A Comparison of Post-Quantum Code-Based Cryptosytems

Lola Dekhuijzen[1], Kaitai Liang[1], and Huanhuan Chen[1]

[1]TU Delft

July 1, 2021

Abstract

The NIST PQC Standardization Process aims to find new cryptographic standards resistant to both classical and quantum computers. Several categories of cryptographic schemes are currently being evaluated by both NIST and the cryptographic community. Schemes are compared against one another with an emphasis on security, run-time performance, and bandwidth. A select few have made it to the third round and NIST hopes to standardize a subset of the schemes after the third round has come to an end. This research paper evaluates the encryption schemes of one of the five categories, code-based cryptography. Classic McEliece is a finalist and its security proof is strong, as its underlying mathematical concepts have been studied for over 40 years. Having said that, its large public key size makes it useful for only specific use cases. For general use, BIKE may be a better option, as its quasi-cyclic codes lead to a significant reduction in bandwidth. They will need to further analyze their decoding failure rate to ensure it is low enough to claim the required security level. Other metrics such as the rank-metric have also been considered. While they offer promising key sizes, further research into potential attacks is needed.

1 Introduction

Our current cryptosystems are based on the hardness of a few mathematical problems. One of them is the integer factorization problem. Quantum computers are evolving rapidly and once they reach a certain level of processing power, they will be able to solve this problem by running Shor's algorithm [1]. Therefore, we must come up with new methods of protecting our data. In 2016, The National Institute of Standards and Technology (NIST) began reviewing algorithms that may be resistant against attacks of quantum computers. This process is commonly referred to as the Post-Quantum Cryptography (PQC) Standardization Process. Various algorithms are being inspected during several thorough rounds and fifteen have made it to the third and current round. NIST expects to have the final results of their review by 2022. Out of the finalists, they will choose a select few for standardization. During this contest, several factors are taken into account [2]. The main focus during the evaluation process is security. The second most important factor is cost. This includes the computational cost of operations concerning key generation, public keys, and private keys. Furthermore, the transmission cost of information exchange and the implementation cost are taken into consideration.

This thesis aims to compare the code-based encryption schemes that are under review by NIST in their search for cryptosystems resistant against both classical and quantum computers. The schemes will be evaluated individually and compared against one another. What are the strengths and weaknesses of each scheme, and for what use cases will they be of help, if any? During this evaluation, the following aspects are considered:

1. claimed level of security.
2. computational complexity of operations, transmission costs, and implementation costs.
3. distinct features of the algorithms.
4. known attacks against the schemes, vulnerabilities of the schemes, and any possible countermeasures.

The paper is structured as follows. In section 2, a description of the research methodology is given. Section 3 analyzes the code-based encryption schemes. In section 4, the security of each scheme is analyzed, while section 5 focuses on bandwidth and runtime. Section 6 considers the ethical aspects of the research. In section 7, the results are discussed. Lastly, section 8 offers an conclusion and indicates what areas are open to further research.

2 Methodology

For each scheme, I will first read the specification of the algorithm. During this, I will focus on grasping the mathematical concepts of the algorithm. Once I have a good understanding of the algorithm, I will start evaluating it. I will read the NIST reports on the algorithm to gather how its security and cost are evaluated. I will also go over the papers that are available for the algorithm to get a more well-rounded view of each of these factors. Along with that, I will assess their performance. I will also read up on potential weaknesses of the algorithm and any existing attacks, and consider possible countermeasures.

Along the way, I will not only look at each algorithm in isolation but also combine the knowledge I obtain to make comparisons between the schemes. I will consider what use cases may be suited for each scheme and the advantages and disadvantages of choosing one scheme over the other. This will allow me to conclude which scheme has the most potential by the end of the project.

Beyond this initial comparison, I will investigate recent methods that these submissions may want to consider using, and offer an analysis on how I expect these alternate techniques can lead to an improvement of the submissions.

3 Code-Based Cryptosystems

Section 3.1 introduces McEliece, the original code-based cryptosystem. In section 3.2, a commonly used variant of this scheme, Niederreiter, is described. These sections describe the general concept of both cryptosystems and define the assumptions underlying their security. Then, in sections 3.3-3.8, code-based encryption schemes partaking in the PQC Standardization Process are described and evaluated.

3.1 McEliece

The first code-based cryptosystem was developed by and named after Robert McEliece in 1978 [3]. Code-based cryptography relies on the idea that decoding a random linear code is computationally hard, NP-complete to be exact [4].

A scenario that makes use of the McEliece cryptosystem is illustrated in Figure 1. Alice wants to securely send a message to Bob. She turns her message into a linear code that has an efficient decoding algorithm A. The linear code Alice decides to use has some generator matrix G which will extend the message M with some parity bits. After multiplying M by G, she will add some random noise in the form of an error vector e. Bob will be able to decrypt the message by using an efficient decoding algorithm. To ensure that Eve will not be able to apply this efficient decoding algorithm, the McEliece Cryptosystem disguises the code by multiplying G by an invertible matrix S and a permutation matrix P. When we multiply all three together, we obtain the public key G'. Eve does not know the factorization of G', so she will not be able to obtain G. The ciphertext, $M * G' + e$, will look like a random linear code to Eve. Bob on the other hand has the private key as well, which tells him how to undo the effects of the matrices S and P. Now, he can efficiently decode the code word to obtain the original message.

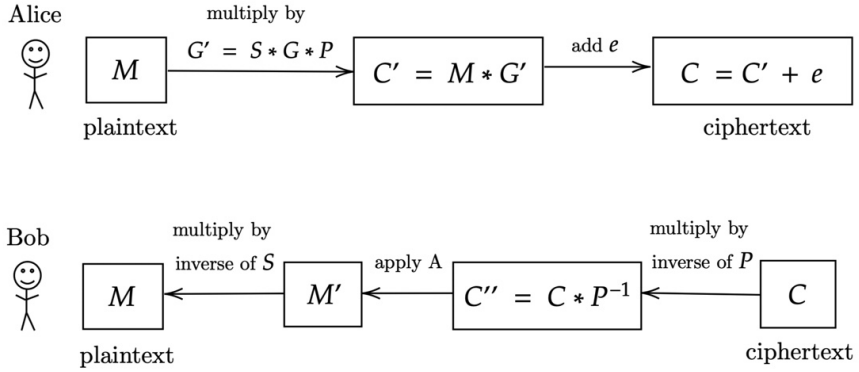


Figure 1: Alice wants to send a message to Bob using the McEliece cryptosystem. She chooses some linear code with efficient decoding algorithm A and generator matrix G.

3.1.1 General decoding problem

Let us formalize the problem an attacker intends to solve. As Equation (1) shows, the ciphertext is made up of the message multiplied by some generator matrix with an error vector added to it.

$$CT = M * G' + e = c + e \tag{1}$$

The goal of the general decoding problem is to find the nearest valid code word to the vector we have been given. Once we have obtained the code word, we decode it to get the original message.

3.2 Niederreiter

The Niederreiter cryptosystem is a variation of the McEliece cryptosystem. While the original McEliece cryptosystem is built upon the general decoding problem, the Niederreiter cryptosystem makes use of the equivalent syndrome decoding problem as shown in section 3.2.1. The problem with McEliece is that the used generator matrix is in systematic form, and therefore consists of the actual message. This makes it rather easy to correct the errors of a meaningful message [5]. In the Niederreiter cryptosystem, we do not have this problem. Our ciphertext consists of a message multiplied by a scrambled parity check matrix.

3.2.1 Syndrome decoding problem

A different way of looking at the decoding problem is to view it as a syndrome decoding problem. We take our McEliece ciphertext, consisting of a code word and an error vector, and multiply this by the parity check matrix.

$$H * (c + e) = 0 + H * e \tag{2}$$

It can be seen from Equation (2) that this results in the term $H * e$, commonly referred to as the syndrome. The goal of the syndrome decoding problem is to extract the error vector from this $H * e$ term when we know H [6]. The general decoding problem and the syndrome decoding problem are equivalent and have been proven to be NP-complete [4].

3.2.2 Security

The McEliece and the Niederreiter cryptosystem are equivalent and secure under the following two assumptions.

1. Finding the nearest valid code word to some random vector we have been given is hard when dealing with a random linear code.

2. If one does not know the private key, finding the nearest valid code word to some vector is just as hard as decoding a random linear code.

The first assumption coincides with the aforementioned syndrome decoding problem and is thus known to be NP-complete. The second assumption depends on the family of codes that is being used. Some codes are known to be insecure, whereas others have resisted any attack so far.

3.3 Classic McEliece

The Classic McEliece cryptosystem uses the Niederreiter cryptosystem, so the ciphertext will consist of a codeword with an additional error vector. A binary Goppa code is a specific error-correcting code with some helpful properties [7]. We take a Goppa code and a random support. A support is a sequence of elements present in F_2^m. This time around, our secret key will not need a matrix S or P. We can simply apply a permutation to our support instead, so we no longer need a permutation matrix P. When we apply a matrix S to our matrix and then bring it to systematic form, we are essentially doing two consecutive matrix multiplications. Therefore, bringing our matrix into systematic form makes the matrix S redundant. A matrix is in the systematic form if its left side is the identity, so we only need to save its right side now.

To find out how many errors our code can correct, we consider the minimal distance between any two code words. For the binary Goppa code, the minimal distance is greater or equal than $2t + 1$, where t is the number of errors. Therefore, the code can correct up to at least $((2t + 1) - 1)/2$ errors [8]. We ensure that the weight of the error vector equals the number of errors that the error-correcting code is capable of correcting so that the cryptosystem is decoding failure-free by design. Decoding a binary square-free Goppa code can be done with Patterson's algorithm. This algorithm makes use of an error locator polynomial since the roots of this polynomial indicate where errors have occurred [9]. For each position of the error vector, we replace x with the syndrome of that position. If the equation now yields 0, we know that this position of the error vector was equal to 1.

3.3.1 Security

Classic McEliece claims to be secure against Chosen Plain-text attacks (CPAs) on basis of the hardness of decoding a general linear code, under the assumption that one cannot distinguish a random binary Goppa code from a random linear code. To prove its resistance against Chosen Cipher-text attacks (CCAs), Classic McEliece applies the widely known implicit rejection version of the Fujisaki-Okamoto (FO) transform, as introduced by Hofheinz [10]. This transform converts a CPA-secure system to a CCA-secure system based on a few key assumptions. Most importantly, a low decoding failure rate (DFR) is required. As mentioned earlier, the binary Goppa codes are decoding failure-free by design.

3.3.2 Discussion

One of the advantages of Classic McEliece is its performance. Encapsulation and decapsulation, which consists of decoding, are particularly efficient. This is mainly due to the fact that operations are between simple objects such as binary vectors and matrices. They do have the smallest ciphertext out of all NIST submissions, the ciphertext is made up of code word and 32 bytes of confirmation. It also has some drawbacks, the most obvious one being the large public key, as it is above a million bytes. The exact numbers are shown in Table 1. These consist of entire parity-check matrices and are of size $n \times k$ where k is the length of the message and n is the length of the codeword [5].

public key	private key	ciphertext
1357824 bytes	14120 bytes	240 bytes

Table 1: bandwidth of Classic McEliece.

Due to its large public key size, Classic McEliece is not suitable for widespread use in internet protocols. However, some applications may benefit from the small ciphertext. One example is WireGuard, which is mainly concerned with the size of ciphertexts, as the Post-quantum WireGuard handshake does not exchange public keys [11].

3.4 BIKE

BIKE is based on the Niederreiter cryptosystem. Whereas Classic McEliece uses binary Goppa codes, BIKE makes use of QC-MDPC codes. The main drawback of Classic McEliece is its large keys. Furthermore, the bigger the key size gets, the harder it becomes to decode the used binary Goppa code [12]. BIKE avoids this by replacing the Goppa codes with

moderate-density parity-check (MDPC) codes [13]. Moderate density implies a parity-check matrix that is sparser than those of the classical high-density parity-check codes, as it has a density of $\mathcal{O}(\sqrt{n})$ [13]. On the other hand, these codes still manage to have a large minimum distance in comparison to low-density parity-check codes. MDPC codes have a slightly smaller error-correcting capability than the similar LDPC codes, as used in the similar submission named LEDAcrypt. This is due to the nature of the matrices, which are slightly denser. However, the number of errors that the MDPC codes can correct still ensures an adequate security level. On top of that, LDPC codes are vulnerable to an attack that makes use of the fact that the dual of the secret code has very low eight codewords and an opponent can look for them and discover the secret parity-check matrix, so MDPC codes have an advantage over LDPC codes.

3.4.1 Security

Since BIKE and Classic McEliece both build upon the Niederreiter cryptosystem, and thus are based on the same hard problem of decoding a random linear code, BIKE is able to prove CPA-security in a similar manner. However, since BIKE makes use of a different family of codes, it does not make use of the same decoding algorithm as Classic McEliece. Unlike Classic McEliece, the decoder of BIKE does not have a DFR of zero by design. Instead, they aim to keep the DFR sufficiently low, that is $2^{-\lambda}$, where λ is the security level. Hofheinz' FO transform requires the DFR to be sufficiently low. The decoding algorithm used in the decapsulation phase of BIKE is the Black-Gray (BG) decoder. It offers a low decryption failure rate, to keep an adversary from finding out what the secret key is by considering these failures. Besides that, it has a fixed number of steps to be able to run the algorithm in constant time [14]. The most efficient variant of the BG decoder is the Black-Gray-Flip (BGF) decoder [15], which consists of a single BG iteration followed by multiple Bit-Flipping iterations. However, in Hofheinz' FO transform, DFR is defined as the maximum over all possible plaintexts and averaged over all the key pairs, whereas when analyzing Niederreiter systems with MDPC codes, DFR is defined as an average over all possible plaintexts, averaged over all key pairs. BIKE attempts to solve the mismatch by means of simulation and extrapolation. However, simulation is hard since the desired DFR is extremely low. Besides that, extrapolation is not reliable since the curve slope may change due to the occurrence of an error floor, a region in the curve where performance suddenly drops as a result of near-codewords. Since this method is not precise enough, BIKE in the end decided against claiming CCA-security formally.

3.4.2 Discussion

The biggest advantage of BIKE over Classic McEliece is its improved bandwidth, as shown in Table 2, since quasi-cyclic codes allow for much more efficient data transfer. In comparison to lattice-based cryptosystems, BIKE still has a slightly larger bandwidth. Nonetheless, out of all code-based submissions, BIKE appears to be the most promising candidate for general use.

public key	private key	ciphertext
5122 bytes	580 bytes	5154 bytes

Table 2: bandwidth of BIKE.

The main disadvantage is the fact that BIKE has not managed to claim CCA-security yet. There are several solutions BIKE may want to consider in order to be more certain about their CCA-security proof.

LEDAcrypt, which will be discussed in section 3.5, runs into the same problem as BIKE in its attempt to claim CCA-security. LEDAcrypt has managed to solve this problem, so continuing forward, BIKE may want to take some inspiration from LEDAcrypt to secure the system. The drawback of this method is an increase in bandwidth.

A recent paper [12] introduces an improved version of the BF decoder BIKE originally used. This decoder introduces a randomization of the order in which we process estimated error positions. As a result, it offers a worst-case estimate of the DFR that does not rely on any assumption surrounding the error floor of the DFR curve. This method claims a twenty-five percent reduction of public key and ciphertext size compared to the decoders BIKE uses [12].

3.5 LEDAcrypt

LEDAcrypt is very similar to BIKE, as its codes are quasi-cyclic too, in order to decrease the bandwidth. LEDAcrypt makes use of LDPC codes, which slightly differ from the MDPC codes that BIKE uses. The extremely low weight of the low-density codes makes it so that its error-correcting capability is $\mathcal{O}(n)$, a bit larger than that of moderate-density codes. This may seem like an advantage, but to reach a sufficient level of security, a capability this high is not strictly necessary. The LDPC codes even have a disadvantage, as they are vulnerable to an attack that makes use of the fact that the dual of the secret code has very low weight codewords and an opponent can look for them and discover the secret parity-check matrix.

3.5.1 Security

Similar to BIKE, LEDAcrypt can easily claim CPA-security but not CCA-security, due to the difference in the definition of the DFR as used in the Hofheinz' FO transform compared to how it is defined in an analysis of any Niederreiter-like system. As stated in section 3.4.1, the first talks of a maximum DFR, while the latter considers an average DFR. LEDAcrypt chooses another approach than BIKE. They resolve the mismatch by introducing a mechanism that prevents the adversary from choosing a selection of input messages that cause a DFR different from and possibly higher than the average. As a result, the average DFR will now coincide with the maximum DFR. This allows them to confidently claim CCA-security, in contrast to BIKE.

3.5.2 Discussion

The quasi-cyclic nature of its codes allows LEDAcrypt to obtain a favorable bandwidth over Classic McEliece, but its bandwidth does not weigh up against that of BIKE. While LEDAcrypt has a stronger CCA-security proof than BIKE, this does come with an increase in bandwidth of about 60 percent. The exact numbers are shown in Table 3.

public key	private key	ciphertext
4424 bytes	2232 bytes	4464 bytes

Table 3: bandwidth of LEDAcrypt.

There may be alternative ways to obtain a CCA-secure proof. One approach would be to try and solve the error floor that occurs when making use of simulation and extrapolation. A recent paper by Karimi and Banihashemi explores a method that aims to lower the error floor in the DFR curve by introducing renewed code constructions [13]. The idea is to eliminate the trapping sets after detecting them with an efficient search algorithm. The occurrence of these sets can cause a higher error floor, since they may consist of check nodes with a low degree [14]. While lowering the error floor may lead to a CCA-secure system without the increased bandwidth, this may not be enough for LEDAcrypt to compete with BIKE.

Moreover, a recent paper [15] claims to have achieved a practical break of LEDAcrypt. They show that due to the existence of weak keys, they can improve upon ISD-based attacks as described in section 3.2.3. After the second round, LEDAcrypt attempted to modify their system to prevent the attack, but NIST decided that the resulting scheme resembled BIKE too closely. Therefore, LEDAcrypt was not selected for the third round.

3.6 HQC

A possible drawback of cryptosystems like Classic McEliece is their reliance on how well the structure of the used family of codes is hidden. Up to this point, cryptosystems using Goppa codes, Classic McEliece for instance, have been able to resist structural attacks. However, many other families of codes have been broken. BIKE approaches this issue by opting for a family of codes that rely on a significantly weaker hidden structure [16], even if it does not remove the hidden structure altogether.

A construct that removes the hidden structure of codes altogether was introduced by Alekhnovich in 2003 [17]. The idea is to make use of a random codeword of a random code, and have the private key be a random error vector that is added to the codeword. As a result, recovering the secret key is equivalent to decoding a random linear code, and there is no hidden structure to be exploited.

Both RQC, discussed in section 3.7, and HQC are built upon this concept. HQC makes use of the Hamming metric, similar to schemes such as BIKE, and utilizes two independent codes. The first one is a random quasi-cyclic code is used to secure the message that is sent. Therefore, uncovering the message would require an attacker to decode a random code, which we know is NP-complete. The rightful recipient can obtain a noisy version of the message by applying the private key. Then, a code of which the generator matrix is known to the public can be used to obtain the message without the added noise. The public code that HQC uses is a concatenated code. Its internal and external codes are a Reed-Muller and Reed-Solomon (RMRS) code respectively.

3.6.1 Security

Similar to other code-based schemes, HQC is based on the syndrome decoding problem, which allows them to claim CPA-security. The DFR of HQC corresponds to that of the concatenated code. Not having to hide the structure of this code allows HQC to opt for codes with efficient decoding and an easily analyzable DFR. The Reed-Solomon code is a bounded-distance decoder and its DFR has a simple upper bound. The DFR of the Reed-Muller can be estimated by taking its maximum likelihood [18]. Putting these two together results in a sufficiently low DFR, so HQC is able to apply Hofheinz' FO transform to claim CCA-security.

3.6.2 Discussion

One of the codes that HQC uses is quasi-cyclic, similar to BIKE and LEDAcrypt, which reduces key sizes somewhat. The other code that HQC makes use of is public which allows them to opt for codes such as Reed-Muller and Reed-Solomon that hard to hide but offer efficient decoding [19]. Sizes are given in Table 4.

public key	private key	ciphertext
7245 bytes	40 bytes	14469 bytes

Table 4: bandwidth of HQC.

A setback is its low encryption rate. Whilst its decoding is efficient, HQC is not able to compete with the performance of lattice-based schemes and its bandwidth does not compete with that of BIKE.

3.7 RQC

Similar to HQC and unlike most other code-based schemes, RQC is not based on how well some structure of a code is hidden. An additional feature of RQC is that it utilizes the rank metric rather than the Hamming metric [20].

Using the rank metric allows for a reduction in key sizes. This is due to the fact that the algorithm that aims to solve the syndrome decoding problem for the Hamming metric, information set decoding, has become fairly efficient. This has forced schemes to adjust their parameters to adhere to security requirements set by NIST. Algorithms that aim at solving the syndrome decoding problem for the rank metric, on the other hand, do not reach this level of efficiency, allowing RQC to obtain a lower bandwidth than HQC.

3.7.1 Security

For the rank metric codes, the codes that we use are the Gabidulin codes. A Gabidulin code of length n and dimension k can correct $(n-k)/2$ errors. By choosing an error vector of which the weight does not exceed the error-correcting capability of the code, we ensure that the scheme is free of any decoding failures. As a result, RQC can apply the Hofheinz' FO transform to obtain CCA-security without any difficulties.

3.7.2 Discussion

While the rank metric allows for smaller key sizes, shown in Table 5, they have been introduced to the field of code-based cryptography rather recently [21]. On the other hand, algorithms that aim to solve the rank metric version of the syndrome decoding problem have been studied extensively since then [22], suggesting that they are indeed a preferable option in comparison to the Hamming metric.

public key	private key	ciphertext
4090 bytes	40 bytes	8164 bytes

Table 5: bandwidth of RQC.

3.8 Rollo-II

Another submission that makes use of the rank metric is Rollo-II. In contrast to RQC, the security of Rollo-II does depend on the structure of some code being hidden. Rollo-II makes use of low-rank parity-check (LRPC) codes, the Hamming metric equivalent would be the LDPC codes that LEDAcrypt uses. LRPC codes have a weaker structure than the Gabidulin codes that RQC uses, making it possible to effectively hide their structure.

3.8.1 Security

In a similar fashion to other code-based schemes, Rollo-II claims CPA-security based on the hardness of solving the syndrome decoding problem. While Rollo-II is not decoding failure-free, they have introduced an efficient decoding algorithm with a well-analyzed DFR [24], allowing them to obtain CCA-security through Hofheinz' FO transform.

3.8.2 Discussion

Table 6 shows how out of all code-based schemes, Rollo-II has the most promising bandwidth. They make use of quasi-cyclic codes, and the rank-metric allows them to opt for smaller parameters.

public key	private key	ciphertext
2559 bytes	40 bytes	2687 bytes

Table 6: bandwidth of Rollo-II.

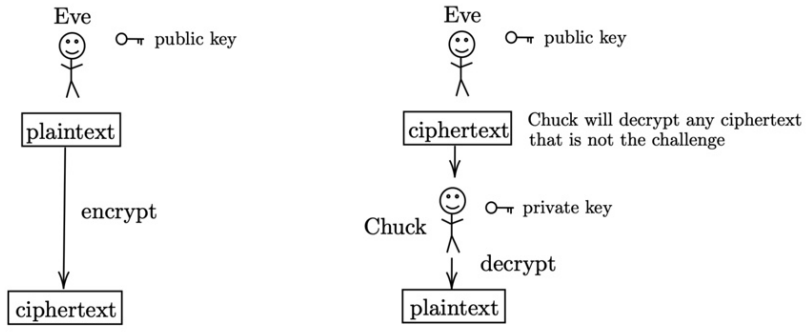
This does pose the question of whether the parameters of Rollo-II are favorable over that of some Hamming metric schemes, due to the fact that rank-metric has not been researched as thoroughly. Despite its improvements succeeding recent rank-metric attacks [25], NIST decided against choosing Rollo-II as a finalist. Nonetheless, NIST encourages the cryptographic community to further investigate the rank-metric.

4 Security Analysis

In this section, the security of the code-based submissions will be analyzed and compared. NIST has defined five different security strength categories for the contest. In this section and the next, we consider parameters aimed at the highest security level which corresponds to AES-256. In section 4.1, we will define the two attack models that NIST considers to be relevant. Section 4.2 looks at how the Decoding Failure Rate determines whether schemes are able to resist the aforementioned attacks. In section 4.3, the Information Set Decoding attack and its impact on different kinds of code-based schemes are considered.

4.1 Attack Models

NIST has specified two different security definitions. Both of these definitions require the property of indistinguishability (IND). An attacker, we will call her Eve, receives two plaintexts. One of these is encrypted, we will call this the challenge ciphertext. It is Eve's job to figure out which of the plaintexts belongs to the challenge.



(a) Chosen Plaintext Attack (b) Chosen Ciphertext Attack

Figure 2: Eve wants to find out which out of two plaintexts corresponds to a challenge ciphertext.

In the model of IND-CPA security, illustrated in Figure 2a, Eve is able to encrypt any arbitrary plaintext to help her figure out which of the two given plaintexts belongs to the challenge ciphertext. Note that encryption cannot be deterministic, since otherwise Eve can encrypt both given plaintexts right away and compare to the challenge ciphertext.

In the CCA-security model, Eve has the additional ability to send any ciphertext that is not the challenge ciphertext to Chuck, who possesses the private key and will return the corresponding plaintext to Eve. This is illustrated in Figure 2b.

4.2 Decoding Failure Rate

Since all code-based submissions are based on some equivalent variant of the syndrome decoding problem as described in section 3.2.1, they are all CPA-secure. Moreover, most of them, BIKE being the exception, have been able to prove their DFR is sufficiently low. This has allows them to apply the Fujisaki-Okamoto (FO) transform and consequently obtain a CCA-secure system.

Some of the schemes are free of decoding failures, such as Classic McEliece and RQC, while others have been able to prove their DFR is sufficiently low. The exact numbers are shown in Table 6. This has allowed them to apply the FO transform and consequently obtain a CCA-secure system. As discussed in sections 3.4.1 and 3.4.2, while BIKE is a strong contestant, its security proof is lacking which makes BIKE the outlier in this aspect.

scheme	DFR
Classic McEliece	zero
BIKE	about 2^{-256}
LEDAcrypt	2^{-256}
HQC	2^{-256}
RQC	zero
Rollo-II	2^{-136}

Table 7: comparison of decoding failure rates (probability of a decoding failure occurring).

4.3 Information Set Decoding

To obtain an upper bound on the security of code-based cryptosystems, we look at Information Set Decoding (ISD). This is a non-structural attack strategy, that is, it does not exploit the structure of the family of codes used, against code-based cryptosystems [26], and is known to be more effective than any structural attacks. ISD was originally introduced by Prange [27]. The key concept of Prange's algorithm is to find a subset of positions of a codeword that does not have any errors, we call this the information set. This set is then used to solve the decoding problem. This is assumed to be easier than finding all error positions, and thus the algorithm offers an improvement over a brute force attack. An improvement on this algorithm was introduced by Stern [28] and allows some errors to occur in the information set, which leads to an increased probability of success compared to Prange's algorithm.

In recent years, further improvements on the ISD attack have been introduced by May, Meurer, and Thomae [31], by Becker, Joux, May, and Meurer [32], and most recently by May and Ozerov [33]. Equation (3) represents the complexity of these algorithms, considering the decoding of a binary linear code of length n and dimension k with an error-correcting capability of w. Note that c is dependent on the specific scheme that is under attack and determined by its code rate k/n and error rate w/n.

$$2^{cw(1+o(1))} \qquad as\ n \longrightarrow \infty \tag{3}$$

In practice, c does not differ a lot from one scheme to another. The original McEliece parameters were broken by an ISD attack made up of $2^{266.94}$ bit operations [34]. Classic McEliece claims that while recent ISD attacks have a better performance, these improvements will not be of use in practice as additional operational costs were not taken into account [35].

Moreover, a recent paper [36] investigates the gain of performance in an ISD attack in the case of quasi-cyclic codes. As BIKE states in [37], an ISD attack can exploit the quasi-cyclic structure, since we can consider shifted versions of the same codeword to be equivalent and thus reduce the number of instances to be considered. The paper proves that while recent ISD attacks have an increase in performance, this improvement becomes negligible when the code rate is low. This implies that schemes with quasi-cyclic codes, such as BIKE and LEDAcrypt, have a clear advantage over Classic McEliece with its error-correcting capability of $\mathcal{O}(n/\log n)$.

5 Performance Analysis

In this section, different aspects of the performance are considered, again considering 256-bit security. Section 5.1 compares claimed bandwidth of the schemes. In section 5.2, the schemes are measured by their runtime.

5.1 Bandwidth Comparison

Table 8 gives an overview of the claimed bandwidth of each scheme. It shows sizes of the public key, private key, and ciphertext in bytes.

scheme	public key	private key	ciphertext
Classic McEliece	1357824	14120	240
BIKE	5122	580	5154
LEDAcrypt	4424	2232	4464
HQC	7245	40	14469
RQC	4090	40	8164
Rollo-II	2559	40	2687

Table 8: comparison of bandwidth in bytes.

5.2 Runtime Comparison

Each of code-based NIST submissions analyzed in this paper is a key encapsulation mechanism (KEM). Whilst a public key encryption (PKE) scheme makes use of a public and private key to securely transfer a message, KEMs use this mechanism to exchange a shared key which can then be used to interchange information. This process of exchanging a shared key consists of three parts; key generation creates a public and private key, encapsulation takes the public key and outputs a session key and a ciphertext, and decapsulation, in turn, takes this ciphertext to output the same session key. Figure 3 provides a visual representation of this process.

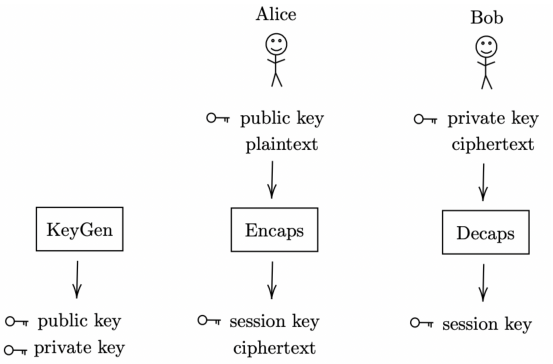


Figure 3: Components of a key encapsulation mechanism (KEM). After the generation of a public and private key, Alice generates a session key to then encrypt and send it to Bob. Bob in turn decrypts the message to obtain the session key.

scheme	machine	keygen	encaps	decaps
Classic McEliece	Intel Xeon E3-1275 3.50GHz	514 489 441	178 093	326 531
BIKE[1]	Intel Core i7-1065G7 1.30GHz	1 780 000	465 000	6 610 000
LEDAcrypt	Intel Core i5-6600 3.2 GHz	34 592 000	1 919 300	15 640 700
HQC	Intel Core i7-7820X CPU 3.6GHz	423 000	738 000	1 286 000
RQC	Intel Core i7-7820X CPU 3.6GHz	2 860 000	5 270 000	36 390 000
Rollo-II	Intel Core i7-7820X CPU 3.6GHz	22 694 000	1 643 000	4 260 000

Table 9: comparison of performance in cycles. [1]performance for the 256-bit security level has not been provided, presented results are for the 192-bit security level.

Table 9 measures the performance of the schemes in terms of CPU cycles. It can be noted that while using binary Goppa codes allows for fast encapsulation and decapsulation, the key generation of Classic McEliece performs poorly. Schemes that make use of quasi-cyclic codes or a generalization thereof are able to speed up this process.

6 Responsible Research

As Computer Science students, we are not part of the cryptographic community and are not involved with NIST. This allows us to view the submissions objectively and without any bias towards any particular submission.

The code-based cryptosystems in this paper are submissions of the NIST PQC Standardization Process. An important element of this process is the cryptanalysis of both NIST and the cryptographic community. Thus, reproducibility plays an important role in this process. Therefore, each of the submissions has published its source code, together with an extensive specification document.

Another thing to consider is the bias submitters may have towards their own submission. We should encourage them to be critical of their own work and to be transparent about potential shortcomings. Other than that, letting the cryptographic community evaluate the algorithms is also an effective way to tackle this issue, as people who are not involved in the making of some cryptosystem will not have a bias towards it.

7 Discussion

This section further discusses the security and performance analysis of sections 4 and 5. Section 7.1 will look at security. In section 7.2, the bandwidth of the schemes is discussed. Section 7.3 considers the run-time of the schemes.

7.1 Security

NIST considers Classic McEliece to be the conservative choice due to its well-studied security. While the hidden structure of the binary Goppa codes is strong, the Niederreiter-like cryptosystem in combination with this family of codes has been studied extensively and no known structural attacks have managed to break it. Most other code-based schemes succeed in claiming the same level of security as Classic McEliece by closely analyzing the DFR.

BIKE and Rollo-II both make use of codes with a weaker structure than that of binary Goppa codes. HQC and RQC remove the hidden structure altogether but pay for this in terms of bandwidth. While it is true that the lack of a hidden structure allows us to generalize the security proof of a scheme to any family of codes, this does necessarily mean that we end up with a more efficient scheme. Having to hide the structure of the used code means families of codes need to be studied carefully before deciding on which one can be applied to some scheme, but once this is done, the result can be favorable over schemes such as HQC and RQC.

7.2 Bandwidth

Classic McEliece has a significantly larger public key size than its competitors due to the strong structure of its codes. While this is not practical in most use cases, it may be advantageous in specific cases that can benefit from its remarkably small ciphertexts.

Other code-based schemes offer a better overall bandwidth by making use of quasi-cyclic codes. In their attempt to avoid relying on a hidden structure, HQC and RQC obtain a bandwidth higher than that of BIKE and LEDAcrypt. Rank-metric still has a slight advantage over the Hamming metric, as its attacks are not as effective. Overall, when it comes to bandwidth, we can conclude that using low to medium density parity-check codes in a Niederreiter-like scheme is preferable.

7.3 Runtime

Classic McEliece performs well when it comes to encapsulation and decapsulation, due to the simple nature of the objects and operations. These consist of binary vectors and matrix-vector multiplications. Its key generation algorithm, however, takes longer than that of any other code-based scheme. Other schemes can improve upon this due to the quasi-cyclic nature of their codes [29].

8 Conclusions and Future Work

This thesis evaluates the code-based encryption schemes that take part in the NIST Post-Quantum Standardization contest. Out of the code-based schemes, Classic McEliece is the only one that is considered as a finalist and thus closest to being standardized by NIST. This is mainly due to its security which has remained stable over the past 40 years. However, its large public key size will not be suitable for most use cases. A possible improvement on the binary Goppa codes that Classic McEliece uses is a construction that makes use of Generalized Srivastava (GS) codes instead of Goppa ones [38]. These are a generalization of Goppa codes and possess an equally efficient decoding algorithm, but their dyadic matrices significantly reduce the public key size. While this improvement would increase the practicality of Classic McEliece, its security analysis requires more work.

Quasi-cyclic codes offer several benefits over binary Goppa codes. Two of the code-based schemes with quasi-cyclic codes are being considered as alternate candidates for the final round, these are BIKE and HQC. BIKE is more suitable for general use as opposed to Classic McEliece, since its overall bandwidth is significantly lower. The main reason for choosing HQC is its thorough security analysis, something which BIKE currently lacks. If BIKE improves its security proof, it may have the potential to be standardized in the future.

The rank metric effectively reduces the key sizes in comparison to the Hamming metric. Rollo-II, for instance, has the lowest bandwidth out of all code-based submissions but was not chosen due to a lack of knowledge on possible attacks. Moreover, this poses the question of whether other metrics may lead to further improvement. The Euclidean metric has been studied extensively in lattice-based cryptography [23], another category amongst the NIST submissions, but there may be some other options as well. The Lee metric, for instance, in combination with quaternary linear codes has been shown to lead to a significant reduction in key sizes and is open to further discussion.

References

[1] L. Chen et al., "Report on Post-Quantum Cryptography", NIST, April 2016.

[2] G. Alagic et al., "Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process", NIST, July 2020.

[3] R. J. McEliece, "A public-key cryptosystem based on algebraic coding theory", California Institute of Technology, 1978.

[4] E. R. Berlekamp, R. J. McEliece and H. C. A. van Tilborg, "On the inherent intractability of certain coding problems", 1978.

[5] Michiel Marcus, "White Paper on McEliece with Binary Goppa Codes", February 2019.

[6] G. Kachigar and J. Tillich, "Quantum Information Set Decoding Algorithms", April 2017.

[7] P. Loidreau, "Codes Derived from Binary Goppa Codes", Problems of Information Transmission, 2001.

[8] E. Berlekamp, "Goppa codes", IEEE Transactions on Information Theory, September 1973.

[9] P. S. L. M. Barreto1, R. Lindner and R. Misoczki, "Decoding square-free Goppa codes over F_p", 2011.

[10] D. Hofheinz, K. Hovelmanns and E. Kiltz, "A Modular Analysis of the Fujisaki-Okamoto Transformation", June 2017.

[11] A. Hulsing, K. Ning, P. Schwabe, F. Weber and P. R. Zimmermann, "Post-quantum WireGuard", 2020.

[12] M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi and P. Santini, "A Failure Rate Model of Bit-flipping Decoders for QC-LDPC and QC-MDPC Code-based Cryptosystems", July 2020.

[13] B. Karimi and A. Banihashemi, "Construction of QC-LDPC Codes with Low Error Floor by Efficient Systematic Search and Elimination of Trapping Sets", February 2019.

[14] T. Richardson, "Error Floors of LDPC Codes", 2003.

[15] D. Apon, R. Perlner, A. Robinson and P. Santini, "Cryptanalysis of LEDAcrypt", August 2020.

[16] C. Aguilar, O. Blazy, J. Deneuville, P. Gaborit and G. Zemor, "Efficient Encryption from Random Quasi-Cyclic Codes", December 2016.

[17] M. Alekhnovich, "More on average case vs approximation complexity", October 2003.

[18] N. Aragon, P. Gaborit and G. Zemor, "HQC-RMRS, an instantiation of the HQC encryption framework with a more efficient auxiliary error-correcting code", May 2020.

[19] E. Couselo, S. Gonzalez, V. Markov, C. Martinez and A. Nechaev, "On insecurity of cryptosystems based on generalized Reed-Solomon codes", 1992.

[20] C. Melchor et al., "Rank Quasi-Cyclic (RQC) Second Round Specification", April 2020.

[21] A. V. Ourivski and T. Johansson, "New Technique for Decoding Codes in the Rank Metric and Its Cryptography Applications", 2002.

[22] P. Gaborit and O. Ruatta, "On the Complexity of the Rank Syndrome Decoding Problem", January 2013.

[23] D. Micciancio and O. Regev, "Lattice-based Cryptography", 2009.

[24] N. Aragon, P. Gaborit, A. Hauteville, O. Ruatta and G. Zemor, "Low Rank Parity Check Codes: New Decoding Algorithms and Applications to Cryptography", March 2019.

[25] M. Bardet et al., "Improvements of Algebraic Attacks for solving the Rank Decoding and MinRank problems", February 2020.

[26] C. Peters, "Information-set decoding for linear codes over F_q", May 2010.

[27] E. Prange, "The use of information sets in decoding cyclic codes", September 1962.

[28] J. Stern, "A method for finding codewords of small weight", 1988.

[29] E. Persichetti, "Improving the Efficiency of Code-Based Cryptography", January 2013.

[30] A. Horlemann-Trautmann and V. Weger, "Information set decoding in the Lee metric with applications to cryptography", 2019.

[31] A. May, A. Meurer and E. Thomae, "Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$", 2011.

[32] A. Becker, A. Joux, A. May and A. Meurer, "Decoding Random Binary Linear Codes in $2^{n/20}$: How 1+1=0 Improves Information Set Decoding", 2012.

[33] A. May and I. Ozerov, "On computing nearest neighbors with applications to decoding of binary linear codes", 2015.

[34] D. J. Bernstein, T. Lange and C. Peters, "Attacking and defending the McEliece cryptosystem", August 2008.

[35] D. J. Bernstein et al., "Classic McEliece Round 2", 2019.

[36] R. Torres and N. Sendrier, "Analysis of Information Set Decoding for a Sub-linear Error Weight", 2016.

[37] N. Aragon et al., "BIKE Round 3 Submission", 2020.

[38] E. Persichetti, "Compact McEliece keys based on Quasi-Dyadic Srivastava codes", January 2011.