

An Efficient Hybrid Key Exchange Mechanism

Benjamin D. Kim, Vipindev Adat Vasudevan, Alejandro Cohen, Rafael G. L. D’Oliveira,
Thomas Stahlbuhk, and Muriel Médard

Abstract—We present CHOKe, a novel code-based hybrid key-encapsulation mechanism (KEM) designed to securely and efficiently transmit multiple session keys simultaneously. By encoding n independent session keys with an individually secure linear code and encapsulating each resulting coded symbol using a separate KEM, CHOKe achieves computational individual security – each key remains secure as long as at least one underlying KEM remains unbroken. Compared to traditional serial or combiner-based hybrid schemes, CHOKe reduces computational and communication costs by an n -fold factor. Furthermore, we show that the communication cost of our construction is optimal under the requirement that each KEM must be used at least once.

I. INTRODUCTION

Public-key cryptography is computationally expensive and inefficient for encrypting large amounts of data [1]. To address this, key encapsulation mechanisms (KEMs) are used to securely transmit session keys for symmetric-key cryptography [2]. The process involves a transmitter (Alice) using the recipient’s (Bob) public key to encrypt a randomly generated session key, producing a ciphertext. The recipient decrypts this ciphertext with their private key to retrieve the session key. This session key, now securely shared, is used within a symmetric cryptosystem, which is far more efficient for encrypting and decrypting large volumes of data. This method leverages the security of public-key cryptography for key exchange while benefiting from the speed and efficiency of symmetric encryption.

Current widely deployed KEMs are vulnerable to quantum attacks [3], [4], making them unsafe in a future where quantum computers become operational. Consequently, there is an urgent need for quantum-resistant KEMs to safeguard cryptographic systems against these emerging threats. However, many quantum-resistant algorithms are relatively new and have not undergone the extensive scrutiny of traditional (non-quantum secure) cryptographic algorithms [5]. Table I summarizes the current status of selected KEMs that were proposed during the NIST Post-Quantum Cryptography (PQC) standardization process as of April 2025. In particular, CRYSTALS-Kyber has been officially selected and standardized [6]. Classic McEliece, BIKE, and HQC are alternate finalists and remain under evaluation in Round 4 [7]–[9]. SIKE as well was an alternate candidate proposed in Round 4 [10], but has since been broken via an efficient key-recovery attack [11].

B. D. Kim is with University of Illinois Urbana-Champaign, USA (e-mail: bdkim4@illinois.edu). V. Adat Vasudevan and M. Médard are with Massachusetts Institute of Technology, USA (email: {vipindev, medard}@mit.edu). A. Cohen is with Technion, Israel (e-mail: alecohen@technion.ac.il). R. G. L. D’Oliveira is with Clemson University, USA (e-mail: rdolive@clemson.edu). T. Stahlbuhk is with MIT Lincoln Laboratory, USA (e-mail: thomas.stahlbuhk@ll.mit.edu).

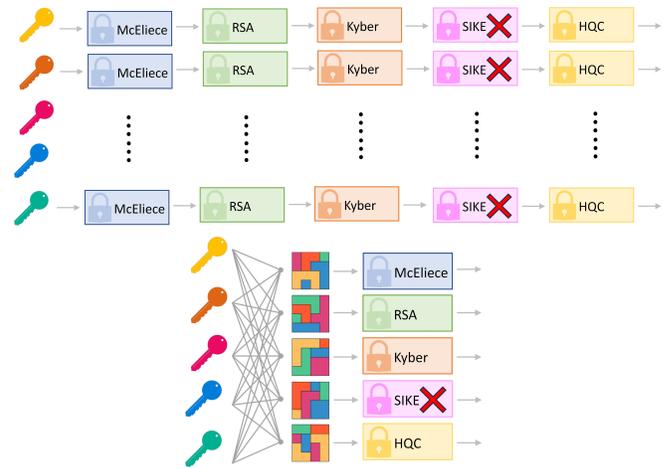


Fig. 1. KEM schemes: Serial Encapsulation (top, see Example 1) compared to CHOKe (bottom, see Example 3).

Algorithm	Status	Cryptographic Type	Ref.
CRYSTALS-Kyber	Standardized	Lattice-based (MLWE)	[6]
Classic McEliece	Alternate finalist	Code-based (Goppa)	[7]
BIKE	Alternate finalist	Code-based (QC-MDPC)	[8]
HQC	Alternate finalist	Code-based (LPN-style)	[9]
SIKE	Broken (2022) [11]	Isogeny-based (SIDH)	[10]

TABLE I
SELECTED KEMs FROM THE NIST PQC PROJECT (AS OF APRIL 2025).

To address this uncertainty, there has been a demand for hybrid key exchange algorithms [12]. These algorithms combine quantum-resistant KEMs with traditional, well-established KEMs. This approach provides a safety net, allowing users to benefit from the potential security of post-quantum algorithms while retaining the longer lived reliability of traditional cryptosystems. Hybrid systems are particularly valuable in contexts where regulatory requirements, such as FIPS compliance, mandate the continued use of traditional algorithms [13]. For users concerned about the future threat of retroactive decryption, also known as “Harvest now, decrypt later”, where adversaries can currently store encrypted data now which they might potentially decrypt later after a cryptographic breakthrough, hybrid key exchange offers a practical interim solution.

The primary goal of a hybrid key exchange mechanism is to establish a session key that remains secure as long as at least one of the component key exchange methods remains unbroken. A straightforward approach to achieving security in a hybrid key exchange mechanism is to concatenate KEMs together. In this scheme, the security of the session key is ensured as long as at least one KEM remains unbroken. However, this method multiplies the computational cost by

the number of KEMs utilized, as it requires encrypting the data once with each KEM.

In this paper, we present CHOKe (Code-based Hybrid Optimal Key Exchange), an efficient hybrid key encapsulation mechanism designed to transmit multiple session keys simultaneously. The protocol operates by encoding each session key into a linear combination using an individually secure code and encapsulating each encoded part with its corresponding KEM. This guarantees individual secrecy, as the adversary learns nothing about any single session key provided at least one KEM remains secure.

Our main contributions are as follows:

- In Theorem 1, we analyze the computational cost of CHOKe and show an n -fold reduction compared to conventional hybrid schemes. Specifically, CHOKe requires performing only one encapsulation and decapsulation per KEM, regardless of the number of session keys.
- In Theorem 2, we determine the exact communication cost of CHOKe, showing that the total bandwidth is the sum of ciphertext lengths from each KEM. Furthermore, in Theorem 3, we prove this cost is optimal under the assumption that session keys are incompressible and that each KEM must be invoked at least once to securely encapsulate all the keys.
- In Theorem 4, we prove the security of CHOKe using a simulation-based approach, showing that an adversary who breaks all but one of the underlying KEMs learns no information about any individual session key.
- In Section VI, we discuss potential security risks associated with related-key attacks. While CHOKe guarantees that individual session keys remain secure if at least one KEM is unbroken, breaking certain KEMs allows an adversary to learn linear combinations of keys. Therefore, we highlight the necessity of employing symmetric-key encryption schemes robust against related-key attacks, such as AES, when using keys transported by CHOKe.

A. An Example with Two KEMs

To show how our protocol works we compare it with two other schemes, simple concatenation, and the KEM Combiners scheme of [14].

Consider Alice wants to send two session keys, $k_1, k_2 \in \mathbb{F}_q^d$ chosen uniformly at random, to Bob. Because of regulations, Bob must utilize a traditionally approved public-key encryption scheme KEM_1 .¹ But because of his concerns about the future of quantum computing he also wants to utilize a post-quantum scheme KEM_2 . For each KEM_i we denote the encapsulation function by $\text{KEM.enc}_i : \mathbb{F}_q^{d_i} \times \mathcal{P}_i \rightarrow \mathbb{F}_q^{\ell_i}$, the decapsulation function by $\text{KEM.dec}_i : \mathbb{F}_q^{\ell_i} \times \mathcal{S}_i \rightarrow \mathbb{F}_q^{d_i}$, the public key by $pk_i \in \mathcal{P}_i$ and the private key by $sk_i \in \mathcal{S}_i$. We assume anyone can know the public keys, but only Bob knows the private keys.

¹To simplify our presentation, we assume throughout that we are working over a finite field \mathbb{F}_q of sufficiently large size.

Example 1. (Serial Encapsulation). Alice encapsulates the session key k_i with KEM_2 and then encapsulates that with KEM_1 to obtain the ciphertexts

$$\begin{aligned} c_1 &= \text{KEM.enc}_2(\text{KEM.enc}_1(k_1, pk_1), pk_2), \\ c_2 &= \text{KEM.enc}_2(\text{KEM.enc}_1(k_2, pk_1), pk_2), \end{aligned}$$

which are then sent to Bob.

Bob utilizes the private key sk_i to decapsulate the two session keys,

$$\begin{aligned} k_1 &= \text{KEM.dec}_1(\text{KEM.dec}_2(c_1, sk_2), sk_1) \\ k_2 &= \text{KEM.dec}_1(\text{KEM.dec}_2(c_2, sk_2), sk_1). \end{aligned}$$

- Security: In order to obtain either key, k_1 or k_2 , the adversary must break both KEM_1 and KEM_2 .
- Computation cost: Each key must be encapsulated twice. Thus, the computational cost is that of performing two KEM_1 operations plus two KEM_2 operations.
- Communication cost: Alice transmits the two outer ciphertexts (c_1, c_2) produced by KEM_1 . Each ciphertext lies in $\mathbb{F}_q^{\ell_1}$, so the total bandwidth is $2\ell_1$ symbols of \mathbb{F}_q .

Example 2. (KEM Combining). Alice generates four random symbols p_1, p_2, p_3, p_4 . The session keys are created utilizing a pseudorandom function, $k_1 = \text{PRF}(p_1, p_2)$ and $k_2 = \text{PRF}(p_3, p_4)$. Alice performs the following encapsulations

$$\begin{aligned} c_1 &= \text{KEM.enc}_1(p_1, pk_1), & c_2 &= \text{KEM.enc}_2(p_2, pk_2), \\ c_3 &= \text{KEM.enc}_1(p_3, pk_1), & c_4 &= \text{KEM.enc}_2(p_4, pk_2), \end{aligned}$$

and sends c_1, c_2, c_3, c_4 to Bob.

Bob decapsulates the four ciphertexts

$$\begin{aligned} p_1 &= \text{KEM.dec}_1(c_1, sk_1), & p_2 &= \text{KEM.dec}_2(c_2, sk_2), \\ p_3 &= \text{KEM.dec}_1(c_3, sk_1), & p_4 &= \text{KEM.dec}_2(c_4, sk_2), \end{aligned}$$

to obtain p_1, p_2, p_3, p_4 and reconstructs the session keys $k_1 = \text{PRF}(p_1, p_2)$ and $k_2 = \text{PRF}(p_3, p_4)$.

- Security: To recover a session key, say k_1 , an adversary must obtain both p_1 and p_2 . Each block is protected by a different KEM, so breaking a single KEM_i is insufficient.
- Computation Cost: The scheme performs two KEM_1 and two KEM_2 operations – the same computational complexity as in Example 1. However, all four can execute in parallel. Hence, the time complexity is reduced to one encapsulation (or decapsulation) of the slower KEM, rather than the sum of all four.
- Communication cost: Alice sends four ciphertexts, (c_1, c_3) produced by KEM_1 and (c_2, c_4) produced by KEM_2 . Hence the total bandwidth is $2\ell_1 + 2\ell_2$ symbols of \mathbb{F}_q , whereas the serial construction of Example 1 needs only $2\ell_1$ symbols.

Example 3. (CHOKe). Alice concatenates the two session keys $K = [k_1, k_2]$ into a vector and multiplies it with a public generator matrix $\mathbf{G} = \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix} \in \mathbb{F}_q^{2 \times 2}$, obtaining

$$X = K\mathbf{G} = [k_1 + k_2, k_1 + 2k_2] = [X_1, X_2].$$

Then, she encapsulates each linear combination and sends the ciphertexts to Bob:

$$c_1 = \text{KEM.enc}_1(X_1, pk_1), \quad c_2 = \text{KEM.enc}_2(X_2, pk_2).$$

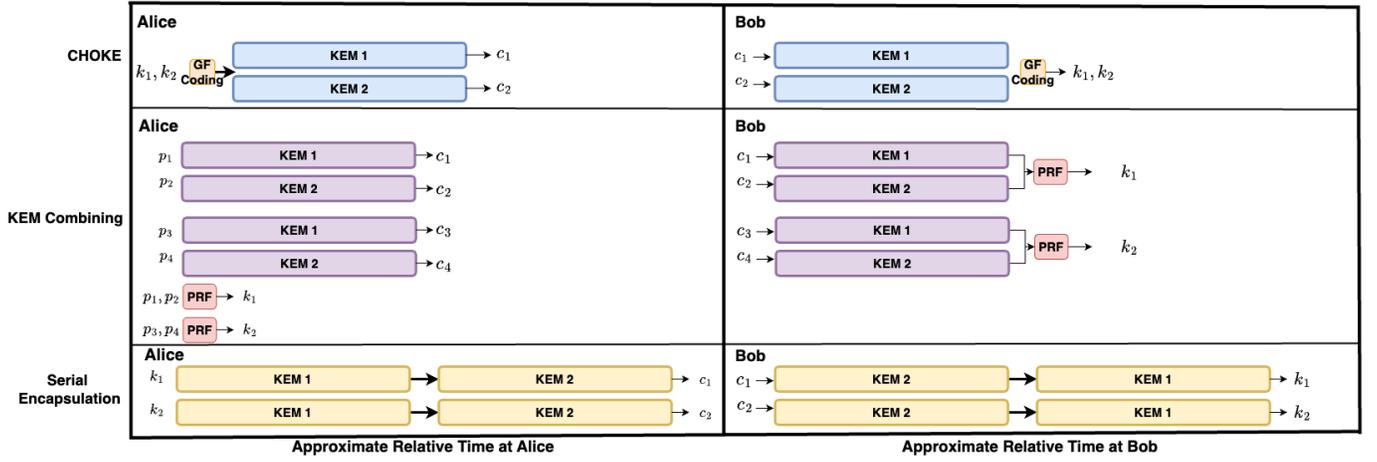


Fig. 2. Visual comparison of computational operations for transporting two session keys using the three hybrid KEM schemes described in the Introduction. The left side represents encapsulation operations at Alice (sender), and the right side represents decapsulation operations at Bob (receiver). Block sizes illustrate relative computational costs (not to scale). Even in this basic scenario, CHOKE reduces encapsulation and decapsulation operations by half compared to the other two schemes. Generalizing to n session keys, CHOKE requires only n encapsulations and decapsulations, whereas the other methods require n^2 .

Bob decapsulates

$$X_1 = \text{KEM.dec}_1(c_1, sk_1), \quad X_2 = \text{KEM.dec}_2(c_2, sk_2),$$

and recovers the original session keys $[k_1, k_2] = \mathbf{G}^{-1}X$.

- **Security:** Both ciphertexts are required to reconstruct K . If an adversary breaks one KEM he learns a linear combination of the session keys. In Theorem 4 we show that this reveals no information about any individual session key, i.e., each session key k_i is computationally indistinguishable from a uniform random one.
- **Computation Cost:** The scheme performs a single KEM_1 and KEM_2 operation – halving the computation cost when compared with the schemes in Examples 1 and 2.
- **Communication Cost:** Only two ciphertexts are sent— c_1 of length ℓ_1 and c_2 of length ℓ_2 . The total bandwidth is therefore $\ell_1 + \ell_2$ symbols of \mathbb{F}_q . This is less than both the symbols required by the serial scheme of Example 1 and by KEM Combining in Example 2.

B. Related Work

Secret sharing was originally proposed by Blakley and Shamir as a technique for safeguarding cryptographic keys. Karnin et al. [15] extended this concept by showing that when sharing multiple cryptographic keys simultaneously, performance can be significantly improved through the notion of individual security [16], where each key remains independently secure. Individual security has since been successfully applied across diverse communication and storage domains. Notable examples include single communication links [17], broadcast channels [18]–[21], multiple-access channels [22], [23], networks and multicast communications [24]–[26], algebraic security schemes [27], [28], terahertz wireless systems [29], angularly dispersive optical links [30], and distributed storage systems [31]–[35]. Individual security ensures that an eavesdropper obtaining any limited subset of the shared information learns no useful information about each message

individually, although they may acquire some insignificant, controlled leakage about combinations of the messages.

Our work builds upon the framework proposed by Cohen et al. [36], where Hybrid Universal Network-Coded Cryptography (HUNCC) was introduced to enhance the efficiency of cryptographic systems through partial encryption. Our scheme can be viewed as a specialized instance of HUNCC tailored explicitly for transmitting cryptographic keys, where we instead utilize full encryption for each coded symbol. If certain underlying KEMs become compromised, the security assurances of our scheme directly parallel those established by HUNCC under analogous partial encryption conditions. In this paper, we also introduce a new proof of security using the real-world/ideal-world (simulationist) paradigm, which can readily extend to provide an alternative, simulation-based security proof for the general HUNCC framework [37].

II. SECURITY DEFINITIONS

In this section we introduce security definitions we utilize in the paper. All honest parties and adversaries are modeled as probabilistic algorithms whose running time is bounded by a polynomial in the global security parameter κ .

Definition 1 (Negligible function). A function $f : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$ is negligible if for every constant $c > 0$ there exists N such that for all $\kappa > N$, $f(\kappa) < \kappa^{-c}$.

Definition 2 (Probabilistic polynomial-time (PPT)). An algorithm is probabilistic polynomial-time (PPT) if its running time is bounded by a polynomial in the length of its input.

A key encapsulation mechanism (KEM) enables two parties to securely establish a session key over an insecure channel.

Definition 3 (Key Encapsulation Mechanism over \mathbb{F}_q).¹ Fix a security parameter $\kappa \in \mathbb{N}$ and let $q = q(\kappa)$ be a prime power bounded by $\text{poly}(\kappa)$. For integers $d = d(\kappa)$ and $\ell = \ell(\kappa)$ define the key space $\mathcal{K} = \mathbb{F}_q^d$ and the ciphertext space $\mathcal{C} = \mathbb{F}_q^\ell$. A key-encapsulation mechanism (KEM) is a triple

of PPT algorithms $\text{KEM} = (\text{KEM.gen}, \text{KEM.enc}, \text{KEM.dec})$ with public/secret key sets \mathcal{P} and \mathcal{S} that satisfy:

- 1) **Key generation:** $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\kappa)$.
- 2) **Encapsulation:** given $\text{pk} \in \mathcal{P}$ and a uniformly random session key $m \in \mathcal{K}$, output $c := \text{KEM.enc}(m, \text{pk}) \in \mathcal{C}$.
- 3) **Decapsulation:** given $\text{sk} \in \mathcal{S}$ and $c \in \mathcal{C}$, output $m' := \text{KEM.dec}(c, \text{sk}) \in \mathcal{K}$.

Correctness: There exists a negligible function $\text{negl}(\cdot)$ such that for every κ ,

$$\Pr[\text{KEM.dec}(\text{KEM.enc}(m, \text{pk}), \text{sk}) = m] \geq 1 - \text{negl}(\kappa).$$

We now formalize the standard IND-CPA security notion a real-vs-ideal experiment.

Definition 4 (IND-CPA security via real/ideal experiments). Fix a KEM with key space $\mathcal{K} = \mathbb{F}_q^{d(\kappa)}$ and ciphertext space $\mathcal{C} = \mathbb{F}_q^{\ell(\kappa)}$. We say that KEM is indistinguishable under a chosen-plaintext attack (IND-CPA) if for every PPT distinguisher \mathcal{D} there exists a negligible function $\text{negl}(\kappa)$ such that, for all security parameters κ and for all $m, m' \in \mathcal{K}$, the two experiments below differ in \mathcal{D} 's acceptance probability by at most $\text{negl}(\kappa)$.

Real experiment $\text{Real}_m(\kappa)$

- 1) $(\text{pk}, \text{sk}) \leftarrow \text{KEM.gen}(1^\kappa)$;
- 2) $c \leftarrow \text{KEM.enc}(m, \text{pk})$;
- 3) output $\mathcal{D}(c, \text{pk})$.

Ideal experiment $\text{Ideal}_{m'}(\kappa)$

- 1) $(\text{pk}, \text{sk}) \leftarrow \text{KEM.gen}(1^\kappa)$;
- 2) $c^* \leftarrow \text{KEM.enc}(m', \text{pk})$;
- 3) output $\mathcal{D}(c^*, \text{pk})$.

Formally,

$$|\Pr[\mathcal{D}(\text{Real}_m(\kappa)) = 1] - \Pr[\mathcal{D}(\text{Ideal}_{m'}(\kappa)) = 1]| \leq \text{negl}(\kappa).$$

In Theorem 4 we show that if all KEMs in CHOKE's algorithm besides one are compromised, no information is leaked about any individual key. The proof works by showing that a computationally bounded adversary cannot distinguish between that setting and an information-theoretically secure setting with individual security.

Definition 5 (Individual security). An encoding of n independent uniform messages M_1, \dots, M_n into outputs X_1, \dots, X_n is individually secure if for every subset of at most $n-1$ outputs, say $(X_{i_1}, \dots, X_{i_{n-1}})$, and for each index $j \in \{1, \dots, n\}$,

$$I(M_j; X_{i_1}, \dots, X_{i_{n-1}}) = 0,$$

where $I(\cdot; \cdot)$ denotes the mutual information.

III. CHOKE ALGORITHM

We now present our construction, CHOKE, as Algorithm 1. In our setting, Alice wants to send n session keys k_1, \dots, k_n to Bob. Bob has n key encapsulation mechanisms $\text{KEM}_1, \dots, \text{KEM}_n$ with corresponding public and private keys $(pk_1, sk_1), \dots, (pk_n, sk_n)$. The public keys are made public.

Algorithm 1 CHOKE: Code-based Hybrid KEM

Key generation at Bob

- 1: **for** $i \leftarrow 1$ **to** n **do**
- 2: $(pk_i, sk_i) \leftarrow \text{KEM.gen}_i$
- 3: **end for**
- 4: publish (pk_1, \dots, pk_n)

Encapsulation at Alice

- 5: **Input:** session keys $K = [k_1, \dots, k_n] \in \mathbb{F}_q^n$
- 6: $X \leftarrow K \cdot G$ $\triangleright G$ is the public generator matrix
- 7: **for** $i \leftarrow 1$ **to** n **do**
- 8: $c_i \leftarrow \text{KEM.enc}_i(X_i, pk_i)$
- 9: **end for**
- 10: send (c_1, \dots, c_n) to Bob

Decapsulation at Bob

- 11: **for** $i \leftarrow 1$ **to** n **do**
 - 12: $X_i \leftarrow \text{KEM.dec}_i(c_i, sk_i)$
 - 13: **end for**
 - 14: $K \leftarrow G^{-1} \cdot X$ \triangleright recover $[k_1, \dots, k_n]$
 - 15: **return** K
-

Encapsulation: Alice concatenates the session keys into a vector $K = [k_1, \dots, k_n]$ and multiplies it with a public generator matrix G of an individually secure code to obtain $KG = [X_1, \dots, X_n]$. Then, she encapsulates each linear combination into the ciphertext $c_i = \text{KEM.enc}_i(X_i, pk_i)$ and sends it to Bob.

Decapsulation: Bob decapsulates each ciphertext c_i to obtain $X_i = \text{KEM.dec}_i(\text{KEM.enc}_i(X_i, pk_i), sk_i)$ and then recovers the session keys $[k_1, \dots, k_n] = G^{-1}[X_1, \dots, X_n]$.

IV. PERFORMANCE

In this section, we analyze both the computation and communication cost of CHOKE. Moreover, we show that CHOKE achieves an optimal communication cost.

A. Computation Cost

Theorem 1 (Computation Cost). *Let $\text{KEM}_1, \dots, \text{KEM}_n$ be the mechanisms used in CHOKE, and denote by E_i and D_i the computational cost of calling KEM.enc_i and KEM.dec_i , respectively. Then, the computational cost at Alice and Bob is $\sum_{i=1}^n E_i$ and $\sum_{i=1}^n D_i$, respectively.*

Proof. Algorithm 1 encapsulates each coded block X_i once and decapsulates the corresponding ciphertext c_i once. Hence KEM_i is invoked exactly one time for encapsulation and one time for decapsulation. \square

In both alternative hybrids every one of the n session keys is handled by all underlying mechanisms: once when Alice encapsulates the key and once when Bob decapsulates it. In the Serial Encapsulation scheme (exemplified in Example 1) every mechanism KEM_i is executed n times across the n keys, giving Alice a total cost of

$$\sum_{j=1}^n \sum_{i=1}^n E_i = n \sum_{i=1}^n E_i, \quad \text{and Bob a cost of } n \sum_{i=1}^n D_i.$$

The KEM-Combining construction (exemplified in Example 2) performs its encapsulations in parallel rather than in series, yet it still invokes each KEM_i once per key; hence its transmit-side and receive-side costs are the same as in the serial scheme, namely $n \sum_{i=1}^n E_i$ and $n \sum_{i=1}^n D_i$.

Since CHOKe protects all n keys with a single invocation of each mechanism, only $\sum_{i=1}^n E_i$ and $\sum_{i=1}^n D_i$ encapsulation and decapsulation operations are performed by Alice and Bob respectively. Thus CHOKe achieves an n -fold reduction in computational effort at both ends of the channel compared with either conventional hybrid approach.

B. Communication Cost

Theorem 2 (Communication cost). *Let $\text{KEM}_1, \dots, \text{KEM}_n$ be the mechanisms used in CHOKe, and let $\mathbb{F}_q^{\ell_i}$ be the output space of each $\text{KEM}_i.\text{enc}_i$. Then, the communication cost of CHOKe is $\sum_{i=1}^n \ell_i$ bits.*

Proof. Algorithm 1 outputs the tuple (c_1, \dots, c_n) with each $|c_i| = \ell_i$ and transmits it once. \square

In the Serial Encapsulation scheme (exemplified in Example 1) every session key is encapsulated by the outermost mechanism KEM_1 ; consequently Alice emits n ciphertexts, each of length at most $\ell_{\max} := \max_i \ell_i$, so the communication volume is $n\ell_{\max}$ bits. The KEM-Combining scheme (exemplified in Example 2) is more demanding. Because each key is protected by all mechanisms, the sender must transmit n ciphertexts for every KEM_i , giving a total of $n \sum_{i=1}^n \ell_i$ bits.

CHOKe, by contrast, needs only the $\sum_i \ell_i$ bits stated in Theorem 2. Thus its bandwidth is strictly smaller than that of KEM-Combining by a factor of $\approx n$, and it never exceeds the serial scheme's cost.

We now show that CHOKe achieves optimality in terms of communication cost. Specifically, we demonstrate that any hybrid KEM scheme that securely encapsulates n independent session keys and invokes each KEM_i at least once must transmit at least as many symbols as CHOKe does, namely $\sum_{i=1}^n \ell_i$ symbols of \mathbb{F}_q .

Theorem 3 (Optimality of CHOKe). *Let $\text{KEM}_1, \dots, \text{KEM}_n$ be mechanisms whose ciphertext lengths are ℓ_1, \dots, ℓ_n . Consider a hybrid protocol that, on input a tuple of session keys $k_1, \dots, k_n \in \mathcal{K}$, must invoke each KEM_i at least once. Then, the protocol must transmit at least $\sum_{i=1}^n \ell_i$ symbols of \mathbb{F}_q .*

Proof. By correctness, Bob must obtain each ciphertext $c_i \in \mathbb{F}_q^{\ell_i}$ exactly as produced by $\text{KEM}_i.\text{enc}$, since $\text{KEM}_i.\text{dec}$ may fail on any other string. Hence the message M that Alice sends in the protocol must allow Bob to recover the tuple (c_1, \dots, c_n) without error. Formally, the map $\mathcal{E} : (c_1, \dots, c_n) \rightarrow M$ implemented by the protocol must be injective. Otherwise, two distinct ciphertext tuples would be mapped to the same M , and Bob could not determine which tuple to decapsulate, contradicting correctness.

Because \mathcal{E} is injective, $|M|$ (the number of \mathbb{F}_q symbols in M) must be at least the length of the concatenation $c_1 \parallel \dots \parallel c_n$, which equals $\sum_{i=1}^n \ell_i$. Therefore every correct

protocol that invokes each KEM once necessarily transmits at least $\sum_{i=1}^n \ell_i$ symbols. \square

V. SECURITY

We now show that CHOKe retains the desired hybrid KEM property, namely that if all but one of the underlying KEMs are compromised, the adversary learns no information about any individual session key. Our proof uses a standard simulation-based argument [38]: we demonstrate that no computationally bounded adversary can distinguish between CHOKe's ciphertexts and an idealized scenario in which the ciphertexts achieve information-theoretic individual security.

Theorem 4 (Individual key IND security). *Suppose KEM_i is IND-CPA secure. Then in the CHOKe protocol, even if an adversary Eve breaks all KEMs except for KEM_i , no PPT distinguisher can learn any information about any individual session key k_i except with negligible probability in the security parameter κ .*

Proof. We prove the claim by a standard real-vs-ideal simulation argument. Fix any PPT adversary Eve. Let (k_1, \dots, k_n) be the uniformly random session keys and let

$$X = G[k_1; \dots; k_n] = (X_1, X_2, \dots, X_n),$$

be the coded linear combinations.

Without loss of generality, suppose that Eve breaks all the KEMs but the first one. Then, in the real execution, Eve's view is $\text{Real} = (c_1, X_2, \dots, X_n)$, where $c_1 = \text{KEM}_1.\text{enc}_1(X_1, pk_1)$. Define an ideal execution in which Eve instead sees $\text{Ideal} = (c_1^*, X_2, \dots, X_n)$ where $c_1^* = \text{KEM}_1.\text{enc}_1(0, pk_1)$. Because KEM_1 is IND-CPA secure, for every PPT distinguisher \mathcal{D} there is a negligible function $\text{negl}(\kappa)$ such that

$$|\Pr[\mathcal{D}(\text{Real}) = 1] - \Pr[\mathcal{D}(\text{Ideal}) = 1]| \leq \text{negl}(\kappa).$$

In particular this holds for any adversary Eve attempting to distinguish the two executions.

We observe that in the ideal execution c_1^* is independent of X_1, \dots, X_n . Since G is individually secure, it follows that $I(k_i; \text{Ideal}) = 0$ for every $i = 1, \dots, n$. Thus, Eve gains no information about any individual key in the ideal world. But by the IND-CPA bound above, Eve's view in the real world is computationally indistinguishable from her view in the ideal world. Therefore, Eve's advantage in learning any information about any individual k_i in the real view is at most $\text{negl}(\kappa)$. \square

VI. CHOKe AND RELATED-KEY ATTACKS

In CHOKe, if an underlying KEM is compromised, even though no information about any individual key is leaked, the adversary learns linear combinations of the session keys. When these session keys are subsequently used in symmetric-key encryption schemes, the exposure of linear relationships between the keys can potentially make the encryption vulnerable to related-key attacks [39].

Although modern symmetric-key encryption schemes, such as AES, are explicitly designed to be robust against related-key attacks [40]–[42], there have been theoretical scenarios where

knowledge of key relationships increased the adversary's success probability [43], [44].

This vulnerability is not unique to CHOKE; it also arises in other multi-secret sharing protocols. Multi-secret sharing schemes inherently leak correlations between secrets, similarly exposing the system to related-key attacks. Therefore, when deploying CHOKE, it is crucial to ensure that the symmetric encryption scheme utilizing the transported keys is resilient to related-key attacks.

REFERENCES

- [1] M. Tracy, W. Jansen, and M. McLarnon, "Guidelines on securing public web servers," *NIST Special Publication*, vol. 800, p. 44, 2002.
- [2] J. Katz and Y. Lindell, *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.
- [3] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134.
- [4] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997. [Online]. Available: <https://doi.org/10.1137/S0097539795293172>
- [5] E. Dubrova, K. Ngo, J. Gärtner, and R. Wang, "Breaking a fifth-order masked implementation of crystals-kyber by copy-paste," in *Proceedings of the 10th ACM Asia Public-Key Crypt. Workshop*, 2023, pp. 10–20.
- [6] R. Avanzi, J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Kyber: Algorithm Specifications and Supporting Documentation," 2022, NIST PQC Round 3 finalist, now standardized. [Online]. Available: <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022#kyber>
- [7] D. J. Bernstein, T. Lange, and P. Schwabe, "Classic McEliece: Submission and Documentation," 2022, NIST PQC Round 4 alternate finalist. [Online]. Available: <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>
- [8] P. S. L. M. Barreto, R. Misoczki, J.-P. Tillich *et al.*, "BIKE: Bit Flipping Key Encapsulation," 2022, NIST PQC Round 4 alternate finalist. [Online]. Available: <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>
- [9] P. Gaborit, A. Hauteville, J.-P. Tillich, and G. Zémor, "HQC: Hamming Quasi-Cyclic Key Encapsulation Mechanism," 2022, NIST PQC Round 4 alternate finalist. [Online]. Available: <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>
- [10] D. J. et al, "SIKE – Supersingular Isogeny Key Encapsulation," 2022. [Online]. Available: <https://sike.org/>
- [11] W. Castryck and T. Decru, "An efficient key recovery attack on SIDH," *Cryptology ePrint Archive*, no. 975, 2022. [Online]. Available: <https://eprint.iacr.org/2022/975>
- [12] D. Connolly, "Hybrid PQ/T Key Encapsulation Mechanisms," Internet Engineering Task Force, Internet-Draft draft-irtf-cfrg-hybrid-kems-03, Feb. 2025, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-irtf-cfrg-hybrid-kems/03/>
- [13] National Institute of Standards and Technology, "Module-lattice-based key-encapsulation mechanism standard." U.S. Department of Commerce, Washington, D.C., Tech. Rep. Federal Information Processing Standards Publications (FIPS) 203, 2024.
- [14] F. Giacon, F. Heuer, and B. Poettering, "Kem combiners," in *Public-Key Cryptography–PKC 2018: 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part 1 21*. Springer, 2018, pp. 190–218.
- [15] E. D. Karnin, J. W. Greene, and M. E. Hellman, "On secret sharing systems," *IEEE Trans. Inform. Theory*, vol. 29, no. 1, pp. 35–41, Jan. 1983.
- [16] A. Carleial and M. E. Hellman, "A note on wyner's wiretap channel (corresp.)," *IEEE Trans. Inform. Theory*, vol. 23, no. 3, pp. 387–390, May 1977.
- [17] D. Kobayashi, H. Yamamoto, and T. Ogawa, "Secure multiplex coding attaining channel capacity in wiretap channels," *IEEE Trans. on Inf. Theory*, vol. 59, no. 12, pp. 8131–8143, 2013.
- [18] A. S. Mansour, R. F. Schaefer, and H. Boche, "Secrecy measures for broadcast channels with receiver side information: Joint vs individual," in *2014 IEEE Inf. Theory Works. (ITW 2014)*. IEEE, 2014, pp. 426–430.
- [19] Y. Chen, O. O. Koyluoglu, and A. Sezgin, "On the individual secrecy rate region for the broadcast channel with an external eavesdropper," in *2015 IEEE Int. Symp. on Inf. Theory (ISIT)*. IEEE, 2015, pp. 1347–1351.
- [20] A. S. Mansour, R. F. Schaefer, and H. Boche, "The individual secrecy capacity of degraded multi-receiver wiretap broadcast channels," in *2015 IEEE Int. Conf. on Comm. (ICC)*. IEEE, 2015, pp. 4181–4186.
- [21] —, "On the individual secrecy capacity regions of the general, degraded, and gaussian multi-receiver wiretap broadcast channel," *IEEE Trans. on Inf. Fore. and Sec.*, vol. 11, no. 9, pp. 2107–2122, 2016.
- [22] M. Goldenbaum, R. F. Schaefer, and H. V. Poor, "The multiple-access channel with an external eavesdropper: Trusted vs. untrusted users," in *2015 49th Asilomar Conference on Signals, Systems and Computers*. IEEE, 2015, pp. 564–568.
- [23] Y. Chen, O. O. Koyluoglu, and A. H. Vinck, "On secure communication over the multiple access channel," in *2016 Int. Symp. on Inf. Theory and Its Applications (ISITA)*. IEEE, 2016, pp. 350–354.
- [24] K. Bhattad and K. R. Narayanan, "Weakly secure network coding," *NetCod, Apr.* vol. 104, 2005.
- [25] D. Silva and F. R. Kschischang, "Universal weakly secure network coding," in *2009 IEEE Inf. Theory Works. on Networking and Information Theory*. IEEE, 2009, pp. 281–285.
- [26] A. Cohen, A. Cohen, M. Médard, and O. Gurewitz, "Secure multi-source multicast," *IEEE Trans. on Comm.*, vol. 67, no. 1, pp. 708–723, 2018.
- [27] L. Lima, M. Médard, and J. Barros, "Random linear network coding: A free cipher?" in *2007 IEEE Int. Symp. on Inf. Theory*. IEEE, 2007, pp. 546–550.
- [28] J. Claridge and I. Chatzigeorgiou, "Probability of partially decoding network-coded messages," *IEEE Communications Letters*, vol. 21, no. 9, pp. 1945–1948, 2017.
- [29] A. Cohen, R. G. L. D'Oliveira, C.-Y. Yeh, H. Guerboukha, R. Shrestha, Z. Fang, E. Knightly, M. Médard, and D. M. Mittleman, "Absolute security in terahertz wireless links," *IEEE Journal of Selected Topics in Signal Processing*, 2023.
- [30] C.-Y. Yeh, A. Cohen, R. G. L. D'Oliveira, M. Médard, D. M. Mittleman, and E. W. Knightly, "Securing angularly dispersive terahertz links with coding," *IEEE Trans. on Inf. Forensics and Security*, 2023.
- [31] S. Kadhe and A. Sprintson, "Weakly secure regenerating codes for distributed storage," in *2014 International Symposium on Network Coding (NetCod)*. IEEE, 2014, pp. 1–6.
- [32] —, "On a weakly secure regenerating code construction for minimum storage regime," in *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2014, pp. 445–452.
- [33] N. Paunkoska, V. Kafedziski, and N. Marina, "Improved perfect secrecy of distributed storage systems using interference alignment," in *2016 8th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. IEEE, 2016, pp. 240–245.
- [34] —, "Improving the secrecy of distributed storage systems using interference alignment," in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2018, pp. 261–266.
- [35] J. Bian, S. Luo, Z. Li, and Y. Yang, "Optimal weakly secure minimum storage regenerating codes scheme," *IEEE Access*, vol. 7, pp. 151 120–151 130, 2019.
- [36] A. Cohen, R. G. L. D'Oliveira, S. Salamatian, and M. Médard, "Network coding-based post-quantum cryptography," *IEEE journal on selected areas in information theory*, vol. 2, no. 1, pp. 49–64, 2021.
- [37] A. Cohen, R. G. L. D'Oliveira, K. R. Duffy, and M. Médard, "Partial encryption after encoding for security and reliability in data systems," in *2022 IEEE Int. Symp. on Inf. Theory (ISIT)*. IEEE, 2022, pp. 1779–1784.
- [38] Y. Lindell, "How to simulate it—a tutorial on the simulation proof technique," *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*, pp. 277–346, 2017.
- [39] E. Biham, "New types of cryptanalytic attacks using related keys," *Journal of Cryptology*, vol. 7, pp. 229–246, 1994.
- [40] A. Biryukov and D. Khovratovich, "Related-key cryptanalysis of the full aes-192 and aes-256," in *Advances in Cryptology—ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings 15*. Springer, 2009, pp. 1–18.

- [41] A. Biryukov, D. Khovratovich, and I. Nikolić, “Distinguisher and related-key attack on the full AES-256,” in *Annual International Cryptology Conference*. Springer, 2009, pp. 231–249.
- [42] M. Bellare and T. Kohno, “A theoretical treatment of related-key attacks: Rka-prps, rka-prfs, and applications,” in *Int. Conf. on the Theory and App. of Crypt. Techniques*. Springer, 2003, pp. 491–506.
- [43] E. Biham, O. Dunkelman, and N. Keller, “Related-key boomerang and rectangle attacks,” in *Annual Int. Conf. on the Theory and App. of Crypt. Techniques*. Springer, 2005, pp. 507–525.
- [44] M. Roetteler and R. Steinwandt, “A note on quantum related-key attacks,” *Inf. Proc. Letters*, vol. 115, no. 1, pp. 40–44, 2015.