





Key Encapsulation Mechanism with Tight Enhanced Security in the Multi-User Setting: Impossibility Result and Optimal Tightness

Shuai Han^{1,2} , Shengli Liu^{1,2,3}  , and Dawu Gu¹ 

¹ School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University, Shanghai 200240, China
{dalen17, slliu, dwgu}@sjtu.edu.cn

² State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China

³ Westone Cryptologic Research Center, Beijing 100070, China

Abstract. For Key Encapsulation Mechanism (KEM) deployed in a multi-user setting, an adversary may corrupt some users to learn their secret keys, and obtain some encapsulated keys due to careless key managements of users. To resist such attacks, we formalize Enhanced security against Chosen Plaintext/Ciphertext Attack (ECPA/ECCA), which ask the pseudorandomness of unrevealed encapsulated keys under uncorrupted users. This enhanced security for KEM serves well for the security of a class of Authenticated Key Exchange protocols built from KEM.

In this paper, we study the achievability of tight ECPA and ECCA security for KEM in the multi-user setting, and present an impossibility result and an optimal security loss factor that can be obtained. The existing meta-reduction technique due to Bader et al. (EUROCRYPT 2016) rules out some KEMs, but many well-known KEMs, e.g., Cramer-Shoup KEM (SIAM J. Comput. 2003), Kurosawa-Desmedt KEM (CRYPTO 2004), run out. To solve this problem, we develop a new technique tool named rank of KEM and a new secret key partitioning strategy for meta-reduction. With this new tool and new strategy, we prove that KEM schemes with polynomially-bounded ranks have no tight ECPA and ECCA security from non-interactive complexity assumptions, and the security loss is at least linear in the number n of users. This impossibility result covers lots of well-known KEMs, including the Cramer-Shoup KEM, Kurosawa-Desmedt KEM and many others. Moreover, we show that the linear security loss is optimal by presenting concrete KEMs with security loss $\Theta(n)$. This is justified by a non-trivial security reduction with linear loss factor from ECPA/ECCA security to the traditional multi-challenge CPA/CCA security.

1 Introduction

The security of a cryptographic primitive is generally formalized by setting up a reasonable security model and defining a proper security notion. The security model formalizes resources obtained by an adversary \mathcal{A} and also the attacks implemented by \mathcal{A} in the real-life settings. For a primitive Π (or a computational problem P), its security model is described by an experiment (game) Exp_{Π} in which the environment (challenger) and an adversary interact with each other. The environment (challenger) in Exp_{Π} provides the resources with which \mathcal{A} implements attacks, then environment detects whether \mathcal{A} wins. Here \mathcal{A} wins in Exp_{Π} means that the aim of its attacks is achieved. Without any resources for help, there also exists a threshold winning probability $\text{thre}_{\text{Exp}_{\Pi}}$ for any adversary. Then \mathcal{A} 's attacking advantage is given by $\epsilon_{\mathcal{A}} := |\Pr[\mathcal{A} \text{ wins}] - \text{thre}_{\text{Exp}_{\Pi}}|$. We call $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}})$ - \mathcal{A} successfully attacks Π if \mathcal{A} 's running time is $t_{\mathcal{A}}$ and advantage is $\epsilon_{\mathcal{A}}$. Parameters $t_{\mathcal{A}}$ and $\epsilon_{\mathcal{A}}$ are measured by security parameter λ . If for all probabilistic polynomial-time (PPT) adversaries, their advantages are all negligible in the security parameter λ , then Π is (asymptotically) secure.

Security Reduction and Tightness. The security proof of primitive Π generally proceeds with a reduction algorithm \mathcal{R} , which transforms a $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}})$ -adversary \mathcal{A} against Π to an algorithm $(t_{\mathcal{R}}, \epsilon_{\mathcal{R}})$ - $\mathcal{R}^{\mathcal{A}}$ against a computational problem P (or another cryptographic primitive). Generally, we only consider simple black-box reduction [2], where \mathcal{R} has oracle access to \mathcal{A} by choosing the inputs for \mathcal{A} , running the adversary sequentially and observing its outputs (but not the codes or internal states of \mathcal{A}). Note that the working effort for \mathcal{A} to win the security game is measured by

the work factor $t_{\mathcal{A}}/\epsilon_{\mathcal{A}}$ [3], which captures the expected running time of \mathcal{A} to break the security. The quotient of \mathcal{R} 's working factor and \mathcal{A} 's working factor is defined as the security loss factor $\ell_{\mathcal{R}}$, i.e., $\ell_{\mathcal{R}} := \frac{\epsilon_{\mathcal{A}}}{\epsilon_{\mathcal{R}}} \cdot \frac{t_{\mathcal{R}}}{t_{\mathcal{A}}}$. If $\ell_{\mathcal{R}}$ is a polynomial in the security parameter λ , then the security of Π is successfully reduced to the hardness of P . This implies that Π is secure as long as P is computationally hard. On the other hand, a large loss factor $\ell_{\mathcal{R}}$ will lead to a gap between Π 's security level and P 's hardness. To fill the gap, Π has to choose a large security parameter to make P hard enough, and this might make Π less efficient. Therefore, the smaller the security loss $\ell_{\mathcal{R}}$ is, the better the security reduction. If $\ell_{\mathcal{R}}$ is a constant, the reduction is called a *tight* one. If $\ell_{\mathcal{R}}$ is an a priori fixed polynomial in λ , then the reduction is called *almost tight* (or *linear-preserving* according to [32]). Such reductions (tight or almost tight) are desirable, since the loss factor is independent of adversarial behavior.

In cryptography, most of long-standing hard problem assumptions are non-interactive ones including the Decision Diffie-Hellman (DDH) assumption, the Factoring assumption, the Learning with Error (LWE) assumption, the existence of one-way function assumption, etc. If the security of a primitive can be (almost) tightly reduced to a Non-Interactive Complexity Assumption (NICA) [2], we call the primitive has (almost) tight security.

(Almost) tight security proofs for cryptographic schemes are always preferable. But the first question to be answered is whether (almost) tight reductions exist for the schemes.

KEM and Its Traditional Security. Key Encapsulation Mechanism (KEM) $\text{KEM} = (\text{Gen}, \text{Encap}, \text{Decap})$ is an important public-key primitive. Its key generation algorithm Gen is able to generate a pair of public key pk and secret key sk . With the public key pk , the encapsulation algorithm Encap can output an encapsulation c and an encapsulated key K . With the secret key sk , the decapsulation algorithm Decap can recover the encapsulated key K from c . KEM found wide applications in theoretic community and real world. For example, public-key encryption (PKE) schemes can always be constructed in a KEM + DEM (Data Encapsulation Mechanism) style [10], which include the well-known ElGamal scheme [12], Cramer-Shoup (CS) scheme [8, 9, 10], Kurosawa-Desmedt (KD) scheme [27], etc. Meanwhile, KEM usually serves as an essential building block in key exchange protocols. For example, the Diffie-Hellman key exchange protocol [11] can be regarded as exchanging $pk = g^a$, $c = g^b$ and establishing $K = g^{ab}$, with $(pk = g^a, sk = a) \leftarrow \text{Gen}$ and $(c = g^b, K = g^{ab}) \leftarrow \text{Encap}(pk)$. In [1, 30, 23, 17], the authenticated key exchange (AKE) protocols are built from KEM and signature schemes. Due to the importance of KEM, NIST included KEM in their calling list for standards of post-quantum algorithms.

Traditionally, the security of KEM is defined in a single-user setting. In the single-user and multi-challenge Chosen-Plaintext Attack (mCPA) security model, the adversary sees the public key pk of KEM, and the environment invokes Encap multiple times to obtain $(c_i, K_i) \leftarrow \text{Encap}(pk)$. The mCPA security of KEM asks the pseudorandomness of $\{K_i\}$ given pk and $\{c_i\}$ to adversaries. Multi-challenge Chosen-Ciphertext Attack (mCCA) security of KEM can be similarly defined, except that the adversary additionally has access to a decapsulation oracle which provides decapsulation services for any c other than $\{c_i\}$.

Multi-User Security for KEM. We note that the traditional mCPA (mCCA) security notion in the single user setting does not cover the attacks in our real-life deployment of KEM. In the era of Internet, cryptographic schemes should presume to be deployed in multi-user systems. Moreover, in reality, we can never rule out the possibility that the secret keys of some users are stolen by hackers, or leaked to adversaries due to careless key management.

As for KEM, the practical attacks, including *corruption of users' secret keys* and *revealing of some encapsulated keys*, have to be considered in the security model. Concretely, in a system of n users with $(pk_i, sk_i) \leftarrow \text{Gen}$, $1 \leq i \leq n$, the adversary may corrupt a subset I of users of its choice and obtain their secret keys $\{sk_i\}_{i \in I}$. For any uncorrupted user $j \notin I$, the adversary is able to see all the encapsulations $\{c_{j,t}\}_{1 \leq t \leq Q}$ under pk_j from a public channel, where $(c_{j,t}, K_{j,t}) \leftarrow \text{Encap}(pk_j)$, $1 \leq t \leq Q$. The adversary may reveal some encapsulated keys $\{K_{j,r}\}_{r \in R}$ for a subset $R \subseteq \{1, \dots, Q\}$. The security of KEM asks pseudorandomness of the unrevealed keys $\{K_{j,t}\}_{j \notin I, t \notin R}$.

Such a security notion also meets the security requirement for KEM used as a building block in many applications. For example, in the KEM+DEM framework for constructing PKE [10], if an adversary sees a pair of plaintext & ciphertext of PKE, then the involved encapsulated key of KEM might be uniquely determined by the adversary [12] or partially leaked to the adversary [10, 27]. Another example is AKE schemes built from KEM, where KEM's public/secret keys serve as (part of) AKE's long-term public/secret keys and KEM's encapsulated keys are used to

derive AKE’s session keys [6, 23]. The security model of AKE (like the CK [5], CK+ [26], eCK [28] models) allows corruption of long-term secret keys and revealing of session keys, which in turn requires the underlying KEM supporting corruptions and key reveals.

In conclusion, the *proper* security for KEM in the multi-user setting should allow adversaries to implement corruptions & key reveals and ask the pseudorandomness of the unrevealed encapsulated keys under public keys of uncorrupted users. We name such notion *Enhanced* security, including Enhanced CPA (ECPA) security and Enhanced CCA (ECCA) security. Obviously, ECPA (resp., ECCA) security is stronger and more desirable than the traditional mCPA (resp., mCCA) security. There are two natural questions to be answered:

- (1) Do the well-known KEM schemes have tight ECPA security (or ECCA security)? For example, the ElGamal-KEM [12], CS-KEM [8, 9, 10] and KD-KEM [27] are among the most efficient KEMs. The GHKW-KEM [14] and HLLG-KEM [18] are core building blocks in achieving (almost) tightly mCCA security for PKE. The Naor-Yung paradigm [34] is a generic approach to CCA-secure PKE, which further results in Naor-Yung type CCA-secure KEM (NY-KEM) by encrypting a uniformly random encapsulated key. We would like to investigate whether they achieve tight ECPA (or ECCA) security.
- (2) How to identify those KEMs incapable of achieving tight ECPA or tight ECCA security?

Impossibility Results on Security Tightness of KEM. It is highly desirable to identify those KEMs for which it is impossible to reduce the ECPA (ECCA) security to any non-interactive complexity assumption (NICA) tightly.

In [2], Bader et al. made use of meta-reduction [7] to prove impossibility of tight security for some KEMs, namely, (almost) tight security reduction from multi-user mCPA (mCCA) security to NICA does not exist for a class of KEM schemes. These KEM schemes are characterized by the properties of “secret key checkability” and “secret key uniqueness/re-randomizability”. “Secret key checkability” means that there exists an efficient algorithm checking whether (pk, sk) is a valid public/secret key pair output by Gen; “Secret key uniqueness” means that there is at most one valid secret key sk for each pk ; “secret key re-randomizability” means that given a valid pair (pk, sk) , there exists an efficient algorithm randomly choosing another secret key sk' from the set of secret keys that validly match pk .

Clearly, ECPA (resp., ECCA) security tightly implies multi-user mCPA (resp., mCCA) security, so all KEM schemes that satisfy “secret key checkability” and “secret key uniqueness/re-randomizability” do not have (almost) tight enhanced security. Recall that the ElGamal KEM [12] has public key $pk = g^a$ and secret key a , which obviously satisfies “secret key checkability” and “secret key uniqueness”. This rules out the (almost) tight ECPA security of ElGamal KEM.

However, lots of other KEM schemes, including the CS-KEM [8, 9, 10] and KD-KEM [27], have neither “secret key uniqueness” nor “secret key re-randomizability”. For example, for the CCA-secure CS-KEM [8, 9, 10], its public key pk contains $h = g_1^{z_1} g_2^{z_2}$ where $(z_1, z_2) \in (\mathbb{Z}_p)^2$ is part of secret key sk . For a fixed $pk = (h, \dots)$, the set of valid secret keys is $\{(z'_1, z'_2, \dots) \mid h = g_1^{z'_1} g_2^{z'_2}\}$. There are at least p valid secret keys, hence “secret key uniqueness” does not hold. Any two secret keys with distinct (z_1, z_2) and (z'_1, z'_2) can determine the value of $\log_{g_1} g_2 = (z_1 - z'_1)(z'_2 - z_2)^{-1} \bmod p$, hence solving the discrete logarithm problem. So the CS-KEM does not satisfy the property of “secret key re-randomizability”, unless the discrete logarithm problem is easy to solve. Therefore, determining whether tightness impossibility holds for such KEM schemes needs new techniques.

Our Contribution. We work on impossibility of tight reduction on KEM, and show that for certain KEM schemes, there exists no (almost) tight security reduction from the ECPA (ECCA) security to non-interactive complexity assumptions (NICA). We also present the optimal tightness bound of security loss factor and identify those KEM schemes that can achieve the optimal tightness bound. Our contribution is detailed as follows.

- We develop a useful tool named *rank of KEM* to identify a class of KEM schemes for which impossibility of (almost) tight reduction holds. More precisely, we proved that as long as the rank of a KEM scheme is *polynomially bounded* (in the security parameter λ), the incurred security loss factor of KEM is $\Omega(n)$ when the enhanced security of KEM is reduced to any NICA. Here n denotes the number of users.
- We compute the ranks or provide upper bounds of ranks for the well-known KEM schemes including the ElGamal-KEM [12], CS-KEM [8, 9, 10], KD-KEM [27], GHKW-KEM [14], HLLG-

KEM [18], and many instantiations of NY-KEM [34]. Their polynomially-bounded ranks indicate that these KEMs suffer from a security loss factor $\Omega(n)$.

- On the other hand, we proved that any tightly mCPA (resp., mCCA) secure KEM is able to achieve ECPA (resp., ECCA) security with loss factor $O(n)$. As a result, the ElGamal-KEM [12], CS-KEM [8, 9, 10] and KD-KEM [27] all have ECPA security with security loss factor $\Theta(n)$ when reduced to the DDH assumption. Similarly, the HLLG-KEM [18] has ECCA security with security loss factor $\Theta(n)$ based on the matrix DDH (MDDH) assumption [13] (which corresponds to the standard DDH, k -Linear assumptions under different parameters). This suggests that the optimal security loss factor for ECPA (ECCA) is $\Theta(n)$ and achievable.

We highlight that our impossibility result is the first that does not impose any requirement on the (pk, sk) relation (like checkability [2, 19], uniqueness, or re-randomizability [7, 25, 21, 2, 31]), nor limited to deterministic primitives [32].

1.1 Technique Overview

The Meta-Reduction Paradigm. Our impossibility result about KEM is built upon a line of research on using “meta-reductions” [4, 7, 25, 21, 2, 35, 32]. To the best of our knowledge, up to now all known black-box separations using the meta-reduction paradigm only apply to primitives that either embody some form of uniqueness or re-randomizability [7, 2] or are deterministic ones like pseudorandom function (PRF) or message authentication code (MAC) with deterministic tagging [32].

The high-level idea of the meta-reduction paradigm for a primitive works as follows. Let \mathcal{R} be any reduction algorithm from the security of the primitive to any NICA. Firstly, we construct a hypothetical (inefficient) adversary \mathcal{A}^* that breaks the security of the primitive with advantage $\epsilon_{\mathcal{A}^*} \geq 1 - \alpha$. Here α means the failure probability of \mathcal{A}^* . Let $\epsilon_{\mathcal{R}\mathcal{A}^*}$ be the advantage with which $\mathcal{R}\mathcal{A}^*$ breaks the NICA via black-box access to \mathcal{A}^* . Secondly, we construct an efficient meta-reduction algorithm \mathcal{B} , which “emulates” \mathcal{A}^* while running \mathcal{R} . Suppose that \mathcal{B} emulates $\mathcal{R}\mathcal{A}^*$ perfectly except with probability at most δ , then \mathcal{B} ’s advantage $\epsilon_{\mathcal{B}}$ against NICA satisfies $|\epsilon_{\mathcal{B}} - \epsilon_{\mathcal{R}\mathcal{A}^*}| \leq \delta$. Obviously, the running time $t_{\mathcal{R}\mathcal{A}^*}$ is lower-bounded by $t_{\mathcal{A}^*}$. Consequently, the loss factor of reduction \mathcal{R} is

$$\ell_{\mathcal{R}} \geq \frac{\epsilon_{\mathcal{A}^*}}{\epsilon_{\mathcal{R}\mathcal{A}^*}} \cdot \frac{t_{\mathcal{R}\mathcal{A}^*}}{t_{\mathcal{A}^*}} \geq \frac{1 - \alpha}{\epsilon_{\mathcal{B}} + \delta}.$$

By the NICA assumption, $\epsilon_{\mathcal{B}}$ is negligibly small for any efficient \mathcal{B} . So

$$\ell_{\mathcal{R}} = \Omega\left(\frac{1 - \alpha}{\delta}\right), \tag{1}$$

which suggests that the failure probability α of \mathcal{A}^* and the failure probability δ of \mathcal{B} are the key factors for the lower bound of loss factor $\ell_{\mathcal{R}}$.

Let us take [2] and [32] as examples, both of which rule out (almost) tight (i.e., linear-preserving) reductions for the multi-user security of some primitives. Let n denote the number of users.

- In [32], $\alpha = 1/\text{poly}(\lambda)$ for some polynomial poly . If MAC is deterministic, then $\delta = 1/\sqrt{n}$.¹ Therefore, the security reduction for such MACs loses a factor of $\Omega(\sqrt{n})$.

Note that the construction of \mathcal{A}^* and meta-reduction \mathcal{B} in [32] are tailored for deterministic primitives like PRF, deterministic MAC, deterministic signature, etc.

- In [2], $\alpha = 0$. If KEM satisfies the properties of “secret key checkability” and “secret key uniqueness/re-randomizability”, then $\delta = 1/n$. Therefore, the security reduction for such KEMs loses a factor of $\Omega(n)$.

Note that in [2] the secret keys are partitioned according to an efficient algorithm $\text{SKCheck}(\cdot, \cdot)$ which checks the relation between pk and sk . For each pk , let \mathcal{SK}_{pk} be the set of all secret keys corresponding to pk , i.e., $\mathcal{SK}_{pk} := \{sk' \mid \text{SKCheck}(pk, sk') = 1\}$. “Secret key uniqueness” means that there is unique sk in \mathcal{SK}_{pk} . “Secret key re-randomizability” requires that there is another efficient algorithm for sampling sk' from \mathcal{SK}_{pk} uniformly at random, given a pair of (pk, sk) . The construction of \mathcal{A}^* and \mathcal{B} in [2] works so that $\delta = 1/n$ when (pk, sk) relation has

¹ Their results apply to more general reductions supporting rewinding and concurrency, based on bounded-round interactive complexity assumptions.

checkability and uniqueness/re-randomizability. Such a condition is satisfied by the ElGamal-KEM [12], but lots of other KEMs run out, including the CS-KEM [8, 9, 10], KD-KEM [27], GHKW-KEM [14], HLLG-KEM [18], etc. Therefore, we have to resort to new techniques to identify whether these KEMs have (almost) tight ECPA (ECCA) security or not.

New Partitioning Technique: Decapsulation Equivalence of Secret Keys. In this paper, we take full advantage of the resources provided to adversary in the ECPA (ECCA) game, and provide a novel technique of partitioning secret keys. We do not impose any requirement on the (pk, sk) relation (like checkability, uniqueness, or re-randomizability), and the secret keys are no longer partitioned according to public keys like [2] does. Our new strategy is partitioning secret keys according to their functionality when they are used to decapsulate a set of ciphertexts \mathcal{X} .

We define a decapsulation equivalence relation on the secret key space \mathcal{SK} w.r.t. a subset of ciphertexts $\mathcal{X} \subseteq \mathcal{CT}$. Any two secret keys $sk, sk' \in \mathcal{SK}$ are decapsulation-equivalent w.r.t. \mathcal{X} if they result in the same decapsulated key for each ciphertext in \mathcal{X} . In formula,

$$sk \sim_{\mathcal{X}} sk' \iff \forall c \in \mathcal{X}, \text{Decap}(sk, c) = \text{Decap}(sk', c).$$

Then the equivalence relation parameterized by ciphertext set \mathcal{X} is

$$\text{EquivSK}(\mathcal{X}) := \{(sk, sk') \in \mathcal{SK}^2 \mid \forall c \in \mathcal{X}, \text{Decap}(sk, c) = \text{Decap}(sk', c)\}.$$

Our Meta-Reduction. With the new partitioning of secret keys, we are able to present our new meta reduction. Here we give a high-level overview of the hypothetical adversary \mathcal{A}^* and meta-reduction algorithm \mathcal{B} in our meta-reduction. Define $[n] := \{1, 2, \dots, n\}$ and $[n \setminus i] := \{1, 2, \dots, n\} \setminus \{i\}$. \mathcal{A}^* works as follows.

Hypothetical \mathcal{A}^*

- STEP 1 (SETUP): \mathcal{A}^* receives public keys $\{pk_1, \dots, pk_n\}$ of n users, which are generated by $(pk_i, sk_i) \leftarrow \text{Gen}$.
- STEP 2 (ENCAPSULATION): \mathcal{A}^* issues Q encapsulation queries per user, and obtains nQ encapsulations $\{c_{i,j}\}_{i \in [n], j \in [Q]}$, where $(c_{i,j}, K_{i,j}) \leftarrow \text{Encap}(pk_i)$.
- STEP 3 (KEY REVEAL): \mathcal{A}^* reveals $Q - 1$ encapsulated keys per user randomly, and obtains $n(Q - 1)$ encapsulated keys $\{K_{i,j}\}_{i \in [n], j \in [Q \setminus j_i]}$, where $j_1, j_2, \dots, j_n \leftarrow_s [Q]$ are the indices of unrevealed keys.
- STEP 4 (CORRUPTION & CHECK): \mathcal{A}^* corrupts all users except one, and obtains $n - 1$ secret keys $\{sk_i\}_{i \in [n \setminus i^]}$, where $i^* \leftarrow_s [n]$ is the index of the uncorrupted user.

Then \mathcal{A}^* checks whether the decapsulation relation $\text{Decap}(sk_i, c_{i,j}) = K_{i,j}$ holds for each $i \in [n \setminus i^*]$ and $j \in [Q \setminus j_i]$, and aborts if the check fails.

- STEP 5 (CHALLENGE & OUTPUT): \mathcal{A}^* obtains a challenge K^* w.r.t. $c_{i^*, j_{i^*}}$, which is either the real key $K_{i^*, j_{i^*}}$ encapsulated in $c_{i^*, j_{i^*}}$ or a random key.

By brute-force search, \mathcal{A}^* picks a random sk^* from the set

$$\{sk \in \mathcal{SK} \mid \text{Decap}(sk, c_{i^*, j}) = K_{i^*, j}, \forall j \in [Q \setminus j_{i^*}]\}. \quad (2)$$

Finally, \mathcal{A}^* outputs 1 if and only if $K^* = \text{Decap}(sk^*, c_{i^*, j_{i^*}})$ holds.

Note that by the perfect correctness of KEM, the real secret key sk_{i^*} of user i^* also belongs to the above set shown in (2), so the real secret key sk_{i^*} and the sk^* chosen by \mathcal{A}^* have the same decapsulation functionality when they are used to decapsulate the set of ciphertexts $\{c_{i^*, j}\}_{j \in [Q \setminus j_{i^*}]}$. Consequently, by the equivalence relation we defined, we have

$$(sk^*, sk_{i^*}) \in \text{EquivSK}(\{c_{i^*, 1}, \dots, c_{i^*, Q}\} \setminus \{c_{i^*, j_{i^*}}\}).$$

Observe that \mathcal{A}^* will have advantage 1 if $\text{EquivSK}(\{c_{i^*, 1}, \dots, c_{i^*, Q}\} \setminus \{c_{i^*, j_{i^*}}\}) \subseteq \text{EquivSK}(\{c_{i^*, j_{i^*}}\})$, i.e., all the secret keys that have the same decapsulation results on the $Q - 1$ ciphertexts also result

in the same decapsulation result on one more ciphertext $c_{i^*, j_{i^*}}$. Define

$$\alpha := \max_{c_1, c_2, \dots, c_Q} \left(\Pr_{j \leftarrow [Q]} [\text{EquivSK}(\{c_1, \dots, c_Q\} \setminus \{c_j\}) \not\subseteq \text{EquivSK}(\{c_j\})] \right). \quad (3)$$

Then \mathcal{A}^* has advantage $\epsilon_{\mathcal{A}^*} \geq 1 - \alpha$.

Now we construct a meta-reduction \mathcal{B} that emulates \mathcal{A}^* efficiently while running \mathcal{R} as the challenger. Note that all steps of \mathcal{A}^* are efficient except step 5. So \mathcal{B} can emulate steps 1-4 of \mathcal{A}^* honestly. Then \mathcal{B} adds a rewinding step 4.5 which rewinds the corruption procedure $n - 1$ times. With the help of information obtained from the rewindings, \mathcal{B} derives a secret key to emulate \mathcal{A}^* with an efficient step 5', which is different from the step 5 of \mathcal{A}^* . A high-level overview of \mathcal{B} is as follows.

Meta-Reduction \mathcal{B}

- STEPS 1-4: \mathcal{B} runs \mathcal{R} as the challenger and emulates \mathcal{A}^* honestly. Suppose that in step 4, the index of the uncorrupted user is i^* .
- STEP 4.5 (REWINDING): \mathcal{B} rewinds the corruption procedure $n - 1$ times. In the ι -th rewind ($\iota \in [n \setminus i^*]$), \mathcal{B} corrupts all users except user ι and obtains the corrupted secret keys $\{sk_i^{(\iota)}\}_{i \in [n \setminus \iota]}$ from \mathcal{R} , where $sk_i^{(\iota)}$ denotes the corrupted secret key of user i obtained in the ι -th rewind.
- STEP 5' (CHALLENGE & OUTPUT): \mathcal{B} runs \mathcal{R} to obtain the challenge K^* , but has a different strategy for the output bit.

More precisely, \mathcal{B} checks whether it ever obtained a corrupted $sk_{i^*}^{(\iota)}$ of user i^* in one of the $n - 1$ rewindings, such that

$$\text{Decap}(sk_{i^*}^{(\iota)}, c_{i^*, j}) = K_{i^*, j}, \quad \text{for } \forall j \in [Q \setminus j_{i^*}]. \quad (4)$$

If \mathcal{B} finds such a $sk_{i^*}^{(\iota)}$, then \mathcal{B} uses $sk_{i^*}^{(\iota)}$ to test whether $K^* = \text{Decap}(sk_{i^*}^{(\iota)}, c_{i^*, j_{i^*}})$, and returns 1 to \mathcal{R} if and only if the equation holds.

In step 5', as long as there exists a rewinding in which \mathcal{R} responds with a corrupted secret key $sk_{i^*}^{(\iota)}$ such that (4) holds, then \mathcal{B} will not abort. Since i^* is randomly chosen from $[n]$, by a similar argument as [2, 19], we can bound the the probability that \mathcal{B} aborts by $1/n$. If (4) holds, then the $sk_{i^*}^{(\iota)}$ obtained by \mathcal{B} also belongs to the set defined in (2), from which \mathcal{A} chooses its sk^* , thus

$$(sk_{i^*}^{(\iota)}, sk^*) \in \text{EquivSK}(\{c_{i^*, 1}, \dots, c_{i^*, Q}\} \setminus \{c_{i^*, j_{i^*}}\}).$$

In this case, \mathcal{B} will perfectly emulate \mathcal{A}^* as long as $\text{EquivSK}(\{c_{i^*, 1}, \dots, c_{i^*, Q}\} \setminus \{c_{i^*, j_{i^*}}\}) \subseteq \text{EquivSK}(\{c_{i^*, j_{i^*}}\})$, which happens with probability at least $1 - \alpha$ according to the definition of α in (3). Taking into account the probability that \mathcal{B} aborts, we know that \mathcal{B} perfectly emulates \mathcal{A}^* for \mathcal{R} except with probability at most $\alpha + 1/n$. Therefore,

$$|\epsilon_{\mathcal{B}} - \epsilon_{\mathcal{R}\mathcal{A}^*}| \leq \delta = \alpha + 1/n. \quad (5)$$

By plugging (5) into (1), we obtain a lower bound of the security loss factor in our meta-reduction:

$$\ell_R = \Omega\left(\frac{1-\alpha}{\delta}\right) = \Omega\left(\frac{1-\alpha}{\alpha+1/n}\right),$$

where α is defined in (3).

Observe that as long as $\alpha = O(1/n)$, the loss factor is $\ell_R = \Omega(n)$, at least linear in the number n of users. Next, we identify a class of KEMs with $\alpha = O(1/n)$ with a new technique tool called *rank of KEM*.

New Technique Tool: Rank of KEM. We define the *rank* of a KEM scheme KEM, denoted by Rank_{KEM} , as the cardinality of the largest *independent* subset $\mathcal{X}' \subseteq \mathcal{CT}$ such that $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{CT})$. Here we explain the intuitions behind this new notion and the meaning of independent set.

- $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{CT})$ indicates that, all the secret keys that have the same decapsulation functionality on \mathcal{X}' also have the same decapsulation functionality on the whole ciphertext space \mathcal{CT} . Intuitively, this means that \mathcal{X}' “determines” the decapsulation functionality of secret keys on the whole ciphertext space \mathcal{CT} .
- We require \mathcal{X}' to be an *independent* set in the sense that every ciphertext c in \mathcal{X}' contributes to $\text{EquivSK}(\mathcal{X}')$, i.e., $\text{EquivSK}(\mathcal{X}' \setminus \{c\}) \neq \text{EquivSK}(\mathcal{X}')$.

Intuitively, the relation between \mathcal{X}' and \mathcal{CT} is analogous to the relation between a basis and a linear space, and the rank of KEM is analogous to the size of (the largest) basis (i.e., the dimension of linear space).

However, we note that in general the decapsulation algorithm `Decap` of KEM is not a linear function, especially for CCA-secure KEMs. So the rank of KEM is *different from* the dimension of \mathcal{CT} even if \mathcal{CT} is indeed a linear space. Moreover, we highlight that our notion of rank for KEM is more general and purely defined based on the equivalence relation “`EquivSK`” on secret keys, and we in fact do not require any algebraic structure from \mathcal{CT} .

Bounding the Security Loss with KEM’s Rank. The notion of rank for KEM is a useful tool for analyzing the failure probability α defined in (3) in our meta-reduction. Let us name a ciphertext c a *bad* one in \mathcal{X} if $\text{EquivSK}(\mathcal{X} \setminus \{c\}) \not\subseteq \text{EquivSK}(\{c\})$. We prove an important core lemma (see Lemma 3 in Subsect. 4.3). The core lemma shows that the number of bad ciphertexts in any ciphertext subset \mathcal{X} is upper bounded by Rank_{KEM} . As a result, we have

$$\Pr_{c \leftarrow \mathcal{X}} [\text{EquivSK}(\mathcal{X} \setminus \{c\}) \not\subseteq \text{EquivSK}(\{c\})] \leq \frac{\text{Rank}_{\text{KEM}}}{\#\mathcal{X}}. \quad (6)$$

Combining (6) and (3), we have

$$\alpha \leq \frac{\text{Rank}_{\text{KEM}}}{Q}.$$

Note that Q is the number of encapsulation queries made by \mathcal{A}^* for each user. As long as Rank_{KEM} is bounded by an a priori fixed polynomial (in the security parameter λ), we can always choose Q such that $\alpha \leq \text{Rank}_{\text{KEM}}/Q < 1/n$. Then the security loss factor is $\ell_{\mathcal{R}} = \Omega(\frac{1-\alpha}{\delta}) = \Omega(\frac{1-\alpha}{\alpha+1/n}) = \Omega(n)$.

Consequently, with our new technique tool, rank of KEM, we identify a class of KEMs for which impossibility of (almost) tight reduction holds. Namely, any KEM with polynomially-bounded rank has no (almost) tight (i.e., linear-preserving) reduction from its ECPA (ECCA) security to any NICA.

A careful computation of ranks for many well-known KEM schemes (including the ElGamal-KEM [12], CS-KEM [8, 9, 10], KD-KEM [27], GHKW-KEM [14], HLLG-KEM [18] and many instantiations of NY-KEM [34]) shows that our impossibility result applies to these KEM schemes. See Subsect. 5.2 and Appendix A for more details.

1.2 Application of Our Impossibility Result in AKE

Authenticated Key Exchange (AKE) is one of the most widely deployed protocols on Internet and it allows two parties to establish a session key over public channels. Most of AKE constructions make use of KEM explicitly or implicitly, for instance, the well-known Signed Diffie-Hellman Protocol, modular AKE constructions in [1, 37, 30, 23, 17]. Therefore, the security of AKE is closely related to the security of KEM. Defining a proper security for KEM can directly serve the security proof of AKE.

The well-known security notions of AKE are defined with the CK model [5], eCK model [28], or CK+ model [26], all of which consider both passive attacks and active attacks in the multi-user setting. Passive attacks allow the adversary to see the messages over public channel, while active attacks not only allow the adversary to modify, drop, replay, or inject messages on the public channel, but also allow the adversary to corrupt user’s long-term secret key in AKE and reveal session keys of some AKE protocol instances. The security of AKE requires the pseudorandomness of session keys between two users, if the session keys are not revealed and the two users’ long-term secret keys are not corrupted.

Let us consider the case that the public and secret keys $(pk_{\text{KEM}}, sk_{\text{KEM}})$ of KEM serve as part of a user’s long-term public key $pk_{\text{AKE}} = (pk_{\text{KEM}}, \dots)$ and secret key $sk_{\text{AKE}} = (sk_{\text{KEM}}, \dots)$ of AKE. Furthermore, the session key of AKE is derived from the encapsulated key of KEM. As a result, the

corruption of sk_{AKE} requires the security of the underlying KEM to support corruption of sk_{KEM} , and the reveal of session keys in AKE asks the underlying KEM to support reveal of encapsulated keys. Therefore, the ECPA (ECCA) security of KEM is exactly the right security notion needed by AKE in this case. Combined with our impossibility result, any KEM with polynomially-bounded rank cannot be tightly secure, hence such construction of AKE cannot be tightly secure as well.

Therefore, we have the following rules for constructing tightly secure AKE: either (i) the secret key of KEM does not appear in the long-term secret key of AKE, like [1, 30, 17]; or (ii) the tight security proof of AKE relies on the Random Oracle model, like [16, 23, 36]; or (iii) AKE avoids the usage of KEM with polynomially-bounded rank. Up to now, most of the well-known efficient KEM schemes with tight mCPA-security in the multi-user setting have polynomially-bounded rank. Hence rule (iii) eliminates the possibility of constructing tightly AKE with aforementioned KEMs if KEM’s secret keys are used as AKE’s long-term keys.

1.3 Related Works

Meta-reduction paradigm was proposed in [4] and used to show black-box impossibility results. Later, Coron [7] made use of meta-reductions to prove the impossibility of tight reductions for certain digital signature schemes and showed the lower bounds on security loss. This technique was further extended in [25, 2, 31].

Hofheinz et al. [21] showed that any black-box security proof for a signature scheme with re-randomizable signatures must have a reduction loss of at least Q , the number of signature queries from the adversary.

Lewko and Waters [29] used the technique in [21] to identify certain conditions for hierarchical identity-based encryption (HIBE) under which HIBE has an exponential loss.

Bader et al. [2] developed a new meta-reduction technique to obtain a bundle of impossibility results. Their results rule out tight reductions from non-interactive complexity assumptions (NICA) for certain class of public-key encryption (PKE), KEM and digital signatures with multi-user security allowing secret key corruptions. This class of public-key primitives is characterized by secret key’s checkable relation with public key and property of secret key uniqueness or re-randomizability.

Jager et al. [24] considered symmetric encryption schemes in multi-user setting in which adversaries can adaptively corrupt encryption keys. They ruled out linear-preserving black-box reductions from adaptive multi-user security to single-user security for any authenticated encryption scheme with a strong “key uniqueness” property.

Very recently, Morgan et al. [32] studied black-box reductions to “standard” assumptions for message authentication code (MAC). Their black-box reduction is a general one which allows reduction algorithm to concurrently run or rewind adversary, and the complexity assumption is extended from NICA to any interactive assumption with pre-defined bounded number of interactions. They showed that linear-preserving security reduction does not exist for adaptive multi-user secure deterministic stateless MACs. Their results also hold for PRFs and deterministic stateless signatures. However, the meta-reduction paradigm in [32] only applies to deterministic primitives.

2 Preliminaries

2.1 Notations

Let $\lambda \in \mathbb{N}$ denote the security parameter throughout the paper. Let \emptyset denote the empty set. If x is defined by y or the value of y is assigned to x , we write $x := y$. For $n \in \mathbb{N}$, define $[n] := \{1, 2, \dots, n\}$, and for $i \in [n]$, define $[n \setminus i] := [n] \setminus \{i\}$. For a set $\{x_1, \dots, x_n\}$ and $i \in [n]$, define $\{x_1, \dots, x_n \setminus x_i\} := \{x_1, \dots, x_n\} \setminus \{x_i\}$. For a set \mathcal{X} , denote by $\#\mathcal{X}$ the cardinality of \mathcal{X} . Denote by $x \leftarrow_s \mathcal{X}$ the procedure of sampling x from set \mathcal{X} uniformly at random. If \mathcal{D} is distribution, $x \leftarrow_s \mathcal{D}$ means that x is sampled according to \mathcal{D} . All our algorithms are probabilistic unless stated otherwise. We use $y \leftarrow_s \mathcal{A}(x)$ to define the random variable y obtained by executing algorithm \mathcal{A} on input x . We use $y \in \mathcal{A}(x)$ to indicate that y lies in the support of $\mathcal{A}(x)$. If \mathcal{A} is deterministic we write $y \leftarrow \mathcal{A}(x)$. We also use $y \leftarrow \mathcal{A}(x; r)$ to make explicit the random coins r used in the probabilistic computation. Denote by $t_{\mathcal{A}}$ the running time of \mathcal{A} .

2.2 Key Encapsulation Mechanisms

Definition 1 (KEM). A key encapsulation mechanism (KEM) scheme $\text{KEM} = (\text{Setup}, \text{Gen}, \text{Encap}, \text{Decap})$ consists of four algorithms:

- **Setup:** The setup algorithm outputs public parameters pp , which determine public key & secret key spaces $\mathcal{PK} \times \mathcal{SK}$, an encapsulation key space \mathcal{K} , and a ciphertext space \mathcal{CT} .
- **Gen(pp):** Taking pp as input, the key generation algorithm outputs a pair of public key and secret key $(pk, sk) \in \mathcal{PK} \times \mathcal{SK}$.
- **Encap(pk):** Taking pk as input, the encapsulation algorithm outputs a pair of ciphertext $c \in \mathcal{CT}$ and encapsulated key $K \in \mathcal{K}$.
- **Decap(sk, c):** Taking as input sk and c , the deterministic decapsulation algorithm outputs $K \in \mathcal{K} \cup \{\perp\}$.

Correctness require that for all $pp \in \text{Setup}$, $(pk, sk) \in \text{Gen}(pp)$, $(c, K) \in \text{Encap}(pk)$, it holds that $\text{Decap}(sk, c) = K$.

We recall the traditional IND-CPA/CCA security notions for KEMs in the single-user and multi-challenge setting, denoted by IND-mCPA/IND-mCCA for short.

Definition 2 (IND-mCPA/IND-mCCA Security). We say that an adversary \mathcal{A} ($t_{\mathcal{A}}, \epsilon_{\mathcal{A}}$)-breaks the IND-mCPA (resp., IND-mCCA) security of KEM, if it runs in time $t_{\mathcal{A}}$, and $\text{Adv}_{\text{KEM}}^{\text{ind-mcpa}}(\mathcal{A}) := 2 \cdot |\Pr[\text{Exp}_{\text{KEM}}^{\text{ind-mcpa}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}| \geq \epsilon_{\mathcal{A}}$ (resp., $\text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{A}) := 2 \cdot |\Pr[\text{Exp}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}| \geq \epsilon_{\mathcal{A}}$), where the experiments are defined in Fig. 1.

$\text{Exp}_{\text{KEM}}^{\text{ind-mcpa}}(\mathcal{A}), \text{Exp}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{A}) :$ $pp \leftarrow_{\$} \text{Setup}$ $(pk, sk) \leftarrow_{\$} \text{Gen}(pp)$ $\text{EnclList} := \emptyset$ //Records the encapsulation queries $b \leftarrow_{\$} \{0, 1\}$ //challenge bit $b' \leftarrow_{\$} \mathcal{A}^{\mathcal{O}_{\text{Enc}}(\cdot), \mathcal{O}_{\text{Dec}}(\cdot, \cdot)}(pp, pk)$ If $b' = b$: Return 1; Else: Return 0	$\mathcal{O}_{\text{Enc}}() :$ $(c, K) \leftarrow_{\$} \text{Encap}(pk)$ $\text{EnclList} := \text{EnclList} \cup \{c\}$ $K_0 := K; K_1 \leftarrow_{\$} \mathcal{K}$ Return (c, K_b) $\mathcal{O}_{\text{Dec}}(c') :$ If $c' \notin \text{EnclList}$: Return $K' \leftarrow \text{Decap}(sk, c')$ Else: Return \perp
--	--

Fig. 1. The IND-mCPA security experiment $\text{Exp}_{\text{KEM}}^{\text{ind-mcpa}}(\mathcal{A})$ and the IND-mCCA security experiment $\text{Exp}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{A})$ of KEM, where in the latter the adversary has also access to a decapsulation oracle $\mathcal{O}_{\text{Dec}}(\cdot)$.

2.3 Non-Interactive Assumptions

We recall the definition of non-interactive complexity assumptions (NICA).

Definition 3 (NICA [2]). A non-interactive complexity assumption (NICA) $N = (T, V, U)$ consists of three algorithms. The instance generation algorithm T outputs a problem instance x and a witness w . U is a PPT algorithm, which takes x as input and outputs a candidate solution s . The verification algorithm V takes as input (x, w) and a candidate solution s . If $V(x, w, s) = 1$, then we say that s is a correct solution to the challenge x .

We say that an adversary \mathcal{B} ($t_{\mathcal{B}}, \epsilon_{\mathcal{B}}$)-breaks an NICA $N = (T, V, U)$, if it runs in time $t_{\mathcal{B}}$, and $\text{Adv}_N^{\text{nica}}(\mathcal{B}) := |\Pr[\text{Exp}_N^{\text{nica}}(\mathcal{B}) \Rightarrow 1] - \Pr[\text{Exp}_N^{\text{nica}}(U) \Rightarrow 1]| \geq \epsilon_{\mathcal{B}}$, where the experiment $\text{Exp}_N^{\text{nica}}(Z)$ ($Z \in \{\mathcal{B}, U\}$) runs $(x, w) \leftarrow_{\$} T$, executes $s \leftarrow_{\$} Z(x)$, and outputs $V(x, w, s)$.

Intuitively, U is an algorithm which implements a suitable “trivial” attack strategy for N , and $\Pr[\text{Exp}_N^{\text{nica}}(U) \Rightarrow 1]$ is the winning probability of trivial attacks.

3 Enhanced Security Notions for KEMs

In this section, we introduce Enhanced CPA/CCA security notions for KEM in the Multi-User and Multi-Challenge (MUMC) setting, called *MUMC-ECPA*/*MUMC-ECCA*, which allow user corruptions and encapsulated key reveals.

Definition 4 (MUMC-ECPA/ECCA Security). We say that an adversary $\mathcal{A}(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, n, Q_e, Q_t)$ -breaks the *MUMC-ECPA* (resp., *MUMC-ECCA*) security of KEM, if it runs in time $t_{\mathcal{A}}$, and $\text{Adv}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecpa}}(\mathcal{A}) := 2 \cdot \left| \Pr[\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecpa}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2} \right| \geq \epsilon_{\mathcal{A}}$ (resp., $\text{Adv}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) := 2 \cdot \left| \Pr[\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2} \right| \geq \epsilon_{\mathcal{A}}$), where the experiments are defined in Fig. 2 and the scalar 2 is added so that the advantages are between 0 and 1.

$\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecpa}}(\mathcal{A}), \text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) :$ $pp \leftarrow \text{Setup}$ For $i \in [n]$: $(pk_i, sk_i) \leftarrow \text{Gen}(pp)$ $\text{EnclList} := \emptyset$ //Records the encapsulation queries $\text{RevList} := \emptyset$ //Records the reveal queries $\text{CorrList} := \emptyset$ //Records the corruption queries $\text{TestList} := \emptyset$ //Records the test queries $\beta \leftarrow \{0, 1\}$ //Single challenge bit $\text{PKList} := \{pk_i\}_{i \in [n]}$ $\beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{ENC}}(\cdot), \mathcal{O}_{\text{DEC}}(\cdot, \cdot), \mathcal{O}_{\text{REV}}(\cdot, \cdot), \mathcal{O}_{\text{CORR}}(\cdot), \mathcal{O}_{\text{TEST}}(\cdot, \cdot)}(pp, \text{PKList})$ If $\beta' = \beta$: Return 1; Else: Return 0	$\mathcal{O}_{\text{ENC}}(i):$ //At most Q_e times in total $(c, K) \leftarrow \text{Encap}(pk_i)$ $\text{EnclList} := \text{EnclList} \cup \{(i, c, K)\}$ Return c //Only c is returned $\mathcal{O}_{\text{DEC}}(i, c') :$ If $(i, c', \cdot) \notin \text{EnclList}$: Return $K' \leftarrow \text{Decap}(sk_i, c')$ Else: Return \perp $\mathcal{O}_{\text{REV}}(i, c):$ If $(i, c, K) \in \text{EnclList}$ for some K $\wedge (i, c) \notin \text{TestList}$: $\text{RevList} := \text{RevList} \cup \{(i, c)\}$ Return K Else: Return \perp $\mathcal{O}_{\text{CORR}}(i):$ If $(i, \cdot) \notin \text{TestList}$: $\text{CorrList} := \text{CorrList} \cup \{i\}$ Return sk_i Else: Return \perp
$\mathcal{O}_{\text{TEST}}(i, c):$ //At most Q_t times in total If $(i, c, K) \in \text{EnclList}$ for some $K \wedge (i, c) \notin \text{RevList} \cup \text{TestList}$ $\wedge i \notin \text{CorrList}$: $\text{TestList} := \text{TestList} \cup \{(i, c)\}$ $K_0 := K; K_1 \leftarrow \mathcal{K}$ Return K_β Else: Return \perp	

Fig. 2. The MUMC-ECPA security experiment $\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecpa}}(\mathcal{A})$ and the MUMC-ECCA security experiment $\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A})$ of KEM, where in the latter the adversary has also access to a decapsulation oracle $\mathcal{O}_{\text{DEC}}(\cdot, \cdot)$. In both experiments, \mathcal{A} is allowed to query \mathcal{O}_{ENC} at most Q_e times and query $\mathcal{O}_{\text{TEST}}$ at most Q_t times.

In the MUMC-ECPA and MUMC-ECCA security experiments defined in Fig. 2, the adversary \mathcal{A} is allowed to make several kinds of oracle queries.

- **Encapsulation query.** Through $\mathcal{O}_{\text{ENC}}(i)$ query, \mathcal{A} obtains an encapsulation c under pk_i . We note that the corresponding encapsulated key K is not given out along with c through \mathcal{O}_{ENC} , different from the IND-mCPA/mCCA experiment (cf. Fig. 1). In contrast, the key K encapsulated in c can be later revealed by Key Reveal query or tested by Test query.
- **Key Reveal query.** Upon a Key Reveal query $\mathcal{O}_{\text{REV}}(i, c)$, if c is an output of $\mathcal{O}_{\text{ENC}}(i)$, the key K encapsulated in c is returned to \mathcal{A} .
- **SBG-style Test query.** Upon a Test query $\mathcal{O}_{\text{TEST}}(i, c)$, if c is an output of $\mathcal{O}_{\text{ENC}}(i)$, the real key $K_0 = K$ encapsulated in c or a random key K_1 is returned to \mathcal{A} , depending on the challenge bit β . We note that this is defined in the Single-Bit-Guess (SBG) style [6, 23], which is desirable due to its well composability with symmetric cryptographic primitives like DEM. Such an SBG-style security of KEM also serves well as a building block for the SBG-style security of more sophisticated primitives or protocols like AKE.
- **Decapsulation query.** A decapsulation oracle $\mathcal{O}_{\text{DEC}}(i, c')$ is provided in the MUMC-ECCA security experiment to decapsulate ciphertexts c' that are not returned by $\mathcal{O}_{\text{ENC}}(i)$.
- **Corruption query.** Via $\mathcal{O}_{\text{CORR}}(i)$ query, \mathcal{A} can corrupt a user and obtain its secret key sk_i .

Finally, we stress that some trivial attacks are forbidden. For example, (1) \mathcal{A} is not allowed to both corrupt some user and test encapsulated keys of this user; (2) \mathcal{A} is not allowed to reveal an encapsulated key and test the same key; (3) \mathcal{A} is not allowed to test an encapsulated key twice due to the SBG-style definition we adopt.

The MUMC-ECPA (MUMC-ECCA) security is more reasonable than the mCPA (mCCA) notion (cf. Definition 2), since it captures the practical attacks, like corrupting users' secret keys, revealing users' encapsulated keys, in the multi-user and multi-challenge setting.

We also define the enhanced security notions in the Multi-User and Single-Challenge (MUSC) setting, called *MUSC-ECPA/MUSC-ECCA*, which allow at most one $\mathcal{O}_{\text{TEST}}$ query in total.

Definition 5 (MUSC-ECPA/ECCA Security). *We say that an adversary \mathcal{A} ($t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, n, Q_e$)-breaks the MUSC-ECPA (resp., MUSC-ECCA) security of KEM, if it ($t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, n, Q_e, 1$)-breaks the MUMC-ECPA (resp., MUMC-ECCA) security, and we denote the corresponding advantage function by $\text{Adv}_{\text{KEM}, n, Q_e}^{\text{musc-ecpa}}(\mathcal{A})$ (resp., $\text{Adv}_{\text{KEM}, n, Q_e}^{\text{musc-ecca}}(\mathcal{A})$).*

4 Decap-Equivalence of Secret Keys & Rank of KEMs

In this section, we study the equivalence of secret keys for KEM schemes when decapsulating a set of ciphertexts, and define a new notion called *rank* for KEMs. This will be our main technique tool in the establishment of the impossibility result later in Sect. 5.

“Two-Step” Decapsulation. Generally, the decapsulation algorithm $\text{Decap}(sk, c)$ of KEM schemes can be decomposed into two parts according to their functionality: an (optional) verification part $\text{Decap}_{\text{vrfy}}(sk, c)$ checking the well-formedness of ciphertext, and a key-derivation part $\text{Decap}_{\text{kd}}(sk, c)$ deriving a decapsulated key $K \in \mathcal{K}$ from the ciphertext. If $\text{Decap}_{\text{vrfy}}(sk, c) = 1$, then $K \leftarrow \text{Decap}_{\text{kd}}(sk, c)$ is invoked and $\text{Decap}(sk, c)$ will output K . If $\text{Decap}_{\text{vrfy}}(sk, c) = 0$, then $\text{Decap}(sk, c)$ will output a fixed symbol like \perp indicating the mal-formedness of c .

We note that some KEM schemes (like CPA-secure KEMs) do not have $\text{Decap}_{\text{vrfy}}$ and $\text{Decap}(sk, c) = \text{Decap}_{\text{kd}}(sk, c)$. Nevertheless, $\text{Decap}_{\text{kd}}(sk, c)$ contributes the core of $\text{Decap}(sk, c)$ in all KEM schemes. Clearly, if $\text{Decap}(sk, c) = K \in \mathcal{K}$, it must hold that $\text{Decap}_{\text{kd}}(sk, c) = K$.

4.1 Decap-Equivalence of Secret Keys

For KEM schemes, we study the decapsulation equivalence of secret keys when they are used to decapsulate a set \mathcal{X} of ciphertexts. Since Decap_{kd} is the essential part of the decapsulation algorithm, the decapsulation equivalence is defined with Decap_{kd} , as shown below.

Definition 6 (\mathcal{X} -Decap-Equivalence of Secret Keys). *Let KEM be a KEM scheme with ciphertext space \mathcal{CT} and secret key space \mathcal{SK} . We define a relation $\text{EquivSK}(\mathcal{X})$ on \mathcal{SK} , parameterized by a set of ciphertexts $\mathcal{X} \subseteq \mathcal{CT}$, as follows:*

$$\text{EquivSK}(\mathcal{X}) := \{(sk, sk') \in \mathcal{SK}^2 \mid \forall c \in \mathcal{X}, \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c)\}.$$

We also define $\text{EquivSK}(\emptyset) := \mathcal{SK}^2$ for the empty set \emptyset .

Clearly, $\text{EquivSK}(\mathcal{X})$ defines an equivalence relation on \mathcal{SK} . We show useful properties of EquivSK in the following lemma.

Lemma 1 (Properties of EquivSK). *For all $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{CT}$,*

- (1) $\text{EquivSK}(\mathcal{X} \cup \mathcal{Y}) = \text{EquivSK}(\mathcal{X}) \cap \text{EquivSK}(\mathcal{Y})$.
- (2) $\mathcal{X} \subseteq \mathcal{Y} \Rightarrow \text{EquivSK}(\mathcal{X}) \supseteq \text{EquivSK}(\mathcal{Y})$.

Proof. Note that (2) follows from (1) directly. It suffices to prove (1). By Definition 6, for any $(sk, sk') \in \mathcal{SK}^2$,

$$\begin{aligned} & (sk, sk') \in \text{EquivSK}(\mathcal{X} \cup \mathcal{Y}) \\ \iff & \forall c \in \mathcal{X} \cup \mathcal{Y}, \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \\ \iff & \forall c \in \mathcal{X}, \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \wedge \forall c \in \mathcal{Y}, \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \\ \iff & (sk, sk') \in \text{EquivSK}(\mathcal{X}) \wedge (sk, sk') \in \text{EquivSK}(\mathcal{Y}) \\ \iff & (sk, sk') \in \text{EquivSK}(\mathcal{X}) \cap \text{EquivSK}(\mathcal{Y}). \quad \square \end{aligned}$$

We also define independence of a set $\mathcal{X} \subseteq \mathcal{CT}$ as follows. If $c \in \mathcal{X}$ but $\text{EquivSK}(\mathcal{X} \setminus \{c\}) = \text{EquivSK}(\mathcal{X})$, then the element c , compared to $\mathcal{X} \setminus \{c\}$, does not contribute to $\text{EquivSK}(\mathcal{X})$. In this case we call c a *dependent element* in \mathcal{X} . Otherwise, if $\text{EquivSK}(\mathcal{X} \setminus \{c\}) \supsetneq \text{EquivSK}(\mathcal{X})$, we call c an *independent element* in \mathcal{X} . For set \mathcal{X} , we call \mathcal{X} an *independent set*, if every $c \in \mathcal{X}$ is an *independent element* in \mathcal{X} . Below we present the formal definition.

Definition 7 (Independent Set for Decap-Equivalence). *Let $\mathcal{X} \subseteq \mathcal{CT}$ be a set of ciphertexts. \mathcal{X} is called an independent set, if for all $c \in \mathcal{X}$, it holds that*

$$\text{EquivSK}(\mathcal{X} \setminus \{c\}) \supsetneq \text{EquivSK}(\mathcal{X}).$$

In particular, we define the empty set \emptyset as an independent set.

4.2 Rank of KEMs

For set \mathcal{X} , there may exist many independent subsets \mathcal{X}' such that $\mathcal{X}' \subseteq \mathcal{X}$ and $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X})$. We define the rank of \mathcal{X} as the cardinality of the largest subset.

Definition 8 (Rank of Set & Rank of KEM for Decap-Equivalence). *Let $\mathcal{X} \subseteq \mathcal{CT}$ be a set of ciphertexts. The rank of \mathcal{X} is defined as*

$$\text{Rank}(\mathcal{X}) := \max\{\#\mathcal{X}' \mid \mathcal{X}' \subseteq \mathcal{X} \wedge \text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X}) \wedge \mathcal{X}' \text{ is independent}\}.$$

In particular, the rank of KEM scheme KEM is defined as $\text{Rank}_{\text{KEM}} := \text{Rank}(\mathcal{CT})$, where \mathcal{CT} is the ciphertext space of KEM.

Obviously, we have $\text{Rank}(\mathcal{X}) \leq \#\mathcal{X}$ and $\text{Rank}_{\text{KEM}} = \text{Rank}(\mathcal{CT}) \leq \#\mathcal{CT}$.

Here, we demonstrate that Rank is well-defined. Namely, there always exists an independent subset $\mathcal{X}' \subseteq \mathcal{X}$ such that $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X})$. We find such an \mathcal{X}' by iteration. In the first step, we set $\mathcal{X}' := \mathcal{X}$. Clearly, $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X})$. If \mathcal{X}' is independent, then we are done. Otherwise \mathcal{X}' is not independent, then $\exists c \in \mathcal{X}'$ such that $\text{EquivSK}(\mathcal{X}' \setminus \{c\}) = \text{EquivSK}(\mathcal{X}')$. So, we remove c from \mathcal{X}' , i.e., $\mathcal{X}' \leftarrow \mathcal{X}' \setminus \{c\}$. Then $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X})$ still holds, while $\#\mathcal{X}'$ is reduced by 1. If \mathcal{X}' is independent, then we are done. Otherwise, we repeat the above procedures until $\#\mathcal{X}' = 0$. Since \mathcal{X} is a finite set, we can always stop with an independent \mathcal{X}' after finite steps, possibly with $\mathcal{X}' = \emptyset$ (which is also independent by definition). Therefore, we can always find an \mathcal{X}' such that $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X})$ and \mathcal{X}' is independent.

We show useful properties of Rank in the following lemma.

Lemma 2 (Properties of Rank). *For all $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{CT}$,*

- (1) $\mathcal{X} \subseteq \mathcal{Y} \Rightarrow \text{Rank}(\mathcal{X}) \leq \text{Rank}(\mathcal{Y})$.
- (2) $\mathcal{X} \subseteq \mathcal{Y}$ and \mathcal{Y} is an independent set $\Rightarrow \mathcal{X}$ is an independent set.
- (3) If \mathcal{X} is an independent set, then $\text{Rank}(\mathcal{X}) = \#\mathcal{X}$.

Proof. To show (1), it suffices to prove $\text{Rank}(\mathcal{X}) \leq \text{Rank}(\mathcal{X} \cup \{c\})$ for a single element $c \in \mathcal{Y} \setminus \mathcal{X}$, then (1) follows by induction. Suppose \mathcal{X}' is the largest independent subset such that $\mathcal{X}' \subseteq \mathcal{X}$ and $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X})$. By definition, $\text{Rank}(\mathcal{X}) = \#\mathcal{X}'$. We consider two cases.

- In the case $\text{EquivSK}(\mathcal{X}) = \text{EquivSK}(\mathcal{X} \cup \{c\})$, \mathcal{X}' is also an independent subset such that $\mathcal{X}' \subseteq \mathcal{X} \cup \{c\}$ and $\text{EquivSK}(\mathcal{X}') = \text{EquivSK}(\mathcal{X} \cup \{c\})$, so $\text{Rank}(\mathcal{X} \cup \{c\}) \geq \#\mathcal{X}'$.
- In the case $\text{EquivSK}(\mathcal{X}) \supsetneq \text{EquivSK}(\mathcal{X} \cup \{c\})$, $\mathcal{X}' \cup \{c\}$ is an independent subset such that $\mathcal{X}' \cup \{c\} \subseteq \mathcal{X} \cup \{c\}$ and $\text{EquivSK}(\mathcal{X}' \cup \{c\}) = \text{EquivSK}(\mathcal{X}') \cap \text{EquivSK}(c) = \text{EquivSK}(\mathcal{X}) \cap \text{EquivSK}(c) = \text{EquivSK}(\mathcal{X} \cup \{c\})$, so $\text{Rank}(\mathcal{X} \cup \{c\}) \geq \#(\mathcal{X}' \cup \{c\}) = \#\mathcal{X}' + 1$.

In either case, we have $\text{Rank}(\mathcal{X}) = \#\mathcal{X}' \leq \text{Rank}(\mathcal{X} \cup \{c\})$. This proves (1).

Next, we prove (2). Since \mathcal{Y} is an independent set, by definition, for every $c \in \mathcal{Y}$, $\text{EquivSK}(\mathcal{Y} \setminus \{c\}) \supsetneq \text{EquivSK}(\mathcal{Y})$. Observe that $\text{EquivSK}(\mathcal{Y}) = \text{EquivSK}(\mathcal{Y} \setminus \{c\}) \cap \text{EquivSK}(c)$, so it implies that $\text{EquivSK}(\mathcal{Y} \setminus \{c\}) \not\subseteq \text{EquivSK}(c)$. Then for every $c \in \mathcal{X}$, since $\mathcal{X} \subseteq \mathcal{Y}$, by Lemma 1, it holds $\text{EquivSK}(\mathcal{Y} \setminus \{c\}) \subseteq \text{EquivSK}(\mathcal{X} \setminus \{c\})$. Combining $\text{EquivSK}(\mathcal{Y} \setminus \{c\}) \subseteq \text{EquivSK}(\mathcal{X} \setminus \{c\})$ with $\text{EquivSK}(\mathcal{Y} \setminus \{c\}) \not\subseteq \text{EquivSK}(c)$, we get that $\text{EquivSK}(\mathcal{X} \setminus \{c\}) \not\subseteq \text{EquivSK}(c)$, and consequently,

$\text{EquivSK}(\mathcal{X} \setminus \{c\}) \supsetneq \text{EquivSK}(\mathcal{X}) = \text{EquivSK}(\mathcal{X} \setminus \{c\}) \cap \text{EquivSK}(c)$. Therefore, \mathcal{X} is also an independent set.

For (3), when \mathcal{X} is an independent set, \mathcal{X} itself is the largest independent subset of \mathcal{X} such that $\text{EquivSK}(\mathcal{X}) = \text{EquivSK}(\mathcal{X})$, so $\text{Rank}(\mathcal{X}) = \#\mathcal{X}$. \square

Lastly, we stress that we do not require any algebraic structure from the secret key space \mathcal{SK} or the ciphertext space \mathcal{CT} . The notions (like independent set, set rank and rank of KEMs) are purely defined based on the equivalence relation ‘‘EquivSK’’ on secret keys.

4.3 Core Lemma

In this subsection, we develop a core lemma, which is crucial in the establishment of the impossibility result later in Sect. 5.

For the ease of notation, by $\text{EquivSK}(c_1, \dots, c_Q)$ we denote $\text{EquivSK}(\{c_1, \dots, c_Q\})$, and by $\text{EquivSK}(c_1, \dots, c_Q \setminus c_i)$ we denote $\text{EquivSK}(\{c_1, \dots, c_Q\} \setminus \{c_i\})$.

Lemma 3 (Core Lemma). *Let KEM be a KEM scheme with ciphertext space \mathcal{CT} . For any ciphertexts $c_1, \dots, c_Q \in \mathcal{CT}$ with $Q \in \mathbb{N}$,*

$$\#\{i \in [Q] \mid \text{EquivSK}(c_1, \dots, c_Q \setminus c_i) \not\subseteq \text{EquivSK}(c_i)\} \leq \text{Rank}_{\text{KEM}}. \quad (7)$$

Proof. Denote by BadIndex the set in the left-hand side of (7) and denote by d the rank of KEM (i.e., $\text{Rank}_{\text{KEM}} = d$).

If $Q \leq d$, the lemma trivially holds. Now, we consider the case $Q \geq d + 1$. Suppose towards a contradiction that $\#\text{BadIndex} \geq d + 1$, which means that BadIndex contains at least $d + 1$ distinct indices, say i_1, \dots, i_{d+1} .

We claim that $\{c_{i_1}, \dots, c_{i_{d+1}}\}$ is an independent set. To prove this claim, it suffices to show $\text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}} \setminus c_i) \supsetneq \text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}})$ for any $i \in \{i_1, \dots, i_{d+1}\}$. Since $\{i_1, \dots, i_{d+1}\} \subseteq \text{BadIndex}$, we have

$$\text{EquivSK}(c_1, \dots, c_Q \setminus c_i) \not\subseteq \text{EquivSK}(c_i) \quad (8)$$

for each $i \in \{i_1, \dots, i_{d+1}\}$. We also have $\text{EquivSK}(c_1, \dots, c_Q \setminus c_i) \subseteq \text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}} \setminus c_i)$ by Lemma 1, then by combining it with (8), we get that

$$\text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}} \setminus c_i) \not\subseteq \text{EquivSK}(c_i). \quad (9)$$

(9) in turn implies that

$$\begin{aligned} \text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}} \setminus c_i) &\supsetneq \text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}} \setminus c_i) \cap \text{EquivSK}(c_i) \\ &= \text{EquivSK}(c_{i_1}, \dots, c_{i_{d+1}}). \end{aligned}$$

This shows the independence of set $\{c_{i_1}, \dots, c_{i_{d+1}}\}$.

Since $\{c_{i_1}, \dots, c_{i_{d+1}}\}$ is an independent subset of \mathcal{CT} , by Lemma 2, we have $\text{Rank}_{\text{KEM}} = \text{Rank}(\mathcal{CT}) \geq \text{Rank}(\{c_{i_1}, \dots, c_{i_{d+1}}\}) = d + 1$, which contradicts with $\text{Rank}_{\text{KEM}} = d$. So it must hold that $\#\text{BadIndex} \leq d$ and Lemma 3 follows. \square

5 Impossibility of Tight Enhanced Security for KEMs

In this section, we present an impossibility result on the tight enhanced security for a class of KEMs whose ranks are *polynomially bounded*. In Subsect. 5.1, we give the main theorem of our impossibility result. Then in Subsect. 5.2, we compute ranks for some well-known KEM schemes, and apply our impossibility result to these KEMs. The applications indicate that for these KEMs there exists no (almost) tight (i.e., linear-preserving) black-box reduction from their enhanced security to any non-interactive complexity assumption.

5.1 Impossibility of Tight Enhanced Security for KEMs

As in [2, 19], we will only consider *simple* reductions, since most reductions in cryptography are simple ones. We recall the definition of simple reduction.

Definition 9 (Simple Reduction [2, 19, 32]). We call an algorithm \mathcal{R} a $(t_{\mathcal{R}}, \epsilon_{\mathcal{R}}, \epsilon_{\mathcal{A}}, n, Q_e)$ -reduction from breaking an NICA $N = (T, U, V)$ to breaking the MUSC-ECPA security of KEM, if \mathcal{R} turns an adversary \mathcal{A} that $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, n, Q_e)$ -breaks the MUSC-ECPA security of KEM (cf. Definition 5) into an algorithm \mathcal{B} that $(t_{\mathcal{R}}, \epsilon_{\mathcal{R}})$ -breaks N (cf. Definition 3).

We call \mathcal{R} *simple*, if \mathcal{R} has only black-box access to \mathcal{A} and executes \mathcal{A} only once (and in particular without rewinding).

The security loss of \mathcal{R} is defined by $\ell_{\mathcal{R}} := \frac{\epsilon_{\mathcal{A}}}{\epsilon_{\mathcal{R}, \mathcal{A}}} \cdot \frac{t_{\mathcal{R}, \mathcal{A}}}{t_{\mathcal{A}}}$. If $\ell_{\mathcal{R}}$ is a small constant, \mathcal{R} is called a *fully tight reduction*; if $\ell_{\mathcal{R}}$ is an a priori fixed polynomial in the security parameter λ , \mathcal{R} is called an *almost tight reduction* or a *linear preserving reduction*.

In the following theorem, we show the impossibility of tight MUSC-ECPA security which is defined in the multi-user and *single-challenge* setting.

Theorem 1 (Impossibility of Tight MUSC-ECPA Security). Let $N = (T, U, V)$ be a non-interactive complexity assumption, and let KEM be an MUSC-ECPA secure KEM scheme with rank $\text{Rank}_{\text{KEM}} = d$. Then any simple $(t_{\mathcal{R}}, \epsilon_{\mathcal{R}}, \epsilon_{\mathcal{A}}, n, Q_e)$ -reduction \mathcal{R} from breaking N to breaking the MUSC-ECPA security of KEM has to lose a factor that is at least linear in the number n of users, assuming N is hard and $Q_e \geq 3dn(n+1)$.

Proof of Theorem 1. We prove the impossibility result by meta-reduction. Following the meta-reduction routine [21, 29, 2], we first describe a hypothetical and inefficient adversary \mathcal{A}^* , then we show how to construct an algorithm \mathcal{B} simulating \mathcal{A}^* efficiently while running the reduction \mathcal{R} .

THE HYPOTHETICAL ADVERSARY \mathcal{A}^* . Let $Q := Q_e/n$. The hypothetical adversary \mathcal{A}^* attacks the MUSC-ECPA security of KEM (cf. Definition 5) as follows.

- **Setup.** \mathcal{A}^* receives (pp, PKList) with $\text{PKList} = \{pk_i\}_{i \in [n]}$.
 \mathcal{A}^* will execute the following procedures, and in particular make the queries therein, in order.
- **Preparation.** For each user $i \in [n]$,
 - (1) \mathcal{A}^* makes $\mathcal{O}_{\text{ENC}}(i)$ query Q times: in the j -th query ($j \in [Q]$), it receives $c_{i,j}$ from $\mathcal{O}_{\text{ENC}}(i)$;
 - (2) \mathcal{A}^* picks an index $j_i \leftarrow_{\$} [Q]$ uniformly at random, and for each $j \in [Q \setminus j_i]$, it queries $\mathcal{O}_{\text{REV}}(i, c_{i,j})$ and receives $K_{i,j}$.
- **Corruption.** \mathcal{A}^* picks a user index $i^* \leftarrow_{\$} [n]$ uniformly at random, and for each $i \in [n \setminus i^*]$, it queries $\mathcal{O}_{\text{CORR}}(i)$ and receives sk_i .
- **Check.** For each $i \in [n \setminus i^*]$, \mathcal{A}^* checks whether $\text{Decap}_{\text{kd}}(sk_i, c_{i,j}) = K_{i,j}$ holds for all $j \in [Q \setminus j_i]$. It aborts immediately if one of these checks fails.
- **Test.** \mathcal{A}^* queries $\mathcal{O}_{\text{TEST}}(i^*, c_{i^*, j_{i^*}})$ and receives a challenge K^* .
- **Output.**
 - (1) (Inefficient step) \mathcal{A}^* picks a secret key sk^* uniformly at random from the set $\{sk \mid \forall j \in [Q \setminus j_{i^*}], \text{Decap}_{\text{kd}}(sk, c_{i^*, j}) = K_{i^*, j}\}$, which is an equivalence class of $\text{EquivSK}(c_{i^*, 1}, \dots, c_{i^*, Q} \setminus c_{i^*, j_{i^*}})$.
 - (2) Using the above sk^* , \mathcal{A}^* computes $K := \text{Decap}_{\text{kd}}(sk^*, c_{i^*, j_{i^*}})$. If $K = K^*$, it outputs $\beta' = 1$; otherwise it outputs $\beta' = 0$.

Note that \mathcal{A}^* makes $nQ (= Q_e)$ \mathcal{O}_{ENC} queries in total and makes at most one $\mathcal{O}_{\text{TEST}}$ query.

ANALYSIS OF \mathcal{A}^* 'S ADVANTAGE. Let sk_{i^*} denote the secret key of user i^* chosen by the experiment. By the perfect correctness of KEM, it holds that $\text{Decap}_{\text{kd}}(sk_{i^*}, c_{i^*, j}) = \text{Decap}(sk_{i^*}, c_{i^*, j}) = K_{i^*, j}$ for each $j \in [Q \setminus j_{i^*}]$. Consequently,

$$\text{Decap}_{\text{kd}}(sk_{i^*}, c_{i^*, j}) = K_{i^*, j} = \text{Decap}_{\text{kd}}(sk^*, c_{i^*, j})$$

for each $j \in [Q \setminus j_{i^*}]$, where sk^* is the secret key chosen by \mathcal{A}^* . It implies that

$$(sk_{i^*}, sk^*) \in \text{EquivSK}(c_{i^*, 1}, \dots, c_{i^*, Q} \setminus c_{i^*, j_{i^*}}).$$

Let bad denote the event that $\text{EquivSK}(c_{i^*,1}, \dots, c_{i^*,Q} \setminus c_{i^*,j_{i^*}}) \not\subseteq \text{EquivSK}(c_{i^*,j_{i^*}})$. By Lemma 3 (Core Lemma) and the uniformity of j_{i^*} over $[Q]$, we have $\Pr[\text{bad}] \leq d/Q = nd/Q_e$. Let β denote the single challenge bit in the experiment.

- In the case $\beta = 0$, the K^* output by $\mathcal{O}_{\text{TEST}}$ is the real key encapsulated in $c_{i^*,j_{i^*}}$. By the perfect correctness of KEM, it holds that $\text{Decap}_{\text{kd}}(sk_{i^*}, c_{i^*,j_{i^*}}) = \text{Decap}(sk_{i^*}, c_{i^*,j_{i^*}}) = K^*$. If bad does not occur,

$$(sk_{i^*}, sk^*) \in \text{EquivSK}(c_{i^*,1}, \dots, c_{i^*,Q} \setminus c_{i^*,j_{i^*}}) \subseteq \text{EquivSK}(c_{i^*,j_{i^*}}),$$

thus $K = \text{Decap}_{\text{kd}}(sk^*, c_{i^*,j_{i^*}}) = \text{Decap}_{\text{kd}}(sk_{i^*}, c_{i^*,j_{i^*}}) = K^*$. It implies $\Pr[\beta' = 1 \mid \beta = 0 \wedge \neg \text{bad}] = 1$, and consequently,

$$\begin{aligned} \Pr[\beta' = 1 \mid \beta = 0] &\geq \Pr[\neg \text{bad}] \cdot \Pr[\beta' = 1 \mid \beta = 0 \wedge \neg \text{bad}] \\ &= \Pr[\neg \text{bad}] \cdot 1 = 1 - \Pr[\text{bad}] \geq 1 - nd/Q_e. \end{aligned}$$

- In the case $\beta = 1$, the K^* output by $\mathcal{O}_{\text{TEST}}$ is a random key uniformly chosen from \mathcal{K} , so $K = K^*$ holds with probability exactly $1/\#\mathcal{K}$. It implies $\Pr[\beta' = 1 \mid \beta = 1] = 1/\#\mathcal{K}$.

Overall, the advantage of \mathcal{A}^* in the MUSC-ECPA security experiment is

$$\begin{aligned} \epsilon_{\mathcal{A}^*} &= 2 \cdot \left| \Pr[\beta' = \beta] - \frac{1}{2} \right| = \left| \Pr[\beta' = 1 \mid \beta = 0] - \Pr[\beta' = 1 \mid \beta = 1] \right| \\ &\geq 1 - nd/Q_e - 1/\#\mathcal{K}. \end{aligned} \tag{10}$$

THE META-REDUCTION \mathcal{B} . Next, we construct an efficient algorithm \mathcal{B} , which runs reduction \mathcal{R} as a subroutine and attempts to break the NICA N . \mathcal{B} will play the role of the hypothetical adversary \mathcal{A}^* to interact with \mathcal{R} . For the sake of efficiently emulating \mathcal{A}^* , \mathcal{B} will rewind \mathcal{R} to learn more information from the its responses. More precisely, given an instance x of N , where $(x, w) \leftarrow_s T$, \mathcal{B} works as follows.

- **Setup.** \mathcal{B} runs $\mathcal{R}(x)$ to obtain (pp, PKList) where $\text{PKList} = \{pk_i\}_{i \in [n]}$. \mathcal{B} initializes two arrays of n entries, $SK[\cdot]$ and $SK^*[\cdot]$, by \emptyset .
 \mathcal{B} plays the role of adversary, executes the following procedures and makes the queries to \mathcal{R} in order.
- **Preparation.** For each user $i \in [n]$,
 - (1) \mathcal{B} makes $\mathcal{O}_{\text{ENC}}(i)$ query Q times: in the j -th $\mathcal{O}_{\text{ENC}}(i)$ query ($j \in [Q]$), it receives $c_{i,j}$ from \mathcal{R} ;
 - (2) \mathcal{B} picks an index $j_i \leftarrow_s [Q]$ uniformly at random, and for each $j \in [Q \setminus j_i]$, it queries $\mathcal{O}_{\text{REV}}(i, c_{i,j})$ and receives $K_{i,j}$ from \mathcal{R} .
 Let the state after this preparation step be st_{prep} .
 \mathcal{B} picks a user index $i^* \leftarrow_s [n]$ uniformly at random.
- **Rewinding.** Next, \mathcal{B} will rewind \mathcal{R} n times, all starting from state st_{prep} . In the ι -th rewind ($\iota \in [n]$), \mathcal{B} proceeds as follows:
 - (1) \mathcal{B} rewinds \mathcal{R} to the state st_{prep} . For each $i \in [n \setminus \iota]$, \mathcal{B} queries $\mathcal{O}_{\text{CORR}}(i)$ and receives $sk_i^{(\iota)}$ from \mathcal{R} .
 - (2) For each $i \in [n \setminus \iota]$, \mathcal{B} checks whether or not $\text{Decap}_{\text{kd}}(sk_i^{(\iota)}, c_{i,j}) = K_{i,j}$ holds for all $j \in [Q \setminus j_i]$, and if so, it sets $SK[i] := sk_i^{(\iota)}$. If $\iota = i^*$, \mathcal{B} additionally sets $SK^*[i] := sk_i^{(i^*)}$.
 - (3) Let the state at the moment be $st_{\text{rewind}}^{(\iota)}$. If $\iota < n$, \mathcal{B} goes to the next rewind (i.e., $\iota \leftarrow \iota + 1$).
- **Check.** For each $i \in [n \setminus i^*]$, \mathcal{B} checks whether or not $SK^*[i] \neq \emptyset$ (i.e., $\text{Decap}_{\text{kd}}(sk_i^{(i^*)}, c_{i,j}) = K_{i,j}$ holds for all $j \in [Q \setminus j_i]$). It aborts immediately if one of these check fails, and sets a flag $\text{checkfail}_1 := \text{true}$.
- **Test.** \mathcal{B} rewinds \mathcal{R} back to the state $st_{\text{rewind}}^{(i^*)}$. \mathcal{B} queries $\mathcal{O}_{\text{TEST}}(i^*, c_{i^*,j_{i^*}})$ and receives a challenge K^* from \mathcal{R} .
- **Output.**
 - (1) \mathcal{B} checks whether or not $SK[i^*] \neq \emptyset$ (i.e., $SK[i^*] = sk_{i^*}^{(\iota^*)}$ for some $\iota^* \neq i^*$, s.t. $\text{Decap}_{\text{kd}}(sk_{i^*}^{(\iota^*)}, c_{i^*,j}) = K_{i^*,j}$ for all $j \in [Q \setminus j_{i^*}]$). It aborts if the check fails, and sets a flag $\text{checkfail}_2 := \text{true}$.
 - (2) Using $SK[i^*]$, \mathcal{B} computes $K := \text{Decap}_{\text{kd}}(SK[i^*], c_{i^*,j_{i^*}})$. If $K = K^*$, it outputs $\beta' = 1$ to \mathcal{R} ; otherwise it outputs $\beta' = 0$ to \mathcal{R} .

Finally, \mathcal{B} receives a solution s from \mathcal{R} , and outputs s to its own challenger.

\mathcal{B} 'S RUNNING TIME. \mathcal{B} essentially runs \mathcal{R} one complete run plus $(n-1)$ incomplete runs. Moreover it executes Decap_{kd} at most $n(n-1)(Q-1)+1$ times. Thus the total running time of \mathcal{B} is

$$t_{\mathcal{B}} \leq n \cdot t_{\mathcal{R}} + n^2 Q \cdot t_{\text{Decap}} = n \cdot t_{\mathcal{R}} + n Q_e \cdot t_{\text{Decap}},$$

where t_{Decap} denotes the running time of the Decap algorithm of KEM.

ANALYSIS OF \mathcal{B} 'S ADVANTAGE. Denote by **bad** the event that $\text{EquivSK}(c_{i^*,1}, \dots, c_{i^*,Q} \setminus c_{i^*,j_{i^*}}) \not\subseteq \text{EquivSK}(c_{i^*,j_{i^*}})$. We first show that in the case of $\text{checkfail}_1 \vee (\neg \text{checkfail}_2 \wedge \neg \text{bad})$, \mathcal{B} simulates the hypothetical adversary \mathcal{A}^* perfectly.

- If checkfail_1 occurs, \mathcal{B} aborts, and \mathcal{A}^* would also abort in the check step since $\text{Decap}_{\text{kd}}(sk_i^{(i^*)}, c_{i,j}) \neq K_{i,j}$ for some $i \in [n \setminus i^*]$ and some $j \in [Q \setminus j_{i^*}]$.
- If $\neg \text{checkfail}_1 \wedge \neg \text{checkfail}_2 \wedge \neg \text{bad}$, \mathcal{B} obtains a secret key $SK[i^*]$ such that $\text{Decap}_{\text{kd}}(SK[i^*], c_{i^*,j}) = K_{i^*,j}$ for each $j \in [Q \setminus j_{i^*}]$. Since \mathcal{A}^* 's sk^* also satisfies $\text{Decap}_{\text{kd}}(sk^*, c_{i^*,j}) = K_{i^*,j}$ for each $j \in [Q \setminus j_{i^*}]$, it implies that

$$(SK[i^*], sk^*) \in \text{EquivSK}(c_{i^*,1}, \dots, c_{i^*,Q} \setminus c_{i^*,j_{i^*}}).$$

Since **bad** does not occur,

$$(SK[i^*], sk^*) \in \text{EquivSK}(c_{i^*,1}, \dots, c_{i^*,Q} \setminus c_{i^*,j_{i^*}}) \subseteq \text{EquivSK}(c_{i^*,j_{i^*}}).$$

Consequently, the $K = \text{Decap}_{\text{kd}}(SK[i^*], c_{i^*,j_{i^*}})$ computed by \mathcal{B} is identical to the $K = \text{Decap}_{\text{kd}}(sk^*, c_{i^*,j_{i^*}})$ computed by \mathcal{A}^* , so the simulation is perfect.

Therefore, \mathcal{B} simulates \mathcal{A}^* perfectly for \mathcal{R} when $\text{checkfail}_1 \vee (\neg \text{checkfail}_2 \wedge \neg \text{bad})$, and by the difference lemma, we have

$$\begin{aligned} |\epsilon_{\mathcal{B}} - \epsilon_{\mathcal{R}, \mathcal{A}^*}| &\leq \Pr[\neg \text{checkfail}_1 \wedge (\text{checkfail}_2 \vee \text{bad})] \\ &\leq \Pr[\neg \text{checkfail}_1 \wedge \text{checkfail}_2] + \Pr[\text{bad}]. \end{aligned}$$

By Lemma 3 (Core Lemma) and the uniformity of j_{i^*} over $[Q]$, we have $\Pr[\text{bad}] \leq d/Q = dn/Q_e$. Next we bound the probability $\Pr[\neg \text{checkfail}_1 \wedge \text{checkfail}_2]$. Note that checkfail_2 can only occur if the event $E : \exists i \in [n], SK[i] = \emptyset$ occurs. As i^* is chosen uniformly at random from $[n]$ and the view of \mathcal{R} before the Test query is independent of i^* , we have $i \in [n \setminus i^*]$ with probability $1-1/n$. In this case checkfail_1 occurs and thus $\Pr[\text{checkfail}_1 | E] \geq 1-1/n$. Now since $\text{checkfail}_2 \Rightarrow E$ it holds that $\Pr[\neg \text{checkfail}_1 \wedge \text{checkfail}_2] \leq \Pr[\neg \text{checkfail}_1 \wedge E] = \Pr[\neg \text{checkfail}_1 | E] \cdot \Pr[E] \leq \Pr[\neg \text{checkfail}_1 | E] = 1 - \Pr[\text{checkfail}_1 | E] \leq 1/n$. Overall, it holds that $|\epsilon_{\mathcal{B}} - \epsilon_{\mathcal{R}, \mathcal{A}^*}| \leq 1/n + dn/Q_e$, thus,

$$\epsilon_{\mathcal{R}, \mathcal{A}^*} \leq \epsilon_{\mathcal{B}} + 1/n + dn/Q_e. \quad (11)$$

BOUNDING THE SECURITY LOSS. Assuming that no adversary \mathcal{B} is able to (t_N, ϵ_N) -break the NICA N with $t_N = t_{\mathcal{B}} \leq n \cdot t_{\mathcal{R}} + n Q_e \cdot t_{\text{Decap}}$, we must have $\epsilon_{\mathcal{B}} \leq \epsilon_N$. By combining (10) and (11), the security loss of reduction \mathcal{R} is

$$\begin{aligned} \ell_{\mathcal{R}} &\geq \frac{\epsilon_{\mathcal{A}^*}}{\epsilon_{\mathcal{R}, \mathcal{A}^*}} \cdot \frac{t_{\mathcal{R}, \mathcal{A}^*}}{t_{\mathcal{A}^*}} \geq \frac{1 - dn/Q_e - 1/\#\mathcal{K}}{\epsilon_{\mathcal{B}} + 1/n + dn/Q_e} \cdot 1 \geq \frac{1 - dn/Q_e - 1/\#\mathcal{K}}{\epsilon_N + 1/n + dn/Q_e} \\ &\geq n \cdot (1 - n\epsilon_N - dn(n+1)/Q_e - 1/\#\mathcal{K}), \end{aligned}$$

where the last inequality holds by inspection, namely, $n \cdot (1 - n\epsilon_N - dn(n+1)/Q_e - 1/\#\mathcal{K}) \cdot (\epsilon_N + 1/n + dn/Q_e) = 1 - dn/Q_e - 1/\#\mathcal{K} - (n\epsilon_N + n^2 d/Q_e) \cdot (n\epsilon_N + dn(n+1)/Q_e + 1/\#\mathcal{K}) \leq 1 - dn/Q_e - 1/\#\mathcal{K}$. Thus, any reduction \mathcal{R} from breaking N to breaking the MUSC-ECPA security of KEM loses at least a factor of

$$\ell = n \cdot (1 - n\epsilon_N - dn(n+1)/Q_e - 1/\#\mathcal{K}),$$

where n denotes the number of users, ϵ_N represents the hardness of NICA N , d is the rank of KEM, Q_e is the number of \mathcal{O}_{ENC} queries allowed in the MUSC-ECPA experiment, and $\#\mathcal{K}$ denotes the size of the encapsulated key space \mathcal{K} .

Assuming that N is hard and $Q_e \geq 3dn(n+1)$, we compute the security loss factor ℓ in two cases as examples. In the first case, we only make very *weak* assumptions, and in the second case, we make *mild* but still far more realistic assumptions.

- **Weak case (in the concrete setting).** In the case that $\epsilon_N \leq 1/(12n)$, $Q_e \geq 3dn(n+1)$ and $\#\mathcal{K} \geq 2$, we have $\ell \geq n/12$.
- **Mild case (in the asymptotic setting).** In the case that $\epsilon_N \leq 1/(\lambda n)$, $Q_e \geq \lambda dn(n+1)$ and $\#\mathcal{K} \geq \lambda$, where λ is the security parameter, we have $\ell = n(1 - 3/\lambda) = n(1 - o(1)) \approx n$.

In either case, the security loss ℓ is at least linear in n . This completes the proof of Theorem 1. ■

Observe that MUSC-ECPA is tightly implied by all of the MUSC-ECCA (multi-user and single-challenge ECCA), MUMC-ECPA (multi-user and *multi-challenge* ECPA) and MUMC-ECCA (multi-user and *multi-challenge* ECCA) securities. Hence the impossibility of tight MUSC-ECPA security shown in Theorem 1 directly yields the impossibility of tight MUSC-ECCA, tight MUMC-ECPA and tight MUMC-ECCA, as well. We conclude these in the following corollary.

Corollary 1 (Impossibility of Tight MUSC-ECCA, MUMC-ECPA & MUMC-ECCA).

Let $N = (T, U, V)$ be a non-interactive complexity assumption, and let KEM be an MUSC-ECCA (resp., MUMC-ECPA, MUMC-ECCA) secure KEM with rank $\text{Rank}_{\text{KEM}} = d$. Then any simple $(t_{\mathcal{R}}, \epsilon_{\mathcal{R}}, \epsilon_{\mathcal{A}}, n, Q_e)$ -reduction \mathcal{R} from breaking N to breaking the MUSC-ECCA (resp., MUMC-ECPA, MUMC-ECCA) security of KEM has to lose a factor that is at least linear in the number n of users, assuming N is hard and $Q_e \geq 3dn(n+1)$.

Remark 1. Following [2], our impossibility results can be naturally generalized to reductions that may execute the adversary algorithm several times sequentially.

5.2 Applications of Our Impossibility Result to Well-Known KEMs

In the last two decades, many PKE schemes [12, 34, 8, 9, 10, 27, 14, 15, 18] (to name a few) were proposed, explicitly or implicitly, in the KEM + DEM paradigm [10] and their securities are proved in the standard model. All the KEMs inherent in these PKEs have their own charm. For example, the ElGamal-KEM [12], CS-KEM [8, 9, 10] and KD-KEM [27] are among the most efficient KEMs. The GHKW-KEM [14] and HLLG-KEM [18] are core building blocks in achieving (almost) tightly IND-mCCA security for PKE. The NY-KEM [34] is a generic approach to CCA-secure PKE/KEM from CPA-secure PKE, which in turn can be built upon CPA-secure KEM. Note that these KEMs (except the ElGamal-KEM) have neither secret key uniqueness nor re-randomizability, so the impossibility results in existing works [2] do not apply to them.

Next, we will compute the ranks for these KEMs and apply our impossibility result on them. The computation results show that the ranks of these KEM are either small constants or upper bounded by small polynomials in λ .

- The CPA-secure ElGamal-KEM [12] has rank 1 (cf. Appendix A).
- The CCA-secure CS-KEM in [8] has rank 1 (cf. Appendix A) and another version in [9, 10] has rank 2 (see next).
- The CCCA (constrained CCA) secure KD-KEM [27] has rank at most 4 (cf. Appendix A).
- The PCA (plaintext check attacks) secure GHKW-KEM used in the tightly IND-mCCA secure GHKW-PKE [14] has rank at most $6k\lambda$, with k the parameter of the MDDH assumptions [13] (e.g., MDDH corresponds to the DDH assumption when $k = 1$ and includes k -Linear assumptions for a general $k \geq 2$) (cf. Appendix A).
- The tightly mCCA-secure HLLG-KEM [18] has rank at most $2k$, with k the parameter of the MDDH assumptions (cf. Appendix A).
- The CCA-secure NY-KEM [34] has polynomially-bounded rank, as long as the underlying CPA-secure KEM does (cf. Appendix A). Thus many concrete instantiations of NY-KEM have polynomially-bounded rank, e.g., the NY-KEMs whose underlying CPA-secure KEMs are instantiated with the KEMs shown above (such as ElGamal).

Our impossibility result works well on these KEMs. Their polynomially-bounded ranks indicate that the MUSC-ECPA (or even MUSC-ECCA, MUMC-ECPA, MUMC-ECCA) security of these KEM schemes suffer from a security loss factor $\Omega(n)$ with n the number of users, when reducing to non-interactive complexity assumptions.

Here we only show how to compute rank for the CS-KEM [9, 10], and put the rank computations of other KEMs in Appendix A.

Rank Computation for Cramer-Shoup's CCA-secure KEM [9, 10]. Let us first recall the construction of the CS-KEM in [9, 10].

Let $(\mathbb{G}, p, g_1, g_2)$ be a group of prime order p and with random generators g_1, g_2 . Let H be a hash function from \mathbb{G}^2 to \mathbb{Z}_p .

- The public key is $pk := (g_1, g_2, c, d, h)$ where $c := g_1^{x_1} g_2^{x_2}$, $d := g_1^{y_1} g_2^{y_2}$ and $h := g_1^{z_1} g_2^{z_2}$ for uniformly chosen $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow_{\$} \mathbb{Z}_p$, and the secret key is $sk := (x_1, x_2, y_1, y_2, z_1, z_2)$.
- $\text{Encap}(pk)$ samples $r \leftarrow_{\$} \mathbb{Z}_p$ uniformly, computes $u_1 := g_1^r$, $u_2 := g_2^r$, $\alpha := H(u_1, u_2)$, $v := c^r d^{r^\alpha}$, $K := h^r$, and outputs $c := (u_1, u_2, v)$ and K .
- $\text{Decap}(sk, c = (u_1, u_2, v))$ outputs $K := u_1^{z_1} u_2^{z_2}$ if $u_1^{x_1+y_1\alpha} \cdot u_2^{x_2+y_2\alpha} = v$ holds, where $\alpha := H(u_1, u_2)$, and outputs \perp otherwise.

The secret key space is $\mathcal{SK} = \mathbb{Z}_p^6$, the ciphertext space is $\mathcal{CT} = \mathbb{G}^3$, and the key-derivation part $\text{Decap}_{\text{kd}}(sk, c)$ outputs $K := u_1^{z_1} u_2^{z_2}$.

We show that the CS-KEM in [9, 10] has rank 2. For a ciphertext $c = (u_1, u_2, v) \in \mathcal{CT}$, we can always write $u_1 = g_1^{r_1}$ and $u_2 = g_1^{r_2}$ with $r_1, r_2 \in \mathbb{Z}_p$. We compute $\text{EquivSK}(c)$: for any $sk = (x_1, x_2, y_1, y_2, z_1, z_2)$, $sk' = (x'_1, x'_2, y'_1, y'_2, z'_1, z'_2) \in \mathcal{SK}$, $(sk, sk') \in \text{EquivSK}(c) \iff \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \iff u_1^{z_1} u_2^{z_2} = u_1^{z'_1} u_2^{z'_2} \iff r_1 \cdot z_1 + r_2 \cdot z_2 = r_1 \cdot z'_1 + r_2 \cdot z'_2$. So, $\text{EquivSK}(c) = \{(sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \mid r_1 \cdot z_1 + r_2 \cdot z_2 = r_1 \cdot z'_1 + r_2 \cdot z'_2\}$.

Consequently, we have the following facts.

- (1) $\text{EquivSK}(\mathcal{CT}) = \bigcap_{c \in \mathcal{CT}} \text{EquivSK}(c) = \{(sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \mid \bigwedge_{r_1, r_2 \in \mathbb{Z}_p} r_1 \cdot z_1 + r_2 \cdot z_2 = r_1 \cdot z'_1 + r_2 \cdot z'_2\} = \{(sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \mid z_1 = z'_1 \wedge z_2 = z'_2\}$.
- (2) For any two ciphertexts $c^{(1)} = (u_1^{(1)} = g_1^{r_1^{(1)}}, u_2^{(1)} = g_1^{r_2^{(1)}}, v^{(1)})$, $c^{(2)} = (u_1^{(2)} = g_1^{r_1^{(2)}}, u_2^{(2)} = g_1^{r_2^{(2)}}, v^{(2)})$, if $(r_1^{(1)}, r_2^{(1)}) \in \mathbb{Z}_p^2$ is linearly independent of $(r_1^{(2)}, r_2^{(2)})$, e.g., $(r_1^{(1)}, r_2^{(1)}) = (1, 0)$ and $(r_1^{(2)}, r_2^{(2)}) = (0, 1)$, the matrix $\begin{pmatrix} r_1^{(1)} & r_2^{(1)} \\ r_1^{(2)} & r_2^{(2)} \end{pmatrix}$ is invertible, thus

$$\begin{aligned} \text{EquivSK}(c^{(1)}, c^{(2)}) &= \text{EquivSK}(c^{(1)}) \cap \text{EquivSK}(c^{(2)}) \\ &= \left\{ (sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \mid \begin{array}{l} r_1^{(1)} \cdot z_1 + r_2^{(1)} \cdot z_2 = r_1^{(1)} \cdot z'_1 + r_2^{(1)} \cdot z'_2 \\ \wedge r_1^{(2)} \cdot z_1 + r_2^{(2)} \cdot z_2 = r_1^{(2)} \cdot z'_1 + r_2^{(2)} \cdot z'_2 \end{array} \right\} \\ &= \left\{ (sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \mid \begin{pmatrix} r_1^{(1)} & r_2^{(1)} \\ r_1^{(2)} & r_2^{(2)} \end{pmatrix} \cdot \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} r_1^{(1)} & r_2^{(1)} \\ r_1^{(2)} & r_2^{(2)} \end{pmatrix} \cdot \begin{pmatrix} z'_1 \\ z'_2 \end{pmatrix} \right\} \\ &= \{(sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \mid z_1 = z'_1 \wedge z_2 = z'_2\} = \text{EquivSK}(\mathcal{CT}). \end{aligned}$$

Clearly, we have both $\text{EquivSK}(c^{(1)}, c^{(2)}) \subsetneq \text{EquivSK}(c^{(1)})$ and $\text{EquivSK}(c^{(1)}, c^{(2)}) \subsetneq \text{EquivSK}(c^{(2)})$, thus $\{c^{(1)}, c^{(2)}\}$ is an independent set.

- (3) For any three ciphertexts $c^{(1)} = (u_1^{(1)} = g_1^{r_1^{(1)}}, u_2^{(1)} = g_1^{r_2^{(1)}}, v^{(1)})$, $c^{(2)} = (u_1^{(2)} = g_1^{r_1^{(2)}}, u_2^{(2)} = g_1^{r_2^{(2)}}, v^{(2)})$, $c^{(3)} = (u_1^{(3)} = g_1^{r_1^{(3)}}, u_2^{(3)} = g_1^{r_2^{(3)}}, v^{(3)})$, since the linear space \mathbb{Z}_p^2 has dimension 2, the three vectors $(r_1^{(1)}, r_2^{(1)})$, $(r_1^{(2)}, r_2^{(2)})$, $(r_1^{(3)}, r_2^{(3)})$ in \mathbb{Z}_p^2 must be linearly dependent. Say $(r_1^{(3)}, r_2^{(3)}) = (a \cdot r_1^{(1)} + b \cdot r_1^{(2)}, a \cdot r_2^{(1)} + b \cdot r_2^{(2)})$ for some coefficients $a, b \in \mathbb{Z}_p$. Then we have $\text{EquivSK}(c^{(1)}, c^{(2)}) \subseteq \text{EquivSK}(c^{(3)})$, as shown below.

For any $(sk = (\dots, z_1, z_2), sk' = (\dots, z'_1, z'_2)) \in \text{EquivSK}(c^{(1)}, c^{(2)})$, it holds $r_1^{(1)} \cdot z_1 + r_2^{(1)} \cdot z_2 = r_1^{(1)} \cdot z'_1 + r_2^{(1)} \cdot z'_2$ and $r_1^{(2)} \cdot z_1 + r_2^{(2)} \cdot z_2 = r_1^{(2)} \cdot z'_1 + r_2^{(2)} \cdot z'_2$, thus

$$\begin{aligned} r_1^{(3)} \cdot z_1 + r_2^{(3)} \cdot z_2 &= (a \cdot r_1^{(1)} + b \cdot r_1^{(2)}) \cdot z_1 + (a \cdot r_2^{(1)} + b \cdot r_2^{(2)}) \cdot z_2 \\ &= a \cdot (r_1^{(1)} \cdot z_1 + r_2^{(1)} \cdot z_2) + b \cdot (r_1^{(2)} \cdot z_1 + r_2^{(2)} \cdot z_2) \\ &= a \cdot (r_1^{(1)} \cdot z'_1 + r_2^{(1)} \cdot z'_2) + b \cdot (r_1^{(2)} \cdot z'_1 + r_2^{(2)} \cdot z'_2) \\ &= (a \cdot r_1^{(1)} + b \cdot r_1^{(2)}) \cdot z'_1 + (a \cdot r_2^{(1)} + b \cdot r_2^{(2)}) \cdot z'_2 = r_1^{(3)} \cdot z'_1 + r_2^{(3)} \cdot z'_2, \end{aligned}$$

so $(sk, sk') \in \text{EquivSK}(c^{(3)})$.

The fact that $\text{EquivSK}(c^{(1)}, c^{(2)}) \subseteq \text{EquivSK}(c^{(3)})$ implies $\text{EquivSK}(c^{(1)}, c^{(2)}, c^{(3)}) = \text{EquivSK}(c^{(1)}, c^{(2)}) \cap \text{EquivSK}(c^{(3)}) = \text{EquivSK}(c^{(1)}, c^{(2)})$. Therefore, $\{c^{(1)}, c^{(2)}, c^{(3)}\}$ is not independent for any three ciphertexts $c^{(1)}, c^{(2)}, c^{(3)}$.

Overall, the largest independent subset $\mathcal{X} \subseteq \mathcal{CT}$ such that $\text{EquivSK}(\mathcal{X}) = \text{EquivSK}(\mathcal{CT})$ has two ciphertexts. So, the CS-KEM in [9, 10] has rank 2.

6 Enhancedly Secure KEM with Optimal Tightness

In this section, we present KEMs with enhanced security, where the security reduction has a loss factor $\Theta(n)$ with n the number of users. Combining with the impossibility result shown in Sect. 5, the enhanced security of these KEMs are *optimal* regarding tightness.

More precisely, we will prove that, any IND-mCPA/mCCA secure KEM is itself MUMC-ECPA/ECCA secure, with security reduction losing a factor of $O(n)$. Therefore, to obtain MUMC-ECPA/ECCA secure KEMs with optimal security reduction (i.e., security loss = $\Theta(n)$), it suffices to construct tightly IND-mCPA/mCCA secure KEMs (i.e., the security loss = $\Theta(1)$). Luckily, there were already a handful of such KEMs.

- The ElGamal public-key encryption (PKE) [12] is tightly IND-mCPA secure based on the DDH assumption with security loss $\Theta(1)$ [33].
- In 2012, Hofheinz and Jager [20] presented the first tightly IND-mCCA secure PKE based on (matrix) DDH assumptions [13], with security loss $\Theta(1)$.
- Recent works [14, 15, 18] proposed efficient IND-mCCA secure PKE schemes based on (matrix) DDH assumptions [13], with security loss $O(\lambda)$.

Note that PKE can be used as KEM naturally by encrypting a random key K . These yield (almost) tightly IND-mCPA/mCCA secure KEMs with security loss $\Theta(1)$ (resp., $O(\lambda)$). Combining with our new result, the KEMs derived from [12, 20, 14, 15, 18] achieve MUMC-ECPA/ECCA security based on the standard (matrix) DDH assumptions with security loss $\Theta(n)$ [12, 20] (resp., $O(\lambda n)$ [14, 15, 18]), thus the tightness of their MUMC-ECPA/ECCA security is optimal (resp., almost optimal).

The Non-Triviality of Our Reduction. We stress that our reduction from MUMC-ECPA/ECCA security to IND-mCPA/mCCA security is non-trivial. A straightforward reduction works as follows. An IND-mCPA/mCCA adversary \mathcal{B} simulates the MUMC-ECPA/ECCA experiment for \mathcal{A} by guessing the set of corrupted users, generating the public keys and secret keys of the corrupted users itself, and embedding the public keys in the IND-mCPA/mCCA experiment into (one of) the uncorrupted users.

Note that guessing the set of corrupted users will incur two problems in the security reduction.

- Firstly, it will incur an exponential loss factor, since there are 2^n possibilities of corrupted users in total, which is exponentially large when $n \geq \lambda$.
- Moreover, it is hard for \mathcal{B} to answer key reveal queries w.r.t. uncorrupted users for \mathcal{A} , since the IND-mCPA/mCCA experiment does not provide a key reveal oracle \mathcal{O}_{REV} .

We addressed the above two problems and provide a new reduction which loses only a linear factor $O(n)$. Our reduction goes with n hybrids. In the η -th hybrid ($\eta \in [n]$), we change the encapsulated keys in $\mathcal{O}_{\text{TEST}}$ w.r.t. user η from real keys K_0 to random keys K_1 .

- **One user at a time.** To avoid an exponential loss factor, our reduction focuses on only a single user at a time. In the η -th hybrid, our reduction embeds the public key in the IND-mCPA/mCCA experiment into the public key of user η . There are two cases. If \mathcal{A} never corrupts user η , \mathcal{B} can simulate the MUMC-ECPA/ECCA experiment perfectly for \mathcal{A} . So the change of $\mathcal{O}_{\text{TEST}}$ for user η is unnoticeable to \mathcal{A} by the IND-mCPA/mCCA security. If \mathcal{A} asks to corrupt user η , \mathcal{B} aborts immediately. Note that in the latter case, \mathcal{A} is not allowed to query $\mathcal{O}_{\text{TEST}}$ for user η when user η is (going to be) corrupted. So the change of $\mathcal{O}_{\text{TEST}}$ for user η is conceptual.
- **Key reveal with random keys.** To handle key reveal queries for user η , we borrow the ideas from [30]. If \mathcal{A} never corrupts user η , \mathcal{B} can output a random key for key reveal queries since \mathcal{A} never sees the secret key of user η . If \mathcal{A} asks to corrupt user η , \mathcal{B} can also output a random key for key reveal queries before the corruption and aborts immediately when the corruption happens.

With only n hybrids, we change all encapsulated keys in $\mathcal{O}_{\text{TEST}}$ to random. This shows the indistinguishability of $\beta = 0$ and $\beta = 1$ in the MUMC-ECPA/ECCA experiment. Overall, our reduction only loses a linear factor $O(n)$ from MUMC-ECPA/ECCA to the IND-mCPA/mCCA security.

Theorem 2 (IND-mCPA/mCCA $\xrightarrow{O(n)}$ MUMC-ECPA/ECCA for KEM). Let KEM be an IND-mCPA (resp., IND-mCCA) secure KEM scheme. Then KEM is MUMC-ECPA (resp., MUMC-ECCA) secure.

Concretely, for any adversary \mathcal{A} that $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, n, Q_e, Q_t)$ -breaks the MUMC-ECPA (resp., MUMC-ECCA) security of KEM and makes at most Q_{total} times of queries in total, there exists an algorithm \mathcal{B} that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$ -breaks the IND-mCPA (resp., IND-mCCA) security of KEM, with

$$t_{\mathcal{B}} \leq t_{\mathcal{A}} + (n + Q_{\text{total}}) \cdot t_{\text{KEM}} \quad \text{and} \quad \epsilon_{\mathcal{B}} \geq \epsilon_{\mathcal{A}} / (2n),$$

where t_{KEM} is a parameter depending only on the algorithms of KEM and is independent of $t_{\mathcal{A}}$.

Proof of Theorem 2. We prove the theorem only for the CCA case, and the CPA case follows similarly by simply ignoring the decapsulation queries. We will define a sequence of games G_0 - G_n , and show adjacent games indistinguishable.

Game G_η , $\eta \in [0, n]$: This game is identical to $\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A})$ (cf. Fig. 2), except that, when answering $\mathcal{O}_{\text{TEST}}(i, c)$, the challenger outputs the real key K_0 for users $i > \eta$ and outputs a random key K_1 for users $i \leq \eta$. Clearly, $G_0 = \text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A})$ where $\beta = 0$ and $G_n = \text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A})$ where $\beta = 1$.

By definition, $\text{Adv}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) = 2 \cdot |\Pr[\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) \Rightarrow 1] - 1/2| = |\Pr[\beta' = 1 \mid \beta = 0] - \Pr[\beta' = 1 \mid \beta = 1]| = |\Pr[G_0 = 1] - \Pr[G_n = 1]| \leq \sum_{\eta \in [n]} |\Pr[G_{\eta-1} = 1] - \Pr[G_\eta = 1]|$. We prove the following lemma.

Lemma 4 ($G_{\eta-1} \rightarrow G_\eta$). For any $\eta \in [n]$, there exist algorithms $\mathcal{B}_1, \mathcal{B}_2$ against the IND-mCCA security, s.t. $t_{\mathcal{B}_1} \approx t_{\mathcal{B}_2} \leq t_{\mathcal{A}} + (n + Q_{\text{total}}) \cdot t_{\text{KEM}}$, and $|\Pr[G_{\eta-1} = 1] - \Pr[G_\eta = 1]| \leq \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_1) + \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_2)$.

Proof of Lemma 4. Note that the only difference between $G_{\eta-1}$ and G_η lies in the $\mathcal{O}_{\text{TEST}}(\eta, c)$ oracle for user η : in $G_{\eta-1}$, $\mathcal{O}_{\text{TEST}}(\eta, c)$ outputs the real key K_0 ; in G_η , $\mathcal{O}_{\text{TEST}}(\eta, c)$ outputs a random key K_1 .

Let corr_η denote the event that \mathcal{A} corrupts user η , i.e., \mathcal{A} ever queries $\mathcal{O}_{\text{CORR}}(\eta)$ when $(\eta, \cdot) \notin \text{TestList}$ and obtains sk_η . In the case that corr_η occurs, η is appended to CorrList , thus \mathcal{A} is not allowed to query $\mathcal{O}_{\text{TEST}}(\eta, c)$, and $G_{\eta-1}$ and G_η are identical. Consequently,

$$|\Pr[G_{\eta-1} = 1] - \Pr[G_\eta = 1]| = |\Pr[G_{\eta-1} = 1 \wedge \neg \text{corr}_\eta] - \Pr[G_\eta = 1 \wedge \neg \text{corr}_\eta]|. \quad (12)$$

To bound (12), we introduce an intermediate game \tilde{G}_η between $G_{\eta-1}$ and G_η , and analyze $|\Pr[G_{\eta-1} = 1 \wedge \neg \text{corr}_\eta] - \Pr[\tilde{G}_\eta = 1 \wedge \neg \text{corr}_\eta]|$ and $|\Pr[\tilde{G}_\eta = 1 \wedge \neg \text{corr}_\eta] - \Pr[G_\eta = 1 \wedge \neg \text{corr}_\eta]|$ separately.

Game \tilde{G}_η , $\eta \in [0, n]$: This game is identical to G_η , except that, when answering $\mathcal{O}_{\text{REV}}(\eta, c)$ for the user η , if $(\eta, c, K) \in \text{EnList}$ and $(\eta, c) \notin \text{TestList}$, the challenger outputs a random key $R \leftarrow_s \mathcal{K}$ (instead of the real key K in $(\eta, c, K) \in \text{EnList}$).

We have the following two claims, with proofs appeared in Appendix B.

Claim 1. $|\Pr[G_{\eta-1} = 1 \wedge \neg \text{corr}_\eta] - \Pr[\tilde{G}_\eta = 1 \wedge \neg \text{corr}_\eta]| \leq \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_1)$.

Claim 2. $|\Pr[\tilde{G}_\eta = 1 \wedge \neg \text{corr}_\eta] - \Pr[G_\eta = 1 \wedge \neg \text{corr}_\eta]| \leq \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_2)$.

Taking (12), Claim 1 and Claim 2 together, we have

$$|\Pr[G_{\eta-1} = 1] - \Pr[G_\eta = 1]| \leq \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_1) + \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_2).$$

This yields Lemma 4. \square

Finally, we get $\text{Adv}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) = |\Pr[G_0 = 1] - \Pr[G_n = 1]|$

$$\leq \sum_{\eta \in [n]} |\Pr[G_{\eta-1} = 1] - \Pr[G_\eta = 1]| \leq n \cdot \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_1) + n \cdot \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_2).$$

By unifying \mathcal{B}_1 and \mathcal{B}_2 into a single algorithm \mathcal{B} attacking the IND-mCCA security, where \mathcal{B} behaves according to \mathcal{B}_1 with probability 1/2 and according to \mathcal{B}_2 with probability 1/2, we have $\text{Adv}_{\text{KEM}, n, Q_e, Q_t}^{\text{mumc-ecca}}(\mathcal{A}) \leq 2n \cdot \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B})$. \blacksquare

Acknowledgments. We would like to thank the reviewers for their helpful comments. Shuai Han and Shengli Liu were partially supported by National Natural Science Foundation of China (Grant Nos. 62002223, 61925207), Guangdong Major Project of Basic and Applied Basic Research (2019B030302008), Shanghai Sailing Program (20YF1421100), and Young Elite Scientists Sponsorship Program by China Association for Science and Technology. Dawu Gu was partially supported by National Key Research and Development Project 2020YFA0712300.

References

- [1] Bader, C., Hofheinz, D., Jager, T., Kiltz, E., Li, Y.: Tightly-secure authenticated key exchange. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part I. LNCS, vol. 9014, pp. 629–658. Springer, Heidelberg (Mar 2015)
- [2] Bader, C., Jager, T., Li, Y., Schäge, S.: On the impossibility of tight cryptographic reductions. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 273–304. Springer, Heidelberg (May 2016)
- [3] Bellare, M., Ristenpart, T.: Simulation without the artificial abort: Simplified proof and improved concrete security for Waters’ IBE scheme. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 407–424. Springer, Heidelberg (Apr 2009)
- [4] Boneh, D., Venkatesan, R.: Breaking RSA may not be equivalent to factoring. In: Nyberg, K. (ed.) EUROCRYPT’98. LNCS, vol. 1403, pp. 59–71. Springer, Heidelberg (May / Jun 1998)
- [5] Canetti, R., Krawczyk, H.: Universally composable notions of key exchange and secure channels. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 337–351. Springer, Heidelberg (Apr / May 2002)
- [6] Cohn-Gordon, K., Cremers, C., Gjøsteen, K., Jacobsen, H., Jager, T.: Highly efficient key exchange protocols with optimal tightness. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 767–797. Springer, Heidelberg (Aug 2019)
- [7] Coron, J.S.: Optimal security proofs for PSS and other signature schemes. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 272–287. Springer, Heidelberg (Apr / May 2002)
- [8] Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO’98. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (Aug 1998)
- [9] Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (Apr / May 2002)
- [10] Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.* 33(1), 167–226 (2003)
- [11] Diffie, W., Hellman, M.E.: New directions in cryptography. *IEEE Trans. Inf. Theory* 22(6), 644–654 (1976)
- [12] ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO’84. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (Aug 1984)
- [13] Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.: An algebraic framework for Diffie-Hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 129–147. Springer, Heidelberg (Aug 2013)
- [14] Gay, R., Hofheinz, D., Kiltz, E., Wee, H.: Tightly CCA-secure encryption without pairings. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 1–27. Springer, Heidelberg (May 2016)
- [15] Gay, R., Hofheinz, D., Kohl, L.: Kurosawa-Desmedt meets tight security. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part III. LNCS, vol. 10403, pp. 133–160. Springer, Heidelberg (Aug 2017)
- [16] Gjøsteen, K., Jager, T.: Practical and tightly-secure digital signatures and authenticated key exchange. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 95–125. Springer, Heidelberg (Aug 2018)
- [17] Han, S., Jager, T., Kiltz, E., Liu, S., Pan, J., Riepel, D., Schäge, S.: Authenticated key exchange and signatures with tight security in the standard model. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 670–700. Springer, Heidelberg, Virtual Event (Aug 2021)
- [18] Han, S., Liu, S., Lyu, L., Gu, D.: Tight leakage-resilient CCA-security from quasi-adaptive hash proof system. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 417–447. Springer, Heidelberg (Aug 2019)
- [19] Hesse, J., Hofheinz, D., Kohl, L.: On tightly secure non-interactive key exchange. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 65–94. Springer, Heidelberg (Aug 2018)
- [20] Hofheinz, D., Jager, T.: Tightly secure signatures and public-key encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 590–607. Springer, Heidelberg (Aug 2012)

- [21] Hofheinz, D., Jager, T., Knapp, E.: Waters signatures with optimal security reduction. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 66–83. Springer, Heidelberg (May 2012)
- [22] Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (Aug 2007)
- [23] Jager, T., Kiltz, E., Riepel, D., Schäge, S.: Tightly-secure authenticated key exchange, revisited. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 117–146. Springer, Heidelberg (Oct 2021)
- [24] Jager, T., Stam, M., Stanley-Oakes, R., Warinschi, B.: Multi-key authenticated encryption with corruptions: Reductions are lossy. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 409–441. Springer, Heidelberg (Nov 2017)
- [25] Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 537–553. Springer, Heidelberg (Apr 2012)
- [26] Krawczyk, H.: HMQV: A high-performance secure Diffie-Hellman protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (Aug 2005)
- [27] Kurosawa, K., Desmedt, Y.: A new paradigm of hybrid encryption scheme. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (Aug 2004)
- [28] LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger security of authenticated key exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (Nov 2007)
- [29] Lewko, A.B., Waters, B.: Why proving HIBE systems secure is difficult. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 58–76. Springer, Heidelberg (May 2014)
- [30] Liu, X., Liu, S., Gu, D., Weng, J.: Two-pass authenticated key exchange with explicit authentication and tight security. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 785–814. Springer, Heidelberg (Dec 2020)
- [31] Morgan, A., Pass, R.: On the security loss of unique signatures. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part I. LNCS, vol. 11239, pp. 507–536. Springer, Heidelberg (Nov 2018)
- [32] Morgan, A., Pass, R., Shi, E.: On the adaptive security of MACs and PRFs. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part I. LNCS, vol. 12491, pp. 724–753. Springer, Heidelberg (Dec 2020)
- [33] Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: 38th FOCS. pp. 458–467. IEEE Computer Society Press (Oct 1997)
- [34] Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. pp. 427–437. ACM Press (May 1990)
- [35] Niehues, D.: Verifiable random functions with optimal tightness. In: Garay, J. (ed.) PKC 2021, Part II. LNCS, vol. 12711, pp. 61–91. Springer, Heidelberg (May 2021)
- [36] Pan, J., Qian, C., Ringerud, M.: Signed Diffie-Hellman key exchange with tight security. In: Paterson, K.G. (ed.) CT-RSA 2021. LNCS, vol. 12704, pp. 201–226. Springer (2021)
- [37] Xue, H., Lu, X., Li, B., Liang, B., He, J.: Understanding and constructing AKE via double-key key encapsulation mechanism. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part II. LNCS, vol. 11273, pp. 158–189. Springer, Heidelberg (Dec 2018)

A Computing Ranks for Well-Known KEM Schemes: More Instances

We show the KEM schemes (implicitly) used in many well-known PKE schemes [12, 8, 27, 14, 18, 34] and elaborate how to compute their ranks.

Before presenting the computations, we show a simple but useful lemma which gives an upper bound on the rank of KEM schemes.

Lemma 5 (Upper Bound on the Rank of KEM). *Let KEM be a KEM scheme with ciphertext space \mathcal{CT} . Let $d \in \mathbb{N}$. Suppose that every set \mathcal{X} containing exactly $d + 1$ ciphertexts is not independent. Then we have $\text{Rank}_{\text{KEM}} \leq d$.*

Proof. According to the definition of Rank_{KEM} , there exists an independent ciphertext subset \mathcal{X} such that $\#\mathcal{X} = \text{Rank}_{\text{KEM}}$ and $\text{EquivSK}(\mathcal{X}) = \text{EquivSK}(\mathcal{CT})$. Suppose towards a contradiction that $\text{Rank}_{\text{KEM}} > d$, then $\#\mathcal{X} \geq d + 1$. By Lemma 2, any $(d + 1)$ -sized subset \mathcal{X}' of \mathcal{X} is also independent. This contradicts to the condition that every ciphertext set of size $d + 1$ is not independent. Therefore, it must hold that $\text{Rank}_{\text{KEM}} \leq d$. \square

Example 1 (ElGamal's CPA-secure KEM). The ElGamal public-key encryption (PKE) [12] is tightly IND-mCPA secure based on the DDH assumption with security loss $O(1)$ [33].

We recall the KEM deriving from the ElGamal PKE [12] as follows. Let (\mathbb{G}, p, g) be a (multiplicative) group of prime order p and with random generator g . The public key is $pk := (g, h)$ where $h := g^s$ for a uniformly chosen $s \leftarrow_s \mathbb{Z}_p$, and the secret key is $sk := s$. The encapsulation algorithm $\text{Encap}(pk)$ samples $r \leftarrow_s \mathbb{Z}_p$ uniformly, and outputs $(c := g^r, K := h^r)$. The decapsulation algorithm $\text{Decap}(sk, c)$ outputs $K := (c)^{sk}$.

The secret key space is $\mathcal{SK} = \mathbb{Z}_p$, the ciphertext space is $\mathcal{CT} = \mathbb{G}$, and in decapsulation the essential key deriving part Decap_{kd} is Decap itself.

Next we show that the ElGamal-KEM has rank 1. For a ciphertext $c \in \mathcal{CT} = \mathbb{G}$, we compute $\text{EquivSK}(c)$: for any $(sk, sk') \in (\mathcal{SK})^2$, $(sk, sk') \in \text{EquivSK}(c) \iff \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \iff (c)^{sk} = (c)^{sk'}$. There are two cases:

- If $c = 1 \in \mathbb{G}$, $(sk, sk') \in \text{EquivSK}(c) \iff 1^{sk} = 1^{sk'} \iff 1 = 1$, which always holds, so $\text{EquivSK}(c) = (\mathbb{Z}_p)^2$, the full relation on \mathcal{SK} .
- If $c \in \mathbb{G} \setminus \{1\}$, $(sk, sk') \in \text{EquivSK}(c) \iff (c)^{sk} = (c)^{sk'} \iff sk = sk'$, so $\text{EquivSK}(c) = \{(sk, sk) \mid sk \in \mathbb{Z}_p\}$, the identity relation on \mathcal{SK} .

Consequently, we have the following two facts.

- (1) For any two ciphertexts $c^{(1)}, c^{(2)} \in \mathcal{CT}$, it must hold that $\text{EquivSK}(c^{(1)}) \subseteq \text{EquivSK}(c^{(2)})$ or $\text{EquivSK}(c^{(1)}) \supseteq \text{EquivSK}(c^{(2)})$, thus $\text{EquivSK}(c^{(1)}, c^{(2)}) = \text{EquivSK}(c^{(1)}) \cap \text{EquivSK}(c^{(2)}) = \text{EquivSK}(c^{(1)})$ or $\text{EquivSK}(c^{(2)})$, which implies that $\{c^{(1)}, c^{(2)}\}$ is not independent.
- (2) $\text{EquivSK}(\mathcal{CT}) = \bigcap_{c \in \mathcal{CT}} \text{EquivSK}(c) = \{(sk, sk) \mid sk \in \mathbb{Z}_p\}$.

Therefore, the largest independent subset $\mathcal{X} \subseteq \mathcal{CT}$ such that $\text{EquivSK}(\mathcal{X}) = \text{EquivSK}(\mathcal{CT})$ is singleton set $\mathcal{X} = \{c\}$ with $c \in \mathbb{G} \setminus \{1\}$. So, $\text{Rank}_{\text{KEM}} = 1$.

Example 2 (Cramer-Shoup's CCA-secure KEM). Cramer and Shoup [8] proposed an efficient IND-CCA secure PKE (CS-PKE) based on the DDH assumption. We recall the KEM deriving from the CS-PKE as follows. Let $(\mathbb{G}, p, g_1, g_2)$ be a group of prime order p and with random generators g_1, g_2 . Let H be a hash function from \mathbb{G}^2 to \mathbb{Z}_p .

- The public key is $pk := (g_1, g_2, c, d, h)$ where $c := g_1^{x_1} g_2^{x_2}$, $d := g_1^{y_1} g_2^{y_2}$ and $h := g_1^z$ for uniformly chosen $x_1, x_2, y_1, y_2, z \leftarrow_s \mathbb{Z}_p$, and the secret key is $sk := (x_1, x_2, y_1, y_2, z)$.
- $\text{Encap}(pk)$ samples $r \leftarrow_s \mathbb{Z}_p$ uniformly, computes $u_1 := g_1^r$, $u_2 := g_2^r$, $\alpha := H(u_1, u_2)$, $v := c^r d^{r\alpha}$, $K := h^r$, and outputs $c := (u_1, u_2, v)$ and K .
- $\text{Decap}(sk = (x_1, x_2, y_1, y_2, z), c = (u_1, u_2, v))$ outputs $K := (u_1)^z$ if $u_1^{x_1 + y_1\alpha} \cdot u_2^{x_2 + y_2\alpha} = v$ holds, where $\alpha := H(u_1, u_2)$, and outputs \perp otherwise.

The secret key space is $\mathcal{SK} = \mathbb{Z}_p^5$, the ciphertext space is $\mathcal{CT} = \mathbb{G}^3$, and the key-derivation part $\text{Decap}_{\text{kd}}(sk, c)$ simply outputs $K := (u_1)^z$. We note that the $\text{Decap}_{\text{kd}}(sk, c)$ is identical to the decapsulation algorithm of ElGamal-KEM. Thus, by a similar analysis, the CS-KEM has rank 1 as well.

Example 3 (Kurosawa-Desmedt’s CCCA-secure KEM). Kurosawa and Desmedt [27] proposed the most efficient IND-CCA secure PKE (KD-PKE) based on the DDH assumption by far. At the heart of their PKE is a more efficient KEM scheme which was later shown to be IND-CCCA (constrained CCA) secure, a security notion weaker than IND-CCA but stronger than IND-CPA, in [22].

We recall the KD-KEM as follows. Let $(\mathbb{G}, p, g_1, g_2)$ be a group of prime order p and with random generators g_1, g_2 . Let H be a hash function from \mathbb{G}^2 to \mathbb{Z}_p .

- The public key is $pk := (g_1, g_2, c, d)$ where $c := g_1^{x_1} g_2^{x_2}$ and $d := g_1^{y_1} g_2^{y_2}$ for uniformly chosen $x_1, x_2, y_1, y_2 \leftarrow_s \mathbb{Z}_p$, and the secret key is $sk := (x_1, x_2, y_1, y_2)$.
- $\text{Encap}(pk)$ samples $r \leftarrow_s \mathbb{Z}_p$ uniformly, computes $u_1 := g_1^r$, $u_2 := g_2^r$, $\alpha := H(u_1, u_2)$, $K := c^r d^{r\alpha}$, and outputs $c := (u_1, u_2)$ and K .
- $\text{Decap}(sk = (x_1, x_2, y_1, y_2), c = (u_1, u_2))$ outputs $K := u_1^{x_1+y_1\alpha} \cdot u_2^{x_2+y_2\alpha}$, where $\alpha := H(u_1, u_2)$.

The secret key space is $\mathcal{SK} = \mathbb{Z}_p^4$, the ciphertext space is $\mathcal{CT} = \mathbb{G}^2$, and the key-derivation part Decap_{kd} is Decap itself.

Firstly, we compute $\text{EquivSK}(c)$ for any ciphertext $c = (u_1, u_2) \in \mathcal{CT}$. We can always write $u_1 = g_1^{r_1}$ and $u_2 = g_1^{r_2}$ with $r_1, r_2 \in \mathbb{Z}_p$ the discrete logarithms of u_1, u_2 w.r.t. g_1 . Let $\alpha := H(u_1, u_2)$. Then for any $sk = (x_1, x_2, y_1, y_2), sk' = (x'_1, x'_2, y'_1, y'_2) \in \mathcal{SK}$,

$$\begin{aligned} (sk, sk') \in \text{EquivSK}(c) &\iff \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \\ &\iff u_1^{x_1+y_1\alpha} \cdot u_2^{x_2+y_2\alpha} = u_1^{x'_1+y'_1\alpha} \cdot u_2^{x'_2+y'_2\alpha} \\ &\iff r_1 \cdot (x_1 + y_1\alpha) + r_2 \cdot (x_2 + y_2\alpha) = r_1 \cdot (x'_1 + y'_1\alpha) + r_2 \cdot (x'_2 + y'_2\alpha) \\ &\iff (r_1, r_2, r_1\alpha, r_2\alpha) \cdot (x_1, x_2, y_1, y_2)^\top = (r_1, r_2, r_1\alpha, r_2\alpha) \cdot (x'_1, x'_2, y'_1, y'_2)^\top. \end{aligned}$$

So, $\text{EquivSK}(c) = \{(sk = (x_1, x_2, y_1, y_2), sk' = (x'_1, x'_2, y'_1, y'_2)) \mid (r_1, r_2, r_1\alpha, r_2\alpha) \cdot (x_1, x_2, y_1, y_2)^\top = (r_1, r_2, r_1\alpha, r_2\alpha) \cdot (x'_1, x'_2, y'_1, y'_2)^\top\}$.

Next, we show that any subset consisting any five ciphertexts $\{c^{(1)}, \dots, c^{(5)}\}$ is not an independent set. Let $c^{(i)} = (u_1^{(i)} = g_1^{r_1^{(i)}}, u_2^{(i)} = g_1^{r_2^{(i)}})$ and $\alpha^{(i)} := H(u_1^{(i)}, u_2^{(i)})$ for $i \in [5]$. Since the linear space \mathbb{Z}_p^4 has dimension 4, the five vectors $\{(r_1^{(i)}, r_2^{(i)}, r_1^{(i)}\alpha^{(i)}, r_2^{(i)}\alpha^{(i)})\}_{i \in [5]}$ in \mathbb{Z}_p^4 must be linearly dependent. Say $(r_1^{(5)}, r_2^{(5)}, r_1^{(5)}\alpha^{(5)}, r_2^{(5)}\alpha^{(5)}) = \sum_{j=1}^4 a^{(j)} \circ (r_1^{(j)}, r_2^{(j)}, r_1^{(j)}\alpha^{(j)}, r_2^{(j)}\alpha^{(j)})$ for some coefficients $a^{(1)}, \dots, a^{(4)} \in \mathbb{Z}_p$, where “ \circ ” denotes scalar multiplication. Then we prove that $\text{EquivSK}(c^{(1)}, \dots, c^{(4)}) \subseteq \text{EquivSK}(c^{(5)})$ as follows: for any $(sk = (x_1, x_2, y_1, y_2), sk' = (x'_1, x'_2, y'_1, y'_2)) \in \text{EquivSK}(c^{(1)}, \dots, c^{(4)}) = \bigcap_{j=1}^4 \text{EquivSK}(c^{(j)})$, it holds $(r_1^{(j)}, r_2^{(j)}, r_1^{(j)}\alpha^{(j)}, r_2^{(j)}\alpha^{(j)}) \cdot (x_1, x_2, y_1, y_2)^\top = (r_1^{(j)}, r_2^{(j)}, r_1^{(j)}\alpha^{(j)}, r_2^{(j)}\alpha^{(j)}) \cdot (x'_1, x'_2, y'_1, y'_2)^\top$ for all $j \in [4]$, thus

$$\begin{aligned} &(r_1^{(5)}, r_2^{(5)}, r_1^{(5)}\alpha^{(5)}, r_2^{(5)}\alpha^{(5)}) \cdot (x_1, x_2, y_1, y_2)^\top \\ &= \sum_{j=1}^4 a^{(j)} \circ (r_1^{(j)}, r_2^{(j)}, r_1^{(j)}\alpha^{(j)}, r_2^{(j)}\alpha^{(j)}) \cdot (x_1, x_2, y_1, y_2)^\top \\ &= \sum_{j=1}^4 a^{(j)} \circ (r_1^{(j)}, r_2^{(j)}, r_1^{(j)}\alpha^{(j)}, r_2^{(j)}\alpha^{(j)}) \cdot (x'_1, x'_2, y'_1, y'_2)^\top \\ &= (r_1^{(5)}, r_2^{(5)}, r_1^{(5)}\alpha^{(5)}, r_2^{(5)}\alpha^{(5)}) \cdot (x'_1, x'_2, y'_1, y'_2)^\top, \end{aligned}$$

so $(sk, sk') \in \text{EquivSK}(c^{(5)})$. The fact that $\text{EquivSK}(c^{(1)}, \dots, c^{(4)}) \subseteq \text{EquivSK}(c^{(5)})$ implies $\text{EquivSK}(c^{(1)}, \dots, c^{(5)}) = \text{EquivSK}(c^{(1)}, \dots, c^{(4)}) \cap \text{EquivSK}(c^{(5)}) = \text{EquivSK}(c^{(1)}, \dots, c^{(4)})$. Therefore, $\{c^{(1)}, \dots, c^{(5)}\}$ is not independent for any five ciphertexts.

Finally, by Lemma 5, the KD-KEM has rank at most 4.

Example 4 (Gay-Hofheinz-Kiltz-Wee’s PCA-secure KEM). In 2016, Gay, Hofheinz, Kiltz and Wee [14] proposed the first (almost) tightly IND-mCCA secure PKE (GHKW-PKE) based on the matrix DDH (MDDH) assumptions [13] and without pairing. At the heart of their PKE is a KEM scheme which is (almost) tightly PCA-secure (plaintext check attacks), a security strictly weaker than IND-CCA but stronger than IND-CPA.

We recall the GHKW-KEM as follows. Let (\mathbb{G}, p, g) be a group of prime order p and with random generators g . We use the implicit representation of elements in \mathbb{G} according to [13].

For $a \in \mathbb{Z}_p$, define $[a] := g^a \in \mathbb{G}$. More generally, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_p^{n \times m}$, define $[\mathbf{A}] := (g^{a_{ij}}) \in \mathbb{G}^{n \times m}$. Obviously, given $[a], [b] \in \mathbb{G}$ and a scalar $x \in \mathbb{Z}_p$, one can efficiently compute $[ax] \in \mathbb{G}$ and $[a + b] \in \mathbb{G}$. Similarly, for $\mathbf{A} \in \mathbb{Z}_p^{m \times n}, \mathbf{B} \in \mathbb{Z}_p^{n \times t}$, given \mathbf{A}, \mathbf{B} or $[\mathbf{A}], [\mathbf{B}]$ or $\mathbf{A}, [\mathbf{B}]$, one can efficiently compute $[\mathbf{AB}] \in \mathbb{G}^{m \times t}$. Let $k \in \mathbb{N}$ denote the parameter of MDDH assumption. E.g, $k = 1$ corresponds to the DDH assumption. Let H be a hash function from \mathbb{G}^k to $\{0, 1\}^\lambda$.

- The public key is $pk := \left([\mathbf{M}], ([\mathbf{M}^\top \mathbf{k}_{j,\beta}])_{1 \leq j \leq \lambda, \beta \in \{0,1\}} \right)$ where $\mathbf{M} \leftarrow_s \mathbb{Z}_p^{3k \times k}$ and $\mathbf{k}_{j,\beta} \leftarrow_s \mathbb{Z}_p^{3k}$, and the secret key is $sk := (\mathbf{k}_{j,\beta})_{1 \leq j \leq \lambda, \beta \in \{0,1\}}$.
- $\text{Encap}(pk)$ samples $\mathbf{r} \leftarrow_s \mathbb{Z}_p^k$ uniformly, computes $[\mathbf{y}] := [\mathbf{M}\mathbf{r}] \in \mathbb{G}^{3k}$, $\tau := H([\bar{\mathbf{y}}]) \in \{0, 1\}^\lambda$ with $\bar{\mathbf{y}}$ the first k entries of \mathbf{y} , $K := [\mathbf{r}^\top \mathbf{M}^\top \sum_{j=1}^\lambda \mathbf{k}_{j,\tau_j}]$ with τ_j the j -th bit of τ , and outputs $c := [\mathbf{y}]$ and K .
- $\text{Decap}(sk, c = [\mathbf{y}])$ computes $\tau := H([\bar{\mathbf{y}}]) \in \{0, 1\}^\lambda$, and outputs $K := [\mathbf{y}^\top \sum_{j=1}^\lambda \mathbf{k}_{j,\tau_j}]$.

The secret key space is $\mathcal{SK} = (\mathbb{Z}_p^{3k})^{2\lambda}$, the ciphertext space is $\mathcal{CT} = \mathbb{G}^{3k}$, and the key-derivation part Decap_{kd} is Decap itself.

Firstly, we compute $\text{EquivSK}(c)$ for any ciphertext $c = [\mathbf{y}] \in \mathcal{CT}$. Let $\tau := H([\bar{\mathbf{y}}]) \in \{0, 1\}^\lambda$. Then for any $sk = (\mathbf{k}_{j,\beta})_{1 \leq j \leq \lambda, \beta \in \{0,1\}}, sk' = (\mathbf{k}'_{j,\beta})_{1 \leq j \leq \lambda, \beta \in \{0,1\}} \in \mathcal{SK}$, $(sk, sk') \in \text{EquivSK}(c) \iff \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \iff \mathbf{y}^\top \sum_{j=1}^\lambda \mathbf{k}_{j,\tau_j} = \mathbf{y}^\top \sum_{j=1}^\lambda \mathbf{k}'_{j,\tau_j}$. For the ease of notations, we let

$$\mathbf{v}_c^\top := ((1 - \tau_1)\mathbf{y}^\top, \tau_1\mathbf{y}^\top, (1 - \tau_2)\mathbf{y}^\top, \tau_2\mathbf{y}^\top, \dots, (1 - \tau_\lambda)\mathbf{y}^\top, \tau_\lambda\mathbf{y}^\top) \in \mathbb{Z}_p^{1 \times 6k\lambda}$$

be a row vector determined by $c = [\mathbf{y}] \in \mathcal{CT}$, and let

$$\mathbf{u}_{sk} := (\mathbf{k}_{1,0}^\top, \mathbf{k}_{1,1}^\top, \mathbf{k}_{2,0}^\top, \mathbf{k}_{2,1}^\top, \dots, \mathbf{k}_{\lambda,0}^\top, \mathbf{k}_{\lambda,1}^\top)^\top \in \mathbb{Z}_p^{6k\lambda}$$

be a column vector determined by $sk \in \mathcal{SK}$. Note that with such notations, we have $\mathbf{y}^\top \sum_{j=1}^\lambda \mathbf{k}_{j,\tau_j} = \mathbf{v}_c^\top \mathbf{u}_{sk}$. So, $\text{EquivSK}(c) = \{(sk, sk') \mid \mathbf{v}_c^\top \mathbf{u}_{sk} = \mathbf{v}_c^\top \mathbf{u}_{sk'}\}$.

Next, we show that any subset consisting $6k\lambda + 1$ ciphertexts $\{c^{(i)}\}_{1 \leq i \leq 6k\lambda + 1}$ is not an independent set. Let $\mathbf{v}_{c^{(i)}}^\top \in \mathbb{Z}_p^{1 \times 6k\lambda}$ be the row vector determined by $c^{(i)}$ for $1 \leq i \leq 6k\lambda + 1$. Since the linear space $\mathbb{Z}_p^{1 \times 6k\lambda}$ has dimension $6k\lambda$, the $6k\lambda + 1$ row vectors $\{\mathbf{v}_{c^{(i)}}^\top\}_{1 \leq i \leq 6k\lambda + 1}$ in $\mathbb{Z}_p^{1 \times 6k\lambda}$ must be linearly dependent. Say $\mathbf{v}_{c^{(6k\lambda+1)}}^\top = \sum_{j=1}^{6k\lambda} a^{(j)} \circ \mathbf{v}_{c^{(j)}}^\top$ for some coefficients $a^{(1)}, \dots, a^{(6k\lambda)} \in \mathbb{Z}_p$, where “ \circ ” denotes scalar multiplication. Then we prove that $\text{EquivSK}(c^{(1)}, \dots, c^{(6k\lambda)}) \subseteq \text{EquivSK}(c^{(6k\lambda+1)})$ as follows: for any $(sk, sk') \in \text{EquivSK}(c^{(1)}, \dots, c^{(6k\lambda)}) = \bigcap_{j=1}^{6k\lambda} \text{EquivSK}(c^{(j)})$, it holds $\mathbf{v}_{c^{(j)}}^\top \mathbf{u}_{sk} = \mathbf{v}_{c^{(j)}}^\top \mathbf{u}_{sk'}$ for all $1 \leq j \leq 6k\lambda$, thus

$$\mathbf{v}_{c^{(6k\lambda+1)}}^\top \mathbf{u}_{sk} = \sum_{j=1}^{6k\lambda} a^{(j)} \circ \mathbf{v}_{c^{(j)}}^\top \mathbf{u}_{sk} = \sum_{j=1}^{6k\lambda} a^{(j)} \circ \mathbf{v}_{c^{(j)}}^\top \mathbf{u}_{sk'} = \mathbf{v}_{c^{(6k\lambda+1)}}^\top \mathbf{u}_{sk'},$$

so $(sk, sk') \in \text{EquivSK}(c^{(6k\lambda+1)})$. The fact that $\text{EquivSK}(c^{(1)}, \dots, c^{(6k\lambda)}) \subseteq \text{EquivSK}(c^{(6k\lambda+1)})$ implies $\text{EquivSK}(c^{(1)}, \dots, c^{(6k\lambda)}, c^{(6k\lambda+1)}) = \text{EquivSK}(c^{(1)}, \dots, c^{(6k\lambda)}) \cap \text{EquivSK}(c^{(6k\lambda+1)}) = \text{EquivSK}(c^{(1)}, \dots, c^{(6k\lambda)})$. Therefore, $\{c^{(1)}, \dots, c^{(6k\lambda)}, c^{(6k\lambda+1)}\}$ is not independent for any $6k\lambda + 1$ ciphertexts.

Finally, by Lemma 5, the GHKW-KEM has rank at most $6k\lambda$.

Example 5 (Han-Liu-Lyu-Gu’s CCA-secure KEM). Han, Liu, Lyu and Gu [18] presented an (almost) tightly leakage-resilient mCCA secure PKE (HLLG-PKE) based on the matrix DDH (MDDH) assumptions [13] over pairing-friendly groups. Their PKE contains a KEM scheme which is also (almost) tightly leakage-resilient mCCA secure.

We recall the HLLG-KEM over symmetric pairing group as follows. Let $(\mathbb{G}, \mathbb{G}_T, p, g, g_T, e)$ be a description of symmetric pairing group, where \mathbb{G}, \mathbb{G}_T are groups of prime order p , $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a non-degenerated bilinear pairing, and g, g_T are generators of \mathbb{G}, \mathbb{G}_T respectively with $g_T := e(g, g)$. We use the implicit representations $[\cdot]$ and $[\cdot]_T$ to denote elements in \mathbb{G} and \mathbb{G}_T respectively, according to [13]. Let $k \in \mathbb{N}$ denote the parameter of MDDH assumption. Let H be a hash function from \mathbb{G}^{2k} to \mathbb{G} .

- The public key is $pk := \left([\mathbf{A}], [\mathbf{k}^\top \mathbf{A}], [\mathbf{A}^\top \widehat{\mathbf{K}} \mathbf{A}]_T, [\widehat{\mathbf{K}} \mathbf{A}] \right)$ where $\mathbf{A} \leftarrow_s \mathbb{Z}_p^{2k \times k}$, $\mathbf{k} \leftarrow_s \mathbb{Z}_p^{2k}$, $\widehat{\mathbf{K}} \leftarrow_s \mathbb{Z}_p^{2k \times 2k}$, $\widetilde{\mathbf{K}} \leftarrow_s \mathbb{Z}_p^{2 \times 2k}$, and the secret key is $sk := (\mathbf{k}, \widehat{\mathbf{K}}, \widetilde{\mathbf{K}})$.

- $\text{Encap}(pk)$ samples $\mathbf{r} \leftarrow_s \mathbb{Z}_p^k$ uniformly, computes $[\mathbf{y}] := [\mathbf{A}\mathbf{r}] \in \mathbb{G}^{2k}$, $[\tau] := H([\mathbf{y}]) \in \mathbb{G}$, $[\pi]_T := [\mathbf{r}^\top \mathbf{A}^\top \widehat{\mathbf{K}}\mathbf{A}\mathbf{r} + (1, \tau)\widehat{\mathbf{K}}\mathbf{A}\mathbf{r}]_T$, $K := [\mathbf{k}^\top \mathbf{A}\mathbf{r}]$, and outputs $c := ([\mathbf{y}], [\pi]_T)$ and K .
- $\text{Decap}(sk, c = ([\mathbf{y}], [\pi]_T))$ computes $[\tau] := H([\mathbf{y}]) \in \mathbb{G}$, outputs $K := [\mathbf{k}^\top \mathbf{y}]$ if $[\pi]_T := [\mathbf{y}^\top \widehat{\mathbf{K}}\mathbf{y} + (1, \tau)\widehat{\mathbf{K}}\mathbf{y}]_T$ holds, and output \perp otherwise.

The secret key space is $\mathcal{SK} = \mathbb{Z}_p^{2k} \times \mathbb{Z}_p^{2k \times 2k} \times \mathbb{Z}_p^{2k \times 2k}$, the ciphertext space is $\mathcal{CT} = \mathbb{G}^{2k} \times \mathbb{G}_T$, and the key-derivation part Decap_{kd} simply outputs $K := [\mathbf{k}^\top \mathbf{y}]$.

Firstly, we compute $\text{EquivSK}(c)$ for any ciphertext $c = ([\mathbf{y}], [\pi]_T) \in \mathcal{CT}$. For any $sk = (\mathbf{k}, \widehat{\mathbf{K}}, \widetilde{\mathbf{K}})$, $sk' = (\mathbf{k}', \widehat{\mathbf{K}}', \widetilde{\mathbf{K}}') \in \mathcal{SK}$, $(sk, sk') \in \text{EquivSK}(c) \iff \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \iff \mathbf{k}^\top \mathbf{y} = \mathbf{k}'^\top \mathbf{y}$. So, $\text{EquivSK}(c) = \{(sk = (\mathbf{k}, \dots), sk' = (\mathbf{k}', \dots)) \mid \mathbf{k}^\top \mathbf{y} = \mathbf{k}'^\top \mathbf{y}\}$.

Next, we show that any subset consisting $2k+1$ ciphertexts $\{c^{(i)}\}_{1 \leq i \leq 2k+1}$ is not an independent set. Let $c^{(i)} = ([\mathbf{y}^{(i)}], [\pi^{(i)}]_T)$ for $1 \leq i \leq 2k+1$. Since the linear space \mathbb{Z}_p^{2k} has dimension $2k$, the $2k+1$ vectors $\{\mathbf{y}^{(i)}\}_{1 \leq i \leq 2k+1}$ in \mathbb{Z}_p^{2k} must be linearly dependent. Say $\mathbf{y}^{(2k+1)} = \sum_{j=1}^{2k} a^{(j)} \circ \mathbf{y}^{(j)}$ for some coefficients $a^{(1)}, \dots, a^{(2k)} \in \mathbb{Z}_p$, where “ \circ ” denotes scalar multiplication. Then we prove that $\text{EquivSK}(c^{(1)}, \dots, c^{(2k)}) \subseteq \text{EquivSK}(c^{(2k+1)})$ as follows: for any $(sk = (\mathbf{k}, \dots), sk' = (\mathbf{k}', \dots)) \in \text{EquivSK}(c^{(1)}, \dots, c^{(2k)}) = \bigcap_{j=1}^{2k} \text{EquivSK}(c^{(j)})$, it holds $\mathbf{k}^\top \mathbf{y}^{(j)} = \mathbf{k}'^\top \mathbf{y}^{(j)}$ for all $1 \leq j \leq 2k$, thus

$$\mathbf{k}^\top \mathbf{y}^{(2k+1)} = \sum_{j=1}^{2k} a^{(j)} \circ \mathbf{k}^\top \mathbf{y}^{(j)} = \sum_{j=1}^{2k} a^{(j)} \circ \mathbf{k}'^\top \mathbf{y}^{(j)} = \mathbf{k}'^\top \mathbf{y}^{(2k+1)},$$

so $(sk, sk') \in \text{EquivSK}(c^{(2k+1)})$. The fact that $\text{EquivSK}(c^{(1)}, \dots, c^{(2k)}) \subseteq \text{EquivSK}(c^{(2k+1)})$ implies $\text{EquivSK}(c^{(1)}, \dots, c^{(2k)}, c^{(2k+1)}) = \text{EquivSK}(c^{(1)}, \dots, c^{(2k)}) \cap \text{EquivSK}(c^{(2k+1)}) = \text{EquivSK}(c^{(1)}, \dots, c^{(2k)})$. Therefore, $\{c^{(1)}, \dots, c^{(2k)}, c^{(2k+1)}\}$ is not independent for any $2k+1$ ciphertexts.

Finally, by Lemma 5, the HLLG-KEM has rank at most $2k$.

Example 6 (Naor-Yung’s CCA-secure KEM). The celebrated Naor-Yung [34] paradigm is one of the principal techniques to construct CCA-secure PKE/KEM. Here we recall the Naor-Yung paradigm in the setting of KEM, which transforms a CPA-secure KEM and a non-interactive zero-knowledge proof/argument (NIZK) scheme to a CCA-secure KEM that we call NY-KEM.

We recall the NY-KEM as follows. Let $\text{KEM}_{\text{cpa}} = (\text{Setup}_{\text{cpa}}, \text{Gen}_{\text{cpa}}, \text{Encap}_{\text{cpa}}, \text{Decap}_{\text{cpa}})$ be a CPA-secure KEM with secret key space $\mathcal{SK}_{\text{cpa}}$, ciphertext space $\mathcal{CT}_{\text{cpa}}$ and encapsulated key space $\{0, 1\}^\lambda$, and let NIZK be a suitable NIZK scheme with proof space Π .

- The public key is $pk := (pk_{\text{cpa},0}, pk_{\text{cpa},1})$ and the secret key is $sk := sk_{\text{cpa},0}$, where $(pk_{\text{cpa},0}, sk_{\text{cpa},0})$ and $(pk_{\text{cpa},1}, sk_{\text{cpa},1})$ are two public/secret key pairs of the underlying KEM_{cpa} .
- $\text{Encap}(pk)$ samples $K \leftarrow_s \{0, 1\}^\lambda$ uniformly, invokes $(c_{\text{cpa},0}, K_{\text{cpa},0}) \leftarrow_s \text{Encap}_{\text{cpa}}(pk_{\text{cpa},0})$ and $(c_{\text{cpa},1}, K_{\text{cpa},1}) \leftarrow_s \text{Encap}_{\text{cpa}}(pk_{\text{cpa},1})$, computes $K_0 := K \oplus K_{\text{cpa},0}$ and $K_1 := K \oplus K_{\text{cpa},1}$, generates a NIZK proof $\pi \in \Pi$, and outputs $c := (c_{\text{cpa},0}, K_0, c_{\text{cpa},1}, K_1, \pi)$ and K . Here \oplus denotes the bitwise XOR operation.
- $\text{Decap}(sk, c = (c_{\text{cpa},0}, K_0, c_{\text{cpa},1}, K_1, \pi))$ invokes $K_{\text{cpa},0} \leftarrow \text{Decap}_{\text{cpa}}(sk_{\text{cpa},0}, c_{\text{cpa},0})$, outputs $K := K_{\text{cpa},0} \oplus K_0$ if the NIZK proof π is valid, and output \perp otherwise.

The secret key space is $\mathcal{SK} = \mathcal{SK}_{\text{cpa}}$, the ciphertext space is $\mathcal{CT} = \mathcal{CT}_{\text{cpa}} \times \{0, 1\}^\lambda \times \mathcal{CT}_{\text{cpa}} \times \{0, 1\}^\lambda \times \Pi$, and the key-derivation part Decap_{kd} simply outputs $K := \text{Decap}_{\text{cpa}}(sk_{\text{cpa},0}, c_{\text{cpa},0}) \oplus K_0$.

We compute $\text{EquivSK}(c)$ for any ciphertext $c = (c_{\text{cpa},0}, K_0, c_{\text{cpa},1}, K_1, \pi) \in \mathcal{CT}$. For any $sk = sk_{\text{cpa},0}$, $sk' = sk'_{\text{cpa},0} \in \mathcal{SK}$, $(sk, sk') \in \text{EquivSK}(c) \iff \text{Decap}_{\text{kd}}(sk, c) = \text{Decap}_{\text{kd}}(sk', c) \iff \text{Decap}_{\text{cpa}}(sk_{\text{cpa},0}, c_{\text{cpa},0}) \oplus K_0 = \text{Decap}_{\text{cpa}}(sk'_{\text{cpa},0}, c_{\text{cpa},0}) \oplus K_0 \iff \text{Decap}_{\text{cpa}}(sk_{\text{cpa},0}, c_{\text{cpa},0}) = \text{Decap}_{\text{cpa}}(sk'_{\text{cpa},0}, c_{\text{cpa},0}) \iff (sk_{\text{cpa},0}, sk'_{\text{cpa},0}) \in \text{EquivSK}_{\text{cpa}}(c_{\text{cpa},0})$, where $\text{EquivSK}_{\text{cpa}}$ denotes the decapsulation equivalence relation of KEM_{cpa} . So,

$$\text{EquivSK}(c) = \text{EquivSK}_{\text{cpa}}(c_{\text{cpa},0}).$$

This shows that the relation EquivSK of NY-KEM is essentially the same as the relation $\text{EquivSK}_{\text{cpa}}$ of KEM_{cpa} , by simply mapping a ciphertext $c = (c_{\text{cpa},0}, K_0, c_{\text{cpa},1}, K_1, \pi)$ of NY-KEM to a ciphertext $c_{\text{cpa},0}$ of KEM_{cpa} . Thus the CCA-secure NY-KEM and the underlying KEM_{cpa} have essentially the same meaning of set independence, set rank, and consequently, *the same rank*.

Therefore, the CCA-secure NY-KEM has polynomially-bounded rank, as long as the underlying CPA-KEM scheme KEM_{cpa} does. This suggests that many concrete instantiations of NY-KEM have polynomially-bounded rank, e.g., the NY-KEMs whose underlying CPA-secure KEMs are instantiated with the KEMs shown above (such as ElGamal).

B Omitted Proofs

B.1 Proof of Claim 1

The only difference between $\mathsf{G}_{\eta-1}$ and $\tilde{\mathsf{G}}_\eta$ lies in the $\mathcal{O}_{\text{TEST}}(\eta, c)$ oracle and $\mathcal{O}_{\text{REV}}(\eta, c)$ for the user η : in $\mathsf{G}_{\eta-1}$, $\mathcal{O}_{\text{TEST}}(\eta, c)$ outputs the real key K_0 and $\mathcal{O}_{\text{REV}}(\eta, c)$ outputs the real key K ; in $\tilde{\mathsf{G}}_\eta$, $\mathcal{O}_{\text{TEST}}(\eta, c)$ outputs a random key K_1 and $\mathcal{O}_{\text{REV}}(\eta, c)$ outputs a random key R .

We construct \mathcal{B}_1 against the IND-mCCA security of KEM by invoking \mathcal{A} . Let b denote the challenge bit chosen by \mathcal{B}_1 's challenger. \mathcal{B}_1 has access to two oracles $\mathcal{O}_{\text{ENC}}()$ and $\mathcal{O}_{\text{DEC}}(\cdot)$ (cf. Fig. 1). Given $(\text{pp}_{\text{KEM}}, pk)$ as input, \mathcal{B}_1 generates $(pk_i, sk_i) \leftarrow \text{Gen}(\text{pp}_{\text{KEM}})$ for all $i \in [n \setminus \eta]$, sets $pk_\eta := pk$, and sends $(\text{pp}_{\text{KEM}}, \text{PKList} := \{pk_i\}_{i \in [n]})$ to \mathcal{A} . \mathcal{B}_1 initializes a flag $\text{corr}_\eta := \text{false}$. Then \mathcal{B}_1 answers the oracle queries $\mathcal{O}_{\text{ENC}}(\cdot)$, $\mathcal{O}_{\text{DEC}}(\cdot, \cdot)$, $\mathcal{O}_{\text{REV}}(\cdot, \cdot)$, $\mathcal{O}_{\text{CORR}}(\cdot)$, $\mathcal{O}_{\text{TEST}}(\cdot, \cdot)$ made by \mathcal{A} in the following way.

For oracle queries $\mathcal{O}_{\text{ENC}}(i)$, $\mathcal{O}_{\text{DEC}}(i, c')$, $\mathcal{O}_{\text{REV}}(i, c)$, $\mathcal{O}_{\text{CORR}}(i)$, $\mathcal{O}_{\text{TEST}}(i, c)$ where $i \neq \eta$, \mathcal{B}_1 prepares the responses exactly the same as $\mathsf{G}_{\eta-1}$ and $\tilde{\mathsf{G}}_\eta$ using (pk_i, sk_i) , and returns the answers to \mathcal{A} . In particular, for $\mathcal{O}_{\text{TEST}}(i, c)$ query where $i \neq \eta$, \mathcal{B}_1 outputs K_0 if $i > \eta$ and outputs K_1 if $i < \eta$.

Below we show how \mathcal{B}_1 answers the oracle queries for $i = \eta$.

- For query $\mathcal{O}_{\text{ENC}}(\eta)$, \mathcal{B}_1 queries its own oracle $\mathcal{O}_{\text{ENC}}()$, obtains (c, K_b) , appends (η, c, K_b) to EnclList , and returns c to \mathcal{A} . Clearly, \mathcal{B}_1 simulates oracle $\mathcal{O}_{\text{ENC}}(\eta)$ in $\text{Exp}_{\text{KEM}, n, Q_\epsilon, Q_t}^{\text{muc-ecca}}$ (thus $\mathsf{G}_{\eta-1}$ and $\tilde{\mathsf{G}}_\eta$) perfectly for \mathcal{A} .
- For query $\mathcal{O}_{\text{DEC}}(\eta, c')$, \mathcal{B}_1 returns \perp to \mathcal{A} if $(\eta, c', \cdot) \in \text{EnclList}$. Otherwise, \mathcal{B}_1 queries its own oracle $\mathcal{O}_{\text{DEC}}(c')$, obtains K' , and returns K' to \mathcal{A} . Clearly, \mathcal{B}_1 simulates oracle $\mathcal{O}_{\text{DEC}}(\eta, c')$ in $\text{Exp}_{\text{KEM}, n, Q_\epsilon, Q_t}^{\text{muc-ecca}}$ (thus $\mathsf{G}_{\eta-1}$ and $\tilde{\mathsf{G}}_\eta$) perfectly for \mathcal{A} .
- For query $\mathcal{O}_{\text{REV}}(\eta, c)$, if $(\eta, c, K) \in \text{EnclList}$ for some $K = K_b$ and $(\eta, c) \notin \text{TestList}$, \mathcal{B}_1 appends (η, c) to RevList , and returns K_b to \mathcal{A} . Otherwise, \mathcal{B}_1 returns \perp to \mathcal{A} .

Note that, in the case $b = 0$, $K_b = K_0$ is the real key encapsulated in c , thus \mathcal{B}_1 simulates oracle $\mathcal{O}_{\text{REV}}(\eta, c)$ in $\mathsf{G}_{\eta-1}$ perfectly for \mathcal{A} ; in the case $b = 1$, $K_b = K_1$ is a random key uniformly chosen from \mathcal{K} , thus \mathcal{B}_1 simulates oracle $\mathcal{O}_{\text{REV}}(\eta, c)$ in $\tilde{\mathsf{G}}_\eta$ perfectly for \mathcal{A} .

- For query $\mathcal{O}_{\text{CORR}}(\eta)$, if $(\eta, \cdot) \in \text{TestList}$, \mathcal{B}_1 returns \perp to \mathcal{A} . Otherwise, \mathcal{B}_1 sets $\text{corr}_\eta := \text{true}$, terminates the interaction with \mathcal{A} , and aborts the game.
- For query $\mathcal{O}_{\text{TEST}}(\eta, c)$, if $(\eta, c, K) \in \text{EnclList}$ for some $K = K_b$ and $(\eta, c) \notin \text{RevList} \cup \text{TestList}$ and $\eta \notin \text{CorrList}$, \mathcal{B}_1 appends (η, c) to TestList , and returns K_b to \mathcal{A} . Otherwise, \mathcal{B}_1 returns \perp to \mathcal{A} .

Note that, in the case $b = 0$, $K_b = K_0$ is the real key encapsulated in c , thus \mathcal{B}_1 simulates oracle $\mathcal{O}_{\text{TEST}}(\eta, c)$ in $\mathsf{G}_{\eta-1}$ perfectly for \mathcal{A} ; in the case $b = 1$, $K_b = K_1$ is a random key uniformly chosen from \mathcal{K} , thus \mathcal{B}_1 simulates oracle $\mathcal{O}_{\text{TEST}}(\eta, c)$ in $\tilde{\mathsf{G}}_\eta$ perfectly for \mathcal{A} .

Finally, \mathcal{B}_1 receives a guessing bit from \mathcal{A} . \mathcal{B}_1 outputs 1 to its own challenger if and only if \mathcal{A} outputs 1 and $\neg \text{corr}_\eta$. Clearly, $t_{\mathcal{B}_1} \leq t_{\mathcal{A}} + (n + Q_{\text{total}}) \cdot t_{\text{KEM}}$.

For the game simulated by \mathcal{B}_1 ,

- In the case $b = 0$, \mathcal{B}_1 simulates $\mathsf{G}_{\eta-1}$ perfectly for \mathcal{A} unless corr_η occurs.
- In the case $b = 1$, \mathcal{B}_1 simulates $\tilde{\mathsf{G}}_\eta$ perfectly for \mathcal{A} unless corr_η occurs.

Therefore, we have

$$\begin{aligned} \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_1) &= 2|\Pr[\mathcal{B}_1 \Rightarrow b] - 1/2| = |\Pr[\mathcal{B}_1 \Rightarrow 1 | b = 0] - \Pr[\mathcal{B}_1 \Rightarrow 1 | b = 1]| \\ &= |\Pr[\mathcal{A} \Rightarrow 1 \wedge \neg \text{corr}_\eta | b = 0] - \Pr[\mathcal{A} \Rightarrow 1 \wedge \neg \text{corr}_\eta | b = 1]| \\ &= |\Pr[\mathsf{G}_{\eta-1} = 1 \wedge \neg \text{corr}_\eta] - \Pr[\tilde{\mathsf{G}}_\eta = 1 \wedge \neg \text{corr}_\eta]|. \end{aligned}$$

This completes the proof of Claim 1.

B.2 Proof of Claim 2

The only difference between $\tilde{\mathsf{G}}_\eta$ and G_η lies in the $\mathcal{O}_{\text{REV}}(\eta, c)$ for the user η : in $\tilde{\mathsf{G}}_\eta$, $\mathcal{O}_{\text{REV}}(\eta, c)$ outputs a random key R ; in G_η , $\mathcal{O}_{\text{REV}}(\eta, c)$ outputs the real key K . We note that, the $\mathcal{O}_{\text{TEST}}(\eta, c)$ oracle outputs a random key K_1 in both $\tilde{\mathsf{G}}_\eta$ and G_η .

We construct \mathcal{B}_2 against the IND-mCCA security of KEM by invoking \mathcal{A} . \mathcal{B}_2 behaves almost the same as the \mathcal{B}_1 constructed in the proof of Claim 1, except that when answering $\mathcal{O}_{\text{TEST}}(\eta, c)$ query made by \mathcal{A} , \mathcal{B}_2 always responds with a random key uniformly chosen from \mathcal{K} .

More precisely, \mathcal{B}_2 works as follows. Let b denote the challenge bit chosen by \mathcal{B}_2 's challenger. \mathcal{B}_2 has access to two oracles $\mathcal{O}_{\text{ENC}}()$ and $\mathcal{O}_{\text{DEC}}(\cdot)$ (cf. Fig. 1). Given $(\text{pp}_{\text{KEM}}, pk)$ as input, \mathcal{B}_2 generates $(pk_i, sk_i) \leftarrow \text{Gen}(\text{pp}_{\text{KEM}})$ for all $i \in [n \setminus \eta]$, sets $pk_\eta := pk$, and sends $(\text{pp}_{\text{KEM}}, \text{PKList} := \{pk_i\}_{i \in [n]})$ to \mathcal{A} . \mathcal{B}_2 initializes a flag $\text{corr}_\eta := \text{false}$. Then \mathcal{B}_2 answers the oracle queries $\mathcal{O}_{\text{ENC}}(\cdot)$, $\mathcal{O}_{\text{DEC}}(\cdot, \cdot)$, $\mathcal{O}_{\text{REV}}(\cdot, \cdot)$, $\mathcal{O}_{\text{CORR}}(\cdot)$, $\mathcal{O}_{\text{TEST}}(\cdot, \cdot)$ made by \mathcal{A} in the following way.

For oracle queries $\mathcal{O}_{\text{ENC}}(i)$, $\mathcal{O}_{\text{DEC}}(i, c')$, $\mathcal{O}_{\text{REV}}(i, c)$, $\mathcal{O}_{\text{CORR}}(i)$, $\mathcal{O}_{\text{TEST}}(i, c)$ where $i \neq \eta$, \mathcal{B}_2 prepares the responses exactly the same as $\mathcal{G}_{\eta-1}$ and $\tilde{\mathcal{G}}_\eta$ using (pk_i, sk_i) , and returns the answers to \mathcal{A} . In particular, for $\mathcal{O}_{\text{TEST}}(i, c)$ query where $i \neq \eta$, \mathcal{B}_2 outputs K_0 if $i > \eta$ and outputs K_1 if $i < \eta$.

Below we show how \mathcal{B}_2 answers the oracle queries for $i = \eta$.

- For query $\mathcal{O}_{\text{ENC}}(\eta)$, \mathcal{B}_2 queries its own oracle $\mathcal{O}_{\text{ENC}}()$, obtains (c, K_b) , appends (η, c, K_b) to EncList , and returns c to \mathcal{A} . Clearly, \mathcal{B}_2 simulates oracle $\mathcal{O}_{\text{ENC}}(\eta)$ in $\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{muc-ecca}}$ (thus $\tilde{\mathcal{G}}_\eta$ and \mathcal{G}_η) perfectly for \mathcal{A} .
- For query $\mathcal{O}_{\text{DEC}}(\eta, c')$, \mathcal{B}_2 returns \perp to \mathcal{A} if $(\eta, c', \cdot) \in \text{EncList}$. Otherwise, \mathcal{B}_2 queries its own oracle $\mathcal{O}_{\text{DEC}}(c')$, obtains K' , and returns K' to \mathcal{A} . Clearly, \mathcal{B}_2 simulates oracle $\mathcal{O}_{\text{DEC}}(\eta, c')$ in $\text{Exp}_{\text{KEM}, n, Q_e, Q_t}^{\text{muc-ecca}}$ (thus $\tilde{\mathcal{G}}_\eta$ and \mathcal{G}_η) perfectly for \mathcal{A} .
- For query $\mathcal{O}_{\text{REV}}(\eta, c)$, if $(\eta, c, K) \in \text{EncList}$ for some $K = K_b$ and $(\eta, c) \notin \text{TestList}$, \mathcal{B}_2 appends (η, c) to RevList , and returns K_b to \mathcal{A} . Otherwise, \mathcal{B}_2 returns \perp to \mathcal{A} .

Note that, in the case $b = 0$, $K_b = K_0$ is the real key encapsulated in c , thus \mathcal{B}_2 simulates oracle $\mathcal{O}_{\text{REV}}(\eta, c)$ in \mathcal{G}_η perfectly for \mathcal{A} ; in the case $b = 1$, $K_b = K_1$ is a random key uniformly chosen from \mathcal{K} , thus \mathcal{B}_2 simulates oracle $\mathcal{O}_{\text{REV}}(\eta, c)$ in $\tilde{\mathcal{G}}_\eta$ perfectly for \mathcal{A} .

- For query $\mathcal{O}_{\text{CORR}}(\eta)$, if $(\eta, \cdot) \in \text{TestList}$, \mathcal{B}_2 returns \perp to \mathcal{A} . Otherwise, \mathcal{B}_2 sets $\text{corr}_\eta := \text{true}$, terminates the interaction with \mathcal{A} , and aborts the game.
- For query $\mathcal{O}_{\text{TEST}}(\eta, c)$, if $(\eta, c, K) \in \text{EncList}$ for some $K = K_b$ and $(\eta, c) \notin \text{RevList} \cup \text{TestList}$ and $\eta \notin \text{CorrList}$, \mathcal{B}_2 appends (η, c) to TestList , picks $R \leftarrow \mathcal{K}$ randomly, and returns R to \mathcal{A} . Otherwise, \mathcal{B}_2 returns \perp to \mathcal{A} .

Note that, R is a random key uniformly chosen from \mathcal{K} , thus \mathcal{B}_2 simulates oracle $\mathcal{O}_{\text{TEST}}(\eta, c)$ both in $\tilde{\mathcal{G}}_\eta$ and \mathcal{G}_η perfectly for \mathcal{A} .

Finally, \mathcal{B}_2 receives a guessing bit from \mathcal{A} . \mathcal{B}_2 outputs 1 to its own challenger if and only if \mathcal{A} outputs 1 and $\neg \text{corr}_\eta$. Clearly, $t_{\mathcal{B}_2} \leq t_{\mathcal{A}} + (n + Q_{\text{total}}) \cdot t_{\text{KEM}}$.

For the game simulated by \mathcal{B}_2 ,

- In the case $b = 0$, \mathcal{B}_2 simulates \mathcal{G}_η perfectly for \mathcal{A} unless corr_η occurs.
- In the case $b = 1$, \mathcal{B}_2 simulates $\tilde{\mathcal{G}}_\eta$ perfectly for \mathcal{A} unless corr_η occurs.

Therefore, we have

$$\begin{aligned} \text{Adv}_{\text{KEM}}^{\text{ind-mcca}}(\mathcal{B}_2) &= 2|\Pr[\mathcal{B}_2 \Rightarrow b] - 1/2| = |\Pr[\mathcal{B}_2 \Rightarrow 1 | b = 0] - \Pr[\mathcal{B}_2 \Rightarrow 1 | b = 1]| \\ &= |\Pr[\mathcal{A} \Rightarrow 1 \wedge \neg \text{corr}_\eta | b = 0] - \Pr[\mathcal{A} \Rightarrow 1 \wedge \neg \text{corr}_\eta | b = 1]| \\ &= |\Pr[\mathcal{G}_\eta = 1 \wedge \neg \text{corr}_\eta] - \Pr[\tilde{\mathcal{G}}_\eta = 1 \wedge \neg \text{corr}_\eta]|. \end{aligned}$$

This completes the proof of Claim 2.

Table of Contents

Key Encapsulation Mechanism with Tight Enhanced Security in the Multi-User Setting: Impossibility Result and Optimal Tightness	1
<i>Shuai Han</i> ^{id} , <i>Shengli Liu</i> ^{(✉) id} , and <i>Dawu Gu</i> ^{id}	
1 Introduction	1
1.1 Technique Overview	4
1.2 Application of Our Impossibility Result in AKE	7
1.3 Related Works	8
2 Preliminaries	8
2.1 Notations	8
2.2 Key Encapsulation Mechanisms	9
2.3 Non-Interactive Assumptions	9
3 Enhanced Security Notions for KEMs	10
4 Decap-Equivalence of Secret Keys & Rank of KEMs	11
4.1 Decap-Equivalence of Secret Keys	11
4.2 Rank of KEMs	12
4.3 Core Lemma	13
5 Impossibility of Tight Enhanced Security for KEMs	13
5.1 Impossibility of Tight Enhanced Security for KEMs	14
5.2 Applications of Our Impossibility Result to Well-Known KEMs	17
6 Enhancedly Secure KEM with Optimal Tightness	19
A Computing Ranks for Well-Known KEM Schemes: More Instances	23
B Omitted Proofs	27
B.1 Proof of Claim 1	27
B.2 Proof of Claim 2	27