





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

Shuai Han<sup>1,2</sup> , Shengli Liu<sup>1,2,3</sup>  , and Dawu Gu<sup>1</sup> 

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

<sup>2</sup> State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China

<sup>3</sup> 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  $\Pi$  (or a computational

problem  $P$ ), its security model is described by an experiment (game)  $\text{Exp}_\Pi$  in which the environment (challenger) and an adversary interact with each other. The environment (challenger) in  $\text{Exp}_\Pi$  provides the resources with which  $\mathcal{A}$  implements attacks, then environment detects whether  $\mathcal{A}$  wins. Here  $\mathcal{A}$  wins in  $\text{Exp}_\Pi$  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  $\Pi$  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  $\lambda$ . If for all probabilistic polynomial-time (PPT) adversaries, their advantages are all negligible in the security parameter  $\lambda$ , then  $\Pi$  is (asymptotically) secure.

**Security Reduction and Tightness.** The security proof of primitive  $\Pi$  generally proceeds with a reduction algorithm  $\mathcal{R}$ , which transforms a  $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}})$ -adversary  $\mathcal{A}$  against  $\Pi$  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  $\lambda$ , then the security of  $\Pi$  is successfully reduced to the hardness of  $P$ . This implies that  $\Pi$  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  $\Pi$ 's security level and  $P$ 's hardness. To fill the gap,  $\Pi$  has to choose a large security parameter to make  $P$  hard enough, and this might make  $\Pi$  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  $\lambda$ , then the reduction is called *almost tight* (or *linear-preserving* according to [33]). 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  $\text{Gen}$  is able to generate a pair of public key  $pk$  and secret key  $sk$ . With the public key  $pk$ , the encapsulation algorithm  $\text{Encap}$  can output an encapsulation  $c$  and an encapsulated key  $K$ . With the secret key  $sk$ , the de-

capsulation 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 [28], 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, 31, 24, 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, 28]. 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, 24]. The security model of AKE (like the CK [5], CK+ [27], eCK [29] 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 pseudo-randomness 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 [28] are among the most efficient KEMs. The GHKW-KEM [14] and HLLG-KEM [19] are core building blocks in achieving (almost) tightly mCCA security for PKE. The Naor-Yung paradigm [35] 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 [28], 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  $\log_{g_1} g_2 = (z_1 - z'_1)(z'_2 - z_2)^{-1} \pmod p$ , 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  $\lambda$ ), 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 [28], GHKW-KEM [14], HLLG-KEM [19], and many instantiations of NY-KEM [35]. 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 [28] all have ECPA security with security loss factor  $\Theta(n)$  when reduced to the DDH assumption. Similarly, the HLLG-KEM [19] 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, 20], uniqueness, or re-randomizability [7, 26, 22, 2, 32]), nor limited to deterministic primitives [33].

## 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, 26, 22, 2, 36, 33]. 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 [33].

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  $\alpha$  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  $\delta$ , 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  $\alpha$  of  $\mathcal{A}^*$  and the failure probability  $\delta$  of  $\mathcal{B}$  are the key factors for the lower bound of loss factor  $\ell_{\mathcal{R}}$ .

Let us take [2] and [33] 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 [33],  $\alpha = 1/\text{poly}(\lambda)$  for some polynomial  $\text{poly}$ . If MAC is deterministic, then  $\delta = 1/\sqrt{n}$ .<sup>1</sup> 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 [33] 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 checkability and uniqueness/re-randomizability. Such a condition is satisfied by the ElGamal-KEM [12], but

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

lots of other KEMs run out, including the CS-KEM [8, 9, 10], KD-KEM [28], GHKW-KEM [14], HLLG-KEM [19], 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  $[n \setminus i] := \{1, 2, \dots, n\} \setminus \{i\}$ .

A high-level overview of  $\mathcal{A}^*$  is presented in Fig. 1. Note that by the perfect correctness of KEM, the real secret key  $sk_{i^*}$  of user  $i^*$  also belongs to the 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



### 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  $j_1, j_2, \dots, j_n \leftarrow_{\$} [Q]$  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  $i^* \leftarrow_{\$} [n]$  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.

**Fig. 1.** High-level overview of the hypothetical adversary  $\mathcal{A}^*$  in our meta-reduction.

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 presented in Fig. 2.

In step 5', as long as there exists a rewinding in which  $\mathcal{R}$  responds with a corrupted secret key  $sk_{i^*}^{(l)}$  such that (4) holds, then  $\mathcal{B}$  will not abort. Since  $i^*$  is randomly chosen from  $[n]$ , by a similar argument as [2, 20], we can bound the probability that  $\mathcal{B}$  aborts by  $1/n$ . If (4) holds, then the  $sk_{i^*}^{(l)}$  obtained by  $\mathcal{B}$  also belongs to the set defined in (2), from which  $\mathcal{A}$  chooses its  $sk^*$ , thus

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

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  $\alpha$  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,

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



### 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  $\iota$ -th rewind ( $\iota \in [n \setminus i^*]$ ),  $\mathcal{B}$  corrupts all users except user  $\iota$  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  $\iota$ -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.

**Fig. 2.** High-level overview of the meta-reduction algorithm  $\mathcal{B}$  in our meta-reduction.

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  $\alpha$  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  $\text{Rank}_{\text{KEM}}$ , 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  $\alpha$  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 Subject. 4.3). The core lemma shows that the number of bad ciphertexts in any ciphertext subset  $\mathcal{X}$  is upper bounded by  $\text{Rank}_{\text{KEM}}$ . 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  $\text{Rank}_{\text{KEM}}$  is bounded by an a priori fixed polynomial (in the security parameter  $\lambda$ ), 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 [28], GHKW-KEM [14], HLLG-KEM [19] and many instantiations of NY-KEM [35]) shows that our impossibility result applies to these KEM schemes. See Subject. 5.2 and the full version [18] 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, 38, 31, 24, 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 [29], or CK+ model [27], 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  $sk_{\text{AKE}}$  requires the security of the underlying KEM to support corruption of  $sk_{\text{KEM}}$ , 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, 31, 17]; or (ii) the tight security proof of AKE relies on the Random Oracle model, like [16, 24, 37]; 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 [26, 2, 32].

Hofheinz et al. [22] 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 [30] used the technique in [22] 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. [25] 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. [33] 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 [33] 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  $K \in \mathcal{K} \cup \{\perp\}$ .

We 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. 3.

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

**Fig. 3.** 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  $(x, w) \leftarrow_s T$ , executes  $s \leftarrow_s Z(x)$ , 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. 4 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 $i \in [n]$ : $(pk_i, sk_i) \leftarrow \text{Gen}(pp)$ $\text{EncList} := \emptyset$ //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 $(c, K) \leftarrow \text{Encap}(pk_i)$ $\text{EncList} := \text{EncList} \cup \{(i, c, K)\}$ Return $c$ //Only $c$ is returned  $\mathcal{O}_{\text{DEC}}(i, c):$ If $(i, c, \cdot) \notin \text{EncList}$ : Return $K' \leftarrow \text{Decap}(sk_i, c')$ Else: Return $\perp$  $\mathcal{O}_{\text{REV}}(i, c):$ If $(i, c, K) \in \text{EncList}$ 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 $(i, c, K) \in \text{EncList}$ 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 $K_{\beta}$ Else: Return $\perp$	

**Fig. 4.** 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  $\mathcal{O}_{\text{ENC}}$  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. 4, 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  $\mathcal{O}_{\text{ENC}}$ , different from the IND-mCPA/mCCA experiment (cf. Fig. 3). 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  $\beta$ . We note that this is defined in the Single-Bit-Guess (SBG) style [6, 24], 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  $\text{Decap}_{\text{verify}}(sk, c)$  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  $\text{Decap}_{\text{verify}}(sk, c) = 1$ , then  $K \leftarrow \text{Decap}_{\text{kd}}(sk, c)$



is invoked and  $\text{Decap}(sk, c)$  will output  $K$ . If  $\text{Decap}_{\text{verify}}(sk, c) = 0$ , 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  $\text{Decap}_{\text{verify}}$  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  $\text{Decap}_{\text{kd}}$  is the essential part of the decapsulation algorithm, the decapsulation equivalence is defined with  $\text{Decap}_{\text{kd}}$ , 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  $\text{EquivSK}$  in the following lemma.

**Lemma 1 (Properties of  $\text{EquivSK}$ ).** *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  $(sk, sk') \in \mathcal{SK}^2$ ,

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

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

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

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  $\text{BadIndex}$  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  $\text{BadIndex}$  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

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

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, 20], 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, 20, 33]).** *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  $\lambda$ ,  $\mathcal{R}$  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 [22, 30, 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  $(pp, \text{PKList})$  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  $nQ (= Q_e)$   $\mathcal{O}_{\text{ENC}}$  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  $\text{bad}$  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  $\beta$  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  $\text{bad}$  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

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

**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  $(x, w) \leftarrow_{\$} T$ ,  $\mathcal{B}$  works as follows.

- **Setup.**  $\mathcal{B}$  runs  $\mathcal{R}(x)$  to obtain  $(pp, \text{PKList})$  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  $j_i \leftarrow_{\$} [Q]$  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  $st_{\text{prep}}$ .  
 $\mathcal{B}$  picks a user index  $i^* \leftarrow_{\$} [n]$  uniformly at random.
- **Rewinding.** Next,  $\mathcal{B}$  will rewind  $\mathcal{R}$   $n$  times, all starting from state  $st_{\text{prep}}$ . In the  $\iota$ -th rewind ( $\iota \in [n]$ ),  $\mathcal{B}$  proceeds as follows:
  - (1)  $\mathcal{B}$  rewinds  $\mathcal{R}$  to the state  $st_{\text{prep}}$ . 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  $\text{Decap}_{\text{kd}}$  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  $t_{\text{Decap}}$  denotes the running time of the Decap algorithm of KEM.

**ANALYSIS OF  $\mathcal{B}$ 'S ADVANTAGE.** Denote by  $\text{bad}$  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  $\text{checkfail}_1$  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  $\text{bad}$  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  $\text{checkfail}_2$  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  $\text{checkfail}_1$  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  $(t_N, \epsilon_N)$ -break the NICA  $N$  with  $t_N = t_{\mathcal{B}} \leq n \cdot t_{\mathcal{R}} + nQ_e \cdot t_{\text{Decap}}$ , 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,  $n \cdot (1 - n\epsilon_N - dn(n+1)/Q_e - 1/\#\mathcal{K}) \cdot (\epsilon_N + 1/n + dn/Q_e) = 1 - dn/Q_e - 1/\#\mathcal{K} - (n\epsilon_N + n^2d/Q_e) \cdot (n\epsilon_N + dn(n+1)/Q_e + 1/\#\mathcal{K}) \leq 1 - dn/Q_e - 1/\#\mathcal{K}$ . 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,  $\epsilon_N$  represents the hardness of NICA  $N$ ,  $d$  is the rank of KEM,  $Q_e$  is the number of  $\mathcal{O}_{\text{ENC}}$  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  $\ell$  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  $\lambda$  is the security parameter, we have  $\ell = n(1 - 3/\lambda) = n(1 - o(1)) \approx n$ .

In either case, the security loss  $\ell$  is at least linear in  $n$ . This completes the proof of Theorem 1.  $\blacksquare$

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, 8, 9, 10, 28, 14, 15, 19] (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 [28] are among the most efficient KEMs. The GHKW-KEM [14] and HLLG-KEM [19] are core building blocks in achieving (almost) tightly IND-mCCA security for PKE. The NY-KEM [35] 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  $\lambda$ .

- The CPA-secure ElGamal-KEM [12] has rank 1 (cf. the full version [18]).
- The CCA-secure CS-KEM in [8] has rank 1 (cf. the full version [18]) and another version in [9, 10] has rank 2 (see next).
- The CCCA (constrained CCA) secure KD-KEM [28] has rank at most 4 (cf. the full version [18]).

- 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. the full version [18]).
- The tightly mCCA-secure HLLG-KEM [19] has rank at most  $2k$ , with  $k$  the parameter of the MDDH assumptions (cf. the full version [18]).
- The CCA-secure NY-KEM [35] has polynomially-bounded rank, as long as the underlying CPA-secure KEM does (cf. the full version [18]). 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.

Due to space limitations, here we show how to compute rank for the CS-KEM [9, 10], and put the rank computations of other KEMs in the full version [18].

**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

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

Clearly, we have both  $\text{EquivSK}(c^{(1)}, c^{(2)}) \subseteq \text{EquivSK}(c^{(1)})$  and  $\text{EquivSK}(c^{(1)}, c^{(2)}) \subseteq \text{EquivSK}(c^{(2)})$ , 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  $\Theta(1)$  [34].
- In 2012, Hofheinz and Jager [21] presented the first tightly IND-mCCA secure PKE based on (matrix) DDH assumptions [13], with security loss  $\Theta(1)$ .
- Recent works [14, 15, 19] 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, 21, 14, 15, 19] achieve MUMC-ECPA/ECCA security based on the standard (matrix) DDH assumptions with security loss  $\Theta(n)$  [12, 21] (resp.,  $O(\lambda n)$  [14, 15, 19]), 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  $\mathcal{O}_{\text{REV}}$ .

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  $\eta$ -th hybrid ( $\eta \in [n]$ ), we change the encapsulated keys in  $\mathcal{O}_{\text{TEST}}$  w.r.t. user  $\eta$  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  $\eta$ -th hybrid, our reduction

embeds the public key in the IND-mCPA/mCCA experiment into the public key of user  $\eta$ . There are two cases. If  $\mathcal{A}$  never corrupts user  $\eta$ ,  $\mathcal{B}$  can simulate the MUMC-ECPA/ECCA experiment perfectly for  $\mathcal{A}$ . So the change of  $\mathcal{O}_{\text{TEST}}$  for user  $\eta$  is unnoticeable to  $\mathcal{A}$  by the IND-mCPA/mCCA security. If  $\mathcal{A}$  asks to corrupt user  $\eta$ ,  $\mathcal{B}$  aborts immediately. Note that in the latter case,  $\mathcal{A}$  is not allowed to query  $\mathcal{O}_{\text{TEST}}$  for user  $\eta$  when user  $\eta$  is (going to be) corrupted. So the change of  $\mathcal{O}_{\text{TEST}}$  for user  $\eta$  is conceptual.

- **Key reveal with random keys.** To handle key reveal queries for user  $\eta$ , we borrow the ideas from [31]. If  $\mathcal{A}$  never corrupts user  $\eta$ ,  $\mathcal{B}$  can output a random key for key reveal queries since  $\mathcal{A}$  never sees the secret key of user  $\eta$ . If  $\mathcal{A}$  asks to corrupt user  $\eta$ ,  $\mathcal{B}$  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.

Formally, we have the following theorem, with proof appeared in the full version [18] due to space limitations.

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

**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, pp. 629–658 (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, pp. 273–304 (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, pp. 407–424 (2009)
- [4] Boneh, D., Venkatesan, R.: Breaking RSA may not be equivalent to factoring. In: Nyberg, K. (ed.) EUROCRYPT 1998, pp. 59–71 (1998)
- [5] Canetti, R., Krawczyk, H.: Universally composable notions of key exchange and secure channels. In: Knudsen, L.R. (ed.) EUROCRYPT 2002, pp. 337–351 (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, pp. 767–797 (2019)
- [7] Coron, J.S.: Optimal security proofs for PSS and other signature schemes. In: Knudsen, L.R. (ed.) EUROCRYPT 2002, pp. 272–287 (2002)
- [8] Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998, pp. 13–25 (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, pp. 45–64 (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), pp. 167–226 (2003)
- [11] Diffie, W., Hellman, M.E.: New directions in cryptography. *IEEE Trans. Inf. Theory* 22(6), pp. 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 1984, pp. 10–18 (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, pp. 129–147 (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, pp. 1–27 (2016)
- [15] Gay, R., Hofheinz, D., Kohl, L.: Kurosawa-Desmedt meets tight security. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, pp. 133–160 (2017)
- [16] Gjøsteen, K., Jager, T.: Practical and tightly-secure digital signatures and authenticated key exchange. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, pp. 95–125 (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, pp. 670–700 (2021)
- [18] Han, S., Liu, S., Gu, D.: Key encapsulation mechanism with tight enhanced security in the multi-user setting: Impossibility result and optimal tightness. *Cryptology ePrint Archive, Report 2021/1146* (2021), <https://eprint.iacr.org/2021/1146>
- [19] 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, pp. 417–447 (2019)
- [20] Hesse, J., Hofheinz, D., Kohl, L.: On tightly secure non-interactive key exchange. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, pp. 65–94 (2018)



- [21] Hofheinz, D., Jager, T.: Tightly secure signatures and public-key encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012, pp. 590–607 (2012)
- [22] Hofheinz, D., Jager, T., Knapp, E.: Waters signatures with optimal security reduction. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012, pp. 66–83 (2012)
- [23] Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) CRYPTO 2007, pp. 553–571 (2007)
- [24] Jager, T., Kiltz, E., Riepel, D., Schäge, S.: Tightly-secure authenticated key exchange, revisited. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, pp. 117–146 (2021)
- [25] 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, pp. 409–441 (2017)
- [26] Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012, pp. 537–553 (2012)
- [27] Krawczyk, H.: HMQV: A high-performance secure Diffie-Hellman protocol. In: Shoup, V. (ed.) CRYPTO 2005, pp. 546–566 (2005)
- [28] Kurosawa, K., Desmedt, Y.: A new paradigm of hybrid encryption scheme. In: Franklin, M. (ed.) CRYPTO 2004, pp. 426–442 (2004)
- [29] LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger security of authenticated key exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007, pp. 1–16 (2007)
- [30] Lewko, A.B., Waters, B.: Why proving HIBE systems secure is difficult. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014, pp. 58–76 (2014)
- [31] 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, pp. 785–814 (2020)
- [32] Morgan, A., Pass, R.: On the security loss of unique signatures. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, pp. 507–536 (2018)
- [33] Morgan, A., Pass, R., Shi, E.: On the adaptive security of MACs and PRFs. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, pp. 724–753 (2020)
- [34] Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: 38th FOCS. pp. 458–467. IEEE Computer Society Press (1997)
- [35] Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. pp. 427–437. ACM Press (1990)
- [36] Niehues, D.: Verifiable random functions with optimal tightness. In: Garay, J. (ed.) PKC 2021, pp. 61–91 (2021)
- [37] Pan, J., Qian, C., Ringerud, M.: Signed Diffie-Hellman key exchange with tight security. In: Paterson, K.G. (ed.) CT-RSA 2021, pp. 201–226 (2021)
- [38] 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, pp. 158–189 (2018)