TRIKE

Algorithm Specifications and Supporting Documentation (v1.6)

Tianrui Wang
Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China

Yuxuan Li
Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China

Jing Tian
School of Integrated Circuits, Nanjing University, Suzhou, China

Yangcan Zhou
School of Integrated Circuits, Sun Yat-sen University, Shenzhen, China

Zijian Zhang
School of Integrated Circuits, Nanjing University, Suzhou, China

Anyu Wang
Institute for Advanced Study, Tsinghua University, Beijing, China

Contact: Anyu Wang anyuwang@tsinghua.edu.cn

Table of Contents

1 Algorithm Specification . . . 3
1.1 Properties of TRIKE . . . 3
1.2 Notations . . . 3
1.3 Security Definitions . . . 4
1.4 Specification of TRIKE . . . 5
1.4.1 Setup . . . 5
1.4.2 Public Key Encryption . . . 6
1.4.3 Key Encapsulation Mechanism . . . 6
1.4.4 Decoder . . . 7
1.5 Parameter Sets of TRIKE . . . 8
2 Performance Analysis . . . 9
2.1 Experimental Setup . . . 9
2.2 Storage and Communication Bandwidth . . . 10
2.3 Reference Implementation . . . 11
2.4 Optimized Implementation . . . 11
3 Decryption Failure Rate Analysis . . . 12
3.1 Theoretical DFR Model . . . 12
3.2 Validation of the FLS-Based DFR Model . . . 13
3.3 Concrete DFR Estimate . . . 13
3.4 Weak-Key Rejection . . . 14
3.5 Inter-Block Weak Keys . . . 15
3.6 Intra-Block Weak Keys . . . 16
4 Security Proofs . . . 18
4.1 Security Assumptions and Hardness Reduction . . . 18
4.2 OW-CPA Security . . . 21
4.3 IND-CPA Security . . . 24
4.4 IND-CCA Security . . . 28
5 Security Against Known Attacks . . . 31
5.1 Information Set Decoding . . . 32
A Rejection Model for Gathering Weak Keys . . . 33
B Confidence Intervals and DFR Extrapolation . . . 34

1 Algorithm Specification

This document presents **TRIKE**, a code-based key encapsulation mechanism called Triple Quasi-Cyclic Code Key Encapsulation Mechanism.

TRIKE uses a quasi-cyclic structure to reduce the public-key size from $O(n^2)$ to $O(n)$, where n is the code length. Its security analysis relies on the Quasi-Cyclic Syndrome Decoding (QCSD) and Ratio Quasi-Cyclic Codeword Finding (RQCCF) problems. RQCCF forms a public ratio from linear combinations of three sparse secret polynomials, combining the secret components into a single public ring element. An efficient distinguisher for RQCCF yields an efficient distinguisher for decisional 3-QCCF, as shown in Section 4.

The construction is conceptually related to BIKE [1] and HQC [2]. Its two algebraic ciphertext components are formed from three sparse error terms. Using the secret key, the receiver combines them into a syndrome $s = h_0e_0 + h_1e_1 + h_2e_2$, which is decoded using a bit-flipping decoder. The underlying deterministic encryption primitive encrypts an error vector and is analyzed for one-wayness. The KEM is constructed using the Fujisaki–Okamoto (FO) transformation [3] and the HHK framework with implicit rejection [4]. Following the salted FO construction [5], it includes a fresh 128-bit public salt in error generation and key derivation, and verifies both the recovered error vector and the re-encrypted ciphertext.

This document is organized as follows. Section 1 presents the algorithm specification. Section 2 provides implementation details and performance results. Section 3 analyzes the decryption failure rate. Section 4 presents the security reductions for the salted construction, and Section 5 analyzes the concrete complexity of known attacks.

1.1 Properties of TRIKE

TRIKE is built on a triple quasi-cyclic framework with a ratio-based mechanism for hiding the sparse secret structure. A direct three-component extension of BIKE would publish two separate ratios and expose additional relations between the secret components. The mixed relation of RQCCF avoids these additional constraints. The lower weight per secret component reduces the work in parity-check updates and unsatisfied-check counting, while shorter block lengths yield a compact public key.

Reliable decryption-failure-rate (DFR) estimates are needed at levels far below what can be measured directly in simulation. We adopt a finite-length scaling (FLS) model motivated by iteratively decoded LDPC codes, fitting the DFR at experimentally accessible block lengths and extrapolating to the target parameters. We also apply a hybrid weak-key rejection procedure combining a within-block 4-cycle statistic with a cyclic-intersection test across blocks [6,7]. For the selected parameter sets, the overall rejection rate is below 1%. The residual DFR contribution of the analyzed weak-key classes is quantified in Sections 3.5 and 3.6. The DFR model and key-validation procedure address complementary aspects of decoding reliability.

1.2 Notations

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 $\mathbf{y} = (y_0, \cdots, y_{n-1}) \in \mathbb{F}_2^n$, denote $w_H(\mathbf{y})$ to be the Hamming weight of $\mathbf{y}$, denote $\mathrm{Supp}(\mathbf{y})$ to be the support of $\mathbf{y}$, and denote $\mathbf{y}^{[a,b)} \coloneqq (y_a, y_{a+1}, \cdots, y_{b-1})$, where the subscripts are taken mod n.
- Let $\star$ be the vector operator Schur product. For vector $\mathbf{u}, \mathbf{v} \in \mathbb{F}_2^n$, the result of $\mathbf{u} \star \mathbf{v}$ is defined as
$$\mathbf{u} \star \mathbf{v} = (u_0 \cdot v_0, u_1 \cdot v_1, \cdots, u_{n-1} \cdot v_{n-1})$$
- Let $\mathcal{R} \coloneqq \mathbb{F}_2[x]/(x^r - 1)$. Every element $y \in \mathcal{R}$ has a unique representative $y(x) = y_0 + y_1x + \cdots + y_{r-1}x^{r-1}$ of degree less than r. We identify y with its coefficient vector $\mathbf{y} = (y_0, \ldots, y_{r-1}) \in \mathbb{F}_2^r$

whenever no ambiguity arises. A circulant matrix corresponding to the coefficients of y can be defined as

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

Then for any $y, z \in \mathcal{R}$, the coefficient vector of yz equals $\mathtt{rot}(\mathbf{y}) \cdot \mathbf{z}$. Moreover, the i-th column of $\mathtt{rot}(\mathbf{y})$ 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 $\mathcal{R}_{odd} = \{y \in \mathcal{R} \mid w_H(y) \equiv 1 \mod 2\}$, $\mathcal{R}_{even} = \{y \in \mathcal{R} \mid w_H(y) \equiv 0 \mod 2\}$
- Denote
  - $\mathrm{K}(w) \coloneqq \{(h_0, h_1, h_2) \in \mathcal{R}^3 \mid w_H(h_0) = w_H(h_1) = w_H(h_2) = w/3\}$,
  - $\mathrm{E}(t) \coloneqq \{(e_0, e_1, e_2) \in \mathcal{R}^3 \mid w_H(e_0) + w_H(e_1) + w_H(e_2) = t\}$,

1.3 Security Definitions

We use OW-CPA and OW-PCA for the deterministic error-vector encryption primitive, IND-CPA for a randomized transformed PKE where applicable, and IND-CCA for a KEM.

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 plaintext space* $\mathcal{P}$*, and let* $\mathcal{P}_{\mathrm{CO}}$ *be its public consistency predicate. For the deterministic TRIKE primitive,* $\mathcal{P} = \mathrm{E}(t)$ *and*

$$\mathcal{P}_{\mathrm{CO}}(m, c) = [m \in \mathcal{P} \ \wedge \ \mathrm{Encrypt}(\mathsf{pk}, m) = c].$$

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

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 plaintext space* $\mathcal{P}$*, and let* $\mathcal{P}_{\mathrm{CO}}$ *be the consistency oracle 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. $m^* \xleftarrow{\$} \mathcal{P}$
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)[1]). *Let* $\mathrm{KEM} = (\mathrm{KeyGen}, \mathrm{Encaps}, \mathrm{Decaps})$ *be a key encapsulation mechanism. For a PPT adversary* $\mathcal{A}$ *with a decapsulation oracle* $\mathcal{O}_{\mathrm{Decaps}}(c)$ *such that* $c \neq c^*$*, the* IND-CCA *game is defined as:*

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

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

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

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

1.4 Specification of TRIKE

1.4.1 Setup Given a parameter-set identifier, setup fixes r, w, t, l, the salt length $l_s = 128$, the hash functions, and the decoder.

- r is a prime block length for which 2 is a primitive root modulo r; the code length is $n = 3r$.
- $w = 3d$ is the total secret weight, with odd weight d in each block.
- t is the total error weight across the three blocks.
- l is the seed, message, fallback-secret, and shared-secret length in bits.

[1] 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.

Hash Functions. Write $\mathrm{M} = \{0,1\}^l$ and $\mathcal{S} = \{0,1\}^{128}$. The abstract algorithms use

$$\begin{aligned} \mathtt{H}_1, \mathtt{H}_2 &: \mathrm{M} \longrightarrow \mathcal{R}_{even}, & \mathtt{H}_3 &: \mathrm{M} \longrightarrow \mathcal{R}_{odd}, \\ \mathtt{H}_4 &: \mathrm{M} \times \mathcal{R}_{odd} \times \mathcal{S} \longrightarrow \mathrm{E}(t), & \mathtt{L} &: \mathcal{R}^3 \longrightarrow \mathrm{M}, \\ \mathtt{K} &: \mathrm{M} \times \mathcal{C} \longrightarrow \mathrm{M}, & \mathcal{C} &= \mathcal{R}^2 \times \mathrm{M} \times \mathcal{S}. \end{aligned}$$

The security analysis idealizes the relevant functions as random oracles. The domain of $\mathtt{L}$ includes arbitrary decoder candidates; its honest inputs lie in $\mathrm{E}(t)$.

Division and Decoding. Division denotes multiplication by the inverse in $\mathcal{R}$. The decoder is specified in Section 1.4.4.

1.4.2 Public Key Encryption The underlying PKE primitive has plaintext space $\mathrm{E}(t)$ and consists of the procedures in Algorithms 1–3. Encryption is deterministic for a supplied error vector. The base PKE samples unfiltered keys; the final KEM additionally applies weak-key rejection.

Algorithm 1: PKE.KeyGen

Input: Fixed system parameters
Output: Public key pk and secret key sk

$(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$, $\sigma \xleftarrow{\$} \mathrm{M}$
$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$t_0 \leftarrow (r_1 h_0 + h_1)/(r_1 + t_1)$, $r_2 \leftarrow (h_2 + t_0 t_2)/(t_0 + h_0)$
$\mathsf{pk} \leftarrow (\sigma, r_2)$, $\mathsf{sk} \leftarrow (\sigma, h_0, h_1, h_2)$
return $(\mathsf{pk}, \mathsf{sk})$

Algorithm 2: PKE.Encrypt

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

$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$u \leftarrow e_0 + e_1 r_1 + e_2 r_2$, $v \leftarrow e_0 + e_1 t_1 + e_2 t_2$
return (u, v)

Algorithm 3: PKE.Decrypt

Input: Secret key $\mathsf{sk} = (\sigma, h_0, h_1, h_2)$ and ciphertext (u, v)
Output: Candidate error vector e'

$(t_1, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_3(\sigma))$
$t_0 \leftarrow (r_1 h_0 + h_1)/(r_1 + t_1)$
$s \leftarrow (h_0 + t_0)u + t_0 v$
$e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$
return e'

1.4.3 Key Encapsulation Mechanism Algorithms 4–6 specify the salted KEM. A fresh independent salt $\zeta \in \mathcal{S}$ is included in both the error-generation input and the ciphertext input to key derivation. The fallback secret σ' is private. Key generation repeats the sparse-key sampling until the hybrid rejection test in Algorithm 8 accepts.

Algorithm 4: KEM.KeyGen

Input: Fixed system parameters and weak-key thresholds
Output: Public key pk and secret key sk

repeat
  $(h_0, h_1, h_2) \xleftarrow{\$} \mathrm{K}(w)$
until the weak-key rejection test accepts (h_0, h_1, h_2)
$\sigma \xleftarrow{\$} \mathrm{M}$, $\sigma' \xleftarrow{\$} \mathrm{M}$
$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$t_0 \leftarrow (r_1 h_0 + h_1)/(r_1 + t_1)$, $r_2 \leftarrow (h_2 + t_0 t_2)/(t_0 + h_0)$
$\mathsf{pk} \leftarrow (\sigma, r_2)$, $\mathsf{sk} \leftarrow (\sigma, \sigma', h_0, h_1, h_2)$
return $(\mathsf{pk}, \mathsf{sk})$

Algorithm 5: KEM.Encaps

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

$m \xleftarrow{\$} \mathrm{M}$, $\zeta \xleftarrow{\$} \mathcal{S}$
$e = (e_0, e_1, e_2) \leftarrow \mathtt{H}_4(m, r_2, \zeta)$
$(u, v) \leftarrow \mathrm{PKE.Encrypt}(\mathsf{pk}, e)$
$c_2 \leftarrow m \oplus \mathtt{L}(e)$
$c \leftarrow (u, v, c_2, \zeta)$, $K \leftarrow \mathtt{K}(m, c)$
return (K, c)

1.4.4 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 7.

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 $w_H\left(\mathrm{col}_i(h_j) \star s'\right)$, where $\mathrm{col}_i(h_j)$ 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 exactly $I_{\max} = 7$ iterations, without early termination. The final estimate e' is then returned as the decoding result.

Threshold Selection. The threshold used in Algorithm 7 depends on both the initial syndrome weight and the current syndrome weight. Let

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

The threshold is defined as

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

Algorithm 6: KEM.Decaps

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

$(u, v, c_2, \zeta) \leftarrow c$
$(t_1, t_2, r_1) \leftarrow (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma))$
$t_0 \leftarrow (r_1 h_0 + h_1)/(r_1 + t_1)$, $r_2 \leftarrow (h_2 + t_0 t_2)/(t_0 + h_0)$
$s \leftarrow (h_0 + t_0)u + t_0 v$, $e' \leftarrow \mathsf{Decoder}(h_0, h_1, h_2, s)$
$m' \leftarrow c_2 \oplus \mathtt{L}(e')$
$e'' \leftarrow \mathtt{H}_4(m', r_2, \zeta)$
$(u', v') \leftarrow \mathrm{PKE.Encrypt}((\sigma, r_2), e'')$
if $e' = e''$ **and** $(u', v') = (u, v)$ **then**
    $\mu \leftarrow m'$
else
    $\mu \leftarrow \sigma'$
end if
return $\mathtt{K}(\mu, c)$

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 described in [8,6]. 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 3.

1.5 Parameter Sets of TRIKE

We propose five parameter sets, TRIKE-1/3/5/7/9, targeting classical security levels of 128, 192, 256, 384, and 512 bits and quantum security levels of 80, 96, 128, 192, and 256 bits, respectively,

Algorithm 7: 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 $\lceil \max(T_{\text{now}}, T') \rceil$

for the underlying code-based problems. TRIKE-1/3/5 correspond to NIST security levels 1/3/5. The ICCS instantiation provides TRIKE-1/5/7/9. The detailed parameters are given in Table 1, and the decoder coefficients are listed in Table 2. The DFR targets are evaluated using the model and estimates in Section 3; the security of the complete KEM also depends on its symmetric instantiation and the reductions in Section 4.

Table 1. Suggested TRIKE parameters.

Instance	Classical	Quantum	r	w	t	l	Target DFR
TRIKE-1	128	80	9,437	81	201	256	2^{-128}
TRIKE-3	192	96	19,181	123	319	256	2^{-192}
TRIKE-5	256	128	32,987	165	429	256	2^{-256}
TRIKE-7	384	192	62,981	249	659	512	2^{-384}
TRIKE-9	512	256	103,043	333	877	512	2^{-512}

Weak-key thresholds and observed rejection rates are given in Section 3.4, with further analysis in Sections 3.5 and 3.6; serialized sizes and performance results are given in Section 2.

2 Performance Analysis

2.1 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 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

Table 2. Suggested decoder coefficients, with $I_{\max} = 7$.

Instance	c_a	c_b	δ
TRIKE-1	0.003234304810124655	6.772697599995158	2
TRIKE-3	0.002270835815881611	8.361152569725370	4
TRIKE-5	0.001814289691331794	10.245471078802690	5
TRIKE-7	0.001307161747313046	13.918322678122260	5
TRIKE-9	0.001063919519694423	18.546349651657760	6

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.

2.2 Storage and Communication Bandwidth

Let $B = \lceil r/8 \rceil$ and $M = \lceil l/8 \rceil$ denote the byte lengths of a ring element and a message, respectively. The serialization stores each of the $3d$ nonzero-bit positions in the private key as a 32-bit integer and pads the private key to a multiple of four bytes. Thus, the serialized lengths in bytes are

$$|\mathsf{sk}| = 4\left\lceil \frac{12d + 3B + 2M}{4} \right\rceil, \quad |\mathsf{pk}| = B + M,$$
$$|\mathsf{ct}| = 2B + M + 16, \qquad |\mathsf{ss}| = M.$$

The additional 16 ciphertext bytes encode the 128-bit salt ζ. The communication bandwidth, defined as the sum of the public-key and ciphertext lengths, is $3B + 2M + 16$ bytes. Table 3 gives the resulting serialized sizes. The ICCS implementations support TRIKE-1/5/7/9; TRIKE-3 is also included in the size comparison at NIST Security Level 3.

Table 3. TRIKE serialized key, salted ciphertext, and shared-secret sizes (in bytes).

Instance	TRIKE-1	TRIKE-3	TRIKE-5	TRIKE-7	TRIKE-9
Private key	3,928	7,752	13,096	24,744	40,104
Public key	1,212	2,430	4,156	7,937	12,945
Ciphertext	2,408	4,844	8,296	15,826	25,842
Shared secret	32	32	32	64	64
Bandwidth	3,620	7,274	12,452	23,763	38,787

The private key stores h_0, h_1, h_2 as lists of nonzero-bit positions, the packed binary vectors h_0, t_0, r_2, and the seeds σ, σ'. This representation avoids reconstructing data used during decapsulation. Other private-key representations, including seed-based storage, offer different space–time trade-offs. We therefore compare public-key size, ciphertext size, and bandwidth with other code-based KEMs.

Table 4 compares the TRIKE parameters with the cited parameter sets of Classic McEliece [9], BIKE [1], and HQC [2], at NIST Security Levels 1, 3, and 5. The TRIKE ciphertext sizes include the salt. Among these parameter sets, TRIKE has the smallest public keys; its bandwidth is approximately 54%, 54%, and 57% of the corresponding HQC bandwidth at Levels 1, 3, and 5, respectively.

Table 4. Communication sizes of the cited code-based KEM parameter sets (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	1,212	2,408	3,620
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	4,844	7,274
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	8,296	12,452

2.3 Reference Implementation

The reference implementation is written in ISO C99 for little-endian platforms and does not require 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 seven fixed decoding iterations and integer arithmetic for the threshold computation. The performance of the reference implementation with the ICCS interfaces is reported in Table 5.

Table 5. ICCS reference implementation of TRIKE (in 10^3 TSC ticks).

Instance	TRIKE-1	TRIKE-5	TRIKE-7	TRIKE-9
KeyGen	88,117	832,874	918,204	2,644,543
Encaps	9,022	76,320	81,118	232,851
Decaps	39,439	299,212	722,666	1,754,817

2.4 Optimized Implementation

The optimized implementation is written in C11 and uses AVX512 and Intel vector-PCLMULQDQ instructions on x86-64 processors to accelerate the main arithmetic and decoding operations.

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. The AVX512 code path retains the fixed seven-iteration decoder structure and runs in constant time.

The performance of the optimized implementation with the ICCS hash and PRG interfaces is reported in Table 6.

The performance figures in Table 6 are influenced by the hash and PRG interfaces provided by the ICCS benchmarking framework. Since these interfaces differ from those commonly used

Table 6. ICCS optimized implementation of TRIKE (in 10^3 TSC ticks).

Instance	TRIKE-1	TRIKE-5	TRIKE-7	TRIKE-9
KeyGen	2,730	11,871	23,792	46,297
Encaps	1,627	4,513	7,769	12,651
Decaps	2,103	9,357	21,525	48,281

in NIST PQC software benchmarks, we also benchmarked optimized TRIKE-1, TRIKE-3, and TRIKE-5 using AES for pseudorandom generation and SHA2 for hashing, consistent with the primitive choices in the optimized BIKE implementation. Table 7 compares the performance of optimized TRIKE and BIKE [1] at NIST Security Levels 1, 3, and 5, using similar benchmarking methods and API interfaces on different platforms.

Table 7. Performance comparison between optimized TRIKE and BIKE using AES/SHA2 (TRIKE: 10^3 TSC ticks; BIKE: 10^3 cycles).

	Level 1		Level 3		Level 5	
Instance	TRIKE-1	BIKE-1	TRIKE-3	BIKE-3	TRIKE-5	BIKE-5
KeyGen	2,162	366	5,551	1,049	10,525	2,643
Encaps	200	74	450	164	963	325
Decaps	682	1,177	2,150	3,512	5,750	9,106
Encaps + Decaps	882	1,251	2,600	3,676	6,713	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, the Decaps costs in Table 7 are reduced by 42.1%, 38.8%, and 36.9%, respectively, compared with BIKE. The combined encapsulation and decapsulation costs are reduced by 29.5%, 29.3%, and 28.8%, respectively. 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.

3 Decryption Failure Rate Analysis

3.1 Theoretical DFR Model

The DFR analysis of the bit-flipping decoder uses a model inspired by the finite-length scaling (FLS) behavior of iteratively decoded LDPC codes [12]. 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, with variable-node degree d and check-node degree $3d$. 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.

3.2 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 [6]. 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.

The fit supports the use of the FLS model over the measured range. Extrapolation far below the observable failure rates remains a modeling assumption; the fit alone does not establish a rigorous DFR upper bound or exclude an error floor.

3.3 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

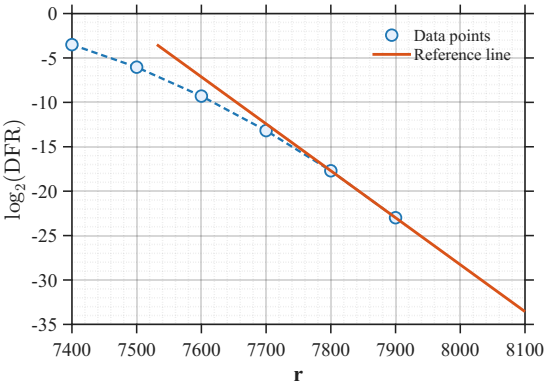


Fig. 1. Conventional linear extrapolation.

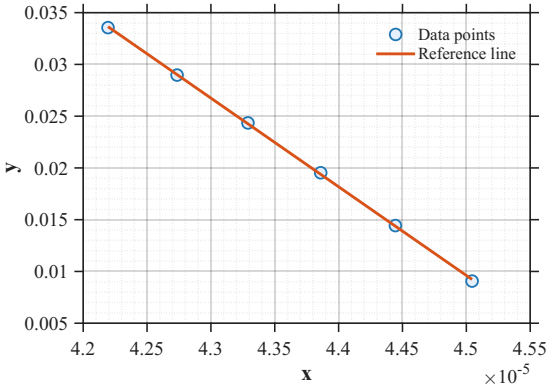


Fig. 2. FLS-based least-squares fitting.

FLS transformation and pairwise linear extrapolation. The interval construction and selection procedure are detailed in Appendix B. The resulting model-based DFR estimates are presented in Table 8. The measured input intervals use a 95% confidence level. The contribution of structured weak keys is examined separately below.

Table 8. Model-based DFR estimates for the selected parameter sets.

	TRIKE-1[2]	TRIKE-3	TRIKE-5	TRIKE-7	TRIKE-9
r	$9,437$	$19,181$	$32,987$	$62,981$	$103,043$
$\log_2(\text{DFR})$	-174.40	-193.41	-271.58	-393.30	-560.01

The FLS-based DFR model provides an extrapolated 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 [1], 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.

3.4 Weak-Key Rejection

To address the weak-key classes identified in [6,7], we employ a hybrid rejection test, combining an intra-block 4-cycle statistic inspired by [13] with a direct bound on cyclic inter-block intersections. For each circulant block h_i, let $\mu(\delta, h_i)$ count unordered support pairs at circular distance δ, where the distance between positions a, b is $\min\{|a-b|, r-|a-b|\}$. 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.

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

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

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

Here $\star$ denotes coefficientwise multiplication, so the Hamming weight counts the intersection of the two shifted supports.

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 8 summarizes the test, and rejected candidates are resampled during key generation.

Algorithm 8: 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 9. The resulting rejection rate is below 1% for all instances, corresponding to a security loss below 0.015 bits according to the analysis in Section 4.2.

Table 9. Weak-key rejection thresholds and measured rejection rates.

	TRIKE-1	TRIKE-3	TRIKE-5	TRIKE-7	TRIKE-9
s	28	59	105	235	433
κ	5	5	5	6	6
Rejection rate	0.79%	0.82%	0.53%	0.92430%	0.97583%

3.5 Inter-Block Weak Keys

The inter-block test directly targets Type III weak keys of [6]. 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 $\delta \in \{0, \ldots, r-1\}$. For independent uniform supports of size d, fixing the block pair and shift gives the hypergeometric probability

$$\frac{\binom{d}{m}\binom{r-d}{d-m}}{\binom{r}{d}}.$$

A union bound over the three block pairs and r shifts therefore gives

$$\tau_{III}(m) \le \min\left\{1, \frac{3r\binom{d}{m}\binom{r-d}{d-m}}{\binom{r}{d}}\right\}.$$

This bounds the class with an intersection of exactly m somewhere; a bound on the class with an intersection at least κ sums over $m \geq \kappa$.

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 TRIKE-1, TRIKE-3, and TRIKE-5, and $\kappa = 6$ for TRIKE-7 and TRIKE-9. Across all five parameter sets, we observed no decoding failures for $m < 13$ in our experiments, suggesting that the chosen thresholds $\kappa = 5$ and $\kappa = 6$ are conservative. Table 10 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.

Table 10. 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
TRIKE-7	6	-11.79
TRIKE-9	6	-10.19

3.6 Intra-Block Weak Keys

The Type I and Type II structures of [6] 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 [7], 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 $\epsilon = 0$ and evaluate the gathering classes listed in Table 11. The resulting estimates concern these classes and do not exhaust all possible weak-key structures. 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.

We 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

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

Writing $\rho = \Pr[\neg\mathsf{Acc}]$, the contribution of this class to the DFR among accepted keys satisfies

$$\Pr[\mathrm{Fail} \wedge \mathcal{G} \mid \mathsf{Acc}] \leq \frac{\Pr[\mathcal{G}]\Pr[\mathsf{Acc} \mid \mathcal{G}]}{1-\rho} = \frac{\Pr[\mathcal{G}]\, p_{\mathrm{miss}}}{1-\rho}.$$

Let $\widehat{p}_{\mathrm{miss}}$ denote the modeled probability that the gathering block passes its intra-block test under the distance-spectrum approximation of [14]. Since passing the full rejection test requires this block to pass, $\widehat{p}_{\mathrm{miss}}$ estimates an upper bound on p_{miss} under the model. The detailed model and an independent empirical consistency check are given in Appendix A. Substituting this estimate into the contribution bound gives a model-based evaluation; the extremely small probabilities in the resulting extrapolation are not measured directly.

For the post-rejection evaluation, we define

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

where Density denotes the density bound above. The product 2^{Sum} bounds the surviving weak-key density under the model.[3]

Table 11 summarizes the selected gathering weak-key classes, their density bounds and measured conditional DFRs, and the evaluation of the rejection test. The logarithmic miss-rate and Sum entries are $-\infty$ for the deterministic zero-miss cases.

Table 11. Evaluation of filtering effectiveness against weak keys satisfying the gathering property.

Instance	s	r_s	$\log_2$(DFR)	$\log_2$(Density)	$\log_2(\widehat{p}_{\mathrm{miss}})$	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	$-\infty$	$-\infty$
TRIKE-5	105	1050	−20.08	−249.20	$-\infty$	$-\infty$
		1000	−18.61	−253.11	$-\infty$	$-\infty$
		950	−17.70	−257.22	$-\infty$	$-\infty$
TRIKE-7	235	1700	−16.84	−407.28	$-\infty$	$-\infty$
		1800	−17.76	−400.36	$-\infty$	$-\infty$
		1900	−18.61	−393.81	$-\infty$	$-\infty$
TRIKE-9	433	1800	−14.41	−623.18	$-\infty$	$-\infty$
		1900	−15.44	−614.33	$-\infty$	$-\infty$
		2000	−16.61	−605.95	$-\infty$	$-\infty$

For TRIKE-1 and TRIKE-3, the largest values of Sum among the evaluated classes are −198.70 and −652.85, respectively. Under the distance-spectrum model, the corresponding per-class contribution bounds remain below the target DFRs even if every surviving weak key causes a decoding failure. For TRIKE-5, TRIKE-7, and TRIKE-9, all evaluated gathering classes are rejected deterministically. These conclusions concern only the evaluated weak-key classes.

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

4 Security Proofs

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[4].

All distributions in this subsection use ideal uniform sampling. For a decision problem, the distinguishing advantage is the absolute difference between the probabilities of outputting 1 on its structured and uniform distributions. This convention differs from the centered guessing advantage used for IND experiments.

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).

Definition 6 (3-QCCF Problem). *The 3-QCCF problem asks to decide with non-negligible advantage whether* (x, y) *came from the 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 the 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

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

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

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

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

Thus, it suffices to show that

$$y h_2 + t_1 y h_1 = h_0 + x h_1 + t_1 y h_1.$$

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

$$y h_2 = h_0 + x h_1,$$

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 the 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 the 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 the 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) $x = \frac{h_2 + t_2 h_0}{h_1 + t_1 h_0}$ as in the RQCCF distribution. Then (t_1, t_2, r_1, r_2) is distributed exactly as 3-TRIKE by the definition 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*

$$(e_0, e_1, e_2) \xleftarrow{\$} \mathrm{E}(t), \qquad x, y \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{odd},$$

computes $u \leftarrow e_0 + x e_2$, $v \leftarrow e_1 + y e_2$ *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 the (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 $u \leftarrow e_0 + r_1 e_1 + r_2 e_2$, $v \leftarrow e_0 + t_1 e_1 + t_2 e_2$ 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 the (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 the (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 $t_1, r_1 \xleftarrow{\$} \mathcal{R}_{even} \times \mathcal{R}_{odd}$.
- $\mathcal{B}$ sets $(t_2, r_2, u', v') = (t_1 y + x, r_1 y + x, r_1 v + u, t_1 v + u)$.
- $\mathcal{B}$ 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 the (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 the (2,3)-TRIKE distribution.

(2) (x, y, u, v) follows the 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 $\mathcal{A}$ 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 1.4.2 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 $t = O(\sqrt{r})$. The construction and proof below establish the following bound.

Theorem 1 (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*

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

Construction and Proof.
The algorithms of PKE_0 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 $u \leftarrow e_0^* + e_1^* r_1 + e_2^* r_2, \quad v \leftarrow e_0^* + e_1^* t_1 + e_2^* t_2$.
- 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,

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

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 $\binom{N}{t} \le (eN/t)^t$ gives

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

because

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

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

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

Weak-Key Rejection.

Let $\mathrm{K}_{\$}(w) \subseteq \mathrm{K}(w)$ denote the keys that pass Algorithm 8, and let

$$\mathsf{Acc} := [(h_0,h_1,h_2) \in \mathrm{K}_{\$}(w)], \qquad \Pr[\mathsf{Acc}] = 1-\rho.$$

We assume that $1-\rho$ is non-negligible. Repeated sampling until acceptance gives exactly the original key distribution conditioned on Acc. Write $\mathrm{PKE}_{0,\$}$ for this restricted-key variant; the same subscript will denote the corresponding variants of the subsequent constructions.

Lemma 2 (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*

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

Proof. Given an OW-CPA instance (t_1,t_2,r_1,r_2,u,v) of PKE_0, $\mathcal{A}'$ runs $\mathcal{A}'_{\$}$ on the same instance and returns its candidate error witness. Conditioned on Acc, this instance has exactly the distribution of an OW-CPA challenge of $\mathrm{PKE}_{0,\$}$. Therefore,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_{0,\$}}(\mathcal{A}'_{\$}) &= \Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1 \mid \mathsf{Acc}] \\ &= \frac{\Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1 \wedge \mathsf{Acc}]}{1-\rho} \\ &\le \frac{\Pr[G_{\mathsf{OW\text{-}CPA}}(\mathcal{A}') = 1]}{1-\rho} = \frac{\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}')}{1-\rho}.\end{aligned}$$

This argument concerns a nonnegative success probability, rather than a signed difference between two games. We use it at the OW-CPA stage and carry the resulting bound through the subsequent reductions. The reduction need not determine whether the challenge key is accepted. The multiplicative loss corresponds to at most $-\log_2(1-\rho)$ bits in the inherited hardness term. Rejection sampling here has expected polynomial running time; capping it after T independent attempts introduces a separate key-generation failure probability ρ^T and is not part of the exact conditional distribution above.

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 below yields the following consequence of Theorem 1.

Theorem 2 (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*

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

Construction and Proof.

The algorithms of PKE_1 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 $\mathtt{L}$, 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 $\mathtt{L}$, and we denote by L the set of queries made by $\mathcal{A}$ to the random oracle $\mathtt{L}$.

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 $\neg$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 3. *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 3 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 1 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

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

Weak-Key Rejection.

The game sequence and the witness extraction in Lemma 3 remain unchanged when the keys are sampled from $\mathrm{K}_{\$}(w)$. Thus, for any PPT IND-CPA adversary $\mathcal{A}_{\$}$ against $\mathrm{PKE}_{1,\$}$, there exist PPT OW-CPA adversaries $\mathcal{A}'_{\$}$ and $\mathcal{A}'$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_{1,\$}}(\mathcal{A}_{\$}) &\leq \frac{1}{2}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_{0,\$}}(\mathcal{A}'_{\$}) \\ &\leq \frac{1}{2(1-\rho)}\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathrm{PKE}_0}(\mathcal{A}'),\end{aligned} \tag{3}$$

where the second inequality is Lemma 2.

4.4 IND-CCA Security

We obtain the KEM from PKE_1 using the salted Fujisaki–Okamoto transformation with implicit rejection [4,5]. We first describe the public-key representation and the transformation, then state the correctness and security bounds.

Public-Key Representation.

The expanded public key of PKE_1 is (t_1, t_2, r_1, r_2). In the KEM, it is represented by (σ, r_2), where

$$\sigma \xleftarrow{\$} \mathrm{M}, \qquad (t_1, t_2, r_1) = (\mathtt{H}_1(\sigma), \mathtt{H}_2(\sigma), \mathtt{H}_3(\sigma)).$$

The remaining key-generation computations are unchanged. With independent random oracles, this representation preserves the key distribution used in the IND-CPA proof: the reduction samples σ and programs these three entries to its expanded public-key components before starting the adversary. This is an exact simulation when the adversary has no oracle state before key generation. It also holds with weak-key rejection, since acceptance depends only on $h = (h_0, h_1, h_2)$.

Salted Transformation.

Use the salt space $\mathcal{S} = \{0,1\}^{l_s}$, with $l_s = 128$ as in Section 1.4.1. For a fixed public key, write

$$F_{\mathsf{pk}}(m, \zeta) := \big(\mathrm{Encrypt}_1(\mathsf{pk}, m; \mathtt{H}_4(m, r_2, \zeta)), \zeta\big),$$

where the third argument of $\mathrm{Encrypt}_1$ specifies its error vector. Thus the intermediate encryption scheme PKE_2 samples $\zeta \xleftarrow{\$} \mathcal{S}$ and returns $F_{\mathsf{pk}}(m, \zeta)$; encryption is deterministic once (pk, m, ζ) is fixed.

The resulting KEM is the construction in Algorithms 4–6:

- Key generation uses the public-key representation above and stores an independent fallback secret $\sigma' \xleftarrow{\$} \mathrm{M}$.
- Encapsulation samples independent uniform $m \in \mathrm{M}$ and $\zeta \in \mathcal{S}$, sets $c = F_{\mathsf{pk}}(m, \zeta)$, and returns $(\mathtt{K}(m, c), c)$.

– Decapsulation parses $c = (u, v, c_2, \zeta)$, recovers a candidate e', and sets $m' = c_2 + \mathtt{L}(e')$. It returns $\mathtt{K}(m', c)$ if $e' \in \mathrm{E}(t)$, $e' = \mathtt{H}_4(m', r_2, \zeta)$, and $\mathrm{Encrypt}_0(\mathsf{pk}, e') = (u, v)$; otherwise it returns $\mathtt{K}(\sigma', c)$.

Explicit decoder failure, out-of-domain candidates, and non-canonical encodings take the rejection branch. The ciphertext domain of $\mathtt{K}$ extends to all byte strings of the prescribed length; decapsulation inputs are restricted to this length. Both branches hash the complete received ciphertext, including the salt.

Model and Correctness.

We work in the single-user classical random-oracle model, with ideal uniform sampling, independent domain-separated hash functions, and a deterministic, stateless decoder. The oracle $\mathtt{L}$ remains the same throughout the PKE and KEM experiments. For the fixed public key, the two additional FO oracles are $G(m, \zeta) = \mathtt{H}_4(m, r_2, \zeta)$ and $H(m, c) = \mathtt{K}(m, c)$; queries with other r_2 values are answered independently. All IND advantages use the centered guessing convention of Definitions 3 and 4.

Retain the accepted-key notation $\mathrm{KEM}_\$$ and rejection probability ρ from the preceding reductions. Let

$$p(h) := \Pr_{e \xleftarrow{\$} \mathrm{E}(t)} \left[\mathsf{Decoder}\left(h_0, h_1, h_2, \sum_{i=0}^{2} h_i e_i \right) \neq e \right], \qquad \mathbb{E}[p(h) \mid \mathsf{Acc}] \leq \delta_\$.$$

For an honest ciphertext, the decoder's input is $s = h_0 e_0 + h_1 e_1 + h_2 e_2$. Thus error recovery depends only on h and e, and successful error recovery implies successful message recovery and consistency checks. In particular,

$$\mathbb{E}\left[\max_{m \in \mathrm{M}} \Pr[\mathrm{Decrypt}_{1,\$}(\mathsf{sk}, \mathrm{Encrypt}_{1,\$}(\mathsf{pk}, m)) \neq m] \right] \leq \delta_\$. \tag{4}$$

Since a fresh $\mathtt{H}_4$ output is uniform in $\mathrm{E}(t)$, the average KEM correctness error is also at most $\delta_\$$. If only an unfiltered DFR bound δ is available, conditioning gives the valid choice $\delta_\$ = \min\{1, \delta/(1-\rho)\}$. Setting $\rho = 0$ and $\delta_\$ = \delta$ recovers the unfiltered scheme.

KEM Construction.

We collect the three procedures below in the notation of the preceding PKE constructions. Whenever an underlying PKE algorithm is invoked, the compressed keys are expanded as

$$\overline{\mathsf{pk}} = (t_1, t_2, r_1, r_2), \qquad \overline{\mathsf{sk}} = (t_1, r_1, h_0, h_1, h_2),$$

using σ and, for secret-key computations, h. This also makes explicit the key expansion implicit in the notation F_{pk} above. The resulting algorithms are equivalent to Algorithms 4–6.

– $\mathrm{KeyGen}_\$()$: Sample $h \xleftarrow{\$} \mathrm{K}_\$(w)$ and independent $\sigma, \sigma' \xleftarrow{\$} \mathrm{M}$. Derive (t_1, t_2, r_1) from σ and compute

$$t_0 \leftarrow \frac{r_1 h_0 + h_1}{r_1 + t_1}, \qquad r_2 \leftarrow \frac{h_2 + t_0 t_2}{t_0 + h_0}.$$

Return $\mathsf{pk} = (\sigma, r_2)$ and $\mathsf{sk} = (\sigma, \sigma', h_0, h_1, h_2)$.

– $\mathrm{Encaps}_\$(\mathsf{pk})$: Expand pk, sample independent $m \xleftarrow{\$} \mathrm{M}$ and $\zeta \xleftarrow{\$} \mathcal{S}$, and set

$$\begin{aligned} e &\leftarrow G(m, \zeta), & (u, v) &\leftarrow \mathrm{Encrypt}_0(\overline{\mathsf{pk}}, e), \\ c_2 &\leftarrow m + \mathtt{L}(e), & c &\leftarrow (u, v, c_2, \zeta). \end{aligned}$$

Return $(H(m, c), c)$.

– $\mathrm{Decaps}_\$(\mathsf{sk}, c)$: Parse $c = (u, v, c_2, \zeta)$; if the encoding is non-canonical, return $H(\sigma', c)$. Reconstruct the expanded keys and compute $e' \leftarrow \mathrm{Decrypt}_0(\overline{\mathsf{sk}}, (u, v))$. If no candidate in $\mathrm{E}(t)$ is returned, return $H(\sigma', c)$. Otherwise set

$$m' \leftarrow c_2 + \mathtt{L}(e'), \qquad e'' \leftarrow G(m', \zeta), \qquad (u', v') \leftarrow \mathrm{Encrypt}_0(\overline{\mathsf{pk}}, e'').$$

Return

$$\begin{cases} H(m', c), & e' = e'' \text{ and } (u', v') = (u, v), \\ H(\sigma', c), & \text{otherwise.} \end{cases}$$

Single-Challenge Security.

We use the implicit-rejection framework of [4] and the classical-ROM reductions in Theorems 1 and 3 of [5]. Their single-user, single-challenge specialization has no challenge-repetition term. The following bound also accounts for our explicit error-vector check and the accepted-key IND-CPA reduction.

Theorem 3 (IND-CCA Security). *Let $\mathcal{B}_\$$ be a PPT IND-CCA adversary against* $\mathrm{KEM}_\$$ *making at most q_H queries to $\mathtt{H}_4$, q_K queries to $\mathtt{K}$, and q_D decapsulation queries, and put $Q = q_H + 2q_K$. There exist PPT adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

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

Proof. The cited reductions first simulate decapsulation using public consistency checks, then replace the challenge by an independent randomized $\mathrm{PKE}_{1,\$}$ encryption with a uniform key. For non-challenge ciphertexts, a matching $H(m,c)$ query and a decapsulation query receive the same stored answer, in either order. If no matching H query has occurred, the simulation samples and stores a uniform decapsulation answer. Already issued challenge ciphertexts are excluded from this association. Checking an H query may require one additional G evaluation. Thus the simulated randomness-oracle budget is at most $q_H + q_K$, and the combined randomness-oracle and key-oracle budget is at most $Q = q_H + 2q_K$.

Only the error-vector check requires an additional observation. If $F_{\mathsf{pk}}(m,\zeta) = c$ and the decoder recovers $e = G(m,\zeta)$, our decapsulation accepts and returns $H(m,c)$. Conversely, acceptance supplies such a public candidate. Two distinct candidates for the same ciphertext cannot both have successful error recovery, because the decoder is deterministic. Thus any discrepancy caused by the error-vector check entails an error-recovery failure. For fixed h, each fresh G output is uniform; a union bound over at most $Q + q_D$ such outputs in each of the real and simulated executions, followed by averaging over accepted keys, gives the allowance $2(Q+q_D)\delta_\$$. Queries exposing the challenge pair are accounted for by the challenge-query reduction instead. Counting first evaluations of G suffices even for adaptive queries.

The implicit-rejection simulation contributes at most $q_K/|\mathrm{M}|$ for guessing the fallback secret. The single-challenge passive-security reduction contributes at most $2Q/|\mathrm{M}|$ and twice the underlying IND-CPA advantage. Together with the preceding failure allowance, this gives a PPT adversary $\mathcal{A}_1$ such that

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{KEM}_\$}(\mathcal{B}_\$) \le 2(Q+q_D)\delta_\$ + \frac{q_K+2Q}{|\mathrm{M}|} + 2\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE}_{1,\$}}(\mathcal{A}_1).$$

The simulation forwards all $\mathtt{L}$ queries, including those used in public consistency checks, to the underlying PKE experiment. Apply $q_K + 2Q \le 3Q$, Eq. (3), and Theorem 1 to obtain the stated bound.

Multi-Ciphertext Security.

Now allow at most N challenge encapsulations under the same public key, all using one hidden bit to choose between real and independent uniform keys. Challenges and decapsulation queries may be interleaved; only ciphertexts already returned as challenges are excluded from decapsulation. The budgets q_H, q_K, q_D count all queries over the entire experiment. Write $\mathsf{IND\text{-}CPA}, N$ for the corresponding PKE experiment with at most N left-or-right encryption queries under one hidden bit. For $1 \le a \le N$, define

$$\varepsilon_{\mathsf{occ}}(N,a) := \min\left\{1, \binom{N}{a+1}|\mathcal{S}|^{-a}\right\},$$

with $\binom{N}{N+1} = 0$. This bounds the probability that some salt occurs more than a times among N independent challenge salts. Following the multi-target argument of [5], we give a conservative bound that also explicitly accounts for a later challenge coinciding with an earlier decapsulation query.

Theorem 4 (Multi-Ciphertext Security). *Under the random-oracle model and total query budgets above, for any PPT adversary $\mathcal{B}_\$$ receiving at most N challenges and any $1 \le a \le N$, there exists a PPT adversary $\mathcal{A}_N$ such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CCA},N}_{\mathrm{KEM}_\$}(\mathcal{B}_\$) \le 2(Q+q_D)\delta_\$ &+ \frac{(2a+1)Q}{|\mathrm{M}|} \\ &+ \frac{N(N-1)+Nq_D}{|\mathrm{M}|\,|\mathcal{S}|} + \varepsilon_{\mathsf{occ}}(N,a) \\ &+ 2\mathsf{Adv}^{\mathsf{IND\text{-}CPA},N}_{\mathrm{PKE}_{1,\$}}(\mathcal{A}_N).\end{aligned}$$

The Nq_D summand is omitted when all challenges precede all decapsulation queries. This summand comes from the explicit ciphertext-coincidence estimate below.

Proof. Use the same public simulation and failure allowance as above. The multi-target argument for salted FO [5] controls challenge-message queries by the maximum salt multiplicity. For completeness, in the PKE reduction sample two independent messages for each challenge and append the same independent salt to the left-or-right encryption. The unused messages remain uniform and independent of the simulated transcript. Presample all N salts; this also covers an adaptively chosen number of challenges. Any fixed $a+1$ salts coincide with probability $|\mathcal{S}|^{-a}$, so a union bound gives $\varepsilon_{\mathsf{occ}}(N,a)$. Outside the salt-occupancy event, each of the at most Q queried pairs can match at most a unused messages. Let p and u be the probabilities of a hit on the encrypted and unused message lists, respectively. A reduction guessing the PKE bit from which list is hit, and guessing randomly if both or neither are hit, has centered advantage $|p-u|/2$. Therefore the probability p of querying an actual challenge pair is at most

$$2\mathsf{Adv}^{\mathsf{IND\text{-}CPA},N}_{\mathrm{PKE}_{1,\$}}(\mathcal{A}_N) + \frac{aQ}{|\mathrm{M}|} + \varepsilon_{\mathsf{occ}}(N,a).$$

This comparison includes queries made before the corresponding challenge. Guessing the independent fallback secret contributes at most $q_K/|\mathrm{M}|$, and repetition of a challenge message–salt pair has probability at most $\binom{N}{2}/(|\mathrm{M}|\,|\mathcal{S}|)$.

For interleaved queries, a fresh challenge in the public simulation may equal a previous decapsulation query. Conditional on its error vector, the fresh uniform message makes the masked message component uniform in M, independently of the fresh salt. A union bound therefore gives $Nq_D/(|\mathrm{M}|\,|\mathcal{S}|)$ for these coincidences. Combining the bounds and relaxing $q_K + aQ \le (2a+1)Q$ and $\binom{N}{2} \le N(N-1)$ proves the claim.

A standard hybrid over the N PKE challenges, followed by Eq. (3) and Theorem 1, gives

$$\begin{aligned}2\mathsf{Adv}^{\mathsf{IND\text{-}CPA},N}_{\mathrm{PKE}_{1,\$}}(\mathcal{A}_N) \le \frac{N}{1-\rho}\big(&\mathsf{Adv}_{\text{3-QCCF}}(\mathcal{B}_1) + \mathsf{Adv}_{\text{(2,3)-QCSD}}(\mathcal{B}_2) \\ &+ \varepsilon_{\mathsf{rnd}}(r,t)\big).\end{aligned}$$

The running times of the resulting adversaries include the public encryptions and oracle queries used in the simulations. Thus salting limits reuse of a message guess to challenges with the same salt. It does not increase the secret message entropy: the message-query denominator remains $|\mathrm{M}|$, and the reduction to the single-instance coding assumptions still incurs the generic factor N. These results concern the single-user classical ROM.

5 Security Against 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. Efficient distinguishers for these TRIKE problems yield efficient distinguishers for the 3-QCCF problem and the (2,3)-QCSD problem, respectively, as shown in Section 4.1.

5.1 Information Set Decoding

The best known attacks against the 3-QCCF problem and the (2,3)-QCSD problem are based on Information Set Decoding (ISD) algorithms. In our analysis, we estimate the attack complexities using the framework of [15], considering several representative ISD variants, including Prange's algorithm [16], Stern's algorithm [17,18,19], BJMM [20,21], BJMMplus [22], Both-May [23] and May-Ozerov [24].

The classical time complexities in Table 12 and Table 13 already account for the quasi-cyclic structure through decoding one out of many (DOOM), so no additional reduction is applied.

Table 12. Key recovery time complexity (in $\log_2$ scale).

Instance	Prange	Stern	BJMM	BJMMplus	BothMay	MayOzerov	Complexity
TRIKE-1	155.13	**133.83**	134.20	134.20	134.02	134.02	**133.83**
TRIKE-3	223.65	**200.01**	200.32	200.32	200.12	200.11	**200.01**
TRIKE-5	291.69	266.44	266.46	**266.32**	266.37	266.37	**266.32**
TRIKE-7	426.66	399.11	398.35	398.35	398.25	**398.25**	**398.25**
TRIKE-9	561.19	532.01	531.25	530.92	530.91	**530.91**	**530.91**

Table 13. Message recovery time complexity (in $\log_2$ scale).

Instance	Prange	Stern	BJMM	BJMMplus	BothMay	MayOzerov	Complexity
TRIKE-1	147.04	131.90	131.96	132.20	131.75	**131.74**	**131.74**
TRIKE-3	218.20	201.01	200.74	200.39	200.04	**200.03**	**200.03**
TRIKE-5	284.12	265.59	264.66	264.54	264.45	**264.45**	**264.45**
TRIKE-7	420.65	400.08	398.50	398.31	398.12	**398.12**	**398.12**
TRIKE-9	549.65	527.69	526.36	525.71	525.63	**525.63**	**525.63**

Appendices

A Rejection Model for Gathering Weak Keys

We describe the distance-spectrum model used to estimate the miss rates in Table 11. Following [14], 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.

The model also includes a deterministic zero-miss case for $\epsilon = 0$. Before applying ϕ_δ, the $\binom{d}{2}$ pairwise distances occupy at most $r_s - 1$ classes; shifts and ϕ_δ preserve the cyclic distance multiplicities up to permutation. Since $\binom{\mu}{2} \ge \mu - 1$ for every occupied class,

$$s_0(h_i) \ge \binom{d}{2} - (r_s - 1). \tag{5}$$

The same counting bound applies to Z. Thus, if $\binom{d}{2} > r_s - 1 + s$, every key in the class is rejected and $p_{\text{miss}} = \widehat{p}_{\text{miss}} = 0$.

As an independent empirical check, we apply the rejection test to 10^7 independently sampled gathering weak keys and observe no key passing the test. For the tested sampling distribution, zero misses in 10^7 independent trials yield the one-sided 95% upper confidence limit

$$p_{\text{miss}} \le 1 - 0.05^{1/10^7} < 2^{-21.67}.$$

This observation is consistent with the filtering effect predicted by the model at the available experimental resolution. It does not resolve the much smaller nonzero modeled tail probabilities in Table 11 and is used only as an empirical consistency check, not in the security bounds.

B 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 3.1 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

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

For TRIKE-1/3/5/7/9, the estimates in Table 8 use $\delta = 0.05$ for each input interval.

References

1. Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Santosh Ghosh, Shay Gueron, Tim Güneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Jan Richter-Brockmann, Nicolas Sendrier, Jean-Pierre Tillich, Valentin Vasseur, and Gilles Zémor. BIKE: bit flipping key encapsulation. Technical report, National Institute of Standards and Technology, 2024. available at https://bikesuite.org/files/v5.2/BIKE_Spec.2024.10.10.1.pdf.
2. Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, Jurjen Bos, Arnaud Dion, Jérôme Lacan, Jean-Marc Robert, Pascal Véron, Paulo L. Barreto, Santosh Ghosh, Shay Gueron, Tim Güneysu, Rafael Misoczki, Jan Richter-Brokmann, Nicolas Sendrier, Jean-Pierre Tillich, and Valentin Vasseur. Hamming quasi-cyclic (HQC). Technical report, National Institute of Standards and Technology, 2025. available at https://pqc-hqc.org/doc/hqc_specifications_2025_08_22.pdf.
3. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, pages 537–554, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.
4. Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Cham, November 2017.
5. Lewis Glabush, Kathrin Hövelmanns, and Douglas Stebila. On the multi-target security of post-quantum key encapsulation mechanisms. *IACR Communications in Cryptology*, 3(1), 2026.
6. Valentin Vasseur. *Post-quantum cryptography: a study of the decoding of QC-MDPC codes*. PhD thesis, Université de Paris, 2021.
7. Tianrui Wang, Anyu Wang, and Xiaoyun Wang. Exploring decryption failures of BIKE: New class of weak keys and key recovery attacks. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 70–100. Springer, Cham, August 2023.
8. Julia Chaulet. *Étude de cryptosystèmes à clé publique basés sur les codes MDPC quasi-cycliques*. PhD thesis, Paris 6, 2017.
9. Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wen Wang. Classic McEliece. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions.
10. Nir Drucker, Shay Gueron, and Dusan Kostic. Fast polynomial inversion for post quantum QC-MDPC cryptography. *Information and Computation*, 281:104799, 2021.
11. Nir Drucker, Shay Gueron, and Vlad Krasnov. Fast multiplication of binary polynomials with the forthcoming vectorized VPCLMULQDQ instruction. In *2018 IEEE 25th Symposium on Computer Arithmetic (ARITH)*, pages 115–119, 2018.
12. Abdelaziz Amraoui, Andrea Montanari, Tom Richardson, and Rüdiger Urbanke. Finite-length scaling for iteratively decoded LDPC ensembles. *IEEE Transactions on Information Theory*, 55(2):473–498, 2009.
13. Gretchen L. Matthews and Emily McMillon. A combinatorial approach to avoiding weak keys in the BIKE cryptosystem. *DCC*, 93(8):3413–3436, 2025.
14. Nicolas Sendrier. BIKE decoding failure, weak keys & error floors. In *2024 NIST Post-Quantum Cryptography Seminar*, Gaithersburg, United States, September 2024. NIST.
15. Andre Esser, Javier A. Verbel, Floyd Zweydinger, and Emanuele Bellini. SoK: CryptographicEstimators - a software library for cryptographic hardness estimation. In *AsiaCCS*. ACM, 2024.
16. Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.
17. Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Attacking and defending the McEliece cryptosystem. In Johannes Buchmann and Jintai Ding, editors, *PQCrypto 2008*, pages 31–46. Springer, Berlin, Heidelberg, October 2008.
18. Christiane Peters. Information-set decoding for linear codes over F_q. In Nicolas Sendrier, editor, *PQCrypto 2010*, pages 81–94. Springer, Berlin, Heidelberg, May 2010.
19. Jacques Stern. A method for finding codewords of small weight. In Gérard Cohen and Jacques Wolfmann, editors, *Coding Theory and Applications*, pages 106–113, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg.
20. Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 520–536. Springer, Berlin, Heidelberg, April 2012.

21. Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 107–124. Springer, Berlin, Heidelberg, December 2011.
22. Andre Esser and Floyd Zweydinger. New time-memory trade-offs for subset sum: Improving ISD in theory and practice. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 360–390. Springer, Cham, April 2023.
23. Leif Both and Alexander May. Decoding linear codes with high error rate and its impact for LPN security. In Tanja Lange and Rainer Steinwandt, editors, *PQCrypto 2018*, pages 25–46. Springer, Cham, 2018.
24. Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 203–228. Springer, Berlin, Heidelberg, April 2015.