TRIKE: Triple Quasi-Cyclic Code-Based Key Encapsulation Mechanism

Abstract. We introduce the ratio quasi-cyclic codeword finding (RQCCF) problem, a novel variant of the syndrome decoding problem, and show that its security can be reduced to that of the 3-quasi-cyclic codeword finding (3-QCCF) problem. Based on this hardness assumption, we propose TRIKE, a new code-based key encapsulation mechanism (KEM). TRIKE adopts a more accurate decryption failure rate (DFR) model, with observed failure rates closely matching the theoretical predictions. It also applies a rejection procedure to mitigate the contribution of the analyzed weak-key classes. TRIKE offers the smallest public keys among code-based KEMs. At NIST Security Levels 1, 3, and 5, it reduces the total encapsulation and decapsulation time by at least 33% compared with BIKE, while maintaining a combined public-key and ciphertext size comparable to that of BIKE.

Keywords: Post-Quantum Cryptography · Code-Based Cryptography · TRIKE · Ratio Quasi-Cyclic Codeword Finding

1 Introduction

In recent decades, the rapid development of quantum computing and quantum algorithms has posed significant challenges to classical cryptographic systems. In particular, the advent of Shor's algorithm renders widely deployed public-key schemes such as RSA insecure against quantum adversaries. To address this threat, the National Institute of Standards and Technology (NIST) launched a standardization process for post-quantum cryptography (PQC) [1], and code-based cryptography has emerged as one of the most promising foundations for standardized key encapsulation mechanisms.

Among code-based candidates, BIKE [5] offers one of the most efficient designs. Its public key is a single ratio $q = h_1 h_0^{-1}$ of two sparse secret polynomials, and the resulting key and ciphertext sizes scale linearly with the block length. However, this ratio-based design tightly couples security and efficiency. At a fixed block length, the security of the public ratio is governed by the weight of the secret components, so reaching a higher security level requires heavier secrets. Since decapsulation is dominated by parity-check updates whose cost grows with the secret weight, higher security directly translates into slower decoding. The decoding-failure constraint forces the block length up accordingly, inflating both keys and ciphertexts.

1.1 Our Approach

In this work we introduce a new framework that relaxes this coupling. Our starting point is the *Ratio Quasi-Cyclic Codeword Finding* (RQCCF) problem, a variant of the classical Syndrome Decoding problem in which the public ratio is formed from a *linear combination of three* sparse secret polynomials, combining all secret components into a single public ring element. We show that an efficient distinguisher for RQCCF yields an efficient distinguisher for the decisional 3-QCCF problem, establishing its hardness. A naive three-component extension of BIKE, i.e., publishing two separate ratios $q_1 = h_1 h_0^{-1}$ and $q_2 = h_2 h_0^{-1}$, would expose the relations $q_1 h_0 + h_1 = 0$ and $q_2 h_0 + h_2 = 0$, which together define a 2×3 QCCF instance whose attack cost is lower at the same block length and secret weight. Compensating for this would require heavier secrets and would defeat the purpose of adding a third component. The mixed relation of RQCCF avoids exposing such additional constraints while still deriving security from three-QCCF-type assumptions.

Building on RQCCF, we construct **TRIKE**, a code-based key encapsulation mechanism whose ciphertext consists of two components formed from three sparse error terms. With the secret key, the receiver combines them into a syndrome of the form $s = h_0 e_0 + h_1 e_1 + h_2 e_2$, which is then decoded by a sparse parity-check by employing a bit-flipping-like decoder. Because RQCCF permits a substantially lower weight per secret component at a given security level, the parity-check updates and unsatisfied-check counting in the decoder involve far fewer nonzero entries, which translates directly into faster decapsulation. Together with the DFR analysis described below, the reduced weights also allow shorter block lengths, yielding a compact public key.

Realizing these gains requires reliable decoding-failure rate (DFR) estimates far below what is measurable in simulation. Because the extrapolation relation determines the block length needed to meet the target DFR, the accuracy of this model feeds directly into the communication and computational cost of the scheme. We adopt a finite-length scaling (FLS) model motivated by iteratively decoded LDPC codes, which transforms the DFR curve so that it can be fitted at experimentally accessible block lengths and extrapolated to the target parameters. For our representative parameter set this model fits the measured data noticeably better than the linear fit of the logarithmic DFR used by BIKE. We further address weak keys whose decoding behavior departs from the typical trend: TRIKE applies a hybrid rejection procedure combining a 4-cycle statistic within each circulant block with a cyclic-intersection test across blocks [23,24]. For our parameter sets the overall rejection rate is below 1%. The DFR model and the key-validation procedure thus address complementary aspects of decoding reliability.

Contributions.

- **A new hardness problem and KEM (RQCCF and TRIKE).** We define the RQCCF problem, reduce its decisional version to 3-QCCF, and build TRIKE on it. A single public ratio over three sparse secrets replaces

BIKE's two-secret ratio, and a two-component ciphertext still decodes to a syndrome via the secret key.

- **Better efficiency at NIST Levels 1, 3, and 5.** The lower secret weights enabled by RQCCF reduces the block length and Hamming weights of the scheme. Although TRIKE performs additional ring arithmetic during key generation and encapsulation, the decoding savings dominate, cutting the combined encapsulation and decapsulation time by more than 33% compared to BIKE, while keeping the combined public-key and ciphertext size comparable to that of BIKE.
- **Improved DFR prediction and weak-key handling.** An FLS model gives more accurate extrapolation than the model used in BIKE, and a hybrid rejection step removes the analyzed weak-key classes at a cost of below 1% rejected keys. Besides, the residual DFR contribution is quantified rather than assumed away.
- **Optimized implementations.** We instantiate TRIKE at the three NIST levels and evaluate all tradeoffs through optimized implementations, confirming the theoretical savings in practice.

2 Preliminaries

2.1 Notions

The following notations will be used in this paper.

- Let $\mathbb{F}_q$ be the finite field with q elements, and $\mathbb{F}_q^n$ be the set of vectors where each element is in $\mathbb{F}_q$.
- For a vector $y = (y_0, \cdots, y_{n-1}) \in \mathbb{F}_2^n$, denote $w_H(y)$ to be the Hamming weight of y.
- Let $\star$ be the vector operator Schur product. For vector $u, v \in \mathbb{F}_2^n$, the result of $u \star v$ is defined as

$$u \star v = (u_0 \cdot v_0, u_1 \cdot v_1, \cdots, u_{n-1} \cdot v_{n-1})$$

- Let $\mathcal{R} \coloneqq \mathbb{F}_2[x]/(x^r - 1)$. Every element $y \in \mathcal{R}$ has a unique representative $y(x) = y_0 + y_1 x + \cdots + y_{r-1}x^{r-1}$ of degree less than r. We identify y with its coefficient vector $y = (y_0, \ldots, y_{r-1}) \in \mathbb{F}_2^r$ whenever no ambiguity arises. A circulant matrix corresponding to the coefficients of y can be defined as

$$\mathtt{rot}(y) = \begin{pmatrix} y_0 & y_{r-1} & \cdots & y_1 \\ y_1 & y_0 & \cdots & y_2 \\ \vdots & \vdots & \ddots & \vdots \\ y_{r-1} & y_{r-2} & \cdots & y_0 \end{pmatrix}. \tag{1}$$

  Then for any $y, z \in \mathcal{R}$, the coefficient vector of yz equals $\mathtt{rot}(y) \cdot z$. Moreover, the i-th column of $\mathtt{rot}(y)$ is denoted by $\mathtt{Col}_i(y)$.

– Suppose i is co-prime to r, then the map

$$\phi_i : y(x) \to y(x^i)$$

defines an isomorphism of $\mathcal{R}$ to $\mathcal{R}$. Particularly, this isomorphism preserves the Hamming weight.
– To describe the spectrum of elements in $\mathcal{R}$, the distance between two positions $i, j \in \{0, \cdots, r-1\}$ is defined as $d(i,j) = \min\{\pm(j-i) \bmod r\}$. For a polynomial $h \in \mathcal{R}$, the multiplicity of a distance δ counts the number of pairs of its nonzero coefficients separated by δ, which is formally expressed as $\mu(\delta, h) = |\{(i,j) \mid d(i,j) = \delta, h_i = h_j = 1, 0 \le i \le j < r\}|$. Then, the spectrum of h is defined as the set of all valid distances alongside their corresponding multiplicities, given by $\mathrm{Spec}(h) = \{(\delta, \mu(\delta, h)) \mid \delta \in \{1, \cdots, \lfloor r/2 \rfloor\}\}$.
– Let $\mathcal{R}_{odd} = \{y \in \mathcal{R} \mid w_H(y) \equiv 1 \mod 2\}$, $\mathcal{R}_{even} = \{y \in \mathcal{R} \mid w_H(y) \equiv 0 \mod 2\}$. [1]
– Denote
  • $\mathrm{K}(w) \coloneqq \{(h_0, h_1, h_2) \in \mathcal{R}^3 \mid w_H(h_0) = w_H(h_1) = w_H(h_2) = w/3\}$,
  • $\mathrm{E}(t) \coloneqq \{(e_0, e_1, e_2) \in \mathcal{R}^3 \mid w_H(e_0) + w_H(e_1) + w_H(e_2) = t\}$,

2.2 Security Definitions

In this paper, we employ four security notions: OW-CPA, OW-PCA and IND-CPA for PKE, and IND-CCA for KEM, respectively.

Definition 1 (One-Wayness under Chosen Plaintext Attacks (OW-CPA)). *Let* $\mathrm{PKE} = (\mathrm{KeyGen}, \mathrm{Encrypt}, \mathrm{Decrypt})$ *be a public-key encryption scheme with message space* M, $\mathcal{P}_{\mathrm{CO}}$ *be a plaintext-checking oracle such that*

$$\mathcal{P}_{\mathrm{CO}}(m, c) = (\mathrm{Decrypt}(\mathsf{sk}, c) = m).$$

For a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$*, the* OW-CPA *game is defined as:*

Game $G_{\mathsf{OW\text{-}CPA}}(\mathcal{A})$
1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KeyGen}()$
2. $m^* \xleftarrow{\$} \mathrm{M}$
3. $c^* \leftarrow \mathrm{Encrypt}(\mathsf{pk}, m^*)$
4. $m' \leftarrow \mathcal{A}(\mathsf{pk}, c^*)$
5. return $\mathcal{P}_{\mathrm{CO}}(m', c^*)$

[1] Since r is prime and 2 is primitive modulo r, every element of $\mathcal{R}_{odd}$ is invertible except $\Phi_r(x) = 1 + x + \cdots + x^{r-1}$. A uniformly random element of $\mathcal{R}_{odd}$ equals this exceptional element with probability $2^{-(r-1)}$. Throughout this paper, whenever this exceptional case occurs in a division, the corresponding randomness is resampled. This changes the relevant distributions by at most a negligible statistical distance of $O(2^{-r})$, which is omitted from the advantage bounds for readability.

The advantage of the OW-CPA *adversary* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}}(\mathcal{A}) = \Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A})].$$

Definition 2 (One-Wayness under Plaintext Checking Attacks (OW-PCA)). *Let* $\mathrm{PKE} = (\mathrm{KeyGen}, \mathrm{Encrypt}, \mathrm{Decrypt})$ *be a public-key encryption scheme with message space* M*,* $\mathcal{P}_{\mathrm{CO}}$ *be a plaintext-checking oracle, as defined above.*

For a PPT adversary $\mathcal{A}$ *that is allowed to query the plaintext-checking oracle* $\mathcal{P}_{\mathrm{CO}}$*, the* OW-PCA *game is defined as:*

Game $G_{\mathsf{OW\text{-}PCA}}(\mathcal{A})$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KeyGen}()$
2. $m^* \xleftarrow{\$} \mathrm{M}$
3. $c^* \leftarrow \mathrm{Encrypt}(\mathsf{pk}, m^*)$
4. $m' \leftarrow \mathcal{A}^{\mathcal{P}_{\mathrm{CO}}}(\mathsf{pk}, c^*)$
5. *return* $\mathcal{P}_{\mathrm{CO}}(m', c^*)$

The advantage of the OW-PCA *adversary* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathrm{PKE}}(\mathcal{A}) = \Pr[G_{\mathsf{OW\text{-}PCA}}(\mathcal{A})].$$

Definition 3 (Indistinguishability under Chosen Plaintext Attacks (IND-CPA)). *Let* $\mathrm{PKE} = (\mathrm{KeyGen}, \mathrm{Encrypt}, \mathrm{Decrypt})$ *be a public-key encryption scheme. For a PPT adversary* $\mathcal{A} = (\mathcal{A}_{choose}, \mathcal{A}_{guess})$*, the* IND-CPA *game is defined as:*

Game $G_{\mathsf{IND\text{-}CPA}}(\mathcal{A})$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KeyGen}()$
2. $(m_0, m_1) \leftarrow \mathcal{A}_{choose}(\mathsf{pk})$
3. $b \xleftarrow{\$} \{0, 1\}$
4. $c^* \leftarrow \mathrm{Encrypt}(\mathsf{pk}, m_b)$
5. $b' \leftarrow \mathcal{A}_{guess}(\mathsf{pk}, c^*)$
6. *return* $(b = b')$

The advantage of the IND-CPA *adversary* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}}(\mathcal{A}) = \left| \Pr[G_{\mathsf{IND\text{-}CPA}}(\mathcal{A})] - \frac{1}{2} \right|.$$

Definition 4 (Indistinguishability under Chosen Ciphertext Attacks (IND-CCA)[2]). *Let* $\mathrm{KEM} = (\mathrm{KeyGen}, \mathrm{Encaps}, \mathrm{Decaps})$ *be a key encapsulation*

[2] Although we refer to our security notion as IND-CCA, it is equivalent to the standard IND-CCA2 definition, since the decapsulation oracle remains accessible after the challenge ciphertext is given.

mechanism. For a PPT adversary $\mathcal{A}$ with a decapsulation oracle $\mathcal{O}_{\text{Decaps}}(c)$ such that $c \neq c^*$, the IND-CCA *game is defined as:*

Game $G_{\mathsf{IND\text{-}CCA}}(\mathcal{A})$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \text{KeyGen}()$
2. $b \xleftarrow{\$} \{0,1\}$
3. $(k_0, c^*) \leftarrow \text{Encaps}(\mathsf{pk})$
4. $k_1 \xleftarrow{\$} \text{M}$
5. $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{Decaps}}}(\mathsf{pk}, k_b, c^*)$
6. *return* $(b = b')$

The advantage of the IND-CCA *adversary $\mathcal{A}$ is defined as:*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\text{KEM}}(\mathcal{A}) = \left| \Pr[G_{\mathsf{IND\text{-}CCA}}(\mathcal{A})] - \frac{1}{2} \right|.$$

3 Specification

3.1 Setup

Input: The security level λ
Output: Parameters $\{r, w, t, l\}$, hash functions $\mathtt{H}_1, \mathtt{H}_2, \mathtt{H}_3, \mathtt{H}_4, \mathtt{K}, \mathtt{L}$ and a decoder.
System Parameters.

- r is the block length such that r is a prime number and 2 is a primitive root module r, where $n = 3r$ is the code length
- w is the total weight of secret vectors, where $d = w/3$ is an odd number
- t is the total weight of error vectors
- l is the length of seed and message

Hash Functions. We define the seed or message space as $\text{M} = \{0,1\}^l$. Then we select hash functions uniformly at random with the following domains and ranges.

- $\mathtt{H}_1 : \text{M} \to \mathcal{R}_{even}$
- $\mathtt{H}_2 : \text{M} \to \mathcal{R}_{even}$
- $\mathtt{H}_3 : \text{M} \to \mathcal{R}_{odd}$
- $\mathtt{H}_4 : \text{M} \times \mathcal{R}_{odd} \to \text{E}(t)$
- $\mathtt{K} : \text{M} \times \mathcal{R} \times \mathcal{R} \times \text{M} \to \text{M}$
- $\mathtt{L} : \text{E}(t) \to \text{M}$

Decoder. The TRIKE bit-flipping decoder is discussed in Section 3.3.

Public Key Encryption. The underlying PKE primitive has plaintext space $\text{E}(t)$ and consists of the following procedures. Its full pseudocode is provided in Appendix B.1.

– KeyGen(): Let $(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$, $\sigma \xleftarrow{\$} \mathrm{M}$, $t_1 \leftarrow \mathtt{H}_1(\sigma)$, $t_2 \leftarrow \mathtt{H}_2(\sigma)$, and $r_1 \leftarrow \mathtt{H}_3(\sigma)$. Calculate $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$ and $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$. Set $\mathsf{pk} \leftarrow (\sigma, r_2)$ and $\mathsf{sk} \leftarrow (\sigma, h_0, h_1, h_2)$, and output $(\mathsf{pk}, \mathsf{sk})$.
– Encrypt(pk,e): For $e = (e_0, e_1, e_2) \in \mathrm{E}(t)$, let $t_1 \leftarrow \mathtt{H}_1(\sigma)$, $t_2 \leftarrow \mathtt{H}_2(\sigma)$, and $r_1 \leftarrow \mathtt{H}_3(\sigma)$. Calculate $u \leftarrow e_0 + e_1 r_1 + e_2 r_2$ and $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$. Output $c \leftarrow (u, v)$.
– Decrypt(sk,c): Let $t_1 \leftarrow \mathtt{H}_1(\sigma)$ and $r_1 \leftarrow \mathtt{H}_3(\sigma)$. Calculate $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$ and $s \leftarrow (h_0 + t_0)u - t_0 v$. Compute $e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$ and output e'.

3.2 Key Encapsulation Mechanism

TRIKE KEM is constructed from its PKE using the Fujisaki-Okamoto (FO) transformation [14] and the HHK framework with implicit rejection [15]. The final KEM additionally applies the weak-key rejection test in Algorithm 2 during key generation, corresponding to $\mathrm{KEM}_{\$}$ analyzed in Section 4.5. The formal KEM procedures are provided in Appendix B.2.

3.3 Decoder

We adopt a bit-flipping (BF) decoder derived from the BIKE-Flipping decoder and adapted to the structure of TRIKE. The decoder is formalized in Algorithm 1.

Given the secret key (h_0, h_1, h_2) and an input syndrome s, the decoder maintains an estimated error vector $e' = (e'_0, e'_1, e'_2)$ and a current syndrome s', initially set to s.

At each iteration, a threshold T is computed as a function of the initial syndrome weight $w_H(s)$, the current syndrome weight $w_H(s')$, and the iteration number. The threshold selection rule is discussed separately below.

For every position $i \in \{0, \ldots, r-1\}$, the decoder evaluates a counter value for each circulant block. The counter corresponds to the number of unsatisfied parity-check equations involving the considered bit and is computed as $w_H\left(\mathtt{Col}_i(h_j) \star s'\right)$, where $\mathtt{Col}_i(h_j)$ denotes the i-th column of the circulant matrix generated by h_j, and $\star$ denotes the Schur product.

Whenever the counter associated with a bit position is greater than or equal to the threshold T, the corresponding bit of the estimated error vector is flipped. This procedure is applied independently to all three circulant blocks h_0, h_1, and h_2.

After all candidate flips have been processed, the syndrome is recomputed as

$$s' = s + h_0 e'_0 + h_1 e'_1 + h_2 e'_2.$$

The process is repeated for a fixed number of iterations $I_{\max}$. The final estimate e' is then returned as the decoding result.

Threshold Selection. The threshold used in Algorithm 1 depends on both the initial syndrome weight and the current syndrome weight. Let

$$w = w_H(s), \qquad w' = w_H(s').$$

Algorithm 1: TRIKE Bit-Flipping Decoder

Input: Secret key (h_0, h_1, h_2), syndrome s, maximum number of iterations $I_{\max}$, threshold coefficients c_a, c_b, δ
Output: Error vector e'

Initialize $e' \leftarrow 0, s' = s$
for it $= 1$ to $I_{\max}$ **do**
  $T = \mathsf{threshold}(w_H(s), w_H(s'), it, c_a, c_b, \delta)$
  for $i = 0$ to $r-1$ **do**
    if $w_H(\mathtt{Col}_i(h_0) \star s') \geq T$ **then**
      $e'_{0i} = e'_{0i} \oplus 1$
    end if
  end for
  for $i = 0$ to $r-1$ **do**
    if $w_H(\mathtt{Col}_i(h_1) \star s') \geq T$ **then**
      $e'_{1i} = e'_{1i} \oplus 1$
    end if
  end for
  for $i = 0$ to $r-1$ **do**
    if $w_H(\mathtt{Col}_i(h_2) \star s') \geq T$ **then**
      $e'_{2i} = e'_{2i} \oplus 1$
    end if
  end for
  $s' = s + h_0 e'_0 + h_1 e'_1 + h_2 e'_2$
end for
return e'

threshold$(w, w', it, c_a, c_b, \delta)$

$T_{\text{now}} \leftarrow c_a \cdot w' + c_b$
$T_{\text{init}} \leftarrow c_a \cdot w + c_b$
$M \leftarrow (d+1)/2$
if $it = 1$ **then**
  $T' \leftarrow T_{\text{init}} + \delta$
else if $it = 2$ **then**
  $T' \leftarrow (2T_{\text{init}} + M)/3 + \delta$
else if $it = 3$ **then**
  $T' \leftarrow (T_{\text{init}} + 2M)/3 + \delta$
else
  $T' \leftarrow M + \delta$
end if
return $\max(T_{\text{now}}, T')$

The threshold is defined as

$$T = \max\left(c_a w' + c_b, T'\right),$$

where

$$T' = \begin{cases} c_a w + c_b + \delta, & \text{it} = 1, \\ \dfrac{2(c_a w + c_b) + M}{3} + \delta, & \text{it} = 2, \\ \dfrac{(c_a w + c_b) + 2M}{3} + \delta, & \text{it} = 3, \\ M + \delta, & \text{it} \geq 4, \end{cases}$$

and

$$M = \frac{d+1}{2}.$$

The first term, $c_a w' + c_b$, provides an adaptive threshold based on the current syndrome weight, while T' is computed from the initial syndrome weight and varies with the iteration number. During the first three iterations, T' gradually

transitions from $c_a w + c_b + \delta$ to $M + \delta$. From the fourth iteration onward, it remains constant. The decoder uses the larger of the two values as the threshold for the current iteration.

The coefficients c_a and c_b were initially chosen according to the affine-threshold methodology proposed by [9,23]. For a syndrome of weight S, the theoretical threshold can be expressed as a function $F_t(S)$, where t denotes the error weight. Since direct evaluation of $F_t(S)$ is impractical during decoding, it is approximated by an affine function

$$f_t(S) = aS + b.$$

The coefficients are obtained from a local linear approximation around the average syndrome weight S_{avg},

$$a = F_t(S_{\text{avg}} + 1) - F_t(S_{\text{avg}}),$$

$$b = F_t(S_{\text{avg}}) - aS_{\text{avg}}.$$

This affine approximation provides a reference relationship between syndrome weight and threshold, which serves as the basis for selecting the coefficients c_a and c_b. In practice, the values suggested by the affine-threshold analysis were used only as initial reference points. The final parameters (c_a, c_b, δ) were obtained through empirical tuning to optimize decoding performance and minimize the decoding failure rate. The resulting parameter sets are reported in Table 2. The same parameters are subsequently used for the DFR estimation presented in Section 5.

3.4 Parameters

We propose three parameter sets, TRIKE-1/3/5, targeting classical security levels of 128/192/256 bits and quantum security levels of 64/96/128 bits, which correspond to NIST security levels 1/3/5. The detailed parameters are given in Table 1. In addition, the recommended threshold coefficients for the TRIKE bit-flipping Decoder are listed in Table 2.

Table 1. Suggested TRIKE parameters.

Instance	Classical Security	Quantum Security	r	w	t	l	Target DFR
TRIKE-1	128	64	$9,437$	81	201	128	2^{-128}
TRIKE-3	192	96	$19,181$	123	319	256	2^{-192}
TRIKE-5	256	128	$32,987$	165	429	256	2^{-256}

Table 2. Suggested parameters for TRIKE bit-flipping Decoder.

Instance	$I_{\max}$	c_a	c_b	δ
TRIKE-1	7	0.003234304810124655	6.772697599995158	2
TRIKE-3	7	0.002270835815881611	8.361152569725370	4
TRIKE-5	7	0.001814289691331794	10.24547107880269	5

The choice of (w, r, l) directly determines the storage and communication costs of TRIKE. In our implementations, the private key has size $4 \cdot \lceil w + (3 \cdot \lceil r/8 \rceil + 2 \cdot \lceil l/8 \rceil)/4 \rceil$, while the public key has size $\lceil r/8 \rceil + \lceil l/8 \rceil$. The ciphertext has size $2 \cdot \lceil r/8 \rceil + \lceil l/8 \rceil$. The communication bandwidth is the sum of the public key and the ciphertext, i.e., $3 \cdot \lceil r/8 \rceil + 2 \cdot \lceil l/8 \rceil$. The resulting private key, public key, ciphertext sizes, and bandwidth for different TRIKE instances are given in bytes in Table 3.

Table 3. TRIKE Private Key, Public Key and Ciphertext sizes (in bytes).

Instance	TRIKE-1	TRIKE-3	TRIKE-5
Private Key	3,896	7,752	13,096
Public Key	1,196	2,430	4,156
Ciphertext	2,376	4,828	8,280
Bandwidth	3,572	7,258	12,436

More generally, TRIKE private keys can be represented either as sparse binary vectors, as lists of nonzero-bit positions, or by a seed from which the key is deterministically generated, providing different space-time trade-offs. To maximize decapsulation performance, our implementations retain all three representations, resulting in a private key size of $4 \cdot \lceil w + (3 \cdot \lceil r/8 \rceil + 2 \cdot \lceil l/8 \rceil)/4 \rceil$.

Then we focus on public-key size, ciphertext size, and communication bandwidth when comparing TRIKE with other code-based KEMs. To provide a broader perspective on communication efficiency, we further compare TRIKE with representative NIST code-based schemes, including Classic McEliece [3], BIKE [5], and HQC [2], at NIST Security Levels 1, 3 and 5. The comparison focuses on public key size, ciphertext size, and communication bandwidth, as summarized in Table 4.

TRIKE achieves the smallest public-key size across all security levels, reducing the public-key size by 22%, 21%, and 19% compared with BIKE at NIST Security Levels 1, 3, and 5, respectively. This reduction comes at the cost of larger ciphertexts, which are approximately 51%, 55%, and 61% larger than those of BIKE. As a result, the total bandwidth of TRIKE is only moderately higher,

by approximately 15%, 17%, and 21% at the three security levels, respectively. Compared to the standardized HQC, TRIKE significantly reduces bandwidth requirements, consuming only 54%, 54% and 57% of HQC's bandwidth at the NIST-1, NIST-3 and NIST-5 security levels, respectively.

Table 4. Comparison with NIST Code-Based Standards (in bytes).

Security Level	Scheme	Public Key	Ciphertext	Bandwidth
Level 1	McEliece348864	261,120	96	261,216
	BIKE-1	1,541	1,573	3,114
	HQC-1	2,241	4,433	6,674
	TRIKE-1	1,196	2,376	3,572
Level 3	McEliece460896	524,160	156	524,316
	BIKE-3	3,083	3,115	6,198
	HQC-3	4,514	8,978	13,492
	TRIKE-3	2,430	4,828	7,258
Level 5	McEliece6960119	1,047,319	194	1,047,513
	BIKE-5	5,122	5,154	10,276
	HQC-5	7,237	14,421	21,658
	TRIKE-5	4,156	8,280	12,436

4 The Security of TRIKE

4.1 Security Assumptions and Hardness Reduction

In this subsection, we present the formal definitions of the underlying security assumptions. We then show that the hardness of these assumptions can be reduced to well-established problems, such as 3-QCCF and (2,3)-QCSD.

Specifically, we first introduce the Ratio Quasi-Cyclic Codeword Finding (RQCCF) problem and prove that its hardness reduces to that of the standard 3-QCCF problem[3].

Definition 5 (3-QCCF Distribution). *The 3-QCCF distribution samples vectors* $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$, $x \xleftarrow{\$} \mathcal{R}_{even}$, *computes* $y \leftarrow \frac{h_0 + x h_1}{h_2}$ *and outputs* (x, y).

[3] Throughout this section, we omit the negligible statistical term $O(2^{-r})$ resulting from the unique non-invertible element of $\mathcal{R}_{odd}$.

Definition 6 (3-QCCF Problem). *The 3-QCCF problem asks to decide with non-negligible advantage whether (x, y) came from 3-QCCF distribution or the uniform distribution $\mathcal{R}_{even} \times \mathcal{R}_{odd}$.*

Definition 7 (RQCCF Distribution). *The RQCCF distribution samples vectors $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$, $t_1, t_2 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even}$, computes $x \leftarrow \frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$ and outputs (t_1, t_2, x).*

Definition 8 (RQCCF Problem). *The RQCCF problem asks to decide with non-negligible advantage whether (t_1, t_2, x) came from RQCCF distribution or the uniform distribution $\mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$.*

Lemma 1 (Bijection Preserves Uniform Independence). *Let $S_1, \ldots, S_m$ and $T_1, \ldots, T_m$ be finite sets, and let*

$$f : S_1 \times \cdots \times S_m \to T_1 \times \cdots \times T_m$$

be a bijection. If $(x_1, \ldots, x_m)$ is uniformly distributed over $S_1 \times \cdots \times S_m$, then

$$(y_1, \ldots, y_m) = f(x_1, \ldots, x_m)$$

is uniformly distributed over $T_1 \times \cdots \times T_m$. In particular, $y_1, \ldots, y_m$ are mutually independent and uniformly distributed over $T_1, \ldots, T_m$, respectively.

Proof. Let $N = \prod_{i=1}^{m} |S_i| = \prod_{i=1}^{m} |T_i|$, where the equality follows from the bijectivity of f.

Since $(x_1, \ldots, x_m)$ is uniformly distributed over $S_1 \times \cdots \times S_m$, for any $(s_1, \ldots, s_m)$,

$$\Pr[(x_1, \ldots, x_m) = (s_1, \ldots, s_m)] = \frac{1}{N}.$$

Let $(y_1, \ldots, y_m) = f(x_1, \ldots, x_m)$. For any $(t_1, \ldots, t_m)$, by bijectivity there exists a unique $(s_1, \ldots, s_m)$ such that

$$f(s_1, \ldots, s_m) = (t_1, \ldots, t_m).$$

Hence

$$\Pr[(y_1, \ldots, y_m) = (t_1, \ldots, t_m)] = \frac{1}{N},$$

which shows that $(y_1, \ldots, y_m)$ is uniformly distributed over $T_1 \times \cdots \times T_m$.

For any $i \in \{1, \ldots, m\}$ and $t_i \in T_i$, we have

$$\Pr[y_i = t_i] = \sum_{\substack{t_j \in T_j \\ j \neq i}} \Pr[(y_1, \ldots, y_m) = (t_1, \ldots, t_m)] = \frac{\prod_{j \neq i} |T_j|}{N} = \frac{1}{|T_i|}.$$

Finally, for any $(t_1, \ldots, t_m)$,

$$\Pr[y_1 = t_1, \ldots, y_m = t_m] = \frac{1}{N} = \prod_{i=1}^{m} \frac{1}{|T_i|} = \prod_{i=1}^{m} \Pr[y_i = t_i],$$

which proves mutual independence. □

Theorem 1. *If there exists a PPT adversary that solves the RQCCF problem with advantage ϵ, then there exists a PPT adversary that solves the 3-QCCF problem with the same advantage ϵ.*

Proof. **Adversary Construction.** Let $\mathcal{A}$ be an adversary for the RQCCF problem. We construct an adversary $\mathcal{B}$ for the 3-QCCF problem as follows.

- Given a challenge (x, y), $\mathcal{B}$ samples $t_1 \xleftarrow{\$} \mathcal{R}_{even}$ and sets $t_2 \leftarrow x + t_1 y$.
- $\mathcal{B}$ runs $\mathcal{A}$ with input (t_1, t_2, y) and receives the output of $\mathcal{A}$.
- If $\mathcal{A}$ answers the RQCCF distribution, then $\mathcal{B}$ outputs the 3-QCCF distribution. Otherwise, $\mathcal{B}$ outputs the uniform distribution.

Distribution matching. We then analyze the two cases.

(1) (x, y) is sampled from the 3-QCCF distribution. Then there exist $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$ such that $y = \frac{h_0 + x h_1}{h_2}$. We first show that t_2 is uniformly distributed over $\mathcal{R}_{even}$ and independent of t_1.

Substituting y, we rewrite t_2 as

$$t_2 = x + t_1 y = x + t_1 \cdot \frac{h_0 + x h_1}{h_2} = x \left(1 + \frac{t_1 h_1}{h_2}\right) + \frac{t_1 h_0}{h_2}.$$

Let

$$A = 1 + \frac{t_1 h_1}{h_2}, \quad B = \frac{t_1 h_0}{h_2}.$$

With $t_1 \in \mathcal{R}_{even}$ and $1 \in \mathcal{R}_{odd}$, we have $A \in \mathcal{R}_{odd}$. Thus the map

$$(t_1, x) \mapsto (t_1, t_2 = Ax + B)$$

is a bijection. With the bijection from (t_1, x) to (t_1, t_2) and Lemma 1, (t_1, t_2) is uniform random on $\mathcal{R}_{even} \times \mathcal{R}_{even}$.

Next, we show that (t_1, t_2, y) follows the RQCCF distribution. By definition, it suffices to show that there exist $\bar{h}_0, \bar{h}_1, \bar{h}_2 \xleftarrow{\$} \mathrm{K}(w)$ such that

$$y = \frac{\bar{h}_2 + t_2 \bar{h}_0}{\bar{h}_1 + t_1 \bar{h}_0}.$$

Define

$$\bar{h}_0 := h_1, \quad \bar{h}_1 := h_2, \quad \bar{h}_2 := h_0.$$

Then the above equality becomes

$$y = \frac{h_0 + t_2 h_1}{h_2 + t_1 h_1}.$$

To prove this equality, it suffices to verify that

$$y(h_2 + t_1 h_1) = h_0 + t_2 h_1.$$

Substituting $t_2 = x + t_1 y$, the right-hand side becomes

$$h_0 + (x + t_1 y)h_1 = h_0 + xh_1 + t_1 y h_1.$$

Thus, it suffices to show that

$$yh_2 + t_1 y h_1 = h_0 + xh_1 + t_1 y h_1.$$

Cancelling the common term $t_1 y h_1$ from both sides, we reduce to

$$yh_2 = h_0 + xh_1,$$

which holds by the definition of y in the 3-QCCF distribution.

Therefore,

$$y = \frac{\bar{h}_2 + t_2 \bar{h}_0}{\bar{h}_1 + t_1 \bar{h}_0},$$

and hence (t_1, t_2, y) is distributed exactly as an RQCCF sample.

(2) (x, y) is uniformly sampled from $\mathcal{R}_{even} \times \mathcal{R}_{odd}$. With $t_2 = x + t_1 y$, we can construct a natural bijection from (t_1, x, y) to (t_1, t_2, y). With Lemma 1, (t_1, t_2, y) is uniformly distributed over $\mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$.

Conclusion. In both cases, the distribution given to $\mathcal{A}$ is identical to its respective distribution of RQCCF problem. Therefore, $\mathcal{B}$ perfectly simulates the RQCCF problem, and its advantage is equal to that of $\mathcal{A}$, namely ϵ. □

Then we introduce the definition of 3-TRIKE problem, which corresponds to the security against key recovery attack of our scheme. Furthermore, we prove that the hardness of 3-TRIKE can be reduced to that of RQCCF.

Definition 9 (3-TRIKE Distribution). *The 3-TRIKE distribution samples vectors* $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$, $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$, *computes* $r_2 \leftarrow t_2 + (r_1 + t_1)\frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$ *and outputs* (t_1, t_2, r_1, r_2).

Definition 10 (3-TRIKE Problem). *The 3-TRIKE problem asks to decide with non-negligible advantage whether* (t_1, t_2, r_1, r_2) *came from 3-TRIKE distribution or the uniform distribution* $\mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{odd}$.

Theorem 2. *If there exists a PPT adversary that solves the 3-TRIKE problem with advantage* ϵ, *then there exists a PPT adversary that solves the RQCCF problem with advantage* ϵ.

Proof. **Adversary Construction.** Let $\mathcal{A}$ be an adversary for 3-TRIKE problem. Then we construct an adversary $\mathcal{B}$ for the RQCCF problem as follows.

- Given an instance (t_1, t_2, x) of the RQCCF problem, $\mathcal{B}$ samples $r_1 \xleftarrow{\$} \mathcal{R}_{odd}$ and computes $r_2 := t_2 + (r_1 + t_1)x$.
- $\mathcal{B}$ runs $\mathcal{A}$ with input (t_1, t_2, r_1, r_2) and receives the output of $\mathcal{A}$.

– If $\mathcal{A}$ outputs that the input is from the 3-TRIKE distribution, $\mathcal{B}$ outputs the RQCCF distribution. Otherwise, $\mathcal{B}$ outputs the uniform distribution.

Distribution matching. We also analyze two cases.

(1) $x = \frac{h_2+t_2h_0}{h_1+t_1h_0}$ as in the RQCCF distribution. Then (t_1, t_2, r_1, r_2) is distributed exactly as 3-TRIKE with the difinition of 3-TRIKE.

(2) x is uniform in $\mathcal{R}_{odd}$. Then we can construct a bijection from (t_1, t_2, r_1, x) to (t_1, t_2, r_1, r_2). With Lemma 1, (t_1, t_2, r_1, r_2) is uniform random distribution on $\mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{odd}$.

Conclusion. Therefore, the advantage of $\mathcal{B}$ in distinguishing RQCCF from uniform is exactly the same as the advantage of $\mathcal{A}$ in distinguishing TRIKE distribution from uniform.

□

Next, we introduce the (2,3)-TRIKE problem, which characterizes the security against message recovery attacks, and prove that its hardness reduces to that of (2,3)-QCSD.

Definition 11 ((2,3)-QCSD Distribution). *The (2,3)-QCSD distribution samples vectors* $e_0, e_1, e_2 \xleftarrow{\$} \mathrm{E}(t)$, $x, y \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{odd}$, *computes* $u \leftarrow e_0 + xe_2$, $v \leftarrow e_1 + ye_2$ *and outputs* (x, y, u, v).

For any integer t, its parity is denoted $p(t) \in \{odd, even\}$. Let $\mathcal{R}_{2,p(t)} = (\mathcal{R}_{even} \times \mathcal{R}_{p(t)}) \cup (\mathcal{R}_{odd} \times \mathcal{R}_{p(t+1)})$.

Definition 12 ((2,3)-QCSD Problem). *The (2,3)-QCSD problem asks to decide with non-negligible advantage whether* (x, y, u, v) *came from (2,3)-QCSD distribution or the uniform distribution* $\mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{2,p(t)}$.

Definition 13 ((2,3)-TRIKE Distribution). *The (2,3)-TRIKE distribution samples vectors* $e_0, e_1, e_2 \xleftarrow{\$} \mathrm{E}(t)$, $t_1, t_2, r_1, r_2 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{odd}$, *computes* $u \leftarrow e_0 + r_1e_1 + r_2e_2$, $v \leftarrow e_0 + t_1e_1 + t_2e_2$ *and outputs* $(t_1, t_2, r_1, r_2, u, v)$.

Definition 14 ((2,3)-TRIKE Problem). *The (2,3)-TRIKE problem asks to decide with non-negligible advantage whether* $(t_1, t_2, r_1, r_2, u, v)$ *came from (2,3)-TRIKE distribution or the uniform distribution* $\mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{odd} \times \mathcal{R}_{p(t)} \times \mathcal{R}$.

Theorem 3. *If there exists a PPT adversary that solves the (2,3)-TRIKE problem with advantage* ϵ, *then there exists a PPT adversary that solves the (2,3)-QCSD problem with advantage* ϵ.

Proof. **Adversary Construction.** Let $\mathcal{A}$ be an adversary for solving (2,3)-TRIKE problem. Then we construct an adversary $\mathcal{B}$ for the (2,3)-QCSD problem as follows.

– Given an instance (x, y, u, v) of (2,3)-QCSD problem, $\mathcal{B}$ samples $t_1, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{odd}$.

- $\mathcal{B}$ set $(t_2, r_2, u', v') = (t_1 y + x, r_1 y + x, r_1 v + u, t_1 v + u)$.
- $\mathcal{B}$ runs $\mathcal{A}$ with input $(t_1, t_2, r_1, r_2, u', v')$.
- If $\mathcal{A}$ outputs that the input is from (2,3)-TRIKE distribution, $\mathcal{B}$ outputs the (2,3)-QCSD distribution. Otherwise, $\mathcal{B}$ outputs the uniform distribution.

Distribution Matching. We first prove that the distribution (t_1, r_1, t_2, r_2) is identical to the uniform distribution over $\mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$. Since $t_1 + r_1 \in \mathcal{R}_{odd}$, the equations $t_2 = t_1 y + x, r_2 = r_1 y + x$ have a unique solution for (x, y). Thus the map from (t_1, r_1, x, y) to (t_1, r_1, t_2, r_2) is a bijection. With Lemma 1, (t_1, r_1, t_2, r_2) is a uniform distribution over $\mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$.

Then we analyze two cases.

(1) (x, y, u, v) follows (2,3)-QCSD distribution. So $\exists e_0, e_1, e_2 \xleftarrow{\$} \mathrm{E}(t)$ s.t. $u = e_0 + x e_2$ and $v = e_1 + y e_2$. With $u' = r_1 v + u$ and $v' = t_1 v + u$, we can verify that $u' = r_1 v + u = r_1 e_1 + r_1 y e_2 + e_0 + x e_2 = e_0 + r_1 e_1 + r_2 e_2$ and $v' = t_1 v + u = t_1 e_1 + t_1 y e_2 + e_0 + x e_2 = e_0 + t_1 e_1 + t_2 e_2$, which is exactly the definition of (2,3)-TRIKE distribution.

(2) (x, y, u, v) follows uniform distribution. Since $t_1 + r_1 \in \mathcal{R}_{odd}$, the equations $u' = r_1 v + u, v' = t_1 v + u$ have a unique solution for (u, v). Thus the map from $(t_1, r_1, t_2, r_2, u, v)$ to $(t_1, r_1, t_2, r_2, u', v')$ is a bijection. With Lemma 1, $(t_1, t_2, r_1, r_2, u', v')$ is a uniform distribution over $\mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd} \times \mathcal{R}_{odd} \times \mathcal{R}_{p(t)} \times \mathcal{R}$.

Conclusion. As a result, $\mathcal{B}$ succeeds if and only if the $\mathcal{A}$ solves the (2,3)-TRIKE problem, which completes the proof. □

4.2 OW-CPA Security

We first analyze PKE_0, which differs from the PKE in Section 3.1 only by sampling (t_1, t_2, r_1) uniformly and storing them explicitly rather than deriving them from σ. Its plaintext space is $\mathrm{E}(t)$, and OW-CPA success means recovering any error vector consistent with the challenge ciphertext. Let

$$\varepsilon_{\mathsf{rnd}}(r, t) := \min\left\{1, \frac{\binom{3r}{t}}{2^{2r-1}}\right\},$$

which is negligible for $t = O(\sqrt{r})$. Appendix A.1 gives the full construction and establishes the following bound.

Theorem 4 (OW-CPA Security). *For any PPT OW-CPA adversary $\mathcal{A}'$ against the PKE_0 scheme, there exist PPT distinguishers $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}') \le \mathsf{Adv}_{3\text{-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{(2,3)\text{-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r, t).$$

4.3 IND-CPA Security

To encrypt messages in M, PKE_1 keeps the key generation of PKE_0, samples $e \xleftarrow{\$} \mathrm{E}(t)$, and outputs $c = (u, v, m + \mathtt{L}(e))$, where $(u, v) \leftarrow \mathrm{Encrypt}_0(\mathsf{pk}, e)$. Decryption recovers e and unmasks m, returning $\perp$ if error recovery fails. With $\mathtt{L}$ modeled as a random oracle, the reduction in Appendix A.2 yields the following consequence of Theorem 4.

Theorem 5 (IND-CPA Security). *For any PPT* IND-CPA *adversary* $\mathcal{A}$ *against the* PKE_1 *scheme, there exist PPT adversaries* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) \leq \frac{1}{2}\left(\mathsf{Adv}_{3\text{-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{(2,3)\text{-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right). \tag{2}$$

4.4 IND-CCA Security

Let δ bound the average DFR of PKE_1. Since its decoding failure probability is independent of the message, δ also bounds its correctness error. Following the HHK framework, we obtain deterministic PKE_2 by deriving (t_1, t_2, r_1) from a uniform seed $\sigma \in \mathrm{M}$ using $\mathtt{H}_1$, $\mathtt{H}_2$, and $\mathtt{H}_3$, setting $e = \mathtt{H}_4(m, r_2)$, and adding ciphertext consistency checks to decryption. Encapsulation then samples $m \xleftarrow{\$} \mathrm{M}$ and outputs $c \leftarrow \mathrm{Encrypt}_2(\mathsf{pk}, m)$ together with $K = \mathtt{K}(m, c)$; decapsulation uses implicit rejection. With these hash functions modeled as random oracles, applying the reductions of [15, Theorems 2 and 4], as detailed in Appendix A.3, gives the following bound.

Theorem 6 (IND-CCA Security). *For any PPT* IND-CCA *adversary* $\mathcal{B}$ *against the* KEM *making at most* q *queries to* $\mathtt{K}$ *or* $\mathtt{H}_4$*, there exist PPT adversaries* $\mathcal{A}$*,* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{KEM}}(\mathcal{B}) &\leq q\cdot\delta + \frac{3q+1}{|\mathrm{M}|} + 3\cdot\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) \\ &\leq q\cdot\delta + \frac{3q+1}{|\mathrm{M}|} + \frac{3}{2}\left(\mathsf{Adv}_{3\text{-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{(2,3)\text{-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).\end{aligned}$$

4.5 Security Proof with Weak Key Rejection

Finally, let $\mathrm{KEM}_\$$ denote the scheme with weak key rejection, ρ its rejection rate, and $\delta_\$$ the average DFR over accepted keys. The argument in Appendix A.4 conditions the OW-CPA experiment on acceptance, introducing a factor of $1/(1-\rho)$, and then applies the same IND-CPA and HHK reductions to obtain the following bound.

Theorem 7. *For any PPT* IND-CCA *adversary* $\mathcal{B}_\$$ *against* $\mathrm{KEM}_\$$ *making at most* q *queries to* $\mathtt{K}$ *or* $\mathtt{H}_4$*, there exist PPT adversaries* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{KEM}_\$}(\mathcal{B}_\$) \leq\ & q\cdot\delta_\$ + \frac{3q+1}{|\mathrm{M}|} \\ &+ \frac{3}{2(1-\rho)}\left(\mathsf{Adv}_{3\text{-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{(2,3)\text{-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).\end{aligned}$$

Thus, rejection costs at most $-\log_2(1-\rho)$ bits in the hardness-based bound, while $\delta_{\$} \leq \delta/(1-\rho)$. Concrete rejection rates and weak-key-induced failures are analyzed in Section 6.1.

4.6 Known Attacks

For TRIKE, both key recovery and message recovery attacks can be reduced to decoding problems over quasi-cyclic codes. In particular, key recovery attacks correspond to the 3-TRIKE setting, while message recovery attacks correspond to the (2,3)-TRIKE setting. These problems can be further reduced to the 3-QCCF problem and the (2,3)-QCSD problem, respectively, as shown in Section 4.1.

The best known attacks against these problems are based on Information Set Decoding (ISD) algorithms. In our analysis, we estimate the attack complexities using the framework of [12], considering several representative ISD variants, including Prange's algorithm [20], Stern's algorithm [7,19,22], BJMM [6,13,17], Both-May [8] and May-Ozerov [18].

Due to the quasi-cyclic structure of QC-MDPC codes, the overall attack complexities are further reduced by a factor of r for key recovery attacks and by a factor of $\sqrt{r}$ for message recovery attacks. The resulting time complexities at the targeted security levels are summarized in Table 5 and Table 6.

Table 5. Key recovery time complexity (in $\log_2$ scale).

Instance	Prange	Stern	BJMM	BothMay	MayOzerov	Complexity
TRIKE-1	168.33	**147.03**	147.40	147.22	147.22	**133.82**
TRIKE-3	237.87	**214.24**	214.55	214.34	214.34	**200.01**
TRIKE-5	306.70	281.45	**281.33**	281.38	281.38	**266.32**

Table 6. Message recovery time complexity (in $\log_2$ scale).

Instance	Prange	Stern	BJMM	BothMay	MayOzerov	Complexity
TRIKE-1	158.57	138.45	138.80	138.35	**138.34**	**131.74**
TRIKE-3	230.66	207.91	207.51	207.15	**207.15**	**200.03**
TRIKE-5	297.29	272.86	272.04	271.95	**271.95**	**264.45**

5 Analysis of Decryption Failure Rate

5.1 Theoretical DFR Model

The DFR analysis of the iterative decoder is based on the finite-length scaling (FLS) behavior of iteratively decoded LDPC codes [4]. In the waterfall region, the decoding failure probability around the asymptotic threshold is characterized, to first order, by a Gaussian scaling law with a transition width of order $1/\sqrt{n}$.

In our setting, the error vector is sampled uniformly from all vectors of weight t, which can be viewed as transmission over a uniform fixed-weight additive noise channel. We use

$$\sigma = \frac{t}{n}$$

as its normalized channel parameter. Although the errors are not independent due to the fixed-weight constraint, their distribution on any bounded local neighborhood converges to that of independent Bernoulli errors with parameter σ as n increases. Moreover, the QC-MDPC Tanner graph is regular, and its degree distributions are monomials determined only by the row weight d. Therefore, for fixed d and a fixed decoder configuration, varying the circulant size r (and hence n) preserves the asymptotic local decoding dynamics, while the dominant variation is captured by the finite-length scaling.

Accordingly, we model the DFR in the waterfall region as

$$\mathrm{DFR} \approx Q\left(\frac{\sqrt{n}\,(\sigma^* - \sigma)}{\alpha}\right),$$

where Q is the Gaussian Q-function, σ^* denotes the asymptotic decoding threshold, and α is the scaling parameter that characterizes the width of the waterfall region. This form is equivalent, up to a rescaling of α, to the canonical FLS expression in terms of channel capacity, since any smooth channel parameterization is locally equivalent around the decoding threshold.

For the practical decoder considered here, analytically determining σ^* and α is difficult due to the fixed number of iterations and the specific bit-flipping update rules. We therefore estimate these parameters from empirical DFR measurements at smaller code lengths and use the resulting FLS model for extrapolation.

For a fixed error weight t, substituting $\sigma = t/n$ into the FLS model and applying the inverse Q-function gives

$$\frac{Q^{-1}(\mathrm{DFR})}{\sqrt{n}} = -\frac{t}{\alpha}\frac{1}{n} + \frac{\sigma^*}{\alpha}.$$

Setting $x = 1/n$ and $y = Q^{-1}(\mathrm{DFR})/\sqrt{n}$ yields $y = ax + b$, where $a = -t/\alpha$ and $b = \sigma^*/\alpha$. We estimate a and b by ordinary least squares on the transformed measurements and use the fitted relation to extrapolate the DFR to the target code length.

5.2 Validation of the FLS-Based DFR Model

To validate the suitability of the proposed FLS-based linear model, we compare its fitting performance with the conventional extrapolation method commonly used in QC-MDPC-based cryptosystems from [23]. In that approach, the curve in the $(r, \log_2(\mathrm{DFR}))$ domain is typically concave over the parameter range of interest, and a conservative upper bound is obtained by extrapolating along the secant determined by the last two available data points.

Using a representative dataset measured for the TRIKE-1 instance, the FLS linear model exhibits a significantly better fitting accuracy. In particular, the coefficient of determination reaches $R^2 = 0.9998$, compared to $R^2 = 0.9843$ for the direct linear fit in the $(r, \log_2(\mathrm{DFR}))$ domain.

This difference is consistent with the observed geometric behavior of the two models. The function $r \mapsto \log_2(\mathrm{DFR})$ exhibits a concave profile with decreasing slope, so a global linear approximation is inherently inaccurate. For this reason, previous estimates typically rely only on the secant determined by the last two points to obtain a conservative upper bound, as illustrated in Figure 1. However, such a construction does not exploit all available data and generally produces a looser estimate.

By contrast, after the FLS transformation, the relation between

$$y = \frac{Q^{-1}(\mathrm{DFR})}{\sqrt{n}} \quad \text{and} \quad x = \frac{1}{n}$$

is observed to be highly linear, as shown in Figure 2, making ordinary least-squares fitting directly applicable. The resulting regression captures all measured points with substantially smaller residual error and provides a tighter extrapolation toward the target parameter.

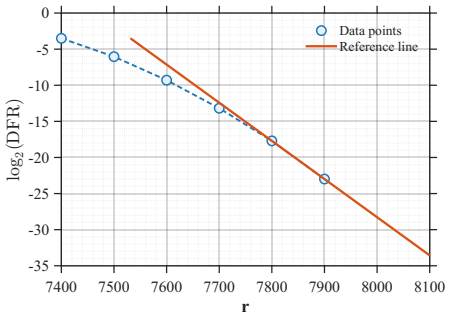


Fig. 1. Conventional linear extrapolation.

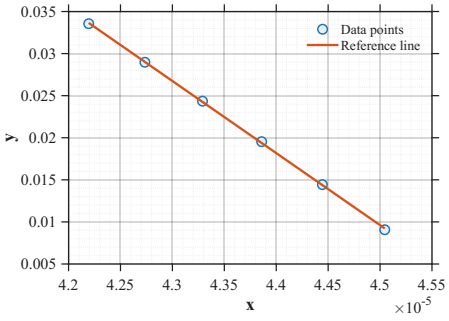


Fig. 2. FLS-based least-squares fitting.

This comparison suggests that the FLS-based method yields a more accurate upper bound on the DFR than conventional concavity-based extrapolation, while retaining a principled analytical justification. We therefore adopt the FLS model in the concrete DFR estimation.

5.3 Concrete DFR Estimate

To estimate the DFR at our target parameter r_{test}, we perform independent decoding experiments at smaller circulant sizes r, with bit-flipping decoding fixed to 7 iterations. We construct Clopper–Pearson confidence intervals for the measured DFRs and propagate them through the FLS transformation and pairwise linear extrapolation. The interval construction and selection procedure are detailed in Appendix D. The resulting estimates, based on 95% input confidence intervals, are presented in Table 7.

Table 7. DFR estimates using a 95% input confidence intervals.

	TRIKE-1[4]	TRIKE-3	TRIKE-5
r	$9,437$	$19,181$	$32,987$
$\log_2(\text{DFR})$	-174.40	-193.41	-271.58

6 Analysis of Weak Keys

The FLS-based DFR model provides an accurate estimate of the decoding failure rate of the TRIKE bit-flipping decoder for the selected parameter sets. However, it characterizes the typical behaviour of the decoder and may not fully capture failures caused by specific structural properties of the key or error patterns. In particular, two types of phenomena may affect the DFR: weak keys, namely decoding failures induced by structured keys, and error floors, namely decoding failures induced by structured error patterns. While our simulations show good agreement with the FLS-based predictions, it remains difficult to precisely quantify the contribution of the error floor in the very low-DFR regime. In fact, in coding theory, accurately characterizing the low-DFR regime has long been a challenging open problem. As noted in the BIKE specification [5], it remains unclear whether error patterns close to identified near-codewords dominate the DFR in the error-floor region. Accordingly, in this section we consider weak keys and show how they can be efficiently detected and rejected.

6.1 Weak-Key Rejection

To address the weak-key classes identified in [23,24], we employ a hybrid rejection test, combining an intra-block 4-cycle statistic inspired by [16] with a direct

[4] For TRIKE-1, smaller primitive primes offer no significant size reduction. We therefore choose $r = 9{,}437$ to provide a more conservative DFR margin.

bound on cyclic inter-block intersections. For each circulant block h_i, let $\mu(\delta, h_i)$ denote the multiplicity of distance δ in its distance spectrum, and define

$$s_0(h_i) = \sum_{\delta=1}^{(r-1)/2} \binom{\mu(\delta, h_i)}{2},$$

where $rs_0(h_i)$ is the number of intra-block 4-cycles.

For two distinct blocks h_i and h_j, define their maximum cyclic intersection as

$$\iota(h_i, h_j) = \max_{0 \le \ell < r} w_H(h_i \star x^\ell h_j).$$

A candidate key is rejected if $s_0(h_i) > s$ for any block or $\iota(h_i, h_j) \ge \kappa$ for any pair of distinct blocks. Algorithm 2 summarizes the test, and rejected candidates are resampled during key generation.

Algorithm 2: Weak-Key Rejection Test

Input: Candidate key (h_0, h_1, h_2), thresholds s, κ
Output: Rejection flag b (1 for rejection)

$b \leftarrow 0$
for $i = 0, 1, 2$ **do**
    $b \leftarrow b \vee [s_0(h_i) > s]$
end for
for $0 \le i < j \le 2$ **do**
    $b \leftarrow b \vee [\iota(h_i, h_j) \ge \kappa]$
end for
return b

The thresholds used for the selected parameter sets are given in Table 8. The resulting rejection rate is below 1% for all instances, corresponding to a security loss below 0.015 bits according to the analysis in Appendix A.4.

Table 8. Reference thresholds for weak-key rejection.

	TRIKE-1	TRIKE-3	TRIKE-5
s	28	59	105
κ	5	5	5
Rejection Rate	0.79%	0.82%	0.53%

6.2 Inter-Block Weak Keys

The inter-block test directly targets Type III weak keys of [23]. We define a Type III weak key with parameter m as a key $h = (h_0, h_1, h_2)$ such that

$$w_H(h_i \star x^\delta h_j) = m$$

for some distinct $i, j \in \{0, 1, 2\}$ and $\delta \in \{0, \ldots, r-1\}$. The density of such weak keys is bounded by

$$\tau_{III}(m) \leq \frac{3r\binom{d}{m}\binom{r-d}{d-m}}{\binom{r}{d}}.$$

Since $\iota(h_i, h_j) \geq m$, every such key with $m \geq \kappa$ is rejected deterministically, giving a conditional miss rate of $p_{\text{miss}} = 0$.

We set $\kappa = 5$ for all selected parameter sets. Table 9 summarizes the corresponding parameters and density bounds at the rejection threshold $m = \kappa$. These results show that the inter-block test deterministically rejects the targeted Type III weak-key classes even when their density bounds are relatively large, demonstrating the effectiveness of the rejection procedure. Moreover, no decoding failures were observed in 10^7 trials for the cases considered.

Table 9. Parameters and densities of Type III weak keys.

Instance	m	$\log_2$(Density)
TRIKE-1	5	−11.80
TRIKE-3	5	−9.48
TRIKE-5	5	−8.20

6.3 Intra-Block Weak Keys

The Type I and Type II structures of [23] induce atypically large multiplicities in the distance spectrum and are suppressed by the intra-block statistic. In our evaluation, their contributions to the average DFR are smaller than those of weak keys satisfying the gathering property [24], so we do not consider them separately.

We call a key $h = (h_0, h_1, h_2)$ a weak key satisfying the (r_s, ϵ)-gathering property if

$$h_i = \phi_\delta(x^l h_i' + h_i'')$$

for some $i \in \{0, 1, 2\}$, $\delta \in \{1, \ldots, (r-1)/2\}$, and $l \in \{0, \ldots, r-1\}$, where $\deg(h_i') < r_s$, $w_H(h_i') = d - \epsilon$, $w_H(h_i'') = \epsilon$, and $w_H(h_i) = d$. The density of such weak keys is bounded by

$$\tau_{gather} \leq \frac{3r(r-1)\sum_{i=d-\epsilon}^{r_s}\binom{i-2}{d-2-\epsilon}\binom{r-i}{\epsilon}}{2\binom{r}{d}}.$$

Our experiments consider only $\epsilon = 0$, since it has a greater impact on decoding. We first identify gathering weak-key classes whose conditional DFRs are experimentally observable, on the order of 10^{-6} to 10^{-5}. For each prescribed gathering property, we repeatedly sample secret keys satisfying this property and execute the TRIKE bit-flipping decoder on independently sampled ciphertext errors. The observed failure frequency estimates the conditional DFR of the corresponding class before rejection.

Unlike Type III weak keys above, the gathering property does not deterministically force the intra-block statistic above the rejection threshold. We therefore evaluate the probability that these weak keys survive the rejection test. To avoid any assumption of independence between decoding failure and acceptance, we conservatively allow every surviving weak key to cause a decoding failure.

Specifically, let $\mathcal{G}$ denote a gathering weak-key class and Acc the event that a key passes the full rejection test. Define the conditional miss rate as

$$p_{\text{miss}} = \Pr[\mathsf{Acc} \mid \mathcal{G}].$$

We bound the residual contribution using the density and conditional miss rate, conservatively allowing every surviving weak key to cause a decoding failure.

Let $\widehat{p}_{\text{miss}}$ denote the modeled probability that the gathering block passes its intra-block test under the distance-spectrum approximation of [21]. Since passing the full rejection test requires this block to pass, $\widehat{p}_{\text{miss}}$ estimates an upper bound on p_{miss} under the model. The detailed model and an independent empirical consistency check are given in Appendix C.

For the post-rejection evaluation, we define

$$\text{Sum} = \log_2(\text{Density}) + \log_2(\widehat{p}_{\text{miss}}),$$

where Density denotes the density bound above. The product 2^{Sum} bounds the surviving weak-key density under the model.[5] This bound does not use the measured conditional DFR.

Table 10 summarizes the selected gathering weak-key classes, their density bounds and measured conditional DFRs, and the model-based evaluation of the rejection test.

For TRIKE-1, TRIKE-3, and TRIKE-5, the largest values of Sum among the evaluated classes are -198.70, -652.85, and -1728.84, respectively. The corresponding model-based contribution bounds remain far below 2^{-128}, 2^{-192}, and 2^{-256}, respectively, even if every surviving weak key causes a decoding failure.

7 The Performance of TRIKE

7.1 Experimental Setup

The benchmarking platform. The experiments were carried out on a server node equipped with an Intel®Xeon® Platinum 8362 processor (Ice Lake generation) operating at a base frequency of 2.80 GHz. The platform is configured

[5] Density is measured over the full key space. Normalizing to the accepted-key space gives the contribution bound $2^{\text{Sum}}/(1-\rho)$. Since $\rho < 1\%$, this correction is negligible.

Table 10. Evaluation of filtering effectiveness against weak keys satisfying the gathering property.

Instance	s	r_s	$\log_2(\text{DFR})$	$\log_2(\text{Density})$	$\log_2(\widehat{p}_{\text{miss}})$	Sum
TRIKE-1	28	550	−19.45	−88.96	−109.74	−198.70
		500	−18.40	−92.63	−127.41	−220.04
		450	−16.16	−96.70	−149.47	−246.17
TRIKE-3	59	850	−18.35	−161.03	−491.82	−652.85
		800	−17.40	−164.61	−534.25	−698.86
		750	−16.03	−168.44	−582.74	−751.18
TRIKE-5	105	1050	−20.08	−249.20	−1479.64	−1728.84
		1000	−18.61	−253.11	−1573.50	−1826.61
		950	−17.70	−257.22	−1677.67	−1934.89

with approximately 1.5 TB of RAM and a multi-level cache hierarchy, including 48 KB L1d cache, 32 KB L1i cache, 1,280 KB L2 cache per core, and 49,152 KB shared L3 cache. Hyper-threading and Turbo Boost were disabled during benchmarking to ensure stable cycle measurements.

OS and compilation. The implementations were written in C and built using CMake (version 4.2.3). The code was compiled with gcc (Ubuntu 11.4.0) in 64-bit mode, using the optimization flags "-O3", "-flto", and "-funroll-all-loops". The implementation uses the vector-PCLMULQDQ instruction and AVX512 architecture extensions, and all experiments were conducted on a Linux (Ubuntu 22.04.1 LTS) operating system.

7.2 Reference Implementation

The reference implementation is written in ISO C and supports the three parameter sets TRIKE-1/3/5. It is designed primarily for portability and does not rely on architecture-specific instruction sets. As with the BIKE reference implementation, constant-time execution is not guaranteed.

Polynomial arithmetic over $\mathcal{R}$ is implemented using Karatsuba multiplication together with the fast polynomial inversion method described in [10]. Decapsulation employs the TRIKE bit-flipping decoder described above, with a fixed number of decoding iterations. The performance of the reference implementation is reported in Table 11.

7.3 Optimized Implementation

The optimized implementation targets mainstream 64-bit PC processors and supports TRIKE-1/3/5. It employs AVX512 and Intel vector-PCLMULQDQ

Table 11. Performances of the reference implementation of TRIKE (in kiloCycles).

Instance	TRIKE-1	TRIKE-3	TRIKE-5
KeyGen	87,063	294,056	830,069
Encaps	7,469	23,321	71,757
Decaps	37,947	114,865	292,153

instructions to accelerate the main arithmetic and decoding operations. The pseudorandom generation and hash functions are implemented using AES and SHA2, respectively, consistently with the optimized implementation used for comparison with BIKE.

For arithmetic on $\mathcal{R}$, polynomial multiplication is implemented using recursive Karatsuba multiplication, with the 512-bit base case accelerated by vector-PCLMULQDQ following [11]. Polynomial inversion follows the fast inversion approach of [10]. The TRIKE bit-flipping decoder is also vectorized with AVX512 to accelerate its dominant operations. In particular, the optimized decoding path retains its fixed-iteration structure and constant-time behavior.

Table 12 reports the performance of optimized TRIKE and BIKE at NIST Security Levels 1, 3, and 5 under the same implementation interface.

Table 12. Performance comparison between optimized TRIKE and BIKE under the same implementation interface (in kiloCycles).

	Level 1		Level 3		Level 5	
Instance	TRIKE-1	BIKE-1	TRIKE-3	BIKE-3	TRIKE-5	BIKE-5
KeyGen	2,365	366	6,217	1,049	12,860	2,643
Encaps	205	74	457	164	1,104	325
Decaps	630	1,177	1,792	3,512	4,439	9,106
Encaps + Decaps	835	1,251	2,249	3,676	5,543	9,431

TRIKE requires more ring arithmetic than BIKE in KeyGen and Encaps, resulting in higher execution costs for these two operations. Nevertheless, TRIKE benefits from a significantly smaller parameter d, which directly reduces the complexity of the bit-flipping decoding process and leads to substantial gains in decapsulation performance. At NIST Security Levels 1, 3, and 5, TRIKE's Decaps time is 54%, 51%, and 49% of BIKE's, respectively, while the combined encapsulation and decapsulation time is 67%, 61%, and 59%. This performance trade-off may be beneficial in practical deployments, such as TLS servers where decapsulation accounts for a substantial share of the workload. In such settings, TRIKE's faster decapsulation could improve throughput and reduce server-side latency.

8 Conclusion

We presented TRIKE, a code-based key encapsulation mechanism built on the RQCCF problem, together with its security analysis, DFR modeling, and weak-key rejection procedure. By combining low secret component weights with suitable parameter choices, TRIKE reduces the combined public-key and ciphertext size by more than 42% compared to HQC and the combined encapsulation and decapsulation time by more than 33% compared to BIKE at all three targeted security levels.

Several directions remain for future work. First, further optimization of the decoder may improve both decapsulation efficiency and decoding reliability. Second, the current decoder uses the two ciphertext components only through their linear combination. Exploiting the additional information available in the individual components may help reduce decoding failures. Finally, exploring KEM constructions based on n-QCCF for $n > 3$ may reveal further tradeoffs among security, decoding performance, and communication cost.

References

1. National institute of standards and technology: Post-quantum cryptography project. https://csrc.nist.gov/projects/post-quantum-cryptography (2016)
2. Aguilar Melchor, C., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Persichetti, E., Zémor, G., Bos, J., Dion, A., Lacan, J., Robert, J.M., Véron, P., Barreto, P.L., Ghosh, S., Gueron, S., Gúneysu, T., Misoczki, R., Richter-Brokmann, J., Sendrier, N., Tillich, J.P., Vasseur, V.: Hamming quasi-cyclic (hqc). Tech. rep., National Institute of Standards and Technology (2025), available at https://pqc-hqc.org/doc/hqc_specifications_2025_08_22.pdf
3. Albrecht, M.R., Bernstein, D.J., Chou, T., Cid, C., Gilcher, J., Lange, T., Maram, V., von Maurich, I., Misoczki, R., Niederhagen, R., Paterson, K.G., Persichetti, E., Peters, C., Schwabe, P., Sendrier, N., Szefer, J., Tjhai, C.J., Tomlinson, M., Wang, W.: Classic McEliece. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions
4. Amraoui, A., Montanari, A., Richardson, T., Urbanke, R.: Finite-length scaling for iteratively decoded ldpc ensembles. IEEE Transactions on Information Theory **55**(2), 473–498 (2009). https://doi.org/10.1109/TIT.2008.2009580
5. Aragon, N., Barreto, P., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Ghosh, S., Gueron, S., Gúneysu, T., Aguilar Melchor, C., Misoczki, R., Persichetti, E., Richter-Brockmann, J., Sendrier, N., Tillich, J.P., Vasseur, V., Zémor, G.: Bike: bit flipping key encapsulation. Tech. rep., National Institute of Standards and Technology (2024), available at https://bikesuite.org/files/v5.2/BIKE_Spec.2024.10.10.1.pdf
6. Becker, A., Joux, A., May, A., Meurer, A.: Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 520–536. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_31
7. Bernstein, D.J., Lange, T., Peters, C.: Attacking and defending the McEliece cryptosystem. In: Buchmann, J., Ding, J. (eds.) PQCrypto 2008. pp. 31–46. Springer, Berlin, Heidelberg (Oct 2008). https://doi.org/10.1007/978-3-540-88403-3_3
8. Both, L., May, A.: Decoding linear codes with high error rate and its impact for LPN security. In: Lange, T., Steinwandt, R. (eds.) PQCrypto 2018. pp. 25–46. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-79063-3_2
9. Chaulet, J.: Étude de cryptosystèmes à clé publique basés sur les codes MDPC quasi-cycliques. Ph.D. thesis, Paris 6 (2017)
10. Drucker, N., Gueron, S., Kostic, D.: Fast polynomial inversion for post quantum qc-mdpc cryptography. Information and Computation **281**, 104799 (2021). https://doi.org/https://doi.org/10.1016/j.ic.2021.104799, https://www.sciencedirect.com/science/article/pii/S0890540121001152
11. Drucker, N., Gueron, S., Krasnov, V.: Fast multiplication of binary polynomials with the forthcoming vectorized vpclmulqdq instruction. In: 2018 IEEE 25th Symposium on Computer Arithmetic (ARITH). pp. 115–119 (2018). https://doi.org/10.1109/ARITH.2018.8464777
12. Esser, A., Verbel, J.A., Zweydinger, F., Bellini, E.: {SoK: CryptographicEstimators - a Software Library for Cryptographic Hardness Estimation}. In: AsiaCCS. {ACM} (2024)
13. Esser, A., Zweydinger, F.: New time-memory trade-offs for subset sum: Improving ISD in theory and practice. In: Hazay, C., Stam, M. (eds.) EURO-

CRYPT 2023, Part V. LNCS, vol. 14008, pp. 360–390. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30589-4_13
14. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) Advances in Cryptology — CRYPTO' 99. pp. 537–554. Springer Berlin Heidelberg, Berlin, Heidelberg (1999)
15. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 341–371. Springer, Cham (Nov 2017). https://doi.org/10.1007/978-3-319-70500-2_12
16. Matthews, G.L., McMillon, E.: A combinatorial approach to avoiding weak keys in the BIKE cryptosystem. DCC **93**(8), 3413–3436 (2025). https://doi.org/10.1007/s10623-025-01643-7
17. May, A., Meurer, A., Thomae, E.: Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 107–124. Springer, Berlin, Heidelberg (Dec 2011). https://doi.org/10.1007/978-3-642-25385-0_6
18. May, A., Ozerov, I.: On computing nearest neighbors with applications to decoding of binary linear codes. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 203–228. Springer, Berlin, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_9
19. Peters, C.: Information-set decoding for linear codes over F_q. In: Sendrier, N. (ed.) PQCrypto 2010. pp. 81–94. Springer, Berlin, Heidelberg (May 2010). https://doi.org/10.1007/978-3-642-12929-2_7
20. Prange, E.: The use of information sets in decoding cyclic codes. IRE Transactions on Information Theory **8**(5), 5–9 (1962). https://doi.org/10.1109/TIT.1962.1057777
21. Sendrier, N.: BIKE decoding failure, weak keys & error floors. In: 2024 NIST Post-Quantum Cryptography Seminar. NIST, Gaithersburg, United States (Sep 2024), https://inria.hal.science/hal-04914446
22. Stern, J.: A method for finding codewords of small weight. In: Cohen, G., Wolfmann, J. (eds.) Coding Theory and Applications. pp. 106–113. Springer Berlin Heidelberg, Berlin, Heidelberg (1989)
23. Vasseur, V.: Post-quantum cryptography: a study of the decoding of QC-MDPC codes. Ph.D. thesis, Université de Paris (2021)
24. Wang, T., Wang, A., Wang, X.: Exploring decryption failures of BIKE: New class of weak keys and key recovery attacks. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part III. LNCS, vol. 14083, pp. 70–100. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38548-3_3

Appendices

A Detailed Security Proofs

A.1 OW-CPA Security

We begin with the expanded-key scheme PKE_0 used in Theorem 4. It samples (t_1, t_2, r_1) uniformly and stores them explicitly rather than deriving them from σ. Its algorithms are as follows.

- KeyGen_0:
  - Output: a public key pk and a secret key sk.
  - $(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$.
  - $(t_1, t_2, r_1) \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$.
  - $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$, $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$.
  - $\mathsf{pk} \leftarrow (t_1, t_2, r_1, r_2)$, $\mathsf{sk} \leftarrow (t_1, r_1, h_0, h_1, h_2)$.
  - Return $(\mathsf{pk}, \mathsf{sk})$.
- $\mathrm{Encrypt}_0$:
  - Input: $\mathsf{pk} = (t_1, t_2, r_1, r_2)$ and $e = (e_0, e_1, e_2) \in \mathrm{E}(t)$.
  - Output: $(u, v) \in \mathcal{R}_{p(t)} \times \mathcal{R}$.
  - $u \leftarrow e_0 + e_1 r_1 + e_2 r_2$, $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$.
  - Return (u, v).
- $\mathrm{Decrypt}_0$:
  - Input: $\mathsf{sk} = (t_1, r_1, h_0, h_1, h_2)$ and (u, v).
  - Output: $e' \in \mathrm{E}(t) \cup \{\perp\}$.
  - $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$.
  - $s \leftarrow (h_0 + t_0)u - t_0 v$.
  - $e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$.
  - Return e'.

We use a relation-based formulation of OW-CPA security. In particular, the adversary succeeds by outputting any valid error witness for the challenge ciphertext.

Definition 15 (Valid Error Witness). *For any candidate $e \in \mathrm{E}(t) \cup \{\perp\}$, define the Boolean predicate*

$$\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$$

to be true if and only if

$$e = (e_0, e_1, e_2) \in \mathrm{E}(t)$$

and

$$u = e_0 + e_1 r_1 + e_2 r_2, \qquad v = e_0 + e_1 t_1 + e_2 t_2.$$

For every other input, including $e = \perp$, the predicate evaluates to false.

Let $\mathcal{A}'$ be a PPT OW-CPA adversary against PKE_0. For a Boolean security game G, we write

$$\mathsf{Adv}_G(\mathcal{A}') := \Pr[G(\mathcal{A}') = 1].$$

The proof proceeds through the following sequence of games.

Game 1 $G_1(\mathcal{A}')$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \leftarrow t_2 + (r_1 + t_1)\frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$
6. $e \leftarrow \mathcal{A}'(t_1, t_2, r_1, r_2, u, v)$
7. return $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$

G_1 is the OW-CPA game of PKE_0. By definition,

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}') = \mathsf{Adv}_{G_1}(\mathcal{A}').$$

Game 2 $G_2(\mathcal{A}')$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \xleftarrow{\$} \mathcal{R}_{odd}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$
6. $e \leftarrow \mathcal{A}'(t_1, t_2, r_1, r_2, u, v)$
7. return $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$

In G_2, r_2 is replaced by a uniformly random element sampled from $\mathcal{R}_{odd}$.
We construct a distinguisher $\mathcal{A}_1$ for the 3-TRIKE problem as follows:

- Given a challenge (t_1, t_2, r_1, r_2).
- The distinguisher samples $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$.
- It then computes $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, \qquad v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$.
- It runs the adversary to obtain $e \leftarrow \mathcal{A}'(t_1, t_2, r_1, r_2, u, v)$.
- Then it outputs $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$.

If the challenge is sampled from the 3-TRIKE distribution, then the view of $\mathcal{A}'$ is distributed exactly as in G_1. If the challenge is uniformly distributed, then its view is distributed exactly as in G_2. Therefore,

$$|\mathsf{Adv}_{G_1}(\mathcal{A}') - \mathsf{Adv}_{G_2}(\mathcal{A}')| = \mathsf{Adv}_{\text{3-TRIKE}}(\mathcal{A}_1) \leq \mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1).$$

Game 3 $G_3(\mathcal{A}')$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \xleftarrow{\$} \mathcal{R}_{odd}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u, v \xleftarrow{\$} \mathcal{R}_{p(t)} \times \mathcal{R}$
6. $e \leftarrow \mathcal{A}'(t_1, t_2, r_1, r_2, u, v)$
7. return $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$

G_3 is obtained from G_2 by replacing (u, v) with uniformly random elements. We construct a distinguisher $\mathcal{A}_2$ for the (2,3)-TRIKE problem as follows:

- Given a instance $(t_1, t_2, r_1, r_2, u, v)$.
- The distinguisher runs the adversary to obtain $e \leftarrow \mathcal{A}'(t_1, t_2, r_1, r_2, u, v)$.
- Then it outputs $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$.

If the challenge is sampled from the $(2, 3)$-TRIKE distribution, the simulation is identical to G_2. If it is uniformly distributed, the simulation is identical to G_3. Consequently,

$$|\mathsf{Adv}_{G_2}(\mathcal{A}') - \mathsf{Adv}_{G_3}(\mathcal{A}')| = \mathsf{Adv}_{(2,3)\text{-TRIKE}}(\mathcal{A}_2) \leq \mathsf{Adv}_{(2,3)\text{-QCSD}}(\mathcal{B}_2).$$

In G_3, the adversary succeeds only if the uniformly sampled pair (u, v) admits a witness in $\mathrm{E}(t)$. For every fixed tuple (t_1, t_2, r_1, r_2), each $e \in \mathrm{E}(t)$ determines a single pair

$$(e_0 + e_1 r_1 + e_2 r_2,\ e_0 + e_1 t_1 + e_2 t_2)\,.$$

Hence, the number of pairs admitting a valid witness is at most $|\mathrm{E}(t)|$. Since (u, v) is uniform over $\mathcal{R}_{p(t)} \times \mathcal{R}$, whose cardinality is 2^{2r-1}, we have

$$\mathsf{Adv}_{G_3}(\mathcal{A}') \leq \varepsilon_{\mathsf{rnd}}(r, t), \qquad \varepsilon_{\mathsf{rnd}}(r, t) := \min\left\{1, \frac{\binom{3r}{t}}{2^{2r-1}}\right\}.$$

Under the QC-MDPC parameter regime $t = O(\sqrt{r})$, the standard bound $\binom{N}{t} \leq (\mathrm{e}N/t)^t$ gives

$$\varepsilon_{\mathsf{rnd}}(r, t) \leq 2^{-2r+1+t\log_2(3\mathrm{e}r/t)} = 2^{-2r+o(r)},$$

because

$$t\log_2(3\mathrm{e}r/t) = O(\sqrt{r}\log r) = o(r).$$

Consequently, $\varepsilon_{\mathsf{rnd}}(r, t)$ is negligible in r.

The preceding game transitions imply that, for any PPT OW-CPA adversary $\mathcal{A}'$ against PKE_0, there exist PPT distinguishers $\mathcal{B}_1$ and $\mathcal{B}_2$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}') \leq & |\mathsf{Adv}_{G_1}(\mathcal{A}') - \mathsf{Adv}_{G_2}(\mathcal{A}')| + |\mathsf{Adv}_{G_2}(\mathcal{A}') - \mathsf{Adv}_{G_3}(\mathcal{A}')| \\ & + \mathsf{Adv}_{G_3}(\mathcal{A}') \\ \leq & \mathsf{Adv}_{3\text{-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{(2,3)\text{-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r, t).\end{aligned}$$

A.2 IND-CPA Security

We now extend PKE_0 to the randomized encryption scheme PKE_1 with message space M. The full algorithms used in Theorem 5 are as follows.

- KeyGen_1: Run KeyGen_0 and return the same $(\mathsf{pk}, \mathsf{sk})$.
- $\mathrm{Encrypt}_1$:
  - Input: $\mathsf{pk} = (t_1, t_2, r_1, r_2)$ and $m \in \mathrm{M}$.
  - Output: $c \in \mathcal{R}_{p(t)} \times \mathcal{R} \times \mathrm{M}$.
  - $(e_0, e_1, e_2) \xleftarrow{\$} \mathrm{E}(t)$.
  - $u \leftarrow e_0 + e_1 r_1 + e_2 r_2$, $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$.
  - $c \leftarrow (u, v, m + \mathtt{L}(e_0, e_1, e_2))$.
  - Return c.
- $\mathrm{Decrypt}_1$:
  - Input: $\mathsf{sk} = (t_1, r_1, h_0, h_1, h_2)$ and $c = (u, v, c_2)$.
  - Output: $m' \in \mathrm{M} \cup \{\perp\}$.
  - $e' \leftarrow \mathrm{Decrypt}_0(\mathsf{sk}, (u, v))$.
  - If $e' = \perp$, return $\perp$.
  - $m' \leftarrow c_2 + \mathtt{L}(e')$.
  - Return m'.

The scheme follows a hybrid paradigm, where the algebraic structure induced by the secret-dependent relations implicitly defines a key encapsulation mechanism, while the masking function L, modeled as a random oracle, serves as a one-time pad for the message.

Let $\mathcal{A} = (\mathcal{A}_{choose}, \mathcal{A}_{guess})$ be a PPT IND-CPA adversary against PKE_1. The adversary is allowed to query the random oracle L, and we denote by L the set of queries made by $\mathcal{A}$ to the random oracle L.

We consider a series of games to prove that PKE_1 is IND-CPA.

Game 4 $G_4(\mathcal{A})$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \leftarrow t_2 + (r_1 + t_1)\frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$
6. $(m_0, m_1) \leftarrow \mathcal{A}_{choose}(t_1, t_2, r_1, r_2)$
7. $b \xleftarrow{\$} \{0, 1\}$
8. $c_2 \leftarrow m_b + \mathtt{L}(e_0^*, e_1^*, e_2^*)$
9. $b' \leftarrow \mathcal{A}_{guess}(t_1, t_2, r_1, r_2, u, v, c_2)$
10. return $(b = b')$

G_4 corresponds exactly to the IND-CPA security experiment of PKE_1 such that

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) = \left| \Pr[G_4(\mathcal{A}) = 1] - \frac{1}{2} \right|.$$

To analyze the advantage of the adversary $\mathcal{A}$ based on whether it queries any valid witness, we define the set of valid witnesses queried by the adversary as

$$L' := \{e \in L \mid \mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)\}.$$

We define the bad event:

$$\mathsf{BAD} := (L' \neq \emptyset).$$

This event occurs if the adversary queries $\mathtt{L}$ on any valid witness.

We now define the game G_5, in which the ciphertext component c_2 is replaced by a uniformly random element in M.

Game 5 $G_5(\mathcal{A})$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \leftarrow t_2 + (r_1 + t_1)\frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$
6. $(m_0, m_1) \leftarrow \mathcal{A}_{choose}(t_1, t_2, r_1, r_2)$
7. $b \xleftarrow{\$} \{0, 1\}$
8. $c_2 \xleftarrow{\$} \mathrm{M}$
9. $b' \leftarrow \mathcal{A}_{guess}(t_1, t_2, r_1, r_2, u, v, c_2)$
10. return $(b = b')$

We condition on the event BAD not occurring. Since e^* is a valid witness, we have $e^* \notin L$. The adversary never queries the random oracle with the challenge error vector e^*. Under the hypothesis that $\mathtt{L}$ is a random oracle, the value $\mathtt{L}(e^*)$ is indistinguishable from a uniformly random element of M. Hence, $c_2 \leftarrow m_b + \mathtt{L}(e_0^*, e_1^*, e_2^*)$ and $c_2 \xleftarrow{\$} \mathrm{M}$ follow the same distribution, that is, G_4 and G_5 are indistinguishable conditioned on $\neg\mathsf{BAD}$:

$$\Pr[G_4(\mathcal{A}) = 1 \mid \neg\mathsf{BAD}] = \Pr[G_5(\mathcal{A}) = 1 \mid \neg\mathsf{BAD}].$$

In G_5, the ciphertext component c_2 is sampled uniformly at random from M, independently of the challenge bit b. The adversary $\mathcal{A}$ obtains no information about the challenge bit. Thus, conditioned on the event BAD not occurring, we have

$$\Pr[G_5(\mathcal{A}) = 1 \mid \neg\mathsf{BAD}] = \frac{1}{2}.$$

To analyze the probability of the bad event BAD, we introduce a modified game based on G_4. In this game, we keep the execution identical to G_4, but change the success condition of $\mathcal{A}$ to whether it queries the random oracle $\mathtt{L}$ on a valid witness, i.e., whether $L' \neq \emptyset$.

Game 6 $G_6(\mathcal{A})$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \leftarrow t_2 + (r_1 + t_1)\frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$
6. $(m_0, m_1) \leftarrow \mathcal{A}_{choose}(t_1, t_2, r_1, r_2)$
7. $b \xleftarrow{\$} \{0,1\}$
8. $c_2 \leftarrow m_b + \mathtt{L}(e_0^*, e_1^*, e_2^*)$
9. $b' \leftarrow \mathcal{A}_{guess}(t_1, t_2, r_1, r_2, u, v, c_2)$
10. return $(L' \neq \emptyset)$

By construction,

$$\Pr[\mathsf{BAD}] = \Pr[G_6(\mathcal{A}) = 1].$$

Although G_6 captures the bad event in the original IND-CPA experiment, it cannot be simulated directly by an OW-CPA adversary, since constructing

$$c_2 = m_b + \mathtt{L}(e_0^*, e_1^*, e_2^*)$$

requires knowledge of the challenge error vector e^*. We therefore introduce the following game, in which c_2 is sampled uniformly at random.

Game 7 $G_7(\mathcal{A})$

1. $h_0, h_1, h_2 \xleftarrow{\$} \mathrm{K}(w)$
2. $t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$
3. $r_2 \leftarrow t_2 + (r_1 + t_1)\frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$
4. $(e_0^*, e_1^*, e_2^*) \xleftarrow{\$} \mathrm{E}(t)$
5. $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2,\ v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$
6. $(m_0, m_1) \leftarrow \mathcal{A}_{choose}(t_1, t_2, r_1, r_2)$
7. $b \xleftarrow{\$} \{0,1\}$
8. $c_2 \xleftarrow{\$} \mathrm{M}$
9. $b' \leftarrow \mathcal{A}_{guess}(t_1, t_2, r_1, r_2, u, v, c_2)$
10. return $(L' \neq \emptyset)$

The only difference between G_6 and G_7 is the generation of c_2. Until the first query $e \in L$ satisfying $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$, no valid witness has been queried. In particular, the challenge witness e^* has not been queried. Therefore, $\mathtt{L}(e^*)$ remains a fresh uniformly random element of M, and the value

$$m_b + \mathtt{L}(e^*)$$

is also uniformly distributed over M.

Consequently, the adversary's views in G_6 and G_7 are identically distributed until the first query on a valid witness. Any difference between the two games after this query cannot affect whether BAD has occurred. It follows that replacing $c_2 \leftarrow m_b + \mathtt{L}(e^*)$ with $c_2 \xleftarrow{\$} \mathrm{M}$ does not change the probability of the bad event. Hence,

$$\Pr[G_6(\mathcal{A}) = 1] = \Pr[G_7(\mathcal{A}) = 1].$$

The probability of G_7 can now be reduced to the OW-CPA security of PKE_0.

Lemma 2. *For any PPT IND-CPA adversary $\mathcal{A}$ against PKE_1 with queries set L on the random oracle $\mathtt{L}$, there exists a PPT OW-CPA adversary $\mathcal{A}'$ against PKE_0 such that*

$$\Pr[G_7(\mathcal{A}) = 1] = \mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}')$$

Proof. Let $\mathcal{A}$ be a PPT IND-CPA adversary against PKE_1 with queries set L on the random oracle $\mathtt{L}$. Then we construct a PPT OW-CPA adversary $\mathcal{A}'$ against PKE_0 as follows.

- Given an instance $(t_1, t_2, r_1, r_2, u, v)$ of the OW-CPA problem of PKE_0.
- $\mathcal{A}'$ samples $c_2 \xleftarrow{\$} \mathrm{M}$.
- Then it runs $\mathcal{A}$ following the steps of G_7 in $(t_1, t_2, r_1, r_2, u, v, c_2)$ to obtain a set $L \subset \mathrm{E}(t)$ that $\mathcal{A}$ queries the random oracle $\mathtt{L}$.
- For each $e = (e_0, e_1, e_2) \in L$, checks whether $\mathsf{Valid}_t(e, t_1, t_2, r_1, r_2, u, v)$ is true.
- If such an e exists, output it; otherwise output $\perp$.

By construction, if $\mathcal{A}$ queries a valid witness, then $\mathcal{A}'$ succeeds in the OW-CPA game. Otherwise, $\mathcal{A}'$ outputs $\perp$ and fails in the OW-CPA game. Therefore,

$$\Pr[G_7(\mathcal{A}) = 1] = \mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}').$$

□

Since

$$\Pr[G_4(\mathcal{A}) = 1 \mid \neg\mathsf{BAD}] = \Pr[G_5(\mathcal{A}) = 1 \mid \neg\mathsf{BAD}] = \frac{1}{2},$$

the law of total probability gives

$$\Pr[G_4(\mathcal{A}) = 1] = \frac{1}{2}\big(1 - \Pr[\mathsf{BAD}]\big) + \Pr[G_4(\mathcal{A}) = 1 \mid \mathsf{BAD}]\Pr[\mathsf{BAD}].$$

Therefore,

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) = \left|\Pr[G_4(\mathcal{A}) = 1 \mid \mathsf{BAD}] - \frac{1}{2}\right| \Pr[\mathsf{BAD}] \leq \frac{1}{2}\Pr[\mathsf{BAD}].$$

By Lemma 2 and the preceding game transition,

$$\Pr[\mathsf{BAD}] = \Pr[G_7(\mathcal{A}) = 1] = \mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}').$$

Applying Theorem 4 to the preceding bound, we conclude that, for any PPT IND-CPA adversary $\mathcal{A}$ against PKE_1, there exist PPT adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) &\leq \frac{1}{2}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}') \\ &\leq \frac{1}{2}\left(\mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{\text{(2,3)-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).\end{aligned}$$

A.3 IND-CCA Security

To apply the HHK transformations to PKE_1, we first relate its correctness error to the decoder's DFR.

Since our scheme may exhibit decryption failures, we rely on the notion of δ-correctness to quantify the failure probability and incorporate it into the security reduction. A public-key encryption scheme is said to be δ-correct if

$$\mathbb{E}\left[\max_{m\in \mathrm{M}} \Pr[\mathrm{Decrypt}(\mathsf{sk}, c) \neq m | c \leftarrow \mathrm{Encrypt}(\mathsf{pk}, m)]\right] \leq \delta,$$

where the expectation is taken over $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KeyGen}()$.

In our construction, the decryption failure occurs only if the decoding procedure fails. Observe that the variables $(h_0, h_1, h_2, t_1, t_2, r_1, r_2, u, v)$ do not depend on the message m, and the decoding procedure operates solely on these variables. Hence, the failure probability is independent of m, i.e., the scheme is message-agnostic. Therefore, the maximum over m becomes redundant, and the PKE is δ-correct if and only if the DFR of the decoder is less than δ.

Following the HHK framework, we first transform PKE_1 into a deterministic PKE_2 by using random oracles $\mathtt{H}_1$, $\mathtt{H}_2$, $\mathtt{H}_3$, $\mathtt{H}_4$:

- KeyGen_2:
  - Output: a public key pk and a secret key sk.
  - $(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$.
  - $\sigma \xleftarrow{\$} \mathrm{M}$.
  - $(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$.
  - $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$, $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$.
  - $\mathsf{pk} \leftarrow (\sigma, r_2)$, $\mathsf{sk} \leftarrow (\sigma, h_0, h_1, h_2)$.
  - Return $(\mathsf{pk}, \mathsf{sk})$.
- $\mathrm{Encrypt}_2$:
  - Input: $\mathsf{pk} = (\sigma, r_2)$ and $m \in \mathrm{M}$.
  - Output: $c \in \mathcal{R}_{p(t)} \times \mathcal{R} \times \mathrm{M}$.
  - $(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$.
  - $(e_0, e_1, e_2) \leftarrow \mathtt{H}_4(m, r_2)$.
  - $u \leftarrow e_0 + e_1 r_1 + e_2 r_2$, $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$.
  - $c \leftarrow (u, v, m + \mathtt{L}(e_0, e_1, e_2))$.
  - Return c.
- $\mathrm{Decrypt}_2$:

- Input: $\mathsf{sk} = (\sigma, h_0, h_1, h_2)$ and $c = (u, v, c_2)$.
- Output: $m' \in \mathrm{M} \cup \{\perp\}$.
- $(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$.
- $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$, $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$.
- $s \leftarrow (h_0 + t_0)u - t_0 v$.
- $e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$.
- If $e' = \perp$, return $\perp$.
- $m' \leftarrow c_2 + \mathtt{L}(e')$.
- $u' \leftarrow e'_0 + e'_1 r_1 + e'_2 r_2$, $v' \leftarrow e'_0 + e'_1 t_1 + e'_2 t_2$.
- If $e' \neq \mathtt{H}_4(m', r_2)$ or $(u', v') \neq (u, v)$, return $\perp$.
- Return m'.

Under the hypothesis of random oracle, replacing the explicit uniform sampling

$$t_1, t_2, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{even} \times \mathcal{R}_{odd}$$

with the hash outputs

$$t'_1 = \mathtt{H}_1(\sigma), \quad t'_2 = \mathtt{H}_2(\sigma), \quad r'_1 = \mathtt{H}_3(\sigma)$$

does not affect security because (t_1, t_2, r_1) and (t'_1, t'_2, r'_1) are computationally indistinguishable.

For any PPT OW-PCA adversary $\mathcal{B}'$ of PKE_2 that issues at most q_H queries to $\mathtt{H}_4$ and q_K queries to a plaintext-checking oracle $\mathcal{P}_{\mathrm{CO}}$, by applying the OW-PCA version of [15, Theorem 2], there exists a PPT IND-CPA adversary $\mathcal{A}$ against PKE_1 such that

$$\mathsf{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathrm{PKE}_2}(\mathcal{B}') \leq q_H \cdot \delta + \frac{2q_H + 1}{|\mathrm{M}|} + 3 \cdot \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) \tag{3}$$

We now apply the HHK transformation to convert PKE_2 into KEM with implicit rejection:

– KeyGen:
  - Run KeyGen_2 to obtain $\mathsf{pk} = (\sigma, r_2)$ and $\mathsf{sk}_2 = (\sigma, h_0, h_1, h_2)$.
  - $\sigma' \xleftarrow{\$} \mathrm{M}$.
  - $\mathsf{sk} \leftarrow (\sigma, \sigma', h_0, h_1, h_2)$.
  - Return $(\mathsf{pk}, \mathsf{sk})$.

– Encaps:
  - Input: $\mathsf{pk} = (\sigma, r_2)$.
  - Output: a shared key $K \in \mathrm{M}$ and a ciphertext $c \in \mathcal{R}_{p(t)} \times \mathcal{R} \times \mathrm{M}$.
  - $m \xleftarrow{\$} \mathrm{M}$.
  - $c \leftarrow \mathrm{Encrypt}_2(\mathsf{pk}, m)$.
  - $K \leftarrow \mathtt{K}(m, c)$.
  - Return (K, c).

– Decaps:
  - Input: $\mathsf{sk} = (\sigma, \sigma', h_0, h_1, h_2)$ and $c \in \mathcal{R}_{p(t)} \times \mathcal{R} \times \mathrm{M}$.

- Output: a shared key $K \in \mathrm{M}$.
- $m' \leftarrow \mathrm{Decrypt}_2((\sigma, h_0, h_1, h_2), c)$.
- If $m' \neq \perp$, set $K \leftarrow \mathtt{K}(m', c)$; otherwise, set $K \leftarrow \mathtt{K}(\sigma', c)$.
- Return K.

It is shown in [15, Theorem 4] that for any PPT IND-CCA adversary $\mathcal{B}$ against KEM making at most q_D queries to the decapsulation oracle $\mathcal{O}_{\mathrm{Decaps}}$ and q_K queries to the random oracle $\mathtt{K}$, there exists a PPT OW-PCA adversary $\mathcal{B}'$ against PKE_2 that issues at most q_K queries to $\mathcal{P}_{\mathrm{CO}}$ such that

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{KEM}}(\mathcal{B}) \leq \frac{q_K}{|\mathrm{M}|} + \mathsf{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathrm{PKE}_2}(\mathcal{B}'). \tag{4}$$

Combining (3) and (4) with Theorem 5, we conclude that, for any PPT $\mathsf{IND\text{-}CCA}$ adversary $\mathcal{B}$ against KEM making at most q queries to $\mathtt{K}$ or $\mathtt{H}_4$, there exist PPT adversaries $\mathcal{A}$, $\mathcal{B}_1$, and $\mathcal{B}_2$ such that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{KEM}}(\mathcal{B}) &\leq q \cdot \delta + \frac{3q+1}{|\mathrm{M}|} + 3 \cdot \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_1}(\mathcal{A}) \\
&\leq q \cdot \delta + \frac{3q+1}{|\mathrm{M}|} \\
&\quad + \frac{3}{2}\left(\mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{\text{(2,3)-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).
\end{aligned}$$

A.4 Security Proof with Weak-Key Rejection

We now consider the security implications of introducing weak-key rejection into our scheme.

Let $\mathrm{K}_{\$}(w) \subseteq \mathrm{K}(w)$ denote the subset of keys that successfully pass the weak key test. And

$$\mathsf{Acc} := [(h_0, h_1, h_2) \in \mathrm{K}_{\$}(w)]$$

denote the event that a key sampled from $\mathrm{K}(w)$ passes the weak key test. Let ρ be the rejection rate, so that

$$\Pr[\mathsf{Acc}] = 1 - \rho.$$

Since weak-key rejection repeatedly samples from $\mathrm{K}(w)$ until Acc occurs, the resulting key distribution is exactly the original key distribution conditioned on Acc.

For $i \in \{0, 1, 2\}$, let $\mathrm{PKE}_{i,\$}$ denote the variant of PKE_i in which (h_0, h_1, h_2) is sampled from $\mathrm{K}_{\$}(w)$. The rest of the algorithms remain unchanged. The KEM obtained by applying the same HHK transformation to $\mathrm{PKE}_{2,\$}$ is denoted by $\mathrm{KEM}_{\$}$.

Lemma 3 (OW-CPA Security under Weak-Key Rejection). *For any PPT OW-CPA adversary $\mathcal{A}'_{\$}$ against $\mathrm{PKE}_{0,\$}$, there exists a PPT OW-CPA adversary $\mathcal{A}'$ against PKE_0 such that*

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_{0,\$}}(\mathcal{A}'_{\$}) \leq \frac{1}{1-\rho}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}').$$

Proof. Let $\mathcal{A}'_{\$}$ be a PPT $\mathsf{OW\text{-}CPA}$ adversary against $\mathrm{PKE}_{0,\$}$. We construct a PPT $\mathsf{OW\text{-}CPA}$ adversary $\mathcal{A}'$ against the original PKE_0 as follows.

Given an OW-CPA instance $(t_1, t_2, r_1, r_2, u, v)$ of PKE_0, the adversary $\mathcal{A}'$ runs $\mathcal{A}'_{\$}$ on the same instance and returns exactly the candidate error witness output by $\mathcal{A}'_{\$}$.

Conditioned on Acc, the challenge given to $\mathcal{A}'_{\$}$ is distributed exactly as an OW-CPA challenge of $\mathrm{PKE}_{0,\$}$. Therefore,

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_{0,\$}}(\mathcal{A}'_{\$}) = \Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1 \mid \mathsf{Acc}].$$

Using $\Pr[\mathsf{Acc}] = 1 - \rho$, we obtain

$$\begin{aligned}\Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1 \mid \mathsf{Acc}] &= \frac{\Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1 \wedge \mathsf{Acc}]}{1-\rho} \\ &\leq \frac{\Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1]}{1-\rho} \\ &= \frac{1}{1-\rho}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}').\end{aligned}$$

□

Combining Lemma 3 with Theorem 4, for any PPT OW-CPA adversary $\mathcal{A}'_{\$}$, there exist PPT distinguishers $\mathcal{B}_1$ and $\mathcal{B}_2$ such that

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_{0,\$}}(\mathcal{A}'_{\$}) \leq \frac{1}{1-\rho}\left(\mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{\text{(2,3)-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).$$

Moreover, the game sequence and proof of Lemma 2 remain unchanged for the restricted-key variants. Therefore, for any PPT IND-CPA adversary $\mathcal{A}_{\$}$ against $\mathrm{PKE}_{1,\$}$, there exist PPT distinguishers $\mathcal{B}_1$ and $\mathcal{B}_2$ such that

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_{1,\$}}(\mathcal{A}_{\$}) \leq \frac{1}{2(1-\rho)}\left(\mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{\text{(2,3)-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).$$

Let $\delta_{\$}$ denote the average error-recovery failure probability under the accepted-key distribution. As above, it is a correctness bound for $\mathrm{PKE}_{1,\$}$.

The HHK reductions apply unchanged to the restricted-key variants, since the same accepted-key distribution is used throughout. Combining these reductions with the preceding IND-CPA bound for $\mathrm{PKE}_{1,\$}$, we conclude that, for any PPT $\mathsf{IND\text{-}CCA}$ adversary $\mathcal{B}_{\$}$ against $\mathrm{KEM}_{\$}$ making at most q queries to $\mathtt{K}$ or $\mathtt{H}_4$, there exist PPT adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{KEM}_{\$}}(\mathcal{B}_{\$}) \leq q \cdot \delta_{\$} + \frac{3q+1}{|\mathrm{M}|} \\ + \frac{3}{2(1-\rho)}\left(\mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{\text{(2,3)-QCSD}}(\mathcal{B}_2) + \varepsilon_{\mathsf{rnd}}(r,t)\right).\end{aligned}$$

Weak-key rejection introduces a factor of at most $1/(1-\rho)$ in the security bound inherited from the OW-CPA reduction, corresponding to a loss of at most

$-\log_2(1-\rho)$ bits. Therefore, for sufficiently small ρ, this loss is negligible. A concrete analysis is provided in Section 6.1.

The correctness error $\delta_{\$}$ is evaluated over the accepted-key distribution and satisfies $\delta_{\$} \leq \delta/(1-\rho)$. The effect of rejection on weak-key-induced failures is analyzed in Section 6.1.

B PKE and KEM Algorithms

B.1 Public Key Encryption

Algorithm 3: PKE.KeyGen

Input: None
Output: Public key pk and secret key sk

1: $(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$, $\sigma \xleftarrow{\$} \mathrm{M}$
2: $(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
3: $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$, $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$
4: $\mathsf{pk} \leftarrow (\sigma, r_2)$, $\mathsf{sk} \leftarrow (\sigma, h_0, h_1, h_2)$
5: **return** $(\mathsf{pk}, \mathsf{sk})$

Algorithm 4: PKE.Encrypt

Input: Public key $\mathsf{pk} = (\sigma, r_2)$ and error vector $e = (e_0, e_1, e_2) \in \mathrm{E}(t)$
Output: Ciphertext c

1: $(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
2: $u \leftarrow e_0 + e_1 r_1 + e_2 r_2$
3: $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$
4: $c \leftarrow (u, v)$
5: **return** c

Algorithm 5: PKE.Decrypt

Input: Secret key $\mathsf{sk} = (\sigma, h_0, h_1, h_2)$ and ciphertext c
Output: Candidate error vector $e' \in \mathrm{E}(t) \cup \{\perp\}$

1: $(u, v) \leftarrow c$
2: $(t_1, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_3(\sigma))$
3: $t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$
4: $s \leftarrow (h_0 + t_0)u - t_0 v$
5: $e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$
6: **return** e'

B.2 Key Encapsulation Mechanism

Algorithm 6: KEM.KeyGen

Input: Security parameter 1^λ
Output: Public key pk and secret key sk

repeat
  $(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$
  $b \leftarrow \mathsf{WeakKeyTest}((h_0, h_1, h_2), s, \kappa)$ {Algorithm 2}
until $b = 0$
$\sigma \xleftarrow{\$} \mathrm{M}$, $\sigma' \xleftarrow{\$} \mathrm{M}$
$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$, $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$
$\mathsf{pk} \leftarrow (\sigma, r_2)$, $\mathsf{sk} \leftarrow (\sigma, \sigma', h_0, h_1, h_2)$
return $(\mathsf{pk}, \mathsf{sk})$

Algorithm 7: KEM.Encaps

Input: Public key $\mathsf{pk} = (\sigma, r_2)$
Output: Transformed key K and ciphertext c

$m \xleftarrow{\$} \mathrm{M}$
$(e_0, e_1, e_2) \leftarrow \mathtt{H}_4(m, r_2)$
$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$u \leftarrow e_0 + e_1 r_1 + e_2 r_2$, $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$
$c \leftarrow (u, v, m + \mathtt{L}(e_0, e_1, e_2))$
$K \leftarrow \mathtt{K}(m, c)$
return K, c

Algorithm 8: KEM.Decaps

Input: Secret key $\mathsf{sk} = (\sigma, \sigma', h_0, h_1, h_2)$ and ciphertext c
Output: Transformed key K

$(u, v, c_2) \leftarrow c$
$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}$, $r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}$
$s \leftarrow (h_0 + t_0)u - t_0 v$
$e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$
if $e' = \perp$ **then**
  $K \leftarrow \mathtt{K}(\sigma', c)$
else
  $(e'_0, e'_1, e'_2) \leftarrow e'$
  $m' \leftarrow c_2 + \mathtt{L}(e'_0, e'_1, e'_2)$
  $u' \leftarrow e'_0 + e'_1 r_1 + e'_2 r_2$
  $v' \leftarrow e'_0 + e'_1 t_1 + e'_2 t_2$
  if $e' = \mathtt{H}_4(m', r_2)$ **and** $(u', v') = (u, v)$ **then**
    $K \leftarrow \mathtt{K}(m', c)$
  else
    $K \leftarrow \mathtt{K}(\sigma', c)$
  end if
end if
return K

C Rejection Model for Gathering Weak Keys

We describe the distance-spectrum model used to estimate the miss rates in Table 10. Following [21], we approximate the distances in the spectrum as independent random variables and count their collisions.

Let $g = d - \epsilon$ denote the number of support positions contained in the gathering interval, and let

$$0 \le X_1 < \cdots < X_g \le r_s - 1$$

be uniformly sampled from $\{0, \ldots, r_s - 1\}$. For $1 \le i < j \le g$, define the corresponding internal distance by

$$X'_{ij} = X_j - X_i.$$

We classify these $\binom{g}{2}$ distances according to $j - i = k$, where $1 \le k \le g - 1$. The k-th class contains $g - k$ distances, each following the distribution

$$\Pr[Y_k = u] = \frac{\binom{u-1}{k-1}\binom{r_s-u}{g-k}}{\binom{r_s}{g}}.$$

Distances involving support positions outside the gathering interval are approximated as independent and uniformly distributed over $\{1, \ldots, (r-1)/2\}$, and are grouped into an additional class.

Let $X'_1, \ldots, X'_N$ denote all modeled distances, where $N = \binom{d}{2}$. Their collision count is

$$Z = \frac{1}{2}\left(\sum_{a,b=1}^{N} [X'_a = X'_b] - N\right).$$

This is the model analogue of the normalized intra-block 4-cycle statistic

$$s_0(h_i) = \sum_{\delta=1}^{(r-1)/2} \binom{\mu(\delta, h_i)}{2}.$$

By enumerating collisions within and across the distance classes, we obtain the modeled distribution of Z. For the rejection threshold s, we define

$$\widehat{p}_{\text{miss}} = \Pr[Z \le s].$$

Since passing the full rejection test requires the gathering block to pass its intra-block test, this quantity estimates an upper bound on p_{miss} under the distance-spectrum approximation. The other checks can only further reduce the probability of acceptance.

We also test 10^7 independently sampled gathering weak keys and observe no key passing the rejection test. This provides empirical support for the effectiveness of the rejection test over the tested weak-key classes.

D Confidence Intervals and DFR Extrapolation

We describe the confidence-interval construction and extrapolation procedure used for the concrete DFR estimates. Recall that the FLS model in Section 5.1 gives the linear relation

$$y = ax + b, \qquad y = \frac{Q^{-1}(\mathrm{DFR})}{\sqrt{n}}, \qquad x = \frac{1}{n},$$

where n is the code length, and Q is the Gaussian Q-function. The coefficients a and b are fixed for the chosen key weight, error weight, and decoder configuration.

At each sampled circulant size r, let F denote the number of failures in N independent decoding experiments. Then

$$F \sim \mathrm{Bin}(N, \theta),$$

where θ is the true failure probability.

Let $\delta \in (0, 1)$ denote the significance level. Using the Clopper–Pearson interval, a $(1 - \delta)$-confidence interval for θ is given by

$$\theta_- = B^{-1}(\delta/2, F, N - F + 1), \qquad \theta_+ = B^{-1}(1 - \delta/2, F + 1, N - F),$$

with $\theta_- = 0$ when $F = 0$ and $\theta_+ = 1$ when $F = N$. Here, B^{-1} is the inverse of $x \mapsto B(x, a, b)$, the regularized incomplete beta function:

$$B(x, a, b) = \frac{\Gamma(a + b)}{\Gamma(a)\Gamma(b)} \int_0^x u^{a-1}(1 - u)^{b-1}\, du.$$

Since Q^{-1} is decreasing, the confidence interval for θ is transformed into the interval $[y^-, y^+]$, where

$$y^- = \frac{Q^{-1}(\theta_+)}{\sqrt{n}}, \qquad y^+ = \frac{Q^{-1}(\theta_-)}{\sqrt{n}}.$$

For any pair (r_i, r_j) such that $r_i < r_j < r_{test}$, let n_i, n_j, and n_{test} denote the corresponding code lengths, and write

$$x_i = \frac{1}{n_i}, \qquad x_j = \frac{1}{n_j}, \qquad x_{test} = \frac{1}{n_{test}}.$$

Under the FLS linear relation, the value at the target parameter is obtained by linear extrapolation:

$$y_{test} = (1 + A)y_j - Ay_i,$$

where

$$A = \frac{x_j - x_{test}}{x_i - x_j} > 0.$$

An interval for y_{test} is obtained by combining the extrema of the intervals for y_i and y_j. Specifically, the upper (lower) endpoint of y_{test} is obtained by using the lower (upper) endpoint of y_i together with the upper (lower) endpoint of y_j:

$$y^-_{test} = (1 + A)y^-_j - Ay^+_i, \qquad y^+_{test} = (1 + A)y^+_j - Ay^-_i.$$

Under the linear model, this interval contains y_{test} whenever both input intervals contain their respective values.

Repeating this procedure for all pairs (i, j) yields multiple candidate intervals. Among these candidates, we select the interval that yields the smallest upper DFR estimate. Since Q is decreasing, this is equivalent to selecting the interval with the largest lower endpoint y^-_{test}. The resulting estimate is

$$\mathrm{DFR}_{test} = Q\left(y^-_{test}\sqrt{n_{test}}\right).$$

The estimates in Table 7 use $\delta = 0.05$.