J. Cryptol. (2008) 21: 97–130
DOI: 10.1007/s00145-007-9010-x

Journal of
CRYPTOLOGY
© 2007 International Association for Cryptologic Research

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

Masayuki Abe
NTT Information Sharing Platform Laboratories, NTT Corporation, Tokyo, Japan
abe.masayuki@lab.ntt.co.jp

Rosario Gennaro
IBM T.J.Watson Research Center, Yorktown Heights, USA
Rosario@us.ibm.com

Kaoru Kurosawa
Ibaraki University, Hitachi, Japan
kurosawa@mx.ibaraki.ac.jp

Communicated by Ronald Cramer

Received 22 May 2005 and revised 22 June 2007
Online publication 8 November 2007

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 into 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 threshold hybrid one without considerable overhead, which may not be possible in the previous approach.

Key words. Tag-KEM, Hybrid encryption, Key encapsulation, Threshold encryption

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 [32] and developed by Rackoff and Simon [34], and Dolev, Dwork, and Naor [21]. Security against a chosen ciphertext attack (CCA security, in short) means that, even if the adversary is allowed to query a *decryption oracle* on ciphertexts of her choosing, then she obtains no useful information about messages encrypted in other ciphertexts. The first CCA-secure cryptosystems were presented in [21,32,34], 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 is 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. 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* (DEM) 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 [18,37] 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 [29], Kurosawa and Desmedt introduced a hybrid encryption scheme which is a modification of the hybrid scheme presented in [35]. 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 [26]. Nevertheless, the resulting scheme is CCA-secure and more efficient than those in [18,37] 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., [4,33] 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., [37], 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 [29].
– 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 (Sect. 2.1), DEM (Sect. 2.2), and PKE (Sect. 2.3) are used in Sect. 3 to construct our main result. Other building blocks are used in specific constructions or applications shown in Sects. 4 and 5, respectively.

2.1. *Key Encapsulation Mechanism with Tags (Tag-KEM)*

In Shoup's model, a KEM consists of three algorithms: key generation, encryption and decryption. The difference from public-key encryption is that the encryption algorithm takes as input only the public key pk and outputs a random one-time DEM key and its encryption. (See Sect. 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 efficiently sampleable spaces for tags and encapsulated keys denoted by $\mathcal{T}$ and $\mathcal{K}_D$, respectively.
$(\omega, dk) \leftarrow \mathsf{TKEM.Key}(pk)$	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 $\tau \in \mathcal{T}$, 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 represent abnormal termination.

$\mathsf{TKEM.Enc}$ may also output $\perp$ for irregular input $\tau \notin \mathcal{T}$.

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 correct one-time DEM key embedded in the challenge ciphertext (ψ, τ) or just a random string. The adversary is allowed to access to the decryption oracle, denoted by $\mathcal{O}$, which computes $\mathsf{TKEM.Dec}_{sk}(\cdot, \cdot)$. Let A_{T} be a probabilistic polynomial-time (ppt) oracle machine that plays the following game.

[GAME.TKEM: $\delta \in \{0, 1\}$]

Step 1. $(pk, sk) \leftarrow \mathsf{TKEM.Gen}(1^\lambda)$, $(\omega, dk_1) \leftarrow \mathsf{TKEM.Key}(pk)$, $dk_0 \leftarrow \mathcal{K}_D$.
Step 2. $(\tau, \rho) \leftarrow {A_{\mathrm{T}}}^{\mathcal{O}}(pk, dk_\delta)$.
Step 3. $\psi \leftarrow \mathsf{TKEM.Enc}(\omega, \tau)$.
Step 4. $\tilde{\delta} \leftarrow {A_{\mathrm{T}}}^{\mathcal{O}}(\rho, \psi)$.

In Step 4, A_{T} is not allowed not to ask (ψ, τ) to decryption oracle $\mathcal{O}$. The variable ρ is a state information of A_{T}, and dk_δ is set to either dk_0 or dk_1 according to the value of δ. Such convention is used throughout the paper unless otherwise noted.

Let $\mathrm{Exp}^{(\delta)}_{\mathrm{tkem}, A_{\mathrm{T}}}$ denote the event that A_{T} outputs 1 in GAME.TKEM at δ. Let $\epsilon_{\mathrm{tkem}, A_{\mathrm{T}}}$ denote the advantage of A_{T} in GAME.TKEM defined as

$$\epsilon_{\mathrm{tkem}, A_{\mathrm{T}}} = \left| \Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem}, A_{\mathrm{T}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem}, A_{\mathrm{T}}}] \right|. \tag{1}$$

Tag-KEM is CCA-secure if there exists a negligible function ϵ_{tkem} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{tkem}, A_{\mathrm{T}}} \leq \epsilon_{\mathrm{tkem}}$ holds for all ppt A_{T}. The probability is taken over all coin flips during the game.

Note that the above security definition simplifies the one presented in [3] (a preliminary version of this paper) 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 Similar Notions Tags associated to PKE or KEM can be found in the literature (e.g. see [37,38]), but they have different syntax and purposes; 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 the differences, their particular implementation fits into our model without essential modifications.

Tag-based PKE is also introduced in [30] with the same syntax as that of [37,38] but with a weaker security notion. In their work, the adversary is restricted so that the same tag associated to the challenge ciphertext must not be sent to the decryption oracle. Such a weak security is sufficient for some cryptographic applications, as shown in [30]. Though it does not fit into our framework, one of the constructions in [30] is identical to the one presented in Sect. 5.3 and indeed achieves our higher level of security.

The work in [28] 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 parameterized 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 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 ppt algorithm that plays the following game.

[GAME.DEM: $\xi \in \{0, 1\}$]

Step 1. $(m_0, m_1, \rho) \leftarrow A_\mathrm{D}(1^\lambda)$.
Step 2. $dk \leftarrow \mathcal{K}_D$, $\chi \leftarrow \mathsf{DEM.Enc}_{dk}(m_\xi)$.
Step 3. $\tilde{\xi} \leftarrow A_\mathrm{D}(\rho, \chi)$.

The messages, m_0 and m_1 must be the same length.

Let $\mathrm{Exp}^{(\xi)}_{\mathrm{dem}, A_\mathrm{D}}$ denote the event that A_D outputs 1 in GAME.DEM at ξ. Let $\epsilon_{\mathrm{dem}, A_\mathrm{D}}$ denote the advantage of A_D defined as

$$\epsilon_{\mathrm{dem}, A_\mathrm{D}} = \left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{dem}, A_\mathrm{D}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{dem}, A_\mathrm{D}}]\right|. \tag{2}$$

A DEM is one-time secure if there exists a negligible function ϵ_{dem} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{dem}, A_\mathrm{D}} \leq \epsilon_{\mathrm{dem}}$ holds for all ppt A_D. 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, $\mathsf{PKE.Gen}$, $\mathsf{PKE.Enc}$, and $\mathsf{PKE.Dec}$:

$(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$	A ppt 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 ppt algorithm that encrypts a message $m \in \mathcal{M}$ into a ciphertext c.

$m \leftarrow \mathsf{PKE.Dec}_{sk}(c)$	A polynomial-time 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 ppt oracle algorithm that plays the following game. By $\mathcal{O}$, we denote the decryption oracle, $\mathsf{PKE.Dec}_{sk}(\cdot)$

[GAME.PKE: $b \in \{0, 1\}$]

Step 1. $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^{\lambda})$.
Step 2. $(m_0, m_1, \rho) \leftarrow {A_{\mathrm{E}}}^{\mathcal{O}}(pk)$.
Step 3. $c \leftarrow \mathsf{PKE.Enc}_{pk}(m_b)$.
Step 4. $\tilde{b} \leftarrow {A_{\mathrm{E}}}^{\mathcal{O}}(\rho, c)$.

In Step 4, A_{E} is not allowed to ask c to $\mathcal{O}$. In addition, m_0 and m_1 must be of the same length. Let $\mathrm{Exp}^{(b)}_{\mathrm{pke},A_{\mathrm{E}}}$ denote the event that A_{E} outputs 1 in GAME.TKEM at b. Let $\epsilon_{\mathrm{pke},A_{\mathrm{E}}}$ denote the advantage of A_{E} in GAME.PKE defined as

$$\epsilon_{\mathrm{pke},A_{\mathrm{E}}} = \left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{pke},A_{\mathrm{E}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{pke},A_{\mathrm{E}}}]\right|. \tag{3}$$

A PKE is CCA-secure if there exists a negligible function ϵ_{pke} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{pke},A_{\mathrm{E}}} \leq \epsilon_{\mathrm{pke}}$ holds for all ppt A_{E}.

2.4. *Key Encapsulation Mechanism (KEM)*

A key encapsulation mechanism consists of three algorithms, KEM.Gen, KEM.Enc, and KEM.Dec defined as follows.

$(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^{\lambda})$	A ppt 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 ppt algorithm that generates key $K \in \mathcal{K}_K$ and its encryption ϕ.
$K \leftarrow \mathsf{KEM.Dec}_{sk}(\phi)$	A polynomial-time 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 building block to construct Tag-KEM in this paper, we consider 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 A be a ppt oracle machine that plays the following game. By $\mathcal{O}$ we denote the decryption oracle, $\mathsf{KEM.Dec}_{sk}(\cdot)$.

[GAME.KEM: $b \in \{0, 1\}$]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^{\lambda})$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $K_0 \leftarrow \mathcal{K}_K$.
Step 2. $\tilde{b} \leftarrow A_{\mathrm{K}}^{\mathcal{O}}(pk, \phi, K_b)$.

In Step 2, A_{K} is not allowed to ask ϕ to KEM.Dec.

Let $\mathrm{Exp}^{(b)}_{\mathrm{kem},A_{\mathrm{K}}}$ denote the event that A_{K} outputs 1 in $\mathsf{GAME.KEM}$ at b. Let $\epsilon_{\mathrm{kem},A_{\mathrm{K}}}$ denote the advantage of A_{K} in $\mathsf{GAME.KEM}$ defined as

$$\epsilon_{\mathrm{kem},A_{\mathrm{K}}} = \left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{kem},A_{\mathrm{K}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{kem},A_{\mathrm{K}}}]\right|. \tag{4}$$

A KEM is secure against adaptive chosen ciphertext attacks (CCA secure) if there exists a negligible function ϵ_{kem} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{kem},A_{\mathrm{K}}} \le \epsilon_{\mathrm{kem}}$ holds for all ppt A_{K}.

2.5. *Message Authentication Code (MAC)*

A MAC scheme is a pair of algorithms ($\mathsf{MAC.Sign}$, $\mathsf{MAC.Ver}$) and a key-space $\mathcal{K}_M$ indexed by a security parameter λ. Typically, $\mathcal{K}_M = \{0,1\}^{\lambda}$. Functions work as follows.

$\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$	A deterministic algorithm that takes a mac-key $mk \in \mathcal{K}_M$ and a message $\tau \in \{0,1\}^*$. It outputs a string σ called MAC.
$0/1 \leftarrow \mathsf{MAC.Ver}_{mk}(\sigma,\tau)$	A deterministic algorithm that takes mac-key $mk \in \mathcal{K}_M$, MAC σ, and message τ and outputs 1 if $\sigma = \mathsf{MAC.Sign}_{mk}(\tau)$, or outputs 0, otherwise.

A message with MAC, (σ,τ), is valid with regard to mk if $1 = \mathsf{MAC.Ver}_{mk}(\sigma,\tau)$.

Let A_{M} be a ppt algorithm that plays the following game

[GAME.MAC]

Step 1. $(\tau,\rho) \leftarrow A_{\mathrm{M}}(1^{\lambda})$.
Step 2. $mk \leftarrow \mathcal{K}_M$, $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$.
Step 3. $(\sigma',\tau') \leftarrow A_{\mathrm{M}}(\rho,\sigma)$.

Let $\mathrm{Exp}_{\mathrm{mac},A_{\mathrm{M}}}$ denote the event that $(\sigma',\tau') \neq (\sigma,\tau)$ and $\mathsf{MAC.Ver}_{mk}(\sigma,\tau) = 1$. Let $\epsilon_{\mathrm{mac},A_{\mathrm{M}}}$ denote the success probability of A_{M} in $\mathsf{GAME.MAC}$ defined as

$$\epsilon_{\mathrm{mac},A_{\mathrm{M}}} = \Pr[\mathrm{Exp}_{\mathrm{mac},A_{\mathrm{M}}}]. \tag{5}$$

A MAC is secure against one-time chosen message attacks if there exists a negligible function ϵ_{mac} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{mac},A_{\mathrm{M}}} \le \epsilon_{\mathrm{mac}}$ holds for all ppt A_{M}.

2.6. *Key Derivation Function (KDF)*

We require a key derivation function, say KDF_2, that maps a key K generated by KEM into a pair of keys (dk, mk) for DEM and MAC. More precisely, let KDF_2 be a map: $\mathcal{K}_K \to \mathcal{K}_D \times \mathcal{K}_M$, where KDF_2, $\mathcal{K}_K$, $\mathcal{K}_D$ and $\mathcal{K}_M$ are all indexed by a security parameter λ. (Extra keys may also be used as an index if needed.)

We require that (dk, mk) is pseudorandom when K is uniformly chosen. Let

$$\mathcal{D}_1 = \{(dk,mk) \mid K \leftarrow \mathcal{K}_K, (dk,mk) \leftarrow \mathsf{KDF}_2(K)\}, \quad \text{and}$$
$$\mathcal{D}_0 = \{(dk,mk) \mid (dk,mk) \leftarrow \mathcal{K}_D \times \mathcal{K}_M\}.$$

Let A_F be a ppt algorithm that plays the following game. Input $\mathcal{D}$ is a distribution defined over $\mathcal{K}_D \times \mathcal{K}_M$.

[GAME.KDF: $\mathcal{D}$]

Step 1. $(dk, mk) \leftarrow \mathcal{D}$.
Step 2. $\tilde{b} \leftarrow A_F(dk, mk)$.

Let $\Pr[\mathrm{Exp}^{(\mathcal{D}_i)}_{\mathrm{kdf},A_F}]$ denote the event that A_F outputs 1 in GAME.KDF with distribution $\mathcal{D}_i \in \{\mathcal{D}_0, \mathcal{D}_1\}$. A KDF_2 is secure if there exists a negligible function ϵ_{kdf} in λ such that, for sufficiently large λ,

$$\left|\Pr[\mathrm{Exp}^{(\mathcal{D}_0)}_{\mathrm{kdf},A_F}] - \Pr[\mathrm{Exp}^{(\mathcal{D}_1)}_{\mathrm{kdf},A_F}]\right| \le \epsilon_{\mathrm{kdf}} \tag{6}$$

holds for all ppt algorithm A_F.

2.7. *Security Properties of Hash Functions*

Collision-Resistance (CR): A hash function H is said to be collision resistant if it is hard to find (x, x') such that $H(x) = H(x')$. Formally, let $\mathcal{H}_\lambda$ be a collection of hash functions such that $\mathcal{H}_\lambda = \{H : \{0,1\}^* \to \{0,1\}^\lambda\}$. Define $\mathcal{H} = \{\mathcal{H}_\lambda\}_{\lambda\in\mathbb{N}}$. Consider the following game, where A_{CR} is a ppt algorithm called a collision finder.

[GAME.CR]

Step 1. $H \leftarrow \mathcal{H}_\lambda$.
Step 2. $(x, x') \leftarrow A_{\mathrm{CR}}(H)$.

Let $\mathrm{Exp}_{\mathrm{cr}}$ denote the event such that $H(x') = H(x)$ and $x' \neq x$ in GAME.CR. We say that $\mathcal{H}$ is collision resistant if there exists a negligible function ϵ_{cr} in λ such that, for sufficiently large λ, $\Pr[\mathrm{Exp}_{\mathrm{cr}}] \le \epsilon_{\mathrm{cr}}$ holds for all ppt A_{CR}. For simplicity, we also say that $\mathcal{H}_\lambda$ (or even H) is collision resistant.

Target Collision-Free (TC): A hash function H is target collision-free if, for a randomly chosen (H, x), it is hard to find x' such that $H(x) = H(x')$. It is a special case of universal one-way. Formally, let $\mathcal{X}_\lambda = \{X\}$ be a collection of domains and $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$. Let $\mathcal{H}_\lambda = \{H : X \to \{0,1\}^\lambda \mid X \in \mathcal{X}_\lambda\}$ 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 ppt algorithm that plays the following game.

[GAME.TCH]

Step 1. $H \leftarrow \mathcal{H}_\lambda$, $x \leftarrow X$.
Step 2. $x' \leftarrow A_{\mathrm{TCH}}(H, x)$.

Let $\mathrm{Exp}_{\mathrm{tch}}$ denote the event such that $x' \in X$ and $H(x') = H(x)$ in GAME.TCH. We say that $\{\mathcal{H}_\lambda\}$ is target collision-free with respect to $\mathcal{X}$ if there exists a negligible function ϵ_{tch} in λ such that $\Pr[\mathrm{Exp}_{\mathrm{tch}}] \le \epsilon_{\mathrm{tch}}$ holds for all polynomial-time A_{TCH} for sufficiently large λ.

Random Prefix Collision-Free (RPH): Random prefix collision-free is a notion in between target collision-free and collision-free; For a randomly chosen H, the adversary first outputs x. It is then given a random prefix r and asked to output r' and x' such that $H(r,x) = H(r',x')$.

Formally, 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 ppt algorithm that plays the following game.

[GAME.RPH]

```
Step 1. $H \leftarrow \mathcal{H}_\lambda$.
Step 2. $(\rho, x) \leftarrow A_{\mathrm{RPH}}(H)$.
Step 3. $r \leftarrow R$.
Step 4. $(r', x') \leftarrow A_{\mathrm{RPH}}(\rho, r)$ such that $r' \in R$ and $x' \in X$.
```

Let $\mathrm{Exp}_{\mathrm{rph}}$ denote the event such that $H(r',x') = H(r,x)$ and $(r',x') \neq (r,x)$ in GAME.RPH. We say that $\mathcal{H}$ is random prefix collision-free with regard to $\mathcal{X}$ and $\mathcal{R}$ if there exists a negligible function ϵ_{rph} in λ such that, for sufficiently large λ, $\Pr[\mathrm{Exp}_{\mathrm{rph}}] \le \epsilon_{\mathrm{rph}}$ holds for all ppt A_{RPH}.

3. Generic Construction of Hybrid PKE

In GAME.TKEM, it is important to see that, for challenge ciphertext (ψ, τ) the adversary is allowed to ask (ψ_j, τ_j) to the decryption oracle even if $\psi = \psi$ as long as $\tau_j \neq \tau$. Therefore, to achieve the CCA-security, Tag-KEM must provide some sort of integrity to τ so that making any query with $\tau_j \neq \tau$ cannot be useful for the adversary. We exploit this property to protect the DEM part and relax the required security.

Suppose that there exists a Tag-KEM such that any string χ can be used as a tag, i.e., $\mathcal{T} = \{0,1\}^*$. We then construct a hybrid PKE as follows. 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: PKE.Enc$_{pk}(m)$
    $(\omega, dk) \leftarrow$ TKEM.Key$(pk)$
    $\chi \leftarrow$ DEM.Enc$_{dk}(m)$
    $\psi \leftarrow$ TKEM.Enc$(\omega, \chi)$
    Output $c = (\psi, \chi)$
```

```
Function: PKE.Dec$_{sk}(c)$
    $(\psi, \chi) \leftarrow c$
    $dk \leftarrow$ TKEM.Dec$_{sk}(\psi, \chi)$
    $m \leftarrow$ DEM.Dec$_{dk}(\chi)$
    Output $m$
```

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

The following theorem states the security of the above hybrid PKE.

Theorem 1 (Tag-KEM/DEM composition theorem). *If the Tag-KEM is CCA secure and the DEM is one-time secure then the above hybrid PKE scheme is CCA secure. In particular,* $\epsilon_{\mathrm{pke},A_{\mathrm{E}}} \le 2\epsilon_{\mathrm{tkem}} + \epsilon_{\mathrm{dem}}$ *holds for all ppt algorithm* A_{E}.

Proof. We consider a series of games GAME.0, ..., GAME.3. Let X_i denote the event that adversary A_{E} outputs 1 in GAME.i.

GAME.0: Let GAME.0 be GAME.PKE with $b = 0$. Then

$$\Pr[\mathrm{Exp}^{(0)}_{\mathrm{pke},A_{\mathrm{E}}}] = \Pr[X_0]. \tag{7}$$

Note that m_0 is encrypted at Step-3 in GAME.0.

GAME.1: Use a random key, $dk^{\times} \leftarrow \mathcal{K}_D$, to encrypt m_0 in Step-3 of GAME.0.
We claim that

$$|\Pr[X_0] - \Pr[X_1]| \le \epsilon_{\mathrm{tkem}}. \tag{8}$$

The claim is proven by constructing A_{T} that attacks the underlying Tag-KEM scheme as in GAME.TKEM by using A_{E} as follows. Given (pk, dk_δ), A_{T} sends pk to A_{E}. Given (m_0, m_1) from A_{E}, A_{T} computes $\chi = \mathsf{DEM.Enc}_{dk_\delta}(m_0)$ and outputs χ as the target tag. It then receives ψ as a challenge in GAME.TKEM. Ciphertext (ψ, χ) is then sent to A_{E} as a challenge in GAME.PKE. Each decryption query, (ψ_i, χ_i), from A_{E} is forwarded to decryption oracle TKEM.Dec that takes ψ_i as a ciphertext and χ_i as a tag. On receiving dk_i from TKEM.Dec, A_{T} returns $m_j = \mathsf{DEM.Dec}_{dk_i}(\chi_i)$ to A_{E}. (If $dk_i = \perp$, A_{T} returns $m_j = \perp$.) When A_{E} outputs $\tilde{b}$, A_{T} outputs $\tilde{\delta} = \tilde{b}$ and halts.

Observe that the decryption oracle for A_{E} is perfectly simulated because the correct decryption key is obtained from TKEM.Dec for every query. Next observe that, when $\delta = 1$, dk_δ is the correct key embedded in ψ and the view of A_{E} is thus identical to that in GAME.0. Hence we have $\Pr[X_0] = \Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem},A_{\mathrm{T}}}]$. Similarly, when $\delta = 0$, dk_δ is just a random key used only for computing χ and the view of A_{E} is thus identical to that in GAME.1. Hence $\Pr[X_1] = \Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_{\mathrm{T}}}]$. Accordingly, we have $|\Pr[X_0] - \Pr[X_1]| = |\Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem},A_{\mathrm{T}}}] - \Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_{\mathrm{T}}}]| \le \epsilon_{\mathrm{tkem}}$ as claimed.

GAME.2: Encrypt m_1 instead of m_0 in GAME.1.
We claim that

$$|\Pr[X_1] - \Pr[X_2]| \le \epsilon_{\mathrm{dem}}. \tag{9}$$

Namely, A_{E} playing GAME.1 and GAME.2 essentially conducts a passive attack to DEM. The claim is proven by constructing A_{D} that plays GAME.DEM by using A_{E} as follows. A_{D} first generates (pk, sk) by using PKE.Gen and gives pk to A_{E}. Given (m_0, m_1) from A_{E}, A_{D} outputs (m_0, m_1) and receives challenge ciphertext χ. It then computes ψ by following TKEM.Key and TKEM.Enc by using χ as a tag, and sends challenge ciphertext $c = (\psi, \chi)$ to A_{E}. All decryption queries are appropriately processed by using sk. When A_{E} outputs $\tilde{b}$, A_{D} outputs $\tilde{\xi} = \tilde{b}$.

The major factor of the running time of A_{D} is that of A_{E} and that for simulating the decryption oracle which grows only linearly in the number of decryption queries. Observe now that, when $\xi = 0$ in GAME.DEM, χ is an encryption of m_0 as in GAME.1. Also observe that key $dk^{\times}$ used for making the challenge ciphertext in GAME.DEM and the key embedded in ψ are independent and randomly chosen just as well as those in GAME.1. Hence the view of A_{E} is identical to that in GAME.1. We thus have $\Pr[X_1] = \Pr[\mathrm{Exp}^{(0)}_{\mathrm{dem},A_{\mathrm{D}}}]$. One can argue in the same way that the view of A_{E} is identical to that in GAME.2 when $\xi = 1$. Hence $\Pr[X_2] = \Pr[\mathrm{Exp}^{(1)}_{\mathrm{dem},A_{\mathrm{D}}}]$ holds. Accordingly, $|\Pr[X_1] - \Pr[X_2]| = |\Pr[\mathrm{Exp}^{(0)}_{\mathrm{dem},A_{\mathrm{D}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{dem},A_{\mathrm{D}}}]| \le \epsilon_{\mathrm{dem}}$ as in the claim.

GAME.3: Use correct dk generated by TKEM.Key for DEM.Enc in Step-3 of GAME.2. Just as well as (8), we obtain

$$|\Pr[X_2] - \Pr[X_3]| \leq \epsilon_{\text{tkem}}. \tag{10}$$

Observe also that GAME.3 is the same as GAME.PKE with $b = 1$. Hence

$$\Pr[X_3] = \Pr[\text{Exp}^{(1)}_{\text{pke},A_\text{E}}]. \tag{11}$$

From (7), (8), (9), (10) and (11), we have

$$\epsilon_{\text{pke},A_\text{E}} = \left|\Pr[\text{Exp}^{(0)}_{\text{pke},A_\text{E}}] - \Pr[\text{Exp}^{(1)}_{\text{pke},A_\text{E}}]\right| = |\Pr[X_0] - \Pr[X_3]| \leq 2\epsilon_{\text{tkem}} + \epsilon_{\text{dem}}$$

as in the theorem. □

4. Construction of Tag-KEM

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

4.1. *Based on PKE with Long Plaintext*

Our first construction of Tag-KEM is just to encrypt (dk, τ) by using a CCA-secure PKE. It indeed works well if the PKE accepts a long enough plaintext. Lengthy tags τ would be compressed by using a hash function.

Formally, we construct Tag-KEM from PKE as follows. TKEM.Gen is essentially the same as 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.) TKEM.Key chooses random dk from $\mathcal{K}_D$. It also outputs state information $\omega = pk \parallel dk$. TKEM.Enc and TKEM.Dec are as follows.

```
Function: TKEM.Enc($\omega, \tau$)
    $(pk, dk) \leftarrow \omega$
    $\tau' = H(\tau)$
    $\psi = \text{PKE.Enc}_{pk}(dk \parallel \tau')$
    Output $\psi$
```

```
Function: TKEM.Dec$_{sk}(\psi, \tau)$
    $dk \parallel \tau' \leftarrow \text{PKE.Dec}(sk, \psi)$
    If $\tau' = H(\tau)$, return $dk$
    Return $\perp$, otherwise
```

Theorem 2. *If PKE is CCA-secure and H is collision resistant, the above Tag-KEM is CCA-secure. In particular, $\epsilon_{\text{tkem},A_\text{T}} \leq \epsilon_{\text{pke}} + 2\epsilon_{\text{cr}}$ holds for all ppt algorithm A_T.*

Proof is given in Appendix A. One efficient implementation would be to use Rabin-SAEP+ [8] encryption, where the message length is shorter than that of RSA but sufficient for encrypting a standard DEM key and a hashed tag. One can also apply the technique of [24] to shorten the ciphertext.

4.2. *Based on CCA-Secure KEM and MAC*

In this section we construct a CCA-secure Tag-KEM from a CCA-secure KEM and a MAC. The idea is to encrypt a random key K using the KEM, and apply KDF_2 to K to extract two keys dk and mk; one for encryption and the other for creating MAC.

Formally, we construct a Tag-KEM as follows. Let $\Pi_{\mathrm{L}} = (\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$, $\mathrm{M} = (\mathsf{MAC.Sign}, \mathsf{MAC.Ver})$, and $\mathsf{KDF}_2 : \mathcal{K}_K \to \mathcal{K}_D \times \mathcal{K}_M$ be KEM, MAC, and KDF, respectively. $\mathsf{TKEM.Gen}$ 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 $\omega = (mk, \phi)$.

$\mathsf{TKEM.Enc}$ and $\mathsf{TKEM.Dec}$ are as follows.

```
Function: $\mathsf{TKEM.Enc}(\omega, \tau)$
    $(mk, \phi) \leftarrow \omega$
    $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$
    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 ϕ. On the other hand the security of the MAC will prevent an adversary from gaining any advantage by manipulating MAC σ.

Applying Theorem 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. 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 construction, there is some redundancy at play. If a KEM is combined with a MAC as shown in Sect. 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, we show a new security notion of KEM that can be strictly weaker than CCA but sufficient to yield CCA-secure Tag-KEM when combined with a MAC.

Predicate-dependent CCA Security Let Π_{L} be a KEM as in Sect. 4.2. Let $\mathcal{P} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be a poly-time computable predicate.

For Π_{L} and $\mathcal{P}$, let $\mathcal{VD}$ be a decryption oracle that takes $(\phi_i, \eta_i) \in \{0,1\}^* \times \{0,1\}^*$ and returns $\mathsf{KEM.Dec}_{sk}(\phi_i)$ if $\mathcal{P}(\mathsf{KEM.Dec}_{sk}(\phi_i), \eta_i) = 1$, or returns $\perp$ otherwise. We then define the following attack game.

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

[GAME.LKEM: $b \in \{0, 1\}$]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $K_0 \leftarrow \mathcal{K}_K$.
Step 2. $\tilde{b} \leftarrow A_{\mathrm{L}}^{\mathcal{V}D(\cdot,\cdot)}(pk, \phi, K_b)$.

It is required that $\phi_i \neq \phi$ for all ϕ_i sent to $\mathcal{V}D$. Let $\mathrm{Exp}^{(b)}_{\mathrm{lkem},A_{\mathrm{L}}}$ denote the event that A_{L} outputs 1 in GAME.LKEM with b. Define $\epsilon_{\mathrm{lkem},A_{\mathrm{L}}}$ as

$$\epsilon_{\mathrm{lkem},A_{\mathrm{L}}} = \left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{lkem},A_{\mathrm{L}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{lkem},A_{\mathrm{L}}}]\right|. \tag{12}$$

A KEM is LCCA secure with respect to predicate $\mathcal{P}$ if there exists a negligible function ϵ_{lkem} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{lkem},A_{\mathrm{L}}} \leq \epsilon_{\mathrm{lkem}}$ for all ppt algorithm A_{L}.

The above definition may seem too general since in the useful case shown later we only consider MAC.Ver as $\mathcal{P}$. Nevertheless, a general treatment is helpful to specify what property is really needed. Also in some cases, it makes the security analysis slightly simpler as will be 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 Sect. 4.2. Hence a very weak instance may exist in the class.

Proof of the Tag-KEM in Sect. 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 Sect. 4.2, $\mathcal{P}$ is not used in 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 the underlying KEM. As we could have predicted, this is very close to the combination of a CCA KEM and a MAC (but not exactly the same). Nevertheless, we formally prove the security to see that the MAC plays the different roles without inconsistency.

Theorem 3. *If Π_{L} is LCCA secure with respect to $\mathcal{P}^{\mathrm{mac}}$ then the Tag-KEM defined in Sect. 4.2 is CCA secure. In particular, $\epsilon_{\mathrm{tkem},A_{\mathrm{T}}} \leq 2\epsilon_{\mathrm{lkem}} + q_D\,\epsilon_{\mathrm{mac}} + 2\epsilon_{\mathrm{kdf}}$ holds for any ppt algorithm A_{T} that makes at most q_D 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 is LCCA-secure (with respect to $\mathcal{P}^{\mathrm{mac}}$) 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 particular 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 Sect. 5.4 for further discussion.

4.4. *Based on KEM with Hash Function*

We show another approach of constructing Tag-KEM. It is useful when your PKE does not have enough plaintext length as needed in Sect. 4.1 and/or increasing ciphertext length as in Sect. 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. Following this approach, we constructs two Tag-KEMs based on OAEP+ [36] and Cramer-Shoup KEM [35]. The security proofs are essentially unchanged from those of the original schemes. (Although a generic construction of this approach can be shown, it would not be quite useful due to its complexity. Showing that a PKE/KEM fits into the generic framework may not be meaningfully 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 scheme.)

In the following Tag-KEM, the original PKE or KEM is obtained just by deleting the tag τ.

4.4.1. *From OAEP+*

Let f be a one-way trapdoor permutation. The encryption function of Tag-KEM is constructed from OAEP+ by encrypting dk and tag τ as follows:

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

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

The security is argued in the same way as the original OAEP+ except for the case in which the adversary successfully creates a valid ciphertext (ψ, τ') from challenge ciphertext (ψ, τ). In such a case, (ψ, τ') is valid only if $H'(r \parallel dk \parallel \tau) = H'(r \parallel dk \parallel \tau')$ holds since ψ uniquely identifies r, r' and K. 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 GAME.0' in such a way that the decryption oracle returns $\perp$ for any (ψ, τ') with $\tau' \neq \tau$. The rest of the security proof is done in the same way as in the original paper [36] except 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*

Based on the Cramer-Shoup Encryption (actually its KEM variant), a Tag-KEM is obtained by including a tag in the hash function used in the encryption algorithm. Let G_q be a multiplicative group of prime order q. 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 algorithm of Tag-KEM outputs $dk = h^r$ and ciphertext (u_1, u_2, v) such that

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

where $r \in Z_q$ is random and H is a hash function. The decryption algorithm outputs $dk = u_1^{z_1} u_2^{z_2}$ if $v \overset{?}{=} u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$. It outputs $\perp$ otherwise. Applying Theorem 1 results in the hybrid PKE mentioned briefly in [16].

The Cramer-Shoup encryption requires H to be Target Collision-Free (as defined in Sect. 2.7) since all inputs to the hash function, i.e., u_1 and u_2, are chosen randomly in advance. On the other hand, $\mathsf{GAME.TKEM}$ allows the adversary to choose τ after seeing the description of H and then u_1 and u_2 are randomly chosen. Hence we require H to be Random Prefix Collision-Free as formally defined in Sect. 2.7.

It holds that

$$\text{(Collision-Free)} \Rightarrow \text{(Random Prefix Collision-Free)} \Rightarrow \text{(Target Collision-Free)}.$$

Hence it is reasonable to use cryptographic hash functions like SHA-1 which can be assumed collision-free.

Theoretically, we do not know if there exists a separation between Collision-Freeness and Random Prefix Collision-Freeness. It is not clear how to launch a birthday attack in $\mathsf{GAME.RPH}$ (if the randomness of r affects to the output in $\mathsf{GAME.RPH}$). Also, we do not know constructions of random prefix collision-free hash functions from target collision-free or universal one-way hash functions. We thus 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 it hardly affects to the performance as long as $|\alpha|$ is smaller than $|q|$.

4.5. *Based on ID-based PKE*

An ID-based encryption scheme is selective-ID secure if 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. Efficient selective-ID ID-based encryption schemes (sIBE in short) based on standard cryptographic assumptions are presented in [9].

It is shown in [15] that sIBE can be transformed into a CCA-secure PKE by using strong one-time signature. In [10], Boneh and Katz improved the efficiency of [15] by replacing the one-time signature with a commitment scheme (using hash function) and a MAC. We show that the conversion from sIBE to CCA PKE also yields a CCA-secure Tag-KEM (without adopting the result of Sect. 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. We then construct a Tag-KEM scheme as follows: $\mathsf{TKEM.Key}(pk)$ simply chooses dk randomly. Then $\mathsf{TKEM.Enc}$ encrypts dk and τ into ciphertext $\psi = (vk, \phi, \sigma)$ as

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

Decryption is rather trivial; $\mathsf{TKEM.Dec}$ first verifies the signature σ using vk and then decrypts the rest of the parts.

The CCA-secure PKE of [15] is obtained by deleting τ from the above Tag-KEM. The security proof is almost the same as in [15] 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].

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. In the (n, k)-threshold model, 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. The corruption can be static (i.e. done before starting the protocol) or dynamic (i.e., done adaptively during the protocol). For simplicity, we assume that a trusted party generates the key, and shares it among the servers. A distributed key generation protocol can be used instead if desired.

The CCA-security of threshold PKE is defined as a natural extension of the CCA-security of regular (non-threshold) PKE as in [38]. 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.

Theoretically, results from general multi-party computation, e.g., [5,25], 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 constructions have been pursued starting from [20]. In the standard model, the first CCA-secure threshold PKE is presented in [13] followed by, e.g., [1,2,11,12,27]. However, no efficient threshold *hybrid* PKE, is known in the standard model, in particular 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 CCA-secure threshold hybrid PKE like the standard KEM/DEM composition. However, an efficient threshold DEM seems difficult to obtain due to its use of symmetric key techniques such as block ciphers and MAC.

Can we then combine a CCA-secure threshold KEM and a CCA-secure standard (i.e., non-threshold) 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). Hence the adversary can 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 one-time secure (as defined in Sect. 2.2) so that the session-key can be securely exposed on decryption. Remember that one-time 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 the security in the Tag-KEM/DEM framework. The combination of threshold Tag-KEM and a one-time secure DEM thus results in a CCA-secure threshold hybrid PKE. In other words, by replacing Tag-KEM with its *threshold* version, we can obtain a *threshold* Tag-KEM/DEM framework.

A formal security definition of threshold Tag-KEM can be derived from the definition of threshold PKE in [38]. The following composition theorem can be proven by translating the proof of Theorem 1 to the threshold setting. The proof is omitted to avoid redundancy.

Theorem 4 (Threshold Tag-KEM/DEM composition theorem). *If a threshold Tag-KEM is threshold-CCA secure and a DEM is one-time secure then their Tag-KEM/DEM composition yields a threshold-CCA secure hybrid PKE scheme which retains the same threshold and corruption model as those for the threshold Tag-KEM. In particular,* $\epsilon^{(n,k)}_{\text{th-pke},A} \leq 2\epsilon^{(n,k)}_{\text{th-tkem}} + \epsilon_{\text{dem}}$ *holds for all polynomial-time adversary A where* $\epsilon^{(n,k)}_{\text{th-pke},A}$ *and* $\epsilon^{(n,k)}_{\text{th-tkem}}$ *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 [38] 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 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 [1,13], and the conversion technique in Sect. 4.4 (or result of Sect. 4.1 with larger security parameter) can be used to obtain a threshold Cramer-Shoup Tag-KEM. Accordingly, by following the threshold version of Theorem 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 [29], 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 those in the Cramer-Shoup encryption shown in Sect. 4.4. Encryption of message $m \in \{0, 1\}^*$ is:

$$u_1 = g_1^r, \qquad u_2 = g_2^r, \qquad \alpha = H(u_1 \parallel u_2), \qquad v = c^r d^{\alpha r},$$
$$(dk, mk) \leftarrow \mathsf{KDF}_2(v), \qquad \chi = G(dk) \oplus m, \qquad \sigma = \mathsf{MAC.Sign}_{mk}(\chi),$$

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-secure [26].

Our framework offers 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 [26]. Hence it does not fulfill the requirement stated in Sect. 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 [22] 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:

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

K is uniformly chosen from $\{0, 1\}^\lambda$. The ciphertext is (ψ, χ). The resulting hybrid PKE is CCA-secure if PKE is one-way, DEM is one-time secure and $\mathsf{DEM.Enc}$ is a bijection.

Now one can observe that $\mathsf{PKE.Enc}_{pk}(K; H(K \parallel \tau))$ 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:

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

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;

$$\psi = f(r), \qquad \sigma = H(r \parallel m), \qquad \chi = G(r) \oplus m.$$

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

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

REACT REACT-RSA [33] is very similar to the above Bellare-Rogaway scheme;

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

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 $\sigma = H(r \parallel \chi)$ 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 $\psi = (f(r), H(r \parallel \tau))$ where H is a random oracle. Such a scheme also appears in [30].

Some ISO Standard Candidates Finally, as mentioned in Sect. 4.1, KEM schemes based on RSA and HIME described in [37] 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 [14] 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 was introduced in [14]. 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 [37]), 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 [14], it is shown that combining an RCCA-secure PKE and a CCA-secure symmetric encryption can yield a CCA-secure hybrid PKE. For a one-time secure DEM and a one-time MAC, their construction is summarized as follows. Given message m, output ciphertext (ϕ, χ, σ) such that;

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

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.

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

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.

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

It can be seen as a variant of the construction shown in Sect. 4.2; MAC is applied to $\tau \parallel \phi$ rather than to τ. In Appendix E, we give a 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 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 schemes by extending the CCA KEM/DEM framework. The new Tag-KEM/DEM framework captures a wide variety of schemes and can yield better hybrid encryption schemes especially in the size of the ciphertext. Several schemes are improved on other aspects as well according to our framework.

Yet there are some situations where the traditional KEM/DEM framework is useful. For instance, schemes that follow the 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 an entire ciphertext to derive dk. We also note that some Tag-KEM/DEM schemes provide the streaming feature. (The scheme based on Cramer-Shoup shown in Sect. 4.4 is an example). It is also known that the KEM/DEM framework can be extended to establish some limited form of secure channels [31] (where no forward security is considered) while such extension is not available in Tag-KEM/DEM.

Finally, we list some open problems as follows.

On the Tag-KEM Security Can the security of Tag-KEM be relaxed? Although the tag is chosen by the adversary in GAME.TKEM the adversary cannot select an arbitrary tag any more once the Tag-KEM is combined with a DEM since the tag is a ciphertext encrypted with a random one-time key. If the DEM provides the strong property such that the ciphertext is indistinguishable from random strings of the same length, replacing the ciphertext with a random string offsets the choice of the adversary. This observation suggests that we may be able to relax the security requirement for Tag-KEM in such a way that the tag is chosen randomly rather than chosen by the adversary. Can we prove a variant of Theorem 1 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 Sect. 4.4? This questions is closely related to the previous one. If the tag is chosen randomly rather than chosen by the adversary, target collision free 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).

Acknowledgements

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

Appendix A. Proof of Theorem 2

Let A_{T} be an adversary for the Tag-KEM given in Sect. 4.1. By using A_{T}, we construct A_{E} that plays GAME.PKE to attack the underlying PKE as follows.

1. Given pk, A_{E} chooses $(dk_0, dk_1) \leftarrow \mathcal{K}_D \times \mathcal{K}_D$ and sends pk and dk_1 to A_{T}.
2. Given τ from A_{T}, A_{E} outputs $(dk_0 \parallel H(\tau), dk_1 \parallel H(\tau))$ as target messages and receives ψ as in GAME.PKE. It then sends ψ to A_{T}.
3. For each decryption query (ψ_i, τ_i) made by 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 GAME.PKE. Given $dk_i \parallel \tau_i'$ from the decryption oracle, if $\tau'_i \neq \tau_i$, return $\perp$ to A_{T}. Otherwise, return dk_i.
4. When A_{T} outputs $\tilde{\delta}$, A_{E} outputs $\tilde{b} = \tilde{\delta}$.

Observe that all the decryption queries (ψ_i, τ_i) with $\psi_i \neq \psi$ made by A_{T} are perfectly answered by A_{E} since ψ_i is correctly decrypted by the decryption oracle of GAME.PKE.

Let Col denote an event that $H(\tau) = H(\tau_i)$ for some i. If Col never happens, then $\perp$ is the correct answer for all decryption queries (ψ, τ_i) with $\tau_i \neq \tau$ because $H(\tau) \neq H(\tau_i)$. Observe that when $b = 1$ in GAME.PKE the view of A_{T} in the simulated GAME.TKEM is the same as that in GAME.TKEM with $\delta = 1$ since dk_1 is correctly embedded in ψ. So we have $\Pr[\mathsf{Exp}^{(1)}_{\mathrm{tkem},A_{\mathrm{T}}} | \neg\mathsf{Col}] = \Pr[\mathsf{Exp}^{(1)}_{\mathrm{pke},A_{\mathrm{E}}} \neg|\mathsf{Col}]$. Similarly, when $b = 0$ in GAME.PKE the view of A_{T} is the same as that in GAME.TKEM with $\delta = 0$ since dk_1 given to A_{T} is independent of ψ. So we have $\Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem},A_{\mathrm{T}}}|\neg\mathsf{Col}] = \Pr[\mathrm{Exp}^{(1)}_{\mathrm{pke},A_{\mathrm{E}}}\neg|\mathsf{Col}]$. The following lemma [18] is then applied to the above equalities.

Lemma 1. *If event $X \wedge \neg Z$ occurs if and only if $Y \wedge \neg Z$ occurs, then* $|\Pr[X]-\Pr[Y]| \le \Pr[Z]$.

We thus have

$$\left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_{\mathrm{T}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem},A_{\mathrm{T}}}]\right| \le \left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{pke},A_{\mathrm{E}}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{pke},A_{\mathrm{E}}}]\right| + 2\Pr[\mathsf{Col}]. \tag{13}$$

Next we consider the case of Col. The adversary finds a collision $H(\tau) = H(\tau_i)$ where τ and τ_i are both chosen by the adversary after seeing hash function H defined as a part of the public-key. It therefore corresponds to finding a collision in GAME.CR. Accordingly, Pr[Col] is upper bound by

$$\Pr[\mathsf{Col}] \le \epsilon_{\mathrm{cr}}. \tag{14}$$

From (13) and (14),

$$\epsilon_{\mathrm{tkem},A_{\mathrm{T}}} \le \epsilon_{\mathrm{pke}} + 2\epsilon_{\mathrm{cr}}. \tag{15}$$

Appendix B. Proof of Theorem 3

The following is the CCA attack against the Tag-KEM shown in Sect. 4.3. Decryption oracle $\mathcal{O}$ is $\mathsf{TKEM.Dec}_{sk}(\cdot,\cdot)$.

```
[GAME.TKEM: $\delta \in \{0,1\}$]

Step 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$, $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$,
        $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_1)$, $dk_0 \leftarrow \mathcal{K}_D$.
Step 2. $(\tau, \rho) \leftarrow A_{\mathrm{T}}{}^{\mathcal{O}}(pk, dk_\delta)$.
Step 3. $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$.
Step 4. $\tilde{\delta} \leftarrow A_{\mathrm{T}}{}^{\mathcal{O}}(\rho, (\phi, \sigma))$.
```

In Step 4, A_{T} is not allowed to send $((\phi,\sigma),\tau)$ to the decryption oracle.

The outline of our proof is the same as that of [18]. Defining a series of games, GAME.0, ...,GAME.5 by modifying GAME.TKEM and examining fluctuation of probability. Let X_i denote the event that $\tilde{\delta} = \delta$ in GAME.i. In each game, (dk_1, mk) will be generated in Step 1 in a different way.

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

GAME.0: This game is the same as GAME.TKEM with $\delta = 0$, where dk_0 is given to A_{T} in Step 2. Since it is just a notational change, we have

$$\Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_{\mathrm{T}}}] = \Pr[X_0]. \tag{16}$$

GAME.1: Replace K_1 with random $K_0 \leftarrow \mathcal{K}_D$ in Step 1. Then apply the following rule to the decryption oracle: For every query $(\phi_i, \sigma_i, \tau_i)$ such that $\phi_i = \phi$, the oracle uses mk created in Step 1 to verify MAC (σ_i, τ_i) and returns dk_1 created also in Step 1 if the MAC is valid. (It returns $\perp$ if the MAC is invalid.) It then holds that

$$|\Pr[X_0] - \Pr[X_1]| \leq \epsilon_{\text{lkem}}. \tag{17}$$

GAME.2: Generate dk_1 and mk independently at random as $dk_1 \leftarrow \mathcal{K}_D$, $mk \leftarrow \mathcal{K}_M$. It then holds that

$$|\Pr[X_1] - \Pr[X_2]| \leq \epsilon_{\text{kdf}}. \tag{18}$$

GAME.3: Replace dk_0 with dk_1 in Step 2. We claim that the view of the adversary differs only if the adversary is successful in forging MAC. Hence it holds that

$$|\Pr[X_2] - \Pr[X_3]| \leq q_D\, \epsilon_{\text{mac}}. \tag{19}$$

GAME.4: Generate dk_1 and mk together by $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_0)$ from random $K_0 \leftarrow \mathcal{K}_D$. It then holds that

$$|\Pr[X_3] - \Pr[X_4]| \leq \epsilon_{\text{kdf}}. \tag{20}$$

GAME.5: Replace K_0 with K_1 and remove the rule applied to the decryption oracle in GAME.1. It then holds that

$$|\Pr[X_4] - \Pr[X_5]| \leq \epsilon_{\text{lkem}}. \tag{21}$$

Observe that GAME.5 turns out to be GAME.TKEM with $\delta = 1$. Hence

$$\Pr[X_5] = \Pr[\text{Exp}^{(1)}_{\text{tkem}, A_\text{T}}]. \tag{22}$$

Conclusion: From (16–22), we have

$$|\Pr[\text{Exp}^{(0)}_{\text{tkem}, A_\text{T}}] - \Pr[\text{Exp}^{(1)}_{\text{tkem}, A_\text{T}}]| = |\Pr[X_0] - \Pr[X_5]| \leq 2\epsilon_{\text{lkem}} + q_D\, \epsilon_{\text{mac}} + 2\epsilon_{\text{kdf}}$$

as stated in the theorem.

In the following we show proofs for (17), (18), and (19). The proofs for (20) and (21) are obtained from those for (18) and (17), respectively, with trivial modifications.

Proof of (17) We construct adversary A_L in GAME.LKEM by using A_T as a black-box as follows.

- (Setup) Given (pk, ϕ, K_b) as input, A_L computes $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_b)$. It also selects $dk_0 \leftarrow \mathcal{K}_D$. Run A_T by sending (pk, dk_0).
- (Oracle Queries) For every query $((\phi_j, \sigma_j), \tau_j)$ from A_T, if $\phi_j \neq \phi$, A_L forwards $(\phi_j, (\sigma_j, \tau_j))$ to oracle $\mathcal{VD}$. Given $K_j \in \mathcal{K}_K$ from $\mathcal{VD}$, A_L computes $(dk_j, *) \leftarrow \mathsf{KDF}_2(K_j)$ and returns dk_j to A_T. If $K_j = \perp$, A_L returns $\perp$ to A_T. For every query with $\phi_j = \phi$, A_L first verifies MAC σ_j with mk. If it is valid, A_L returns dk_1 to A_T. Otherwise, A_L returns $\perp$.

– (Challenge) Given τ from A_{T}, A_{L} computes $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau)$ and returns (ϕ, σ) to A_{T}.
– (Output) When A_{T} outputs $\tilde{\delta}$, A_{L} outputs $\tilde{b} = \tilde{\delta}$.

In the above simulation, when $b = 1$, the view of A_{T} is the same as that in $\mathsf{GAME.0}$ since K_1 given to A_{L} is correctly related to ϕ. On the other hand, when $b = 0$, given K_0 is independent of ϕ and the view of A_{T} is the same as that in $\mathsf{GAME.1}$. Therefore, the probability that A_{L} outputs $\tilde{b} = 1$ is the same as that A_{T} outputs $\tilde{\delta} = 1$. $|\Pr[X_0] - \Pr[X_1]|$ is thus upper bound by ϵ_{lkem}.

Proof of (18) In game $\mathsf{GAME.1}$, dk_1 and mk are correctly generated by $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_0)$. On the other hand, these are generated at random in game $\mathsf{GAME.2}$. Distinguishing these two cases are exactly the same as distinguishing distribution D_1 and D_0 as defined in Sect. 2.6. It is however important to stress that dk_0 is independent of the rest of variables observed in game $\mathsf{GAME.1}$ (and $\mathsf{GAME.2}$). It is therefore straightforward to construct adversary A_{F} that plays $\mathsf{GAME.KDF}$ by using A_{T} as a black-box. $|\Pr[X_1] - \Pr[X_2]|$ is thus upper bound by ϵ_{kdf}.

Proof of (19) Let F_3 denote an event that A_{T} makes a query, $(\phi_j, \sigma_j, \tau_j)$, for which the decryption oracle returns dk_1. Observe that the view of A_{T} in $\mathsf{GAME.2}$ and $\mathsf{GAME.3}$ are identical unless F_3 happens and dk_1 is included in the view. Therefore $|\Pr[X_2] - \Pr[X_3]| \leq \Pr[F_3]$ holds due to Lemma 1.

To upper bind $\Pr[F_3]$, we construct adversary A_{M} that attacks MAC as in $\mathsf{GAME.MAC}$ by using A_{T} as follows. A_{M} chooses $J \leftarrow \{1, \ldots, q_D\}$. It then generates (pk, sk) and ϕ as specified in $\mathsf{GAME.3}$ and runs A_{T}. When A_{T} outputs τ, A_{M} outputs τ as a chosen message and obtains MAC σ. It then sends challenge ciphertext (ϕ, σ) to A_{T}. When A_{T} makes J-th query, A_{M} outputs (τ_J, σ_J) as a forgery and stops. For every other query, $(\phi_i, \sigma_i, \tau_i)$, such that $\phi_j \neq \phi$, A_{M} decrypts ϕ_j by using sk. If $\phi_j = \phi$, A_{M} simply returns $\perp$.

The simulation is perfect by the moment F_3 happens at the first time. If F_3 happens at index J at the first time, the output of A_{M} is a successful forgery (σ_J, τ_J) since the query must be different from (σ, τ). Since A_{M} correctly guesses J with probability $1/q_D$, the success probability of A_{M} is $\Pr[F_3]/q_D$, which is upper bound by ϵ_{mac} by definition. We thus have $|\Pr[X_2] - \Pr[X_3]| \leq \Pr[F_3] \leq q_D\,\epsilon_{\mathrm{mac}}$.

Appendix C. Kurosawa-Desmedt KEM

We define the Kurosawa-Desmedt KEM as follows. Key generation function $\mathsf{KEM.Gen}$ is as illustrated in Sect. 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 key K and ciphertext $\phi = (u_1, u_2)$ such that

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

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

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

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

We start by describing GAME.LKEM against the Kurosawa-Desmedt KEM.

1. Run 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
$$u_1^* = g_1^{r^*}, \qquad u_2^* = g_2^{r^*}, \qquad \alpha^* = H(u_1^*, u_2^*), \qquad K^* = c^{r^*} d^{r^* \alpha^*}.$$
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 $\mathcal{V}D$ arbitrarily. Let $K_j = \text{KEM.Dec}_{sk}(u_{1j}, u_{2j})$. Oracle $\mathcal{V}D$ returns $\perp$ if (K_j, η_j) does not satisfy $\mathcal{P}^{\text{mac}}$. 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 [23,29], 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 GAME.0 be the original game as shown above. We will define a sequence of games GAME.1, ... Let X_i be the event that $\tilde{\delta} = \delta$ in GAME.*i*.

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

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

GAME.3 is the same as GAME.2, except that $u_1^*, u_2^* \in G_q$ are chosen at random. By DDH assumption, $|\Pr[X_3] - \Pr[X_2]|$ is negligible.

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

Suppose that A_{L} makes an invalid query, (u_{1j}, u_{2j}), at step 2 of GAME.3. (Remember that $(u_{1j}, u_{2j}) = (u_1^*, u_2^*)$ is not allowed.) 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. The query is therefore rejected with overwhelming probability due to $\mathcal{P}^{\text{mac}}$. Consequently, $\Pr[X_3] - \Pr[X_2]$ is negligible.

GAME.5 is the same as GAME.4, except that K^* is chosen at random from G_q. In GAME.4, (u_1^*, u_2^*) is invalid with overwhelming probability. Hence K^* is uniformly distributed over G_q. Therefore $\Pr[X_5] - \Pr[X_4]$ is negligible. Also in game GAME.5, it is clear that $\Pr[X_5] = 1/2$ because the view of A_{L} is independent of δ. This completes the proof on LCCA security.

Appendix D. Details of Refined Fujisaki-Okamoto Conversion

The Scheme Let $\Pi_{\mathrm{E}} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a public-key encryption scheme. By $\mathcal{R}$, we denote a space of random coins used in 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.

A Tag-KEM obtained from a refined Fujisaki-Okamoto Conversion is as follows. TKEM.Gen is the same as PKE.Gen; Given 1^{λ}, it outputs key pair (pk, sk). Given pk, TKEM.Key outputs $(dk, (pk, K))$ such that $K \leftarrow \{0,1\}^{\lambda}$, $dk \leftarrow G(K)$. The pair (pk, K) is the state information. TKEM.Enc and TKEM.Dec are as follows.

```
Function: TKEM.Enc($(pk, K), \tau$)
    $r \leftarrow H(K \parallel \tau)$
    $\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K; r)$
    Output $\psi$
```

```
Function: TKEM.Dec$_{sk}(\psi, \tau)$
    $K \leftarrow \mathsf{PKE.Dec}_{sk}(\psi)$
    $r \leftarrow H(K \parallel \tau)$
    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. (Roughly, a PKE is one-way against chosen plaintext attack if it is infeasible to compute K from $\mathsf{PKE.Enc}_{pk}(K; r)$ except for negligible probability, say ϵ_{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 [22] for details.) Then the following holds.

Theorem 5. *The above refined Fujisaki-Okamoto Tag-KEM is CCA secure in the random oracle model. In particular,* $\epsilon_{\mathrm{tkem},A_{\mathrm{T}}} \leq 2 q_D \gamma + 2 q_G \epsilon_{\mathrm{ow}}$.

Proof. GAME.TKEM for the above Tag-KEM is as follows. By $\mathcal{O}$, we denote the decryption oracle $\mathsf{TKEM.Dec}_{sk}(\cdot, \cdot)$.

GAME.TKEM: $\delta \in \{0,1\}$

Step 1. $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^{\lambda})$, $K_1 \leftarrow \{0,1\}^{\lambda}$, $dk_1 = G(K_1)$, $dk_0 \leftarrow \mathcal{K}_D$.
Step 2. $(\tau, \rho) \leftarrow A_{\mathrm{T}}{}^{\mathcal{O},H,G}(pk, dk_{\delta})$.
Step 3. $\psi \leftarrow \mathsf{PKE.Enc}_{pk}(K_1; H(K_1 \parallel \tau))$.
Step 4. $\tilde{\delta} \leftarrow A_{\mathrm{T}}{}^{\mathcal{O},H,G}(\rho, \psi)$.

We define games, GAME.0, GAME.1 and GAME.2 as shown in the sequel. In this proof, we consider the uniform choice of δ as a part of the game and define X_i as an event that $\tilde{\delta} = \delta$ in GAME.i.

In each game, 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 $K_j \parallel \tau_j$, oracle H outputs random r_j. Since K_j, τ_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.

GAME.0: The same as GAME.TKEM. Recall that, in the original definition of GAME.TKEM in Sect. 2.1, the choice of δ is not included in the probability space for events $\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_\mathrm{T}}$ and $\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_\mathrm{T}}$ while it is included for event X_0. By simple calculation, however, we have

$$\left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_\mathrm{T}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem},A_\mathrm{T}}]\right| = 2\left|\Pr[X_0] - \frac{1}{2}\right|. \tag{23}$$

GAME.1: For every decryption query (ψ_j, τ_j), if the table of H does not contain any entry $(*, \tau_j, *, \psi_j)$ the decryption oracle returns $\perp$.

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

$$|\Pr[X_0] - \Pr[X_1]| \le q_D\,\gamma. \tag{24}$$

GAME.2: If A_T sends K_1 to oracle G, abort the game.

Let AskG denote the event that A_T sends K_1 to G. Observe that GAME.2 is identical to GAME.1 unless AskG happens. Hence we have

$$|\Pr[X_1] - \Pr[X_2]| \le \Pr[\mathsf{AskG}].$$

To upper bound Pr[AskG], we construct an adversary, say A_ow, that breaks the one-way property of the underlying PKE by using A_T as follows.

Given ψ and pk, A_ow first flips coin $J \leftarrow \{1, \dots, q_G\}$ and simulates GAME.2 in the following way.

- Use given pk and ψ in Step 1 and 4, respectively.
- 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-th query K_J to G, output K_J and halt.
- For every decryption query (ψ_j, τ_j), if the table of H contains 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 $\perp$.

If J-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_ow perfectly simulates GAME.2 up to the moment J-th query to G is made. And once event AskG happens at J-th query, the output of A_ow is PKE.Dec(sk, ψ).

Running time of A_ow is almost the same as that of A_T plus computing time of encryption function q_H times. Now we have

$$|\Pr[X_1] - \Pr[X_2]| \le \Pr[\mathsf{AskG}] \le q_G\,\epsilon_\mathrm{ow}. \tag{25}$$

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 $\Pr[X_2] = \frac{1}{2}$. From (23), (24), and (25), we have

$$\left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem},A_\mathrm{T}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{tkem},A_\mathrm{T}}]\right| = 2\left|\Pr[X_0] - \frac{1}{2}\right| \leq 2q_D\,\gamma + 2q_G\,\epsilon_{\mathrm{ow}}. \qquad \square$$

Appendix E. Tag-KEM from RCCA-secure KEM

RCCA security for KEM is defined in the same way as that for PKE. We modify GAME.KEM in Sect. 2.4 in such a way that decryption oracle $\mathcal{O}$ returns 'test' when the result of decryption is in $\{K_1, K_0\}$.[2] Call this modified game GAME.RKEM. Let $\mathrm{Exp}^{(b)}_{\mathrm{rkem},A_\mathrm{R}}$ denote the event that adversary A_R outputs 1 in GAME.RKEM with b. Define $\epsilon_{\mathrm{rkem},A_\mathrm{R}}$ as

$$\epsilon_{\mathrm{rkem},A_\mathrm{R}} = \left|\Pr[\mathrm{Exp}^{(0)}_{\mathrm{rkem},A_\mathrm{R}}] - \Pr[\mathrm{Exp}^{(1)}_{\mathrm{rkem},A_\mathrm{R}}]\right|. \tag{26}$$

A KEM is RCCA secure if there exists a negligible function ϵ_{rkem} in λ such that, for sufficiently large λ, $\epsilon_{\mathrm{rkem},A_\mathrm{R}} \leq \epsilon_{\mathrm{rkem}}$ for all ppt algorithm A_R.

An RCCA-secure KEM can be constructed from an RCCA-secure PKE: just encrypt dk by the PKE. An RCCA-secure Tag-KEM can be constructed from an RCCA-secure KEM in the same way as in Sect. 4.2 except for that MAC.Sign and MAC.Ver take $\tau \parallel \phi$ instead of τ as an input. That is, TKEM.Gen is the same as KEM.Gen. 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 $\omega = (mk, \phi)$. TKEM.Enc and TKEM.Dec are as follows.

```
Function: TKEM.Enc($\omega$, $\tau$)
    $(mk, \phi) \leftarrow \omega$
    $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau \parallel \phi)$
    Output $\psi = (\phi, \sigma)$
```

```
Function: 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 $\mathsf{MAC.Ver}_{mk}(\sigma, \tau \parallel \phi) \neq 1$
    output $\perp$
    Otherwise, output $dk$
```

Theorem 6. *If a KEM is RCCA-secure, a MAC is one-time secure, and a KDF is secure, the above Tag-KEM is CCA-secure. In particular,* $\epsilon_{\mathrm{tkem},A_\mathrm{T}} \leq 2\epsilon_{\mathrm{rkem}} + q_D\,\epsilon_{\mathrm{mac}} + 2\epsilon_{\mathrm{kdf}} + 2q_D\,/\,|\mathcal{K}_K|$.

Proof. The CCA attack game to the above Tag-KEM is as follows.

[2] Note that, when $\delta = 1$, K_0 is independent from the view of the adversary. Nevertheless, the decryption oracle returns 'test' if the decryption coincidentally yields K_0. Such a treatment is necessary to handle the case where the size of $\mathcal{K}_K$ is very limited.

```
Step 1. 1. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\lambda)$.
        2. $(K_1, \phi) \leftarrow \mathsf{KEM.Enc}_{pk}()$, $K_0 \leftarrow \mathcal{K}_K$.
        3. $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_1)$.
        4. $dk_0 \leftarrow \mathcal{K}_D$.
Step 2. $(\tau, \rho) \leftarrow A_{\mathrm{T}}{}^{\mathcal{O}}(pk, dk_\delta)$.
Step 3. $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau \parallel \phi)$.
Step 4. $\tilde{\delta} \leftarrow A_{\mathrm{T}}{}^{\mathcal{O}}(\rho, (\phi, \sigma))$.
```

Note that K_0 in Step 1.2 is included only for later use and unused in this original GAME.TKEM.

For each query $((\phi_j, \sigma_j), \tau_j)$ decryption oracle $\mathcal{O}$ works as follows.

[Decryption Oracle]

```
O-1. $K_j = \mathsf{KEM.Dec}_{sk}(\phi_j)$.
O-2. $(dk_j, mk_j) \leftarrow \mathsf{KDF}_2(K_j)$.
O-3. If $\mathsf{MAC.Ver}_{mk_j}(\sigma_j, \tau_j \parallel \phi_j) = 1$, return $dk_j$. Return $\perp$, otherwise.
```

The outline of the proof is similar to that for Theorem 3 in Appendix B. We define a series of games, GAME.0, ..., GAME.7 by gradually modifying the above GAME.TKEM. Let X_i denote the event that $\tilde{\delta} = \delta$ in GAME.*i*.

GAME.0: Fix dk_δ to dk_0 in Step 2. We apparently have

$$\Pr[\mathrm{Exp}^{(0)}_{\mathrm{tkem}, A_{\mathrm{T}}}] = \Pr[X_0]. \tag{27}$$

GAME.1: Apply the following rule to the decryption oracle.

Rule.1 If $K_j \in \{K_1, K_0\}$ in Step O-1, then in Step O-3, use dk_1 and mk defined in Step 1.3. instead of dk_j and mk_j, respectively.

We claim that

$$|\Pr[X_0] - \Pr[X_1]| \le q_D \,/\, |\mathcal{K}_K|. \tag{28}$$

GAME.2: Replace K_1 with K_0 in Step 1.3. We claim that

$$|\Pr[X_1] - \Pr[X_2]| \le \epsilon_{\mathrm{rkem}}. \tag{29}$$

GAME.3: Generate dk_1 and mk in Step 1.3 randomly and independently by $dk_1 \leftarrow \mathcal{K}_D$ and $mk \leftarrow \mathcal{K}_M$. Also replace Rule.1 with the following Rule.2.

Rule.2 If $K_j = K_1$ in Step O-1, then in Step O-3, use dk_1 and mk defined in Step 1.3 instead of dk_j and mk_j, respectively.

Applying Rule.2 completely removes K_0 from the view of the adversary in GAME.3. It then holds that

$$|\Pr[X_2] - \Pr[X_3]| \le \epsilon_{\mathrm{kdf}}. \tag{30}$$

GAME.4: Replace dk_0 in Step 2 with dk_1. We claim that

$$|\Pr[X_3] - \Pr[X_4]| \le q_D\, \epsilon_{\mathrm{mac}}. \tag{31}$$

GAME.5: In Step 1.3 Generate dk_1 and mk by $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_0)$ in Step 1.3. Also replace Rule.2 with Rule.1. Just as well as (30), we have

$$|\Pr[X_4] - \Pr[X_5]| \leq \epsilon_{\text{kdf}}. \tag{32}$$

GAME.6: Replace K_0 with K_1 in Step 1.3. As well as (29), we have

$$|\Pr[X_5] - \Pr[X_6]| \leq \epsilon_{\text{rkem}}. \tag{33}$$

GAME.7: Remove Rule.1 from the decryption oracle. As well as (28), we claim that

$$|\Pr[X_6] - \Pr[X_7]| \leq q_D \,/\, |\mathcal{K}_K|. \tag{34}$$

Finally, by inspection, one can see that GAME.7 is the same as GAME.TKEM with $\delta = 1$. Hence

$$\Pr[X_7] = \Pr[\text{Exp}^{(1)}_{\text{tkem},A_\text{T}}]. \tag{35}$$

Conclusion: By summing from (27) to (35), we have

$$\begin{aligned} |\Pr[\text{Exp}^{(0)}_{\text{tkem},A_\text{T}}] - \Pr[\text{Exp}^{(1)}_{\text{tkem},A_\text{T}}]| &= |\Pr[X_0] - \Pr[X_7]| \\ &\leq 2\epsilon_{\text{rkem}} + q_D\,\epsilon_{\text{mac}} + 2\epsilon_{\text{kdf}} + 2q_D \,/\, |\mathcal{K}_K| \end{aligned}$$

as in the theorem.

Proof of (28) There are two events that happen only in GAME.1.

– There is a query accepted in GAME.0 but rejected in GAME.1.
– There is a query rejected in GAME.0 but accepted in GAME.1.

Let E_1 denote the event that any of the above events happen. Unless E_1 takes place, the view of the adversary is identical in GAME.0 and GAME.1. Hence,

$$|\Pr[X_0] - \Pr[X_1]| \leq \Pr[E_1]. \tag{36}$$

Observe that, if $K_j = K_1$, then $mk_j = mk$ and none of the events in E_1 can occur. If E_1 happens, then $K_j = K_0$, which occurs with probability at most $1/|\mathcal{K}_K|$ for each query since K_0 is chosen randomly and it is independent from the view of the adversary. We thus have

$$\Pr[E_1] \leq q_D \,/\, |\mathcal{K}_K|. \tag{37}$$

Proof of (29) The claim is proven by constructing A_R that attacks the underlying RCCA-secure KEM by using A_T as a black-box. The construction is straightforward. We however stress that, when the decryption oracle of the RCCA game returns 'test' in response to a query from A_R, the corresponding query from A_T can be perfectly answered simply by following Rule.1.

Just for completeness, we give the details in the following. Adversary A_R works as follows.

– (Setup) Given (pk, ϕ, K_b) as input, A_{R} computes $(dk_1, mk) \leftarrow \mathsf{KDF}_2(K_b)$. It also selects $dk_0 \leftarrow \mathcal{K}_D$. Run A_{T} by sending (pk, dk_0).
– (Oracle Simulation) For every query $((\phi_j, \sigma_j), \tau_j)$ from A_{T}, A_{R} forwards ϕ_j to $\mathcal{O}$. If 'test' is returned, A_{R} executes Step O-3 with (dk_1, mk). Otherwise, if K_j is returned, A_{R} executes Step O-2 and O-3 as specified.
– (Challenge) Given τ from A_{T}, A_{R} computes $\sigma \leftarrow \mathsf{MAC.Sign}_{mk}(\tau \parallel \phi)$ and returns (ϕ, σ) to A_{T}.
– (Output) When A_{T} outputs $\tilde{\delta}$, A_{R} outputs $\tilde{b} = \tilde{\delta}$.

Suppose that $b = 1$. Then dk_1 and mk are generated as well as those in $\mathsf{GAME.1}$. When 'test' is returned during the simulation of the decryption oracle, the use of these dk_1 and mk just follows Rule.1 and yields the same view as in $\mathsf{GAME.1}$. Hence $\Pr[X_1] = \Pr[\mathrm{Exp}^{(1)}_{\mathrm{rkem}, A_{\mathrm{R}}}]$. On the other hand, if $b = 0$, dk_1 and mk are generated as well as those in $\mathsf{GAME.2}$. Even in this case, using these dk_1 and mk in the case of 'test' follows Rule.1 and yields the same view as in $\mathsf{GAME.2}$. Hence $\Pr[X_2] = \Pr[\mathrm{Exp}^{(0)}_{\mathrm{rkem}, A_{\mathrm{R}}}]$. Accordingly,

$$|\Pr[X_1] - \Pr[X_2]| = |\Pr[\mathrm{Exp}^{(1)}_{\mathrm{rkem}, A_{\mathrm{R}}}] - \Pr[\mathrm{Exp}^{(0)}_{\mathrm{rkem}, A_{\mathrm{R}}}]| \leq \epsilon_{\mathrm{rkem}} \tag{38}$$

as claimed.

Proof of (30) We construct A_{F} that attacks KDF_2 as in $\mathsf{GAME.KDF}$ in Sect. 2.6 by using A_{T} as a black-box as follows. Given (dk^*, mk^*) as input, A_{F} simulates $\mathsf{GAME.2}$ as specified except that (dk^*, mk^*) is assigned to (dk_1, mk) in Step 1.3. To simulate the decryption oracle, A_{F} uses the secret key sk generated by itself. If Step O-1 yields $K_j = K_1$, A_{F} uses (dk_1, mk) in Step O-3. A_{F} never generates K_0 and leaves it undefined in the simulation.

Consider the case where given (dk^*, mk^*) belongs to $\mathcal{D}_1$. In this case, we expect that the view of A_{T} is the same as that in $\mathsf{GAME.2}$. Let K_0 denote the key that generates (dk^*, mk^*), i.e., $(dk^*, mk^*) \leftarrow \mathsf{KDF}_2(K_0)$. Although such K_0 is unknown to A_{F} and the simulation does not care for K_0 at all, it perfectly simulates the case of $K_j = K_0$ as in Rule.1 because, if $K_j = K_0$, then $(dk_j, mk_j) = (dk^*, mk^*) = (dk_1, mk)$ and A_{F} actually use such (dk_1, mk) in the simulation. Accordingly, $\Pr[X_2] = \Pr[\mathrm{Exp}^{(\mathcal{D}_1)}_{\mathrm{kdf}, A_{\mathrm{F}}}]$.

Next consider the case where (dk^*, mk^*) belongs to $\mathcal{D}_0$. In this case, (dk_1, mk) are random and independent as in $\mathsf{GAME.3}$. For such (dk^*, mk^*), we do not define corresponding K_0 and it is the case as specified in Rule.2. Hence the view of A_{T} in the simulation is identical to that in $\mathsf{GAME.3}$. Accordingly, $\Pr[X_3] = \Pr[\mathrm{Exp}^{(\mathcal{D}_0)}_{\mathrm{kdf}, A_{\mathrm{F}}}]$.

In summary, we have

$$|\Pr[X_2] - \Pr[X_3]| = |\Pr[\mathrm{Exp}^{(\mathcal{D}_1)}_{\mathrm{kdf}, A_{\mathrm{F}}}] - \Pr[\mathrm{Exp}^{(\mathcal{D}_0)}_{\mathrm{kdf}, A_{\mathrm{F}}}]| \leq \epsilon_{\mathrm{kdf}} \tag{39}$$

as claimed.

Proof of (31) Since both dk_0 in $\mathsf{GAME.3}$ and dk_1 in $\mathsf{GAME.4}$ are taken randomly from $\mathcal{K}_D$, the view of the adversary at the beginning of Step 2 is unchanged. Let E_4 denote the event that the decryption oracle returns dk_1 as a result of following Rule.2. If E_4 happens, the adversary in $\mathsf{GAME.4}$ obtains the same key as given in Step 2 while it is just a random and independent key in GAME.3. Unless E_4 happens, the view of the adversary is unchanged. We thus have

$$|\Pr[X_3] - \Pr[X_4]| \leq \Pr[E_4]. \tag{40}$$

Since mk is generated independently from any other variables in both GAME.3 and GAME.4, and used only once for creating σ in the challenge ciphertext, event E_4 implies that the adversary is successful in forging MAC with respect to mk. One can then show $\Pr[E_4] \leq q_D\, \epsilon_{\mathrm{mac}}$ in the similar way as for the proof of (19) in Appendix B with obvious modifications. □

References

[1] M. Abe, Robust distributed multiplication without interaction, in *Advances in Cryptology—CRYPTO'99*, ed. by M. Wiener. Lecture Notes in Computer Science, vol. 1666 (Springer, Berlin, 1999), pp. 130–147

[2] M. Abe, 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

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

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

[5] M. Ben-Or, S. Goldwasser, A. Wigderson, Completeness theorems for non-cryptographic fault-tolerant distributed computation, in *Proceedings of the* 20*th Annual ACM Symposium on the Theory of Computing*, pp. 1–10, 1988

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

[7] D. Bleichenbacher, Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1, in *Advances in Cryptology—CRYPTO'98*, ed. by H. Krawczyk. Lecture Notes in Computer Science, vol. 1462 (Springer, Berlin, 1998), pp. 1–12

[8] D. Boneh, Simplified OAEP for the RSA and Rabin functions, in *Advances in Cryptology—CRYPTO 2001*, ed. by J. Killian. Lecture Notes in Computer Science, vol. 2139 (Springer, Berlin, 2001), pp. 275–291

[9] D. Boneh, X. Boyen, Efficient selective-ID secure identity based encryption, in *Advances in Cryptology—EUROCRYPT 2004*. Lecture Notes in Computer Science, vol. 3027 (Springer, Berlin, 2004), pp. 223–238

[10] D. Boneh, J. Katz, Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. Technical Report 2004/261, IACR ePrint archive, 2004

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

[12] D. Boneh, X. Boyen, S. Halevi, Chosen ciphertext secure public key threshold encryption without random oracles, in *Topics in Cryptology—CT-RSA 2006*, ed. by T. Rabin, S. Halevi. Lecture Notes in Computer Science, vol. 3860 (Springer, Berlin, 2006), pp. 226–243

[13] R. Canetti, S. Goldwasser, An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack, in *Advances in Cryptology—EUROCRYPT'99*, ed. by J. Stern. Lecture Notes in Computer Science, vol. 1592 (Springer, Berlin, 1999), pp. 90–106

[14] R. Canetti, H. Krawczyk, J. Nielsen, Relaxing chosen-ciphertext security, in *Advances in Cryptology—CRYPTO 2003*, ed. by D. Boneh. Lecture Notes in Computer Science, vol. 2729 (Springer, Berlin, 2003), pp. 565–582. Also available at IACR ePrint archive 2003/174

[15] R. Canetti, S. Halevi, J. Katz, Chosen-ciphertext security from identity-based encryption, in *Advances in Cryptology—EUROCRYPT 2004*. Lecture Notes in Computer Science, vol. 3027 (Springer, Berlin, 2004), pp. 207–222

[16] R. Cramer, V. Shoup, A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack, in *Advances in Cryptology—CRYPTO'98*, ed. by H. Krawczyk. Lecture Notes in Computer Science, vol. 1462 (Springer, Berlin, 1998), pp. 13–25

[17] R. Cramer, V. Shoup, Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption, in *Advances in Cryptology—EUROCRYPTO 2002*, ed. by L. Knudsen. Lecture Notes in Computer Science, vol. 2332 (Springer, Berlin, 2002), pp. 45–64

[18] R. Cramer, V. Shoup, Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.* **33**(1), 167–226 (2003)

[19] A. Dent, A designer's guide to KEMs, in *9th IMA International Conference on Cryptography and Coding*, ed. by K.G. Paterson. Lecture Notes in Computer Science, vol. 2898 (Springer, Berlin, 2003), pp. 133–151

[20] Y.G. Desmedt, Y. Frankel, Threshold cryptosystems, in *Advances in Cryptology—CRYPTO'89*, ed. by G. Brassard. Lecture Notes in Computer Science, vol. 435 (Springer, Berlin, 1990), pp. 307–315

[21] D. Dolev, C. Dwork, M. Naor, Nonmalleable cryptography. *SIAM J. Comput.* **30**(2), 391–437 (2000)

[22] E. Fujisaki, T. Okamoto, Secure integration of asymmetric and symmetric encryption schemes, in *Advances in Cryptology—CRYPTO'99*, ed. by M. Wiener. Lecture Notes in Computer Science, vol. 1666 (Springer, Berlin, 1999), pp. 537–554

[23] R. Gennaro, V. Shoup, A note on an encryption scheme of Kurosawa and Desmedt. Technical Report 2004/194, IACR ePrint archive, 2004

[24] C. Gentry, How to compress Rabin ciphertexts and signatures (and more), in *Advances in Cryptology—CRYPTO 2004*, ed. by M. Franklin. Lecture Notes in Computer Science, vol. 3152 (Springer, Berlin, 2004), pp. 179–200

[25] O. Goldreich, S. Micali, A. Wigderson, How to play any mental game or a completeness theorem for protocols with honest majority, in *Proceedings of the* 19*th annual ACM Symposium on the Theory of Computing*, New York City, pp. 218–229, 1987

[26] J. Herranz, D. Hofheinz, E. Kiltz, The Kurosawa-Desmedt key encapsulation is not chosen-ciphertext secure. IACR e-print Archive 2006/207, 2005

[27] S. Jarecki, A. Lysyanskaya, Adaptively secure threshold cryptography: introducing concurrency, removing erasures (extended abstract), in *Advances in Cryptology—EUROCRYPT 2000*. Lecture Notes in Computer Science, vol. 1807 (Springer, Berlin, 2000), pp. 221–242

[28] E. Kiltz, Chosen-ciphertext security from tag-based encryption, in *Theory of Cryptography—TCC'06*, ed. by S. Halevi, T. Rabin. Lecture Notes in Computer Science, vol. 3876 (Springer, Berlin, 2006), pp. 581–600

[29] K. Kurosawa, Y. Desmedt, A new paradigm of hybrid encryption scheme, in *Advances in Cryptology—CRYPTO 2004*, ed. by M. Franklin. Lecture Notes in Computer Science, vol. 3152 (Springer, Berlin, 2004), pp. 426–442

[30] P. MacKenzie, M.K. Reiter, K. Yang, Alternatives to non-malleability: definitions, constructions, and applications, in *Theory of Cryptography—TCC'04*, ed. by M. Naor. Lecture Notes in Computer Science, vol. 2951 (Springer, Berlin, 2004), pp. 171–190

[31] W. Nagao, Y. Manabe, T. Okamoto, A universally composable secure channel based on the KEM-DEM framework, in *Theory of Cryptography—TCC'05*. Lecture Notes in Computer Science, vol. 3378 (Springer, Berlin, 2005), pp. 426–444

[32] M. Naor, M. Yung, Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the* 22*nd annual ACM Symposium on the Theory of Computing*, pp. 427–437, 1990

[33] T. Okamoto, D. Pointcheval, REACT: Rapid enhanced-security asymmetric cryptosystem transform, in *RSA'2001*. Lecture Notes in Computer Science (Springer, Berlin, 2001)

[34] C. Rackoff, D. Simon, Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack, in *Advances in Cryptology—CRYPTO'91*. Lecture Notes in Computer Science, vol. 576 (Springer, Berlin, 1992), pp. 433–444

[35] V. Shoup, Using hash functions as a hedge against chosen ciphertext attack, in *Advances in Cryptology—EUROCRYPT 2000*. Lecture Notes in Computer Science, vol. 1807 (Springer, Berlin, 2000), pp. 275–288

[36] V. Shoup, OAEP reconsidered, in *Advances in Cryptology—CRYPTO 2001*. Lecture Notes in Computer Science, vol. 2139 (Springer, Berlin, 2001), pp. 239–259

[37] V. Shoup, ISO 18033-2: An emerging standard for public-key encryption (committee draft). Available at http://shoup.net/iso/, June 3 2004

[38] V. Shoup, R. Gennaro, Securing threshold cryptosystems against chosen ciphertext attack. *J. Cryptol.* **15**(2), 75–96 (2002)