CCA Security and Trapdoor Functions via Key-Dependent-Message Security

Fuyuki Kitagawa[1], Takahiro Matsuda[2], and Keisuke Tanaka[3]

[1] NTT Secure Platform Laboratories, Tokyo, Japan
fuyuki.kitagawa.yh@hco.ntt.co.jp
[2] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
t-matsuda@aist.go.jp
[3] Tokyo Institute of Technology, Tokyo, Japan
keisuke@is.titech.ac.jp

Abstract. We study the relationship among public-key encryption (PKE) satisfying indistinguishability against chosen plaintext attacks (IND-CPA security), that against chosen ciphertext attacks (IND-CCA security), and trapdoor functions (TDF). Specifically, we aim at finding a unified approach and some additional requirement to realize IND-CCA secure PKE and TDF based on IND-CPA secure PKE, and show the following two main results.

As the first main result, we show how to achieve IND-CCA security via a weak form of key-dependent-message (KDM) security. More specifically, we construct an IND-CCA secure PKE scheme based on an IND-CPA secure PKE scheme and a secret-key encryption (SKE) scheme satisfying one-time KDM security with respect to projection functions (projection-KDM security). Projection functions are very simple functions with respect to which KDM security has been widely studied. Since the existence of projection-KDM secure PKE implies that of the above two building blocks, as a corollary of this result, we see that the existence of IND-CCA secure PKE is implied by that of projection-KDM secure PKE.

As the second main result, we extend the above construction of IND-CCA secure PKE into that of TDF by additionally requiring a mild requirement for each building block. Our TDF satisfies adaptive one-wayness. We can instantiate our TDF based on a wide variety of computational assumptions. Especially, we obtain the first TDF (with adaptive one-wayness) based on the sub-exponential hardness of the constant-noise learning-parity-with-noise (LPN) problem.

Keywords: chosen ciphertext security, trapdoor functions, key-dependent-message security

1 Introduction

1.1 Background

Public-key encryption (PKE) is one of the most fundamental cryptographic primitives. The most basic security requirement for PKE is indistinguishability against chosen plaintext attacks (IND-CPA security) [23]. However, in many practical applications, PKE schemes should satisfy the stronger notion of indistinguishability against chosen ciphertext attacks (IND-CCA security) [35,15] in order to take active adversaries into consideration [10].

Since IND-CCA security is stronger than IND-CPA security, the existence of IND-CCA secure PKE implies that of IND-CPA secure one. However, the implication of the opposite direction is not known. While a partial negative result was shown by Gertner, Malkin, and Myers [21], the question whether an IND-CCA secure PKE scheme can be constructed from an IND-CPA secure one has still been standing as a major open question in cryptography.

In addition to IND-CCA secure PKE, a family of trapdoor functions (TDF) is also a fundamental primitive whose relationship with IND-CPA secure PKE has been widely studied. It was shown that an IND-CPA secure PKE can be constructed from TDF [40,6]. For the opposite direction, Gertner, Malkin, and Reingold [22] showed a negative result stating that TDF cannot be built from PKE in a black-box way.

In fact, in the random oracle model [7], we can construct both IND-CCA secure PKE and TDF based solely on IND-CPA secure PKE using a simple and unified derandomization technique [6,19]. However, in the standard model, we cannot use such a simple derandomization technique successfully. Especially, in order to construct IND-CCA secure PKE and TDF in the standard model by circumventing the impossibility results [21,22], we need non-black-box techniques or some additional requirements for the building block PKE scheme.

Hajiabadi and Kapron [24] tackled the above question, and as a main result, they built a TDF based on a PKE scheme satisfying circular security [14] and a randomness re-usability property called reproducibility [5]. Since their TDF satisfies one-wayness under correlated products, based on the same assumption, they also obtained a construction of IND-CCA secure PKE by relying on the result by Rosen and Segev [38]. Their TDF construction is elegant and can also be extended to deterministic encryption [4]. However, due to the somewhat strong additional requirement of randomness re-usability, its instantiations are limited to specific number theoretic assumptions.

In this work, we further study the above question. Especially, we aim at finding a unified approach and some additional requirement to realize IND-CCA secure PKE and TDF based on IND-CPA secure PKE.

1.2 Our Results

We show a unified approach to build IND-CCA secure PKE and TDF based on IND-CPA secure PKE by additionally using secret-key encryption (SKE) satisfying a weak form of *key-dependent-message (KDM) security* [9]. Roughly speaking, an encryption scheme is said to be KDM secure if it can securely encrypt a message of the form $f(\mathsf{sk})$, where sk is the secret key and f is a function. The details of our results are as follows.

IND-CCA Security via Key-Dependent-Message Security. As the first main result, we construct an IND-CCA secure PKE scheme based on an IND-CPA secure

PKE scheme and an SKE scheme satisfying KDM security. The building block SKE scheme is required to be one-time KDM secure with respect to projection functions (projection-KDM secure). Projection functions are very simple functions such that each output bit depends on at most a single bit of an input. An SKE scheme satisfying one-time projection-KDM security can be built from a wide variety of computational assumptions [11,3,12,13,16]. We obtain this result based on a construction technique used by Koppula and Waters [30] who showed how to construct IND-CCA secure attribute-based encryption (ABE) from IND-CPA secure one using a pseudorandom generator (PRG) with a special security property called hinting PRG. We extend the techniques of Koppula and Waters in several aspects. See Section 2 for the details.

The existence of PKE satisfying projection-KDM security against chosen plaintext attacks implies that of the above two building blocks. Therefore, as a corollary of this result, we see that the existence of IND-CCA secure PKE is implied by that of PKE with projection-KDM security (against CPA!).

Given our result and the result by Koppula and Waters, it is natural to ask what is the relationship between hinting PRG and one-time KDM secure SKE. To clarify this, we show that a one-time projection-KDM secure SKE scheme can be built from a hinting PRG. This means that one-time projection-KDM secure SKE is not a stronger assumption than hinting PRG.

Previously, Matsuda and Hanaoka [33] constructed an IND-CCA secure PKE scheme from a PKE scheme satisfying the sender non-committing property and an SKE scheme satisfying one-time KDM security with respect to circuits of a-priori bounded size. We improve their result in the sense that our construction requires weaker security properties for both of the underlying PKE and SKE schemes compared to theirs.

On Black-Box Usage of Building Blocks. Our construction of an IND-CCA secure PKE scheme is *fully-black-box* [36] and *non-shielding* [21]. A construction of a PKE scheme is said to be shielding if the decryption algorithm of the scheme does *not* call the encryption algorithm of the building block schemes, and otherwise it is called non-shielding. We show that our construction being a non-shielding construction is essential by showing that a fully-black-box and shielding construction of an IND-CCA secure PKE scheme based on our assumptions is *impossible* by extending the impossibility result shown by Gertner et al. [21]. More specifically, we show that there is no fully-black-box and shielding construction of an IND-CCA secure PKE scheme based on a projection-KDM secure PKE scheme that trivially implies both of our building blocks.

Extension to TDF. As the second main result, we extend the above construction of an IND-CCA secure PKE scheme into that of a TDF by additionally requiring a mild requirement for each building block. Our TDF satisfies adaptive one-wayness [27]. Adaptive one-wayness ensures that an adversary cannot invert a function in the family even under the existence of the inversion oracle, and thus it is a much stronger security property compared to ordinary one-wayness.

The additional requirements for the building blocks are as follows.

– First, we require that the underlying IND-CPA secure PKE scheme have the *pseudorandom ciphertext property*. Namely, a ciphertext of the underlying IND-CPA secure PKE scheme needs to be indistinguishable from a uniformly random element sampled from the ciphertext space of the scheme.
– Second, we require that the underlying projection-KDM secure SKE scheme be *randomness-recoverable*. Namely, random coins used to encrypt a message needs to be recovered together with the message in the decryption process.

Both of the above two requirements are mild in the following sense.

For the first requirement, a number of IND-CPA secure PKE schemes based on concrete computational assumptions naturally have this property. In fact, as far as we know, an IND-CPA secure PKE scheme satisfying the pseudorandom ciphertext property can be constructed from any concrete computational assumption implying IND-CPA secure PKE.

For the second requirement, the randomness-recovering property is easy to achieve in the secret-key setting while this property is so hard to achieve in the public-key setting that it immediately yields a TDF. Projection-KDM secure PKE schemes based on projective hash functions [11,12,39] can easily be transformed into SKE variants satisfying the randomness-recovering property. Also, projection-KDM secure SKE schemes based on the learning-parity-with-noise (LPN) and learning-with-errors (LWE) assumptions proposed by Applebaum, Cash, Peikert, and Sahai [3] already satisfy this property. Moreover, even the recent constructions of KDM secure PKE schemes based on the computational Diffie-Hellman (CDH) and factoring assumptions [13,16] can be transformed into one-time projection-KDM secure SKE with the randomness-recovering property.

As noted above, the additional requirements needed to realize a TDF are mild. As a result, we can instantiate our TDF based on a wide variety of computational assumptions. Especially, by combining the previous results [42,3], we obtain the first TDF (with adaptive one-wayness) based on the sub-exponential hardness of the constant-noise LPN problem. Moreover, we also obtain the first TDF satisfying adaptive one-wayness based on the low-noise LPN assumption. Previously to our work, a TDF satisfying ordinary one-wayness based on the low-noise LPN assumption was shown by Kiltz, Masny, and Pietrzak [26].

1.3 Concurrent and Subsequent Works

Very recently, in a concurrent work, Lombardi, Quach, Rothblum, Wichs, and Wu [31] showed how to construct a reusable designated-verifier non-interactive zero-knowledge (DV-NIZK) argument system based on the combination of an IND-CPA secure PKE scheme and a hinting PRG. In one of the steps in their construction, they employed the construction methodology of Koppula and Waters [30], and a hinting PRG is used in the step.

Based on our technique in this paper, Lombardi et al. [32] (in their latest update on May 23, 2019) and Kitagawa and Matsuda [28] independently and concurrently observe that a hinting PRG used in Lombardi et al.'s reusable DV-NIZK argument system can also be replaced with a one-time $\mathcal{P}$-KDM secure

SKE in exactly the same way as we do in our work. That is, these works show that a reusable DV-NIZK argument system can be constructed from an IND-CPA secure PKE scheme and a one-time $\mathcal{P}$-KDM secure SKE scheme. This leads to the first reusable DV-NIZK argument system based on the LPN assumption.

Furthermore, Kitagawa and Matsuda [28] show that using the reusable DV-NIZK argument system above and our result on IND-CCA secure PKE, we can transform a KDM-CPA secure PKE scheme into a KDM-CCA secure one without requiring any additional assumption. This leads to the first KDM-CCA secure PKE schemes based on the CDH and LPN assumptions.

1.4 Paper Organization

In Section 2, we show an overview of our techniques. In Section 3, we review definitions of cryptographic primitives. In Section 4, we show our proposed IND-CCA secure KEM. In Section 5, we prove the impossibility of fully-black-box shielding constructions. Finally, in Section 6, we present our proposed TDF.

Many of the details are omitted due to the space limitation. See the full version [29] for all the details.

2 Technical Overview

We give an overview of our techniques.

2.1 Achieving IND-CCA Security via Randomness-Recovering

One of classical mechanisms for achieving IND-CCA security is adopting a validity checking by re-encryption in the decryption process. In this technique, we make an encryption scheme randomness-recoverable, that is, a randomness used to generate a ciphertext is recovered during the decryption process. Then, when decrypting the ciphertext, we can check that the ciphertext was well-formed by re-encrypting the decrypted message using the recovered randomness.

Such a mechanism can be easily implemented in the random oracle model. Fujisaki and Okamoto [19] showed that by designing the encryption algorithm as $\mathsf{Enc}(\mathsf{pk}, \mathsf{r} \| \mathsf{m}; \mathsf{H}(\mathsf{r} \| \mathsf{m}))$, we can construct an IND-CCA secure PKE scheme based on the above strategy, where $\mathsf{Enc}(\mathsf{pk}, \cdot; \cdot)$ is the encryption algorithm of an IND-CPA secure PKE scheme and H is a hash function modeled as a random oracle. On the other hand, in the standard model, realizing a randomness-recoverable encryption scheme is difficult. Almost all existing such schemes are based on a TDF with advanced security properties [34,38,27]. The main theme of this work is how we implement the mechanism in the standard model when starting from an IND-CPA secure PKE scheme.

A naive idea for our goal would be to design the encryption algorithm as $\mathsf{Enc}(\mathsf{pk}, \mathsf{r} \| \mathsf{m}; \mathsf{r})$, where $\mathsf{Enc}(\mathsf{pk}, \cdot; \cdot)$ again denotes the encryption algorithm of an IND-CPA secure PKE scheme. Unfortunately, it seems difficult to prove the security of this construction based on its IND-CPA security, since in order to rely on IND-CPA security, we need to ensure that a message to be encrypted is completely independent of the encryption randomness r.

A natural idea to remove the dependency is to use a variant of the hybrid encryption paradigm. Namely, we design the encryption algorithm as $(\mathsf{Enc}(\mathsf{pk}, \mathsf{s}; \mathsf{r}), \mathsf{E}(\mathsf{s}, \mathsf{r} \| \mathsf{m}))$, where $\mathsf{E}(\mathsf{s}, \cdot)$ is the encryption algorithm of an SKE scheme. At first glance, the dependency is removed, but the construction is in fact at a "dead-lock" and it also seems difficult to prove its security. We can solve the dead-lock by using the *signaling technique*[4] recently introduced by Koppula and Waters [30] who showed how to construct IND-CCA secure ABE from IND-CPA secure one using a PRG with a special security property called hinting PRG.

2.2 Partial Randomness-Recovering Using the Signaling Technique

We now use $2n$ public keys $(\mathsf{pk}_i^v)_{i \in [n], v \in \{0,1\}}$ of the IND-CPA secure PKE scheme to encapsulate a secret key $\mathsf{s} = (\mathsf{s}_1, \ldots, \mathsf{s}_n) \in \{0,1\}^n$ of the SKE scheme, where $[n] := \{1, \ldots, n\}$. Below, let $(\mathsf{sk}_i^v)_{i \in [n], v \in \{0,1\}}$ be secret keys corresponding to $(\mathsf{pk}_i^v)_{i \in [n], v \in \{0,1\}}$. Roughly, we "encode" each bit s_i of s as $(\mathsf{ct}_i^0, \mathsf{ct}_i^1)$, where

$$\mathsf{ct}_i^{\mathsf{s}_i} = \mathsf{Enc}(\mathsf{pk}_i^{\mathsf{s}_i}, 1; \mathsf{r}_i^{\mathsf{s}_i}) \quad \text{and} \quad \mathsf{ct}_i^{1-\mathsf{s}_i} = \mathsf{Enc}(\mathsf{pk}_i^{1-\mathsf{s}_i}, 0; \mathsf{r}_i^{1-\mathsf{s}_i}).$$

Namely, we encapsulate s by using $2n$ ciphertexts $(\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i \in [n]}$. During the decapsulation, we decrypt ct_i^0 by using sk_i^0 and set $\mathsf{s}_i := 0$ if the decryption result is 1 and $\mathsf{s}_i := 1$ otherwise.

Of course, if we encrypt all of the random coins $(\mathsf{r}_i^v)_{i \in [n], v \in \{0,1\}}$ used to encapsulate s by the SKE scheme to make the resulting scheme randomness-recoverable, it leads to a dead-lock as before. However, by using the signaling technique used by Koppula and Waters, we can perform the validity check by re-encrypting n out of $2n$ ciphertexts of the IND-CPA secure PKE scheme in the decryption process, and solve the dead-lock as follows.

We say that "an encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1)$ signals α" when ct_i^α encrypts 1. By using an (ordinary) PRG and adding a "tag" T_i to each encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1)$ as $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$, we can build a mechanism ensuring that it is statistically impossible to generate an encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$ that signals both 0 and 1 at the same time. In order to implement this mechanism, we also add some random strings to the public key that are used to generate tags, but we ignore them for simplicity in this overview. In this case, we can perform the validity check of the key encapsulation part $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i \in [n]}$ by checking whether $(\mathsf{ct}_i^{\mathsf{s}_i})_{i \in [n]}$ are well-formed encryptions of 1 by re-encryption. This is intuitively because if we confirm that these n ciphertexts are encryptions of 1, we can also be sure that the remaining n ciphertexts $(\mathsf{ct}_i^{1-\mathsf{s}_i})_{i \in [n]}$ are not encrypting 1 due to the added mechanism based on the PRG and tags $(\mathsf{T}_i)_{i \in [n]}$, and thus we can finish the pseudo-validity-check of all $2n$ ciphertexts of the key encapsulation part. Thus, in this construction, in addition to a message to be encrypted, the SKE scheme needs to encrypt only n random coins $(\mathsf{r}_i^{\mathsf{s}_i})_{i \in [n]}$ used to generate $(\mathsf{ct}_i^{\mathsf{s}_i})_{i \in [n]}$.

[4] Garg, Gay, and Hajiabadi [20] also used a similar technique called *mirroring*.

2.3 Outline of the Proof: Necessity of KDM Secure SKE

We explain how to prove the IND-CCA security of the above construction. A ciphertext of the scheme is of the form

$$\Big((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i\in[n]},\ \mathsf{E}(\mathsf{s}, (\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}\|\mathsf{m}) \Big).$$

The general picture of the security proof is the same as that for the ordinary hybrid encryption scheme, and thus we first eliminate the information of s from the key encapsulation part $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i\in[n]}$ and then complete the entire proof by using the security of SKE.

We first explain how to eliminate the information of s from the key encapsulation part. In the security proof, thanks to the validity check by re-encryption in the decryption process, we can simulate the decryption oracle correctly by using $(\mathsf{sk}_i^{\mathsf{s}_i})_{i\in[n]}$ instead of $(\mathsf{sk}_i^0)_{i\in[n]}$. In this case, we can change the distribution of $(\mathsf{ct}_i^{1-\mathsf{s}_i})_{i\in[n]}$ in the challenge ciphertext by using the IND-CPA security of the PKE scheme since $(\mathsf{r}_i^{1-\mathsf{s}_i})_{i\in[n]}$ used to generate $(\mathsf{ct}_i^{1-\mathsf{s}_i})_{i\in[n]}$ are not encrypted by the SKE scheme and the decryption oracle can be simulated without $(\mathsf{sk}_i^{1-\mathsf{s}_i})_{i\in[n]}$. We can eliminate the information of s from the key encapsulation part $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i\in[n]}$ by changing $(\mathsf{ct}_i^{1-\mathsf{s}_i})_{i\in[n]}$ encrypting 0 into ciphertexts encrypting 1. This means that after this change, every encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$ contained in the challenge ciphertext signals 0 and 1 at the same time. While an adversary cannot generate such an encoding that signals 0 and 1 at the same time as noted above, the reduction algorithm can do it by programming random strings contained in the public key that are used to generate tags $(\mathsf{T}_i)_{i\in[n]}$.

Since we eliminate the information of s from the key encapsulation part above, it seems that we can complete the entire security proof by using the security of the SKE scheme. However, in order to do so, we need an SKE scheme that satisfies *KDM security*. This is because the underlying SKE scheme needs to encrypt $(\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}$, which is a message depending on the key s. Concretely, $(\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}$ can be seen as $f(\mathsf{s})$ for the function f that has $(\mathsf{r}_i^{v})_{i\in[n],v\in\{0,1\}}$ hardwired, and given $\mathsf{s}\in\{0,1\}^n$ outputs $(\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}$. Such a function is described as a very simple form of functions called projection functions, for which KDM security has been widely studied [11,3,12,2,13,16]. In our construction, we need an SKE scheme satisfying only *one-time* KDM security with respect to projection functions, since our construction is basically a hybrid encryption scheme. This is the reason KDM secure SKE is needed for our construction of IND-CCA secure PKE.

The Construction by Koppula and Waters [30]. The construction we explained so far is in fact almost the same as the PKE variant of the construction proposed by Koppula and Waters, except that a one-time KDM secure SKE scheme is used instead of a *hinting PRG*. Here, we briefly explain the notion of hinting PRG and how it is used in their construction.

A hinting PRG is a PRG that, given an n-bit string x, outputs an $(n+1)\cdot\ell$-bit string $y_0\|y_1\|\cdots\|y_n$, where y_i is an ℓ-bit string for every $i\in[n]$. Then, its security property requires that $Y := y_0\|(y_{i,0}\|y_{i,1})_{i\in[n]} \in \{0,1\}^{(2n+1)\cdot\ell}$ be indistinguishable from a uniformly random string in $\{0,1\}^{(2n+1)\cdot\ell}$, where $y_{i,x_i} = y_i$ and $y_{i,1-x_i}$ is a uniformly random string in $\{0,1\}^\ell$ for every $i \in [n]$. We see that the locations where $y_1, \cdots y_n$ are placed in Y depend on the seed x, and thus Y itself can be seen as a "hint" of the seed x. Therefore, we can say that the security property of a hinting PRG requires that its output be pseudorandom even if such a hint of the seed is revealed.

Koppula and Waters used a hinting PRG HPRG in their construction as follows. When encrypting a message m, their scheme first generates a seed $x = (x_1, \cdots, x_n) \in \{0,1\}^n$ of HPRG and computes $y_0 \| y_1 \| \cdots \| y_n \leftarrow \mathsf{HPRG}(x)$. Then, it generates an encapsulation of x by generating an encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$ of x_i in which y_i is used as the encryption randomness for $\mathsf{ct}_i^{x_i}$ for every $i \in [n]$. Note that $\mathsf{ct}_i^{1-x_i}$ is generated by using truly random coins. Moreover, it generates the data encapsulation part as $\mathsf{m} \oplus y_0$. The resulting ciphertext is of the form

$$\Big((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i\in[n]},\ \mathsf{m} \oplus y_0 \Big).$$

When decrypting the ciphertext, we can first recover x and thus $y_0 \| y_1 \| \cdots \| y_n \leftarrow \mathsf{HPRG}(x)$ from the encapsulation part. Since $(y_1, \cdots, y_n)$ are random coins used to generate $(\mathsf{ct}_i^{x_i})_{i\in[n]}$, we can also perform the pseudo-validity-check of all $2n$ ciphertexts of the key encapsulation part as we explained above. The security proof of their construction also goes through in a similar fashion to the proof of our construction, except that the security property of HPRG is utilized instead of KDM security.

2.4 Extension to TDF

We explain how we extend the above construction of IND-CCA secure PKE based on IND-CPA secure PKE and one-time KDM secure SKE, into a TDF. More concretely, we explain how we make the above construction completely randomness-recoverable.

In the above construction, there are two types of encryption randomness that are not recovered in the decryption process. The first one is $(\mathsf{r}_i^{1-\mathsf{s}_i})_{i\in[n]}$ for the underlying IND-CPA secure PKE scheme. The other one is the encryption randomness for the underlying SKE scheme. We require an additional requirement for each building block to make it possible to recover these two types of encryption randomness.

First, to deal with $(\mathsf{r}_i^{1-\mathsf{s}_i})_{i\in[n]}$ for the IND-CPA secure PKE scheme, we require the underlying IND-CPA secure PKE scheme have the *pseudorandom ciphertext property*. Namely, we require that a ciphertext of the underlying IND-CPA secure PKE scheme be indistinguishable from a uniformly random element sampled from the ciphertext space of the scheme. In the above construction, recall that we encode each bit s_i of s as $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$, where $\mathsf{ct}_i^{\mathsf{s}_i} = \mathsf{Enc}(\mathsf{pk}_i^{\mathsf{s}_i}, 1; \mathsf{r}_i^{\mathsf{s}_i})$ and $\mathsf{ct}_i^{1-\mathsf{s}_i} = \mathsf{Enc}(\mathsf{pk}_i^{1-\mathsf{s}_i}, 0; \mathsf{r}_i^{1-\mathsf{s}_i})$. We now modify the way s_i is encoded so that $\mathsf{ct}_i^{1-\mathsf{s}_i}$ is an element sampled from the ciphertext space uniformly at random. We can still decode s_i correctly with overwhelming probability thanks to the signaling technique even if we add this modification.[5] Then, we see that the issue of recovering $(\mathsf{r}_i^{1-\mathsf{s}_i})_{i\in[n]}$ is solved by designing the TDF such that $(\mathsf{ct}_i^{1-\mathsf{s}_i})_{i\in[n]}$ are directly sampled from the ciphertext space as part of an input to the TDF.

Second, to deal with the random coins for the SKE scheme, we simply require that the SKE scheme be *randomness-recoverable*. Namely, we require that random coins used to encrypt a message be recovered with the message in the decryption process. The randomness-recovering property is easy to achieve in the secret-key setting, and it is also the case even if we require the SKE scheme to be KDM secure. In fact, we can easily construct a one-time projection-KDM secure SKE scheme that is randomness-recoverable by modifying existing projection-KDM secure PKE schemes [11,12,39,13,16]. Moreover, the projection-KDM secure SKE schemes based on the LPN and LWE assumptions proposed by Applebaum et al. [3] already satisfy this property.

With the help of these two additional requirements, we can modify our IND-CCA secure PKE scheme into a TDF. Since our TDF is an extension of IND-CCA secure PKE, it naturally satisfies adaptive one-wayness [27].

2.5 Optimizations and Simplifications

Finally, we explain several optimizations and simplifications that are applied in the actual constructions.

The first optimization is on the number of key pairs of the underlying IND-CPA secure PKE scheme. In the above overview, $2n$ key pairs of the underlying IND-CPA secure PKE scheme are used to construct the key encapsulation part $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i\in[n]}$. In our actual constructions, we use only two key pairs of the underlying IND-CPA secure PKE scheme. More concretely, in our actual constructions, every encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$ is generated by using the same pair of public keys $(\mathsf{pk}^0, \mathsf{pk}^1)$. In fact, if we allow a public parameter shared by all users of the resulting schemes, even one of these public keys, pk^1, can be put into the public parameter, and a public key of the resulting IND-CCA secure scheme and an evaluation key of the resulting TDF consist only of a *single* public key pk^0 of the underlying IND-CPA secure scheme. This optimization is possible by devising at which step of the hybrid games we switch the secret keys of the underlying IND-CPA secure PKE scheme used to simulate the decryption oracle.

The second optimization is on how to make each tag T_i contained in each encoding $(\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)$. In the original signaling technique, a one-time signature scheme is additionally used in order to generate tags. We show that we can replace a one-time signature scheme with a target collision resistant hash function. Such a technique was previously used by Matsuda and Hanaoka [33]. Although both of these primitives can be realized using only a one-way function as an assumption [37], this improvement is critical when constructing a TDF since if we attempt to use a one-time signature scheme for constructing a TDF, we would need to recover the random coins used to generate a key pair of the one-time

[5] While we cannot achieve perfect correctness by this modification, we can still achieve almost-all-keys correctness [18]. For its formal definition, see Section 3.

signature scheme during the inversion process. We can avoid this issue by the use of a target collision resistant hash function instead. This modification is made possible due to the use of a deferred analysis technique in the security proof.

Third, we make a simplification by using key encapsulation mechanism (KEM) instead of PKE. In this overview, we have explained how to construct an IND-CCA secure PKE scheme and a TDF based on IND-CPA secure PKE by additionally using KDM secure SKE. In our actual proposals, we construct an IND-CCA secure KEM (and a TDF) based on IND-CPA secure KEM and KDM secure SKE. As explained above, in the original signaling technique, we use an (ordinary) PRG. More precisely, in the original signaling technique, $\mathsf{ct}_i^{\mathsf{s}_i}$ in each encoding is generated as $\mathsf{ct}_i^{\mathsf{s}_i} = \mathsf{Enc}(\mathsf{pk}_i^{\mathsf{s}_i}, 1\|\mathsf{u}_i; \mathsf{r}_i^{\mathsf{s}_i})$, where u_i is a seed of PRG. In our actual construction, in order to hide the use of a PRG from the description and simplify the construction, we use a KEM whose session-key space is sufficiently larger than its randomness space. We can generically transform an IND-CPA secure PKE scheme into a KEM with such a property. We show that the signaling technique can be implemented by using such a KEM.

For the construction of TDF, we also add an optimization that is made possible by the pseudorandom ciphertext property of the underlying IND-CPA secure PKE scheme. By this optimization, an image of a function consists of n ciphertexts of the IND-CPA secure PKE scheme corresponding to $(\mathsf{ct}_i^{s_i})_{i\in[n]}$, n tags $(\mathsf{T}_i)_{i\in[n]}$, and a ciphertext of the SKE scheme.

We finally remark that all of the above optimizations and simplifications can be brought back to the construction of an IND-CCA secure ABE scheme based on an IND-CPA secure one and a hinting PRG by Koppula and Waters [30].

3 Preliminaries

In this section, we review the basic notation and the definitions of main cryptographic primitives. For the definitions of primitives that are not reviewed here, see the full version of this paper [29].

Basic Notation. $\mathbb{N}$ denotes the set of natural numbers, and for $n \in \mathbb{N}$, we define $[n] := \{1, \ldots, n\}$. For a discrete finite set S, $|S|$ denotes its size, and $x \xleftarrow{\mathsf{r}} S$ denotes choosing an element x uniformly at random from S. For strings x and y, $x\|y$ denotes their concatenation. For a (probabilistic) algorithm or a function A, $y \leftarrow \mathsf{A}(x)$ denotes assigning to y the output of A on an input x, and if we need to specify a randomness r used in A, we denote $y \leftarrow \mathsf{A}(x; r)$ (in which case the computation of A is understood as deterministic on input x and r). $\mathsf{Sup}(\mathsf{A})$ denotes the support of A. For any values x, y, $(x \overset{?}{=} y)$ is defined to be 1 if $x = y$ and 0 otherwise. λ denotes a security parameter. (P)PT stands for *(probabilistic) polynomial time.* A function $f(\lambda)$ is said to be *negligible* if $f(\lambda)$ tends to 0 faster than $\frac{1}{\lambda^c}$ for every constant $c > 0$. We write $f(\lambda) = \mathsf{negl}(\lambda)$ to denote that $f(\lambda)$ is a negligible function. $\mathsf{poly}(\cdot)$ denotes an unspecified positive polynomial.

3.1 Key Encapsulation Mechanism

Here, we review the definitions for a KEM. For the definition of correctness, we formalize "almost-all-keys" correctness, which is naturally adapted from the definition for PKE formalized by Dwork, Naor, and Reingold [18].

A key encapsulation mechanism (KEM) KEM consists of the three PPT algorithms $(\mathsf{KKG}, \mathsf{Encap}, \mathsf{Decap})$. KKG is the key generation algorithm that takes 1^λ as input, and outputs a public/secret key pair $(\mathsf{pk}, \mathsf{sk})$. We assume that the security parameter λ determines the ciphertext space $\mathcal{C}$, the session-key space $\mathcal{K}$, and the randomness space $\mathcal{R}$ of Encap. Encap is the encapsulation algorithm that takes a public key pk as input, and outputs a ciphertext/session-key pair $(\mathsf{ct}, \mathsf{k})$. Decap is the (deterministic) decapsulation algorithm that takes a secret key sk and a ciphertext ct as input, and outputs a session-key k or the invalid symbol $\perp \notin \mathcal{K}$.

Letting $\epsilon : \mathbb{N} \to [0, 1]$, we say that a KEM $\mathsf{KEM} = (\mathsf{KKG}, \mathsf{Encap}, \mathsf{Decap})$ is *ϵ-almost-all-keys correct* if we have

$$\mathsf{Err}_{\mathsf{KEM}}(\lambda) := \Pr_{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KKG}(1^\lambda)}\left[\exists \mathsf{r} \in \mathcal{R} \text{ s.t. } \begin{array}{c} \mathsf{Encap}(\mathsf{pk}; \mathsf{r}) = (\mathsf{ct}, \mathsf{k}) \\ \wedge\ \mathsf{Decap}(\mathsf{sk}, \mathsf{ct}) \neq \mathsf{k} \end{array} \right] = \epsilon(\lambda).$$

(A public key pk under which incorrect decapsulation could occur is called *erroneous*.) Furthermore, we just say that KEM is *correct* (resp. *almost-all-keys correct*) if $\mathsf{Err}_{\mathsf{KEM}}(\lambda)$ is zero (resp. $\mathsf{negl}(\lambda)$).

Now we review the security definitions for a KEM used in this paper, which are *IND-CCA security*, *IND-CPA security*, and the *pseudorandom ciphertext* property. For convenience, we will define the multi-challenge versions for the latter two notions, which are polynomially equivalent to the single-challenge versions via a standard hybrid argument.

Definition 1 (Security Notions for a KEM). *Let* $\mathsf{KEM} = (\mathsf{KKG}, \mathsf{Encap}, \mathsf{Decap})$ *be a KEM whose ciphertext and session-key spaces are* $\mathcal{C}$ *and* $\mathcal{K}$*, respectively. We say that* KEM *satisfies*

- IND-CCA security *if for all PPT adversaries* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{cca}}_{\mathsf{KEM},\mathcal{A}}(\lambda) := 2 \cdot |\Pr[\mathsf{Expt}^{\mathsf{cca}}_{\mathsf{KEM},\mathcal{A}}(\lambda) = 1] - 1/2| = \mathsf{negl}(\lambda)$*, where* $\mathsf{Expt}^{\mathsf{cca}}_{\mathsf{KEM},\mathcal{A}}(\lambda)$ *is defined as in Figure 1 (left), and in the experiment,* $\mathcal{A}$ *is not allowed to submit* ct^* *to the decapsulation oracle* $\mathsf{Decap}(\mathsf{sk}, \cdot)$*.*
- IND-CPA security *if for all PPT adversaries* $\mathcal{A}$ *and all polynomials* $\ell = \ell(\lambda)$*, we have* $\mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda) := 2 \cdot |\Pr[\mathsf{Expt}^{\mathsf{mcpa}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda) = 1] - 1/2| = \mathsf{negl}(\lambda)$*, where* $\mathsf{Expt}^{\mathsf{mcpa}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda)$ *is defined as in Figure 1 (center).*
- *the* pseudorandom ciphertext property *if for all PPT adversaries* $\mathcal{A}$ *and all polynomials* $\ell = \ell(\lambda)$*, we have* $\mathsf{Adv}^{\mathsf{mprct}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda) := 2 \cdot |\Pr[\mathsf{Expt}^{\mathsf{mprct}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda) = 1] - 1/2| = \mathsf{negl}(\lambda)$*, where* $\mathsf{Expt}^{\mathsf{mprct}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda)$ *is defined as in Figure 1 (right).*

3.2 Secret-Key Encryption

A secret-key encryption (SKE) scheme SKE consists of the three PPT algorithms $(\mathsf{K}, \mathsf{E}, \mathsf{D})$. K is the key generation algorithm that takes 1^λ as input, and outputs

```
$\mathsf{Expt}^{\mathsf{cca}}_{\mathsf{KEM},\mathcal{A}}(\lambda)$ :
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KKG}(1^\lambda)$
    $(\mathsf{ct}^*, \mathsf{k}^*_1) \leftarrow \mathsf{Encap}(\mathsf{pk})$
    $\mathsf{k}^*_0 \leftarrow \mathcal{K}$
    $b \xleftarrow{\mathsf{r}} \{0,1\}$
    $b' \leftarrow \mathcal{A}^{\mathsf{Decap}(\mathsf{sk},\cdot)}(\mathsf{pk}, \mathsf{ct}^*, \mathsf{k}^*_b)$
    Return $(b' \overset{?}{=} b)$.
```

```
$\mathsf{Expt}^{\mathsf{mcpa}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda)$ :
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KKG}(1^\lambda)$
    $\forall i \in [\ell]$ :
        $(\mathsf{ct}^*_i, \mathsf{k}^*_{i,1}) \leftarrow \mathsf{Encap}(\mathsf{pk})$
        $\mathsf{k}^*_{i,0} \xleftarrow{\mathsf{r}} \mathcal{K}$
    $b \xleftarrow{\mathsf{r}} \{0,1\}$
    $b' \leftarrow \mathcal{A}(\mathsf{pk}, (\mathsf{ct}^*_i, \mathsf{k}^*_{i,b})_{i \in [\ell]})$
    Return $(b' \overset{?}{=} b)$.
```

```
$\mathsf{Expt}^{\mathsf{mprct}}_{\mathsf{KEM},\ell,\mathcal{A}}(\lambda)$ :
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KKG}(1^\lambda)$
    $\forall i \in [\ell]$ :
        $(\mathsf{ct}^*_{i,1}, \mathsf{k}^*_{i,1}) \leftarrow \mathsf{Encap}(\mathsf{pk})$
        $(\mathsf{ct}^*_{i,0}, \mathsf{k}^*_{i,0}) \xleftarrow{\mathsf{r}} \mathcal{C} \times \mathcal{K}$
    $b \xleftarrow{\mathsf{r}} \{0,1\}$
    $b' \leftarrow \mathcal{A}(\mathsf{pk}, (\mathsf{ct}^*_{i,b}, \mathsf{k}^*_{i,b})_{i \in [\ell]})$
    Return $(b' \overset{?}{=} b)$.
```

Fig. 1. Security experiments for a KEM: IND-CCA experiment (left), (Multi-challenge) IND-CPA experiment (center), and the (multi-challenge) pseudorandom ciphertext property experiment (right).

a secret key sk. We assume that the security parameter λ determines the secret key space $\mathcal{K}$ and the message space $\mathcal{M}$. E is the encryption algorithm that takes a secret key sk and a plaintext m as input, and outputs a ciphertext ct. D is the (deterministic) decryption algorithm that takes a secret key sk and a ciphertext ct as input, and outputs a plaintext m or the invalid symbol $\bot \notin \mathcal{M}$. An SKE scheme $\mathsf{SKE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ is said to be *correct* if for all $\mathsf{sk} \in \mathcal{K}$ and all $\mathsf{m} \in \mathcal{M}$, it holds that $\mathsf{D}(\mathsf{sk}, \mathsf{E}(\mathsf{sk}, \mathsf{m})) = \mathsf{m}$.

In our proposed constructions of a TDF, we will use an SKE scheme that satisfies the "randomness-recovering decryption" property, which requires that for an honestly generate ciphertext, the randomness used to generate it can be recovered in the decryption process. We formally define the property as follows.

Definition 2 (Randomness-Recovering Decryption). *Let* $\mathsf{SKE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ *be an SKE scheme whose secret key space is* $\mathcal{K}$*, whose plaintext space is* $\mathcal{M}$*, and the randomness space of whose encryption algorithm* E *is* $\mathcal{R}$*. We say that* SKE *satisfies the* randomness-recovering decryption *property, if there exists a deterministic PT algorithm* RD *(called the* randomness-recovering decryption *algorithm) such that for all* $\mathsf{sk} \in \mathcal{K}$*, all* $\mathsf{m} \in \mathcal{M}$*, and all* $\mathsf{r} \in \mathcal{R}$*, we have* $\mathsf{RD}(\mathsf{sk}, \mathsf{E}(\mathsf{sk}, \mathsf{m}; \mathsf{r})) = (\mathsf{m}, \mathsf{r})$.

Here, we recall KDM security of an SKE scheme. For simplicity, we only give the definition for the single key setting, which is sufficient for our purpose.

Definition 3 (KDM Security). *Let* $\mathsf{SKE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ *be an SKE scheme with a secret key space* $\mathcal{K}$ *and a plaintext space* $\mathcal{M}$*. For a family of functions* $\mathcal{F}$ *with domain* $\mathcal{K}$ *and range* $\mathcal{M}$ *and an adversary* $\mathcal{A}$*, consider the experiment* $\mathsf{Expt}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{F},\mathcal{A}}(\lambda)$ *defined as in Figure 2 (left), where the KDM-encryption oracle* $\mathcal{O}_{\mathsf{kdm}}$ *is described in Figure 2 (center).*

We say that SKE *is* $\mathcal{F}$-KDM secure *if for all PPT adversaries* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{F},\mathcal{A}}(\lambda) := 2 \cdot |\Pr[\mathsf{Expt}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{F},\mathcal{A}}(\lambda) = 1] - 1/2| = \mathsf{negl}(\lambda)$.

Furthermore, we say that SKE *is* one-time $\mathcal{F}$-KDM secure *if* $\mathsf{Adv}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{F},\mathcal{A}}(\lambda)$ $= \mathsf{negl}(\lambda)$ *for all PPT adversaries* $\mathcal{A}$ *that make a single KDM-encryption query.*

```
$\mathsf{Expt}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{F},\mathcal{A}}(\lambda)$ :
  $\mathsf{sk} \leftarrow \mathsf{K}(1^\lambda)$
  $b \xleftarrow{\mathsf{r}} \{0,1\}$
  $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{kdm}}(\cdot,\cdot)}(1^\lambda)$
  Return $(b' \overset{?}{=} b)$.
```

```
$\mathcal{O}_{\mathsf{kdm}}((f_0, f_1) \in \mathcal{F}^2)$ :
  $\mathsf{ct} \leftarrow \mathsf{E}(\mathsf{sk}, f_b(\mathsf{sk}))$
  Return $\mathsf{ct}$.
```

```
$\mathsf{Expt}^{\mathsf{aow}}_{\mathsf{TDF},\mathcal{A}}(\lambda)$ :
  $(\mathsf{ek}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda)$
  $\mathsf{x}^* \leftarrow \mathsf{Samp}(1^\lambda)$
  $\mathsf{y}^* \leftarrow \mathsf{Eval}(\mathsf{ek}, \mathsf{x}^*)$
  $\mathsf{x}' \leftarrow \mathcal{A}^{\mathsf{Inv}(\mathsf{td},\cdot)}(\mathsf{ek}, \mathsf{y}^*)$
  Return $(\mathsf{x}' \overset{?}{=} \mathsf{x})$.
```

Fig. 2. The KDM security experiment for an SKE (left) scheme, the KDM-encryption oracle used in the KDM security experiment (center), and the adaptive one-wayness experiment for a TDF (right).

Function Families for KDM Security. We will deal with the following function families for KDM security of an SKE scheme with key space $\mathcal{K}$ and plaintext space $\mathcal{M}$:

- $\mathcal{P}$ *(Projection functions):* A function is said to be a projection function if each of its output bits depends on at most a single bit of its input. We denote by $\mathcal{P}$ the family of projection functions with domain $\mathcal{K}$ and range $\mathcal{M}$.
- $\mathcal{B}_{\mathsf{size}}$ *(Circuits of a-priori bounded size* size*):* We denote by $\mathcal{B}_{\mathsf{size}}$, where $\mathsf{size} = \mathsf{size}(\lambda)$ is a polynomial, the function family with domain $\mathcal{K}$ and range $\mathcal{M}$ such that each member in $\mathcal{B}_{\mathsf{size}}$ can be described by a circuit of size size.

3.3 Trapdoor Function

Here, we review the definitions for a TDF. As in the KEM case, for correctness, we will define almost-all-keys correctness.

A trapdoor function (TDF) TDF consists of the four PPT algorithms $(\mathsf{Setup}, \mathsf{Samp}, \mathsf{Eval}, \mathsf{Inv})$: Setup is the setup algorithm that takes 1^λ as input, and outputs an evaluation key/trapdoor pair $(\mathsf{ek}, \mathsf{td})$. We assume that the security parameter λ determines the domain $\mathcal{X}$. Samp is the domain sampling algorithm that takes 1^λ as input, and outputs a domain element $x \in \mathcal{X}$. Eval is the evaluation algorithm that takes an evaluation key ek and a domain element x as input, and outputs some element y. Inv is the (deterministic) inversion algorithm that takes a trapdoor td and an element y as input, and outputs some element x which could be the invalid symbol $\perp \notin \mathcal{X}$.

Letting $\epsilon : \mathbb{N} \rightarrow [0,1]$, we say that a TDF $\mathsf{TDF} = (\mathsf{Setup}, \mathsf{Samp}, \mathsf{Eval}, \mathsf{Inv})$ is *ϵ-almost-all-keys correct* if we have

$$\mathsf{Err}_{\mathsf{TDF}}(\lambda) := \Pr_{(\mathsf{ek},\mathsf{td})\leftarrow\mathsf{Setup}(1^\lambda)}\Big[\ \exists x \in \mathcal{X} \text{ s.t. } \mathsf{Inv}(\mathsf{td}, \mathsf{Eval}(\mathsf{ek}, x)) \neq x\ \Big] = \epsilon(\lambda).$$

Furthermore, we just say that TDF is *correct* (resp. *almost-all-keys correct*) if $\mathsf{Err}_{\mathsf{TDF}}(\lambda)$ is zero (resp. $\mathsf{negl}(\lambda)$).

Definition 4 (Adaptive One-wayness/(Ordinary) One-wayness). *Let* $\mathsf{TDF} = (\mathsf{Setup}, \mathsf{Samp}, \mathsf{Eval}, \mathsf{Inv})$ *be a TDF with domain $\mathcal{X}$. We say that* TDF *is* adaptively one-way *if for all PPT adversaries $\mathcal{A}$, we have* $\mathsf{Adv}^{\mathsf{aow}}_{\mathsf{TDF},\mathcal{A}}(\lambda) :=$

$\Pr[\mathsf{Expt}^{\mathsf{aow}}_{\mathsf{TDF},\mathcal{A}}(\lambda) = 1] = \mathsf{negl}(\lambda)$, *where* $\mathsf{Expt}^{\mathsf{aow}}_{\mathsf{TDF},\mathcal{A}}(\lambda)$ *is defined as in Figure 2 (right), and in the experiment,* $\mathcal{A}$ *is not allowed to submit* y^* *to the inversion oracle* $\mathsf{Inv}(\mathsf{td}, \cdot)$.

Furthermore, we say that TDF *is* one-way *if* $\mathsf{Adv}^{\mathsf{aow}}_{\mathsf{TDF},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$ *for all adversaries that never use the inversion oracle* $\mathsf{Inv}(\mathsf{td}, \cdot)$.

4 Chosen Ciphertext Security via KDM Security

In this section, we show our proposed construction of an IND-CCA secure KEM.

Specifically, in Section 4.1, we present the formal description of our proposed KEM, state theorems regarding its correctness/security, and discuss its consequences and extensions. Then, in Sections 4.2 and 4.3, we prove the correctness and IND-CCA security of our proposed construction, respectively.

4.1 Our Construction

Let $\ell = \ell(\lambda)$ be a polynomial, which will denote the session-key length of the constructed KEM. Our construction uses the building blocks KEM, SKE, and Hash with the following properties:

- $\mathsf{KEM} = (\mathsf{KKG}, \mathsf{Encap}, \mathsf{Decap})$ is a KEM such that (1) its session-key space is $\{0,1\}^{4\lambda}$, (2) the randomness space of Encap is $\{0,1\}^{\lambda}$, and (3) the image size of $\mathsf{Decap}(\mathsf{sk}, \cdot)$ for any sk output by $\mathsf{KKG}(1^{\lambda})$ (other than $\bot$) is at most 2^{λ}.[6]
- $\mathsf{SKE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ is an SKE scheme whose secret key space is $\{0,1\}^n$ for some polynomial $n = n(\lambda)$ and whose plaintext space is $\{0,1\}^{n\cdot\lambda+\ell}$, and we denote the randomness space of E by $\mathcal{R}_{\mathsf{SKE}}$.
- $\mathsf{Hash} = (\mathsf{HKG}, \mathsf{H})$ is a keyed hash function such that the range of H is $\{0,1\}^{\lambda}$, which we are going to assume to be target collision resistant.

Using these building blocks, the proposed KEM $\mathsf{KEM}_{\mathsf{cca}} = (\mathsf{KKG}_{\mathsf{cca}}, \mathsf{Encap}_{\mathsf{cca}}, \mathsf{Decap}_{\mathsf{cca}})$ is constructed as in Figure 3. Its session-key space is $\{0,1\}^{\ell}$, and the randomness space $\mathcal{R}$ of $\mathsf{Encap}_{\mathsf{cca}}$ is $\mathcal{R} = \{0,1\}^n \times (\{0,1\}^{\lambda})^{2n} \times \{0,1\}^{\ell} \times \mathcal{R}_{\mathsf{SKE}}$.

For the correctness and security of $\mathsf{KEM}_{\mathsf{cca}}$, the following theorems hold.

Theorem 1. *Let* $\epsilon = \epsilon(\lambda) \in [0,1]$. *If* KEM *is* ϵ*-almost-all-keys correct and* SKE *is correct, then* $\mathsf{KEM}_{\mathsf{cca}}$ *is* $(\epsilon + n \cdot 2^{-\lambda})$*-almost-all-keys correct.*

Theorem 2. *Assume that* KEM *is almost-all-keys correct and IND-CPA secure,* SKE *is one-time* $\mathcal{P}$*-KDM secure, and* Hash *is target collision resistant. Then,* $\mathsf{KEM}_{\mathsf{cca}}$ *is IND-CCA secure.*

The proofs of Theorems 1 and 2 are given in Sections 4.2 and 4.3, respectively.

[6] These three requirements are without loss of generality for an IND-CPA secure KEM: The properties (1) and (3) can be achieved by stretching a session-key of a KEM with session-key space $\{0,1\}^{\lambda}$ by using a PRG $\mathsf{G} : \{0,1\}^{\lambda} \to \{0,1\}^{4\lambda}$, and the randomness space of Encap can also be freely adjusted by using a PRG.

```
$\mathsf{KKG_{cca}}(1^\lambda)$ :
  $\forall v \in \{0,1\} : (\mathsf{pk}^v, \mathsf{sk}^v) \leftarrow \mathsf{KKG}(1^\lambda)$
  $\mathsf{A}_1, \dots, \mathsf{A}_n, \mathsf{B} \xleftarrow{\mathsf{r}} \{0,1\}^{4\lambda}$
  $\mathsf{hk} \leftarrow \mathsf{HKG}(1^\lambda)$
  $\mathsf{PK} \leftarrow (\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i)_{i \in [n]}, \mathsf{B}, \mathsf{hk})$
  $\mathsf{SK} \leftarrow (\mathsf{sk}^0, \mathsf{PK})$
  Return $(\mathsf{PK}, \mathsf{SK})$.
```

```
$\mathsf{Encap_{cca}}(\mathsf{PK})$ :
  $(\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i)_{i \in [n]}, \mathsf{B}, \mathsf{hk}) \leftarrow \mathsf{PK}$
  $\mathsf{s} = (\mathsf{s}_1, \dots, \mathsf{s}_n) \leftarrow \mathsf{K}(1^\lambda)$
  $\mathsf{r}_1^0, \dots, \mathsf{r}_n^0, \mathsf{r}_1^1, \dots, \mathsf{r}_n^1 \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$
  $\mathsf{k} \xleftarrow{\mathsf{r}} \{0,1\}^\ell$
  $\mathsf{ct_{SKE}} \leftarrow \mathsf{E}(\mathsf{s}, (\mathsf{r}_i^{\mathsf{s}_i})_{i \in [n]} \| \mathsf{k})$
  $\forall (i, v) \in [n] \times \{0,1\}$ :
    $(\mathsf{ct}_i^v, \mathsf{k}_i^v) \leftarrow \mathsf{Encap}(\mathsf{pk}^v; \mathsf{r}_i^v)$
  $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i \in [n]} \| \mathsf{ct_{SKE}})$
  $\forall i \in [n]$ :
    $\mathsf{T}_i \leftarrow \mathsf{k}_i^{\mathsf{s}_i} + \mathsf{s}_i \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h})$   $^{(\dagger)}$
      $= \begin{cases} \mathsf{k}_i^0 & \text{if } \mathsf{s}_i = 0 \\ \mathsf{k}_i^1 + \mathsf{A}_i + \mathsf{B} \cdot \mathsf{h} & \text{if } \mathsf{s}_i = 1 \end{cases}$
  $\mathsf{CT} \leftarrow ((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i \in [n]}, \mathsf{ct_{SKE}})$
  Return $(\mathsf{CT}, \mathsf{k})$.
```

```
$\mathsf{Decap_{cca}}(\mathsf{SK}, \mathsf{CT})$ :
  $(\mathsf{sk}^0, \mathsf{PK}) \leftarrow \mathsf{SK}$
  $(\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i)_{i \in [n]}, \mathsf{B}, \mathsf{hk}) \leftarrow \mathsf{PK}$
  $((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i \in [n]}, \mathsf{ct_{SKE}}) \leftarrow \mathsf{CT}$
  $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i \in [n]} \| \mathsf{ct_{SKE}})$
  $\forall i \in [n]$ :
    $\mathsf{s}_i \leftarrow 1 - (\mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i^0) \overset{?}{=} \mathsf{T}_i)$   $^{(\star)}$
      $= \begin{cases} 0 & \text{if } \mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i^0) = \mathsf{T}_i \\ 1 & \text{otherwise} \end{cases}$
  $\mathsf{s} \leftarrow (\mathsf{s}_1, \dots, \mathsf{s}_n) \in \{0,1\}^n$
  $\mathsf{m} \leftarrow \mathsf{D}(\mathsf{s}, \mathsf{ct_{SKE}})$
  Parse $\mathsf{m}$ as $((\mathsf{r}_i^{\mathsf{s}_i})_{i \in [n]}, \mathsf{k}) \in (\{0,1\}^\lambda)^n \times \{0,1\}^\ell$.
  If $\forall i \in [n]$ :
    $\mathsf{Encap}(\mathsf{pk}^{\mathsf{s}_i}; \mathsf{r}_i^{\mathsf{s}_i}) = (\mathsf{ct}_i^{\mathsf{s}_i}, \mathsf{T}_i - \mathsf{s}_i \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}))$
          then return $\mathsf{k}$ else return $\perp$.   $^{(\dagger)}$
```

Fig. 3. The proposed KEM $\mathsf{KEM_{cca}}$. $^{(\dagger)}$ $\mathsf{h} \in \{0,1\}^\lambda$ is treated as an element of $\{0,1\}^{4\lambda}$ by some canonical injective encoding (say, putting the prefix $0^{3\lambda}$), and the arithmetic is done over $\mathrm{GF}(2^{4\lambda})$ where we identify $\{0,1\}^{4\lambda}$ with $\mathrm{GF}(2^{4\lambda})$. $^{(\star)}$ We call this step the *find step.*

Implications to Black-Box Constructions/Reductions. It is straightforward to see that our construction uses the underlying primitives in a black-box manner. As will be clear from our security proof, our reduction algorithms also treat the underlying primitives and an adversary in a black-box manner. In fact, our construction/reduction is fully black-box in the sense of [36]. Since there exists a black-box construction of a target collision resistant hash function from a one-way function, which can be trivially constructed from an IND-CPA secure PKE scheme/KEM in a black-box manner, and since an IND-CCA/CPA PKE scheme and KEM imply each other (in a black-box manner), we obtain the following result as a corollary of our theorems.

Corollary 1. *There exists a fully black-box construction of an IND-CCA secure PKE scheme/KEM from an IND-CPA secure PKE scheme/KEM and a one-time $\mathcal{P}$-KDM secure SKE scheme that can encrypt plaintexts of length $\Omega(n \cdot \lambda)$, where $n = n(\lambda)$ is the secret key length of the SKE scheme.*

Furthermore, since a $\mathcal{P}$-KDM secure *PKE* scheme trivially implies both an IND-CPA secure PKE scheme/KEM and a one-time $\mathcal{P}$-KDM secure SKE scheme, we obtain another corollary.

Corollary 2. *There exists a fully black-box construction of an IND-CCA secure PKE scheme/KEM from a $\mathcal{P}$-KDM secure PKE scheme.*

In contrast to Corollary 2, in Section 5, we will show that there exists no *shielding* black-box construction [21] of an IND-CCA1 secure PKE scheme from a $\mathcal{P}$-KDM secure PKE scheme.

In [33], Matsuda and Hanaoka showed a construction of an IND-CCA secure PKE scheme/KEM from a PKE scheme satisfying the security notion called the sender non-committing property and a one-time $\mathcal{B}_{\mathsf{size}}$-KDM secure SKE scheme (where size is related to the running time of the sender non-committing encryption scheme). Although their construction uses the underlying primitives as black-boxes, their security reduction (to the $\mathcal{B}_{\mathsf{size}}$-KDM security of the underlying SKE scheme) is non-black-box in the sense that the reduction needs to use the description of one of the algorithms in the sender non-committing encryption scheme as a KDM-encryption query. Compared to the result by Matsuda and Hanaoka, our results are superior in terms of both the strength of the assumptions on the building blocks (IND-CPA security is weaker than the sender non-committing property, and $\mathcal{P}$-KDM security is weaker than $\mathcal{B}_{\mathsf{size}}$-KDM security), and the "black-boxness" of the reductions.

Hinting PRG vs. KDM Secure SKE. As mentioned earlier, the result of Koppula and Waters [30], when specialized to PKE, implies that if there exists an IND-CPA secure PKE scheme and a hinting PRG, one can realize an IND-CCA secure PKE scheme. Given our result in this section and the result of [30], it is natural to ask whether there exists an implication/separation between a (one-time) KDM secure SKE scheme and a hinting PRG. We give a partial affirmative answer to this question. Specifically, we show the following theorem.

Theorem 3. *If there exists a hinting PRG, then for any polynomials $m = m(\lambda)$ and $\mathsf{size} = \mathsf{size}(\lambda) \geq m$, there exists a one-time $\mathcal{B}_{\mathsf{size}}$-KDM secure SKE scheme whose plaintext space is $\{0,1\}^m$. Furthermore, for any polynomial $m = m(\lambda)$, there exists a fully black-box construction of a one-time $\mathcal{P}$-KDM secure SKE scheme with plaintext space $\{0,1\}^m$ from a hinting PRG.*

The formal proof of this theorem is given in the full version of this paper [29]. This result shows that the existence of a KDM-secure SKE scheme is not stronger (as an assumption) than that of a hinting PRG. At this moment, it is not clear if the implication of the opposite direction can be established.

Additional Remarks.

– If we adopt the syntax of a KEM in which there is a public parameter shared by all users, then we can push pk^1, $(\mathsf{A}_i)_{i\in[n]}$, B, and hk in PK to a public parameter, so that a key pair of each user consists only of a single key pair $(\mathsf{pk}^0, \mathsf{sk}^0)$ of the underlying IND-CPA secure KEM.
– Although our proposed construction satisfies only almost-all-keys correctness, a minor variant of our construction can achieve perfect correctness, by using a PKE scheme and a PRG, instead of a KEM, as done in the Koppula-Waters construction [30].

4.2 Proof of Correctness (Proof of Theorem 1)

Let $\mathsf{PK} = (\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i)_{i\in[n]}, \mathsf{B}, \mathsf{hk})$ be a public key. Using pk^0, pk^1, and B in PK, we define the function $f : \{0,1\}^{3\lambda} \to \{0,1\}^{4\lambda}$ by

$$f(\mathsf{r}, \mathsf{r}', \mathsf{h}) : \Big[(\mathsf{ct}, \mathsf{k}) \leftarrow \mathsf{Encap}(\mathsf{pk}^0; \mathsf{r});\ (\mathsf{ct}', \mathsf{k}') \leftarrow \mathsf{Encap}(\mathsf{pk}^1; \mathsf{r}');\ \text{Return } \mathsf{k} - \mathsf{k}' - \mathsf{B} \cdot \mathsf{h}\Big].$$

We say that a public key PK is *bad* if (1) pk^0 is erroneous, or (2) some of $(\mathsf{A}_i)_{i\in[n]}$ belongs to the image of f. Note that the image size of f is at most $2^{3\lambda}$. Since each A_i is chosen uniformly at random from $\{0,1\}^{4\lambda}$, when $\mathsf{KKG}_{\mathsf{cca}}(1^\lambda)$ is executed, the probability that a bad PK is output is at most $\epsilon + n \cdot \frac{2^{3\lambda}}{2^{4\lambda}} = \epsilon + n \cdot 2^{-\lambda}$.

Now, consider the case that $(\mathsf{PK}, \mathsf{SK})$ is output by $\mathsf{KKG}_{\mathsf{cca}}$ and PK is not bad. Let $\mathsf{R} = (\mathsf{s} = (\mathsf{s}_1, \dots, \mathsf{s}_n), (\mathsf{r}_i^0, \mathsf{r}_i^1)_{i\in[n]}, \mathsf{k}, \mathsf{r}_{\mathsf{SKE}}) \in \{0,1\}^n \times (\{0,1\}^\lambda)^{2n} \times \{0,1\}^\ell \times \mathcal{R}_{\mathsf{SKE}}$ be a randomness for $\mathsf{Encap}_{\mathsf{cca}}$, and let $(\mathsf{CT} = ((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i\in[n]}, \mathsf{ct}_{\mathsf{SKE}}), \mathsf{k}) = \mathsf{Encap}_{\mathsf{cca}}(\mathsf{PK}; \mathsf{R})$. Moreover, for each $i \in [n]$, let $\mathsf{s}'_i := 1 - (\mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i^0) \overset{?}{=} \mathsf{T}_i)$.

Note that if $\mathsf{s}'_i = \mathsf{s}_i$ holds for all $i \in [n]$, then the decryption result of $\mathsf{ct}_{\mathsf{SKE}}$ using $\mathsf{s}' = (\mathsf{s}'_1, \dots, \mathsf{s}'_n)$ as a secret key is exactly $(\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]} \| \mathsf{k}$ due to the correctness of SKE. Thus, the validity check done in the last step of $\mathsf{Decap}_{\mathsf{cca}}$ never fails, and $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT})$ will output k.

Hence, it remains to show that $\mathsf{s}'_i = \mathsf{s}_i$ holds for all $i \in [n]$.

- For positions i with $\mathsf{s}_i = 0$, we have $(\mathsf{ct}_i^0, \mathsf{k}_i^0 = \mathsf{T}_i) = \mathsf{Encap}(\mathsf{pk}^0; \mathsf{r}_i^0)$. Thus, the property that pk^0 is not erroneous implies $\mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i^0) = \mathsf{T}_i$, and we have $\mathsf{s}'_i = 0$.
- For positions i with $\mathsf{s}_i = 1$, we have $(\mathsf{ct}_i^0, \mathsf{k}_i^0) = \mathsf{Encap}(\mathsf{pk}^0; \mathsf{r}_i^0)$ and $(\mathsf{ct}_i^1, \mathsf{k}_i^1 = \mathsf{T}_i - \mathsf{A}_i - \mathsf{B} \cdot \mathsf{h}) = \mathsf{Encap}(\mathsf{pk}^1; \mathsf{r}_i^1)$, where $\mathsf{h} = \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i\in[n]} \| \mathsf{ct}_{\mathsf{SKE}})$. Since A_i is not in the image of f, we have

  $$\mathsf{A}_i \neq f(\mathsf{r}_i^0, \mathsf{r}_i^1, \mathsf{h}) = \mathsf{k}_i^0 - \mathsf{k}_i^1 - \mathsf{B} \cdot \mathsf{h} = \mathsf{k}_i^0 - (\mathsf{T}_i - \mathsf{A}_i - \mathsf{B} \cdot \mathsf{h}) - \mathsf{B} \cdot \mathsf{h} \quad \Longleftrightarrow \quad \mathsf{k}_i^0 \neq \mathsf{T}_i.$$

  Furthermore, since pk^0 is not erroneous, we have $\mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i^0) = \mathsf{k}_i^0$. These together imply that we must have $\mathsf{s}'_i = 1$.

The above shows that $\mathsf{s}'_i = \mathsf{s}_i$ holds for all $i \in [n]$.

Putting everything together, except for a probability at most $\epsilon + n \cdot 2^{-\lambda}$ over $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathsf{KKG}_{\mathsf{cca}}(1^\lambda)$, there exists no randomness R satisfying $\mathsf{Encap}_{\mathsf{cca}}(\mathsf{PK}; \mathsf{R}) = (\mathsf{CT}, \mathsf{k})$ and $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) \neq \mathsf{k}$ simultaneously. $\square$ (**Theorem 1**)

4.3 Proof of IND-CCA Security (Proof of Theorem 2)

Let $\epsilon : \mathbb{N} \to [0,1]$ be such that KEM is ϵ-almost-all-keys correct. Let $\mathcal{A}$ be any PPT adversary that attacks the IND-CCA security of $\mathsf{KEM}_{\mathsf{cca}}$ and makes $q_{\mathsf{dec}} = q_{\mathsf{dec}}(\lambda) > 0$ decapsulation queries. We will show that for this $\mathcal{A}$, there exist PPT adversaries $\mathcal{B}_{\mathsf{tcr}}$, $\{\mathcal{B}_{\mathsf{cpa}}^j\}_{j\in[4]}$, $\mathcal{B}'_{\mathsf{cpa}}$, and $\mathcal{B}_{\mathsf{kdm}}$ (which makes a single

KDM-encryption query) satisfying

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{cca}}_{\mathsf{KEM}_{\mathsf{cca}},\mathcal{A}}(\lambda) \leq \\ &\quad 2\cdot \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{Hash},\mathcal{B}_{\mathsf{tcr}}}(\lambda) + 2\cdot \sum_{j\in[4]} \mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},n,\mathcal{B}^{j}_{\mathsf{cpa}}}(\lambda) + 2q_{\mathsf{dec}}\cdot \mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},n,\mathcal{B}'_{\mathsf{cpa}}}(\lambda) \\ &\qquad + 2\cdot \mathsf{Adv}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{P},\mathcal{B}_{\mathsf{kdm}}}(\lambda) + 8\epsilon + n\cdot 2^{-\lambda+3} + n(q_{\mathsf{dec}}+1)\cdot 2^{-4\lambda+1}. \quad (1)\end{aligned}$$

This is negligible by our assumption, and thus will prove the theorem.

Our proof is via a sequence of games argument using the following six games.

Game 1: This is the IND-CCA experiment $\mathsf{Expt}^{\mathsf{cca}}_{\mathsf{KEM}_{\mathsf{cca}},\mathcal{A}}(\lambda)$. However, for making it easier to describe the subsequent games, we change the ordering of the operations for how the key pair $(\mathsf{PK},\mathsf{SK})$ and the challenge ciphertext/session-key pair $(\mathsf{CT}^*,\mathsf{k}^*_b)$ are generated so that the distribution of $(\mathsf{PK},\mathsf{SK},\mathsf{CT}^*,\mathsf{k}^*_b)$ is identical to that in the original IND-CCA experiment.

Specifically, the description of the game is as follows:

- Generate $\mathsf{PK} = (\mathsf{pk}^0,\mathsf{pk}^1,(\mathsf{A}_i)_{i\in[n]},\mathsf{B},\mathsf{hk})$, $\mathsf{SK} = (\mathsf{sk}^0,\mathsf{PK})$, and $\mathsf{CT}^* = ((\mathsf{ct}^{*0}_i,\mathsf{ct}^{*1}_i,\mathsf{T}^*_i)_{i\in[n]},\mathsf{ct}^*_{\mathsf{SKE}})$ as follows:
  1. Compute $(\mathsf{pk}^v,\mathsf{sk}^v) \leftarrow \mathsf{KKG}(1^\lambda)$ for $v\in\{0,1\}$, and pick $\mathsf{B} \xleftarrow{\mathsf{r}} \{0,1\}^{4\lambda}$.
  2. Compute $\mathsf{s}^* = (\mathsf{s}^*_1,\dots,\mathsf{s}^*_n) \leftarrow \mathsf{K}(1^\lambda)$, and pick $\mathsf{r}^{*0}_1,\dots,\mathsf{r}^{*0}_n,\mathsf{r}^{*1}_1,\dots,\mathsf{r}^{*1}_n \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$ and $\mathsf{k}^*_1 \xleftarrow{\mathsf{r}} \{0,1\}^\ell$.
  3. Compute $\mathsf{ct}^*_{\mathsf{SKE}} \leftarrow \mathsf{E}(\mathsf{s}^*,(\mathsf{r}^{*(\mathsf{s}^*_i)}_i)_{i\in[n]}\|\mathsf{k}^*_1)$.
  4. Compute $(\mathsf{ct}^{*v}_i,\mathsf{k}^{*v}_i) \leftarrow \mathsf{Encap}(\mathsf{pk}^v;\mathsf{r}^{*v}_i)$ for every $(i,v)\in[n]\times\{0,1\}$.
  5. Compute $\mathsf{hk} \leftarrow \mathsf{HKG}(1^\lambda)$ and $\mathsf{h}^* \leftarrow \mathsf{H}(\mathsf{hk},(\mathsf{ct}^{*0}_i,\mathsf{ct}^{*1}_i)_{i\in[n]}\|\mathsf{ct}^*_{\mathsf{SKE}})$.
  6. Pick $\mathsf{A}_1,\dots,\mathsf{A}_n \xleftarrow{\mathsf{r}} \{0,1\}^{4\lambda}$.
  7. Compute $\mathsf{T}^*_i \leftarrow \mathsf{k}^{*(\mathsf{s}^*_i)}_i + \mathsf{s}^*_i\cdot(\mathsf{A}_i + \mathsf{B}\cdot\mathsf{h}^*)$ for every $i\in[n]$.
  8. Set $\mathsf{PK} \leftarrow (\mathsf{pk}^0,\mathsf{pk}^1,(\mathsf{A}_i)_{i\in[n]},\mathsf{B},\mathsf{hk})$, $\mathsf{SK} \leftarrow (\mathsf{sk}^0,\mathsf{PK})$, and $\mathsf{CT}^* \leftarrow ((\mathsf{ct}^{*0}_i,\mathsf{ct}^{*1}_i,\mathsf{T}^*_i)_{i\in[n]},\mathsf{ct}^*_{\mathsf{SKE}})$.
- Then, pick the random session-key $\mathsf{k}^*_0 \xleftarrow{\mathsf{r}} \{0,1\}^\ell$ and the challenge bit $b \xleftarrow{\mathsf{r}} \{0,1\}$, and run $\mathcal{A}(\mathsf{PK},\mathsf{CT}^*,\mathsf{k}^*_b)$. From here on, $\mathcal{A}$ may start making decapsulation queries.
- Decapsulation queries $\mathsf{CT} = ((\mathsf{ct}^0_i,\mathsf{ct}^1_i,\mathsf{T}_i)_{i\in[n]},\mathsf{ct}_{\mathsf{SKE}})$ are answered as follows: First, compute $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk},(\mathsf{ct}^0_i,\mathsf{ct}^1_i)_{i\in[n]}\|\mathsf{ct}_{\mathsf{SKE}})$. Next, compute $\mathsf{s}_i \leftarrow 1-(\mathsf{Decap}(\mathsf{sk}^0,\mathsf{ct}^0_i) \overset{?}{=} \mathsf{T}_i)$ for every $i\in[n]$, and set $\mathsf{s} \leftarrow (\mathsf{s}_1,\dots,\mathsf{s}_n)$. Then, compute $\mathsf{m} \leftarrow \mathsf{D}(\mathsf{s},\mathsf{ct}_{\mathsf{SKE}})$ and parse m as $((\mathsf{r}^{\mathsf{s}_i}_i)_{i\in[n]},\mathsf{k}) \in (\{0,1\}^\lambda)^n \times \{0,1\}^\ell$. Finally, if $\mathsf{Encap}(\mathsf{pk}^{\mathsf{s}_i};\mathsf{r}^{\mathsf{s}_i}_i) = (\mathsf{ct}^{\mathsf{s}_i}_i,\mathsf{T}_i - \mathsf{s}_i\cdot(\mathsf{A}_i+\mathsf{B}\cdot\mathsf{h}))$ holds for all $i\in[n]$, then return k to $\mathcal{A}$. Otherwise, return $\perp$ to $\mathcal{A}$.
- At some point, $\mathcal{A}$ terminates with output $b'\in\{0,1\}$.

For convenience, in the following we will use the following sets:

$$\mathcal{S}_{\mathsf{zero}} := \left\{\, j\in[n] \;\middle|\; \mathsf{s}^*_j = 0 \,\right\} \quad \text{and} \quad \mathcal{S}_{\mathsf{one}} := \left\{\, j\in[n] \;\middle|\; \mathsf{s}^*_j = 1 \,\right\} = [n]\setminus\mathcal{S}_{\mathsf{zero}}.$$

Game 2: Same as Game 1, except for an additional rejection rule in the decapsulation oracle. Specifically, in this game, if $\mathcal{A}$'s decapsulation query $\mathsf{CT} = ((\mathsf{ct}^0_i,\mathsf{ct}^1_i,\mathsf{T}_i)_{i\in[n]},\mathsf{ct}_{\mathsf{SKE}})$ satisfies $\mathsf{h} = \mathsf{H}(\mathsf{hk},(\mathsf{ct}^0_i,\mathsf{ct}^1_i)_{i\in[n]}\|\mathsf{ct}_{\mathsf{SKE}}) = \mathsf{h}^*$, then the decapsulation oracle immediately returns $\perp$ to $\mathcal{A}$.

Game 3: Same as Game 2, except for how A_i's for the positions $i \in \mathcal{S}_{\mathsf{zero}}$ are generated. Specifically, in this game, A_i for every position $i \in \mathcal{S}_{\mathsf{zero}}$ is generated by

$$\mathsf{A}_i \leftarrow \mathsf{k}_i^{*0} - \mathsf{k}_i^{*1} - \mathsf{B} \cdot \mathsf{h}^*. \tag{2}$$

(At this point, A_i's for the remaining positions $i \in \mathcal{S}_{\mathsf{one}}$ are unchanged.)

Game 4: Same as Game 3, except for the behavior of the decapsulation oracle. Specifically, for answering $\mathcal{A}$'s decapsulation queries $\mathsf{CT} = ((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i \in [n]}, \mathsf{ct}_{\mathsf{SKE}})$, the oracle in this game first computes $\mathsf{h} = \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i \in [n]} \| \mathsf{ct}_{\mathsf{SKE}})$, and returns $\perp$ to $\mathcal{A}$ if $\mathsf{h} = \mathsf{h}^*$. (This rejection rule is the same as in Game 3.) Otherwise, the oracle uses the "alternative decapsulation algorithm" $\mathsf{AltDecap}$ and the "alternative secret key" SK' defined below for computing the decapsulation result k returned to $\mathcal{A}$.
$\mathsf{AltDecap}$ takes $\mathsf{SK}' := (\mathsf{sk}^1, \mathsf{PK})$ and CT as input, and proceeds identically to $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT})$, except that the "find step" (i.e. the step for computing s_i's) is replaced with the following procedure:

$$\forall i \in [n]: \quad \mathsf{s}_i \leftarrow \Big(\mathsf{Decap}(\mathsf{sk}^1, \mathsf{ct}_i^1) \overset{?}{=} \mathsf{T}_i - \mathsf{A}_i - \mathsf{B} \cdot \mathsf{h} \Big)$$
$$= \begin{cases} 1 & \text{if } \mathsf{Decap}(\mathsf{sk}^1, \mathsf{ct}_i^1) = \mathsf{T}_i - \mathsf{A}_i - \mathsf{B} \cdot \mathsf{h} \\ 0 & \text{otherwise} \end{cases}.$$

Note that due to this change, the decapsulation oracle answers $\mathcal{A}$'s queries without using sk^0.

Game 5: Same as Game 4, except for how A_i's for the positions $i \in \mathcal{S}_{\mathsf{one}}$ are generated. Specifically, in this game, A_i for $i \in \mathcal{S}_{\mathsf{one}}$ is also generated as in Equation 2.
Note that due to this change, all of $(\mathsf{A}_i)_{i \in [n]}$ are generated as in Equation 2. Furthermore, $\mathsf{T}_i^* = \mathsf{k}_i^{*0}$ holds for every $i \in [n]$, no matter whether $\mathsf{s}_i^* = 0$ or $\mathsf{s}_i^* = 1$. Indeed, this is the case for the positions $i \in \mathcal{S}_{\mathsf{zero}}$ by design. For the positions $i \in \mathcal{S}_{\mathsf{one}}$, we have

$$\mathsf{T}_i^* = \mathsf{k}_i^{*1} + \mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}^* = \mathsf{k}_i^{*1} + (\mathsf{k}_i^{*0} - \mathsf{k}_i^{*1} - \mathsf{B} \cdot \mathsf{h}^*) + \mathsf{B} \cdot \mathsf{h}^* = \mathsf{k}_i^{*0}.$$

Hence, in this game, values dependent on s^* appear only in the plaintext of $\mathsf{ct}_{\mathsf{SKE}}^*$ (i.e. $(\mathsf{r}_i^{*(\mathsf{s}_i^*)})_{i \in [n]} \| \mathsf{k}_1^*$).

Game 6: Same as Game 5, except that the information of the challenge bit b is erased from the SKE ciphertext $\mathsf{ct}_{\mathsf{SKE}}^*$. Specifically, in this game, $\mathsf{ct}_{\mathsf{SKE}}^*$ in the challenge ciphertext CT^* is generated by $\mathsf{ct}_{\mathsf{SKE}}^* \leftarrow \mathsf{E}(\mathsf{s}^*, 0^{n \cdot \lambda + \ell})$, instead of $\mathsf{ct}_{\mathsf{SKE}}^* \leftarrow \mathsf{E}(\mathsf{s}^*, (\mathsf{r}_i^{*(\mathsf{s}_i^*)})_{i \in [n]} \| \mathsf{k}_1^*)$.

For $j \in [6]$, let $\mathtt{SUC}_j$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit (i.e. $b' = b$ occurs) in Game j. By definition, we have $\mathsf{Adv}_{\mathsf{KEM}_{\mathsf{cca}}, \mathcal{A}}^{\mathsf{cca}}(\lambda) = 2 \cdot |\Pr[\mathtt{SUC}_1] - 1/2|$. Thus, the triangle inequality implies

$$\mathsf{Adv}_{\mathsf{KEM}_{\mathsf{cca}}, \mathcal{A}}^{\mathsf{cca}}(\lambda) \leq 2 \cdot \left(\sum_{j \in [5]} |\Pr[\mathtt{SUC}_j] - \Pr[\mathtt{SUC}_{j+1}]| + \left| \Pr[\mathtt{SUC}_6] - \frac{1}{2} \right| \right). \tag{3}$$

In the following, we show how the terms appearing in Equation 3 are bounded.

Lemma 1. *There exist PPT adversaries $\mathcal{B}_{\mathsf{tcr}}$, $\{\mathcal{B}^j_{\mathsf{cpa}}\}_{j\in[2]}$, and $\mathcal{B}'_{\mathsf{cpa}}$ satisfying*

$$\begin{aligned}|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \le \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{Hash},\mathcal{B}_{\mathsf{tcr}}}(\lambda) + \sum_{j\in[2]} \mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},n,\mathcal{B}^j_{\mathsf{cpa}}}(\lambda) \\ + q_{\mathsf{dec}} \cdot \mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},n,\mathcal{B}'_{\mathsf{cpa}}}(\lambda) + 3\epsilon + n \cdot 2^{-\lambda+1} + n(q_{\mathsf{dec}}+1)\cdot 2^{-4\lambda}. \quad (4)\end{aligned}$$

Due to the space limitation, we give the formal proof of Lemma 1 in the full version of this paper [29]. The proof relies on the target collision resistance of the underlying keyed hash function Hash, and uses a deferred analysis (up to Game 4) with (slight variants of) the arguments used in the proofs of Lemmas 2 to 4 stated below.

Lemma 2. *There exists a PPT adversary $\mathcal{B}^3_{\mathsf{cpa}}$ such that* $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| = \mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},n,\mathcal{B}^3_{\mathsf{cpa}}}(\lambda)$.

In Games 2 and 3, sk^1 is not used. Moreover, r^{*1}_i used to generate ct^{*1}_i, is not encrypted into $\mathsf{ct}^*_{\mathsf{SKE}}$ for $i \in \mathcal{S}_{\mathsf{zero}}$. Thus, from $\mathcal{A}$, we can construct a PPT adversary $\mathcal{B}^3_{\mathsf{cpa}}$ that attacks the IND-CPA security of KEM under the key pk^1 with the advantage stated in the lemma. For the formal proof, see the full version [29].

Lemma 3. $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \le 2\epsilon + n\cdot 2^{-\lambda+1}$ *holds.*

Proof of Lemma 3. Note that Game 3 and Game 4 proceed identically unless $\mathcal{A}$ makes a decapsulation query $\mathsf{CT} = ((\mathsf{ct}^0_i, \mathsf{ct}^1_i, \mathsf{T}_i)_{i\in[n]}, \mathsf{ct}_{\mathsf{SKE}})$ such that $\mathsf{h} = \mathsf{H}(\mathsf{hk}, (\mathsf{ct}^0_i, \mathsf{ct}^1_i)_{i\in[n]} \| \mathsf{ct}_{\mathsf{SKE}}) \ne \mathsf{h}^*$ and $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) \ne \mathsf{AltDecap}(\mathsf{SK}', \mathsf{CT})$ hold simultaneously. We call such a decapsulation query *bad*. In the following, we will show that if PK is not "bad" in the sense specified below, a bad decapsulation query does not exist in Game 3 and Game 4, and the probability that PK becomes bad is bounded by $2\epsilon + n\cdot 2^{-\lambda+1}$. This will prove the lemma.

Fix the following values in Game 3:

- $(\mathsf{pk}^0, \mathsf{sk}^0), (\mathsf{pk}^1, \mathsf{sk}^1) \in \mathsf{Sup}(\mathsf{KKG}(1^\lambda))$ such that pk^0 and pk^1 are not erroneous, and $\mathsf{hk} \in \mathsf{Sup}(\mathsf{HKG}(1^\lambda))$.
- $\mathsf{s}^* = (\mathsf{s}^*_1, \dots, \mathsf{s}^*_n) \in \mathsf{Sup}(\mathsf{K}(1^\lambda))$, $\mathsf{r}^{*0}_1, \dots, \mathsf{r}^{*0}_n, \mathsf{r}^{*1}_1, \dots, \mathsf{r}^{*1}_n \in \{0,1\}^\lambda$, $\mathsf{k}^*_1 \in \{0,1\}^\ell$, and $\mathsf{r}^*_{\mathsf{SKE}} \in \mathcal{R}_{\mathsf{SKE}}$.
- $(\mathsf{ct}^{*v}_i, \mathsf{k}^{*v}_i) = \mathsf{Encap}(\mathsf{pk}^v; \mathsf{r}^{*v}_i)$ for all $(i, v) \in [n] \times \{0,1\}$.
- $\mathsf{ct}^*_{\mathsf{SKE}} = \mathsf{E}(\mathsf{s}^*, (\mathsf{r}^{*(\mathsf{s}^*_i)}_i)_{i\in[n]} \| \mathsf{k}^*_1; \mathsf{r}^*_{\mathsf{SKE}})$ and $\mathsf{h}^* = \mathsf{H}(\mathsf{hk}, (\mathsf{ct}^{*0}_i, \mathsf{ct}^{*1}_i)_{i\in[n]} \| \mathsf{ct}^*_{\mathsf{SKE}})$.

Let $\mathcal{C}$ be the ciphertext space of KEM. To define the notion of "badness" for a public key, we introduce two types of functions based on the above fixed values.

- For each $i \in \mathcal{S}_{\mathsf{zero}}$ and $v \in \{0,1\}$, we define the function $\widehat{g}_{i,v} : \{0,1\}^\lambda \times \mathcal{C} \times (\{0,1\}^\lambda \setminus \{\mathsf{h}^*\}) \to \{0,1\}^{4\lambda} \cup \{\bot\}$ by

$$\widehat{g}_{i,v}(\mathsf{r}, \mathsf{ct}', \mathsf{h}) : \left[\begin{array}{l}(\mathsf{ct}, \mathsf{k}) \leftarrow \mathsf{Encap}(\mathsf{pk}^v; \mathsf{r});\ \mathsf{k}' \leftarrow \mathsf{Decap}(\mathsf{sk}^{1-v}, \mathsf{ct}');\\ \text{If } \mathsf{k}' = \bot \text{ then return } \bot \text{ else return } \frac{(\mathsf{k}-\mathsf{k}')\cdot(-1)^v - \mathsf{k}^{*0}_i + \mathsf{k}^{*1}_i}{\mathsf{h}-\mathsf{h}^*}\end{array}\right].$$

We say that a string $\mathsf{B} \in \{0,1\}^{4\lambda}$ is *bad* if B belongs to the image of $\widehat{g}_{i,v}$ for some $(i,v) \in \mathcal{S}_{\mathsf{zero}} \times \{0,1\}$. Due to the property that the image size of $\mathsf{Decap}(\mathsf{sk}^{1-v}, \cdot)$ is bounded by 2^{λ}, the image size of $\widehat{g}_{i,v}$ (excluding $\perp$) is at most $2^{3\lambda}$ for every $i \in \mathcal{S}_{\mathsf{zero}}$ and $v \in \{0,1\}$. Hence, when choosing $\mathsf{B} \xleftarrow{\mathsf{r}} \{0,1\}^{4\lambda}$, the probability that B is bad is at most $|\mathcal{S}_{\mathsf{zero}}| \cdot 2 \cdot \frac{2^{3\lambda}}{2^{4\lambda}} = |\mathcal{S}_{\mathsf{zero}}| \cdot 2^{-\lambda+1}$.

– For each $\mathsf{B}' \in \{0,1\}^{4\lambda}$ and $v \in \{0,1\}$, we define the function $g_{\mathsf{B}',v} : \{0,1\}^{\lambda} \times \mathcal{C} \times \{0,1\}^{\lambda} \to \{0,1\}^{4\lambda} \cup \{\perp\}$ by

$$g_{\mathsf{B}',v}(\mathsf{r}, \mathsf{ct}', \mathsf{h}) : \begin{bmatrix} (\mathsf{ct}, \mathsf{k}) \leftarrow \mathsf{Encap}(\mathsf{pk}^v; \mathsf{r});\ \ \mathsf{k}' \leftarrow \mathsf{Decap}(\mathsf{sk}^{1-v}, \mathsf{ct}'); \\ \text{If } \mathsf{k}' = \perp \text{ then return } \perp \text{ else return } (\mathsf{k} - \mathsf{k}') \cdot (-1)^v - \mathsf{B}' \cdot \mathsf{h} \end{bmatrix}.$$

For each $\mathsf{B}' \in \{0,1\}^{4\lambda}$, we say that a string $\mathsf{A}' \in \{0,1\}^{4\lambda}$ is *bad with respect to* B' if A' belongs to the image of $g_{\mathsf{B}',0}$ or that of $g_{\mathsf{B}',1}$. Again, due to the property that the image size of $\mathsf{Decap}(\mathsf{sk}^{1-v}, \cdot)$ is bounded by 2^{λ}, the image size of $g_{\mathsf{B},v}$ (excluding $\perp$) is at most $2^{3\lambda}$ for every $\mathsf{B}' \in \{0,1\}^{4\lambda}$ and $v \in \{0,1\}$. Hence, for any fixed $\mathsf{B}' \in \{0,1\}^{4\lambda}$, when choosing $\mathsf{A}_i \xleftarrow{\mathsf{r}} \{0,1\}^{4\lambda}$ for all $i \in \mathcal{S}_{\mathsf{one}}$, the probability that some of $\{\mathsf{A}_i\}_{i \in \mathcal{S}_{\mathsf{one}}}$ is bad with respect to B' is at most $|\mathcal{S}_{\mathsf{one}}| \cdot 2 \cdot \frac{2^{3\lambda}}{2^{4\lambda}} = |\mathcal{S}_{\mathsf{one}}| \cdot 2^{-\lambda+1}$.

We say that a public key PK generated in Game 3 is *bad* if (1) either pk^0 or pk^1 is erroneous, or (2) either B is bad or A_i for some $i \in \mathcal{S}_{\mathsf{one}}$ is bad with respect to B. By the union bound, the probability that PK is bad in Game 3 is bounded by $2\epsilon + |\mathcal{S}_{\mathsf{zero}}| \cdot 2^{-\lambda+1} + |\mathcal{S}_{\mathsf{one}}| \cdot 2^{-\lambda+1} = 2\epsilon + n \cdot 2^{-\lambda+1}$.

To complete the proof, below we show that if $\mathsf{PK} = (\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i)_{i \in [n]}, \mathsf{B}, \mathsf{hk})$ is not bad, then for any ciphertext $\mathsf{CT} = ((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i \in [n]}, \mathsf{ct}_{\mathsf{SKE}})$ such that $\mathsf{h} = \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i \in [n]} \| \mathsf{ct}_{\mathsf{SKE}}) \neq \mathsf{h}^*$, we always have $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) = \mathsf{AltDecap}(\mathsf{SK}', \mathsf{CT})$.

Let $\mathsf{CT} = ((\mathsf{ct}_i^0, \mathsf{ct}_i^1, \mathsf{T}_i)_{i \in [n]}, \mathsf{ct}_{\mathsf{SKE}})$ be an arbitrary ciphertext satisfying $\mathsf{h} = \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i^0, \mathsf{ct}_i^1)_{i \in [n]} \| \mathsf{ct}_{\mathsf{SKE}}) \neq \mathsf{h}^*$. For each $i \in [n]$, define

$$\mathsf{s}_i := 1 - \Big(\mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i^0) \overset{?}{=} \mathsf{T}_i \Big), \qquad \text{and}$$

$$\mathsf{s}'_i := \Big(\mathsf{Decap}(\mathsf{sk}^1, \mathsf{ct}_i^1) \overset{?}{=} \mathsf{T}_i - \mathsf{A}_i - \mathsf{B} \cdot \mathsf{h} \Big).$$

We consider two cases and show that $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) = \mathsf{AltDecap}(\mathsf{SK}', \mathsf{CT})$ holds in either case.

– **Case 1: For all positions $i \in [n]$, there exists a pair $(\mathsf{r}, v) \in \{0,1\}^{\lambda} \times \{0,1\}$ satisfying $\mathsf{Encap}(\mathsf{pk}^v; \mathsf{r}) = (\mathsf{ct}_i^v, \mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}))$.**
In this case, we show that $\mathsf{s}_i = \mathsf{s}'_i$ holds for all $i \in [n]$. This in turn implies that the output of $\mathsf{Decap}_{\mathsf{cca}}$ and that of $\mathsf{AltDecap}$ agree since these algorithms proceed identically after they respectively compute s.
Fix $i \in [n]$. The condition of this case directly implies $\mathsf{Decap}(\mathsf{sk}^v, \mathsf{ct}_i^v) = \mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h})$. This in turn implies that if $v = 0$ then we have $\mathsf{s}_i = 0$, while if $v = 1$ then we have $\mathsf{s}'_i = 1$. In the following, we will show that

$$\mathsf{k}' := \mathsf{Decap}(\mathsf{sk}^{1-v}, \mathsf{ct}_i^{1-v}) \neq \mathsf{T}_i - (1 - v) \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) \tag{5}$$

holds, which implies that if $v = 0$ then we have $\mathsf{s}'_i = 0$, while if $v = 1$ then we have $\mathsf{s}_i = 1$. Hence, combined together, we will obtain the desired conclusion $\mathsf{s}_i = \mathsf{s}'_i$ (regardless of the value of v). Also, if $\mathsf{k}' = \perp$, then Equation 5 is obviously satisfied. Thus, below we consider the case $\mathsf{k}' \neq \perp$.
The argument for showing Equation 5 differs depending on whether $i \in \mathcal{S}_{\mathsf{zero}}$ or $i \in \mathcal{S}_{\mathsf{one}}$. If $i \in \mathcal{S}_{\mathsf{zero}}$, then since B is not bad, it is not in the image of $\widehat{g}_{i,v}$. Hence, we have

$$\begin{aligned} & \mathsf{B} \neq \widehat{g}_{i,v}(\mathsf{r}, \mathsf{ct}_i^{1-v}, \mathsf{h}) = \frac{\Big(\mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) - \mathsf{k}'\Big) \cdot (-1)^v - \mathsf{k}_i^{*0} + \mathsf{k}_i^{*1}}{\mathsf{h} - \mathsf{h}^*} \\ \Longleftrightarrow \quad & \mathsf{k}' \neq \mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) - (-1)^v \cdot \Big((\mathsf{k}_i^{*0} - \mathsf{k}_i^{*1} - \mathsf{B} \cdot \mathsf{h}^*) + \mathsf{B} \cdot \mathsf{h}\Big) \\ & \overset{(*)}{=} \mathsf{T}_i - \Big(v + (-1)^v\Big) \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) \overset{(**)}{=} \mathsf{T}_i - (1 - v) \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}), \end{aligned}$$

where the equality (*) uses $\mathsf{A}_i = \mathsf{k}_i^{*0} - \mathsf{k}_i^{*1} - \mathsf{B} \cdot \mathsf{h}^*$, which is how A_i is generated for the positions $i \in \mathcal{S}_{\mathsf{zero}}$ in Game 3; The equality (**) is due to $v + (-1)^v = 1 - v$ for $v \in \{0, 1\}$.
Similarly, if $i \in \mathcal{S}_{\mathsf{one}}$, then since A_i is not bad with respect to B, it is not in the image of $g_{\mathsf{B},v}$. Hence, we have

$$\begin{aligned} & \mathsf{A}_i \neq g_{\mathsf{B},v}(\mathsf{r}, \mathsf{ct}_i^{1-v}, \mathsf{h}) = \Big(\mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) - \mathsf{k}'\Big) \cdot (-1)^v - \mathsf{B} \cdot \mathsf{h} \\ \Longleftrightarrow \quad & \mathsf{k}' \neq \mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) - (-1)^v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) \\ & = \mathsf{T}_i - \Big(v + (-1)^v\Big) \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}) = \mathsf{T}_i - (1 - v) \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}), \end{aligned}$$

where the last equality is again due to $v + (-1)^v = 1 - v$ for $v \in \{0, 1\}$.
We have seen that $\mathsf{Decap}(\mathsf{sk}^{1-v}, \mathsf{ct}_i^{1-v}) \neq \mathsf{T}_i - (1 - v) \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h})$ holds regardless of whether $i \in \mathcal{S}_{\mathsf{zero}}$ or $i \in \mathcal{S}_{\mathsf{one}}$, as required. Hence, as mentioned earlier, $\mathsf{s}_i = \mathsf{s}'_i$ holds for all $i \in [n]$, and consequently we have $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) = \mathsf{AltDecap}(\mathsf{SK}', \mathsf{CT})$.

– **Case 2: There exists a position $i \in [n]$ for which there exists no pair $(\mathsf{r}, v) \in \{0, 1\}^\lambda \times \{0, 1\}$ satisfying $\mathsf{Encap}(\mathsf{pk}^v; \mathsf{r}) = (\mathsf{ct}_i^v, \mathsf{T}_i - v \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}))$.** In this case, both $\mathsf{Decap}_{\mathsf{cca}}$ and $\mathsf{AltDecap}$ return $\perp$. Indeed, the condition of this case implies that there exists a position $i \in [n]$ for which there exists no $\mathsf{r} \in \{0, 1\}^\lambda$ satisfying $\mathsf{Encap}(\mathsf{pk}^{\mathsf{s}_i}; \mathsf{r}) = (\mathsf{ct}_i^{\mathsf{s}_i}, \mathsf{T}_i - \mathsf{s}_i \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h}))$. Hence, the validity check done in the last step of $\mathsf{Decap}_{\mathsf{cca}}$ cannot be satisfied at the position i, and thus $\mathsf{Decap}_{\mathsf{cca}}$ outputs $\perp$. Exactly the same argument applies to $\mathsf{AltDecap}$, and thus it also outputs $\perp$. Hence, in this case we have $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) = \mathsf{AltDecap}(\mathsf{SK}', \mathsf{CT}) = \perp$.

As seen above, if PK is not bad, then for any CT with $\mathsf{h} \neq \mathsf{h}^*$, we have $\mathsf{Decap}_{\mathsf{cca}}(\mathsf{SK}, \mathsf{CT}) = \mathsf{AltDecap}(\mathsf{SK}', \mathsf{CT})$, as desired. $\square$ (**Lemma 3**)

Lemma 4. *There exists a PPT adversary $\mathcal{B}_{\mathsf{cpa}}^4$ such that* $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| = \mathsf{Adv}^{\mathsf{mcpa}}_{\mathsf{KEM},n,\mathcal{B}_{\mathsf{cpa}}^4}(\lambda)$.

With a similar reason to for Lemma 2 above, from $\mathcal{A}$, we can construct a PPT adversary $\mathcal{B}^4_{\mathsf{cpa}}$ that attacks the IND-CPA security of KEM under the key pk^0 with the advantage stated in the lemma. For the formal proof, see the full version [29].

Lemma 5. *There exists a PPT adversary* $\mathcal{B}_{\mathsf{kdm}}$ *that makes a single KDM-encryption query and satisfies* $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = \mathsf{Adv}^{\mathsf{kdm}}_{\mathsf{SKE},\mathcal{P},\mathcal{B}_{\mathsf{kdm}}}(\lambda)$.

In Games 5 and 6, s^* is used only when generating $\mathsf{ct}^*_{\mathsf{SKE}}$. Furthermore, we can regard the message $(\mathsf{r}_i^{*(\mathsf{s}_i^*)})_{i\in[n]}\|\mathsf{k}_1^*$ encrypted in $\mathsf{ct}^*_{\mathsf{SKE}}$ as an output of a projection function of s^*. Thus, from $\mathcal{A}$, we can straightforwardly construct a PPT adversary $\mathcal{B}_{\mathsf{kdm}}$ that attacks the one-time $\mathcal{P}$-KDM security of SKE with the advantage stated in the lemma. For the formal proof, see the full version [29].

Lemma 6. $\Pr[\mathtt{SUC}_6] = 1/2$ *holds.*

Proof of Lemma 6. This lemma is true because in Game 6, the information of the challenge bit b is completely erased from $\mathcal{A}$'s view. □ (**Lemma 6**)

Due to Lemmas 1 to 6 and Equation 3, we can conclude that there exist PPT adversaries $\mathcal{B}_{\mathsf{tcr}}$, $\{\mathcal{B}^j_{\mathsf{cpa}}\}_{j\in[4]}$, $\mathcal{B}'_{\mathsf{cpa}}$, and $\mathcal{B}_{\mathsf{kdm}}$ (that makes a single KDM-encryption query) satisfying Equation 1, as desired. □ (**Theorem 2**)

5 Impossibility of Shielding Black-Box Constructions

Gertner et al. [21] showed that there exists no shielding black-box construction of an IND-CCA1 secure PKE scheme from an IND-CPA secure one. Recall that a shielding black-box construction of a PKE scheme $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ from another PKE scheme $\mathsf{pke} = (\mathsf{kg}, \mathsf{enc}, \mathsf{dec})$ is such that the decryption algorithm Dec in PKE does *not* use the encryption algorithm enc of pke. Put differently, we have $\mathsf{PKE}^{\mathsf{pke}} = (\mathsf{KG}^{\mathsf{kg},\mathsf{enc},\mathsf{dec}}, \mathsf{Enc}^{\mathsf{kg},\mathsf{enc},\mathsf{dec}}, \mathsf{Dec}^{\mathsf{kg},\mathsf{dec}})$.

In this section, we extend Gertner et al.'s result and show the following result.

Theorem 4. *There exists no shielding black-box construction of an IND-CCA1 secure PKE scheme from a* $\mathcal{P}$*-KDM secure PKE scheme.*

This theorem is proved as a corollary of Theorems 5 and 6 stated below.

We emphasize that this result does not contradict our result in Section 4.1 (in particular, Corollary 2), because our construction $\mathsf{KEM}_{\mathsf{cca}}$ is a non-shielding black-box construction in which the decapsulation algorithm $\mathsf{Decap}_{\mathsf{cca}}$ uses the encapsulation algorithm Encap of the underlying IND-CPA secure KEM.

We also note that our result seems incomparable to a similar result by Hajiabadi and Kapron [24], who showed that a PKE scheme satisfying a form of *randomness-dependent-message (RDM) security* is a primitive from which a non-shielding black-box construction of an IND-CCA secure PKE scheme is possible while shielding black-box constructions of an IND-CCA1 secure PKE scheme are impossible. (We note that they used a tailored definition of RDM security that is different from the original definition by Birrell, Chung, Pass, and Telang [8].[7])

Our impossibility of shielding black-box constructions is shown based largely on the framework and technique of [21] and the technique of [24]. Informally, [21] defined a distribution $\mathbf{\Phi}$ of an oracle $\mathbf{O} = (\mathbf{O}_1, \mathbf{O}_2)$ such that $\mathbf{O}_1$ syntactically constitutes a PKE scheme, $\mathbf{O}_2$ is an attacker's "breaking" oracle, and they showed that the following two items hold with high probability over the choice of $\mathbf{O} = (\mathbf{O}_1, \mathbf{O}_2) \leftarrow \mathbf{\Phi}$:

1. $\mathbf{O}_1$ constitutes an IND-CPA secure PKE scheme against any computationally unbounded adversary $\mathcal{A}^{\mathbf{O}_1,\mathbf{O}_2}$ that makes polynomially many queries.
2. The IND-CCA1 security of any candidate shielding black-box construction $\mathsf{PKE}^{\mathbf{O}_1}$ is broken (with more than a constant advantage) by some computationally unbounded adversary $\mathcal{A}'^{\mathbf{O}_1,\mathbf{O}_2}$ with polynomially many queries.

These two items imply (via a standard argument used in black-box separation results) the impossibility of shielding black-box constructions of an IND-CCA1 secure PKE scheme from an IND-CPA secure one.

Since we use exactly the same distribution $\mathbf{\Phi}$ of oracles $\mathbf{O}$ used by Gertner et al., and the second item was already shown by them, for our result, we only need to prove an extension of the first item, namely, $\mathbf{O}_1$ constitutes a $\mathcal{P}$-KDM secure PKE scheme with high probability over the choice of $\mathbf{O} = (\mathbf{O}_1, \mathbf{O}_2) \leftarrow \mathbf{\Phi}$.

In the following, we first recall the definition of the distribution $\mathbf{\Phi}$ of oracles $\mathbf{O}$ used by Gertner et al., then state their result corresponding to the item 2 above. Finally, we state our result corresponding to the item 1 above.

Definition 5 (Oracle Distribution for Separation [21]). *Consider an oracle $\mathbf{O}$ consisting of the suboracles $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w}, \mathbf{u})$ that are defined for each length parameter $n \in \mathbb{N}$ and satisfy the following syntax*[8]*:*

$\mathbf{g} : \{0,1\}^n \to \{0,1\}^{3n}$**:** *This is an injective function. This oracle can be thought of as the key generation process that takes a secret key $\mathsf{sk} \in \{0,1\}^n$ as input and outputs a public key $\mathsf{pk} \in \{0,1\}^{3n}$.*

$\mathbf{e} : \{0,1\}^{3n} \times \{0,1\} \times \{0,1\}^n \to \{0,1\}^{3n}$**:** *For each $\mathsf{pk} \in \{0,1\}^{3n}$, $\mathsf{e}(\mathsf{pk}, \cdot, \cdot) : \{0,1\} \times \{0,1\}^n \to \{0,1\}^{3n}$ is an injective function. This oracle can be thought of as the encryption process that takes a public key $\mathsf{pk} \in \{0,1\}^{3n}$, a plaintext $\mathsf{m} \in \{0,1\}$, and a randomness $\mathsf{r} \in \{0,1\}^n$ as input, and outputs a ciphertext $\mathsf{ct} \in \{0,1\}^{3n}$.*

[7] Roughly speaking, RDM security used by Hajiabadi and Kapron requires that n ciphertexts encrypting the bit-decomposition of $\mathsf{r} = (\mathsf{r}_1, \dots, \mathsf{r}_n)$ are indistinguishable from n ciphertexts that all encrypt 0 even if they are all encrypted under the same random coin r itself. In the actual definition, an adversary is given multiple sets of the above n ciphertexts. This setting is somewhat unnatural in the usage of PKE, and a PKE scheme satisfying this security notion immediately implies a TDF with one-wayness under correlated products.

[8] Among $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w}, \mathbf{u})$, $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ (resp. $(\mathbf{w}, \mathbf{u})$) corresponds to $\mathbf{O}_1$ (resp. $\mathbf{O}_2$) in the above explanation.

$\mathbf{d} : \{0,1\}^n \times \{0,1\}^{3n} \to \{0,1,\perp\}$**:** *This oracle takes* $\mathsf{sk} \in \{0,1\}^n$ *and* $\mathsf{ct} \in \{0,1\}^{3n}$ *as input, and if there exists* $(\mathsf{pk},\mathsf{m},\mathsf{r}) \in \{0,1\}^{3n} \times \{0,1\} \times \{0,1\}^n$ *such that* $\mathsf{pk} = \mathbf{g}(\mathsf{sk})$ *and* $\mathsf{ct} = \mathbf{e}(\mathsf{pk},\mathsf{m},\mathsf{r})$*, then it outputs* m*. Otherwise, this oracle outputs* $\perp$*. This oracle can be thought of as the decryption process.*

$\mathbf{w} : \{0,1\}^{3n} \times \{0,1\}^n \to \{0,1\}^{3n\times n} \cup \{\perp\}$**:** *This oracle is associated with a "randomness deriving" function* $F_{\mathbf{w}} : \{0,1\}^{3n} \times \{0,1\}^n \to \{0,1\}^{n\times n}$*.*[9] *This oracle takes* $\mathsf{pk} \in \{0,1\}^{3n}$ *and an index* $z \in \{0,1\}^n$ *as input, and if there exists no* $\mathsf{sk} \in \{0,1\}^n$ *such that* $\mathsf{pk} = \mathbf{g}(\mathsf{sk})$*, then the oracle outputs* $\perp$*. Otherwise, let* $\mathsf{sk} = (\mathsf{s}_1,\ldots,\mathsf{s}_n) \in \{0,1\}^n$ *be such that* $\mathsf{pk} = \mathbf{g}(\mathsf{sk})$*. The oracle computes* $(\mathsf{r}_1,\ldots,\mathsf{r}_n) \leftarrow F_{\mathbf{w}}(\mathsf{pk},z)$*, and then* $\mathsf{ct}_i \leftarrow \mathbf{e}(\mathsf{pk},\mathsf{s}_i,\mathsf{r}_i)$ *for every* $i \in [n]$*. Finally, the oracle outputs* $(\mathsf{ct}_i)_{i\in[n]}$*. This oracle is a "weakening" oracle that helps breaking the IND-CCA1 security of any shielding construction.*

$\mathbf{u} : \{0,1\}^{3n} \times \{0,1\}^{3n} \to \{\top,\perp\}$**:** *This oracle takes* $\mathsf{pk} \in \{0,1\}^{3n}$ *and* $\mathsf{ct} \in \{0,1\}^{3n}$ *as input, and if there exists* $(\mathsf{sk},\mathsf{m},\mathsf{r}) \in \{0,1\}^n \times \{0,1\} \times \{0,1\}^n$ *such that* $\mathsf{pk} = \mathbf{g}(\mathsf{sk})$ *and* $\mathsf{ct} = \mathbf{e}(\mathsf{pk},\mathsf{m},\mathsf{r})$*, then the oracle outputs* $\top$*. Otherwise, the oracle outputs* $\perp$*. This oracle can be thought of as the validity checking process of a ciphertext* ct *with respect to a public key* pk*.*

We define the distribution $\mathbf{\Phi}$ *of an oracle* $\mathbf{O} = (\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u})$ *as follows: For each* $n \in \mathbb{N}$*, pick* $\mathbf{g}$*,* $\mathbf{e}$*, and* $F_{\mathbf{w}}$ *uniformly at random, and then define* $\mathbf{d}$*,* $\mathbf{w}$*, and* $\mathbf{u}$ *satisfying the above syntax.*[10]

Note that $(\mathbf{g},\mathbf{e},\mathbf{d})$ in $\mathbf{O}$ naturally constitutes a 1-bit PKE scheme. Gertner et al. [21] showed the following result, which states that with high probability over $\mathbf{O} \leftarrow \mathbf{\Phi}$, the IND-CCA1 security of any candidate shielding black-box construction from the PKE $(\mathbf{g},\mathbf{e},\mathbf{d})$ (defined in $\mathbf{O}$) is broken with more than a constant advantage by some adversary making polynomially many queries.

Theorem 5 (Corollary of Theorem 2 in [21]). *Let* $\mathsf{PKE} = (\mathsf{KG},\mathsf{Enc},\mathsf{Dec})$ *be a shielding construction of a 1-bit PKE scheme based on another 1-bit PKE scheme. For each* $\mathbf{O} = (\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u}) \in \mathsf{Sup}(\mathbf{\Phi})$*, let* $\mathsf{PKE}^{\mathbf{g},\mathbf{e},\mathbf{d}} := (\mathsf{KG}^{\mathbf{g},\mathbf{e},\mathbf{d}}, \mathsf{Enc}^{\mathbf{g},\mathbf{e},\mathbf{d}}, \mathsf{Dec}^{\mathbf{g},\mathbf{d}})$*. Then, there exists a computationally unbounded adversary* $\mathcal{A}$ *that makes at most polynomially many queries and satisfies the following for all sufficiently large* $\lambda \in \mathbb{N}$*:*

$$\Pr_{\mathbf{O}=(\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u})\leftarrow\mathbf{\Phi}}\left[\mathsf{Adv}^{\mathsf{cca1}}_{\mathsf{PKE}^{\mathbf{g},\mathbf{e},\mathbf{d}},\mathcal{A}^{\mathbf{O}}}(\lambda) \geq \frac{1}{2}\right] \geq 1 - \frac{4}{\lambda}.$$

We now show our theorem, which states that with overwhelming probability over the choice of $\mathbf{O} \leftarrow \mathbf{\Phi}$, $(\mathbf{g},\mathbf{e},\mathbf{d})$ constitutes a 1-bit $\mathcal{P}$-KDM secure PKE scheme (secure in the presence of multiple KDM-encryption queries). Since the bit-by-bit encryption preserves $\mathcal{P}$-KDM security, the existence of a 1-bit (many-time) $\mathcal{P}$-KDM secure PKE scheme implies a $\mathcal{P}$-KDM secure PKE scheme that can encrypt plaintexts of arbitrary length in the black-box sense.

[9] The purpose of $F_{\mathbf{w}}$ is to make $\mathbf{w}$ deterministic (after chosen according to the distribution $\mathbf{\Phi}$). When an oracle $\mathbf{O}$ is chosen from $\mathbf{\Phi}$, $F_{\mathbf{w}}$ will work as a truly random function. This treatment is done implicitly in [21].

[10] Note that the behavior of $\mathbf{O}$ is completely determined by $\mathbf{g}$, $\mathbf{e}$, and $F_{\mathbf{w}}$ used in $\mathbf{w}$.

Theorem 6. *For any computationally unbounded adversary $\mathcal{A}$ that makes at most polynomially many queries, there exist negligible functions $\mu(\cdot)$ and $\mu'(\cdot)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, we have*

$$\Pr_{\mathbf{O}=(\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u})\leftarrow\mathbf{\Phi}}\left[\mathsf{Adv}^{\mathsf{kdm}}_{(\mathbf{g},\mathbf{e},\mathbf{d}),\mathcal{P},\mathcal{A}^{\mathbf{O}}}(\lambda) \le \mu(\lambda)\right] \ge 1-\mu'(\lambda).$$

We remark that Theorems 5 and 6 imply Theorem 4 via a standard technique in black-box separation results (using the Borel-Cantelli lemma) (see, e.g. [25]).

Due to the space limitation, the proof of Theorem 4 is given in the full version of this paper [29], and here we give its overview. We call the $\mathcal{P}$-KDM security experiment of the PKE scheme $(\mathbf{g},\mathbf{e},\mathbf{d})$ that takes into account the choice $\mathbf{O}=(\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u})\leftarrow\mathbf{\Phi}$ the *extended KDM security experiment.* Let $\mathcal{A}$ be any computationally unbounded adversary that makes $q=q(\lambda)=\mathsf{poly}(\lambda)$ queries. To prove the theorem, it is sufficient to show that the advantage of $\mathcal{A}^{\mathbf{O},\mathcal{O}_{\mathsf{kdm}}}$ in the extended KDM security experiment is negligible. This is shown by two steps. In the first step, we show that among the suboracles given access to $\mathcal{A}$, $\mathbf{d}$, $\mathbf{w}$, and $\mathbf{u}$ do not help $\mathcal{A}$ much. More specifically, we essentially show that for $\mathcal{A}^{\mathbf{O},\mathcal{O}_{\mathsf{kdm}}}$, there exists another computationally unbounded adversary $\mathcal{B}^{\mathbf{g},\mathbf{e},\mathcal{O}_{\mathsf{kdm}}}$ that make at most $\mathsf{poly}(q)$ queries and whose advantage in the extended KDM experiment is negligibly close to that of $\mathcal{A}$'s. Then, in the second step, we show that the advantage of $\mathcal{B}^{\mathbf{g},\mathbf{e},\mathcal{O}_{\mathsf{kdm}}}$ in the extended KDM experiment is negligible by relying on the property that the suboracles $\mathbf{g}(\cdot)$ and $\mathbf{e}(\mathsf{pk},\cdot,\cdot)$ for each pk are random (almost) length-tripling injective functions when chosen according to $\mathbf{\Phi}$.

6 TDF via KDM Security

In this section, we show our proposed TDF with adaptive one-wayness, which is an extension of our IND-CCA secure KEM presented in Section 4.

Construction. Let $\ell=\ell(\lambda)$ be a polynomial. Our TDF uses the building blocks KEM, SKE, and Hash with the following properties:

- $\mathsf{KEM}=(\mathsf{KKG},\mathsf{Encap},\mathsf{Decap})$ is a KEM such that (1) its session key space is $\{0,1\}^{3\lambda}$, (2) the randomness space of Encap is $\{0,1\}^{\lambda}$, and (3) the ciphertext space $\mathcal{C}$ forms an abelian group (where we use the additive notation) and satisfies $|\mathcal{C}| \ge 2^{2\lambda}$.
- $\mathsf{SKE}=(\mathsf{K},\mathsf{E},\mathsf{D})$ is an SKE scheme such that (1) it has the randomness-recovering decryption property (with the randomness-recovering decryption algorithm RD), (2) its secret key space is $\{0,1\}^n$ for some polynomial $n=n(\lambda)$, and (3) the plaintext space is $\{0,1\}^{n\cdot\lambda+\ell}$.
  We denote the randomness space of E by $\mathcal{R}_{\mathsf{SKE}}$.
- $\mathsf{Hash}=(\mathsf{HKG},\mathsf{H})$ is a keyed hash function such that the range of H is $\{0,1\}^{\lambda}$, which we are going to assume to be target collision resistant.

Using these building blocks, the proposed TDF $\mathsf{TDF}=(\mathsf{Setup},\mathsf{Samp},\mathsf{Eval},\mathsf{Inv})$ is constructed as described in Figure 4. The domain $\mathcal{X}$ of TDF is $\mathcal{X}=\{0,1\}^n\times\{0,1\}^{n\cdot\lambda}\times\{0,1\}^{\ell}\times\mathcal{R}_{\mathsf{SKE}}$.

For the correctness and security of TDF, the following theorems hold.

```
$\mathsf{Setup}(1^\lambda)$ :
    $\forall v \in \{0,1\} : (\mathsf{pk}^v, \mathsf{sk}^v) \leftarrow \mathsf{KKG}(1^\lambda)$
    $\mathsf{A}_1, \dots, \mathsf{A}_n, \mathsf{B} \xleftarrow{\mathsf{r}} \{0,1\}^{3\lambda}$
    $\mathsf{C}_1, \dots, \mathsf{C}_n \xleftarrow{\mathsf{r}} \mathcal{C}$
    $\mathsf{hk} \leftarrow \mathsf{HKG}(1^\lambda)$
    $\mathsf{ek} \leftarrow (\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i, \mathsf{C}_i)_{i\in[n]}, \mathsf{B}, \mathsf{hk})$
    $\mathsf{td} \leftarrow (\mathsf{sk}^0, \mathsf{ek})$
    Return $(\mathsf{ek}, \mathsf{td})$.
```

```
$\mathsf{Samp}(1^\lambda)$ :
    $\mathsf{s} = (\mathsf{s}_1, \dots, \mathsf{s}_n) \leftarrow \mathsf{K}(1^\lambda)$
    $\mathsf{r}_1^{\mathsf{s}_1}, \dots, \mathsf{r}_n^{\mathsf{s}_n} \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$
    $\mathsf{k} \xleftarrow{\mathsf{r}} \{0,1\}^\ell$
    Sample $\mathsf{r}_{\mathsf{SKE}} \in \mathcal{R}_{\mathsf{SKE}}$
                    in the same way as in $\mathsf{E}$.
    Return $\mathsf{x} \leftarrow (\mathsf{s}, (\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}, \mathsf{k}, \mathsf{r}_{\mathsf{SKE}})$.
```

```
$\mathsf{Eval}(\mathsf{ek}, \mathsf{x})$ :
    $(\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i, \mathsf{C}_i)_{i\in[n]}, \mathsf{B}, \mathsf{hk}) \leftarrow \mathsf{ek}$
    $(\mathsf{s} = (\mathsf{s}_1, \dots, \mathsf{s}_n), (\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}, \mathsf{k}, \mathsf{r}_{\mathsf{SKE}}) \leftarrow \mathsf{x}$
    $\mathsf{ct}_{\mathsf{SKE}} \leftarrow \mathsf{E}(\mathsf{s}, (\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]} \| \mathsf{k}; \mathsf{r}_{\mathsf{SKE}})$
    $\forall i \in [n]$ :
        $(\mathsf{ct}_i^{\mathsf{s}_i}, \mathsf{k}_i^{\mathsf{s}_i}) \leftarrow \mathsf{Encap}(\mathsf{pk}^{\mathsf{s}_i}; \mathsf{r}_i^{\mathsf{s}_i})$
        $\mathsf{ct}_i \leftarrow \mathsf{ct}_i^{\mathsf{s}_i} + \mathsf{s}_i \cdot \mathsf{C}_i$   (‡)
            $= \begin{cases} \mathsf{ct}_i^0 & \text{if } \mathsf{s}_i = 0 \\ \mathsf{ct}_i^1 + \mathsf{C}_i & \text{if } \mathsf{s}_i = 1 \end{cases}$
    $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i)_{i\in[n]} \| \mathsf{ct}_{\mathsf{SKE}})$
    $\forall i \in [n]$ :
        $\mathsf{T}_i \leftarrow \mathsf{k}_i^{\mathsf{s}_i} + \mathsf{s}_i \cdot (\mathsf{A}_i + \mathsf{B} \cdot \mathsf{h})$   (†)
            $= \begin{cases} \mathsf{k}_i^0 & \text{if } \mathsf{s}_i = 0 \\ \mathsf{k}_i^1 + \mathsf{A}_i + \mathsf{B} \cdot \mathsf{h} & \text{if } \mathsf{s}_i = 1 \end{cases}$
    Return $\mathsf{y} \leftarrow ((\mathsf{ct}_i, \mathsf{T}_i)_{i\in[n]}, \mathsf{ct}_{\mathsf{SKE}})$.
```

```
$\mathsf{Inv}(\mathsf{td}, \mathsf{y})$ :
    $(\mathsf{sk}^0, \mathsf{ek}) \leftarrow \mathsf{td}$
    $(\mathsf{pk}^0, \mathsf{pk}^1, (\mathsf{A}_i, \mathsf{C}_i)_{i\in[n]}, \mathsf{B}, \mathsf{hk}) \leftarrow \mathsf{ek}$
    $((\mathsf{ct}_i, \mathsf{T}_i)_{i\in[n]}, \mathsf{ct}_{\mathsf{SKE}}) \leftarrow \mathsf{y}$
    $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk}, (\mathsf{ct}_i)_{i\in[n]} \| \mathsf{ct}_{\mathsf{SKE}})$
    $\forall i \in [n]$ :
        $\mathsf{s}_i \leftarrow 1 - (\mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i) \overset{?}{=} \mathsf{T}_i)$   (⋆)
            $= \begin{cases} 0 & \text{if } \mathsf{Decap}(\mathsf{sk}^0, \mathsf{ct}_i) = \mathsf{T}_i \\ 1 & \text{otherwise} \end{cases}$
    $\mathsf{s} \leftarrow (\mathsf{s}_1, \dots, \mathsf{s}_n) \in \{0,1\}^n$
    $(\mathsf{m}, \mathsf{r}_{\mathsf{SKE}}) \leftarrow \mathsf{RD}(\mathsf{s}, \mathsf{ct}_{\mathsf{SKE}})$
    Parse $\mathsf{m}$ as $(\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]} \in \{0,1\}^{n\cdot\lambda}$
                    and $\mathsf{k} \in \{0,1\}^\ell$.
    $\mathsf{x} \leftarrow (\mathsf{s}, (\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}, \mathsf{k}, \mathsf{r}_{\mathsf{SKE}})$
    If $\mathsf{Eval}(\mathsf{ek}, \mathsf{x}) = \mathsf{y}$
            then return $\mathsf{x}$ else return $\perp$.
```

Fig. 4. The proposed TDF TDF. (†) $\mathsf{h} \in \{0,1\}^\lambda$ is treated as an element of $\{0,1\}^{3\lambda}$ by some canonical injective encoding (say, putting the prefix $0^{2\lambda}$), and the arithmetic is done over $\mathrm{GF}(2^{3\lambda})$ where we identify $\{0,1\}^{3\lambda}$ with $\mathrm{GF}(2^{3\lambda})$. (‡) The addition is done over $\mathcal{C}$. (⋆) We call this step the *find step.*

Theorem 7. *Let $\epsilon = \epsilon(\lambda) \in [0,1]$. If* KEM *is ϵ-almost-all-keys correct and* SKE *has the randomness-recovering decryption property, then* TDF *is $(\epsilon + n \cdot 2^{-\lambda})$-almost-all-keys correct.*

Theorem 8. *Assume that* KEM *satisfies the pseudorandom ciphertext property and almost-all-keys correctness,* SKE *is one-time $\mathcal{P}$-KDM secure, and* Hash *is target collision resistant. Then,* TDF *is adaptively one-way.*

The proofs of Theorems 7 and 8 are given in the full version [29]. Other than using additional properties of the building blocks, the proofs for the above theorems go similarly to those for our IND-CCA secure KEM in Section 4.

Adaptively One-Way TDFs Based on the LPN Assumptions. By instantiating the building blocks in our construction TDF properly, we obtain the first adaptively one-way TDF based on the sub-exponential hardness of the *constant-noise* LPN problem. We note that previously, even a TDF with ordinary one-wayness was not known based on the constant-noise LPN assumption. Specifically, the following LPN-based building blocks can be used.

- For KEM, we use the KEM-analogue of the IND-CPA secure PKE scheme based on the sub-exponential hardness of the constant-noise LPN problem proposed by Yu and Zhang [42]. Their security analysis in fact shows that it satisfies the pseudorandom ciphertext property. However, the scheme does not satisfy almost-all-keys correctness as it is. Thus, we apply the transformation by Dwork, Naor, and Reingold [18] that transforms any PKE scheme whose correctness is imperfect into one with almost-all-keys correctness. (This transformation preserves the pseudorandom ciphertext property of the underlying scheme.)
- For SKE, we can use the $\mathcal{P}$-KDM secure SKE scheme proposed by Applebaum et al. [3] based on the (polynomial) hardness of the constant-noise LPN problem. Their scheme clearly admits the randomness-recovering decryption property. In particular, whenever a plaintext is recovered in the decryption, the decryptor can also compute the "noise" used in the encryption process, which is the only encryption randomness of this scheme. In addition, their scheme can be easily made perfectly correct.

Moreover, we can also obtain the first adaptively one-way TDF based on the (polynomial) hardness of the *low-noise* LPN problem, by replacing the Yu-Zhang scheme in the above instantiation with the existing PKE schemes based on the low-noise LPN assumption [1,17,26] (which all satisfy the pseudorandom ciphertext property). Previously, a TDF satisfying ordinary one-wayness based on the low-noise LPN assumption was proposed by Kiltz, Masny, and Pietrzak [26].

Flexible Hard-core Bits k. We note that $\mathsf{k} \in \{0,1\}^\ell$ can be directly used as hard-core bits of an input $\mathsf{x} = (\mathsf{s} = (\mathsf{s}_1, \ldots, \mathsf{s}_n), (\mathsf{r}_i^{\mathsf{s}_i})_{i\in[n]}, \mathsf{k}, \mathsf{r}_{\mathsf{SKE}})$, even in the presence of the inversion oracle. Its proof is a straightforward extension of the proof of Theorem 8, and thus omitted. Since an adaptively one-way TDF with ℓ-bit hard-core bits can be seen as an IND-CCA secure KEM with session-key space $\{0,1\}^\ell$, TDF can be viewed as an IND-CCA secure KEM in which the randomness used to generate a ciphertext is fully recovered during the decapsulation.

Additional Remarks. We remark that due to the structural similarity of our construction TDF to $\mathsf{KEM}_{\mathsf{cca}}$, several properties satisfied by $\mathsf{KEM}_{\mathsf{cca}}$ are inherited to TDF. Specifically, as in the case of our IND-CCA secure KEM $\mathsf{KEM}_{\mathsf{cca}}$, if we adopt the syntax that allows a system-wide public parameter shared by all users, pk^1, $(\mathsf{A}_i)_{i\in[n]}$, $(\mathsf{C}_i)_{i\in[n]}$, B, and hk in ek can be put in it, so that an evaluation key/trapdoor pair of each user consists only of $(\mathsf{pk}^0, \mathsf{sk}^0)$ of the underlying KEM KEM. Moreover, we can consider another variant of TDF in which the underlying $\mathcal{P}$-KDM secure SKE scheme is replaced with a hinting PRG, in a similar manner it is used in the Koppula-Waters construction [30].

Unlike $\mathsf{KEM}_{\mathsf{cca}}$, however, we can*not* make TDF perfectly correct even if we replace the underlying KEM KEM with the combination of a PKE scheme and a PRG. This is because the standard correctness of PKE does not guarantee anything about the decryption result of an element chosen randomly from the ciphertext space, which naturally occurs in the inversion process of TDF.

Acknowledgement A part of this work was supported by NTT Secure Platform Laboratories, JST OPERA JPMJOP1612, JST CREST JPMJCR14D6 and JPMJCR19F6, and JSPS KAKENHI JP16H01705 and JP17H01695.

References

1. M. Alekhnovich. More on average case vs approximation complexity. In *FOCS 2003*, pp. 298–307.
2. B. Applebaum. Key-dependent message security: Generic amplification and completeness. *EUROCRYPT 2011*, pp. 527–546.
3. B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. *CRYPTO 2009*, pp. 595–618.
4. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. *CRYPTO 2007*, pp. 535–552.
5. M. Bellare, A. Boldyreva, and J. Staddon. Randomness re-use in multi-recipient encryption schemeas. *PKC 2003*, pp. 85–99.
6. M. Bellare, S. Halevi, A. Sahai, and S. P. Vadhan. Many-to-one trapdoor functions and their relation to public-key cryptosystems. *CRYPTO 1998*, pp. 283–298.
7. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. *CCS 1993*, pp. 62–73.
8. E. Birrell, K.-M. Chung, R. Pass, and S. Telang. Randomness-dependent message security. *TCC 2013*, pp. 700–720.
9. J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. *SAC 2002*, pp. 62–75.
10. D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. *CRYPTO 1998*, pp. 1–12.
11. D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. *CRYPTO 2008*, pp. 108–125.
12. Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). *CRYPTO 2010*, pp. 1–20.
13. Z. Brakerski, A. Lombardi, G. Segev, and V. Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. *EUROCRYPT 2018, Part I*, pp. 535–564.
14. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. *EUROCRYPT 2001*, pp. 93–118.
15. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography (extended abstract). In *STOC 1991*, pp. 542–552.
16. N. Döttling, S. Garg, M. Hajiabadi, and D. Masny. New constructions of identity-based and key-dependent message secure encryption schemes. *PKC 2018, Part I*, pp. 3–31.
17. N. Döttling, J. Müller-Quade, and A. C. A. Nascimento. IND-CCA secure cryptography based on a variant of the LPN problem. *ASIACRYPT 2012*, pp. 485–503.
18. C. Dwork, M. Naor, and O. Reingold. Immunizing encryption schemes from decryption errors. *EUROCRYPT 2004*, pp. 342–360.
19. E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. *PKC 1999*, pp. 53–68.

20. S. Garg, R. Gay, and M. Hajiabadi. New techniques for efficient trapdoor functions and applications. *EUROCRYPT 2019, Part III*, pp. 33–63.
21. Y. Gertner, T. Malkin, and S. Myers. Towards a separation of semantic and CCA security for public key encryption. *TCC 2007*, pp. 434–455.
22. Y. Gertner, T. Malkin, and O. Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *FOCS 2001*, pp. 126–135.
23. S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC 1982*, pp. 365–377.
24. M. Hajiabadi and B. M. Kapron. Reproducible circularly-secure bit encryption: Applications and realizations. *CRYPTO 2015, Part I*, pp. 224–243.
25. C.-Y. Hsiao and L. Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? *CRYPTO 2004*, pp. 92–105.
26. E. Kiltz, D. Masny, and K. Pietrzak. Simple chosen-ciphertext security from low-noise LPN. *PKC 2014*, pp. 1–18.
27. E. Kiltz, P. Mohassel, and A. O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. *EUROCRYPT 2010*, pp. 673–692.
28. F. Kitagawa and T. Matsuda. CPA-to-CCA transformation for KDM security. *IACR Cryptology ePrint Archive*, 2019:609.
29. F. Kitagawa, T. Matsuda, and K. Tanaka. CCA security and trapdoor functions via key-dependent-message security. *IACR Cryptology ePrint Archive*, 2019:291.
30. V. Koppula and B. Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. *IACR Cryptology ePrint Archive*, 2018:847. To appear in CRYPTO 2019.
31. A. Lombardi, W. Quach, R. D. Rothblum, D. Wichs, and D. J. Wu. New constructions of reusable designated-verifier NIZKs. *IACR Cryptology ePrint Archive*, 2019:242. (Dated on Feb 27, 2019.) To appear in CRYPTO 2019.
32. A. Lombardi, W. Quach, R. D. Rothblum, D. Wichs, and D. J. Wu. New constructions of reusable designated-verifier NIZKs. *IACR Cryptology ePrint Archive*, 2019:242. (Dated on May 23, 2019.) To appear in CRYPTO 2019.
33. T. Matsuda and G. Hanaoka. Constructing and understanding chosen ciphertext security via puncturable key encapsulation mechanisms. *TCC 2015, Part I*, pp. 561–590.
34. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. *STOC 2008*, pp. 187–196.
35. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. *CRYPTO 1991*, pp. 433–444.
36. O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of reducibility between cryptographic primitives. *TCC 2004*, pp. 1–20.
37. J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC 1990*, pp. 387–394.
38. A. Rosen and G. Segev. Chosen-ciphertext security via correlated products. *TCC 2009*, pp. 419–436.
39. H. Wee. KDM-security via homomorphic smooth projective hashing. *PKC 2016, Part II*, pp. 159–179.
40. A. C.-C. Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS 1982*, pp. 80–91.
41. A. C.-C. Yao. How to generate and exchange secrets (extended abstract). In *FOCS 1986*, pp. 162–167.
42. Y. Yu and J. Zhang. Cryptography with auxiliary input and trapdoor from constant-noise LPN. *CRYPTO 2016, Part I*, pp. 214–243.