to various types of attacks. Key recovery and message recovery attacks, which target the weaknesses in the implementation rather than the cryptographic structure itself, are significant concerns. To ensure optimal security, appropriate countermeasures must be applied, especially focusing on side-channel attacks and decoding vulnerabilities.

Side-channel attacks, which exploit leaks such as execution time or power consumption during the decoding process, pose a major threat to HQC. To mitigate these attacks, special attention must be paid to the design of the decoding algorithms. It is advisable to implement constant-time algorithms, ensuring that the execution time remains consistent, regardless of the data being processed. This approach greatly reduces the amount of exploitable information available to an attacker. Moreover, masking techniques can be employed to minimize information leaks related to power consumption or electrical fluctuations. By masking intermediate calculations during decoding, these techniques make side-channel analyses significantly more complex for the attacker.

Fault injection attacks, where deliberate errors are introduced into the system to reveal information about the private key, are another type of threat that must be addressed. To counter these attacks, HQC implementations should incorporate fault detection mechanisms, such as consistency checks during decoding. For instance, redundancy checks on the decoded messages can help identify malicious alterations and respond accordingly, either by resetting the decryption process or by halting execution to prevent information leakage.

In addition to these measures, it is crucial to design hardware-level countermeasures. Side-channel attacks based on timing and power consumption, such as power analysis or electromagnetic fluctuation analysis, can be mitigated with hardware protections. For example, adding random noise to power consumption, or using dedicated circuits designed to obscure cryptographic operations, can significantly hinder the exploitation of these channels by attackers.

Finally, HQCs security also depends on the proper implementation of its decoding algorithms. Poorly designed decoding algorithms can introduce vulnerabilities that allow attackers to exploit errors during the decoding process to recover sensitive information. Therefore, it is essential to continually assess and improve decoding techniques to minimize potential information leaks, particularly by adopting approaches that make decoding less predictable and more resistant to attacks.

In conclusion, while HQC offers robust theoretical security, its practical security depends heavily on the implementation of effective countermeasures. By combining secure decoding algorithms, hardware protections against side-channel attacks, and fault detection mechanisms, HQC can provide enhanced resistance to cryptanalyses, ensuring reliable security in post-quantum environments.

4 Discussion

4.1 Synthesis of Key Findings

This section provides a synthesis of the cryptanalysis results on three major code-based Key Encapsulation Mechanisms (KEMs): Classic McEliece, BIKE , and HQC. These candidates have been extensively analyzed for post-quantum cryptographic systems, and this synthesis highlights the key-recovery and message-recovery attacks reported in the literature.

4.1.1 Classic McEliece

1. **Key-Recovery Attacks**:
   Several side-channel key-recovery attacks have been proposed against Classic McEliece, focusing on both decapsulation and key generation phases:

   - [9] demonstrated a significant key-recovery attack by observing the decapsulation of invalid ciphertexts, exploiting vulnerabilities in the process.
   - [5] introduced a partial key-recovery attack targeting side-channel leaks during the loading of private polynomial coefficients.
   - [24] published an attack during the key generation phase, simulating leakage from Gaussian elimination.
   - [18] proposed methods for full key recovery based on the presence of secret information "hints," although the practical feasibility of obtaining these hints remains uncertain.

2. **Message-Recovery Attacks**:
   Message-recovery attacks against Classic McEliece exploit physical vulnerabilities during decryption:

   - [10] leveraged syndrome enhancements combined with decoding oracle analysis to recover messages using Information Set Decoding (ISD).
   - [14] described a fault injection attack modifying matrix-vector multiplication, allowing message recovery by altering decryption operations.
   - [56] demonstrated a fault-based attack on the Fujisaki-Okamoto transformation, reducing the complexity of message recovery.

4.1.2 BIKE

1. **Key-Recovery Attacks**
   BIKE is vulnerable to key-recovery attacks due to decryption failures and timing side-channels:
   - [77] exploited timing information in rejection-sampling routines to reveal key information from the decapsulation process.
   - [74] combined power analysis with ISD to recover the full private key, demonstrating BIKE's susceptibility to side-channel attacks.
   - [78] presented attacks leveraging decryption failures to recover secret keys by analyzing error-handling mechanisms in BIKE s decoding process.
2. **Message-Recovery Attacks**
   Message-recovery attacks on BIKE target both hardware and algorithmic weaknesses: Timing attacks on weight verification in BIKE 's implementations enable attackers to infer message bits through time variation analysis [77].

4.1.3 HQC

1. **Key-Recovery Attacks**
   Key-recovery attacks on HQC target timing and side-channel vulnerabilities:
   - [75] demonstrated a timing attack exploiting correlations between error weight and execution time in BCH code decoding, recovering key information with high probability.
   - [76] introduced a power side-channel attack that used power consumption during BCH decoding to efficiently extract secret key information.
   - [77] revealed a timing leak in the rejection sampling routine during decapsulation, exposing HQC to key-recovery attacks.
2. **Message-Recovery Attacks**
   Message-recovery attacks on HQC focus on hardware and algorithmic flaws: [83] presented the first horizontal ACS (Access Correlation Side-channel) attack on the HQC RMRS KEM, recovering message information via Hamming weight leakage during decapsulation.

Each of these KEMs presents distinct cryptographic challenges, with key-recovery and message-recovery attacks exploiting various aspects of their respective decapsulation, key generation, or decoding processes. These results emphasize the need for continued research into countermeasures that can mitigate side-channel vulnerabilities and protect against both software and hardware attacks.

4.2 Implications for Future Research

The cryptanalysis results on Classic McEliece, BIKE , and HQC reveal significant vulnerabilities in both theoretical and practical implementations. These findings open avenues for research aimed at strengthening the security of code-based Key Encapsulation Mechanisms (KEM) in the post-quantum era.

1. **Strengthening Resistance to Side-Channel Attacks:** side-channel attacks, such as key-recovery attacks on Classic McEliece [9] or power analysis attacks on HQC [76], demonstrate recurring vulnerabilities in code-based cryptographic schemes. These attacks exploit information leaks during decryption or key generation processes.

   Future research must focus on developing and implementing effective countermeasures to mitigate these vulnerabilities, including:

- **Constant-time implementations:** to prevent leaks related to the execution of sensitive operations vulnerable to the attack proposed in [75].
- **Masking techniques:** designed to make sensitive information less accessible to physical and software-based attacks like in [76].
- **Operation obfuscation:** to obscure signals emitted during cryptographic processes, particularly on embedded devices vulnerable to side-channel attacks as with [77].

These measures could also be combined with hardware approaches to protect embedded and IoT devices.

2. **Improving Fault Injection Resilience:** fault injection attacks, such as those exposed in the work on Classic McEliece [14], reveal the ability of attackers to induce errors in computation to recover sensitive information. Future research should focus on:

   - **Fault detection:** utilizing redundancy or error-correcting codes to identify and correct faults in real-time to protect systems against attacks like in [44].
   - **Fault tolerance:** to protect cryptographic schemes from malicious disturbances during decryption and key generation operations as in [14].

   These techniques would aim to make fault injection attacks significantly more complex and costly.

3. **Exploring New Decoding Algorithms:** Cryptanalysis of schemes such as Classic McEliece and BIKE show that current decoding algorithms are often the targets of attacks as in [10, 78]. Future research could explore:

   - **Development of more resistant decoders:** to prevent attacks using timing or power consumption analysis such as in [75, 74].
   - **Securing decapsulation processes:** specifically by enhancing the robustness of decoders against attacks combining side-channel and Information Set Decoding (ISD) like in [77].

   This could include decoding algorithms that reduce leaks and minimize the impact of errors.

4. **Hybrid Approaches for Post-Quantum Security:** despite improvements, cryptanalysis reveals persistent vulnerabilities in code-based schemes. One potential solution lies in **hybrid schemes**, which combine code-based cryptography with other post-quantum techniques (such as lattice-based cryptography) to balance security and efficiency.

   Hybrid approaches could address the weaknesses of individual schemes, providing enhanced security against various types of attacks like in [78].

5. **Formalizing Security Models:** the cryptanalysis results suggest that current security models, often based on theoretical assumptions, need to be expanded to include implementation vulnerabilities, such as those identified in side-channel or fault injection attacks [77, 14].

   Future research must explore **extended security proofs** that account for these practical attacks, providing better assurance of the robustness of post-quantum schemes in real-world environments.

6. **Continuous Cryptanalysis of NIST Candidates:** ongoing cryptanalysis of candidates for the NIST standardization process, such as Classic McEliece, BIKE , and HQC, is crucial to ensure these schemes withstand new classical and post-quantum attacks. As demonstrated by the

vulnerabilities discovered in BIKE [77] and HQC [83], future research must focus on continuous improvement of these schemes before widespread adoption.

7. **New Directions in Attack Methodologies:** finally, attack methodologies are evolving rapidly, as evidenced by recent attacks combining power analysis and decoding techniques as in [74]. Future research should:

   - **Develop new attack strategies:** targeting previously unexplored weaknesses in post-quantum schemes.
   - **Leverage automation and machine learning:** to facilitate cryptanalysis, especially in the context of side-channel attacks like in [73].

   These approaches will provide a deeper understanding of the limitations of current schemes and help design more resilient systems against emerging threats.

5 Conclusion

5.1 Summary of Key Points

In this survey, we have conducted a comprehensive examination of code-based key encapsulation mechanisms (KEMs), focusing on **Classic McEliece**, **BIKE**, and **HQC** , three of the most prominent candidates in the ongoing NIST post-quantum standardization process. We have highlighted the theoretical security of these systems, grounded in the hardness of coding theory problems such as the Syndrome Decoding Problem (SDP) and its variants, which are believed to be resistant to quantum attacks.

Despite their strong theoretical underpinnings, several cryptanalytic studies have shown that these systems remain vulnerable in practical implementations. In the case of Classic McEliece, key-recovery and message-recovery attacks have demonstrated that information leakage during key generation or decoding can potentially compromise the system's security [83, 5]. Similarly, BIKE has been shown to be vulnerable to attacks that exploit decryption failures and timing leaks during the rejection sampling process [78]. Finally, HQC is susceptible to side-channel attacks, particularly those that leverage power consumption and timing leaks [76, 75]. These vulnerabilities highlight the importance of carefully analyzing and securing both theoretical constructs and their practical implementations.

5.2 Importance and Practical Applications

As quantum computing continues to advance, the need for secure cryptographic systems that can withstand quantum attacks becomes increasingly urgent. Code-based cryptosystems, such as the ones discussed in this survey, are crucial candidates for ensuring the confidentiality, integrity, and authenticity of digital communications in the post-quantum era. Their resistance to attacks from quantum computers, especially those leveraging Shors and Grovers algorithms, makes them essential for securing sensitive sectors like finance, defense, healthcare, and critical infrastructure.

Moreover, the resilience of these cryptosystems against quantum threats ensures long-term protection for data requiring extended confidentiality, such as government secrets and private records. However, as current cryptanalytic results demonstrate, the practical implementation of these systems presents several challenges. **Practical attacks**, such as side-channel and fault-injection attacks, underscore the need for robust countermeasures and careful system design. This requires a coordinated effort from both academia and industry to ensure that cryptographic solutions not only hold up against theoretical attacks but also withstand real-world adversarial scenarios.

5.3 Suggestions for Researchers

The findings presented in this work suggest several promising avenues for future research aimed at improving the security and performance of code-based KEMs. Given the importance of these cryptosystems in the post-quantum world, the following research directions are proposed:

- **Development of Robust Countermeasures**: One of the primary areas of focus for future research should be the development of countermeasures that mitigate the risks posed by side-channel and fault-injection attacks. Techniques such as *timing masking*, *random noise injection*, and *fault detection and correction* have shown promise in protecting cryptosystems from such attacks. These countermeasures must be carefully integrated into implementations without significantly impacting performance.
- **In-depth Study of Key Generation**: The key generation phase has been identified as a critical point of vulnerability. Attacks targeting this phase, such as those described in [24], reveal that even small information leaks can compromise security. Therefore, it is essential to enhance the robustness of key generation algorithms through techniques that minimize leakage while maintaining computational efficiency. Additionally, the randomness and unpredictability of key generation should be further analyzed to ensure resilience against both classical and quantum attacks.
- **Optimizing Performance Without Compromising Security**: Balancing security and efficiency is a central challenge in post-quantum cryptography. For instance, the BIKE KEM [77] has been shown to be vulnerable to attacks that exploit rejection sampling mechanisms. Future research should aim to optimize the decapsulation and verification routines of such systems to reduce their susceptibility to timing leaks or decryption failures. This could involve exploring new algorithmic approaches that streamline performance without weakening the cryptosystems security.
- **Deepening Attack Methodologies**: Researchers should continue to investigate advanced cryptanalytic techniques that combine multiple attack vectors, such as power consumption analysis, electromagnetic emissions, and fault injection, with sophisticated decoding strategies. Attacks combining these vectors, as explored in studies on BIKE [74], can expose new vulnerabilities and help researchers better anticipate future threats. A deeper understanding of these multi-vector attacks will lead to the development of more robust cryptographic defenses.
- **Post-Quantum Security in Constrained Environments**: As many practical applications of cryptography involve resource-constrained devices such as IoT, embedded systems, and mobile platforms, research should focus on adapting post-quantum algorithms like McEliece, BIKE, and HQC for such environments. Efficient implementations, reduced key sizes, and hardware-based optimizations are essential for ensuring that code-based cryptosystems can be deployed on a wide scale without sacrificing security.

By addressing these challenges, the cryptographic community can help ensure that code-based cryptosystems remain secure and practical in the face of emerging threats. These research directions will not only enhance the resilience of current systems but also lay the groundwork for cryptographic solutions capable of withstanding the evolving landscape of quantum computing and cryptanalysis.

5.4 Final Thoughts

The transition to a post-quantum world will require a paradigm shift in how cryptographic systems are designed, implemented, and evaluated. Code-based cryptography, with its long history and

proven resilience against quantum threats, offers a promising path forward. However, as this survey has shown, theoretical security alone is not sufficient. Practical implementations must be fortified against a broad range of attack vectors, particularly in real-world scenarios where side-channel and fault-injection attacks pose significant risks.

In conclusion, while much progress has been made in the development of secure code-based KEMs, continued research and innovation are necessary to ensure their long-term viability. By focusing on both the theoretical and practical aspects of cryptographic security, we can build a future where digital communication remains secure in the face of rapidly advancing quantum technologies.

References

[1] Elwyn R. Berlekamp, Robert J. McEliece andHenk C. A. van Tilborg, *On the inherent intractability of certain coding problems (Corresp.)*, Trans. Inf. Theory, 24(3) (1978), 384–386.

[2] Sidelnikov, VM and Shestakov, SO, *On insecurity of cryptosystems based on generalized Reed-Solomon codes*, Discrete Math. AppL, 2(4) (1992), 439–444.

[3] Wen Wang, Jakub Szefer and Ruben Niederhagen, *FPGA-Based Niederreiter Cryptosystem Using Binary Goppa Codes*,Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, (2018), 77–98.

[4] Tanja Lange and Rainer Steinwandt, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, Lecture Notes in Computer Science, 10786 (2018).

[5] Nadia El Mrabet, Luca De Feo and Sylvain Duquesne, *Progress in Cryptology - AFRICACRYPT 2023 - 14th International Conference on Cryptology in Africa, Sousse, Tunisia, July 19-21, 2023, Proceedings*, Lecture Notes in Computer Science, 14064 (2023).

[6] Brice Colombier, Vlad-Florin Dragoi, Pierre-Louis Cayrel and Vincent Grosso,*Profiled Side-Channel Attack on Cryptosystems Based on the Binary Syndrome Decoding Problem*, Trans. Inf. Forensics Secur, 17 (2022) 3407–3420.

[7] Vincent Grosso Pierre-Louis Cayrel, Brice Colombier and Vlad-Florin Dragoi,*Punctured Syndrome Decoding Problem - Efficient Side-Channel Attacks Against Classic McEliece*, Constructive Side-Channel Analysis and Secure Design - 14th International Workshop, COSADE 2023, Munich, Germany, April 3-4, 2023, Proceedings (2023) 170–192.

[8] Elif Bilge Kavun and Michael Pehl,*Constructive Side-Channel Analysis and Secure Design - 14th International Workshop, COSADE 2023, Munich, Germany, April 3-4, 2023, Proceedings*, Lecture Notes in Computer Science, 13979 (2023).

[9] Qian Guo Andreas Johansson and Thomas Johansson,*A Key-Recovery Side-Channel Attack on Classic McEliece Implementations*, Trans. Cryptogr. Hardw. Embed. Syst., 2022(4) (2022) 800–827.

[10] Norman Lahr and Ruben Niederhagen, Richard Petri and Simona Samardjiska,*Side Channel Information Set Decoding Using Iterative Chunking - Plaintext Recovery from the "Classic McEliece" Hardware Reference Implementation*, Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I, (2020) 881–910.

[11] Shiho Moriai and Huaxiong Wang,*Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, Lecture Notes in Computer Science, 12491 (2020).

[12] Sabine Pircher, Johannes Geier, Julian Danner, Daniel Mueller-Gritschneder and Antonia Wachter-Zeh,*Key-Recovery Fault Injection Attack on the Classic McEliece KEM*, Code-Based Cryptography - 10th International Workshop, CBCrypto 2022, Trondheim, Norway, May 29-30, 2022, Revised Selected Papers, (2022) 37–61.

[13] Jean-Christophe Deneuville,*Code-Based Cryptography - 10th International Workshop, CBCrypto 2022, Trondheim, Norway, May 29-30, 2022, Revised Selected Papers*, Lecture Notes in Computer Science, 13839 (2023).

[14] Pierre-Louis Cayrel, Brice Colombier, Vlad-Florin Dragoi, Alexandre Menu and Lilian Bossuet,*Message-Recovery Laser Fault Injection Attack on the Classic McEliece Cryptosystem*, Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part II, (2021) 438–467.

[15] Sebastian Bitzer, Jeroen Delvaux, Elena Kirshanova, Sebastian Maan, Alexander May and Antonia Wachter-Zeh,*How to Lose Some Weight - A Practical Template Syndrome Decoding Attack*, Cryptology ePrint Archive, Paper 2024/621, (2024).

[16] Sebastian Bitzer, Jeroen Delvaux, Elena Kirshanova, Sebastian Maan, Alexander May and Antonia Wachter-Zeh, *How to Lose Some Weight - A Practical Template Syndrome Decoding Attack*, Cryptology ePrint Archive, Paper 2024/621, (2024).

[17] Anne Canteaut and François-Xavier Standaert, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part II*, Lecture Notes in Computer Science, 12697 (2021).

[18] Elena Kirshanova and Alexander May, *Breaking Goppa-based McEliece with hints*, Inf. Comput., 293 (2023) 105045.

[19] Yuanxing Li, Robert H. Deng and Xinmei Wang, *On the equivalence of McEliece's and Niederreiter's public-key cryptosystems*, Trans. Inf. Theory, 40(1) (1994) 271–273.

[20] Colin O'Flynn and Zhizhang (David) Chen, *ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research*, Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers, (2014) 243–260.

[21] Emmanuel Prouff, *Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers*, Lecture Notes in Computer Science, 8622 (2014).

[22] Suresh Chari, Josyula R. Rao and Pankaj Rohatgi, *Template Attacks*, Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers, (2002) 13–28.

[23] Burton S. Kaliski Jr., Çetin Kaya Koç and Christof Paar, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, Lecture Notes in Computer Science, 2523 (2003).

[24] Marcus Brinkmann, Chitchanok Chuengsatiansup, Alexander May, Julian Nowakowski and Yuval Yarom, *Leaky McEliece: Secret Key Recovery From Highly Erroneous Side-Channel Information*, Cryptol. ePrint Arch., 1536 (2003).

[25] Anna-Lena Horlemann, Sven Puchinger, Julian Renner, Thomas Schamberger and Antonia Wachter-Zeh, *Information-Set Decoding with Hints*, Code-Based Cryptography - 9th International Workshop, CBCrypto 2021, Munich, Germany, June 21-22, 2021 Revised Selected Papers, (2021) 60–83.

[26] Antonia Wachter-Zeh, Hannes Bartz and Gianluigi Liva, *Code-Based Cryptography - 9th International Workshop, CBCrypto 2021, Munich, Germany, June 21-22, 2021 Revised Selected Papers*, Lecture Notes in Computer Science, 13150 (2022).

[27] Eugene Prange, *The use of information sets in decoding cyclic codes*, Trans. Inf. Theory, 8(5) (1962) 5–9.

[28] Jacques Stern, *A method for finding codewords of small weight*, Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings, (1998) 106–113.

[29] Gérard D. Cohen and Jacques Wolfmann, *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*, Lecture Notes in Computer Science, 388 (1989).

[30] Alexander Barg, *Complexity Issues in Coding Theory*, Electron. Colloquium Comput. Complex., TR97-046 (1997).

[31] Goppa, Valerii Denisovich, *A new class of linear correcting codes*, Problemy Peredachi Informatsii, 6(3) (1970).

[32] Bernstein, Daniel J., *Understanding binary-Goppa decoding*, IACR Communications in Cryptology, 1(1) (2024).

[33] Patterson, N., *The algebraic decoding of Goppa codes*, IEEE Transactions on Information Theory, 21(2) (1975) 203-207.

[34] Berlekamp, Elwyn R., *Algebraic Coding Theory - Revised Edition*, World Scientific Publishing Co., Inc., (2015).

[35] James L. Massey, *Shift-register synthesis and BCH decoding*, IEEE Trans. Inf. Theory, 1969(15) (1969) 122–127.

[36] Lothaire, M., *Algebraic Combinatorics on Words*Cambridge University Press (2002).

[37] Matthias J. Kannwischer, Peter Schwabe, Douglas Stebila and Thom Wiggers, *Improving Software Quality in Cryptography Standardization Projects* IEEE European Symposium on Security and Privacy, EuroS&P 2022 - Workshops, Genoa, Italy, June 6-10, (2002) 19–30.

[38] Sendrier, N., *Finding the permutation between equivalent linear codes: the support splitting algorithm* IEEE Transactions on Information Theory 46(4) (2000) 1193-1203.

[39] Shivam Bhasin, Jean-Luc Danger, Sylvain Guilley and Zakaria Najm, *Side-channel leakage and trace compression using normalized inter-class variance*Hardware and Architectural Support for Security and Privacy, Minneapolis, MN, USA, June 15, 2014 46(4) 1–7.

[40] Ruby B. Lee and Weidong Shi, *Hardware and Architectural Support for Security and Privacy, Minneapolis, MN, USA, June 15, 2014*, ACM (2014).

[41] Minjoo Sim, Hyeokdong Kwon, Siwoo Eum, Gyeongju Song, Minwoo Lee and Hwajeong Seo, *Efficient Implementation of the Classic McEliece on ARMv8 Processors*, Information Security Applications - 24th International Conference, WISA 2023, Jeju Island, South Korea, August 23-25, 2023, Revised Selected Papers, 14402 (2023) 324–337.

[42] Vatistas Kostalabros, Jordi Ribes-González, Oriol Farràs, Miquel Moretó and Carles Hernández, *HLS-Based HW/SW Co-Design of the Post-Quantum Classic McEliece Cryptosystem*, IEEE (2021) 52–59.

[43] Shaofen Chen, Haiyan Lin, Wenjin Huang and Yihua Huang, *Hardware Design and Implementation of Classic McEliece Post-Quantum Cryptosystem Based on FPGA*, High Performance Extreme Computing Conference, HPEC 2022, Waltham, (2022) 1–6.

[44] Julian Danner and Martin Kreuzer, *A fault attack on the Niederreiter cryptosystem using binary irreducible Goppa codes*, CoRR, abs/2002.01455 (2020).

[45] Johannes Roth, Evangelos G. Karatsiolis and Juliane Krämer, *Classic McEliece Implementation with Low Memory Footprint*, Smart Card Research and Advanced Applications - 19th International Conference, CARDIS 2020, Virtual Event, November 18-19, 2020, Revised Selected Papers, Lecture Notes in Computer Science 12609 (2020) 34–49.

[46] Mullen, Gary L. and Wan, Daqing and Wang, Qiang , *VALUE SETS OF POLYNOMIAL MAPS OVER FINITE FIELDS*,The Quarterly Journal of Mathematics, 64(4) (2012) 1191-1196.

[47] Reed, I. S. and Solomon, G., *Polynomial Codes Over Certain Finite Fields*, Journal of the Society for Industrial and Applied Mathematics, 8(2) (1960) 300-304.

[48] Stefan Heyse, Amir Moradi and Christof Paar, *Practical Power Analysis Attacks on Software Implementations of McEliece*, Third International Workshop on Post-Quantum Cryptography, 6061 (2010) 108–125.

[49] Cong Chen, Thomas Eisenbarth, Ingo von Maurich and Rainer Steinwandt, *Horizontal and Vertical Side Channel Analysis of a McEliece Cryptosystem*, Transactions on Information Forensics and Secuity, 11(6) (2016) 1093–1105.

[50] Roberto Avanzi, Simon Hoerder, Dan Page and Michael Tunstall, *Side-channel attacks on the McEliece and Niederreiter public-key cryptosystems*, Journal of Cryptograhic Engineering, 1(4) (2011) 271–281.

[51] H. Gregor Molter, Marc Stöttinger, Abdulhadi Shoufan and Falko Strenzke, *A simple power analysis attack on a McEliece cryptoprocessor*, Journal of Cryptographic Engineering, 1(1) (2011) 29–36.

[52] Leif Both and Alexander May, *Decoding Linear Codes with High Error Rate and Its Impact for LPN Security*, Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings, Lecture Notes in Computer Science, 10786 (2018) 25–46.

[53] Léo Ducas, Andre Esser, Simona Etinski and Elena Kirshanova, *Asymptotics and Improvements of Sieving for Codes*, Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI, 14656 (2024) 151–180.

[54] Anja Becker, Antoine Joux, Alexander May and Alexander Meurer, *Decoding Random Binary Linear Codes in 2 n/20: How 1 + 1 = 0 Improves Information Set Decoding*, Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings, Lecture Notes in Computer Science, 7237 (2012) 520–536.

[55] Eliya Nachmani and Lior Wolf, *Autoregressive Belief Propagation for Decoding Block Codes*, CoRR, abs/2103.11780 (2021).

[56] Keita Xagawa, Akira Ito, Rei Ueno, Junko Takahashi and Naofumi Homma, *Fault-Injection Attacks Against NIST's Post-Quantum Cryptography Round 3 KEM Candidates*, 13091 (2021) 33–61.

[57] Andre Esser and Alexander May and Floyd Zweydinger, *McEliece needs a Break Solving McEliece-1284 and Quasi-Cyclic-2918 with Modern ISD*, Cryptology ePrint Archive, Report 2021/1634 (2021).

[58] Daniel Augot and Matthieu Finiasz and Nicolas Sendrier, *Daniel Augot and Matthieu Finiasz and Nicolas Sendrier*, IACR Cryptol. ePrint Arch., 230 (2003).

[59] Harald Niederreiter, *Knapsack-type cryptosystems and algebraic coding theory*, Problems of Control and Information Theory 15, (1986) 159–166.

[60] Gaborit, Philippe, Lauradoux, Cedric and Sendrier, Nicolas, *SYND: a Fast Code-Based Stream Cipher with a Security Reduction*, Gaborit, Philippe and Lauradoux, Cedric and Sendrier, Nicolas, (2007) 186-190.

[61] Jacques Stern, *A New Identification Scheme Based on Syndrome Decoding*, Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings, 773 (1993) 13–21.

[62] Andre Esser, Alexander May and Floyd Zweydinger, *McEliece Needs a Break - Solving McEliece-1284 and Quasi-Cyclic-2918 with Modern ISD*, Lecture Notes in Computer Science, 13277 (2022) 433–457.

[63] Esser, Andre, Bellini, Emanuele, Hanaoka, Goichiro, Shikata, Junji and Watanabe, Yohei, *Syndrome Decoding Estimator*, Public-Key Cryptography – PKC, (2022) 112–141.

[64] Baldi, Marco and Barenghi, Alessandro and Chiaraluce, Franco and Pelosi, Gerardo and Santini, Paolo, *A Finite Regime Analysis of Information Set Decoding Algorithms*, Algorithms, 12(10) (2019) 1999-4893.

[65] Kirshanova, Elena and Laarhoven, Thijs, *Lower Bounds on Lattice Sieving and Information Set Decoding*, Springer-Verlag, (2021) 791–820.

[66] Jeffrey S. Leon, *A probabilistic algorithm for computing minimum weights of large error-correcting codes*, Trans. Inf. Theory, 34(5) (1988) 1354–1359.

[67] Donald E. Knuth, *The Art of Computer Programming, Volume II: Seminumerical Algorithms*, Addison-Wesley, (1969).

[68] Vlad-Florin Dragoi, Pierre-Louis Cayrel, Brice Colombier, Dominic Bucerzan and Sorin Hoara, *Solving a Modified Syndrome Decoding Problem using Integer Programming*, International Journal of Computers Communications & Control, 15(5) (2020).

[69] Robert Dorfman, *The Detection of Defective Members of Large Populations*, Annals of Mathematical Statistics, 14 (1943) 436-440.

[70] Kangwook Lee, Kabir Chandrasekher, Ramtin Pedarsani and Kannan Ramchandran, *The Detection of Defective Members of Large Populations*, Transactions on Signal Processing, 67(17) (2019) 4649–4664.

[71] Oliver Gebhard, Max Hahn-Klimroth, Dominik Kaaser and Philipp Loick, *Quantitative Group Testing in the Sublinear Regime*, arXiv. 1905.01458 (2019).

[72] Leo Breiman, *Random Forests*, Machine Learning, 45(1) (2001) 5–32.

[73] Anna-Lena Horlemann-Trautmann, Sven Puchinger, Julian Renner, Thomas Schamberger and Antonia Wachter-Zeh, *Information-Set Decoding with Hints*,Cryptology ePrint Archive, Report 2021/279 (2021).

[74] Cheriere, Agathe, Aragon, Nicolas, Richmond, Tania and Gérard, Benoît, *BIKE Key-Recovery: Combining Power Consumption Analysis and Information-Set Decoding*, Applied Cryptography and Network Security, (2023) 725–748.

[75] Guillaume Wafo-Tapa, Slim Bettaieb, Loic Bidoux, Philippe Gaborit and Etienne Marcatel, *A Practicable Timing Attack Against HQC and its Countermeasure*, Cryptology ePrint Archive, Paper 2019/909 (2019).

[76] Thomas Schamberger, Julian Renner, Georg Sigl and Antonia Wachter-Zeh, *A Power Side-Channel Attack on the CCA2-Secure HQC KEM*, Cryptology ePrint Archive, Paper 2020/910 (2020).

[77] Qian Guo, Clemens Hlauschek, Thomas Johansson, Norman Lahr, Alexander Nilsson and Robin Leander Schrder, *Don't Reject This: Key-Recovery Timing Attacks Due to Rejection-Sampling in HQC and BIKE*, Cryptology ePrint Archive, Paper 2021/1485 (2021).

[78] Tianrui Wang, Anyu Wang and Xiaoyun Wang, *Exploring Decryption Failures of BIKE: New Class of Weak Keys and Key Recovery Attacks*, Cryptology ePrint Archive, Paper 2023/659 (2023).

[79] Shor, Peter W., *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM Journal on Computing, 26(5) (1997) 1484-1509.

[80] McEliece, R. J., *A Public-Key Cryptosystem Based On Algebraic Coding Theory*, Deep Space Network Progress Report, 44 (1978) 114-116.

[81] Melchor, Carlos Aguilar, Aragon Nicolas, Bettaieb Slim, Bidoux Loıc, Blazy Olivier, Deneuville Jean-Christophe, Gaborit Philippe, Persichetti Edoardo, Zémor Gilles and Bourges IC, *Hamming quasi-cyclic (HQC)*, NIST PQC Round, 2(4) (2018).

[82] Aragon Nicolas, Barreto Paulo, Bettaieb Slim, Bidoux Loïc, Blazy Olivier, Deneuville Jean-Christophe, Gaborit Philippe, Ghosh Santosh, Gueron Shay, Güneysu Tim and others, *BIKE: bit flipping key encapsulation*, (2022).

[83] Goy Guillaume, Loiseau Antoine and Gaborit Phlippe, *Estimating the strength of horizontal correlation attacks in the hamming weight leakage model: A side-channel analysis on HQC KEM*, WCC 2022: The Twelfth International Workshop on Coding and Cryptography, (2022).

[84] Colombier Brice, Dragoi Vlad-Florin, Cayrel Pierre-Louis and Grosso Vincent, *Message-recovery Profiled Side-channel Attack on the Classic McEliece Cryptosystem*, IACR Cryptol. ePrint Arch., 2022 (2022).

[85] Colombier Brice, Grosso Vincent, Cayrel Pierre-Louis and Drăgoi Vlad-Florin, *Horizontal correlation attack on classic McEliece*, Cryptology ePrint Archive, (2023).

[86] Goy Guillaume, Maillard Julien, Gaborit Philippe and Loiseau Antoine, *IACR Transactions on Cryptographic Hardware and Embedded Systems*, IACR Transactions on Cryptographic Hardware and Embedded Systems, 2 (2024) 64–87.

[87] Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson and Wen Wang, *Classic McEliece*, National Institute of Standards and Technology (2022).