Tag-KEM/DEM: A New Framework for Hybrid Encryption

Masayuki Abe[1] Rosario Gennaro[2] Kaoru Kurosawa[4]

October 11, 2006

[1] Information Sharing Platform Laboratories, NTT Corporation, Japan
abe.masayuki@lab.ntt.co.jp

[2] IBM T. J. Watson Rsearch Center, USA.
Rosario@us.ibm.com

[3] Ibaraki University, Japan
kurosawa@mx.ibaraki.ac.jp

Abstract

This paper presents a novel framework for the generic construction of hybrid encryption schemes which produces more efficient schemes than the ones known before. A previous framework introduced by Shoup combines a key encapsulation mechanism (KEM) and a data encryption mechanism (DEM). While it is sufficient to require both components to be secure against chosen ciphertext attacks (CCA-secure), Kurosawa and Desmedt showed a particular example of KEM that is not CCA-secure but can be securely combined with a specific type of CCA-secure DEM to obtain a more efficient, CCA-secure hybrid encryption scheme. There are also many other efficient hybrid encryption schemes in the literature that do not fit Shoup's framework. These facts serve as motivation to seek another framework.

The framework we propose yields more efficient hybrid scheme, and in addition provides insightful explanation about existing schemes that do not fit into the previous framework. Moreover, it allows immediate conversion from a class of threshold public-key encryption to a hybrid one without considerable overhead, which may not be possible in the previous approach.

Keywords: Hybrid Encryption, KEM, DEM, Chosen Ciphertext Security

Contents

1 Introduction

A fundamental task of cryptography is to protect the secrecy of messages transmitted over public communication lines. For this purpose we use *encryption schemes* which use some secret information (a key) to encode messages in a way that an eavesdropper cannot decode. However, as networks become more open and accessible, it becomes apparently clear that an adversary may not be limited to eavesdropping, but may take a more active role. She may try to interact with honest parties, by, for example, sending ciphertexts to them (possibly related to the ciphertexts she intends to decrypt) and analyze their response. Such active attacks can be proven to be much more powerful and hard to combat than passive ones (see for example [7]).

To model this type of attacks, the notion of *chosen-ciphertext security* was introduced by Naor and Yung [33] and developed by Rackoff and Simon [35], and Dolev, Dwork, and Naor [22]. Security against a chosen ciphertext attack (CCA security, in short) means that, even if the adversary is allowed to query a *decryption oracle* on ciphertext of her choosing, then she obtains no useful information about messages encrypted in other ciphertexts. The first CCA-secure cryptosystems were presented in [33, 35, 22], but they were quite impractical, as they rely on generic techniques for non-interactive zero-knowledge. In a breakthrough result, Cramer and Shoup in [16] presented the first truly practical CCA-secure cryptosystem, whose security was based on the hardness of the decisional Diffie-Hellman problem. This construction was generalized in [17], using a new cryptographic primitive called *projective hash functions*.

Public-key encryption schemes often limit the message space to a particular group, which can be restrictive when one wants to encrypt arbitrary messages. For this purpose *hybrid* schemes are devised, composed by the two parts. First a *Key Encapsulation Mechanism* (KEM) is invoked: a random group element is encrypted and then mapped via a key derivation function into a random key. Then a *Data Encapsulation Mechanism* is performed: the random key is used to encrypt the message using a symmetric encryption scheme. A formal treatment of this paradigm can be found in [38, 18] and we refer to it as the KEM/DEM framework.

As mentioned in the literature, it is sufficient that both KEM and DEM are CCA-secure to obtain CCA-secure hybrid encryption. This indeed looks quite reasonable since, if either component is not CCA-secure, then the adversary trying to decrypt a target ciphertext may be able to alter the corresponding part of the ciphertext and use the decryption oracle to get useful information. Recently in [30], Kurosawa and Desmedt introduced a hybrid encryption scheme which is a modification of the hybrid scheme presented in [36]. Their scheme is interesting from a theoretical point of view: when one looks at it as a KEM/DEM scheme, their KEM is not CCA-secure [27]. Nevertheless, the resulting scheme is CCA-secure and more efficient than [38, 18] both in computation and bandwidth. Thus the Kurosawa-Desmedt scheme suggests that requiring both KEM/DEM to be CCA-secure, in order to obtain CCA-secure hybrid encryption, while being a sufficient condition, may not be a necessary one, and might indeed be an overkill.

Moreover, there are other hybrid encryption schemes in the literature, e.g.,[4, 34] in the random oracle model, which are very efficient, but do not fit to the CCA-secure KEM/DEM framework.

Our Contribution. Prompted by the above observation, we set out to investigate another KEM/DEM framework that yields more efficient hybrid encryption schemes and captures a wider variety of existing schemes. Our results can be summarized as follows:

- We introduce *Tag-KEMs*: a form of KEM which also takes as input a *tag*. Though such a notion is known in the literature, e.g., [38], we give an extended syntax and show, somewhat surprisingly, that if one uses a CCA-secure Tag-KEM in a novel way, then it is sufficient for the DEM to be secure simply against a passive attacker, to yield CCA-secure hybrid encryption.

- We present several constructions of CCA-secure Tag-KEMs based on various combination of assumptions on the tools used to build them. A class of KEM that is strictly less secure than CCA-secure ones but can yield CCA-secure Tag-KEM is shown. Importantly, we show that the KEM by Kurosawa and Desmedt belongs to this class, thus providing a theoretical understanding of their scheme. This answers an open question of [30].
- We show that the Tag-KEM/DEM framework provides a simple way to create threshold versions of CCA-secure hybrid encryption schemes, which may not be possible in the KEM/DEM framework.
- Finally, we show how several schemes in the literature can be cast in our Tag-KEM/DEM framework. Furthermore we show that some of those schemes can actually be simplified when considered as instances of our framework.

2 Definitions and Building Blocks

This section introduces all the building blocks used in this paper. Among them, the notion of Tag-KEM (Section 2.1), DEM (Section 2.2), and PKE (Section 2.3) are used in Section 3 to construct our main result. Other building blocks are used in specific constructions or applications shown in Section 4 and 5, respectively.

2.1 Key Encapsulation Mechanism with Tags (Tag-KEM)

In Shoup's model, a KEM consists, similary to public-key encryption, of three algorithms: key generation, encryption and decryption. The difference is that the encryption algorithm takes as input only the public key pk and outputs a random one-time key and its encryption. (See Section 2.4.) The encryption function may also take an arbitrary string (a tag) as an input associated to every ciphertext. In our model, we divide the encryption function into two functions in such a way that the first one selects a random key and the second one encrypts the key along with a given tag. We call a KEM that meets this model a Tag-KEM. Formally:

$(pk, sk) \leftarrow \mathsf{TKEM.Gen}(1^\lambda)$	A probabilistic algorithm that generates public-key pk and private-key sk. The public-key defines all relative spaces, i.e., spaces for tags and encapsulated keys denoted by $\mathcal{T}$ and $\mathcal{K}_K$.
$(\omega, dk) \leftarrow \mathsf{TKEM.Key}(pk)$	A probabilistic algorithm that outputs one-time key $dk \in \mathcal{K}_D$ and internal state information ω. $\mathcal{K}_D$ is the key-space of DEM.
$\psi \leftarrow \mathsf{TKEM.Enc}(\omega, \tau)$	A probabilistic algorithm that encrypts dk (embedded in ω) into ψ along with τ, where τ is called a tag.
$dk \leftarrow \mathsf{TKEM.Dec}_{sk}(\psi, \tau)$	A decryption algorithm that recovers dk from ψ and τ. For soundness, $\mathsf{TKEM.Dec}_{sk}(\psi, \tau) = dk$ must hold for any sk, dk, ψ, and τ, associated by the above three functions. The algorithm can also output special symbol $\perp \notin \mathcal{K}_D$ to present abnormal termination.

Tag-KEM is a generalization of KEM because if the tag is a fixed string, it is a KEM. Note that, in the above syntactic definition, τ is not included in ψ and explicitly given to $\mathsf{TKEM.Dec}$.

Such explicit treatment of τ has some notational advantages when we consider an adversary who tries to alter the tag without affecting the encapsulation ψ.

The security of a Tag-KEM requires that the adversary should fail to distinguish whether a given dk is the one embedded in the cipehrtext (ψ, τ) or not, with adaptive access to the decryption oracle. Let $\mathcal{O}$ be the decryption oracle, $\mathsf{TKEM.Dec}_{sk}(\cdot,\cdot)$. Let A_T be a probabilistic polynomial-time (ppt) oracle machine that plays the following game.

[GAME.TKEM]

Step 1. $(pk, sk) \leftarrow \mathsf{TKEM.Gen}(1^\lambda)$, $(\omega, dk_1) \leftarrow \mathsf{TKEM.Key}(pk)$, $dk_0 \leftarrow \mathcal{K}_D$, $\delta \leftarrow \{0,1\}$.

Step 2. $(\tau, \rho) \leftarrow {A_T}^{\mathcal{O}}(pk, dk_\delta)$

Step 3. $\psi \leftarrow \mathsf{TKEM.Enc}(\omega, \tau)$

Step 4. $\tilde{\delta} \leftarrow {A_T}^{\mathcal{O}}(\rho, \psi)$

In Step 4, A_T is restricted not to ask (ψ, τ) to decryption oracle $\mathcal{O}$. The variable ρ is a state information of the adversary, and dk_δ is set to either dk_0 or dk_1 according to the value of $\delta \in \{0,1\}$. Such convention is used throughout the paper unless otherwise noted. We define $\epsilon_{\text{tkem},A_T} = \left|\Pr[\tilde{\delta} = \delta] - \frac{1}{2}\right|$ and $\epsilon_{\text{tkem}} = \max_{A_T}(\epsilon_{\text{tkem},A_T})$ where maximum is taken over all machines. We say that a Tag-KEM is CCA-secure if ϵ_{tkem} is negligible in λ.

The above security definition simplifies the one presented in [3] in the sense that the adversary is given the key dk_δ at the beginning of the game. It does not affect to the construction but relevant proofs becomes slightly more involved.

Relation to Tag-based PKEs. Tags associated to PKE or KEM can be found in the literature (e.g. see [39, 38]), but their syntactic definition and the purpose are different from ours; A tag is supposed to carry an identity of the encryptor and has to be fixed before the DEM key is selected. (The encryption function takes a tag as an input and outputs a DEM key.) Despite their limitations, this particular implementation also fits into our model without essential modifications. Tag-based PKE is also introduced in [31] with the same syntax as that of [39, 38] but with a weaker security notion. In their work, the adversary is restricted from sending the same tag associated to the challenge ciphertext to the decryption oracle. Such a weak security is sufficient for some cryptographic applications, as shown in [31]. Though it does not fit into our framework, one of the constructions in [31] is identical as the one presented in Section 5.3 and indeed achieves our higher level of security. The work in [29] introduces an even weaker definition where the adversary commits itself to a tag at the beginning of the attack game. It then shows how to convert such weak security into full CCA-security by using an extra component such as a strong one-time signature or a message authentication code.

2.2 Data Encapsulation Mechanism (DEM)

A DEM is a symmetric encryption scheme that consists of two algorithms, $\mathsf{DEM.Enc}$ and $\mathsf{DEM.Dec}$ associated to a key-space and message space defined by λ. We assume the key space $\mathcal{K}_D$ is $\{0,1\}^\lambda$ while the message space is $\{0,1\}^*$.

$\chi \leftarrow \mathsf{DEM.Enc}_{dk}(m)$	An encryption algorithm that encrypts m into ciphertext χ by using symmetric-key $dk \in \mathcal{K}_D$.
$m \leftarrow \mathsf{DEM.Dec}_{dk}(\chi)$	A corresponding decryption algorithm that recovers message m from input ciphertext χ. Obvious soundness condition applies.

We only require passive security for DEM. Let A_D be a polynomial-time machine that plays the following game.

[GAME.DEM]

Step 1. $(m_0, m_1, \rho) \leftarrow A_D(1^\lambda)$.

Step 2. $dk \leftarrow \mathcal{K}_D$, $\xi \leftarrow \{0,1\}$, $\chi \leftarrow \mathsf{DEM.Enc}_{dk}(m_\xi)$.

Step 3. $\tilde{\xi} \leftarrow A_D(\rho, \chi)$

The messages, m_0 and m_1 must be the same length. Let $\epsilon_{\mathrm{dem},A_D} = \left|\Pr[\tilde{\xi} = \xi] - \frac{1}{2}\right|$ and $\epsilon_{\mathrm{dem}} = \max_{A_D}(\epsilon_{\mathrm{dem},A_D})$ where maximum is taken over all machines. We say that a DEM is one-time secure if ϵ_{dem} is negligible in λ. One-time pad is a simple example that fulfills this security notion.

2.3 Public-Key Encryption (PKE)

A public-key encryption scheme consists of three algorithms, PKE.Gen, PKE.Enc, and PKE.Dec:

$(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$	A probabilistic algorithm that on input the security parameter λ, generates public and private keys (pk, sk). The public-key defines the message space $\mathcal{M}$.
$c \leftarrow \mathsf{PKE.Enc}_{pk}(m)$	A probabilistic algorithm that encrypts a message $m \in \mathcal{M}$ into a ciphertext c.
$m \leftarrow \mathsf{PKE.Dec}_{sk}(c)$	An algorithm that decrypts c. It outputs either $m \in \mathcal{M}$ or a special symbol $\perp \notin \mathcal{M}$. An obvious soundness condition applies.

Let A_E be a polynomial-time oracle machine that plays the following game. By $\mathcal{O}$, we denote the decryption oracle, $\mathsf{PKE.Dec}_{sk}(\cdot)$

[GAME.PKE]

Step 1. $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$

Step 2. $(m_0, m_1, \rho) \leftarrow {A_E}^{\mathcal{O}}(pk)$

Step 3. $b \leftarrow \{0,1\}$, $c \leftarrow \mathsf{PKE.Enc}_{pk}(m_b)$.

Step 4. $\tilde{b} \leftarrow {A_E}^{\mathcal{O}}(\rho, c)$

In Step 4, A_E is restricted not to ask c to $\mathcal{O}$. In addition, m_0 and m_1 must be of the same length. We define $\epsilon_{\mathrm{pke},A_E} = \left|\Pr[\tilde{b} = b] - \frac{1}{2}\right|$ and $\epsilon_{\mathrm{pke}} = \max_{A_E}(\epsilon_{\mathrm{pke},A_E})$ where maximum is taken over all ppt machines. We say that a PKE is CCA-secure if ϵ_{pke} is negligible in λ.

2.4 Key Encapsulation Mechanism (KEM)

This section describes the syntax and security definitions for KEM from Shoup [36].

$(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$	A probabilistic algorithm that generates public and private keys (pk, sk). The public-key defines the key space $\mathcal{K}_K$.
$(K, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$	A probabilistic algorithm that generates key $K \in \mathcal{K}_K$ and its encryption ϕ.
$K \leftarrow \mathsf{KEM.Dec}_{sk}(\phi)$	An algorithm that decrypts ϕ to recover K. As well as PKE, an obvious soundness condition applies. It may output a special symbol $\perp \notin \mathcal{K}_K$.

Since we use KEM only as a component to construct Tag-KEM in this paper, we consider $\mathsf{KEM.Enc}$ that outputs $K \in \mathcal{K}_K$ for some specific domain $\mathcal{K}_K$ rather than the ones adjusted to a specific DEM key-space.

Let $\mathcal{O}$ denote the decryption oracle, $\mathsf{KEM.Dec}_{sk}(\cdot)$. Let A be a ppt oracle machine that plays the following game.

[GAME.KEM]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $K_0 \leftarrow \mathcal{K}_K$, $b \leftarrow \{0, 1\}$.

Step 2. $\tilde{b} \leftarrow A^{\mathcal{O}}(pk, \phi, K_b)$.

In Step 2, A is restricted not to ask ϕ to $\mathsf{KEM.Dec}$.

We define $\epsilon_{\mathrm{kem},A} = \left|\Pr[\tilde{b} = b] - \frac{1}{2}\right|$ and $\epsilon_{\mathrm{kem}} = \max_A(\epsilon_{\mathrm{kem},A})$ where the maximum is taken over all machines. We say that a KEM is secure against adaptive chosen ciphertext attacks (CCA secure) if ϵ_{kem} is negligible in λ.

2.5 Message Authentication Code (MAC)

MAC is a pair of algorithms $(\mathsf{MAC.Sign}, \mathsf{MAC.Ver})$ and a key-space $\mathcal{K}_M$ defined by security parameter λ. Typically, $\mathcal{K}_M = \{0, 1\}^\lambda$. $\mathsf{MAC.Sign}$ takes a mac-key $mk \in \mathcal{K}_M$ and a message $\tau \in \{0, 1\}^*$ and outputs a string σ. We say (σ, τ) is valid with regard to mk if $\sigma = \mathsf{MAC.Sign}_{mk}(\tau)$. $\mathsf{MAC.Ver}$ takes a triple (mk, σ, τ) and outputs 1 if (σ, τ) is valid with respect to mk, or outputs 0, otherwise.

We require MAC to be secure against one-time chosen message attack; an adversary chooses an arbitrary message τ and is given its MAC σ created with a MAC key mk randomly chosen from $\mathcal{K}_M$. The adversary outputs (σ', τ') which is different from (σ, τ) as a pair. The adversary wins if the resulting pair is correct with respect to the same mk. Let $\mathsf{GAME.MAC}$ denote this attack game. We say that MAC is secure against one-time chosen message attack if any ppt adversary can win $\mathsf{GAME.MAC}$ at most with negligible probability, say ϵ_{mac}.

2.6 Key Derivation Function (KDF)

Our construction uses a function, say KDF_2, that maps a key K generated by KEM into a pair of keys (dk, mk) for DEM and MAC. We require its output distribution (dk, mk) to be indistinguishable from uniform, when the input K is uniformly distributed. We refer Section 8 of [18] for construction.

Since the input domain depends on a specific choice of KEM, DEM, and MAC, KDF is defined with regard to the key-space of these schemes. Let $\mathsf{KDF}_2 : \mathcal{K}_K \rightarrow \mathcal{K}_D \times \mathcal{K}_M$ and $\{\mathsf{KDF}_2\}_\lambda$ be a family of functions indexed by the key-spaces associated to the same security parameter λ. (Extra keys may also be used as index if needed.) We require that distribution of KDF_2 is indistinguishable from uniform over $\mathcal{K}_D \times \mathcal{K}_M$. Let

$$\begin{aligned} D_1 &= \{(dk, mk) \,|\, K \leftarrow \mathcal{K}_K, (dk, mk) \leftarrow \mathsf{KDF}_2(K)\}, \text{ and} \\ D_0 &= \{(dk, mk) \,|\, (dk, mk) \leftarrow \mathcal{K}_D \times \mathcal{K}_M\} \end{aligned}$$

We say that KDF_2 is secure if, for polynomial time machine A_{KDF},

$$|\Pr[b \leftarrow \{0,1\}, (dk, mk) \leftarrow D_b, \tilde{b} \leftarrow A_{\mathsf{KDF}}(pk, \mathsf{KDF}_2, (dk, mk))\,;\, \tilde{b} = b] - \frac{1}{2}| \leq \epsilon_{\mathrm{kdf}}$$

where ϵ_{kdf} is a negligible function in λ. The probability is taken over the choice of KDF_2 which includes coins of $\mathsf{KEM.Gen}$ that determines $\mathcal{K}_K$ and the choice of (dk, mk), b, and the coins of A_{KDF}.

In our construction of Tag-KEM, dk and mk are derived from independent application of KDF_2 to two different inputs. Let

$$\begin{aligned} U_1 &= D_1, \text{and} \\ U_0 &= \{(dk, mk) | K \leftarrow \mathcal{K}_K, (dk, *) \leftarrow \mathsf{KDF}_2(K), K' \leftarrow \mathcal{K}_K, (*, mk) \leftarrow \mathsf{KDF}_2(K')\}. \end{aligned}$$

Lemma 2.1 If KDF_2 is secure, for all polynomial machine A,

$$\left|\Pr[d \leftarrow \{0,1\}, (dk, mk) \leftarrow U_d, \tilde{d} \leftarrow A(dk, mk)\,;\, \tilde{d} = d] - \frac{1}{2}\right| \leq 2\epsilon_{\mathrm{kdf}}.$$

Proof: Consider a hybrid distribution

$$W = \{(dk, mk) | K \leftarrow \mathcal{K}_K, (dk, *) \leftarrow \mathsf{KDF}_2(K), mk \leftarrow \mathcal{K}_M\}.$$

By hybrid argument, the advantage of distinguishing D_1 and W is upper bounded by ϵ_{kdf}. Let ϵ'_{kdf} be advantage of distinguishing U_1 and U_0. Again by hybrid argument, the advantage of distinguishing U_0 and W is at least $\epsilon'_{\mathrm{kdf}} - \epsilon_{\mathrm{kdf}}$. Then, given a machine that distinguishes U_0 and W with that probability, one can easily construct a machine that distinguishes D_0 and D_1 with the same probability. Hence $\epsilon'_{\mathrm{kdf}} - \epsilon_{\mathrm{kdf}} \leq \epsilon_{\mathrm{kdf}}$. This completes the proof. ∎

By simple computation, we have:

Corollary 2.2 If KDF_2 is secure, for all polynomial machine A,

$$|\Pr[(dk, mk) \leftarrow U_0, 1 \leftarrow A(dk, mk)] - \Pr[(dk, mk) \leftarrow U_1, 1 \leftarrow A(dk, mk)]| \leq 4\epsilon_{\mathrm{kdf}}.$$

2.7 Target Collision-Free and Random Prefix Collision-Free

Target Collision-Free (TCH): Target Collision-Free is a special case of universal one-way; An adversary is given (H, x) (chosen at random in their domain) and then attempts to find x' such that $H(x) = H(x')$. Let $\mathcal{X}_\lambda = \{X\}$ be a collection of domains and $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$. Let $\mathcal{H}_\lambda = \{H : X \rightarrow \{0,1\}^\lambda \mid X \in \mathcal{X}_\lambda\}$ and $\mathcal{H} = \{\mathcal{H}_\lambda\}_{\lambda \in \mathbb{N}}$. Note that X is identified by the description of H. Let A_{TCH} be a machine playing the following game.

[GAME.TCH]

Step 1. $H \leftarrow \mathcal{H}_\lambda$, $x \leftarrow X$.

Step 2. $x' \leftarrow A_{TCH}(H, x)$ such that $x' \in X$.

A_{TCH} wins if $H(x') = H(x)$. We define $\epsilon_{\mathrm{tch},A_{TCH}} = \Pr[A_{TCH}\text{ wins.}]$ in GAME.TCH and $\epsilon_{\mathrm{tch}} = \max_{A_{TCH}}(\epsilon_{\mathrm{tch},A_{TCH}})$ where the maximum is taken over all machines. We say that $\mathcal{H}$ is target collision-free with regard to $\mathcal{X}$ if ϵ_{tch} is negligible in λ. For simplicity, we also say that $\mathcal{H}_\lambda$ (or even H) is target collision-free with regard to $\mathcal{X}_\lambda$ (or X, respectively).

Random Prefix Collision-Free (RPH): Random prefix collision-free is a notion in between target collision-free and collision-free; An adversary is first given H and finds x and then given random r and outputs r' and x' such that $H(r, x) = H(r', x')$. Let $\mathcal{X}_\lambda = \{X\}$ be a collection of domains and $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$. We define $\mathcal{R}_\lambda$ and $\mathcal{R}$ in the same way. Then, let $\mathcal{H}_\lambda = \{H : X \times R \to \{0,1\}^\lambda \mid X \in \mathcal{X}_\lambda, R \in \mathcal{R}_\lambda\}$ and $\mathcal{H} = \{\mathcal{H}_\lambda\}_{\lambda\in\mathbb{N}}$. Let A_{RPH} be a machine playing the following game named GAME.RPH.

[GAME.RPH]

Step 1. $H \leftarrow \mathcal{H}_\lambda$

Step 2. $(\rho, x) \leftarrow A_{RPH}(H)$

Step 3. $r \leftarrow R$

Step 4. $(r', x') \leftarrow A_{RPH}(\rho, r)$ such that $r' \in R$ and $x' \in X$.

A_{TCH} wins if $H(r', x') = H(r, x)$ and $(r', x') \neq (r, x)$. We define $\epsilon_{\mathrm{tch},A_{RPH}}$ and ϵ_{rph} for RPH as well as those for TCH, and say that $\mathcal{H}$ is random prefix collision-free with regard to $\mathcal{X}$ and $\mathcal{R}$ if ϵ_{rph} is negligible in λ.

3 Generic Construction of Hybrid PKE

In GAME.TKEM, it is important to see that the same ψ can be asked to the decryption oracle as long as τ is different. Therefore, to conform to CCA-security, the CCA-secure Tag-KEM must provide integrity to τ. We exploit this property to protect the DEM component via the tag, so as to be non-malleable.

Now in our construction of hybrid PKE, we require that Tag-KEM accepts any string as a tag, i.e., $\mathcal{T} = \{0,1\}^*$. First of all, PKE.Gen is the same as TKEM.Gen; Given security parameter λ, it outputs public-key pk and private-key sk. Encryption and decryption functions are as follows.

Function: $\mathsf{PKE.Enc}_{pk}(m)$	**Function:** $\mathsf{PKE.Dec}_{sk}(c)$
$(\omega, dk) \leftarrow \mathsf{TKEM.Key}(pk)$	$(\psi, \chi) \leftarrow c$
$\chi \leftarrow \mathsf{DEM.Enc}_{dk}(m)$	$dk \leftarrow \mathsf{TKEM.Dec}_{sk}(\psi, \chi)$
$\psi \leftarrow \mathsf{TKEM.Enc}(\omega, \chi)$	$m \leftarrow \mathsf{DEM.Dec}_{dk}(\chi)$
Output $c = (\psi, \chi)$	Output m

When the length of the DEM key varies depending on the length of the message, like one-time pad, the syntax of Tag-KEM will be modified so that TKEM.Enc and TKEM.Dec can take the necessary information.

Theorem 3.1 [Tag-KEM/DEM Composition Theorem] If the Tag-KEM is CCA secure and the DEM is one-time secure then the Hybrid PKE scheme in Section 3 is CCA secure. In particular, $\epsilon_{\text{pke}} < 2\epsilon_{\text{tkem}} + \epsilon_{\text{dem}}$.

Proof: We modify $\mathsf{PKE.Enc}$ in Step-3 of $\mathsf{GAME.PKE}$ so that $\mathsf{DEM.Enc}$ takes random key $dk^{\times}$ instead of the legitimate one generated by $\mathsf{TKEM.Key}$. Call this game $\mathsf{GAME.PKE}$'. Let X and X' be events that $\tilde{b} = b$ happens in $\mathsf{GAME.PKE}$ and $\mathsf{GAME.PKE}$', respectively. Then, we claim that $|\Pr[X] - \Pr[X']| \leq 2\,\epsilon_{\text{tkem}}$, which is shown by constructing A_T that attacks the underlying Tag-KEM scheme by using A_E. First A_T is given public-key pk and passes it to A_E. A_T then requests dk_δ to the encryption oracle of $\mathsf{GAME.TKEM}$. Given m_0 and m_1 from A_E, A_T selects $b \leftarrow \{0,1\}$ and computes $\chi = \mathsf{DEM.Enc}_{dk_\delta}(m_b)$. It then sends χ to $\mathsf{TKEM.Enc}$ as a tag and receives ψ. Ciphertext (ψ, χ) is then sent to A_E. Every decryption query from A_E is forwarded to decryption oracle $\mathsf{TKEM.Dec}$. If $\perp$ is returned, it is forwarded to A_E. Otherwise, A_K decrypts χ by using the key given from oracle $\mathsf{TKEM.Dec}$ and pass the resulting message to A_E. When A_E outputs $\tilde{b} = b$, A_K outputs $\tilde{\delta} = 1$ meaning that dk_δ is the real key. Otherwise, if A_E outputs $\tilde{b} \neq b$, A_K outputs $\tilde{\delta} = 0$ meaning that dk_δ is random. Now observe that the view of A_E is identical to that in $\mathsf{GAME.PKE}$ when $\delta = 1$, and that in $\mathsf{GAME.PKE}$' when $\delta = 0$. Accordingly, $\Pr[\tilde{b} = b|\delta = 1] = \Pr[X]$ and $\Pr[\tilde{b} = b|\delta = 0] = \Pr[X']$. Therefore,

$$\begin{aligned}
\Pr[\tilde{\delta} = \delta] - \frac{1}{2} &= \frac{1}{2}(\Pr[\tilde{\delta} = 1|\delta = 1] - \Pr[\tilde{\delta} = 1|\delta = 0]) \\
&= \frac{1}{2}(\Pr[\tilde{b} = b|\delta = 1] - \Pr[\tilde{b} = b|\delta = 0]) \\
&= \frac{1}{2}(\Pr[X] - \Pr[X'])
\end{aligned}$$

Since $\left|\Pr[\tilde{\delta} = \delta] - \frac{1}{2}\right| \leq \epsilon_{\text{tkem}}$, we have $|\Pr[X] - \Pr[X']| \leq 2\epsilon_{\text{tkem}}$.

Next, we show that A_E playing $\mathsf{GAME.PKE}$' essentially conducts a passive attack to DEM, i.e., $\left|\Pr[X'] - \frac{1}{2}\right| \leq \epsilon_{\text{dem}}$. It is shown by constructing A_D that plays $\mathsf{GAME.DEM}$ by using A_E. A_D first generates (pk, sk) by using $\mathsf{PKE.Gen}$ and gives pk to A_E. When m_0 and m_1 are given from A_E, A_D forwards them to encryption oracle of $\mathsf{GAME.DEM}$ and receives cipehrtext χ. It then computes ψ by following $\mathsf{TKEM.Key}$ and $\mathsf{TKEM.Enc}$ by using χ as a tag, and sends $c = (\psi, \chi)$ to A_E. Note that the key chosen by the encryption oracle of $\mathsf{GAME.DEM}$ and the one embedded in ψ are independent and randomly chosen. All decryption queries are correctly processed by using sk. When A_E outputs $\tilde{b}$, A_D outputs $\tilde{\xi} = \tilde{b}$. It is now easy to see that, in this construction, $\mathsf{GAME.PKE}$' is perfectly simulated and whenever A_E wins, so does A_D. Hence $\left|\Pr[X'] - \frac{1}{2}\right| \leq \epsilon_{\text{dem}}$. The major factors of the running time of A_D is that of A_E and that for simulating the decryption oracle which grows linearly in the number of decryption queries.

In summary, we have:

$$\begin{aligned}
|(\Pr[X] - \frac{1}{2}) - (\Pr[X'] - \frac{1}{2})| &\leq 2\epsilon_{\text{tkem}} \\
\epsilon_{\text{pke}} - \epsilon_{\text{dem}} &\leq 2\epsilon_{\text{tkem}} \\
\epsilon_{\text{pke}} &\leq 2\epsilon_{\text{tkem}} + \epsilon_{\text{dem}}
\end{aligned}$$

where ϵ_{tkem} and ϵ_{dem} are assumed negligible. ▮

4 Construction of Tag-KEM

This section develops some methods for obtaining Tag-KEM from PKE or KEM. (Note that KEM is generally obtained from PKE. Hence starting from a KEM is more generic.) Since some methods are available to convert a weak PKE to a CCA-secure one in various setting, we assume CCA-secure PKE is available. Construction of KEM directly from weaker components is studied in [19].

4.1 Based on PKE with Long Plaintext

When CCA-secure PKE is available, the first idea would be to encrypt the tag as a part of the plaintext together with the DEM key to encapsulate. It indeed works well if there is enough space in a plaintext. Lengthy tags would be compressed by using a hash function. We show that a target collision-free hash function (see Section 2.7) is sufficient for this purpose. Formally, we construct Tag-KEM from PKE as follows. $\mathsf{TKEM.Gen}$ is essentially the same as $\mathsf{PKE.Gen}$; It outputs (pk, sk). It also selects hash function H. (For notational simplicity, we assume that H is included in pk and sk.) $\mathsf{TKEM.Key}$ chooses random dk from $\mathcal{K}_D$. It also outputs state information $\omega = pk||dk$. The encryption and decryption functions are as follows.

Function: $\mathsf{TKEM.Enc}(\omega, \tau)$	**Function:** $\mathsf{TKEM.Dec}_{sk}(\psi, \tau)$
$(pk, dk) \leftarrow \omega$	$dk\|\|\tau' \leftarrow \mathsf{PKE.Dec}(sk, \psi)$
$\tau' = H(\tau)$	If $\tau' = H(\tau)$, return dk.
$\psi = \mathsf{PKE.Enc}_{pk}(dk\|\|\tau')$	Return $\perp$, otherwise.
Output ψ.	

Let ϵ_{tch} be the success probability of finding a collision of H. (Formal security definitions and related notations are given in Section 2.7.)

Theorem 4.1 If PKE is CCA-secure and H is target collision-free, the above Tag-KEM is CCA-secure. In particular, $\epsilon_{\text{tkem}} \leq \epsilon_{\text{pke}} + \epsilon_{\text{tch}}$.

Proof is given in Appendix A. One efficient implementation would be to use Rabin-SAEP+ [8] encryption, where the message length is known to be shorter than that of RSA but sufficient for encrypting a standard DEM key and hashed tag. One can also apply the technique of [25] to shorten the ciphertext.

4.2 Based on CCA-Secure KEM and MAC

In this section we present a CCA-secure Tag-KEM based on a CCA-secure KEM and a secure message authentication code (MAC) scheme (see Appendices 2.4 and 2.5 for formal definitions of these tools).

The idea is to encrypt a random key K using the KEM, and derive from K two keys dk, mk. The first, dk is the actual encrypted key, while mk is used to MAC the tag. The resulting MAC is appended to the ciphertext. A decryptor not only checks that the KEM decryption is correct, but also checks (using the decrypted key mk) that the MAC on the tag is correct. A formal description follows.

Construction of Tag-KEM: Let $\Pi_L = (\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$ be a KEM. Let $\mathsf{MAC} = (\mathsf{MAC.Sign}, \mathsf{MAC.Ver})$ be a MAC. Let $\mathsf{KDF}_2 : \mathcal{K}_K \rightarrow \mathcal{K}_D \times \mathcal{K}_M$ be a key derivation function where $\mathcal{K}_D$ is the key-space of DEM and $\mathcal{K}_M$ is the key-space of MAC. By using these components, we construct a Tag-KEM as follows. $\mathsf{TKEM.Gen}$ is the same as $\mathsf{KEM.Gen}$; It outputs (pk, sk).[1] $\mathsf{TKEM.Key}$ is that, given pk, it computes $(K, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$ and $(dk, mk) \leftarrow \mathsf{KDF}_2(K)$. Then it outputs dk and state information $\omega = (mk, \phi)$. The encryption and decryption functions are as follows.

Function: $\mathsf{TKEM.Enc}(\omega, \tau)$

$(mk, \phi) \leftarrow \omega$
$\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$
Output $\psi = (\phi, \sigma)$

Function: $\mathsf{TKEM.Dec}_{sk}(\psi, \tau)$

$(\phi, \sigma) \leftarrow \psi$
$K \leftarrow \mathsf{KEM.Dec}_{sk}(\phi)$
$(dk, mk) \leftarrow \mathsf{KDF}_2(K)$
If $K = \perp$ or $\mathsf{MAC.Ver}_{mk}(\sigma, \tau) \neq 1$, output $\perp$.
Otherwise, output dk.

Clearly the CCA security of the KEM scheme will prevent an adversary from gaining any advantage by manipulating the KEM ciphertext. On the other hand the security of the MAC will prevent an adversary from gaining any advantage by manipulating the MAC.

Applying Theorem 3.1 to the above Tag-KEM yields the same hybrid encryption scheme as in Shoup's KEM/DEM framework when the DEM part is implemented by following the encrypt-then-MAC paradigm. But by looking at that scheme in a different light, we are able to proceed a step further in refining the assumptions and the efficiency, as shown in the next section.

4.3 Based on weak KEM and MAC

In the previous scheme, there is some redundancy at play. If a KEM is combined with a MAC as shown in Section 4.2, the MAC will be used to preserve the integrity of ciphertexts. Accordingly, one may no longer need the KEM's functionality of verifying ciphertexts. Following this intuition, in this Section we formally describe a new security notion of KEM that can be strictly weaker than CCA but sufficient to yield CCA-secure Tag-KEM when combined with MAC.

Predicate-dependent CCA Security: Let Π_L be a KEM as in Section 4.2. Let $\mathcal{P} : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}$ be a poly-time computable predicate. Let $\mathcal{VD}_\phi$ be a restricted decryption oracle that is specific to challenge ϕ. It takes $(\phi_i, \eta_i) \in \{0,1\}^* \times \{0,1\}^*$ and outputs $\mathsf{KEM.Dec}_{sk}(\phi_i)$ if $\phi_i \neq \phi$ and $\mathcal{P}(\mathsf{KEM.Dec}_{sk}(\phi_i), \eta_i) = 1$. It outputs $\perp$, otherwise. Let A_L be an adversary attacking Π_L. We define $\mathsf{GAME.LKEM}$ by modifying $\mathsf{GAME.KEM}$ so that the decryption oracle receives ciphertexts accompanied by an arbitrary string and the decryption oracle $\mathsf{KEM.Dec}_{sk}(\phi)$ is replaced with $\mathcal{VD}$.

[GAME.LKEM]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $K_0 \leftarrow \mathcal{K}_K$, $b \leftarrow \{0,1\}$.

Step 2. $\tilde{b} \leftarrow A_L^{\mathcal{VD}_\phi(\cdot,\cdot)}(pk, \phi, K_b)$

[1] If KDF_2 requires a key, it is generated in $\mathsf{TKEM.Gen}$ and included to pk and sk. See Section 2.6 for details of KDF.

We define $\epsilon_{\mathrm{lkem},A_L} = \left|\Pr[\tilde{b} = b] - \frac{1}{2}\right|$ and $\epsilon_{\mathrm{lkem}} = \max_{A_L}(\epsilon_{\mathrm{lkem},A_L})$ where the maximum is taken over all machines. We say that a KEM is LCCA secure with respect to predicate $\mathcal{P}$ if ϵ_{lkem} is negligible in λ.

The above definition may seem too generic since the useful case shown later is where $\mathsf{MAC.Ver}$ can be used as $\mathcal{P}$. Nevertheless, the generic treatment may be helpful to specify what property is really needed. Also in some cases, it makes the security analysis slightly simpler like shown in our analysis of the Kurosawa-Desmedt scheme.

When does LCCA become weaker than CCA? The strength of LCCA security is subject to the property of $\mathcal{P}$. If $\mathcal{P}$ outputs 1 for any input, LCCA is clearly equivalent to CCA. If it outputs 0 for any input, LCCA is equivalent to a passive attack, i.e., an attack without the decryption oracle, for which we cannot prove the security of Tag-KEM in Section 4.2. Hence a very weak (i.e. only passively secure) instance may exist in the class.

Proof of the Tag-KEM in Section 4.2. We prove the security of the construction when the underlying KEM is LCCA secure with respect to $\mathcal{P}^{\mathrm{mac}}$ defined as $\mathcal{P}^{\mathrm{mac}}(K, (\sigma, \tau)) = \mathsf{MAC.Ver}_{mk}(\sigma, \tau)$ where mk is $(dk, mk) \leftarrow \mathsf{KDF}_2(K)$. Note that, in the Tag-KEM construction shown in Section 4.2, $\mathcal{P}$ is not used in $\mathsf{TKEM.Dec}$. Hence the underlying KEM Π_L itself might be insecure against CCA as mentioned above. However, since $\mathcal{P}$ is assumed to be $\mathcal{P}^{\mathrm{mac}}$ and it is indeed provided from outside, LCCA security will be achieved. Namely, the MAC has two different roles in the construction; one is to authenticate the tag and the other is to work as a predicate as a part of underlying KEM. As we could have predicted, this is very close to the combination of CCA KEM and MAC (but not exactly the same). Nevertheless, we have to formally prove the security to see the MAC plays the different roles without inconsistency.

Theorem 4.2 If Π_L is LCCA secure with respect to $\mathcal{P}^{\mathrm{mac}}$ then the Tag-KEM defined in Section 4.2 is CCA secure. In particular, $\epsilon_{\mathrm{tkem}} \leq 4\epsilon_{\mathrm{lkem}} + q_D\,\epsilon_{\mathrm{mac}} + 5\epsilon_{\mathrm{kdf}}$ where q_D is the maximum number of decryption queries.

Proof is in Appendix B. We note that the result in this section might be regarded as theoretical. In practice, proving that a KEM conforms to the new notion could only be slightly easier than proving the security of resulting scheme as Tag-KEM. And one can expect better reduction cost by directly proving the security of Tag-KEM by exploiting specific properties.

We finally remark that one can also construct CCA-secure Tag-KEM from RCCA-secure KEM which is strictly weaker than CCA-secure ones. See Section 5.4 for further discussion.

4.4 Based on KEM with Hash function

We show another approach that might be available when your PKE does not have enough plaintext length as needed in Section 4.1 and/or increasing ciphertext length as in Section 4.2 is not acceptable.

If a KEM uses a hash function, probably for verifying ciphertexts, the KEM may be converted to a Tag-KEM simply by including the tag into the hash function input. This approach is correct if the hash function is involved in the scheme in a 'meaningful' way and provides 'sufficient' security. Although a generic construction that follows formal versions of these intuitive terms can be shown, it does not seem quite useful due to its complexity. Showing that a KEM fits into the generic framework may not be simpler than directly proving that the resulting Tag-KEM scheme is secure. Indeed, in all cases we have in mind, the security proof is essentially unchanged from that of the original KEM (or PKE). Therefore, we only show two concrete constructions of Tag-KEM based on well known encryption schemes; OAEP+ [37] and Cramer-Shoup encryption [16].

In the following, the description of the original schemes are obtained just by dropping the tag τ.

4.4.1 From OAEP+.

Let f be a one-way trapdoor permutation. OAEP+ encrypts dk with tag τ into ciphertext ψ in the following way:

$$r' = H'(r||dk||\tau),\ s = (G(r) \oplus dk)||r',\ w = H(s) \oplus r,\ \psi = f(s||w)$$

where r and r' are random and G, H, H' are random oracles [4].

Security is argued in the same way as the original one except the case that, for challenge ciphertext (ψ, τ) the adversary finds another valid ciphertext (ψ, τ'). Since ψ uniquely identifies r, r' and K, (ψ, τ') is valid only if $H'(r||dk||\tau) = H'(r||dk||\tau')$ holds. When H' outputs a k_1-bit string, such an event happens with probability at most $q_{H'}\, 2^{-k_1}$ where $q_{H'}$ is the maximum number of queries to H'. Based on this observation, we define game $\mathsf{GAME.0'}$ where decryption oracle returns $\perp$ for all queries that differs only in the tag part with the challenge ciphertext. The rest of the security proof is done in the same way as in the original paper [37] except for obvious modifications. Accordingly, only $q_{H'}\, 2^{-k_1}$ is an extra reduction cost to that of OAEP+.

4.4.2 From Cramer-Shoup Encryption.

A Tag-KEM scheme based on Cramer-Shoup encryption over a multiplicative group, say G_q, of prime order q is the following. A private-key is $(x_1, x_2, y_1, y_2, z_1, z_2) \in Z_q$ and the public-key is $g_1, g_2 \leftarrow G_q^2$, and $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $h = g_1^{z_1} g_2^{z_2}$. The encryption function yields $dk = h^r$ where r is random, and ciphertext (u_1, u_2, v) such that

$$u_1 = g_1^r,\ u_2 = g_2^r,\ \alpha = H(u_1||u_2||\tau),\ v = c^r d^{\alpha r}$$

where H is a hash function. Decryption first checks if $v \stackrel{?}{=} u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$ and then recovers $dk = u_1^{z_1} u_2^{z_2}$. Applying Theorem 3.1 results in the hybrid PKE briefly mentioned in [16].

In [18], the security of Cramer-Shoup encryption requires H to be Target Collision-Free (as defined in Section 2.7) since, without τ, all inputs to the hash function, i.e., u_1 and u_2, are chosen randomly by the encryption oracle. In this case, however, τ is chosen by the adversary after seeing the description of H. Hence we require H to be Random Prefix Collision-Free as formally defined in Section 2.7.

It holds that (Collision-Free) $\Rightarrow$ (Random Prefix Collision-Free) $\Rightarrow$ (Target Collision-Free). Hence it is reasonable to use cryptographic hash functions like SHA-1 which can be assumed collision-free. Nevertheless, we stress that random prefix collision-freeness may not necessarily be equivalent to collision-free because, for example, it is not clear how to perform a birthday attack in the above game (if the randomness of x affects to the output). Theoretically, we do not know constructions of random prefix collision-free hash functions from target collision-free or universal one-way hash functions, thus we resort to strong collision-freeness. The only drawback is that this requires a longer output (about twice as much because the birthday paradox applies here), but that does not affect our construction.

4.5 Based on ID-based PKE

An ID-based encryption scheme is selective-ID secure when it is secure against chosen ciphertext and chosen ID attacks provided that the target ID is committed at the beginning and the ID must not be included in any decryption query. It is shown in [14] that selective-ID ID-based encryption schemes (sIBE in short) can be strengthened to a full CCA-secure PKE by using strong one-time signature. In [11], Boneh and Katz improved the efficiency of [14] by replacing the one-time signature with a commitment scheme (using hash function) and a MAC. We show that the conversion from sIBE to full-CCA PKE also yields a CCA-secure Tag-KEM (without adopting the result of Section 4.1).

Let $(\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Ver})$ be a strong one-time signature scheme where $\mathsf{SIG.Gen}$ is a key generation algorithm, $\mathsf{SIG.Sign}$ is a signature generation algorithm, and $\mathsf{SIG.Ver}$ is a signature verification algorithm. Let $\mathsf{sIBE.Enc}(pk, \mathrm{ID}, m)$ be the encryption function of an sIBE. Then, we construct a Tag-KEM scheme as follows: $\mathsf{TKEM.Key}(pk)$ simply selects dk randomly. Then $\mathsf{TKEM.Enc}$ encrypts dk and τ into ciphertext $\psi = (vk, \phi, \sigma)$ by computing

$$(vk, sk) \leftarrow \mathsf{SIG.Gen}(1^\lambda),\ \phi \leftarrow \mathsf{sIBE.Enc}(pk, vk, dk),\ \sigma = \mathsf{SIG.Sign}(sk, \phi || \tau).$$

Decryption is rather trivial; first verify the signature σ using vk and then decrypt the rest of the parts.

In the above, removing τ from the description results in the original scheme from [14]. Including τ into the message to be signed provides integrity to the tag without affecting the security of the original scheme. Indeed, the security proof of the above scheme is almost the same as in [14] with obvious modification. The reduction cost does not change, either.

The length of a ciphertext of the above Tag-KEM is the same as the original CCA-secure PKE. But the resulting hybrid PKE may yield shorter ciphertext thanks to the one-time secure DEM that typically yield shorter ciphertext than CCA-secure ones needed to combine with the original CCA-secure PKE.

One can extend the above Tag-KEM to ID-based one in the same way starting from a 2-level hierarchical IBE that is selective-ID secure in the second level and fully CCA secure in the first level. (A given ID is assigned to the first-level ID and vk is assigned to the second-level ID.) ID-based KEM is also studied in [6]. For efficient implementations of sIBE based on standard cryptographic assumptions, we refer to [9].

5 Applications

In this section, we show how our framework yields new hybrid encryption schemes, captures some known schemes, and even finds ways to improve them.

5.1 Threshold Hybrid PKE

Roughly, a threshold (hybrid) PKE is a PKE whose decryption $m \leftarrow \mathsf{PKE.Dec}_{sk}(c)$ is implemented by a multi-party protocol. The private key sk is shared among n decryption servers and they cooperatively compute m from given c without revealing anything but m (or $\perp$ for invalid c) in the presence of an adversary that can corrupt at most $k-1$ decryption servers. For simplicity, we assume that a trusted party generated the key, and shared it among the servers, though distributed key generation protocols can be used.

Threshold CCA-security is defined as a natural extension of the CCA-security for regular (non-threshold) PKE as in [39]. The decryption oracle is replaced by n decryption servers and the adversary is allowed to corrupt up to $k-1$ of them. A corrupted player provides all its view to the adversary and is completely controlled by the adversary.

Results from general multi-party computation, e.g., [26, 5], imply that any (hybrid) PKE can be converted to its threshold version in several settings. Since such a generic conversion suffers from unrealistic complexity, dedicated construction has been pursued starting from [20]. In the standard model, the first CCA-secure threshold PKE is presented in [13] followed by, e.g., [1, 28, 2, 10, 12]. However, no efficient threshold *hybrid* PKE, is known in the standard model, via a generic construction like KEM/DEM. If a *threshold* CCA KEM and a *threshold* CCA DEM are available, their simple combination would yield a threshold CCA PKE like the standard KEM/DEM composition. However, an efficient threshold DEM seems difficult to obtain, given its use of symmetric key techniques such as block ciphers and MAC.

Can we then combine a threshold CCA KEM and a standard, i.e., non-threshold, CCA DEM to obtain a CCA-secure hybrid PKE? Unfortunately, this also seems quite unlikely. A rough argument is the following. Assume an adversary that corrupts at least one decryption server. Given a challenge ciphertext (ϕ, χ), the adversary creates a random χ' and sends (ϕ, χ') to the decryption servers. The decryption servers work on ϕ to decrypt dk. Since the DEM is not threshold, the key dk must be known in its entirety to the servers (at least to one of them, the one who performs the DEM decryption). The adversary then will recover dk by corrupting at least one server, and then will correctly decrypt χ to win the CCA game.

The Tag-KEM/DEM framework offers an attractive way to get around this difficulty. We exploit the feature that the DEM part needs only be CPA-secure and the session-key can be securely exposed. Remember that CPA-secure DEM can be implemented by a one-time pad that leaks the key on decryption. Hence revealing the decryption key dk as a result of decrypting ψ does not impact security in the Tag-KEM/DEM framework. Accordingly, by replacing Tag-KEM with its *threshold* version, we have a *threshold* Tag-KEM/DEM framework. Namely, the combination of threshold Tag-KEM and standard one-time secure DEM results in CCA-secure threshold hybrid PKE.

A formal security definition of threshold Tag-KEM can be derived from the definition of threshold PKE in [39]. The following composition theorem can be proven by translating the proof of Theorem 3.1 to the threshold setting.

Theorem 5.1 [Threshold Tag-KEM/DEM Composition Theorem] If the threshold Tag-KEM is threshold-CCA secure and the DEM is one-time secure then their Tag-KEM/DEM composition yields a threshold-CCA secure hybrid PKE scheme. In particular, $\epsilon_{\text{th-pke}} < 2\epsilon_{\text{th-tkem}} + \epsilon_{\text{dem}}$ where $\epsilon_{\text{th-pke}}$ and $\epsilon_{\text{th-tkem}}$ are the advantages of threshold PKE and threshold KEM, respectively.

A note on the security model. In the above threshold Tag-KEM/DEM construction, the adversary can obtain a correct session-key by querying a valid ciphertext to honest decryption servers. (One-time pad DEM trivially exposes the session-key from a ciphertext and a message, though this is not true for arbitrary DEM.) Such information is irrelevant to conform to the game-based security definition for threshold PKE [39] but becomes an obstacle when a simulation-based security definition [13] is concerned. Roughly, the simulation-based security of [13] compares a threshold PKE with an ideal encryption system managed by a trusted party and states that the threshold PKE is secure if the adversary in the ideal model can be simulated by using the adversary in the real threshold model. It is claimed in [13] that the simulation-based security implies the game-based one but the reverse does not hold. According to the simulation-based security, the adversary in the real threshold model should obtain nothing but a message when a valid ciphertext is sent to the decryption servers since the ideal encryption is defined so. Since the schemes based on the threshold Tag-KEM/DEM composition reveals the session-key, it does not match to this security notion. Since this problem essentially comes from the use of a non-threshold DEM, it is highly unlikely that the simulation-based security is achieved unless the DEM is shared.

Instantiation. Threshold Cramer-Shoup PKE which is CCA-secure against static adversaries is shown in [13, 1], and the conversion technique in Section 4.4 (or result of Section 4.1 with larger security parameter) can be used to obtain a threshold Cramer-Shoup Tag-KEM. Accordingly, by following Theorem 5.1, one can have a secure threshold hybrid PKE scheme in the standard model. Adaptive security can be achieved as well based on the adaptively secure threshold Cramer-Shoup encryption of [2].

5.2 Revisiting the Kurosawa-Desmedt Scheme

In [30], Kurosawa and Desmedt introduced a hybrid encryption scheme based on Cramer-Shoup encryption. The private-key $sk = (x_1, x_2, y_1, y_2)$ and public-key $pk = (g_1, g_2, c, d)$ are a part of that for Cramer-Shoup encryption as shown in Section 4.4. Encryption of message $m \in \{0,1\}^*$ is :

$$u_1 = g_1^r,\ u_2 = g_2^r,\ \alpha = H(u_1||u_2),\ v = c^r d^{\alpha r},\ (dk, mk) \leftarrow \mathsf{KDF}_2(v),$$

$$\chi = G(dk) \oplus m,\ \sigma = \mathsf{MAC.Sign}_{mk}(\chi),$$

where r is random, H is a target collision-free hash function, G is a pseudo-random bit generator, and $\mathsf{MAC.Sign}$ is a MAC generation function. The ciphertext is (u_1, u_2, χ, σ). In this scheme, (u_1, u_2) is considered as the KEM part and (χ, σ) is considered as the CCA-secure DEM part. Though the combination results in a CCA-secure hybrid PKE, the KEM part is not CCA [27].

Our framework reveals another approach to the analysis of the scheme. That is, we consider (u_1, u_2, σ) as the Tag-KEM part and χ as the one-time secure DEM part. The Tag-KEM part is further decomposed to KEM part, (u_1, u_2) and MAC, (σ). It is known that this KEM is not CCA secure [27]. Hence it does not fulfill the requirement stated in Section 4.2. Yet we can prove that (u_1, u_2) constitutes an LCCA secure KEM with regard to a predicate $\mathcal{P}^{\mathrm{mac}}(K = v, \eta = (\chi, \sigma))$. See Appendix C for a proof. Accordingly, the Kurosawa-Desmedt scheme can be thoroughly explained by our framework and their design approach is validated.

5.3 Refined Fujisaki-Okamoto Conversion and More

Fujisaki-Okamoto Conversion: We revisit the Fujisaki-Okamoto conversion [23] that provides secure construction of hybrid encryption in the random oracle model. By fitting their scheme into our framework, we can see that one of their assumptions can be eliminated and a refined version is obtained without loss of efficiency.

Let $\mathsf{PKE.Enc}_{pk}(\cdot\,;\cdot)$ be a public-key encryption function where the last argument denotes the random coins used in the function. The Fujisaki-Okamoto conversion combines PKE and DEM by using two random oracles, H and G, as follows:

$$\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K; H(K||m)), \chi \leftarrow \mathsf{DEM.Enc}_{G(K)}(m).$$

The ciphertext is (ψ, χ). The resulting hybrid PKE is CCA-secure if PKE is one-way and DEM is one-time secure and $\mathsf{DEM.Enc}$ is a bijection between ciphertexts and messages for every fixed key.

Now one can observe that $\mathsf{PKE.Enc}_{pk}(K; H(K||\tau))$ works as a Tag-KEM encryption function that encapsulates the DEM key $G(K)$. Then, according to our framework, we have a slightly modified hybrid encryption:

$$\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K; H(K||\chi)), \chi \leftarrow \mathsf{DEM.Enc}_{G(K)}(m)$$

which does not require $\mathsf{DEM.Enc}$ to be a bijection. Details are given in Appendix D.

Bellare-Rogaway Scheme: The scheme shown by Bellare and Rogaway in [4] is a special case of the Fujisaki-Okamoto construction. The encryption function consists of a one-way permutation f and random oracles H and G;

$$\psi = f(r), \sigma = H(r||m), \chi = G(r) \oplus m$$

This scheme specifies to use one-time pad for the DEM part. According to our framework, we can generalize to any one-time secure DEM by modifying the scheme as

$$\psi = f(r), \sigma = H(r||\chi), \chi = \mathsf{DEM.Enc}_{G(r)}(m).$$

REACT: REACT-RSA [34] is very similar to the above Bellare-Rogaway scheme;

$$\psi = f(r), \sigma = H(r||m||\psi||\chi), \chi = \mathsf{DEM.Enc}_{G(r)}(m),$$

where f is the RSA encryption function. In this case, our framework shows that m can be removed from the inputs to H. Including m to H would result in slightly better reduction in the security proof. But removing it yields more benefit in computation when m is very long. Even ψ can be removed if the decryption function verifies that ψ is in the correct domain. In the case of RSA, domain checking is done just by comparing the ciphertext to the modulus. Hence by setting $\sigma = H(r||\chi)$ we have more efficient scheme. Indeed, the resulting scheme is the same as the modified Bellare-Rogaway scheme shown in this section.

The common factor lying underneath the above-mentioned examples is the Tag-KEM scheme whose ciphertext is $\psi = (f(r), H(r||\tau))$ where H is a random oracle. Such a scheme also appears in [31].

Some ISO standard candidates: Finally, as mentioned in Section 4.1, KEM schemes based on RSA and HIME described in [38] allow to label each ciphertext. This label can be used as a tag in our framework. Hence the DEM no longer need to provide CCA security when combined with those KEMs as suggested by our framework.

5.4 Revisiting RCCA-secure PKE

This section revisits RCCA-secure PKE in [15] and show that their construction of CCA-secure hybrid PKE from RCCA-secure PKE can be improved by following our Tag-KEM/DEM framework.

The notion of RCCA-secure PKE is introduced in [15]. RCCA is a variant of CCA where the decryption oracle returns a special nonce ’test’ when it receives a ciphertext that yields one of the questioned message, m_0 and m_1. Accordingly, even if the adversary can tweak the challenge ciphertext without affecting the embedded plaintext (such a feature is called benign-malleability [38]), sending it to the decryption oracle will give no advantage to the adversary in determining which of the questioned messages is hidden there. 'R' stands for 'replayable' in this sense. RCCA-security is a strict relaxation of CCA-security and proven useful for several cryptographic tasks, though, currently, there is no known instance of RCCA-secure PKE that is more efficient than known CCA-secure ones.

In [15], it is shown that combining RCCA-secure PKE and CCA-secure symmetric encryption can yield CCA-secure hybrid PKE. Suppose that a CCA-secure symmetric encryption is made by combining passively secure DEM and one-time MAC. Then, their construction is summarized as follows. Given message m, output ciphertext (ϕ, χ, σ) such that;

$$\phi \leftarrow \mathsf{PKE.Enc}_{pk}(dk||mk),\ \chi \leftarrow \mathsf{DEM.Enc}_{dk}(m||\phi),\ \sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\chi)$$

where dk and mk, are chosen randomly from appropriate domains. It is stressed that ϕ is encrypted by DEM and this double-encryption structure is essential in their security proof. Due to this special structure, the construction does not fit into our framework. Below, we show a slightly more efficient variant that avoids double encryption and fits into our framework.

$$\phi \leftarrow \mathsf{PKE.Enc}_{pk}(dk||mk),\ \chi \leftarrow \mathsf{DEM.Enc}_{dk}(m),\ \sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\chi||\phi)$$

Intuitively, applying MAC to ϕ offsets the benign-malleability of ϕ. The modified scheme yields shorter ciphertext and needs less computation.

From the above, we derive a Tag-KEM scheme which is summarized as follows.

$$(K, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}(), \quad (dk, mk) \leftarrow \mathsf{KDF}_2(K), \quad \sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau||\phi)$$

It can be seen as a variant of the construction shown in Section 4.2; MAC is applied to $\tau||\phi$ rather than to τ. In Appendix E, we give definition of RCCA-security for KEM, which is an analogue notion of that for PKE, and prove that the above Tag-KEM is CCA-secure if KEM is RCCA-secure. Hence, according to Theorem 3.1, the modified hybrid PKE is CCA-secure. This uncovers the redundancy of the double-encryption in the original construction and obtains a more efficient scheme.

6 Conclusions and Open Problems

We presented a new framework for constructing hybrid encryption by extending the known CCA KEM/DEM framework. The new Tag-KEM/DEM framework yields better schemes especially when it comes to ciphertext length and captures a wide variety of schemes. In addition several schemes can be improved by bringing them in our framework.

Yet there are some situations where the traditional CCA KEM/DEM framework is useful. For instance, schemes that follow the CCA KEM/DEM framework are better suitable for streaming applications where the receiver does not need to buffer the entire ciphertext. Tag-KEM/DEM schemes generally require a more flexible access to the ciphertext[2]. We also note that some Tag-KEM/DEM schemes provide the streaming feature if needed (The scheme based on Cramer-Shoup shown in Section 4.4 is an example). It is also known that the CCA KEM/DEM framework can be extended to establish some limited form of secure channels [32] (where no forward security is considered) while such extension is not available in Tag-KEM/DEM.

[2]Note that, however, streaming encryption/decryption does not necessarily allow the receiver to use a partial plaintext because the CCA DEM usually verifies integrity at the end of decryption.

Finally, we list some open problems as follows.
(On the Tag-KEM security) Can the security of Tag-KEM be weakened? Although in our definition of CCA security of Tag-KEM the tag is chosen by the adversary, once the Tag-KEM is combined with the DEM, the adversary cannot select an arbitrary tag any more since the tag is a ciphertext encrypted with a random key. Especially, if the DEM provides the strong property that the ciphertexts are indistinguishable from random strings of the same length, replacing the ciphertext with a random string offsets the choice of the adversary and target-free hash functions seem to suffice for the construction. This observation seems to suggest that we could weaken the security requirement for Tag-KEM in such a way that the tag is chosen randomly rather than chosen by the adversary. Can Theorem 3.1 hold in such a case?
(On the necessity of stronger hash functions) Is a random prefix collision-free hash function unavoidable in the construction shown in Section 4.4? This questions is closely related to the previous one. If the tag is chosen randomly rather than chosen by the adversary, universal one-way hash functions will suffice.
(More on the Random Prefix Collision-Free) More study is needed about random prefix collision-free hash functions. It would be interesting to show constructions from other primitives, especially from one-way permutations (or alternatively the impossibility of a black-box version of such a construction).

Acknowledgments

The authors would like to thank Hugo Krawczyk, Shai Halevi, Yevgeniy Dodis, Victor Shoup and Eiichiro Fujisaki for valuable discussion.

References

[1] M. Abe. Robust distributed multiplication without interaction. In M. Wiener, editor, *Advances in Cryptology — CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 130–147. Springer-Verlag, 1999. (Cited on page 14, 15.)

[2] M. Abe and S. Fehr. Adaptively secure Feldman VSS and applications to universally-composable threshold cryptography. IACR ePrint Archive 2004/119, June 10 2004. Preliminary version was presented in CRYPTO 2004. (Cited on page 14, 15.)

[3] M. Abe, R. Gennaro, K. Kurosawa, and V. Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In R. Cramer, editor, *Advances in Cryptology — EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 128–146. Springer-Verlag, 2005. Also available at IACR e-print 2005/027 and 2004/194. (Cited on page 3.)

[4] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communication Security*, pages 62–73. Association for Computing Machinery, 1993. (Cited on page 1, 12, 16.)

[5] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cyptographic fault-tolerant distributed computation. In *Proceedings of the* 20*th annual ACM Symposium on the Theory of Computing*, pages 1–10, 1988. (Cited on page 14.)

[6] K. Bentahar, P. Farshim, M. Malone-Lee, and N. Smart. Generic constructions of identity-based and certificateless KEMs. IACR e-print Archive 058/2005, 2005. (Cited on page 13.)

[7] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In H. Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 1–12. Springer-Verlag, 1998. (Cited on page 1.)

[8] D. Boneh. Simplified OAEP for the RSA and Rabin functions. In J. Killian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 275–291. Springer-Verlag, 2001. (Cited on page 9.)

[9] D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption. In *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 2004. (Cited on page 13.)

[10] D. Boneh, X. Boyen, and S. Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In T. Rabin and S. Halevi, editors, *Topics in Cryptology – CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 226–243, 2006. (Cited on page 14.)

[11] D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. Technical Report 2004/261, IACR ePrint archive, 2004. (Cited on page 13.)

[12] X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In *ACM Conference on Computer and Communications Security*, pages 320–329. ACM, 2005. Also available at IACR e-print 2005/288. (Cited on page 14.)

[13] R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 90–106. Springer-Verlag, 1999. (Cited on page 14, 15.)

[14] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer-Verlag, 2004. (Cited on page 13.)

[15] R. Canetti, H. Krawczyk, and J. Nielsen. Relaxing chosen-ciphertext security. In D. Boneh, editor, *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582. Springer-Verlag, 2003. Also available at IACR ePrint archive 2003/174. (Cited on page 16, 17.)

[16] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer-Verlag, 1998. (Cited on page 1, 12.)

[17] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In L. Knudsen, editor, *Advances in Cryptology—EUROCRYPTO 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer-Verlag, 2002. (Cited on page 1.)

[18] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003. (Cited on page 1, 5, 12, 22, 26.)

[19] A. Dent. A designer's guide to KEMs. In K. G. Paterson, editor, *9th IMA International Conference on Cryptography and Coding*, volume 2898 of *Lecture Notes in Computer Science*, pages 133–151. Springer-Verlag, 2003. (Cited on page 9.)

[20] Y. G. Desmedt and Y. Frankel. Threshold cryptosystems. In G. Brassard, editor, *Advances in Cryptology — CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 307–315. Springer-Verlag, 1990. (Cited on page 14.)

[21] Y. Dodis, R. Gennaro, J. Haastad, H. Krawczyk, and T. Rabin. Randomness extraction and key derivation using the CBC, Cascade and HMAC modes. In Matt Franklin, editor, *Advances in Cryptology — CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 494–510. Springer-Verlag, 2004. (Cited on page .)

[22] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. (Cited on page 1.)

[23] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In M. Wiener, editor, *Advances in Cryptology — CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer-Verlag, 1999. (Cited on page 15, 27.)

[24] R. Gennaro and V. Shoup. A note on an encryption scheme of Kurosawa and Desmedt. Technical Report 2004/194, IACR ePrint archive, 2004. (Cited on page 26.)

[25] C. Gentry. How to compress Rabin ciphertexts and signatures (and more). In Matt Franklin, editor, *Advances in Cryptology — CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 179–200. Springer-Verlag, 2004. (Cited on page 9.)

[26] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the 19th annual ACM Symposium on the Theory of Computing*, pages 218–229, New York City, 1987. (Cited on page 14.)

[27] J. Herranz, D. Hofheinz, and E. Kiltz. The Kurosawa-Desmedt key encapsulation is not chosen-ciphertext secure. IACR e-print Archive 2006/207, 2005. (Cited on page 1, 15.)

[28] S. Jarecki and A. Lysyanskaya. Adaptively secure threshold cryptography: Introducing cncurrency, removing erasures (extended abstract). In *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 221–242. Springer-Verlag, 2000. (Cited on page 14.)

[29] E. Kiltz. Chosen-ciphertext security from tag-based encryption. In S. Halevi and T. Rabin, editors, *Theory of Cryptography – TCC'06*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600. Springer-Verlag, 2006. (Cited on page 3.)

[30] K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In Matt Franklin, editor, *Advances in Cryptology — CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer-Verlag, 2004. (Cited on page 1, 2, 15, 26.)

[31] P. MacKenzie, M. K. Reiter, and K. Yang. Alternatives to non-malleability: Definitions, constructions, and applications. In M. Naor, editor, *Theory of Cryptography – TCC'04*, volume 2951 of *Lecture Notes in Computer Science*, pages 171–190. Springer-Verlag, 2004. (Cited on page 3, 16.)

[32] W. Nagao, Y. Manabe, and T. Okamoto. A universally composable secure channel based on the KEM-DEM framework. In *Theory of Cryptography – TCC'05*, volume 3378 of *Lecture Notes in Computer Science*, pages 426–444. Springer-Verlag, 2005. (Cited on page 17.)

[33] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the 22nd annual ACM Symposium on the Theory of Computing*, pages 427–437, 1990. (Cited on page 1.)

[34] T. Okamoto and D. Pointcheval. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In *RSA '2001*, Lecture Notes in Computer Science. Springer-Verlag, 2001. (Cited on page 1, 16.)

[35] C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer-Verlag, 1992. (Cited on page 1.)

[36] V. Shoup. Using hash functions as a hedge against chosen ciphertext attack. In *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 275–288. Springer-Verlag, 2000. (Cited on page 1, 5.)

[37] V. Shoup. OAEP reconsidered. In *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 239–259. Springer-Verlag, 2001. (Cited on page 12.)

[38] V. Shoup. ISO 18033-2: An emerging standard for public-key encryption (committee draft). Available at `http://shoup.net/iso/`, June 3 2004. (Cited on page 1, 3, 16, 17.)

[39] V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology*, 15(2):75–96, 2002. (Cited on page 3, 13, 14.)

Appendices

A Proof of Theorem 4.1

Let A_T be an adversary against the Tag-KEM presented in Section 4.1. By using A_T, we construct A_E that plays $\mathsf{GAME.PKE}$ to attack the underlying PKE as follows.

1. Given pk, A_E chooses $(dk_0, dk_1) \leftarrow \mathcal{K}_K \times \mathcal{K}_K$, $\delta \leftarrow \{0,1\}$. Then send pk and dk_δ to A_T.
2. Given τ from A_T, A_E sends $(dk_0||\mathrm{TH}(\tau), dk_1||\mathrm{TH}(\tau))$ to the encryption oracle of $\mathsf{GAME.PKE}$. It then receives ψ from the encryption oracle and forward it to A_T.
3. For every decryption query (ψ_i, τ_i) from A_T, A_E does the following. If $\psi_i = \psi$, return $\perp$ to A_T. Otherwise, send ψ_i to the decryption oracle of PKE. Given $dk_i||\tau'_i$ from the decryption oracle, if ${\tau'}_i \neq \tau_i$, return $\perp$ to A_T. Otherwise, return dk_i.
4. When A_T outputs $\tilde{\delta} = \delta$, A_E outputs 1. Otherwise, output 0.

Let Col denote an event that $\mathrm{TH}(\tau) = \mathrm{TH}(\tau')$ happens. Since the target ciphertext ψ is uniquely decrypted to $dk_\delta||\tau$, any (ψ, τ') other than (ψ, τ) cannot be a valid ciphertext of Tag-KEM unless Col takes place. Hence $\perp$ is a correct answer to any decryption query with $\psi_i = \psi$. All other decryption queries from A_T are correctly answered since ψ_i is correctly decrypted by the decryption oracle of $\mathsf{GAME.PKE}$. It is also easy to see that the encryption oracle is simulated as given.

Now if $\tilde{\delta} = \delta$, it means that dk_δ is embedded in ψ_i. On the other hand, if $\tilde{\delta} \neq \delta$, dk_δ is independent of ψ, i.e., $dk_{1-\delta}$ is embedded in ψ. Thus, A_E wins in $\mathsf{GAME.PKE}$ with the same probability as A_T wins in $\mathsf{GAME.TKEM}$ in the case of $\neg\mathsf{Col}$. Thus we have;

$$\begin{aligned} \Pr[A_T \text{ wins}|\neg\mathsf{Col}] &= \Pr[A_E \text{ wins}] \\ \Pr[A_T \text{ wins}] &\leq \Pr[A_E \text{ wins}] + Pr[\mathsf{Col}] \\ \epsilon_{\text{tkem}} &\leq \epsilon_{\text{pke}} + \epsilon_{\text{tch}}, \end{aligned}$$

where ϵ_{tch} is the probability of breaking TH as defined in Section 2.7.

B Proof of Theorem 4.2

The following is the CCA attack against the Tag-KEM in Section 4.3. Let $\mathcal{O}$ denote decryption oracle, $\mathsf{TKEM.Dec}_{sk}(\cdot,\cdot)$. Let $\mathsf{GAME.0}$ denote this CCA game.

[GAME.0]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $\delta \leftarrow \{0,1\}$. If $\delta = 1$, set dk by $(dk, *) \leftarrow \mathsf{KDF}_2(K_1)$. Otherwise, $dk \leftarrow \mathcal{K}_D$.

Step 2. $(\tau, \rho) \leftarrow {A_T}^{\mathcal{O}}(pk, dk)$

Step 3. $(*, mk) \leftarrow \mathsf{KDF}_2(K_1)$, $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$.

Step 4. $\tilde{\delta} \leftarrow {A_T}^{\mathcal{O}}(\rho, (\phi, \sigma))$

In Step 4, A_T is restricted not to send $((\phi, \sigma), \tau)$ to the decryption oracle. It is important to see that the MAC is always created with the correct mac-key embedded in ϕ regardless of δ.

The outline of our proof is the same as that of [18]; Defining a series of games, $\mathsf{GAME.1}$, $\mathsf{GAME.2}$, $\mathsf{GAME.3}$ by modifying $\mathsf{GAME.0}$ and examining fluctuation of probability $\Pr[X_0]$, $\cdots$, $\Pr[X_3]$ where X_i denotes the event that $\tilde{\delta} = \delta$ in $\mathsf{GAME}.i$. Our final goal is to upper limit $\Pr[X_0]$.

The following is the description of each game. Every claim is proven formally after all outline is shown.

$\mathsf{GAME.1}$: We modify the encryption oracle in such a way that when $\delta = 0$, dk is chosen by $K_0 \leftarrow \mathcal{K}_K$, $(dk, *) \leftarrow \mathsf{KDF}_2(K_0)$. It is straightforward to show that

$$|\Pr[X_0] - \Pr[X_1]| = \frac{1}{2}|\Pr[X_0 \,|\, \delta = 0] - \Pr[X_1 \,|\, \delta = 0]| \leq \epsilon_{\text{kdf}}. \tag{1}$$

$\mathsf{GAME.2}$: We modify $\mathsf{GAME.1}$ in such a way that the decryption oracle returns $\perp$ to all queries containing $\phi_j = \phi$. We claim that

$$|\Pr[X_1] - \Pr[X_2]| \leq 2\epsilon_{\text{lkem}} + q_D\epsilon_{\text{mac}} + 2\epsilon_{\text{kdf}}. \tag{2}$$

$\mathsf{GAME.3}$: We modify $\mathsf{GAME.2}$ in such a way that the encryption oracle computes the MAC σ by using a key derived from K_δ instead of the legitimate key K_1. Namely, we replace $\mathsf{KDF}_2(K_1)$ in Step 3 with $\mathsf{KDF}_2(K_\delta)$.

We claim that

$$|\Pr[X_2] - \Pr[X_3]| \leq \epsilon_{\mathrm{lkem}} + 2\epsilon_{\mathrm{kdf}}, \tag{3}$$

and

$$|\Pr[X_3] - \frac{1}{2}| \leq \epsilon_{\mathrm{lkem}}. \tag{4}$$

Conclusion: From (1), (2), (3), and (4), we have

$$|\Pr[X_0] - \frac{1}{2}| \leq 4\epsilon_{\mathrm{lkem}} + q_D\, \epsilon_{\mathrm{mac}} + 5\epsilon_{\mathrm{kdf}}$$

as stated in the theorem.

Proof of (2): Let F_2 be an event that the adversary creates at least one query that is rejected in $\mathsf{GAME.2}$ but accepted in $\mathsf{GAME.1}$. Unless event F_2 happens, the view of the adversary is identical in both games. Hence we have

$$|\Pr[X_1] - \Pr[X_2]| \leq \Pr[F_2]. \tag{5}$$

We first consider F_2 in the case of $\delta = 1$. Consider the simulation conducted by machine A_L that launches LCCA attack to KEM by using A_T as follows.

Step 1. Given (pk, ϕ, K_b), A_L computes $(dk, mk) \leftarrow \mathsf{KDF}_2(K_b)$ and sends (pk, dk) to A_T.

Step 2. Given τ from A_T, compute $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$ and return $\psi = (\phi, \sigma)$.

Step 3. For every decryption query $((\phi_j, \sigma_j), \tau_j)$, simulate the decryption oracle as follows.

- If $\phi_j \neq \phi$, send ϕ_j and $\eta_j = (\sigma_j, \tau_j)$ to the decryption oracle of $\mathsf{GAME.LKEM}$ and receive K_j. (If $\perp$ is returned, forward it to A_T.) Then use K_j to simulate the decryption oracle of $\mathsf{GAME.2}$ as prescribed.
- If $\phi_j = \phi$, compute $\mathsf{MAC.Ver}_{mk}(\sigma_j, \tau_j)$. If it is 1, output $\tilde{b} = 1$ and halt. Otherwise, return $\perp$ to A_T and continue simulation.

Step 4. When A_T stops, output $\tilde{b} = 0$.

Remember that K_b is either random ($b = 0$) or correct ($b = 1$) with regard to ϕ. When $b = 1$, the decryption oracle checks every MAC with the correct key. Hence A_L can detect event F_2 and $\tilde{b} = 1$ happens. On the other hand, when $b = 0$, every MAC σ_j is verified with a random key associated only to the MAC σ attached to the challenge ciphertext. Therefore, unless MAC is forged, $\tilde{b} = 1$ does not happen. If the probability that event $\tilde{b} = 1$ occurs is meaningfully different for $b = 0$ and $b = 1$, it contradicts to the LCCA security. Now, we consider these two cases, $b = 1$ and $b = 0$, in detail.

1. When $b = 1$ in $\mathsf{GAME.LKEM}$, both dk and σ are made from correct key embedded in ψ. This simulates the encryption oracle of $\mathsf{GAME.2}$ at $\delta = 1$. The decryption oracle is simulated as given until event F_2 happens. A_T can correctly capture the event when it happens since mk is correct. Therefore, $\Pr[F_2 \,|\, \delta = 1] = \Pr[\tilde{b} = 1 \,|\, b = 1]$.

2. When $b = 0$, we claim $\Pr[\tilde{b} = 1 \,|\, b = 0] \leq q_D\epsilon_{\rm mac} + 2\epsilon_{\rm kdf}$. Hereafter, we only consider the case $b = 0$ where dk and mk are independent of ψ. Let F_2' be the event that $\tilde{b} = 1$ happens when $b = 0$. We modify the simulation by A_L in such a way that it chooses dk and mk just randomly, i.e., $(dk, mk) \leftarrow \mathcal{K}_D \times \mathcal{K}_M$ instead of using KDF_2. Let F_2'' be the event of $\tilde{b} = 1$ in this simulation. We claim that $|\Pr[F_2'] - \Pr[F_2'']| \leq 2\epsilon_{\rm kdf}$. Observe that the input to KDF_2 in the original simulation is random and independent from all other views because $b = 0$. Hence the claim is proven by showing a straightforward reduction from A_T to A_{KDF}, which we omit the details. We also claim that $\Pr[F_2''] \leq q_D\epsilon_{\rm mac}$. In the modified simulation, mk is randomly and independently chosen and bound only to (σ, τ). Therefore, if A_T causes $\tilde{b} = 1$, (σ_j, τ_j) is a correct forgery with regard to (σ, τ). Unfortunately, (σ_j, τ_j) is not verifiable without the key and index j has to be guessed from $\{1, \ldots, q_D\}$. Therefore, one can be successful in forging a MAC with probability $1/q_D$ whenever A_T causes $\tilde{b} = 1$. This proves the claim.

Since $|\Pr[\tilde{b} = 1 \,|\, b = 1] - \Pr[\tilde{b} = 1 \,|\, b = 0]| \leq 2\epsilon_{\rm lkem}$, we have

$$\Pr[F_2 \,|\, \delta = 1] \leq 2\epsilon_{\rm lkem} + q_D\epsilon_{\rm mac} + 2\epsilon_{\rm kdf}. \tag{6}$$

We next consider F_2 in the case of $\delta = 0$ where dk is always made from random K_0. Consider the simulation conducted by machine A_L that launches LCCA attack to KEM by using A_T almost in the same way as above except that the encryption oracle computes dk by $(dk, *) \leftarrow \mathsf{KDF}_2(K_0)$ where K_0 is chosen randomly from $\mathcal{K}_K$. This means that dk distributes independently from all other variables. Accordingly, when $b = 1$, the view of A_T in the simulation is identical to that in $\mathsf{GAME}.2$ at $\delta = 0$. Hence we have $\Pr[F_2 \,|\, \delta = 0] = \Pr[\tilde{b} = 1 \,|\, b = 1]$. Furthermore, one can show that $\Pr[\tilde{b} = 1 \,|\, b = 0] \leq q_D\epsilon_{\rm mac} + 2\epsilon_{\rm kdf}$ in exactly the same way as above. Accordingly, we have

$$\Pr[F_2 \,|\, \delta = 0] \leq 2\epsilon_{\rm lkem} + q_D\epsilon_{\rm mac} + 2\epsilon_{\rm kdf}. \tag{7}$$

From (6) and (7), we have the following as claimed.

$$\begin{aligned} \Pr[F_2] &= \frac{1}{2}\left(\Pr[F_2 \,|\, \delta = 0] + \Pr[F_2 \,|\, \delta = 1]\right) \\ &\leq 2\epsilon_{\rm lkem} + q_D\epsilon_{\rm mac} + 2\epsilon_{\rm kdf} \end{aligned}$$

Proof of (3): First of all, observe that, when $\delta = 1$ in $\mathsf{GAME}.2$ and $\mathsf{GAME}.3$, the views of A_T are identical since there is no difference between the games. Therefore,

$$\begin{aligned} |\Pr[X_2] - \Pr[X_3]| &= \frac{1}{2}|\Pr[X_2 \,|\, \delta = 1] - \Pr[X_3 \,|\, \delta = 1] + \Pr[X_2 \,|\, \delta = 0] - \Pr[X_3 \,|\, \delta = 0]| \\ &= \frac{1}{2}|\Pr[X_2 \,|\, \delta = 0] - \Pr[X_3 \,|\, \delta = 0]| \end{aligned}$$

Accordingly, we only need to consider the case where $\delta = 0$, i.e., dk is always made from random K_0.

We construct machine A_L, that launches LCCA attack to KEM by using A_T. A_L works as follows.

Step 1. Given (pk, ϕ, K_b), compute $K_0' \leftarrow \mathcal{K}_K$, $(dk, *) \leftarrow \mathsf{KDF}_2(K_0')$ and send (pk, dk) to A_T.

Step 2. Given τ from A_T, generate σ by using K_b as $(*, mk) \leftarrow \mathsf{KDF}_2(K_b)$, $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$. Then return (ϕ, σ).

Step 3. For every decryption query $((\phi_j, \sigma_j), \tau_j)$, simulate the decryption oracle as follows. If $\phi_j = \phi$, return $\perp$. Otherwise, send $(\phi_j, (\sigma_j, \tau_j))$ to the decryption oracle of $\mathsf{GAME.LKEM}$ and receive K_j. Then simulate the decryption oracle as prescribed by using K_j.

Step 4. When A_T outputs $\tilde{\delta}$, output it as $\tilde{b}$.

It is important to see that dk is independent of ϕ regardless of b. Also remember that K_b is either random ($b = 0$) or correct ($b = 1$) with respect to ϕ. Now, by inspection, one can see that the following holds.

1. When $b = 1$ in $\mathsf{GAME.LKEM}$, σ is made from correct key embedded in ϕ. The decryption oracle is simulated as given. Hence the view of A_T in the simulation by A_K is identical to that in $\mathsf{GAME.2}$ at $\delta = 0$, i.e., $\Pr[X_2 \,|\, \delta = 0] = \Pr[\tilde{b} = 0 \,|\, b = 1]$.

2. Similarly, when $b = 0$, σ is made from random key independent of that embedded in ϕ as well as the case in $\mathsf{GAME.3}$ at $\delta = 0$. The decryption oracle is simulated just as given. However, the view of A_T in the simulation is slightly different from that in $\mathsf{GAME.3}$ at $\delta = 0$ because dk and mk are made from independent sources, K_0' and K_0, of KDF_2 while they are made from a single input to KDF_2 in $\mathsf{GAME.3}$ at $\delta = 0$. However, from Corollary 2.2, we claim that $|\Pr[X_3 \,|\, \delta = 0] - \Pr[\tilde{b} = 0 \,|\, b = 0]| \leq 4\,\epsilon_{\mathrm{kdf}}$. (Since K_0' and K_0 are randomly chosen and used only as inputs to KDF_2, the claim is proven by a straightforward reduction.)

In summary, we have:

$$
\begin{aligned}
|\Pr[X_2] - \Pr[X_3]| &= \frac{1}{2}|\Pr[X_2 \,|\, \delta = 0] - \Pr[X_3 \,|\, \delta = 0]| \\
&\leq \frac{1}{2}|\Pr[\tilde{b} = 0 \,|\, b = 1] - \Pr[\tilde{b} = 0 \,|\, b = 0] - 4\,\epsilon_{\mathrm{kdf}}| \\
&\leq \epsilon_{\mathrm{lkem}} + 2\epsilon_{\mathrm{kdf}}
\end{aligned}
$$

Proof of (4): Proof is done by constructing A_L playing $\mathsf{GAME.LKEM}$ by using A_T in $\mathsf{GAME.3}$. Basically, what A_L does is to simulate the encryption oracle by creating σ, and to simulate the decryption oracle by sending ϕ_j and (σ_j, τ_j) to the decryption oracle of $\mathsf{GAME.LKEM}$. These simulations are easy because necessary keys are provided by corresponding oracles of $\mathsf{GAME.LKEM}$. A_L finally outputs $\tilde{\delta}$ as A_T does. Simulation is perfect since, in $\mathsf{GAME.3}$, A_L will not send ϕ_j that is identical to ϕ to the decryption oracle but simply reject it. Hence the advantage of A_L in $\mathsf{GAME.LKEM}$ is the same as that of A_T in $\mathsf{GAME.3}$.

C Kurosawa-Desmedt KEM

We define the Kurosawa-Desmedt KEM as follows. Key generation function $\mathsf{KEM.Gen}$ is as illustrated in Section 5.2. It outputs $pk = (g_1, g_2, c, d)$ and $sk = (x_1, x_2, y_1, y_2)$. On input pk,

$\mathsf{KEM.Enc}$ outputs random source key K and ciphertext $\phi = (u_1, u_2)$ such that

$$r \leftarrow \mathbb{Z}_q, u_1 = g_1^r, u_2 = g_2^r, \alpha = H(u_1||u_2), K = c^r d^{\alpha r}.$$

Given sk and ϕ, decryption function $\mathsf{KEM.Dec}$ outputs K such that

$$\alpha = H(u_1||u_2),\ K = {u_1}^{x_1+\alpha y_1} {u_2}^{x_2+\alpha y_2}.$$

We say $\phi = (u_1, u_2)$ is valid if there exists $r \in \mathbb{Z}_q$ such that $u_1 = g_1^r, u_2 = g^r$. Otherwise, it is invalid.

Lemma C.1 The above Kurosawa-Desmedt KEM is LCCA-secure with respect to $\mathcal{P}^{\mathrm{mac}}$ if H is TCR and the DDH assumption holds.

We start by describing $\mathsf{GAME.LKEM}$ against the Kurosawa-Desmedt KEM.

1. Run $\mathsf{KEM.Gen}$ to generate $pk = (g_1, g_2, c, d)$ and $sk = (x_1, x_2, y_1, y_2)$. Choose $r^* \leftarrow Z_q$ and compute
$$u_1^* = g_1^{r^*},\ u_2^* = g_2^{r^*}, \alpha^* = H(u_1^*, u_2^*), K^* = c^{r^*} d^{r^* \alpha^*}.$$
Let $\phi^* = (u_1^*, u_2^*)$ and $K_1 = K^*$. Choose $K_0 \in G_q$ randomly. Then choose $\delta \leftarrow \{0, 1\}$ and give (pk, ϕ^*, K_δ) to A_L.
2. A_L makes query $((u_{1j}, u_{2j}), \eta_j)$ to extended oracle $\mathcal{VD}$ arbitrarily. Let $K_j = \mathsf{KEM.Dec}_{sk}(u_{1j}, u_{2j})$. Oracle $\mathcal{VD}$ returns $\perp$ if (K_j, η_j) does not satisfy $\mathcal{P}^{\mathrm{mac}}$. Otherwise, it returns K_j (which might be $\perp$ anyway). Eventually A_L outputs $\tilde{\delta} \in \{0, 1\}$.

Since the proof is quite similar to that of [30, 24], we only show a sketch of it. From a viewpoint of A_L, there are four unknown variables x_1, x_2, y_1, y_2, and two (linear) equations on them given by (the discrete log of) c and d. Hence there is a freedom of $4 - 2 = 2$ dimensions. We consider this probability space on (x_1, x_2, y_1, y_2) of the 2 dimensions. Let $\mathsf{GAME.0}$ be the original game as shown above. We will define a sequence of games $\mathsf{GAME.1}, \cdots$. Let X_i be the event that $\tilde{\delta} = \delta$ in $\mathsf{GAME}.i$.

$\mathsf{GAME.1}$ is the same as $\mathsf{GAME.0}$, except that K^* is computed as $K^* = (u_1^*)^{x_1+y_1\alpha^*}(u_2^*)^{x_2+y_2\alpha^*}$. It is clear that $\Pr[X_1] = \Pr[X_0]$ because the value of v^* does not change.

$\mathsf{GAME.2}$ is the same as $\mathsf{GAME.1}$, except that query (u_{1j}, u_{2j}, η_j) is rejected if $\alpha^* = H(u_1^*, u_2^*) = H(u_{1j}, u_{2j})$. Since H is assumed target collision-free, $|\Pr[X_2] - \Pr[X_1]|$ is negligible.

$\mathsf{GAME.3}$ is the same as $\mathsf{GAME.2}$, except that the challenger chooses $u_1^*, u_2^* \in G_q$ at random. By DDH assumption, $\Pr[X_3] - \Pr[X_2]$ is negligible.

$\mathsf{GAME.4}$ is the same as $\mathsf{GAME.3}$, except that the extended decryption oracle returns $\perp$ for a query $((u_{1j}, u_{2j}), \eta_j)$ if (u_{1j}, u_{2j}) is invalid.

Suppose that A_L queries with invalid (u_{1j}, u_{2j}) at step 2 of $\mathsf{GAME.3}$. Let $\alpha' = H(u_{1j}, u_{2j})$ and $K_j = {u_{1j}}^{x_1+y_1\alpha'}{u_{2j}}^{x_2+y_2\alpha'}$. If $(u_{1j}, u_{2j}) = (u_1^*, u_2^*)$, then the query is immediately rejected by the extended decryption oracle. Suppose that $(u_{1j}, u_{2j}) \neq (u_1^*, u_2^*)$. Note that (u_1^*, u_2^*) is invalid with overwhelming probability. Then as shown in [18], K^* and K_j are pair-wise independently distributed over G_q. Therefore, as well as in the above case, the query is rejected with overwhelming probability. Consequently, $\Pr[X_3] - \Pr[X_2]$ is negligible.

$\mathsf{GAME.4}$ is the same as $\mathsf{GAME.3}$, except that K^* is chosen at random from G_q. In $\mathsf{GAME.3}$, (u_1^*, u_2^*) is invalid with overwhelming probability and K^* is uniformly distributed over G_q. Hence $\Pr[X_4] - \Pr[X_3]$ is negligible. Also in game $\mathsf{GAME.4}$, it is clear that $\Pr[X_4] = 1/2$ because the view of A_L is independent of δ. This completes the proof on LCCA security.

D Details of Refined Fujisaki-Okamoto Conversion

The scheme. Let $\Pi_E = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a public-key encryption. By $\mathcal{R}$, we denote a space of random coins used in $\mathsf{PKE.Enc}$. Let $H : \{0,1\}^\lambda \times \{0,1\}^* \to \mathcal{R}$ and $G : \{0,1\}^\lambda \to \mathcal{K}_D$ be random oracles. $\mathsf{TKEM.Gen}$ is the same as $\mathsf{PKE.Gen}$; Given 1^λ, it outputs key pair (pk, sk). $\mathsf{TKEM.Key}$ is that, given pk, it outputs $dk \leftarrow G(K)$ where $K \leftarrow \{0,1\}^\lambda$ and state information (pk, K). $\mathsf{TKEM.Enc}$ and $\mathsf{TKEM.Dec}$ are as follows.

Function: $\mathsf{TKEM.Enc}((pk, K), \tau)$

$r \leftarrow H(K||\tau)$
$\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K; r)$
Output ψ.

Function: $\mathsf{TKEM.Dec}_{sk}(\psi, \tau)$

$K \leftarrow \mathsf{PKE.Dec}_{sk}(\psi)$
$r \leftarrow H(K||\tau)$
If $\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K; r)$, output $G(K)$.
Output $\perp$, otherwise.

Suppose that the PKE is one-way against chosen plaintext attack and γ-uniform. [3] Then the following holds.

Theorem D.1 The above refined Fujisaki-Okamoto Tag-KEM is CCA secure. Especially, $\epsilon_{\rm tkem} \leq q_D\,\gamma + q_G\,\epsilon_{\rm ow}$.

Proof: The following is the CCA attack against the Tag-KEM in Section 5.3. By $\mathcal{O}$, we denote the decryption oracle $\mathsf{TKEM.Dec}_{sk}(\cdot,\cdot)$. Let $\mathsf{GAME.0}$ denote this CCA game.

Step 1. $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$, $K_1 \leftarrow \{0,1\}^\lambda$, $dk_1 = G(K_1)$, $dk_0 \leftarrow \mathcal{K}_D$, $\delta \leftarrow \{0,1\}$.

Step 2. $(\tau, \rho) \leftarrow {A_T}^{\mathcal{O},H,G}(pk, dk_\delta)$

Step 3. $\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K_1; H(K_1||\tau))$

Step 4. $\tilde{\delta} \leftarrow {A_T}^{\mathcal{O},H,G}(\rho, \psi)$

We define games, $\mathsf{GAME.1}$ and $\mathsf{GAME.2}$, by modifying $\mathsf{GAME.0}$ and examining fluctuation of probability $\Pr[X_0], \cdots, \Pr[X_2]$ where X_i denotes the event that $\tilde{\delta} = \delta$ in $\mathsf{GAME}.i$.

We treat each random oracle as a table that appends an entry every time it is drawn with a fresh query. Given fresh K_j, oracle G outputs random dk_j and append (dk_j, K_j) to its table. Given fresh $K_j||\tau_j$, oracle H outputs random r_j. Since $K_j||\tau_j$ and r_j uniquely determines corresponding ciphertext, say ψ_j, we assume that H stores $(K_j||\tau_j, r_j, \psi_j)$ to the table. When exact reduction cost is concerned, computation time for the ciphertexts, which is linear in the number of H oracle queries, should be included in the running time of the simulator.

$\mathsf{GAME.1}$: For every decryption query (ψ_j, τ_j), if table of H does not have an entry that contains both ψ_j and τ_j, return $\perp$.

[3] Roughly, a PKE is one-way against chosen plaintext attack if it is infeasible to compute K from $\mathsf{PKE.Enc}_{pk}(K)$ except for negligible probability, say $\epsilon_{\rm ow}$. Also, a PKE is γ-uniform if, for any K and ψ, randomly chosen r causes $\psi = \mathsf{PKE.Enc}_{pk}(K; r)$ with probability less than γ. See [23] for details.

Since (ψ_j, τ_j) is not in the table of H, r_j is chosen randomly in GAME.0. With random r_j, it passes the verification test in the decryption with probability at most γ. If at most q_D queries are made, the probability that there is a query that is accepted in GAME.0 but not in GAME.1 is at most $q_D\gamma$. The view of the adversary is unchanged unless such a query is made. Accordingly,

$$|\Pr[X_0] - \Pr[X_1]| \leq q_D\,\gamma.$$

GAME.2: If oracle G receives K_1 from A_T, abort the game.

Let AskG denote the event that G receives K_1. By using A_T that causes AskG, we construct an adversary, $A_{\rm ow}$, that derives K from given ψ and pk. $A_{\rm ow}$ first flips coin $j^\star \leftarrow \{1, \ldots, q_G\}$ and simulates GAME.2 as follows.

- In Step 1, choose dk uniformly from $\mathcal{K}_D$. Then give (pk, dk) to A_T.
- H and G are simulated just as given. Given $j^\star$-th query $K_{j^\star}$ to G, output it and halt.
- For every decryption query (ψ_j, τ_j), if there is an entry, $(K_i||\tau_i, r_i, \psi_i)$, such that $\psi_j = \psi_i$ and $\tau_j = \tau_i$, return $G(K_i)$. Otherwise, return $\perp$.
- In Step 4, give ψ to A_T.

If $j^\star$-th query to G is made before the encryption oracle is invoked, the simulation is perfect. Even if it happens after the encryption oracle is invoked, randomly chosen dk perfectly simulates the output of the encryption oracle regardless of the choice of δ. Accordingly, $A_{\rm ow}$ perfectly simulates GAME.2 up to the moment $j^\star$-th query to G is made. And once event AskG happens at $j^\star$-th query, the output of $A_{\rm ow}$ is PKE.Dec(sk, ψ).

Running time of $A_{\rm ow}$ is almost the same as that of A_T plus computing time of encryption function q_H times. Now we have

$$|\Pr[X_1] - \Pr[X_2]| \leq q_G\,\epsilon_{\rm ow}.$$

Since A_T never asks K_1 to G in GAME.2, δ is independent from the view of A_T due to the true randomness of G. Hence $\Pr[X_2] = \frac{1}{2}$.

In conclusion, we have $\epsilon_{\rm tkem} = |\Pr[X_0] - \frac{1}{2}| \leq q_D\,\gamma + q_G\,\epsilon_{\rm ow}$ as stated.

∎

E Tag-KEM from RCCA-secure KEM

RCCA security for KEM is defined in the same way as for PKE. That is, we modify GAME.KEM in Section 2.4 in such a way that decryption oracle $\mathcal{O}$ returns 'test' when the result of decryption is in $\{K_1, K_0\}$. [4] Call this modified game GAME.RKEM. The scheme is RCCA-secure if any ppt adversary wins GAME.RKEM only with negligible advantage, say $\epsilon_{\rm rkem}$, as defined in the same

[4] Notice that when $\delta = 1$, K_0 is independent from the transcript between the adversary and the encryption oracle. Nevertheless, the decryption oracle returns 'test' if the decryption coincidentally yields K_0. Though it is totally a definitional matter and can be fixed if necessary, we prefer current definition that gives obvious reduction to RCCA PKE.

way for CCA security. It is clear that the standard use of RCCA secure PKE is sufficient to construct RCCA secure KEM.

One can construct a Tag-KEM based on an RCCA-secure KEM in the similar way as shown in Section 4.2. $\mathsf{TKEM.Gen}$ is the same as $\mathsf{KEM.Gen}$. $\mathsf{TKEM.Key}$ is that, given pk, it computes $(K, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$ and $(dk, mk) \leftarrow \mathsf{KDF}_2(K)$. Then it outputs dk and state information $\omega = (mk, \phi)$. $\mathsf{TKEM.Enc}$ and $\mathsf{TKEM.Dec}$ are as follows.

Function: $\mathsf{TKEM.Enc}(\omega, \tau)$

$(mk, \phi) \leftarrow \omega$
$\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau || \phi)$
Output $\psi = (\phi, \sigma)$

Function: $\mathsf{TKEM.Dec}_{sk}(\psi, \tau)$

$(\phi, \sigma) \leftarrow \psi$
$K \leftarrow \mathsf{KEM.Dec}_{sk}(\phi)$
$(dk, mk) \leftarrow \mathsf{KDF}_2(K)$
If $K = \perp$ or $\mathsf{MAC.Ver}_{mk}(\sigma, \tau || \phi) \neq 1$, output $\perp$.
Otherwise, output dk.

A difference from the construction in Section 4.2 is that $\mathsf{MAC.Sign}$ and $\mathsf{MAC.Ver}$ take $\tau || \phi$ instead of τ.

Theorem E.1 If KEM is RCCA-secure, MAC is one-time secure, and KDF is secure, the above Tag-KEM is CCA-secure. Especially, $\epsilon_{\rm tkem} \leq 2\epsilon_{\rm rkem} + (q_D + 3)\epsilon_{\rm kdf} + \frac{q_D}{2}\epsilon_{\rm mac}$

(Proof.) The attack game to the resulting Tag-KEM scheme is the same as $\mathsf{GAME.0}$ shown in Appendix B with obvious modification that replaces $\mathsf{MAC.Sign}_{mk}(\tau)$ in Step 3 with $\mathsf{MAC.Sign}_{mk}(\tau || \phi)$. For completeness, we show the game in the following.

[GAME.0]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $\delta \leftarrow \{0, 1\}$. If $\delta = 1$, set dk by $(dk, *) \leftarrow \mathsf{KDF}_2(K_1)$. Otherwise, $dk \leftarrow \mathcal{K}_D$.

Step 2. $(\tau, \rho) \leftarrow {A_T}^{\mathcal{O}}(pk, dk)$

Step 3. $(*, mk) \leftarrow \mathsf{KDF}_2(K_1)$, $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau || \phi)$.

Step 4. $\tilde{\delta} \leftarrow {A_T}^{\mathcal{O}}(\rho, (\phi, \sigma))$

Now we give a sketch of a proof of security which is similar to that in Appendix B. The idea of the proof is to continuously modify the game so that the relation among (dk, mk, ϕ) in the encryption oracle changes. Throughout the modifications, we only consider the case of $\delta = 0$. (Hence no modification in the case of $\delta = 1$.) By $[X], [Y]$, we denote that X and Y are independent. Similarly, $[X, Y]$ denotes that X and Y are related each other in some way.

- In $\mathsf{GAME.0}$, dk is chosen randomly from $\mathcal{K}_D$, and mk and ϕ are related via K_1. Hence $([dk], [mk, \phi])$.

- In $\mathsf{GAME.1}$, dk is generated by applying KDF to a random input. The relation is unchanged, i.e., $([dk], [mk, \phi])$.

- In $\mathsf{GAME.2}$, mk is generated independently. Hence $([dk], [mk], [\phi])$.

- In $\mathsf{GAME.3}$, dk and mk are coupled by generating them from a single application of KDF. Hence $([dk, mk], [\phi])$.

Now we consider the case of $\delta = 1$ in $\mathsf{GAME.3}$. Since dk, mk and ϕ are all relative to K_1 embedded in ϕ, we have $([dk, mk, \phi])$. Then, $\mathsf{GAME.3}$ is easily reduced to $\mathsf{GAME.RKEM}$. Remember that in $\mathsf{GAME.RKEM}$, challenge (K_b, ϕ) is either $([K_b], [\phi])$ at $b = 0$ or $([K_b, \phi])$ at $b = 1$. Therefor, by generating dk and mk from K_b, we can create $([dk, mk], [\phi])$ at $b = 0$ and $([dk, mk, \phi])$ at $b = 1$. Therefore, distinguishing δ in $\mathsf{GAME.3}$ implies to distinguishing b in $\mathsf{GAME.RKEM}$. In the following, we show the details where the decryption oracle is also changing in accordance with the modifications of the encryption oracle.

$\mathsf{GAME.1}$: The encryption oracle is modified in such a way that when $\delta = 0$, dk is generated by $(dk, *) \leftarrow \mathsf{KDF}_2(K_r)$ where K_r is chosen randomly from $\mathcal{K}_K$.

Since $\mathsf{GAME.1}$ is exactly the same as that in the proof of Theorem 4.2 as shown in Appendix B, we have

$$|\Pr[X_0] - \Pr[X_1]| = \frac{1}{2} |\Pr[X_0 \mid \delta = 0] - \Pr[X_1 \mid \delta = 0]| \leq \epsilon_{\text{kdf}}. \tag{8}$$

$\mathsf{GAME.2}$: We modify the encryption oracle in such a way that, when $\delta = 0$, MAC key mk in Step 3 is generated from independently chosen random K_0. That is, $(*, mk) \leftarrow \mathsf{KDF}_2(K_1)$ in Step 3 is replaced with $K_0 \leftarrow \mathcal{K}_K$, $(dk', mk) \leftarrow \mathsf{KDF}_2(K_0)$. Then mk is used to create σ. (Challenge DEM key dk returned to the adversary is unchanged. dk' generated here will be used in the decryption oracle as shown below.) We also modify the decryption oracle in such a way that, when the encryption oracle selects $\delta = 0$, every decryption query $(\phi_j, \sigma_j, \tau_j)$ made after the challenge step is handled as follows.

- $K_j \leftarrow \mathsf{KEM.Dec}_{sk}(\phi_j)$
- If $K_j \notin \{K_1, K_0\}$, then proceed as done in $\mathsf{GAME.1}$. Otherwise, verify the MAC by using mk and return dk' if the MAC is correct. Otherwise, return $\perp$.

Note that the modification affects only if $\delta = 0$. We claim that

$$|\Pr[X_1] - \Pr[X_2]| \leq \epsilon_{\text{rkem}}. \tag{9}$$

$\mathsf{GAME.3}$: We modify the encryption oracle in such a way that, when $\delta = 0$, it generates dk and mk together by computing $(dk, mk) \leftarrow \mathsf{KDF}_2(K_0)$ rather than compute them independently. Since dk' is no longer defined, the decryption oracle returns dk when $K_j \in \{K_1, K_0\}$ and the MAC is correct. (Or, the description of the decryption oracle is left unchanged by defining $dk' = dk$.)

We claim

$$|\Pr[X_2] - \Pr[X_3]| \leq \epsilon_{\text{kdf}} + \frac{q_D}{2}(\epsilon_{\text{mac}} + 2\epsilon_{\text{kdf}}) \tag{10}$$

and

$$|\Pr[X_3] - \frac{1}{2}| \leq \epsilon_{\text{rkem}} \tag{11}$$

Conclusion: From (8), (9), (10), and (11), we have

$$\begin{aligned} |\Pr[X_0] - \frac{1}{2}| &\leq \epsilon_{\text{kdf}} + \epsilon_{\text{rkem}} + 2\epsilon_{\text{kdf}} + \frac{q_D}{2}(\epsilon_{\text{mac}} + 2\epsilon_{\text{kdf}}) + \epsilon_{\text{rkem}} \\ &\leq 2\epsilon_{\text{rkem}} + (q_D + 3)\epsilon_{\text{kdf}} + \frac{q_D}{2}\epsilon_{\text{mac}} \end{aligned}$$

as stated in the theorem.

Proof of (9). In the case of $\delta = 1$, $\mathsf{GAME.1}$ and $\mathsf{GAME.2}$ are identical. Hence we only consider the case of $\delta = 0$.

We show a reduction from adversary A_T playing in $\mathsf{GAME.1}$ (and $\mathsf{GAME.2}$) at $\delta = 0$ to adversary A_K that attacks the underlying RCCA-secure KEM. Specification of A_K follows.

Step 1. Forward given pk to A_T.

Step 2. Simulate the encryption oracle as follows.

- Given request from A_T, compute $K_r \leftarrow \mathcal{K}_K$, $(dk, *) \leftarrow \mathsf{KDF}_2(K_r)$ and return dk. Then, make a challenge request to the encryption oracle of $\mathsf{GAME.RKEM}$ and receive (ϕ, K_b). Then compute $(dk', mk) \leftarrow \mathsf{KDF}_2(K_b)$.
- Given τ from A_T, compute $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau || \phi)$ and return (ϕ, σ).

Step 3. Given decryption query $(\phi_j, \sigma_j, \tau_j)$ from A_T, send ϕ_j to the decryption oracle of $\mathsf{GAME.RKEM}$. If K_j is returned, proceed as prescribed; Set $(dk_j, mk_j) \leftarrow \mathsf{KDF}_2(K_j)$ and return dk_j if $\mathsf{MAC.Ver}_{mk_j}(\sigma_j, \tau_j || \phi_j) = 1$. Otherwise return $\perp$. On the other hand, if 'test' is returned, verify the MAC by using mk as $\mathsf{MAC.Ver}_{mk}(\sigma_j, \tau_j || \phi_j) \stackrel{?}{=} 1$. If it holds, return dk'. Otherwise, return $\perp$.

Step 4. When A_T outputs $\tilde{\delta}$, output $\tilde{b} = \tilde{\delta}$.

First consider the case of $b = 1$ in the above simulation. Regarding the simulation of the encryption oracle, observe that dk is made from independent K_r and mk and ϕ are related via K_b in the correct manner as in $\mathsf{GAME.1}$. Hence the view of A_T with respect to the encryption oracle is identical to that in $\mathsf{GAME.1}$ at $\delta = 0$. The decryption oracle also simulates the view of $\mathsf{GAME.1}$ at $\delta = 0$ perfectly unless 'test' is returned from the decryption oracle of $\mathsf{GAME.RKEM}$. Observe that receiving 'test' means that K_b is embedded in both ϕ_j and ϕ (remember that we are in the case of $b = 1$). Hence mk and dk' derived from K_b are the same as those used in the real decryption oracle in $\mathsf{GAME.1}$. Eventually, the simulation is perfect even if 'test' is returned. Therefore, we have;

$$\Pr[X_1 \,|\, \delta = 0] = \Pr[\tilde{b} = 0 \,|\, b = 1]. \tag{12}$$

Next consider the case of $b = 0$ which is much complicated than the previous case. We consider the difference from the simulation and $\mathsf{GAME.2}$ at $\delta = 0$ for this case. First observe that the encryption oracle is perfectly simulated since dk, mk, and ϕ distributes independently both in the simulation and $\mathsf{GAME.2}$ at $\delta = 0$. The decryption oracle is also simulated perfectly since receiving 'test' means $K_j \in \{K_0, K_1\}$ and the MAC is verified as prescribed by using mk defined in the encryption oracle. Accordingly, we have;

$$\Pr[X_2 \,|\, \delta = 0] = \Pr[\tilde{b} = 0 \,|\, b = 0]. \tag{13}$$

From (12) and (13), we have

$$\begin{aligned} |\Pr[X_1 \,|\, \delta = 0] - \Pr[X_2 \,|\, \delta = 0]| &= |\Pr[\tilde{b} = 0 \,|\, b = 1] - \Pr[\tilde{b} = 0 \,|\, b = 0]| \\ &\leq 2\epsilon_{\text{rkem}}. \end{aligned} \tag{14}$$

Accordingly,

$$|\Pr[X_1] - \Pr[X_2]| = \frac{1}{2}|\Pr[X_1 \,|\, \delta = 0] - \Pr[X_2 \,|\, \delta = 0]| \le \epsilon_{\text{rkem}}$$

as claimed in (9).

Proof of (10). We show a reduction from A_T playing in $\mathsf{GAME.2}$ to A_{KDF} that attacks KDF in the sense of distinguishing distribution U_0 and U_1 described in Section 2.6. Given challenge (dk, mk) chosen randomly from distribution U_b where $b \leftarrow \{0, 1\}$, adversary A_{KDF} works as follows.

Step 1. Generate $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$ and give (pk, dk) to A_T.

Step 2. On receiving τ from A_T, compute $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$ and $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau||\phi)$. Then return (ϕ, σ) to A_T.

Step 3. Decryption oracle is simulated by using the real secret key sk; On receiving a query $((\phi_j, \sigma_j), \tau_j)$, decrypt ϕ_j into K_j. If $K_j = K_1$, then return dk if $\mathsf{MAC.Ver}_{mk}(\sigma_j, \tau_j||\phi_j) = 1$, or return $\perp$, otherwise. If $K_j \neq K_1$, compute $(dk_j, mk_j) \leftarrow \mathsf{KDF}_2(K_j)$ and return dk_j if $\mathsf{MAC.Ver}_{mk_j}(\sigma_j, \tau_j||\phi_j) = 1$, or return $\perp$, otherwise.

Step 4. When A_T outputs $\tilde{\delta}$, output it as $\tilde{b}$.

We first inspect the simulation in the case of $b = 0$ where dk and mk are independent each other. Our concern is the similarity of the simulation and $\mathsf{GAME.2}$ at $\delta = 0$. In the encryption oracle, ϕ is randomly generated. Hence dk, mk, and ϕ are all independent. This is the case in $\mathsf{GAME.2}$ at $\delta = 0$. With regard to the decryption oracle, let $E1$ denote an event that $K_j = K_1$ and the MAC is correct. Unless $E1$ happens, the output of the decryption oracle is correct; Especially, observe that dk_j is correctly related to K_j as well as those returned by the real decryption oracle in $\mathsf{GAME.2}$. On the other hand, if $E1$ happens, the simulating decryption oracle returns dk that is independent of mk while the real decryption oracle in $\mathsf{GAME.2}$ returns dk' generated together with mk. Accordingly, the simulation is done just as prescribed for $\mathsf{GAME.2}$ at $\delta = 0$ unless $E1$ happens. Hence we have

$$|\Pr[X2 \,|\, \delta = 0] - \Pr[\tilde{b} = 0 \,|\, b = 0]| \le \Pr[E1]. \tag{15}$$

Leaving $\Pr[E1]$ as is for a while, we next inspect the case of $b = 1$ where dk and mk came from the same application of KDF_2. We consider the similarity of the simulation and $\mathsf{GAME.3}$ at $\delta = 0$. The encryption oracle is perfectly done as prescribed because ϕ is independently generated while dk and mk are correctly related as expected in $\mathsf{GAME.3}$ at $\delta = 0$. Unlike the case of $b = 0$, the decryption oracle is perfect because it is expected to return the correctly related $dk(= dk')$ when $K_j = K_1$ and correct MAC is observed. Accordingly, we have

$$\Pr[X3 \,|\, \delta = 0] = \Pr[\tilde{b} = 0 \,|\, b = 1]. \tag{16}$$

From (15), (16), and Corollary 2.2 in Section 2.6, we have

$$\begin{aligned}|\Pr[X2 \,|\, \delta = 0] - \Pr[X3 \,|\, \delta = 0]| &\le \Pr[E1] + \Pr[\tilde{b} = 0 \,|\, b = 0] - \Pr[\tilde{b} = 0 \,|\, b = 1] \\ |\Pr[X2] - \Pr[X3]| &\le 2\epsilon_{\text{kdf}} + \frac{1}{2}\Pr[E1] \qquad (17)\end{aligned}$$

We now claim that

$$\Pr[E1] \leq q_D(\epsilon_{\mathrm{mac}} + 2\epsilon_{\mathrm{kdf}}). \tag{18}$$

To prove this claim, we first modify the attack game for one-time secure MAC (GAME.MAC in Section 2.5) in such a way that the MAC key mk is generated by $(*, mk) \leftarrow \mathsf{KDF}_2(K)$ where K is chosen randomly from $\mathcal{K}_K$. Let GAME.KDF-MAC refer to this modified game. Let $\epsilon_{\text{kdf-mac}}$ denote the maximum success probability in GAME.KDF-MAC over all ppt machines. By a straightforward reduction, we can prove that if there exists an adversary that wins GAME.KDF-MAC with noticeably better probability than that in GAME.MAC, then there exists an adversary that successfully break the security of KDF. Namely, $|\epsilon_{\text{kdf-mac}} - \epsilon_{\mathrm{mac}}| \leq 2\epsilon_{\mathrm{kdf}}$. Hence we have

$$\epsilon_{\text{kdf-mac}} \leq \epsilon_{\mathrm{mac}} + 2\epsilon_{\mathrm{kdf}}. \tag{19}$$

Now we show a reduction from A_T that causes event $E1$ to an adversary, A_{KM}, that attacks MAC in GAME.KDF-MAC. It is important to see that MAC key mk is totally independent from any other variables such as dk and ϕ in GAME.2 at $\delta = 0$ where $E1$ is defined. Specification of A_{KM} follows.

Step 1. Choose $j^\star \leftarrow \{1, \ldots, q_D\}$.

Step 2. Generate (pk, sk) by following KEM.Gen. Then compute $K_r \leftarrow \mathcal{K}_K$, $(dk, *) \leftarrow \mathsf{KDF}_2(K_r)$ and give (pk, dk) to A_T. Compute also $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$.

Step 3. Given τ from A_T, send $\tau||\phi$ to the MAC oracle of GAME.KDF-MAC and receive σ. Then return ϕ and σ A_T.

Step 4. For every decryption query $((\phi_j, \sigma_j), \tau_j)$ except $j = j^\star$, simulate the decryption oracle as follows. Compute $K_j \leftarrow \mathsf{KEM.Dec}_{sk}(\phi_j)$. If $K_j = K_1$, return $\perp$. Otherwise, compute $(dk_j, mk_j) \leftarrow \mathsf{KDF}_2(K_j)$ and verify $\mathsf{MAC.Ver}_{mk_j}(\sigma, \tau_j||\phi_j)$. If it is correct, return dk_j. Otherwise return $\perp$.

Step 5. On receiving $j^\star$-th decryption query, output σ_j and $\tau_j||\phi_j$ and halt.

Observe that dk and ϕ are created from K_r and K_1, respectively, and they are independent each other. Also observe that hidden mk chosen by the MAC oracle in GAME.KDF-MAC is generated by applying KDF_2 to random and independent K_0 just as well as that in GAME.2. Therefore, these variables distributes as in the case of $\delta = 0$ in GAME.2.

Next we verify the decryption oracle. Assume that $E1$ first happens at $j^\star$-th query. Then, all queries for $j < j^\star$ that causes $K_j \in \{K_0, K_1\}$ should be rejected. This is achieved in the simulation because:

- $K_j = K_1$ is detectable and rejected immediately.
- $K_j = K_0$ is not detectable in the simulation because K_0 is chosen by the MAC oracle. However, verifying the MAC by using given mk does the job since the MAC is incorrect as we assume $E1$ does not happen yet.

Therefore, the decryption oracle is perfectly simulated until $E1$ happens. If $E1$ happens at $j^\star$-th query, A_{KM} wins in GAME.KDF-MAC because the query is assumed not to be exactly the same as the challenge. This concludes the proof of equation (18).

From (17) and (18), we have $|\Pr[X_2] - \Pr[X_3]| \leq 2\epsilon_{\mathrm{kdf}} + \frac{q_D}{2}(\epsilon_{\mathrm{mac}} + 2\epsilon_{\mathrm{kdf}})$ which concludes the proof of (10).

Proof of (11). We show a reduction from A_T in $\mathsf{GAME.3}$ to A_K that attacks the underlying RCCA-secure KEM. The reduction is rather straightforward except the case where 'test' is returned from the decryption oracle of $\mathsf{GAME.RKEM}$. Specification of A_K follows.

Step 1. Given (pk, ϕ, K_b), compute $(dk, mk) \leftarrow \mathsf{KDF}_2(K_b)$. Then give (pk, dk) to A_T.

Step 2. Given τ from A_T, compute $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau||\phi)$ and return (ϕ, σ).

Step 3. Given decryption query $((\phi_j, \sigma_j), \tau_j)$ from A_T, send ϕ_j to the decryption oracle of $\mathsf{GAME.RKEM}$. If 'test' is returned, return dk if $\mathsf{MAC.Ver}_{mk}(\sigma_j, \tau_j||\phi_j) = 1$, otherwise, return $\perp$. If K_j is returned, set $(dk_j, mk_j) \leftarrow \mathsf{KDF}_2(K_j)$ and return dk_j if $\mathsf{MAC.Ver}_{mk_j}(\sigma_j, \tau_j||\phi_j) = 1$, otherwise, return $\perp$.

Step 4. When A_T outputs $\tilde{\delta}$, output $\tilde{b} = \tilde{\delta}$.

When $b = 1$, the encryption oracle is simulated so that (dk, mk) and ϕ are related correctly as in the case $\delta = 1$ in $\mathsf{GAME.3}$. On the other hand, when $b = 0$, the encryption oracle generates (dk, mk) and ϕ independently just as in the case $\delta = 0$ in $\mathsf{GAME.3}$. Similar observation holds for the decryption oracle. Accordingly, when $b = 1$ (and $b = 0$), the view of A_T is that of $\delta = 1$ (and $\delta = 0$, respectively) in $\mathsf{GAME.3}$. Hence

$$\begin{aligned}
\frac{1}{2}\Pr[X_3 \,|\, \delta = 0] + \frac{1}{2}\Pr[X_3 \,|\, \delta = 1] &= \frac{1}{2}\Pr[\tilde{b} = b \,|\, b = 0] + \frac{1}{2}\Pr[\tilde{b} = b \,|\, b = 1] \\
\Pr[X_3] &= \Pr[\tilde{b} = b] \\
&\leq \frac{1}{2} + \epsilon_{\text{rkem}}
\end{aligned}$$

as claimed in (11).