Journal of Computer Virology and Hacking Techniques
https://doi.org/10.1007/s11416-024-00527-z

ORIGINAL PAPER

Design criteria of a new code-based KEM

Victoria Vysotskaya[1,2] · **Ivan Chizhov**[1,2,3]

Received: 15 September 2023 / Accepted: 13 May 2024

Abstract
The advances in quantum technologies became a threat to cryptosystems based on number-theoretic approach. Therefore, the development of post-quantum algorithms is currently underway. One of the areas of research is key encapsulation mechanisms (KEMs), which are supposed to replace the Diffie–Hellman key exchange protocol. When constructing such mechanisms, a modular approach based on a public key cryptosystem is often used. We provide an overview of such approaches for schemes based on error-correcting codes. We present arguments for and against the choice of each component of the modular approach. Moreover, we propose the combinations allowing to build KEMs with most favorable characteristics and present a proof of security for one of them.

Keywords Post-quantum cryptography · Code-based cryptography · KEM

1 Introduction

Post-quantum cryptography is a relatively young field, which appeared as a "counter-measure" to Shor's algorithms [1] solving factorization and discrete logarithm problems with polynomial complexity on a quantum computer whereas they are the basis of security of all modern cryptographic protocols. In order for these protocols to remain secure in the era of powerful quantum computers, it is necessary to modify them so that their security relies on the complexity of problems that are not covered by Shor's algorithms. Such problems include, for example, problems related to error-correcting codes, lattices, multivariate polynomials and hash functions. A large number of new schemes were spawned by the NIST competition for the best post-quantum algorithms for further standardization.

✉ Victoria Vysotskaya
vysotskaya.victory@gmail.com

Ivan Chizhov
ichizhov@cs.msu.ru

[1] Cryptography Laboratory, JSRPC Kryptonite, Shlyuzovaya Naberezhnaya, Moscow, Russia 115114

[2] Information Security Department, Faculty of Computational Mathematics and Cybernetics of Lomonosov Moscow State University, Leninskie Gory, Moscow, Russia 119991

[3] Department 53, Federal Research Center "Informatics and Control" of Russian Academy of Science, Vavilova, Moscow, Russia 119333

NIST has claimed two main areas of research: key encapsulation mechanisms (KEM) and digital signature schemes. This choice is explained by the necessity of replacement of the main public-key cryptosystems which are subject to quantum attacks. These include digital signatures and encryption schemes, as well as key exchange protocols like Diffie–Hellman one. If post-quantum signatures are obviously designed to replace classical ones, then the other two points are closed by KEMs. Public key encryption schemes are included in the KEM proposal as an integral part, and the KEMs themselves are analogues of key exchange protocols, where, however, the key is generated by one party and is transferred to the second party and not created by them jointly.

Known approaches to constructing KEMs on error-correct -ing codes do not depend on the structure of codes as such, but the codes affect cryptosystems' properties and security. A KEM is usually built on a cryptosystem in general form but requires it to satisfy a number of properties. And it is the refinement of this cryptosystem that sets the security level and the performance characteristics of the final scheme. To build such a modular KEM scheme, one needs to fix a family of codes with its specialized decoding algorithm, a cryptosystem based on it and a transformation from this cryptosystem to KEM.

There exist schemes based on various codes, cryptosystems and transformations, but often the proposals lack design rationale and do not explain why one approach rather than

Published online: 13 June 2024

another is chosen. At the same time, these considerations would be very helpful when synthesizing new schemes or for choosing one of the already proposed options. In the present work we aim to consolidate all the disparate discussions on the advantages and disadvantages of various approaches, as well as to analyze the outcomes of applying the most promising among them.

2 Results

In Sects. 4–6 we provide a detailed description of the steps required for synthesis of an error-correcting code-based KEM. By selecting the best approach from a set of options at each step, a scheme with optimal characteristics can be constructed. Such schemes are enumerated in Sect. 7 along with related remarks. Finally, in Sect. 8 we give a complete proof for one of them.

3 Preliminaries

In this section we provide definitions and facts used later in the text.

3.1 Coding theory and problems

Definition 1 (*Linear code*) Let q, m, n be positive integers such that q is prime and $k < n$. And let $\mathrm{GF}(q^m)$ be a Galois field of order q^m and V_n be a linear space over the field $\mathrm{GF}(q^m)$ of dimension n. Then linear block code is a linear k-dimensional subspace $\mathcal{C}$ of the space V_n.

Here k is called the *dimension* of the code, and n is its *length*. One of the most important characteristics of a code is parameter t that corresponds to the number of errors the code can fix. It is determined by the minimum distance of the code.

Definition 2 (*Hamming weight*) Hamming weight $\mathrm{wt}(x)$ of vector x is the number of its nonzero elements.

Definition 3 (*Hamming distance*) Hamming distance $\rho(x, y)$ between vectors x and y is the number of positions at which the corresponding bits in this vectors are different.

Definition 4 (*Minimum distance*) The minimum distance d of the code $\mathcal{C}$ is defined as the minimum Hamming distance between the distinct codewords of $\mathcal{C}$:

$$d = \min_{\substack{x \in \mathcal{C},\ y \in \mathcal{C},\\ x \neq y}} \rho(x, y).$$

A linear code $\mathcal{C}$ with parameters n, k and t can be defined either by its generator or parity-check matrix.

Definition 5 (*Generator matrix*) Matrix G of size $k \times n$ with elements from $\mathrm{GF}(q^m)$ is called the generator matrix of code $\mathcal{C}$ if its rows form a basis of $\mathcal{C}$.

Definition 6 (*Parity-check matrix*) Full-rank matrix H of size $(n - k) \times n$ with elements from $\mathrm{GF}(q^m)$ is called a parity-check matrix of code $\mathcal{C}$ if the equality $Hc^T = 0$ holds if and only if $c \in \mathcal{C}$.

Definition 7 (*Syndrome decoding*) The problem of finding a vector $e \in \mathrm{GF}(q^m)^n$ such that $\mathrm{wt}(e) = t$ and $He^T = s^T$ given as inputs a parity-check $(n - k) \times n$-matrix H of a code over $\mathrm{GF}(q^m)$, a nonzero vector $s \in \mathrm{GF}(q^m)^{n-k}$ and an integer t is called the syndrome decoding problem.

Definition 8 (*Decoding*) The problem of finding a pair of vectors $(x, e) \in \mathrm{GF}(q^m)^k \times \mathrm{GF}(q^m)^n$ such that $\mathrm{wt}(e) = t$ and $y = xG + e$ given as inputs a generator $(k \times n)$-matrix G of a code over $\mathrm{GF}(q^m)$, a nonzero vector $y \in \mathrm{GF}(q^m)^n$ and an integer t is called the decoding problem.

The decision syndrome decoding problem is NP-complete [2, 3]. The decoding problem is equivalent to the syndrome decoding problem in terms of complexity and therefore is also NP-hard. The best known algorithm named Information Set Decoding (ISD) solves this problem requiring $O(2^{0.0465n})$ bit operations [4] for $t = d/2$.

3.2 Cryptosystems

Henceforth we denote the message space by $\mathcal{M}$, the key space by $\mathcal{K}$ and the randomness space by $\mathcal{R}$. We use λ for the security parameter.

Definition 9 (PKE) A public-key cryptosystem (PKE) is a triplet of algorithms (KGen, Enc, Dec) such that

1. KGen is a polynomial probabilistic key generation algorithm such that $\mathsf{KGen}(1^\lambda) = (\mathsf{pk}, \mathsf{sk})$, where pk is the public key and sk is the secret key;
2. Enc is a polynomial (probabilistic) encryption algorithm that for an arbitrary $m \in \mathcal{M}$ returns $\mathsf{Enc}(\mathsf{pk}, m) = c$, where c is called the ciphertext;
3. Dec is a polynomial decryption algorithm such that $\mathsf{Dec}(\mathsf{sk}, c) =$

$$= \begin{cases} m \in \mathcal{M}, & \text{if } c \text{ is a valid ciphertext for } m; \\ \perp \notin \mathcal{M}, & \text{otherwise.} \end{cases}$$

Moreover, for any message $m \in \mathcal{M}$ and any key $\mathsf{sk} \leftarrow \mathsf{KGen}(1^\lambda)$ it holds that $m = \mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m))$.

A cryptosystem having deterministic algorithm Enc is also called *deterministic*. Otherwise we can explicitly indicate the

use of randomness $r \in \mathcal{R}$ by writing $\mathsf{Enc}(\mathsf{pk}, m, r)$. Further in the text, by default we assume that the cryptosystem is non-deterministic.

In [5] several more important properties of public-key cryptosystems were defined.

Definition 10 (*Rigidity*) A deterministic PKE is called rigid if for all key pairs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$ and all ciphertexts c, it holds that either $\mathsf{Dec}(\mathsf{sk}, c) = \perp$ or $\mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, c)) = c$.

Definition 11 (γ-*spreadness*) PKE is said to be γ-spread if, for every key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$, every message $m \in \mathcal{M}$ and every possible ciphertext $c \in \mathcal{C}$ holds

$$\mathbb{P}_{r \leftarrow\$ \mathcal{R}}[c = \mathsf{Enc}(\mathsf{pk}, m, r)] \leqslant 2^{-\gamma}.$$

Definition 12 (δ-*correctness*) PKE is called δ-correct if

$$\mathbb{E}\left[\max_{m \in \mathcal{M}} \mathbb{P}[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) \neq m]\right] \leqslant \delta,$$

where the expectation is taken over $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$ and the probability is taken over internal probabilities of Enc and Dec algorithms.

Definition 13 (OW-CPA PKE) For any adversary $\mathcal{A}$ in the OW-CPA model the advantage against PKE is defined as follows:

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{A}) = \mathbb{P}[\mathbf{Exp}^{\mathsf{OW\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{A}) \Rightarrow 1],$$

where the experiment OW-CPA is described below:

$\mathbf{Exp}^{\mathsf{OW\text{-}CPA}}(\mathcal{A})$

1: $(\mathsf{pk}, \mathsf{sk}) \leftarrow\$ \mathsf{KGen}(1^\lambda)$
2: $m^* \leftarrow\$ \mathcal{M}$
3: $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m^*)$
4: $m' \leftarrow \mathcal{A}(\mathsf{pk}, c^*)$
5: **return** $m' \stackrel{?}{=} m^*$

Definition 14 (IND-CPA & IND-CCA PKE) For any adversary $\mathcal{A}$ in model Model $\in$ {IND-CPA, IND-CCA} the advantage against PKE is defined as follows:

$$\mathsf{Adv}^{\mathsf{Model}}_{\mathsf{PKE}}(\mathcal{A}) = \left|\mathbb{P}[\mathbf{Exp}^{\mathsf{Model}}_{\mathsf{PKE}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right|,$$

where the experiment Model is described below:

$\mathbf{Exp}^{\mathsf{Model}}_{\mathsf{PKE}}(\mathcal{A})$

1: $(\mathsf{pk}, \mathsf{sk}) \leftarrow\$ \mathsf{KGen}(1^\lambda)$
2: $b \leftarrow\$ \{0, 1\}$
3: $(m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{Model}}}(\mathsf{pk})$
4: $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$
5: $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{Model}}}(\mathsf{pk}, c^*, \mathsf{st})$
6: **return** $b' \stackrel{?}{=} b$

DEC(c)

1: **if** $c = c^*$ **then**
2: **return** $\perp$
3: **else**
4: $m = \mathsf{Dec}(\mathsf{sk}, c)$
5: **return** m

$$\mathcal{O}_{\mathsf{Model}} = \begin{cases} -, & \mathsf{Model} = \mathsf{IND\text{-}CPA}, \\ \mathrm{DEC}, & \mathsf{Model} = \mathsf{IND\text{-}CCA}. \end{cases}$$

3.3 Key encapsulation mechanisms

Definition 15 (KEM) For a given key space $\mathcal{K}$ a key encapsulation mechanism (KEM) is a triplet of algorithms (KGen, Encaps, Decaps) such that

1. KGen is a polynomial probabilistic key generation algorithm such that $\mathsf{KGen}(1^\lambda) = (\mathsf{pk}, \mathsf{sk})$, where pk is the public key and sk is the secret key;
2. Encaps is a polynomial probabilistic encapsulation algorithm such that $\mathsf{Encaps}(\mathsf{pk}) = (K, c)$, where c is called the encapsulation of the key $K \in \mathcal{K}$;
3. Decaps is a polynomial decapsulation algorithm such that $\mathsf{Decaps}(\mathsf{sk}, c) =$

$$= \begin{cases} K \in \mathcal{K}, & \text{if } c \text{ is a valid encapsulation of } K; \\ \perp \notin \mathcal{K}, & \text{otherwise.} \end{cases}$$

Moreover, for any key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ and any pair $(K, c) \leftarrow \mathsf{Encaps}(\mathsf{pk})$ it holds that $K = \mathsf{Decaps}(\mathsf{sk}, c)$.

Definition 16 (IND-CPA & IND-CCA KEM) For any adversary $\mathcal{A}$ in model Model $\in$ {IND-CPA, IND-CCA} the advantage against KEM is defined as follows:

$$\mathsf{Adv}^{\mathsf{Model}}_{\mathsf{KEM}}(\mathcal{A}) = \left|\mathbb{P}[\mathbf{Exp}^{\mathsf{Model}}_{\mathsf{KEM}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right|,$$

where the experiment Model is described below:

$\mathbf{Exp}^{\mathsf{Model}}_{\mathsf{KEM}}(\mathcal{A})$

1: $(\mathsf{pk}, \mathsf{sk}) \leftarrow\$ \mathsf{KGen}(1^\lambda)$
2: $b \leftarrow\$ \{0, 1\}$
3: $(K_0^*, c^*) \leftarrow \mathsf{Encaps}(\mathsf{pk})$
4: $K_1^* \leftarrow\$ \mathcal{K}$
5: $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Model}}}(\mathsf{pk}, c^*, K_b^*)$
6: **return** $b' \stackrel{?}{=} b$

DECAPS(c)

1: **if** $c = c^*$ **then**
2: **return** $\perp$
3: **else**
4: $K = \mathsf{Decaps}(\mathsf{sk}, c)$
5: **return** K

$$\mathcal{O}_{\mathsf{Model}} = \begin{cases} -, & \mathsf{Model} = \mathsf{IND\text{-}CPA}, \\ \textsc{Dec}, & \mathsf{Model} = \mathsf{IND\text{-}CCA}. \end{cases}$$

3.4 Pseudo-random functions

Definition 17 (*Advantage in* PRF *model*) Let $\mathsf{F} : \mathcal{K}_\mathsf{F} \times \mathcal{M} \to \mathcal{C}$ be a family of keyed functions and $\mathsf{Func}(\mathcal{M}, \mathcal{C})$ be a set of all functions of the form $\mathcal{M} \to \mathcal{C}$. The advantage in the PRF model of an adversary $\mathcal{A}$ is defined as follows:

$$\mathsf{Adv}^{\mathsf{PRF}}_{\mathsf{F}}(\mathcal{A}) = \\ = \mathbb{P}[\mathbf{Exp}^{\mathsf{PRF\text{-}1}}_{\mathsf{F}}(\mathcal{A}) \Rightarrow 1] - \mathbb{P}[\mathbf{Exp}^{\mathsf{PRF\text{-}0}}_{\mathsf{F}}(\mathcal{A}) \Rightarrow 1],$$

where

$\mathbf{Exp}^{\mathsf{PRF}\text{-}b}_{\mathsf{F}}(\mathcal{A})$

if $b = 1$ **then**
  $K \leftarrow\$ \mathcal{K}_\mathsf{F}$
else
  $F \leftarrow\$ \mathsf{Func}(\mathcal{M}, \mathcal{C})$
$b' \leftarrow\$ \mathcal{A}^{\mathcal{F}^b}$
return b'

$\mathcal{F}^b(m)$

if $b = 1$ **then**
  return $\mathsf{F}(K, m)$
else
  return $F(m)$
return b

Here $\mathsf{Func}(\mathcal{M}, \mathcal{C})$ is the set of all functions mapping $\mathcal{M}$ to $\mathcal{C}$.

Definition 18 (PRF) A function $\mathsf{F} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ is called pseudorandom if:

1. given a key $K \in \mathcal{K}$ and an input $m \in \mathcal{M}$ there is an "efficient" algorithm to compute $F(K, m)$;
2. for any polynomial adversary $\mathcal{A}$ and a small predetermined value ε holds that $\mathsf{Adv}^{\mathsf{PRF}}_{\mathsf{F}}(\mathcal{A}) \leqslant \varepsilon$.

4 Code selection

In this section, we describe the two most promising classes of codes to use in KEM.

4.1 Goppa codes

Goppa codes were used in the original versions of the oldest code-based cryptosystems, namely McEliece and Niederreiter ones (more details about them can be found in Sect. 5). Three schemes based on these codes have been presented at the first round of NIST competition: Classic McEliece [6], NTS-KEM [7] and Edon-K [8]. BIG QUAKE [9] uses the subclass of these codes called quasi-cyclic Goppa codes.

The complex structure of Goppa codes keeps these schemes secure whereas changing the code often results in a vulnerability. The codes are considered indistinguishable from random despite the existence of a distinguisher [10] for a particular subclass of these codes. Both decoding and syndrome decoding problems on Goppa codes traditionally are treated as NP-hard similar to random linear codes (despite of the fact it was never proven).

However, the structure of these codes is also one of their shortcomings. The public key of the cryptosystem cannot be represented compactly, therefore it has a huge size. This property critically restricts the applicability of such schemes. In addition, PKEs on Goppa codes did not avoid side-channel attacks. Still, all these attacks are based on strong assumptions about the adversary capabilities.

A large class of timing attacks exploited the fact Patterson algorithm works non-constant time [11–15]. Later a power attack of this kind also appeared [16]. As countermeasures, modifications were proposed that introduced additional operations to compensate for optimizations and ensure consistent time complexity in all cases. Another countermeasure is masking [17]. But although it straightforward to prevent the attacks of this class, the modifications slowed down the decoding algorithm. Furthermore, as soon as one vulnerability was fixed, another was found almost immediately. Ultimately, the Patterson algorithm was completely abandoned in favor of the constant-time Berlekamp-Massey algorithm.

A more serious threat is a fault-injection attack from [18] that makes it possible to attack the scheme with the NIST highest security level in several seconds. Although no specific countermeasures are known for the attack, one should keep in mind that it uses laser fault injection, that is, the adversary needs to have access to the device. Another attack [19] becoming possible for the FPGA implementations when the attacker is in possession of the device, uses the leakage of electromagnetic radiation. The attack allows to reveal the secret key even when using the Berlekamp-Massey algorithm.

A generalization of Goppa codes is Srivastava codes (used in DAGS [20] from NIST competition) which, in turn, belong to the class of alternant codes which are a special case of algebraic geometry codes. Another subclass of algebraic geometry codes is generalized Reed-Solomon codes. Since the extension of the code class makes its instances more random-looking it is possible to consider transitioning from schemes based on Goppa codes to schemes based on their generalization or another subclass of algebraic geometry codes. Moreover, both alternate and Reed-Solomon codes have efficient decoding algorithms.

There is a variant of KEM based on generalized Srivastava codes [21]. There have also been two unsuccessful attempts. One is based on generalized Reed-Solomon codes (the RLCE

scheme was proposed in the NIST competition [22] and attacked in [23]). The other one is presented in [24] together with an attack on itself. This scheme is based on another specific subclass of algebraic geometry code. Attacks on PKE schemes based on generalized Reed-Solomon codes are also known, see [25, 26].

Thus, the alternatives to Goppa codes mentioned above are promising but poorly studied in terms of their application in cryptography. More research should precede their usage in schemes that are subject to further standardization.

4.2 Quasi-cyclic codes

Cyclic codes are generated by cyclic matrices, i.e. wherein each row is a cyclic shift of the previous row by one position. This particular property enables a cyclic matrix to be described using only n bits, rather than n^2 bits, rendering it highly efficient for storage purposes. A generalization of this concept is seen in quasi-cyclic matrices, which are block matrices consisting of cyclic matrix blocks. Although quasi-cyclic matrices are also storage-optimized, they lack the explicit structure exhibited by cyclic matrices.

In the realm of coding theory literature, codes based on specific subclasses of cyclic matrices, such as QC-LDPC (Quasi-Cyclic Low-Density Parity-Check) and QC-MDPC (Quasi-Cyclic Moderate-Density Parity-Check) codes, are frequently employed. Rows of the cyclic submatrices that comprise the generating matrices of these codes have constant weight ($\mathcal{O}(1)$) in the former case and weight bounded by $\mathcal{O}(\sqrt{n \log_2 n})$ in the latter. Each of these types of codes is represented in the NIST competition. HQC [27] and RQC [28] are based, among others, on a quasi-cyclic code (in Hamming and rank metric respectively). BIKE [29] and QC-MDPC [30] use QC-MDPC codes, LEDAkem [31] uses QC-LDPC codes and Lepton [32] uses repetition BCH code that is also quasi-cyclic due to the cyclicality of BCH codes.

Similar to Goppa codes, non-constant decoding time exposes vulnerabilities to timing attacks, such as those observed in schemes on QC-MDPC codes [33, 34], notably the HQC scheme from the first round of the NIST competition [35, 36] and the final versions of HQC and BIKE [37]. Fending off these attacks necessitates the adoption of fully constant-time implementations. Additionally, it is imperative to keep the generation time of a random vector in the encryption algorithm constant.

However, when using quasi-cyclic codes, ensuring constant execution time of the decapsulation algorithm is not enough to ensure security. Another critical concern arises in the form of power analysis and differential power analysis (DPA) attacks, which exploit the analysis of power traces. The proposed cryptanalysis method effectively recovers the complete secret key through a limited number of decryption observations. These attacks consist of a combination of a differential leakage analysis during the syndrome computation followed by an algebraic step that exploits the relation between the public and private key. The applicability of these attacks extends to schemes based on QC-MDPC codes [38–40], as well as second-round NIST competition schemes, specifically QC-MDPC KEM (utilizing QC-MDPC codes) and LEDA (using QC-LDPC codes) [41]. However, employing countermeasures such as noise introduction and useless operation incorporation, along with decoding randomization and masking techniques, effectively mitigates all attacks of this this class.

A vulnerability was discovered in the NIST competition Round 2 variant of HQC KEM, wherein an attacker could exploit the power consumption patterns of the decoder to expose the secret key [42]. The principle of the attack was to observe and differentiate the power consumption of the decoder depending on whether it corrected an error for a chosen ciphertext. For rare cases when the described approach did not succeed, the authors proposed an adjusted decoding algorithm that incorporated an ISD variant based on side-channel information. It is important to note that the effectiveness of this attack relied on the specific characteristics of BCH codes, which were subsequently replaced within the proposal. However, even for the Round 3 version of HQC a vulnerability of this nature was still present [43]. By leveraging power analysis, an attacker could recover the secret key with an acceptable number of measurements even for the parameter set supposed to provide 256 bits of security.

Subsequently, an efficient reaction attack was built on the QC-MDPC KEM [44] and later a similar attack was devised for the LEDA [45]. In recent theoretical work [46], it was demonstrated that unlike bounded-distance decoders used for algebraic codes such as Goppa ones, iterative decoders used for sparse codes do not have a deterministic decoding radius, and thus the decoding may fail with some probability that is called the decoding failure rate (DFR). Consequently, this parameter is now considered crucial for the code selection and is required not to exceed $2^{-\lambda}$.

Additionally, a notable characteristic of schemes based on quasi-cyclic codes is the reduced complexity of the ISD algorithm. Extensive studies [47] have demonstrated that the work factor of a quasi-cyclic code is equal to the work factor of a random code with equivalent parameters, multiplied by a factor of $1/\sqrt{N}$. Here, N corresponds to the number of rows in the internal quasi-cyclic submatrix, which denotes the number of repetitions of the same row.

5 Cryptosystems

To of the oldest error-correcting code-based cryptosystems are McEliece [48] and Niederreiter [49] ones. They can be considered fundamental in a sense, as all subsequent vari-

ants can be viewed as modifications of either one of these cryptosystems or a combination thereof. That is, Classic McEliece [6] and LEDAkem [31] are built on the original Niederreiter cryptosystem, BIKE [29] and LOCKER [50] use randomized version of this scheme while HQC [27] and RQC [28] use it as one of two basic cryptosystems. At the same time QC-MDPC [30] and Edon-K [8] are built on the McEliece cryptosystem whereas NTS-KEM [7] and DAGS [20] are constructed based on its modification.

The most common alteration is changing the key generation algorithm. It is usually chosen to ensure that the corresponding trapdoor one-way function is difficult for the selected class of codes. Therefore, below we provide only the encryption algorithms of the aforementioned McEliece and Niederreiter cryptosystems, which are more versatile. And we omit the decryption algorithms since they also significantly depend on the chosen code and key generation algorithm.

Both cryptosystems utilize an error vector, which is a vector of length n with a fixed Hamming weight t, where parameter t is small. However, while in the McEliece cryptosystem this vector is chosen uniformly at random from the set of all such vectors $\mathcal{H}_{n,t}$, the Niederreiter cryptosystem employs a specific transformation ϕ that maps the message to the error vector. Consequently, the latter cryptosystem is deterministic.

$\mathsf{Enc}_{\mathsf{McEl}}(\mathsf{pk} = G, m)$
$e \leftarrow\!\$\ \mathcal{H}_{n,t}$
$c \leftarrow mG + e$
return c

$\mathsf{Enc}_{\mathsf{Nieder}}(\mathsf{pk} = H, m)$
$c \leftarrow Hm^T$
return c

From the above the encryption algorithm of the McEliece cryptosystem transforms a message of length k into a ciphertext of length n (that is $|\mathcal{M}| = 2^k$), while the Niederreiter encryption allows obtaining a ciphertext of length $n-k$ from a modified message of length n (and $|\mathcal{M}| = \binom{n}{t}$).

The mapping ϕ can be defined in any way, which is usually chosen for best performance characteristics. One way is the following. We divide the original message M into t parts of length $|M_i|$. Then consider each part as the binary representation of the position of "1" in the corresponding block of m. So it is necessary that $|M| \leqslant t \log_2(n/t)$.

Moving on to additional properties of PKE we should note that both cryptosystems are δ-correct and δ is defined by underlying code. For quasi-cyclic codes parameter choices can make this value less than $2^{-\lambda}$. Bounded-distance decoders of Goppa codes provide $\delta = 0$. Such schemes are also called *perfectly correct*.

Rigidity also depends on the code and, moreover, on the specific decoding algorithm. For example, the Niederreiter cryptosystem based on Goppa codes, in which the Berlekamp-Massey algorithm is used for decoding [51], is rigid. This is due to the ability of the decoding algorithm to detect incorrect inputs, i.e. ciphertexts that were not obtained as a result of the encoding algorithm. Such inputs can be decoded into $\perp$, and on the rest the decoding algorithm $\mathsf{Dec}(\mathsf{sk}, c)$ can have the only output. And then from the determinism of the encryption algorithm for this output m holds that $\mathsf{Enc}(\mathsf{pk}, m) = c$.

However, it is impossible to claim rigidity for the Niederreiter cryptosystem in the general case, even when Goppa codes are used. The same is true for the cryptosystem based on quasi-cyclic codes.

For the McEliece cryptosystem, the property is not fulfilled naturally due to the non-determinism of the encryption algorithm.

However, for its deterministic variant (achieved by using the transformation T described further in Sect. 6.2) is proven to be rigid.

Finally the McEliece cryptosystem is $1/\binom{n}{t}$-spread and for the Niederreiter cryptosystem the definition is undefined.

6 Construction of KEM

KEM schemes are typically built upon PKE schemes. In these constructions, KGen algorithms usually coincide, except for possibly generating an additional value. Algorithm Encaps incorporates algorithm Enc and algorithm Decaps relies on Dec (see Fig. 1).

All PKE-to-KEM transformations can be divided into two main classes: security-preserving transformations and security-amplifying transformations. We consider both of them further.

6.1 Security-preserving transformations

Let us introduce here an intuitive way of building $\mathsf{KEM} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ scheme based on $\mathsf{PKE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$. Note that key generation algorithms are identical.

$\mathsf{Encaps}(\mathsf{pk})$:
1 : $K \leftarrow\!\$\ \mathcal{M}$
2 : $c \leftarrow \mathsf{Enc}(\mathsf{pk}, K)$
3 : **return** (K, c)

$\mathsf{Decaps}(\mathsf{sk}, c)$:
1 : $K \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
2 : **return** K

Below is the proof of the result, which can be considered folklore.

Theorem 1 *For any* IND-CPA (IND-CCA) *adversary* $\mathcal{A}$ *against the resulting* KEM *there exists an* IND-CPA (IND-CCA) *adversary* $\mathcal{B}$ *against the original* PKE*, running in about the same time, such that*

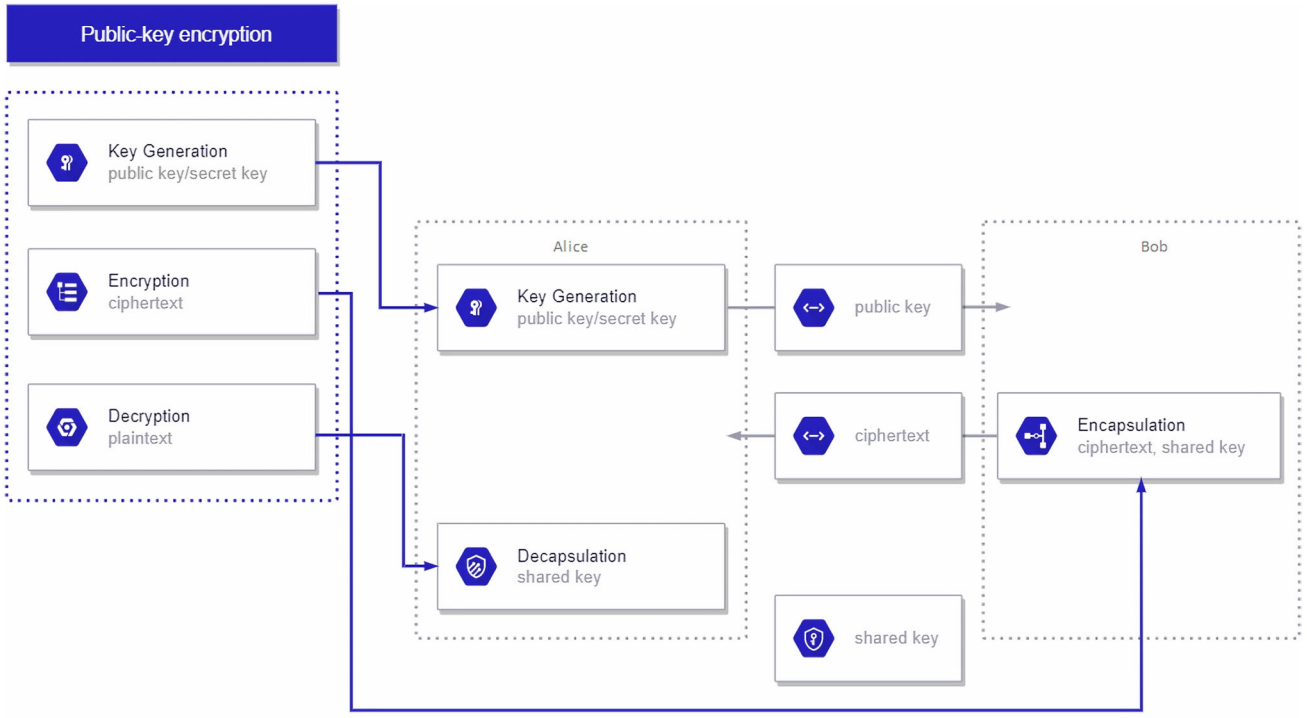

Fig. 1 Transformation from PKE to KEM

$$\mathsf{Adv}^{\text{IND-CPA}}_{\text{KEM}}(\mathcal{A}) = \mathsf{Adv}^{\text{IND-CPA}}_{\text{PKE}}(\mathcal{B}),$$
$$\mathsf{Adv}^{\text{IND-CCA}}_{\text{KEM}}(\mathcal{A}) = \mathsf{Adv}^{\text{IND-CCA}}_{\text{PKE}}(\mathcal{B}).$$

Moreover, if IND-CCA *adversary* $\mathcal{A}$ *was issuing* q_D *queries to the decapsulation oracle* DECAPS *then* IND-CCA *adversary* $\mathcal{B}$ *issues* q_D *queries to the decryption oracle* DEC.

Proof For the adversary $\mathcal{A}$ that attacks KEM in Model (IND-CPA or IND-CCA) the experiment $\mathbf{Exp}^0$ coincides with the classical experiment in this model, i.e.

$$\mathsf{Adv}^{\text{Model}}_{\text{KEM}}(\mathcal{A}) = \left|\mathbb{P}[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right|.$$

$\mathbf{Exp}^0(\mathcal{A})$

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\$\ \mathsf{KGen}(1^\lambda)$
2 : $b \leftarrow\!\$\ \{0,1\}$
3 : $K_0^* \leftarrow\!\$\ \mathcal{M}$
4 : $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}, K_0^*)$
5 : $K_1^* \leftarrow\!\$\ \mathcal{M}$
6 : $b' \leftarrow \mathcal{A}^{\mathcal{O}_1}(\mathsf{pk}, c^*, K_b^*)$
7 : **return** $b' \stackrel{?}{=} b$

DECAPS(c) $(\mathbf{Exp}^0, \mathbf{Exp}^1)$

1 : **if** $c = c^*$ **then**
2 : **return** $\perp$
3 : **else**
4 : $K \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
5 : **return** K

$$\mathcal{O}_1 := \begin{cases} - & \text{for IND-CPA} \\ \text{DECAPS} & \text{for IND-CCA} \end{cases}$$

In the experiment $\mathbf{Exp}^1$ the adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ attacks PKE encryption schemes in Model (IND-CPA or IND-CCA). $\mathcal{B}_1$ chooses a pair of messages at random, sends it to $\mathcal{B}_2$ and $\mathcal{B}_2$ calls the adversary $\mathcal{A}$ against KEM in the corresponding model. Then

$$\mathsf{Adv}^{\text{Model}}_{\text{KEM}}(\mathcal{A}) = \left|\mathbb{P}[\mathbf{Exp}^1(\mathcal{B}) \Rightarrow 1] - \frac{1}{2}\right|.$$

$\mathbf{Exp}^1(\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2))$:

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\$\ \mathsf{KGen}(1^\lambda)$
2 : $b \leftarrow\!\$\ \{0,1\}$
3 : $(m_0, m_1, \mathsf{st}) \leftarrow \mathcal{B}_1(\mathsf{pk})$
4 : $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$
5 : $b' \leftarrow \mathcal{B}_2^{\mathcal{O}_2}(\mathsf{pk}, c^*, \mathsf{st})$
6 : **return** $b' \stackrel{?}{=} b$

$\mathcal{B}_1(\mathsf{pk})$

1 : $(m_0, m_1) \leftarrow\!\$\ \mathcal{M}^2$
2 : $\mathsf{st} \leftarrow (m_0, m_1)$
3 : **return** (m_0, m_1, st)

$\mathcal{B}_2^{\mathcal{O}_2}(\mathsf{pk}, c^*, \mathsf{st})$:

1 : $(m_0, m_1) \leftarrow \mathsf{st}$
2 : $b_1 \leftarrow \mathcal{A}^{\mathcal{O}_2}(\mathsf{pk}, c^*, m_0)$
3 : **return** b_1

$$\mathcal{O}_2 := \begin{cases} - & \text{for IND-CPA} \\ \mathsf{Dec} & \text{for IND-CCA} \end{cases}$$

If $b = 1$ then in $\mathbf{Exp}^0$ the adversary $\mathcal{A}$ receives the pair (c_0, K_1^*), where $c_0 = \mathsf{Enc}(\mathsf{pk}, K_0^*)$, and in $\mathbf{Exp}^1$ the adversary $\mathcal{A}$ is given the pair (c_1, m_0), where $c_1 = \mathsf{Enc}(\mathsf{pk}, m_1)$. These pairs are identical up to renaming, that is, the probability of correctly guessing $b' = b$ in these cases are equal.

Alternatively, if $b = 0$ in $\mathbf{Exp}^0$, the adversary $\mathcal{A}$ is provided with the pair (c_0, K_0^*), where $c_0 = \mathsf{Enc}(\mathsf{pk}, K_0^*)$. And in $\mathbf{Exp}^1$ when $b = 0$, the adversary $\mathcal{A}$ receives the pair (c_0, m_0), with $c_0 = \mathsf{Enc}(\mathsf{pk}, m_0)$. Again probabilities coincide.

Hence

$$\mathbb{P}[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1] = \mathbb{P}[\mathbf{Exp}^1(\mathcal{B}) \Rightarrow 1],$$

which implies the condition of the theorem. □

It can be easily shown that both McEliece and Niederreiter cryptosystems provide security only in OW-CPA model, but not in IND-CPA or IND-CCA ones. Let us show their insecurity in IND-CPA model and then insecurity in IND-CCA model will follow automatically.

So the fact under consideration is obvious for the Niederreiter cryptosystem as it is deterministic and the test can be easily done by recounting both ciphertexts.

In case of McEliece cryptosystem the adversary aims at distinguishing between messages m_b, $b \in \{0, 1\}$ after getting the ciphertext c'. Then for each $b' \in \{0, 1\}$ it can compute the value $e' = m_{b'}G + c' = m_{b'}G + m_bG + e_b$, where e_b for $b \in \{0, 1\}$ denotes error vectors added in the encryption algorithm of the McEliece cryptosystem. The last step is to check whether $\mathrm{wt}(e') = t$. The condition will be obviously fulfilled in case $b = b'$ and with high probability fails otherwise.

That is, in order to obtain an IND-CCA secure KEM by applying a security-preserving transformation it is necessary to apply another preliminary transformation that firstly increases the PKE security to IND-CCA.

Not many transformations of this type are known, and even fewer of them can be applied to cryptosystems of interest to us. Some can only be applied to cryptosystems whose Enc function is a permutation [52] and some others are applicable to IND-CPA secure PKEs [53]. But the most suitable ones are based just on an OW-CPA secure cryptosystem [54, 55]. One more transformation additionally requires underlying PKE to be not only OW-CPA secure, but also γ-spread [56].

Transformation from IND-CPA to IND-CCA secure PKE consists mainly in binding the error vector to the message via hashing. All cryptosystems after transformation from an OW-CPA secure one have extended ciphertexts: some additional information depending on message and randomness is added. The first variant [56] is to encrypt random r and concatenate the result with $c' = \mathsf{G}(r) + m$ where G is a generator of a cryptographically secure pseudo random sequences. One more random value can be used additionally [54], in this case $c' = \mathsf{G}(r) + (m \| r_1)$, where $\|$ denotes concatenation. To decrease the ciphertext size only some part of c' may be added [55].

Unfortunately, all aforementioned transformations have only asymptotic estimates which makes it impossible to select parameters for real applications. There are also no examples of such transformations usage among NIST proposals.

6.2 Security-amplifying transformations

Security-amplifying transformations can be represented by a family of so-called Fujisaki-Okamoto (FO) ones. Though original transformations [53, 57] aimed at conversion from weak PKE to ones secure in IND-CCA model, this idea was lately developed at conversions from PKE to KEM.

The first transformation T determines the scheme and raises security either from OW-CPA or from IND-CPA to OW-PCVA (a stronger version of the OW-CPA model [5]). In the first case a rigid cryptosystem is obtained, but in the second one the reduction is tight. The idea is to bind the randomness to the message m by replacing it with its hash value. So, e.g. in McEliece cryptosystem we can use $e = \mathsf{G}(m)$ for some special hash-function G with output of certain weight. Note that for Niederreiter cryptosystem, that is deterministic and not γ-spread, the reduction is trivial.

Next step can be performed by one of the transformations from Fig. 2. They can be grouped according to different properties. First, subscript m indicates that the output key depends only on the message ($K = \mathsf{H}(m)$), while its absence means that the key is obtained with additional use of the ciphertext ($K = \mathsf{H}(m, c)$) for some hash H with output length ℓ. Next, schemes marked as $\perp$ are ones with *explicit rejection*: in case of appearance of symbol $\perp$ inside the Dec algorithm, these algorithms transfer it to the output of Decaps algorithm. Schemes with *implicit rejection* (marked as $\not\perp$) preventively generate additional randomness in KGen algorithm and use it have a key-like output in Decaps. However, in case of the event $\perp$ keys on the two sides will not match. Finally, prefix Q means that additional hash-value $\mathsf{H}'(m)$ is counted in Encaps and checked in Decaps. Note that the superposition of transformation T and one of transformations from Fig. 2 is usually denoted by FO with the appropriate subscripts.

Some of these transformations (namely, $\mathsf{U}^{\perp}$, $\mathsf{QU}_m^{\perp}$ and $\mathsf{FO}_m^{\perp}$) were proposed by A. Dent in [58]. Afterwords Hofheinz et al. [5] systematized and generalized this approach and provided description of transformations T, $\mathsf{U}^{\perp}$, $\mathsf{U}^{\not\perp}$, $\mathsf{U}_m^{\perp}$ and $\mathsf{U}_m^{\not\perp}$. Both articles present the corresponding security proofs in ROM model (except for transformation $\mathsf{U}_m^{\not\perp}$, for which only the concept of proof is given). Further research on the security of transformations $\mathsf{U}^{\not\perp}$ and $\mathsf{U}_m^{\not\perp}$ in ROM model was presented by D. J. Bernstein and E. Persichetti in [59]. Security of transformation $\mathsf{QU}_m^{\not\perp}$ in ROM model has never been studied.

$\mathsf{KGen}(1^\lambda)$:

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $\mathsf{sk}' \leftarrow \mathsf{sk}$ // $\mathsf{U}^\perp, \mathsf{U}^\perp_m, \mathsf{QU}^\perp, \mathsf{QU}^\perp_m$
3 : $s \leftarrow\!\!\$\ \mathcal{M}$ // $\mathsf{U}^\not\perp, \mathsf{U}^\not\perp_m, \mathsf{QU}^\not\perp, \mathsf{QU}^\not\perp_m$
4 : $\mathsf{sk}' \leftarrow (\mathsf{sk}, s)$ // $\mathsf{U}^\not\perp, \mathsf{U}^\not\perp_m, \mathsf{QU}^\not\perp, \mathsf{QU}^\not\perp_m$
5 : **return** $(\mathsf{pk}, \mathsf{sk}')$

$\mathsf{Encaps}(\mathsf{pk})$:

1 : $m \leftarrow\!\!\$\ \mathcal{M}$
2 : $c \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$
3 : $d \leftarrow \mathsf{H}'(m)$ // $\mathsf{QU}^\perp, \mathsf{QU}^\not\perp, \mathsf{QU}^\perp_m, \mathsf{QU}^\not\perp_m$
4 : $K \leftarrow \mathsf{H}(m, c)$ // $\mathsf{U}^\perp, \mathsf{U}^\not\perp, \mathsf{QU}^\perp, \mathsf{QU}^\not\perp$
5 : $K \leftarrow \mathsf{H}(m)$ // $\mathsf{U}^\perp_m, \mathsf{U}^\not\perp_m, \mathsf{QU}^\perp_m, \mathsf{QU}^\not\perp_m$
6 : **return** (K, c) // $\mathsf{U}^\perp, \mathsf{U}^\not\perp, \mathsf{U}^\perp_m, \mathsf{U}^\not\perp_m$
7 : **return** (K, c, d) // $\mathsf{QU}^\perp, \mathsf{QU}^\not\perp, \mathsf{QU}^\perp_m, \mathsf{QU}^\not\perp_m$

$\mathsf{Decaps}(\mathsf{sk}, c)$:

1 : $\mathsf{sk} = (\mathsf{sk}', s)$
2 : $m' \leftarrow \mathsf{Dec}(\mathsf{sk}', c)$
3 : **if** $m' = \perp$ **then** // $\mathsf{U}^\perp, \mathsf{U}^\not\perp, \mathsf{U}^\perp_m, \mathsf{U}^\not\perp_m$
4 : **if** $m' = \perp$ **or** $\mathsf{H}'(m') \neq d$ **then**
5 : // $\mathsf{QU}^\perp, \mathsf{QU}^\not\perp, \mathsf{QU}^\perp_m, \mathsf{QU}^\not\perp_m$
6 : **return** $\perp$ // $\mathsf{U}^\perp, \mathsf{U}^\perp_m, \mathsf{QU}^\perp, \mathsf{QU}^\perp_m$
7 : $K \leftarrow \mathsf{H}(s, c)$ // $\mathsf{U}^\not\perp, \mathsf{U}^\not\perp_m, \mathsf{QU}^\not\perp, \mathsf{QU}^\not\perp_m$
8 : **return** K // $\mathsf{U}^\not\perp, \mathsf{U}^\not\perp_m, \mathsf{QU}^\not\perp, \mathsf{QU}^\not\perp_m$
9 : **else**
10 : $K \leftarrow \mathsf{H}(m', c)$ // $\mathsf{U}^\perp, \mathsf{U}^\not\perp, \mathsf{QU}^\perp, \mathsf{QU}^\not\perp$
11 : $K \leftarrow \mathsf{H}(m')$ // $\mathsf{U}^\perp_m, \mathsf{U}^\not\perp_m, \mathsf{QU}^\perp_m, \mathsf{QU}^\not\perp_m$
12 : **return** K

Fig. 2 Overview of FO transformations

We don't know anything about the security of transformations $\mathsf{QU}^\perp$ and $\mathsf{QU}^\not\perp$ in QROM model. However post-quantum security of other transformations was studied in several articles [5, 60–63]. Additionally, article [63] establishes the equivalence between transformations $\mathsf{U}^\perp$ and $\mathsf{U}^\perp_m$, as well as between transformations $\mathsf{U}^\not\perp$ and $\mathsf{U}^\not\perp_m$.

Below we present Tables 1 and 2, which compare the bounds of various FO transformations for Niederreiter and McEliece PKEs basing on known results. These comparisons aim to illustrate the process of transforming an OW-CPA PKE into an IND-CCA KEM. The tables focus on cryptosystems based on Goppa codes and consider the specific characteristics of both the selected codes and cryptosystems. Consequently, usage of Goppa codes results in $\delta = 0$. In addition to that, when the Niederreiter PKE is used, the determining transformation T can often be excluded. It makes the estimates tighter. Nevertheless, the inclusion of this transformation is sometimes imperative to facilitate reduction to an OW-CPA PKE.

We wish to highlight that according to the remark before Theorem 3.1 in [5] transformation T may result in OW-PCA PKE instead of OW-PCVA. In this case number of queries to CVO oracle (introduced in the above article) is $q_V = 0$. This fact establishes the estimate for $\mathsf{FO}^\not\perp$ transformation below: firstly an IND-CCA secure KEM is reduced to an OW-PCA secure PKE that is subsequently reduced to an OW-CPA secure PKE.

The theorem producing reduction for $\mathsf{FO}^\not\perp_m$ transformation connects IND-CCA secure KEM and deterministic rigid OW-CPA secure PKE. But since the adversary's advantage in OW-CPA model doesn't exceed it's advantage in OW-PCA model, further a known result can be applied that connects an OW-PCA secure deterministic rigid PKE and a general-type OW-CPA secure PKE.

We will upper-bound the security of $\mathsf{QFO}^\perp_m$ transformation applied to McEliece cryptosystem and $\mathsf{QFO}^\not\perp_m$ transformation applied to both McEliece and Niederreiter cryptosystems in ROM model by the security of these transformations in QROM. Moreover, we will also upper-bound the security of transformation $\mathsf{QU}^\perp_m$ in QROM model by the security of transformation $\mathsf{QFO}^\perp_m$ in the same model.

Note that transformations $\mathsf{QU}^\perp$ and $\mathsf{QU}^\not\perp$ are excluded from the tables as lacking security in QROM model. Transformations $\mathsf{U}^\perp$ and $\mathsf{U}^\perp_m$ are also not presented as no reductions to OW-CPA PKE were obtained for them in QROM.

Everywhere in the tables q means the total number of the adversary's queries to various oracles and ε is the success probability of another adversary against the OW-CPA security of the underlying PKE. Let us note additionally that some transformations require the underlying PKE to be rigid. Also some proofs are made for modified transformations where the hash-function is replaced with PRF at the implicit rejection step. The results are given up to constants.

Those transformations are widely presented at NIST competition. Thus, Classic McEliece [6] uses $\mathsf{U}^\not\perp$, BIKE [29] uses $\mathsf{FO}^\not\perp$, DAGS [64] uses $\mathsf{QFO}^\perp_m$ and HQC, BIG QUAKE [9], RQC [28] and LOCKER [50] use $\mathsf{QFO}^\perp$ transformation.

Table 1 FO transformations applied to Niederreiter PKE on Goppa codes and resulting in an IND-CCA secure KEM

Transform	Security bound (ROM)	Security bound (QROM)
$\mathsf{U}^{\not\perp}$	$\varepsilon + \frac{q}{\lvert\mathcal{M}\rvert}$ [59, Thm. 14.3] (requires rigidity)	$\sqrt{\varepsilon}$ [63, Thm. 2] (requires secure PRF)
$\mathsf{U}^{\not\perp}_m$	$\varepsilon + \frac{q}{\lvert\mathcal{M}\rvert}$ [5, Thm. 3.6] (requires rigidity)	$\sqrt{\varepsilon}$ [63, Thm. 2,5] (requires secure PRF)
$\mathsf{QU}^{\perp}_m$	$\varepsilon + \frac{q}{2^\ell}$ [58, Thm. 4]	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.5]
$\mathsf{QFO}^{\not\perp}_m$	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.6]	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.6]

Table 2 FO transformations applied to McEliece PKE on Goppa codes and resulting in an IND-CCA secure KEM

Transform	Security bound (ROM)	Security bound (QROM)
$\mathsf{FO}^{\not\perp}$	$q\varepsilon + \frac{q}{\lvert\mathcal{M}\rvert}$ [5, Thm. 3.1, 3.4]	$q\sqrt{\varepsilon} + \frac{q}{\sqrt{\lvert\mathcal{M}\rvert}}$ [60, Thm. 1]
$\mathsf{FO}^{\not\perp}_m$	$q\varepsilon + \frac{q}{\lvert\mathcal{M}\rvert}$ [5, Thm. 3.1, 3.6]	$q\sqrt{\varepsilon}$ [60, Thm. 2] (requires secure PRF)
$\mathsf{QFO}^{\perp}_m$	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.6]	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.6]
$\mathsf{QFO}^{\not\perp}_m$	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.6]	$q\sqrt{q\sqrt{\varepsilon}}$ [5, Thm. 4.4, 4.6]

7 Notes on best approaches

It can be seen from Tables 1 and 2 that the best security estimates are obtained by the combination of Niederreiter cryptosystem and one of transformations $\mathsf{U}^{\not\perp}$ and $\mathsf{U}^{\not\perp}_m$. On the whole a rigid Niederreiter PKE provides estimates better than a McEliece one as it doesn't require additional transformations to be deterministic. That's why it is possible to avoid accuracy loss associated with using the determining transformation T.

As a result, Niederreiter-based schemes require smaller code parameters to achieve the same security level, as the complexity of decoding problems is directly dependent on the code's length, dimension, and distance. Moreover, Niederreiter PKE has shorter ciphertexts.

Additionally, Sect. 6.1 discusses the advantage of security-amplifying transformations over security-preserving ones. Thus, the two schemes are the most promising among all known variants.

It worth noting that all proofs for transformations $\mathsf{U}^{\not\perp}_m$ and $\mathsf{FO}^{\not\perp}_m$ and some proofs for $\mathsf{U}^{\not\perp}$ in QROM model in articles [60, 62, 63] require the replacement of the function call $\mathsf{H}(s, c)$ in the implicit rejection with the output of a pseudorandom function $\mathsf{F}(s, c)$. At the same time all proofs that are known for transformation $\mathsf{U}^{\not\perp}_m$ in ROM model (see [5, 59]) are given for the basic version presented in Fig. 2.

In article [59] the authors bring up an important problem: theorems are often presented with incomplete or non-rigorous proofs, or sometimes without any at all. We agree that the security notions must be checked as carefully as possible. That's why we decided to close the gap between proofs in classic and quantum models for the transformation $\mathsf{U}^{\not\perp}_m$. The modified KEM obtained by the application of transformation $\mathsf{U}^{\not\perp}_m$ to Niederreiter PKE along with its security proof can be found in the next section.

8 Security of $\mathsf{U}^{\not\perp}_m$ applied to Niederreiter PKE

The goal of the section is to unify the specification of the transformation $\mathsf{U}^{\not\perp}_m$ by using a pseudorandom function $\mathsf{F} : \mathcal{K}_\mathsf{F} \times \mathcal{C} \to \mathcal{K}$ for the implicit rejection. Here $\mathcal{C}$ is the set of all possible outputs of function Enc.

Note that from this approach it follows that the secret s is chosen from the set $\mathcal{K}_\mathsf{F}$.

The listing of the obtained scheme, that is further referred as $\widetilde{\mathsf{KEM}}$, can be found below. Here KGen' is the key generation algorithm of the underlying Niederreiter cryptosystem.

KGen(1^λ)

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}'(1^\lambda)$
2 : $s \leftarrow\!\!\$\ \mathcal{K}_\mathsf{F}$
3 : $\mathsf{sk}' := (\mathsf{sk}, s)$
4 : **return** $(\mathsf{pk}, \mathsf{sk}')$

Encaps(pk)

1 : $m \leftarrow\!\!\$\ \mathcal{M}$
2 : $c := \mathsf{Enc}(\mathsf{pk}, m)$
3 : $K := \mathsf{H}(m)$
4 : **return** (K, c)

Decaps(sk, c)

1 : **parse** $\mathsf{sk}' = (\mathsf{sk}, s)$
2 : $m' := \mathsf{Dec}(\mathsf{sk}, c)$
3 : **if** $m' \neq \perp$ **then**
4 : **return** $K := \mathsf{H}(m')$
5 : **else return** $K := \mathsf{F}(s, c)$

We now present the security notions for our KEM both in ROM and QROM models.

The first one follows the approach of the paper [5], which however lacks the proof of security of transformation $\mathsf{U}_m^{\not\perp}$. Additional changes arise from the properties of the chosen cryptosystem as well as the introduction of a pseudorandom function F. So it is the first complete proof for the proposed scheme.

Theorem 2 *Assume* Niederreiter $\widetilde{\mathsf{PKE}}$ *to be rigid. For any* IND-CCA *adversary* $\mathcal{B}$ *against* $\widetilde{\mathsf{KEM}}$ *issuing at most* q_D *queries to the decapsulation oracle* DECAPS*, and at most* q_H *queries to the random oracle* H*, there exist an* OW-CPA *adversary* $\mathcal{A}$ *against* Niederreiter PKE *and an adversary* $\mathcal{A}'$ *against the security of* PRF F *with at most* q_D *quires such that*

$$\mathsf{Adv}^{\text{IND-CCA}}_{\widetilde{\mathsf{KEM}}}(\mathcal{B}) \leqslant \mathsf{Adv}^{\text{OW-CPA}}_{\text{Nieder}}(\mathcal{A}) + \mathsf{Adv}^{\mathsf{PRF}}_{\mathsf{F}}(\mathcal{A}')$$

and adversaries $\mathcal{A}$ *and* $\mathcal{A}'$ *are running in about the same time and resources as* $\mathcal{B}$*.*

Proof Let $\mathcal{B}$ be an adversary against the IND-CCA security of $\widetilde{\mathsf{KEM}}$, issuing at most q_D queries to the oracle DECAPS.

$\mathbf{Exp}^0(\mathcal{B})$

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow\$ \mathsf{KGen}(1^\lambda)$
2 : $s \leftarrow\$ \mathcal{K}_\mathsf{F}$
3 : $\mathsf{sk}' := (\mathsf{sk}, s)$
4 : $m^* \leftarrow\$ \mathcal{M}\ K_0^* := \mathsf{H}(m^*)$
5 : $K_1^* \leftarrow\$ \mathcal{K}$
6 : $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m^*)$
7 : $b \leftarrow\$ \{0, 1\}$
8 : $b' \leftarrow \mathcal{B}^{\text{DECAPS},\mathsf{H}}(\mathsf{pk}, c^*, K_b^*)$
9 : **return** $b' \stackrel{?}{=} b$

$\mathsf{H}(m)$ $(\mathbf{Exp}^0, \mathbf{Exp}^1)$

1 : **if** $\exists K : (m, K) \in \Pi^H$ **then**
2 : **return** K
3 : $K \leftarrow\$ \mathcal{K}$
4 : $\Pi^H := \Pi^H \cup \{(m, K)\}$
5 : **return** K

DECAPS$(c \neq c^*)$ $(\mathbf{Exp}^0)$

1 : $m' := \mathsf{Dec}(\mathsf{sk}, c)$
2 : **if** $m' = \perp$ **then**
3 : **return** $K := \mathsf{F}(s, c)$
4 : **return** $K := \mathsf{H}(m')$

The experiment $\mathbf{Exp}^0$ is the original IND-CCA experiment with so-called lazy sampling technique. The idea is to explicitly reflect the nature of the random oracle H: on a new query it outputs a random value, but on a repeated one it outputs the same value as before. To achieve this, the set Π^H is introduced to store the requests and answers for all previous queries made to H so far.

Thus for $\mathbf{Exp}^0$ holds that

$$\left|\mathbb{P}[\mathbf{Exp}^0(\mathcal{B}) \Rightarrow 1] - \frac{1}{2}\right| = \mathsf{Adv}^{\text{IND-CCA}}_{\widetilde{\mathsf{KEM}}}(\mathcal{B}).$$

In $\mathbf{Exp}^1$ pseudorandom function $\mathsf{F}(s, c)$ is replaced by $\mathsf{H}'(c)$, where H' is an independent internal random oracle that cannot be accessed by $\mathcal{B}$. This changes the DECAPS oracle, but the rest of the experiment remains unchanged.

DECAPS$(c \neq c^*)$ $(\mathbf{Exp}^1)$

1 : $m' := \mathsf{Dec}(\mathsf{sk}, c)$
2 : **if** $m' = \perp$ **then**
3 : **return** $K := \mathsf{H}'(c)$
4 : **return** $K := \mathsf{H}(m')$

We construct a PRF-adversary $\mathcal{A}'$ which replaces its calls to F by calls to its oracle, runs $\mathcal{B}$, and outputs 1 if $\mathcal{B}$ wins and 0 otherwise. Now the task of distinguishing experiments $\mathbf{Exp}^0$ and $\mathbf{Exp}^1$ precisely coincides with the experiment PRF of the adversary $\mathcal{A}'$ with q_D queries, that is,

$$\left|\mathbb{P}[\mathbf{Exp}^1(\mathcal{B}) \Rightarrow 1] - \mathbb{P}[\mathbf{Exp}^0(\mathcal{B}) \Rightarrow 1]\right| \leqslant \mathsf{Adv}^{\mathsf{PRF}}_{\mathsf{F}}(\mathcal{A}').$$

In $\mathbf{Exp}^2$ the set Π^D is introduced. It contains the requests and answers of the oracle DECAPS. Also oracles H and DECAPS are modified such that they make no use of the secret key any longer.

$\mathsf{H}(m)$ $(\mathbf{Exp}^2)$

1 : **if** $\exists K : (m, K) \in \Pi^H$ **then**
2 : **return** K
3 : $K \leftarrow\$ \mathcal{K}$
4 : $c' := \mathsf{Enc}(\mathsf{pk}, m)$
5 : **if** $\exists K' : (c', K') \in \Pi^D$ **then**
6 : $K := K'$
7 : **else**
8 : $\Pi^D := \Pi^D \cup \{(c', K)\}$
9 : $\Pi^H := \Pi^H \cup \{(m, K)\}$
10 : **return** K

DECAPS$(c \neq c^*)$ $(\mathbf{Exp}^2, \mathbf{Exp}^3)$

1 : **if** $\exists K : (c, K) \in \Pi^D$ **then**
2 : **return** K
3 : **else**
4 : $K \leftarrow\$ \mathcal{K}$
5 : $\Pi^D := \Pi^D \cup \{(c, K)\}$
6 : **return** K

In $\mathbf{Exp}^1$ for all correct ciphertexts c holds that $\textsc{Decaps}(c) = \mathsf{H}(\mathsf{Dec}(\mathsf{sk}, c))$. Now we show that the changes did not spoil this rule. First, note that both experiments return some random value if $\mathsf{Dec}(\mathsf{sk}, c) = \perp$. Another note is that that there couldn't be the pair (c, K) in the set Π^D before either the first query on c to the oracle $\textsc{Decaps}$ or the query on $m' := \mathsf{Dec}(\mathsf{sk}, c)$ to the oracle H.

Now let us analyze two cases separately: in the first one the adversary $\mathcal{B}$ first queries H on m' and then queries $\textsc{Decaps}$ on c, in the second one it reverses the queries. If H is queried on m' first, at the very moment the pair (m', K) for $K \leftarrow\!\$\ \mathcal{K}$ is added to Π^H and the pair $(c' := \mathsf{Enc}(\mathsf{pk}, m'), K)$ is added to Π^D. As PKE is rigid it holds that $c' = c$ and hence $\textsc{Decaps}(c) = K = \mathsf{H}(m')$. If $\textsc{Decaps}$ is queried on c first, the pair (c, K) for $K \leftarrow\!\$\ \mathcal{K}$ is added to Π^D and this sets $\textsc{Decaps}(c) = K$. Thus, when H is queried on m' afterwards, the condition on the line 5 of listing of H will be satisfied and then the pair (m', K) will be added to Π^H wherefore $K = \mathsf{H}(m')$.

Consequently we have

$$\mathbb{P}[\mathbf{Exp}^2(\mathcal{B}) \Rightarrow 1] = \mathbb{P}[\mathbf{Exp}^1(\mathcal{B}) \Rightarrow 1].$$

Finally, the experiment $\mathbf{Exp}^3$ differs from experiment $\mathbf{Exp}^2$ in that it immediately aborts (with uniformly random output) after $\mathcal{B}$'s query to H on m^*.

$\mathsf{H}(m)$ $(\mathbf{Exp}^3)$

1 : **if** $(m = m^*) \wedge (c^*$ is defined$)$ **then**
2 : **abort**
3 : **if** $\exists K : (m, K) \in \Pi^H$ **then**
4 : **return** K
5 : $K \leftarrow\!\$\ \mathcal{K}$
6 : $c' := \mathsf{Enc}(\mathsf{pk}, m)$
7 : **if** $\exists K' : (c', K') \in \Pi^D$ **then**
8 : $K := K'$
9 : **else**
10 : $\Pi^D := \Pi^D \cup \{(c', K)\}$
11 : $\Pi^H := \Pi^H \cup \{(m, K)\}$
12 : **return** K

We denote the event that the corresponding condition from the line 1 of listing of H is fulfilled by CHAL. Then

$$\left|\mathbb{P}[\mathbf{Exp}^3(\mathcal{B}) \Rightarrow 1] - \mathbb{P}[\mathbf{Exp}^2(\mathcal{B}) \Rightarrow 1]\right| \leqslant \mathbb{P}[\mathsf{CHAL}].$$

So in $\mathbf{Exp}^3$ we avoid the adversary from asking the oracle H queries on m^*. As queries to $\textsc{Decaps}$ on c^* are excluded by definition, $\mathcal{B}$ has no ability to get any information about $\mathsf{H}(m^*)$ and we can claim bit b is independent from $\mathcal{B}$'s view. This gives us

$$\mathbb{P}[\mathbf{Exp}^3(\mathcal{B}) \Rightarrow 1] = \frac{1}{2}.$$

Let us construct the adversary $\mathcal{A}$ against Niederreiter cryptosystem in the $\mathsf{OW\text{-}CPA}$ model that simulates $\mathbf{Exp}^3$ for the adversary $\mathcal{B}$.

$\mathcal{A}(\mathsf{pk}, c^*)$

1 : $K^* \leftarrow\!\$\ \mathcal{K}$
2 : $b' \leftarrow \mathcal{B}^{\textsc{Decaps},\mathsf{H}}(\mathsf{pk}, c^*, K^*)$
3 : **if** $\exists (m', K') \in \Pi^H$:
4 : $\mathsf{Enc}(\mathsf{pk}, m') = c^*$ **then**
5 : **return** m'
6 : **else**
7 : **abort**

This simulation is perfect if CHAL doesn't occur. If it does, then the message m^*, corresponding to the ciphertet c^*, is correctly processed and holds $(m^*, K') \in \Pi^H$ for some K'. Note that, since PKE is deterministic, m^* always follows $\mathsf{Enc}(\mathsf{pk}, m^*) = c^*$ that is condition on the line 4 of listing of $\mathcal{A}$ is fulfilled. Hence,

$$\mathbb{P}[\mathsf{CHAL}] = \mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{Nieder}}(\mathcal{A}).$$

The statement of the theorem is obtained by collecting the probabilities. □

Further we state the theorem on the security of $\widetilde{\mathsf{KEM}}$ in QROM without proof, since it follows right from Theorem 2 (the bound for $\mathsf{U}^{\not\perp}$ transformation) and Theorem 5 (the equivalence of bounds for $\mathsf{U}^{\not\perp}$ and $\mathsf{U}^{\not\perp}_m$ transformations) from the article [63]. It is also useful to mention that the perfect correctness implies zero advantage in "finding failing ciphertexts" experiment set in [63, Definition 3]. And, finally, being deterministic the Niederreiter cryptosystem is 0-injective [63, Definition 6]. Putting together all the comments we claim Theorem 3.

Theorem 3 *Assume* Niederreiter PKE *to be perfectly correct. For any* $\mathsf{IND\text{-}CCA}$ *adversary* $\mathcal{B}$ *against* $\widetilde{\mathsf{KEM}}$*, issuing at most* q_D *queries to the decapsulation oracle* $\textsc{Decaps}$ *and at most* q_H *quantum queries to the random oracle* H*, there exist an* $\mathsf{OW\text{-}CPA}$ *adversary* $\mathcal{A}$ *against Niederreiter* PKE *and an adversary* $\mathcal{A}'$ *against the security of* PRF F *with at most* q_D *quires such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\widetilde{\mathsf{KEM}}}(\mathcal{B}) \leqslant 2\sqrt{\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{Nieder}}(\mathcal{A})} + 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathsf{F}}(\mathcal{A}')$$

and adversaries $\mathcal{A}$ *and* $\mathcal{A}'$ *are running in about the same time and resources as* $\mathcal{B}$*.*

9 Discussion

Further investigation is required to determine the most suitable code for the proposed scheme. Despite the fact that, based on a preliminary analysis, we have so far proposed using Goppa codes known to be the basis of secure and perfectly correct cryptosystems, this option may not be final. Alternative variants such as quasi-cyclic Goppa codes or subcodes of algebraic geometry codes should also be considered. However, a thorough assessment of the security of these codes is necessary. It is important to note that the selection of the code will have a direct impact on the scheme's parameters and performance characteristics in future applications.

10 Conclusion

In this article, we gathered fundamental questions that need to be addressed by anyone considering synthesizing KEM based on error-correcting codes. We described the most well-known code-based cryptosystems, along with their advantages and disadvantages, and discussed approaches that enable to transform these cryptosystems into secure KEMs. Furthermore, we explored the features of schemes depending on different classes of codes. Additionally, we specified two best options to construct a scheme with the best security estimates and for one of them provided a proof of security in ROM model and a statement of security in QROM model.

This work was driven by the observation that there is currently a significant number of proposals emerging worldwide due to the ongoing standardization process of KEMs. However, these proposals often lack the rationale of choosing one solution over another. We believe it is valuable to consolidate all the discussions on this topic into a single resource, allowing researchers to use this article as a reference for synthesizing such schemes. In addition, this work can serve as a foundation for a code-based KEM standard in Russia.

Acknowledgements The authors thank Liliya Akhmetzyanova for valuable comments on the work

Funding No funding was received to assist with the preparation of this manuscript.

Declarations

Conflict of interest The authors have no Conflict of interest to declare that are relevant to the content of this article.

References

1. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997). https://doi.org/10.1137/s0097539795293172
2. Berlekamp, E.R., McEliece, R.J., van Tilborg, H.C.A.: On the inherent intractability of certain coding problems. IEEE Trans. Inf. Theory **24**(3), 384–386 (1978). https://doi.org/10.1109/TIT.1978.1055873
3. Barg, S.: Some new NP-complete coding problems. Probl. Peredachi Inf. **30**(3), 23–28 (1994). https://doi.org/10.18287/0134-2452-2015-39-4-459-461.
4. Both, L., May, A.: Decoding linear codes with high error rate and its impact for LPN security. In: Post-Quantum Cryptography. PQCrypto 2018. LNCS, vol. 10786, pp. 25–46 (2018). https://doi.org/10.1007/978-3-319-79063-3_2
5. Hofheinz, D., Hovelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). LNCS, vol. 10677, pp. 341–371 (2017). https://doi.org/10.1007/978-3-319-70500-2_12
6. Bernstein, D.J., Chou, T., Cid, C., Gilcher, J., Lange, T., Maram, V., von Maurich, I., Misoczki, R., Niederhagen, R., Persichetti, E., Peters, C., Sendrier, N., Szefer, J., Tjhai, C.J., Tomlinson, M., Wang, W.: Classic McEliece: conservative code-based cryptography: cryptosystem specification. NIST proposal, (October), (2022)
7. Albrecht, M., Cid, K., Paterson, K.G., Tjhai, C.J., Tomlinson, M.: NTS-KEM. Technical report (2017)
8. Gligoroski, D.: Post-quantum Key Encapsulation Mechanism EDON-K. NIST proposal, pp. 1–42 (2017)
9. Bardet, M., Barelli, E., Blazy, O., Torres, RC., Couvreur, A., Gaborit, P., Otmani, A., Sendrier, N., Tillich, J.P.: BIG QUAKE: BInary Goppa QUAsi-cyclic Key Encapsulation. Technical report (2017). https://bigquake.inria.fr/
10. Faugere, J.C., Gauthier-Umana, V., Otmani, A., Perret, L., Tillich, J.P.: A distinguisher for high-rate McEliece cryptosystems. IEEE Trans. Inf. Theory **59**(10), 6830–6844 (2013). https://doi.org/10.1109/TIT.2013.2272036
11. Strenzke, F., Erik T.H., Molter, G., Overbeck, R., Shoufan, A.: Side channels in the McEliece PKC. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5299 LNCS (May 2014), pp. 216–229 (2008). https://doi.org/10.1007/978-3-540-88403-3_15
12. Shoufan, A., Strenzke, F., Molter, G.H., Stöttinger, M.: A timing attack against Patterson algorithm in the McEliece PKC. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5984 LNCS, pp. 161–175 (2010). https://doi.org/10.1007/978-3-642-14423-3_12
13. Avanzi, R.M., Hoerder, S., Page, D., Tunstall, M.: Side-channel attacks on the McEliece and Niederreiter public-key cryptosystems. J. Cryptogr. Eng. **1**, 271–281 (2011)
14. Strenzke, F.: A timing attack against the secret permutation in the McEliece PKC. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6061 LNCS, pp. 95–107 (2010). https://doi.org/10.1007/978-3-642-12929-2_8
15. Bucerzan, D., Cayrel, P.L., Dragoi, V.: Improved timing attacks against the secret permutation in the mceliece PKC. Int. J. Comput. Commun. Control **12**(1), 7–25 (2017). https://doi.org/10.15837/ijccc.2017.1.2780
16. Molter, H.G., Stottinger, M., Shoufan, A., Strenzke, F.: A simple power analysis attack on a McEliece cryptoprocessor. J. Cryptogr. Eng. **1**(1), 29–36 (2011). https://doi.org/10.1007/s13389-011-0001-3
17. Colombier, B., Dragoi, V.-F., Cayrel, P.-L., Grosso, V.: Profiled side-channel attack on cryptosystems based on the binary syndrome decoding problem. IACR Cryptology ePrint Archive, pp. 1–14 (2022)

18. Cayrel, P.L., Colombier, B., Dragoi, V.F., Menu, A., Bossuet, L.: Message-recovery laser fault injection attack on the classic McEliece cryptosystem. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12697 LNCS, pp. 438–467 (2021). https://doi.org/10.1007/978-3-030-77886-6_15
19. Lahr, N., Niederhagen, R., Petri, R., Samardjiska, S.: Side channel information set decoding using iterative chunking. In: Advances in Cryptology—ASIACRYPT,: ASIACRYPT 2020. Lecture Notes in Computer Science 12491, pp. 881–910 (2020). https://doi.org/10.1007/978-3-030-64837-4_29
20. Banegas, G., Barreto, P.S.L.M., Boidje, B.O., Cayrel, P.L., Dione, G.N., Gaj, K., Gueye, C.T., Haeussler, R., Klamti, J.B., N'diaye, O., Nguyen, D.T., Persichetti, E., Ricardini, J.E.: DAGS: key encapsulation using dyadic GS codes. Technical report, (2017)
21. Persichetti, E.: Compact McEliece keys based on quasi-dyadic Srivastava codes. J. Math. Cryptol. **6**(2), 149–169 (2012). https://doi.org/10.1515/jmc-2011-0099
22. Wang, Y.: RLCE Key Encapsulation Mechanism (RLCE-KEM) Specification. NIST proposal, pp. 1–64 (2017)
23. Couvreur, A., Lequesne, M., Tillich, J.P.: Recovering short secret keys of RLCE in polynomial time. Lect. Notes Comput. Sci. **11505**, 133–152 (2019). https://doi.org/10.1007/978-3-030-25510-7_8
24. Couvreur, A., Márquez-Corbella, I., Pellikaan, R.: Cryptanalysis of public-key cryptosystems that use subcodes of algebraic geometry codes. Coding Theory Appl. CIM Ser. Math. Sci. **3**, 133–140 (2015). https://doi.org/10.1007/978-3-319-17296-5_13
25. Sidelnikov, V.M., Shestakov, S.O.: On insecurity of cryptosystems based on generalized Reed–Solomon codes. Discrete Math. Appl. **2**(4), 439–444 (1992). https://doi.org/10.1515/dma.1992.2.4.439
26. Couvreur, A., Gaborit, P., Gauthier-Umaña, V., Otmani, A., Tillich, J.-P.: Distinguisher-based attacks on public-key cryptosystems using Reed–Solomon codes. Des. Codes Cryptogr. **73**(2), 641–666 (2014). arXiv:1307.6458
27. Melchor, C.A., Gaborit, N., , Limoges, P., Bettaieb, J., Persichetti, E., Bidoux, L., Robert, J.-M., Blazy, O., Véron, P., Bos, J., Zémor, G., Deneuville, J.-C.: Hamming Quasi-Cyclic (HQC). Technical report, (2019)
28. Melchor, C.A., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Zémor, G.: Rank Quasi-Cyclic (RQC). Technical report, (2017)
29. Aragon, N., Barreto, P., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J., Gaborit, P., Gueron, S., Guneysu, T., Melchor, C.A., Misoczki, R., Persichetti, E., Sendrier, N., Tillich, J.-P., Vasseur, V., Zémor, G.: Bike: Bit Flipping Key Encapsulation—Round 3 Submission. Technical report, (2021)
30. Eaton, E., Parent, A.: QC-MDPC KEM: a key encapsulation mechanism based on the QC-MDPC McEliece encryption scheme. Technical report, (2017)
31. Baldi, M., Chiaraluce, F., Pelosi, G., Santini, P.: LEDAkem: a post-quantum key encapsulation mechanism based on QC-LDPC codes. NIST proposal, pp. 1–22 (2017)
32. Yu, Y., Zhang, J.: Lepton: key encapsulation mechanisms from a variant of learning parity with noise. Technical report, (2017)
33. Eaton, E., Lequesne, M., Parent, A., Sendrier, N.: QC-MDPC: a timing attack and a CCA2 KEM. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10786 LNCS (645622), pp. 47–76 (2018). https://doi.org/10.1007/978-3-319-79063-3_3
34. Von Maurich, I., Guneysu, T.: Towards side-channel resistant implementations of QC-MDPC McEliece encryption on constrained devices. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 8772, pp. 266–282 (2014). https://doi.org/10.1007/978-3-319-11659-4_16
35. Paiva, T.B.,Terada, R.: A timing attack on the HQC encryption scheme. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11959 LNCS (442014), pp. 551–573 (2020). https://doi.org/10.1007/978-3-030-38471-5_22
36. Wafo-Tapa, G., Bettaieb, S., Bidoux, L., Gaborit, P., Marcatel, E.: A practicable timing attack against HQC and its countermeasure. Adv. Math. Commun. **16**(3), 621–642 (2022). https://doi.org/10.3934/amc.2020126
37. Guo, Q., Hlauschek, C., Johansson, T., Lahr, N., Nilsson, A., Schröder, R.L.: Don't reject this: key-recovery timing attacks due to rejection-sampling in HQC and BIKE. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2022**(3), 223–263 (2022). https://doi.org/10.46586/tches.v2022.i3.223-263
38. Chen, C., Eisenbarth, T., Von Maurich, I., Steinwandt, R.: Differential power analysis of a McEliece cryptosystem. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9092, pp. 538–556 (2015). https://doi.org/10.1007/978-3-319-28166-7_26
39. Chen, C., Eisenbarth, T., Von Maurich, I., Steinwandt, R.: Horizontal and vertical side channel analysis of a McEliece cryptosystem. IEEE Trans. Inf. Forensics Secur. **11**(6), 1093–1105 (2016). https://doi.org/10.1109/TIFS.2015.2509944
40. Rossi, M., Hamburg, M., Hutter, M., Marson, M.E: A side-channel assisted cryptanalytic attack against QcBits. In: Cryptographic Hardware and Embedded Systems-CHES 2017-19th International Conference, pp. 3–23 (2017)
41. Sim, B.Y., Kwon, J., Choi, K.Y., Cho, J., Park, A., Han, D.G.: Novel side-channel attacks on quasi-cyclic code-based cryptography. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(4), 180–212 (2019). https://doi.org/10.13154/tches.v2019.i4.180-212
42. Schamberger, T., Renner, J., Sigl, G., Wachter-Zeh, A.: A power side-channel attack on the CCA2-Secure HQC KEM. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12609 LNCS, pp. 119–134 (2021). https://doi.org/10.1007/978-3-030-68487-7_8
43. Ueno, R., Xagawa, K., Tanaka, Y., Ito, A., Takahashi, J., Homma, N.: Curse of re-encryption: a generic power/EM analysis on post-quantum KEMs. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2022**(1), 296–322 (2021). https://doi.org/10.46586/tches.v2022.i1.296-322
44. Guo, Q., Johansson, T., Stankovski, P.: A key recovery attack on MDPC with CCA security using decoding errors. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10031 LNCS, pp. 789–815 (2016). https://doi.org/10.1007/978-3-662-53887-6_29
45. Fabsic, T., Hromada, V., Zajac, P.: A reaction attack on LEDApkc. IACR Cryptology ePrint Archive, pp. 1–12 (2018)
46. Santini, P., Battaglioni, M., Chiaraluce, F., Baldi, M.: Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11666 LNCS, pp. 115–136 (2019). https://doi.org/10.1007/978-3-030-25922-8_7
47. Sendrier, N.: Decoding one out of many. A World of Difference, pp. 257–294 (2008). https://doi.org/10.1007/978-1-137-11037-4_15
48. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. Deep Space Netw. Prog. Rep. **42**(44), 114–116 (1978)
49. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Probl. Control Inf. Theory **15**(2), 159–166 (1986)
50. Aragon, N., Hauteville, A., Blazy, O., Ruatta, O., Deneuville, J.-C., Gaborit, P., Zémor, G., Gaborit, P., Hauteville, A.: LOCKER–LOw rank parity CheckK codes EncRyption. Technical report (2017)

51. Elia, M., Viterbo, E., Bertinetti, G.: Decoding of binary separable Goppa codes using Berlekamp–Massey algorithm. Electron. Lett. **35**(20), 1720–1721 (1999)
52. Bellare, M., Rogaway, P.: Optimal asymmetric encryption. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics 950), pp. 92–111 (1995). https://doi.org/10.1007/bfb0053428
53. Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimum cost. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 1560, pp. 53–68 (1999). https://doi.org/10.1007/3-540-49162-7_5
54. Pointcheval, D.: Chosen-ciphertext security for any one-way cryptosystem. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 1751, pp. 129–146 (2000). https://doi.org/10.1007/978-3-540-46588-1_10
55. Kobara, K., Imai, H.: Semantically secure mceliece public-key cryptosystems—conversions for McEliece PKC-. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 1992, pp. 19–35 (2001). https://doi.org/10.1007/3-540-44586-2_2
56. Fujusaki, E., Okamoto, T.: Secure inregration of asymmetric and symmetric encryption schemes. LNCS.CRYPTO'99, pp. 537–554 (1999)
57. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptol. **26**, 80–101 (2013). https://doi.org/10.1007/s00145-011-9114-1
58. Dent, A.W.: A designer's guide to KEMs. Lect. Notes Comput. Sci. **2898**, 133–151 (2003)
59. Bernstein, D.J., Persichetti, E.: Towards KEM unification. IACR Cryptology ePrint Archive, pp. 1–37 (2018)
60. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10993 LNCS, Tcc, 2017, pp. 96–125, 2018. https://doi.org/10.1007/978-3-319-96878-0_4
61. Saito, T., Xagawa, K., Yamakawa, T.: Tightly-secure key-encapsulation mechanism in the quantum random oracle model. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10822 LNCS, pp. 520–551 (2018). https://doi.org/10.1007/978-3-319-78372-7_17
62. Jiang, H., Zhang, Z., Ma, Z.: Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. LNCS **11505**, 227–248 (2019). https://doi.org/10.1007/978-3-030-25510-7_13
63. Bindel, N., Hamburg, M., Hövelmanns, K., Hülsing, A., Persichetti, E.: Tighter proofs of CCA security in the quantum random oracle model. LNCS **11892**, 61–90 (2019)
64. Banegas, G., Barreto, P.S.L.M., Boidje, B.O., Cayrel, P.L., Dione, G.N., Gaj, K., Gueye, C.T., Haeussler, R., Klamti, J.B., N'diaye, O., Nguyen, D.T., Persichetti, E., Ricardini, J.E.: DAGS: Key encapsulation using dyadic GS codes. J. Math. Cryptol. **12**(4), 221–239 (2018). https://doi.org/10.1515/jmc-2018-0027