On the Key Dependent Message Security of the Fujisaki-Okamoto Constructions

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

[1] Tokyo Institute of Technology, Tokyo, Japan
{kitagaw1,keisuke}@is.titech.ac.jp
[2] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
{t-matsuda,hanaoka-goichiro}@aist.go.jp

Abstract. In PKC 1999, Fujisaki and Okamoto showed how to convert any public key encryption (PKE) scheme secure against chosen plaintext attacks (CPA) to a PKE scheme which is secure against chosen ciphertext attacks (CCA) in the random oracle model. Surprisingly, the resulting CCA secure scheme has almost the same efficiency as the underlying CPA secure scheme. Moreover, in J. Cryptology 2013, they proposed more efficient conversion by using the hybrid encryption framework.
In this work, we clarify whether these two constructions are also secure in the sense of *key dependent message security* against chosen ciphertext attacks (KDM-CCA security), under exactly the same assumptions on the building blocks as those used by Fujisaki and Okamoto. Specifically, we show two results: Firstly, we show that the construction proposed in PKC 1999 does not satisfy KDM-CCA security generally. Secondly, on the other hand, we show that the construction proposed in J. Cryptology 2013 satisfies KDM-CCA security.

Keywords: public key encryption, key dependent message security, chosen ciphertext security.

1 Introduction

1.1 Background and Motivation

Security against chosen ciphertext attacks (CCA) has been considered as a desirable security notion for public key encryption (PKE) schemes. In order to take adversaries who mount active attacks into consideration, it is desirable that PKE schemes satisfy CCA security. Moreover, since CCA security implies non-malleability [18, 7], it is considered that CCA security is strong enough for many applications. Therefore, many standardization bodies for public key cryptography judge whether they include a PKE scheme or not mainly based on whether the scheme satisfies CCA security [28].

On these backgrounds, it has been widely studied how to construct a practical CCA secure PKE scheme [10, 19–21, 26]. Among them, the constructions

proposed by Fujisaki and Okamoto [19, 21] are one of the most famous constructions. In [19], Fujisaki and Okamoto showed how to convert any PKE scheme secure against chosen plaintext attacks (CPA) to a PKE scheme which is CCA secure in the random oracle model. Surprisingly, the resulting CCA secure scheme has almost the same efficiency as the underlying CPA secure scheme. Moreover, in [21], they proposed more efficient conversion by using the hybrid encryption framework. EPOC (Efficient PrObabilistiC public-key encryption) that is one of the concrete instantiations of [19, 21] has been included by IEEE p1363a [1], as it has high practicality, and moreover, its security can be strictly analyzed.

CCA security has been considered as a standard security notion, but it has recently come to light that there are many situations where even CCA security may not guarantee confidentiality of communication. One typical example is situations where secret keys are encrypted in the system. It is known that there is an encryption scheme which is totally insecure when an adversary can get an encryption of secret keys, even though the scheme satisfies CCA security [16].

Black, Rogaway, and Shrimpton [11] introduced a security notion called *key dependent message (KDM) security* which guarantees confidentiality even in the situation of encrypting secret keys. (Around the same time, Camenisch and Lysyanskaya [15] independently formalized a similar notion called circular security.) It is widely known that when an encryption scheme is used as a building block of complicated systems, encrypting secret keys can often occur. Hard disk encryption systems (e.g., BitLocker [11]) and anonymous credential systems [15] are known as such examples. In addition, from the perspective of symbolic cryptography, KDM security is also important [2, 3]. From these facts, we consider that KDM security against chosen ciphertext attacks, that is, KDM-CCA security is one of the desirable security notions for practical encryption schemes.

Since CCA security is regarded as a desirable security notion, the security of standardized PKE schemes has been analyzed only in the sense of CCA security. Therefore, it is not clear whether these schemes remain secure even when an adversary can get an encryption of secret keys. In modern society where encryption schemes can be used as a building block of complicated systems, and can encrypt secret keys, it is very important to clarify whether standardized schemes are secure also in the sense of KDM-CCA security.

1.2 Our Results

Based on this motivation, in this paper, we clarify whether the constructions proposed by Fujisaki and Okamoto [19, 21] satisfy KDM-CCA security [3] under exactly the same assumptions on the building blocks as those used in [19, 21], and show two results. [4] Firstly, we show that the construction of [19] (which

[3] When we refer to "KDM security", unless stated otherwise, we mean KDM security with respect to any polynomial time computable functions. For the details, see Remark after Definition 4 in Section 2.2.

[4] Actually, the construction of [21] is based on that of [20]. In this work, we concentrate on the construction of [21].

we call FO_1) does not satisfy KDM-CCA security generally. Secondly, on the other hand, we show that the construction of [21] (which we call FO_2) satisfies KDM-CCA security. More specifically, we prove the following two theorems.

Theorem 1 (Informal). *Assume that there exists an IND-CPA secure and smooth PKE scheme. Then, there exists an IND-CPA secure and smooth PKE scheme Π such that the PKE scheme FO_1 does not satisfy KDM-CPA security in the random oracle model, where FO_1 is constructed by applying the conversion of [19] to Π.*

Theorem 2 (Informal). *Let Π be a OW-CPA secure and smooth PKE scheme, Σ be a OT-CPA secure symmetric key encryption scheme, and FO_2 be the PKE scheme which is constructed by applying the conversion of [21] to Π and Σ. Then, FO_2 satisfies KDM-CCA security in the random oracle model.*

We note that smoothness is a security notion for PKE schemes introduced by Bellare et al. [9], and essentially equivalent to γ-uniformity which is used in [19, 21]. We review the definition of smoothness in Section 2.

We think it is theoretically very interesting that the construction of [19] does not necessarily satisfy KDM-CCA security, and on the other hand, that of [21] satisfies KDM-CCA security, even though these two constructions are closely related. In addition, due to Theorem 2, we can construct various practical KDM-CCA secure PKE schemes in the random oracle model, by applying the construction of [21] to existing OW-CPA secure PKE schemes and OT-CPA secure symmetric key encryption (SKE) schemes.

The standardized PKE schemes EPOC-1 and EPOC-2 are respectively instantiated by applying the conversion of [19] and [21] to the PKE scheme proposed by Okamoto and Uchiyama [27]. We note that the counter-example we show in the proof of Theorem 1 does not capture the PKE scheme of [27]. Therefore, it is not the case that Theorem 1 states that EPOC-1 is insecure in the sense of KDM security. On the other hand, due to Theorem 2, we can immediately see that EPOC-2 is KDM-CCA secure in the random oracle model.

1.3 Related Work

Backes et al. [6] showed that RSA-OAEP is secure in the sense of KDM security in the random oracle model. More specifically, they defined a security notion called *adKDM* security which takes adaptive corruptions and arbitrary active attacks into consideration, and showed that OAEP is adKDM secure in the random oracle model if the underlying trapdoor permutation satisfies partial domain one-wayness. Recently, Davies and Stam [17] studied KDM security of hybrid encryption in the random oracle model. (Since the construction treated in [17] is associated with the construction of FO_2, we later refer to their work in detail in Section 5.)

Boneh et al. [12] constructed the first KDM secure PKE scheme in the standard model under the decisional Diffie-Hellman (DDH) assumption. Their scheme is KDM secure relative to the family of affine functions (*affine-KDM*

secure, for short) which is a comparatively simple function family. Informally, a PKE scheme is said to be KDM secure relative to a function family $\mathcal{F}$ if the scheme remains secure even when an adversary can get an encryption of $f(sk)$, where sk is the secret key and f is an arbitrary function belonging to $\mathcal{F}$. Also, affine-KDM secure schemes were later constructed under the learning with errors (LWE) [5], quadratic residuosity (QR) [13], decisional composite residuosity (DCR) [13, 25], and learning parity with noise (LPN) [5] assumptions.

Boneh et al.'s scheme is KDM secure only in the CPA setting, and thus how to construct a KDM-CCA secure scheme remained open. Camenisch et al. [14] later showed how to construct a KDM-CCA secure scheme using a KDM-CPA secure scheme and a non-interactive zero-knowledge (NIZK) proof system for NP languages as building blocks. Recently, Hofheinz [22] showed the first construction of a circular-CCA secure scheme whose security can be directly proved based on number theoretic assumptions.

Applebaum [4] showed how to construct a PKE scheme which is KDM secure relative to functions computable by a-priori bounded polynomial time, based on a PKE scheme which is KDM secure relative to a simple function family called projection functions. We note that the result of Applebaum works in both of the CPA and the CCA settings. Bellare et al. [8] showed a similar result that works only in the CPA setting but is more efficient than Applebaum's. Recently, Kitagawa et al. [23] also showed a more efficient result than Applebaum's, which works in the CCA setting. In addition, Kitagawa et al. [24] showed how to expand the plaintext space of a PKE scheme which is KDM secure relative to projection functions, without using any other assumption.

1.4 Outline of the Paper

In Section 2, we review the definitions of the primitives and the security notions that we use in this paper. Then, in Section 3, we prove Theorem 1. In the subsequent sections, we tackle Theorem 2. Our idea for proving Theorem 2 is simple, but the proof of Theorem 2 might look somewhat complicated. Thus, after reviewing the construction of FO_2 in Section 4, in Section 5, we first explain the difficulty which we encounter when trying to prove the KDM security of a hybrid encryption scheme whose key derivation function is regarded as a random oracle. Then, in Section 6, we prove Theorem 2.

In order to help the reader understand the proof of Theorem 2, in the full version of this paper, we also show that the hybrid encryption scheme whose key derivation function is a random oracle satisfies KDM-CPA security in the random oracle model, if the underlying PKE scheme and SKE scheme satisfy OW-CPA security and OT-CPA security, respectively. Since the construction can roughly be seen as a simplification of FO_2, we believe the proof is relatively easy to understand than that of Theorem 2.

2 Preliminaries

In this section we define some notations and cryptographic primitives.

2.1 Notations

$x \xleftarrow{r} X$ denotes choosing an element from a finite set X uniformly at random, and $y \leftarrow \mathsf{A}(x;r)$ denotes assigning y to the output of an algorithm A on an input x and a randomness r. When there is no need to write the randomness clearly, we omit it and simply write $y \leftarrow \mathsf{A}(x)$. For strings x and y, $x\|y$ denotes the concatenation of x and y. λ denotes a security parameter. A function $f(\lambda)$ is a negligible function 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 $f(\lambda)$ being a negligible function. PPT stands for probabilistic polynomial time. $[\ell]$ denotes the set of integers $\{1, \cdots, \ell\}$. $\mathsf{MSB}_n(x)$ denotes the first n bits of x. $\emptyset$ denotes the empty set.

2.2 Public Key Encryption

In this subsection we define public key encryption (PKE).

Definition 1 (Public key encryption). *A PKE scheme Π is a three tuple $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ of PPT algorithms.*

- *The key generation algorithm KG, given a security parameter 1^λ, outputs a public key pk and a secret key sk.*
- *The encryption algorithm Enc, given a public key pk and a message $m \in \mathcal{M}$, outputs a ciphertext c, where $\mathcal{M}$ is the plaintext space of Π.*
- *The decryption algorithm Dec, given a secret key sk and a ciphertext c, outputs a message $\tilde{m} \in \{\perp\} \cup \mathcal{M}$. This algorithm is deterministic.*

Correctness *We require $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$ for every $m \in \mathcal{M}$ and $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$.*

Next, we define one-wayness against chosen plaintext attacks (OW-CPA security) for PKE schemes. KDM security, which we define in this subsection, considers situations where there are many users, and thus KDM security is defined via a security game where there are many keys and an adversary can make many challenge queries. Therefore, for our purpose, it is useful to consider the following one-wayness in the multi-user setting. Specifically, we use a security notion which we call *List-OW-CPA security*. In the security game of List-OW-CPA security, there are many keys, and an adversary can make multiple encryption queries and outputs a list of candidate plaintexts in the final phase.

Definition 2 (List-OW-CPA security). *Let Π be a PKE scheme whose message space is $\mathcal{M}$, and ℓ be the number of keys. We define the List-OW-CPA game between a challenger and an adversary $\mathcal{A}$ as follows.*

Initialization *First, the challenger generates ℓ key pairs $(pk_j, sk_j) \leftarrow \mathsf{KG}(1^\lambda)(j = 1, \cdots, \ell)$. Then, the challenger sends $(pk_1, \cdots, pk_\ell)$ to $\mathcal{A}$. Finally, the challenger sets $L_{enc} = \emptyset$.*
$\mathcal{A}$ may make polynomially many encryption queries.

Encryption queries $j \in [\ell]$ *is an index of a key. The challenger generates* $m \xleftarrow{r} \mathcal{M}$ *and computes* $c \leftarrow \mathsf{Enc}(pk_j, m)$*. Then, the challenger adds* m *to* L_{enc} *and returns* c *to* $\mathcal{A}$*.*

Final phase $\mathcal{A}$ *outputs* L_{ans} *which is a set of plaintexts. (We require the size of* L_{ans} *to be bounded by some polynomial of* λ*.)*

In this game, we define the advantage of the adversary $\mathcal{A}$ *as follows.*

$$\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{A},\ell}(\lambda) = \Pr[L_{enc} \cap L_{ans} \neq \emptyset]$$

We say that Π *is List-OW-CPA secure if for any PPT adversary* $\mathcal{A}$ *and polynomial* $\ell = \ell(\lambda)$*, we have* $\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{A},\ell}(\lambda) = \mathsf{negl}(\lambda)$*.*

A OW-CPA secure PKE scheme Π is also List-OW-CPA secure. Formally, the following lemma holds. We provide the definition of OW-CPA security and the proof of Lemma 1 in Appendix A.

Lemma 1. *Let* Π *be a OW-CPA secure PKE scheme. Then,* Π *is also List-OW-CPA secure.*

Next, we define KDM-CPA security and KDM-CCA security for PKE schemes.

Definition 3 (KDM-CPA security). *Let* Π *be a PKE scheme and* ℓ *be the number of keys. We define the KDM-CPA game between a challenger and an adversary* $\mathcal{A}$ *as follows. In the following,* $\mathbf{sk}$ *denotes* $(sk_1, \cdots, sk_\ell)$*.*

Initialization *First, the challenger chooses a challenge bit* $b \xleftarrow{r} \{0,1\}$*. Next, the challenger generates* ℓ *key pairs* $(pk_j, sk_j) \leftarrow \mathsf{KG}(1^\lambda)(j = 1, \cdots, \ell)$ *and sends* $(pk_1, \cdots, pk_\ell)$ *to* $\mathcal{A}$*.*
$\mathcal{A}$ *may adaptively make polynomially many KDM queries.*

KDM queries (j, f)*, where* j *is a key index and* f *is a function. Here,* f *needs to be efficiently computable. If* $b = 1$ *then the challenger returns* $c \leftarrow \mathsf{Enc}(pk_j, f(\mathbf{sk}))$*; If* $b = 0$ *then the challenger returns* $c \leftarrow \mathsf{Enc}(pk_j, 0^{|f(\cdot)|})$*.*

Final phase $\mathcal{A}$ *outputs* $b' \in \{0,1\}$*.*

In this game, we define the advantage of the adversary $\mathcal{A}$ *as follows.*

$$\mathsf{Adv}^{kdmcpa}_{\Pi,\mathcal{A},\ell}(\lambda) = |\Pr[b = b'] - \frac{1}{2}|$$

We say that Π *is KDM-CPA secure if for any PPT adversary* $\mathcal{A}$ *and polynomial* $\ell = \ell(\lambda)$*, we have* $\mathsf{Adv}^{kdmcpa}_{\Pi,\mathcal{A},\ell}(\lambda) = \mathsf{negl}(\lambda)$*.*

By permitting the adversary to make decryption queries, we can analogously define KDM-CCA security.

Definition 4 (KDM-CCA security). *Let* Π *be a PKE scheme and* ℓ *be the number of keys. We define the KDM-CCA game between a challenger and an adversary* $\mathcal{A}$ *in the same way as the KDM-CPA game except that* $\mathcal{A}$ *is allowed to*

adaptively make decryption queries. In the initialization step of the KDM-CCA game, the challenger first runs in the same way as the KDM-CPA game, and then, prepares the KDM query list L_{kdm} into which pairs of the form (j, c) will be stored, where j is an index and c is a ciphertext, and which is initially empty. When $\mathcal{A}$ makes a KDM query (j, f), the challenger computes the answer c and adds (j, c) to L_{kdm}. $\mathcal{A}$ is not allowed to make a decryption query (j, c) which is contained in L_{kdm}.

Decryption queries $(j, c) \notin L_{kdm}$, *where j is a key index and c is a ciphertext. For this query, the challenger returns* $m \leftarrow \mathsf{Dec}(sk_j, c)$.

In this game, we define the advantage $\mathsf{Adv}^{kdmcca}_{\Pi,\mathcal{A},\ell}(\lambda)$ *of the adversary $\mathcal{A}$ analogously to that in the KDM-CPA game. Then, Π is said to be KDM-CCA secure if for any PPT adversary $\mathcal{A}$ and polynomial $\ell = \ell(\lambda)$, we have* $\mathsf{Adv}^{kdmcca}_{\Pi,\mathcal{A},\ell}(\lambda) = \mathsf{negl}(\lambda)$.

Remarks. Black et al. [11] first defined KDM security. In their paper, they made an assumption that functions which the adversary queries in the security game are length-regular. A function f is said to be length-regular if the output length of $f(\mathbf{sk})$ does not depend on the value of $\mathbf{sk}$, and thus we can uniquely determine the length of $f(\mathbf{sk})$ only from f. In this paper, we also impose the length-regularity of functions which the adversary queries in the security game.

In the KDM-CPA game and KDM-CCA game in the random oracle model, the adversary is allowed to make hash queries to the random oracle. Moreover, it is more appropriate to permit a function which the adversary queries as a KDM query (KDM function) to access to the random oracle, in order to capture more various situations. Actually, Black et al. used the definition which allows KDM functions to access to the random oracle. Therefore, similarly to the definition of Black et al., we allow a KDM function to access to the random oracle.

IND-CPA security is a special case of KDM-CPA security. More specifically, we can define IND-CPA security by restricting functions an adversary can query as a KDM query in the KDM-CPA game to any constant functions. Similarly, IND-CCA security is a special case of KDM-CCA security.

Usually, KDM security is defined with respect to a function family $\mathcal{F}$. $\mathcal{F}$-KDM security is defined by restricting KDM functions used by an adversary to functions belonging to $\mathcal{F}$. In this paper, unless stated otherwise, we allow an adversary to query arbitrary function computable in polynomial-time in the security game, and we omit to write a function family.

Next, we review a security notion for PKE schemes called *smoothness* [9]. Informally, a PKE scheme is said to be smooth if the number of possible ciphertexts is super-polynomially large for any message. We note that many known PKE schemes secure in the sense of indistinguishability have smoothness unconditionally, but it is not the case that any IND-CPA or IND-CCA secure PKE scheme is smooth. However, we can easily transform any non-smooth PKE scheme to a smooth one. Fujisaki and Okamoto [19, 21] proved the security of their scheme

via a property called γ-uniformity. γ-uniformity is a slightly stronger security notion than smoothness in the sense that it considers maximum also over all public keys, but these two notions are essentially the same.

Definition 5 (Smoothness [9]). *Let Π be a PKE scheme. For $\lambda \in \mathbb{N}$, we define* **Smth** *as follows.*

$$\mathbf{Smth}(\lambda) = \mathrm{E}_{(pk,sk)\leftarrow \mathsf{KG}(1^\lambda)}\left[\max_{m,c'} \Pr_{c\leftarrow \mathsf{Enc}(pk,m)}[c = c']\right]$$

We say that Π is smooth if we have $\mathbf{Smth}(\lambda) = \mathsf{negl}(\lambda)$.

We note that Definition 5 is essentially equivalent to the security notion defined via the following game played by a challenger and an adversary $\mathcal{A}$.

Initialization The challenger generates ℓ key pairs $(pk_j, sk_j) \leftarrow \mathsf{KG}(1^\lambda)(j = 1, \cdots, \ell)$ and sends $((pk_1, sk_1), \cdots, (pk_\ell, sk_\ell))$ to $\mathcal{A}$.
Final phase $\mathcal{A}$ outputs (j, m, c'), and the challenger computes $c \leftarrow \mathsf{Enc}(pk_j, m)$.

In this game, we define the advantage of the adversary $\mathcal{A}$ as follows.

$$\mathsf{Adv}^{smth}_{\Pi,\mathcal{A},\ell}(\lambda) = \Pr[c = c']$$

Then, it is straightforward to see that for any computationally unbounded adversary $\mathcal{A}$ and polynomial $\ell = \ell(\lambda)$, we have $\mathsf{Adv}^{smth}_{\Pi,\mathcal{A},\ell}(\lambda) \leq \ell \cdot \mathbf{Smth}(\lambda)$. Therefore, if $\mathbf{Smth}(\lambda)$ is negligible, so is $\mathsf{Adv}^{smth}_{\Pi,\mathcal{A},\ell}(\lambda)$ for any computationally unbounded adversary $\mathcal{A}$ and polynomial $\ell = \ell(\lambda)$.

2.3 Symmetric Key Encryption

In this subsection we define symmetric key encryption (SKE).

Definition 6 (Symmetric key encryption). *A SKE scheme Σ is a two tuple (E, D) of PPT algorithms.*

- *The encryption algorithm E, given a key $K \in \{0,1\}^\lambda$ and a message $m \in \mathcal{M}$, outputs a ciphertext c, where $\mathcal{M}$ is the plaintext space of Σ.*
- *The decryption algorithm D, given a key K and a ciphertext c, outputs a message $\tilde{m} \in \{\bot\} \cup \mathcal{M}$. This algorithm is deterministic.*

Correctness *We require $\mathsf{D}(K, \mathsf{E}(K, m)) = m$ for every $m \in \mathcal{M}$ and $K \in \{0,1\}^\lambda$.*

Next, we review the definition of indistinguishability against one-time chosen plaintext attacks (OT-CPA security) for SKE schemes.

Definition 7 (OT-CPA security). *Let Σ be a SKE scheme whose message space is $\mathcal{M}$. We define the OT-CPA game between a challenger and an adversary $\mathcal{A}$ as follows.*

$\mathsf{KG}_{\mathsf{FO1}}(1^\lambda)$:	$\mathsf{Enc}_{\mathsf{FO1}}(pk, m)$:	$\mathsf{Dec}_{\mathsf{FO1}}(sk, c)$:
$(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$	$r \leftarrow \{0,1\}^n$	$m\|r \leftarrow \mathsf{Dec}(sk, c)$
return (pk, sk)	$R \leftarrow \mathsf{H}(m\|r)$	if $m\|r = \perp$
	$c \leftarrow \mathsf{Enc}(pk, m\|r; R)$	return $\perp$
	return c	else
		$R \leftarrow \mathsf{H}(m\|r)$
		if $c \neq \mathsf{Enc}(pk, m\|r; R)$
		return $\perp$
		else
		return m

Fig. 1. The construction [19] of an IND-CCA secure PKE scheme $\mathsf{FO}_1 = (\mathsf{KG}_{\mathsf{FO1}}, \mathsf{Enc}_{\mathsf{FO1}}, \mathsf{Dec}_{\mathsf{FO1}})$ from a PKE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ which is IND-CPA secure and smooth, and a hash function H.

Initialization *First the challenger chooses a challenge bit* $b \xleftarrow{r} \{0,1\}$. *Next the challenger generates a key* $K \xleftarrow{r} \{0,1\}^\lambda$ *and sends* 1^λ *to* $\mathcal{A}$.

Challenge $\mathcal{A}$ *selects two messages* m_0 *and* m_1 *of equal length, and sends them to the challenger. Then the challenger returns* $c \leftarrow \mathsf{E}(K, m_b)$.

Final phase $\mathcal{A}$ *outputs* $b' \in \{0,1\}$.

In this game, we define the advantage of the adversary $\mathcal{A}$ *as follows.*

$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{A}}(\lambda) = |\Pr[b = b'] - \frac{1}{2}|$$

We say that Σ *is OT-CPA secure if for any PPT adversary* $\mathcal{A}$, *we have* $\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

3 Fujisaki-Okamoto Construction (PKC'99) Does Not Satisfy KDM Security in General

Fujisaki and Okamoto [19] showed how to transform any IND-CPA secure (and smooth) PKE scheme to an IND-CCA secure one by using a random oracle. The resulting scheme has almost the same efficiency as the underlying scheme. In this section, although their construction satisfies IND-CCA security, we show that their construction generally does not satisfy KDM security. In the following, we first review the construction of [19], and then we show our negative result.

Let $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme, and $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$ be a hash function, where $n = n(\lambda)$ is a polynomial. Then, we construct a PKE scheme $\mathsf{FO}_1 = (\mathsf{KG}_{\mathsf{FO1}}, \mathsf{Enc}_{\mathsf{FO1}}, \mathsf{Dec}_{\mathsf{FO1}})$ as described in Fig. 1. Here, we assume that the plaintext space of Π is $\{0,1\}^*$, and thus that of FO_1 is also $\{0,1\}^*$. In addition, let the randomness spaces of Enc and $\mathsf{Enc}_{\mathsf{FO1}}$ be both $\{0,1\}^n$.

In the above construction, Fujisaki and Okamoto showed that if Π is IND-CPA secure and smooth, and H is a random oracle, then FO_1 is IND-CCA secure in the random oracle model. However, as mentioned above, we show that FO_1 does

$\mathsf{KG}(1^\lambda)$:	$\mathsf{Enc}(pk, m)$:	$\mathsf{Dec}(sk, p\|\|c)$:
$sk \xleftarrow{r} \{0,1\}^s$	$c \leftarrow \widehat{\mathsf{Enc}}(pk, m)$	$(\widehat{pk}, \widehat{sk}) \leftarrow \widehat{\mathsf{KG}}(1^\lambda; sk)$
$(\widehat{pk}, \widehat{sk}) \leftarrow \widehat{\mathsf{KG}}(1^\lambda; sk)$	$(pk', sk') \leftarrow \widehat{\mathsf{KG}}(1^\lambda; \mathsf{MSB}_s(m))$	$m \leftarrow \widehat{\mathsf{Dec}}(\widehat{sk}, c)$
$pk \leftarrow \widehat{pk}$	if $pk = pk'$	return m
return (pk, sk)	return $1\|\|c$	
	else	
	return $0\|\|c$	

Fig. 2. The construction of a PKE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ which is IND-CPA secure and smooth but not KDM-CPA secure from an IND-CPA secure and smooth PKE scheme $\widehat{\Pi} = (\widehat{\mathsf{KG}}, \widehat{\mathsf{Enc}}, \widehat{\mathsf{Dec}})$.

not satisfy KDM-CPA security generally under the same assumptions. Formally, we show the following theorem.

Theorem 3. *Assume that there exists an IND-CPA secure and smooth PKE scheme. Then, there exists an IND-CPA secure and smooth PKE scheme Π such that FO_1 does not satisfy KDM-CPA security in the random oracle model.*

Proof of Theorem 3. This proof consists of two steps. In the first step, using any IND-CPA secure and smooth PKE scheme, we construct a PKE scheme which is still IND-CPA secure and smooth, but insecure in the sense of KDM security. Then, in the second step, we show that the PKE scheme which is constructed by applying the conversion of [19] to the PKE scheme we construct in the first step, also does not satisfy KDM-CPA security. In the following, we start with the first step.

Let $\widehat{\Pi} = (\widehat{\mathsf{KG}}, \widehat{\mathsf{Enc}}, \widehat{\mathsf{Dec}})$ be any IND-CPA secure and smooth PKE scheme. Without loss of generality, we assume that the plaintext space of $\widehat{\Pi}$ is $\{0,1\}^*$, and the randomness space of $\widehat{\mathsf{KG}}$ is $\{0,1\}^s$ for some polynomial $s = s(\lambda)$. Then, using $\widehat{\Pi}$, we construct a PKE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ as described in Fig. 2.

It is clear that if $\widehat{\Pi}$ is IND-CPA secure and smooth, then Π satisfies the same security notions. The reason is as follows. In the IND-CPA game regarding Π, since a PPT adversary can find the randomness that was used to run $\widehat{\mathsf{KG}}$ with negligible probability, when the challenger generates the challenge ciphertext, Enc outputs a ciphertext whose first bit is 1 with negligible probability. Thus, if $\widehat{\Pi}$ satisfies IND-CPA, then so is Π. Moreover, regardless of the plaintext, a ciphertext output by Enc includes a ciphertext output by $\widehat{\mathsf{Enc}}$ itself, and thus Π is smooth if so is $\widehat{\Pi}$. On the other hand, Π does not satisfy KDM-CPA security. In order to show it, we consider the following adversary $\mathcal{A}$ which attacks the KDM-CPA security of Π. For simplicity, we consider the case where only one key pair exists. On input pk, $\mathcal{A}$ queries the identity function id, gets the answer $p\|c$, and outputs $b' = p$. Let b be the challenge bit in the KDM-CPA game between the challenger and $\mathcal{A}$. Then, we can estimate the advantage of $\mathcal{A}$ as

follows.

$$\begin{aligned}\mathsf{Adv}^{kdmcpa}_{\Pi,\mathcal{A},1}(\lambda) &= \frac{1}{2}|\Pr[b'=1|b=1]-\Pr[b'=1|b=0]| \\ &= \frac{1}{2}|\Pr[p=1|b=1]-\Pr[p=1|b=0]|\end{aligned}$$

Here, let sk be the secret key corresponding to pk. Then, sk is chosen from $\{0,1\}^s$ at random and $pk = \widehat{pk}$, where $(\widehat{pk}, \widehat{sk}) = \widehat{\mathsf{KG}}(1^\lambda; sk)$. In addition, let $m_1 = sk$ and $m_0 = 0^s$. Then, we note that for any $b \in \{0,1\}$, the probability that p equals 1 is the same as the probability that $pk = pk'$ holds, where $(pk', sk') = \widehat{\mathsf{KG}}(1^\lambda; m_b)$. We note that these probabilities are taken over the choice of sk. When $b = 1$, it is straightforward that $pk = pk'$ always holds, and thus we have $\Pr[p = 1|b = 1] = 1$. On the other hand, when $b = 0$, $pk = pk'$ occurs only with negligible probability. The reason is as follows. If $pk = pk'$ holds, by the correctness of $\widehat{\Pi}$, $\widehat{\mathsf{Dec}}(sk', \widehat{\mathsf{Enc}}(pk, m)) = m$ holds for any $m \in \{0,1\}^*$. Therefore, if $pk = pk'$ holds with non-negligible probability, an adversary can break the IND-CPA security of $\widehat{\Pi}$ by generating $(pk', sk') \leftarrow \widehat{\mathsf{KG}}(1^\lambda; 0^s)$ and decrypting the challenge ciphertext using sk'. This is a contradiction, and thus we have $\Pr[p = 1|b = 0] = \mathsf{negl}(\lambda)$. From these, we have $\mathsf{Adv}^{kdmcpa}_{\Pi,\mathcal{A},1}(\lambda) = \frac{1}{2}(1 - \mathsf{negl}(\lambda))$, and we see that Π does not satisfy KDM-CPA security.

Next, we construct a PKE scheme $\mathsf{FO}_1 = (\mathsf{KG}_{\mathsf{FO1}}, \mathsf{Enc}_{\mathsf{FO1}}, \mathsf{Dec}_{\mathsf{FO1}})$ by applying the conversion in Fig. 1 to the above Π, and show that FO_1 also does not satisfy KDM-CPA security. Let (pk, sk) be a key pair output by $\mathsf{KG}_{\mathsf{FO1}}$. Here, a key pair of FO_1 is a key pair of Π itself. Namely, sk is randomly chosen from $\{0,1\}^s$ and $pk = \widehat{pk}$, where $(\widehat{pk}, \widehat{sk}) \leftarrow \widehat{\mathsf{KG}}(1^\lambda; sk)$. Then, for any $m \in \{0,1\}^s$, the probability that the first bit of the result of $\mathsf{Enc}_{\mathsf{FO1}}(pk, m)$ equals 1 is the same as the probability that $pk = pk'$ holds, where $(pk', sk') = \widehat{\mathsf{KG}}(1^\lambda; \mathsf{MSB}_s(m\|r)) = \widehat{\mathsf{KG}}(1^\lambda; m)$ and r is a randomness generated in $\mathsf{Enc}_{\mathsf{FO1}}$. These probabilities are over the choice of the random oracle H, (pk, sk), and $r \in \{0,1\}^n$. Therefore, if $m = sk$, the first bit of $\mathsf{Enc}_{\mathsf{FO1}}(pk, m; r)$ always equals 1. One the other hand, if m does not depend on sk, similarly to the first step, the first bit of $\mathsf{Enc}_{\mathsf{FO1}}(pk, m; r)$ equals 1 only with negligible probability. From these, FO_1 does not satisfy KDM-CPA security. □ (**Theorem 3**)

4 KDM-CCA Security of Fujisaki-Okamoto Construction (J. Cryptology'13)

Fujisaki and Okamoto [21] showed how to construct an IND-CCA secure PKE scheme in the random oracle model (which we call FO_2) using a OW-CPA secure PKE scheme and a OT-CPA secure SKE scheme. In this section, we show that FO_2 also satisfies KDM-CCA security in the random oracle model, under exactly the same assumptions on the building blocks as those used in [21]. First, we review the construction of FO_2.

Let $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme and $\Sigma = (\mathsf{E}, \mathsf{D})$ be a SKE scheme. Here, we assume that the message space and the randomness space of Π are

$\mathsf{KG}_{\mathsf{FO2}}(1^\lambda)$:	$\mathsf{Enc}_{\mathsf{FO2}}(pk, m)$:	$\mathsf{Dec}_{\mathsf{FO2}}(sk, (c, d))$:
$(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$	$r \xleftarrow{r} \{0,1\}^\lambda$	$r \leftarrow \mathsf{Dec}(sk, c)$
return (pk, sk)	$K \xleftarrow{r} \mathsf{G}(r)$	if $r = \bot$ or $c \neq \mathsf{Enc}(pk, r; \mathsf{H}(r, d))$
	$d \leftarrow \mathsf{E}(K, m)$	return $\bot$
	$R \leftarrow \mathsf{H}(r, d)$	else
	$c \leftarrow \mathsf{Enc}(pk, r; R)$	$K \leftarrow \mathsf{G}(r)$
	return (c, d)	$m \leftarrow \mathsf{D}(K, d)$
		return m

Fig. 3. The construction [21] of a PKE scheme $\mathsf{FO}_2 = (\mathsf{KG}_{\mathsf{FO2}}, \mathsf{Enc}_{\mathsf{FO2}}, \mathsf{Dec}_{\mathsf{FO2}})$ from a PKE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and a SKE scheme $\Sigma = (\mathsf{E}, \mathsf{D})$.

$\{0,1\}^\lambda$ and $\{0,1\}^n$, respectively, where $n = n(\lambda)$ is a polynomial. Moreover, we also assume that the message space and the key space of Σ are $\{0,1\}^*$ and $\{0,1\}^\lambda$, respectively. In addition, let $\mathsf{H} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^n$ and $\mathsf{G} : \{0,1\}^* \to \{0,1\}^\lambda$ be hash functions. Then, we construct a PKE scheme $\mathsf{FO}_2 = (\mathsf{KG}_{\mathsf{FO2}}, \mathsf{Enc}_{\mathsf{FO2}}, \mathsf{Dec}_{\mathsf{FO2}})$ as described in Fig. 3. Here, we note that the message space of FO_2 is $\{0,1\}^*$.

[21] showed that, by regarding H and G as random oracles, if Π is OW-CPA secure and smooth, and Σ is OT-CPA secure, then FO_2 satisfies IND-CCA security in the random oracle model. As mentioned earlier, we show that FO_2 satisfies KDM-CCA security, even though we require exactly the same assumptions for building blocks as those used in [21]. Formally, we show the following theorem.

Theorem 4. *Let Π be a PKE scheme which is OW-CPA secure and smooth, Σ be a OT-CPA secure SKE scheme, and H and G be random oracles. Then, FO_2 is a PKE scheme which is KDM-CCA secure in the random oracle model.*

5 Overview of Our Techniques

Our idea for proving the KDM-CCA security of FO_2 is conceptually simple. However, unfortunately, our security proof might look somewhat complicated. Thus, in this section, we first explain where the difficulty lies and how we overcome it when showing the KDM-CCA security of FO_2.

The difficulty. FO_2 has a somewhat complicated structure at first glance. However, it can roughly be seen as a hybrid encryption scheme Π_{hyb} which has the following encryption algorithm $\mathsf{Enc}_{\mathsf{hyb}}$.

$$\mathsf{Enc}_{\mathsf{hyb}}(pk, m; r) = (\mathsf{Enc}(pk, r), \mathsf{E}(\mathsf{G}(r), m))$$

Here, similarly to FO_2, $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and $\Sigma = (\mathsf{E}, \mathsf{D})$ are a OW-CPA secure PKE scheme and a OT-CPA secure SKE scheme, respectively, and G is a random oracle. The difficulty that lies in the security proof of the KDM-CCA security of FO_2, is almost the same as that of the KDM-CPA security of Π_{hyb}. Therefore, for simplicity, we explain the difficulty we encounter when showing

	Game $[b = 1]$	Game [hybrid]	Game $[b = 0]$
1st	$\mathsf{Enc}_{\mathsf{hyb}}(pk, f_1^{\mathsf{G}}(sk))$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, 0^{\lvert f_1(\cdot)\rvert})$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, 0^{\lvert f_1(\cdot)\rvert})$
2nd	$\mathsf{Enc}_{\mathsf{hyb}}(pk, f_2^{\mathsf{G}}(sk))$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, f_2^{\mathsf{G}}(sk))$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, 0^{\lvert f_2(\cdot)\rvert})$

Fig. 4. The ordinary sequence of games. "1st" and "2nd" indicate the answers to the first and second KDM queries from $\mathcal{A}$, respectively.

the KDM-CPA security of Π_{hyb} in the case where only one key pair (pk, sk) exists. In the following, we call a function an adversary queries as a KDM query in the security game a KDM function. In addition, we call an answer to a KDM query a challenge ciphertext, and randomness r encapsulated by Enc a proto-key.

We first consider a simple case, that is, the case where KDM functions cannot access to the random oracle G. In this case, an adversary who does not query a proto-key r to G cannot distinguish $(\mathsf{Enc}(pk, r), \mathsf{E}(\mathsf{G}(r), f(sk)))$ and $(\mathsf{Enc}(pk, r), \mathsf{E}(\mathsf{G}(r), 0^{|f(\cdot)|}))$ due to the randomness of outputs of G and the OT-CPA security of Σ, where f is a KDM function. Thus, all we have to consider is whether the adversary can query the proto-key r to G, but it is unlikely because of the OW-CPA security of Π. From these, in this case, we can easily see that Π_{hyb} is KDM-CPA secure.

However, in the case where KDM functions can access to the random oracle G, there is a problem. The problem is that an adversary who makes multiple KDM queries can get an encryption of a proto-key r which was used to compute a past challenge ciphertext. In order to take a closer look at this problem, we consider the reduction from the KDM-CPA security of Π_{hyb} to the OT-CPA security of Σ. For simplicity, we consider an adversary $\mathcal{A}$ who makes only two KDM queries in the KDM-CPA game of Π_{hyb}. Let f_1 and f_2 be the KDM functions that $\mathcal{A}$ sends, and b denote the challenge bit between the challenger and $\mathcal{A}$. Then, consider the sequence of games as described in Fig. 4.

Game $[b = 1]$ and Game $[b = 0]$ correspond to the KDM-CPA game when $b = 1$ and $b = 0$, respectively. If the behavior of $\mathcal{A}$ does not change non-negligibly between Game $[b = 1]$ and Game $[b = 0]$, then we can conclude that Π_{hyb} is KDM-CPA secure. In order to show this by using the OT-CPA security of Σ, we typically consider a hybrid game Game [hybrid], and we show that the behavior of $\mathcal{A}$ does not change between Game$[b = 1]$ and Game [hybrid], and between Game [hybrid] and Game $[b = 0]$.

Then, we try to construct an adversary $\mathcal{B}$ who simulates Game $[b = 1]$ or Game [hybrid] for $\mathcal{A}$ according to the value of the challenge bit between $\mathcal{B}$ and the challenger of the OT-CPA game regarding Σ. $\mathcal{B}$ first generates (pk, sk) using KG and sends pk to $\mathcal{A}$. $\mathcal{B}$ simulates G by lazy sampling. For the first KDM query f_1 from $\mathcal{A}$, $\mathcal{B}$ first makes the challenge query $(f_1^{\mathsf{G}}(sk), 0^{|f_1(\cdot)|})$ and gets the answer d_1. Then, $\mathcal{B}$ generates a proto-key r_1, computes $c_1 \leftarrow \mathsf{Enc}(pk, r_1)$, and returns (c_1, d_1) to $\mathcal{A}$. In addition, $\mathcal{B}$ let the value of $\mathsf{G}(r_1)$ be the value of the key of Σ that the challenger used to compute d_1. We note that $\mathcal{B}$ does not know the actual value of the key of Σ. Here, suppose that $\mathcal{A}$ sends the following KDM function f_2 as the second KDM query. $f_2^{\mathsf{G}}(sk)$ computes $r_1 = \mathsf{Dec}(sk, c_1)$, and

then computes $\mathsf{G}(r_1)$ and returns the value. Then, in order to compute the value of $f_2^{\mathsf{G}}(r_1)$, $\mathcal{B}$ needs the value of $\mathsf{G}(r_1)$. However, $\mathcal{B}$ does not know the actual value of $\mathsf{G}(r_1)$, and thus cannot compute $f_2^{\mathsf{G}}(sk)$ correctly. Therefore, $\mathcal{B}$ fails a simulation of Games for $\mathcal{A}$ if $\mathcal{A}$ queries such a second KDM query.

The approach of Davies and Stam [17]. Davies and Stam [17] studied KDM security for hybrid encryption where the key derivation function (KDF) is regarded as a random oracle, and pointed out the above problem. [5] Then, they overcame the problem and showed that if a PKE scheme satisfies OW-CCA security and a SKE scheme satisfies OT-CCA security, the hybrid encryption scheme satisfies KDM-CCA security in the random oracle model.

They approached the above problem by introducing a new security notion for SKE schemes that they call *prior key dependent message security (PKDM security)*. Informally, PKDM security guarantees that an encryption scheme can securely encrypt a message which depends only on keys of the scheme used to generate past ciphertexts. In other words, confidentiality of a ciphertext of a PKDM secure scheme under a key K_i holds even if an adversary can get an encryption of the form $\mathsf{E}(K_i, f(K_1, \cdots, K_{i-1}))$, where $K_1, \cdots, K_{i-1}$ are keys used so far and f is an arbitrary function. Davies and Stam showed that PKDM-CCA security is equivalent to OT-CCA security, and they overcame the above problem by reducing the KDM-CCA security of the hybrid encryption scheme to the PKDM-CCA security of the SKE scheme.

To accomplish this task, their reduction algorithm has to convert a KDM function of the secret keys of the PKE scheme to that of the keys of the SKE scheme. Here, in the KDM-CCA game which the reduction algorithm simulates, there exists a random oracle. On the other hand, in the PKDM-CCA game which the reduction algorithm actually plays, there does not exist a random oracle. Therefore, Davies and Stam used the technique of replacing the random oracle with a pseudorandom functions (PRF) when conducting the above conversion of KDM functions. Therefore, their security bound has a PRF term even though the construction does not include a PRF. In addition, they stated that it is difficult to prove its KDM-CCA security without using PKDM security or a PRF.

Our approach. Both our work and the work of [17] study the KDM-CCA security of the hybrid encryption scheme whose KDF is regarded as a random oracle. However, there is a big difference between our work and [17]. The difference is that the building blocks of [17] already satisfy CCA security. On the other hand, the building blocks of FO_2 that we treat satisfy only CPA security. In order to prove the KDM-CCA security of FO_2 even though the building blocks satisfy only CPA security, similarly to Fujisaki and Okamoto [21], we have to use smoothness [9]. (As mentioned in Section 2, smoothness is essentially the same notion as γ-uniformity.) In addition, the construction of FO_2 contains two random oracles, and one of them is used to generate a randomness for the encryption algorithm of the PKE scheme. Thus, it looks difficult to replace both

[5] Davies and Stam actually treated a hybrid encryption scheme constructed from a SKE scheme and a key encapsulation mechanism (KEM).

	Game $[b = 1]$	Game [reverse]	Game $[b = 0]$
1st	$\mathsf{Enc}_{\mathsf{hyb}}(pk, f_1^{\mathsf{G}}(sk))$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, f_1^{\mathsf{G}}(sk))$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, 0^{\lvert f_1(\cdot)\rvert})$
2nd	$\mathsf{Enc}_{\mathsf{hyb}}(pk, f_2^{\mathsf{G}}(sk))$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, 0^{\lvert f_2(\cdot)\rvert})$	$\mathsf{Enc}_{\mathsf{hyb}}(pk, 0^{\lvert f_2(\cdot)\rvert})$

Fig. 5. The sequence of games which replace the challenge ciphertexts in the "reverse order".

of two random oracles with a PRF, and thus, to directly use the proof technique used in [17]. Therefore, we try to prove the KDM-CCA security of FO_2 by a proof technique which is different from that of [17], especially a technique without using PKDM security or a PRF. In the following, we give our main idea using Π_{hyb}.

As earlier, we consider an adversary $\mathcal{A}$ for the KDM-CPA security of Π_{hyb} who makes a KDM query only twice. Our idea is to replace the challenge ciphertexts in "reverse order". Namely, we consider the sequence of games as described in Fig. 5.

Then, we can avoid the problem that we explained above. We try to construct an adversary $\mathcal{B}$ who simulates Game $[b = 1]$ or Game [reverse] for $\mathcal{A}$ according to the value of the challenge bit between $\mathcal{B}$ and the challenger of the OT-CPA game regarding Σ. $\mathcal{B}$ first generates (pk, sk) using KG and sends pk to $\mathcal{A}$. For the first KDM query f_1 from $\mathcal{A}$, $\mathcal{B}$ generates a proto-key r_1 and a key K_1 of Σ, and returns $(\mathsf{Enc}(pk, r_1), \mathsf{E}(K_1, f_1^{\mathsf{G}}(sk)))$. In addition, $\mathcal{B}$ defines the value of $\mathsf{G}(r_1)$ as K_1 by itself. For the second KDM query f_2 from $\mathcal{A}$, $\mathcal{B}$ makes the challenge query $(f_2^{\mathsf{G}}(sk), 0^{|f_2(\cdot)|})$ and gets the answer d_2. Then, $\mathcal{B}$ generates a proto-key r_2, computes $c_2 \leftarrow \mathsf{Enc}(pk, r_2)$, and returns (c_2, d_2) to $\mathcal{A}$. Since $\mathcal{B}$ defines the value of $\mathsf{G}(r_1)$ by itself, $\mathcal{B}$ can simulate G for f_2 and compute $f_2^{\mathsf{G}}(sk)$ correctly even if f_2 calls G.

Then, we in turn try to construct an adversary $\mathcal{B}'$ who simulates Game [reverse] or Game $[b = 0]$ for $\mathcal{A}$ according to the value of the challenge bit between $\mathcal{B}'$ and the challenger of the OT-CPA game regarding Σ. $\mathcal{B}'$ also generates (pk, sk) using KG and sends pk to $\mathcal{A}$. For the first KDM query f_1 from $\mathcal{A}$, $\mathcal{B}'$ first makes the challenge query $(f_1^{\mathsf{G}}(sk), 0^{|f_1(\cdot)|})$ and gets the answer d_1. Then, $\mathcal{B}'$ generates a proto-key r_1, computes $c_1 \leftarrow \mathsf{Enc}(pk, r_1)$, and returns (c_1, d_1) to $\mathcal{A}$. Here, $\mathcal{B}'$ let the value of $\mathsf{G}(r_1)$ be the value of the key of Σ that the challenger used to compute d_1, and thus $\mathcal{B}$ does not know the value of $\mathsf{G}(r_1)$. However, in this case, $\mathcal{B}'$ does not need the value of $\mathsf{G}(r_1)$ to respond to the second KDM query because the answer to this query is an encryption of $0^{|f_2(\cdot)|}$, and thus $\mathcal{B}'$ does not have to compute $f_2^{\mathsf{G}}(sk)$ actually. We note that since KDM functions are length regular, $\mathcal{B}'$ can know $|f_2(\cdot)|$ without computing $f_2^{\mathsf{G}}(sk)$. Therefore, we can overcome the problem that KDM functions may refer to past proto-keys by replacing the challenge ciphertexts in the reverse order.

When we prove the KDM-CCA security of FO_2, we have to take the OW-CPA security and smoothness of the building block PKE scheme into consideration, and then we use the identical-until-bad technique and the deferred analysis technique. Hence, in this case, whether a KDM function is computed actually or not

is very sensitive, but by dividing the bad events into smaller pieces than Davies and Stam, we are able to complete the proof.

6 Proof of Theorem 4

In this section, we show the formal proof of Theorem 4.

Let $\mathcal{A}$ be an adversary that attacks the KDM-CCA security of FO_2 in the random oracle model, and makes at most q_e KDM queries and q_d decryption queries, where q_e and q_d are polynomials of λ. Let ℓ be a polynomial of λ and denote the number of keys. As mentioned just after Definition 4, similarly to Black et al. [11], we assume that a function which $\mathcal{A}$ queries as a KDM query can access to the random oracles, and is length-regular. We note that since KDM functions can access to the random oracle, it makes security proof simple to clearly distinguish the entries of the hash list used to compute KDM functions and that used to make challenge ciphertexts. Thus, we divide the random oracle into multiple random oracles, which is a technique used by Davies and Stam [17]. Namely, in our sequence of games, there are six random oracles even though the construction of FO_2 contains only two random oracles. Now, consider the following sequence of games.

Game 0 This is the KDM-CCA game in the random oracle model regarding FO_2. See Fig. 6 for how KDM queries and decryption queries are answered, and how random oracles behave in Game 0.

In the original KDM-CCA game regarding FO_2, there exist only two random oracles H and G. However, to define the subsequent games, we consider six random oracles H, H^*, HH^*, G, G^*, and GG^*. Moreover, random oracles are implemented by lazy sampling. More specifically, random oracles run as follows.

- H maintains the list L_{H} which stores query/answer pairs so far, and runs as follows. If some value (r, d) is queried to H, H first checks whether there is an entry of the form $((r, d), R)$ in L_{H}. If so, H returns R. Otherwise, H returns a fresh random value R and adds $((r, d), R)$ to L_{H}.
- H^*, G, and G^* also maintain the query/answer pairs list L_{H^*}, L_{G}, and L_{G^*}, respectively, and run in the same way as H.
- Similarly to the other random oracles, HH^* and GG^* are implemented by lazy sampling. However, HH^* and GG^* do not have their own list. When HH^* samples a fresh random value, HH^* adds a new entry to L_{H}, and when GG^* samples a fresh random value, GG^* adds a new entry to L_{G}. In addition, HH^* runs by referring to both lists L_{H} and L_{H^*}, and GG^* runs by referring to both lists L_{G} and L_{G^*}.

Moreover, in this game, H and H^* are synchronized. Namely, H and H^* refer to not only their own list but also the list of the other one. Similarly, G and G^* are synchronized. In this game, random oracles are called at the following four cases.

- **(1)** When $\mathcal{A}$ makes a hash query.

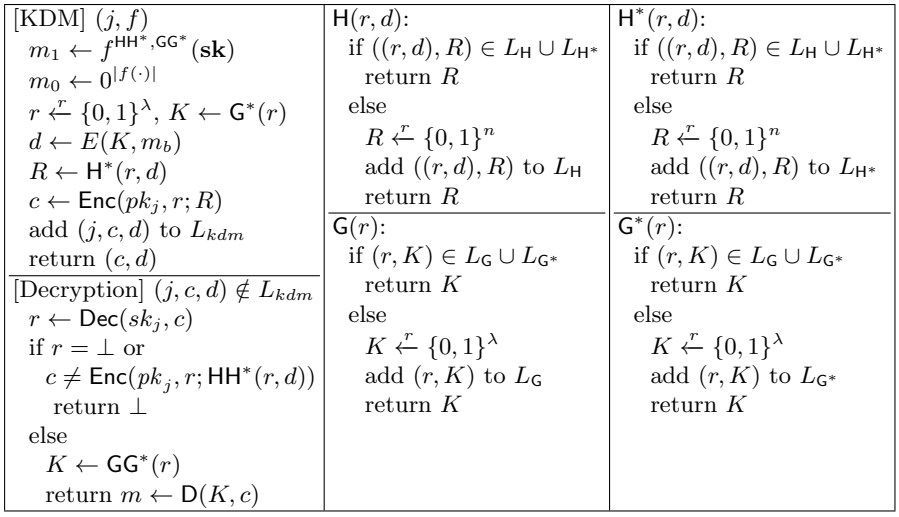

Fig. 6. The manner the challenger responds to a KDM query and a decryption query, and the behavior of H, H^*, G, and G^* in Game 0. We note that in Game 0, HH^* and GG^* run in exactly the same way as H and G, respectively.

- **(2)** When the challenger computes a hash value to respond to a KDM query from $\mathcal{A}$.
- **(3)** When a function which $\mathcal{A}$ sends to the challenger as a KDM query accesses to the random oracles.
- **(4)** When the challenger computes a hash value to respond to a decryption query from $\mathcal{A}$.

H and G are used when **(1)**, H^* and G^* are used when **(2)**, and HH^* and GG^* are used when **(3)** and **(4)**. Then, we note that the difference between this game and the original KDM-CCA game is only conceptual.

Game 1 Same as Game 0, except for the behaviors of H^* and G^*. In this game, H^* runs without referring to L_H. Moreover, every time H^* is given an input $(r, d) \in \{0,1\}^\lambda \times \{0,1\}^*$, H^* generates a uniformly random value R over $\{0,1\}^n$. Then, H^* outputs R after adding $((r, d), R)$ to L_{H^*} even if there already exists an entry whose first component is (r, d) in L_{H^*}. We note that H and HH^* still refer to L_{H^*} in this game. When H and HH^* refer to L_{H^*}, if there are multiple entries whose first components are identical, then H and HH^* adopt the entry which was added first. G, G^*, and GG^* run analogously to H, H^*, and HH^*, respectively. See Fig. 7 for how H^* and G^* behave in Game 1.

Game 2 Same as Game 1, except that H runs without referring to L_{H^*}, and G runs without referring to L_{G^*}. Here, we note that HH^* still refers to both L_H and L_{H^*}, and GG^* also refers to both L_G and L_{G^*}. See Fig. 8 for how H and G behave in Game 2.

Game 3 Same as Game 2, except that if $\mathcal{A}$ makes a decryption query, then the challenger responds as described in Fig. 9. We note that, due to this change,

$\mathsf{H}^*(r,d)$:
  $R \xleftarrow{r} \{0,1\}^n$
  add $((r,d),R)$ to L_{H^*}
  return R

$\mathsf{G}^*(r)$:
  $K \xleftarrow{r} \{0,1\}^\lambda$
  add (r,K) to L_{G^*}
  return K

Fig. 7. The behavior of H^* and G^* in Game 1.

$\mathsf{H}(r,d)$:
  if $((r,d),R) \in L_{\mathsf{H}}$
    return R
  else
    $R \xleftarrow{r} \{0,1\}^n$
    add $((r,d),R)$ to L_{H}
    return R

$\mathsf{G}(r)$:
  if $(r,K) \in L_{\mathsf{G}}$
    return K
  else
    $K \xleftarrow{r} \{0,1\}^\lambda$
    add (r,K) to L_{G}
    return K

Fig. 8. The behavior of H and G in Game 2.

[Decryption]$(j,c,d) \notin L_{kdm}$
  if $\exists((r,d),R) \in L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$
      s.t. $c = \mathsf{Enc}(pk_j, r; R)$
    $K \leftarrow \mathsf{GG}^*(r)$
    $m \leftarrow \mathsf{D}(K,d)$
    return m
  else return $\perp$

Fig. 9. The manner the challenger responds to a decryption query in Game 3.

[Decryption]$(j,c,d) \notin L_{kdm}$
  if $\exists((r,d),R) \in L_{\mathsf{H}}$
      s.t. $c = \mathsf{Enc}(pk_j, r; R)$
    $K \leftarrow \mathsf{G}(r)$
    $m \leftarrow \mathsf{D}(K,d)$
    return m
  else return $\perp$

Fig. 10. The manner the challenger responds to a decryption query in Game 4.

the challenger can respond to a decryption query without using the secret keys in this and subsequent games.

Game 4 Same as Game 3, except that if $\mathcal{A}$ makes a decryption query, the challenger refers to only L_{H} instead of $L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$ when checking the validity of the ciphertext from $\mathcal{A}$, and uses G instead of GG^* to compute the answer. See Fig. 10.

Game 5 Same as Game 4, except that if $\mathcal{A}$ makes a KDM query (j,f), the challenger always returns a ciphertext whose plaintext is $0^{|f(\cdot)|}$. See Fig. 11.

The above completes the description of the games.

We define the following events in Game i $(i = 0, \cdots, 5)$.

$\mathtt{SUC}_i$: $\mathcal{A}$ succeeds in guessing the challenge bit, that is, $b = b'$ occurs.

$\mathtt{COL}_i$: When the challenger generates $r \leftarrow \{0,1\}^\lambda$ to respond to a KDM query from $\mathcal{A}$, there exists an entry of the form $(r,\cdot)$ in $L_{\mathsf{G}} \cup L_{\mathsf{G}^*}$, or there exists an entry of the form $((r,\cdot),\cdot)$ in L_{H}.

$\mathtt{BHQ}_i$: When $\mathcal{A}$ queries r to G or queries (r,d) to H, there exists an entry of the form $(r,\cdot)$ in L_{G^*}. We call such a hash query a "**bad hash query**".

$[\mathrm{KDM}](j, f)$
$r \xleftarrow{r} \{0,1\}^\lambda$
$K \leftarrow \mathsf{G}^*(r)$
$d \leftarrow \mathsf{E}(K, 0^{|f(\cdot)|})$
$R \leftarrow \mathsf{H}^*(r, d)$
$c \leftarrow \mathsf{Enc}(pk_j, r; R)$
add (j, c, d) to L_{kdm}
return (c, d)

Fig. 11. The manner the challenger responds to a KDM query in Game 5.

In addition, we define the following two events related to decryption queries.

$\mathtt{SMTH}_i$**:** $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$ which satisfies the following two conditions, where $\mathsf{Dec}(sk_j, c) = r$: There does not exist an entry of the form $((r, d), \cdot)$ in $L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$, and $c = \mathsf{Enc}(pk, r; \mathsf{HH}^*(r, d))$ holds.

$\mathtt{BDQ}_i$**:** $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$ which satisfies the following condition: There exists an entry $((r, d), R) \in L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$ which satisfies $c = \mathsf{Enc}(pk_j, r; R)$, and for such r, $(r, \cdot) \in L_{\mathsf{G}^*}$ holds. Here, $(r, \cdot) \in L_{\mathsf{G}^*}$ indicates that there exists an entry in L_{G^*} whose first component is r. We call such a decryption query a "**bad decryption query**".

Using the above events, we can estimate $\mathsf{Adv}^{kdmcca}_{\mathsf{FO}_2,\mathcal{A},\ell}(\lambda)$ as Lemma 2 stated below.

Lemma 2. *We can estimate* $\mathsf{Adv}^{kdmcca}_{\mathsf{FO}_2,\mathcal{A},\ell}(\lambda)$ *as follows:*

$$\begin{aligned}\mathsf{Adv}^{kdmcca}_{\mathsf{FO}_2,\mathcal{A},\ell}(\lambda) \leq\ & \Pr[\mathtt{COL}_1] + 2\Pr[\mathtt{SMTH}_3] + |\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| \\ & + |\Pr[\mathtt{BHQ}_4] - \Pr[\mathtt{BHQ}_5]| + 2|\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]| \\ & + \Pr[\mathtt{BHQ}_5] + 2\Pr[\mathtt{BDQ}_5] \end{aligned} \tag{1}$$

Proof of Lemma 2. As mentioned above, the difference between Game 0 and the original KDM-CCA game is only conceptual, and thus we have $\mathsf{Adv}^{kdmcca}_{\mathsf{FO}_2,\mathcal{A},\ell}(\lambda) = |\Pr[\mathtt{SUC}_0] - \frac{1}{2}|$. By using the triangle inequality, we get the following inequality.

$$|\Pr[\mathtt{SUC}_0] - \frac{1}{2}| \leq \sum_{k=0}^{4} |\Pr[\mathtt{SUC}_k] - \Pr[\mathtt{SUC}_{k+1}]| + |\Pr[\mathtt{SUC}_5] - \frac{1}{2}|$$

We note that, in Game 5, the challenger always responds to a KDM query (j, f) from $\mathcal{A}$ by returning an encryption of $0^{|f(\cdot)|}$ regardless of the value of the challenge bit. Therefore, in Game 5, the choice of the challenge bit and the behavior of $\mathcal{A}$ are independent, and thus $|\Pr[\mathtt{SUC}_5] - \frac{1}{2}| = 0$. Below, we estimate $|\Pr[\mathtt{SUC}_k] - \Pr[\mathtt{SUC}_{k+1}]| (k = 0, 1, 2, 3, 4)$.

Game 0 and Game 1 are identical games unless when the challenger generates $r \xleftarrow{r} \{0,1\}^\lambda$, there already exists an entry whose first component is r in $L_{\mathsf{H}} \cup$

$L_{\mathsf{G}} \cup L_{\mathsf{G}^*}$. We note that if L_{G^*} does not have such an entry, the same is true for L_{H^*}. Therefore, we can see that Game 0 and Game 1 are identical unless the event $\mathtt{COL}_0$ (resp. $\mathtt{COL}_1$) occurs in Game 0 (resp. Game 1), and thus we have $|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| \leq \Pr[\mathtt{COL}_1]$.

Next, the only difference between Game 1 and Game 2 is how the challenger responds to a bad hash query from $\mathcal{A}$. In other words, Game 1 and Game 2 are identical unless the event $\mathtt{BHQ}_1$ (resp. $\mathtt{BHQ}_2$) occurs in Game 1 (resp. Game 2), and thus we have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \leq \Pr[\mathtt{BHQ}_2]$.

Moreover, Game 2 and Game 3, and Game 3 and Game 4 are identical games except for how the challenger responds to a decryption query which satisfies the condition we stated in the definition of events $\mathtt{SMTH}_i$ and $\mathtt{BDQ}_i$, respectively. In other words, Game 2 and Game 3 are identical unless the event $\mathtt{SMTH}_2$ (resp. $\mathtt{SMTH}_3$) occurs in Game 2 (resp. Game 3), and Game 3 and Game 4 are identical unless the event $\mathtt{BDQ}_3$ (resp. $\mathtt{BDQ}_4$) occurs in Game 3 (resp. Game 4). Therefore, we have $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq \Pr[\mathtt{SMTH}_3]$ and $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \Pr[\mathtt{BDQ}_4]$. Then, we get the following inequality.

$$\begin{aligned}\mathsf{Adv}^{kdmcca}_{\mathsf{FO}_2,\mathcal{A},\ell}(\lambda) \leq{}& \Pr[\mathtt{COL}_1] + \Pr[\mathtt{BHQ}_2] + \Pr[\mathtt{SMTH}_3] \\ &+ \Pr[\mathtt{BDQ}_4] + |\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| \end{aligned} \tag{2}$$

In addition, $\Pr[\mathtt{BHQ}_2] \leq \sum_{k=2}^{4} |\Pr[\mathtt{BHQ}_k] - \Pr[\mathtt{BHQ}_{k+1}]| + \Pr[\mathtt{BHQ}_5]$ holds. By considering analogously to the above argument, we get $|\Pr[\mathtt{BHQ}_2] - \Pr[\mathtt{BHQ}_3]| \leq \Pr[\mathtt{SMTH}_3]$ and $|\Pr[\mathtt{BHQ}_3] - \Pr[\mathtt{BHQ}_4] \leq \Pr[\mathtt{BDQ}_4]$. Therefore, the following inequality holds.

$$\Pr[\mathtt{BHQ}_2] \leq \Pr[\mathtt{SMTH}_3] + \Pr[\mathtt{BDQ}_4] + |\Pr[\mathtt{BHQ}_4] - \Pr[\mathtt{BHQ}_5]| + \Pr[\mathtt{BHQ}_5]$$

Moreover, we have $\Pr[\mathtt{BDQ}_4] \leq |\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]| + \Pr[\mathtt{BDQ}_5]$. By using these inequalities in the inequality (2), we get the inequality (1). □ (**Lemma 2**)

Below, we show the following lemmas that state each term of the right side of the inequality (1) is negligible.

Lemma 3. $\Pr[\mathtt{COL}_1] = \mathsf{negl}(\lambda)$.

Lemma 4. *Let Π be smooth. Then* $\Pr[\mathtt{SMTH}_3] = \mathsf{negl}(\lambda)$.

Lemma 5. *Let Σ be OT-CPA secure. Then* $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| = \mathsf{negl}(\lambda)$, $|\Pr[\mathtt{BHQ}_4] - \Pr[\mathtt{BHQ}_5]| = \mathsf{negl}(\lambda)$, *and* $|\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]| = \mathsf{negl}(\lambda)$.

Lemma 6. *Let Π be OW-CPA secure. Then* $\Pr[\mathtt{BHQ}_5] = \mathsf{negl}(\lambda)$.

Lemma 7. *Let Π be OW-CPA secure. Then* $\Pr[\mathtt{BDQ}_5] = \mathsf{negl}(\lambda)$.

Proof of Lemma 3. Since $\mathcal{A}$ is a PPT algorithm, there is a polynomial of λ which is the upper bound of the number of total entries in $L_{\mathsf{H}} \cup L_{\mathsf{G}} \cup L_{\mathsf{G}^*}$. Let $Q = Q(\lambda)$ denote this upper bound. Then, the probability that when the challenger generates $r \xleftarrow{r} \{0,1\}^\lambda$ to respond to a KDM query form $\mathcal{A}$, there is an entry of the form $(r, \cdot)$ in $L_{\mathsf{G}} \cup L_{\mathsf{G}^*}$, or there is an entry of the form $((r, \cdot), \cdot)$ in L_{H} is at most $\frac{Q}{2^\lambda}$. $\mathcal{A}$ makes a KDM query at most q_e times, and q_e is a polynomial of λ. Therefore, we have $\Pr[\mathtt{COL}_1] \leq \frac{Q \cdot q_e}{2^\lambda} = \mathsf{negl}(\lambda)$. □ (**Lemma 3**)

Proof of Lemma 4. We first define the following event for every $i \in [q_d]$.

$\mathtt{SMTH}_3^i$: In Game 3, the i-th decryption query (j, c, d) made by $\mathcal{A}$ satisfies the following two conditions, where $\mathsf{Dec}(sk_j, c) = r$: **(1)** There does not exist an entry of the form $((r, d), \cdot)$ in $L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$, and **(2)** $c = \mathsf{Enc}(pk_j, r; \mathsf{HH}^*(r, d))$.

When the condition **(1)** is satisfied, the value of $\mathsf{HH}^*(r, d)$ is defined with a newly generated uniformly random value. Then, the above condition **(2)** means that $c = \mathsf{Enc}(pk_j, r; R)$ holds, where $R \xleftarrow{r} \{0,1\}^n$. In addition, $\Pr[\mathtt{SMTH}_3] \leq \sum_{i \in [q_d]} \Pr[\mathtt{SMTH}_3^i]$ holds. Then, using the adversary $\mathcal{A}$ that attacks FO_2, we construct the following adversary $\mathcal{B}$ that attacks the smoothness of Π.

Initialization On input $((pk_1, sk_1), \cdots, (pk_\ell, sk_\ell))$, $\mathcal{B}$ first chooses $b \xleftarrow{r} \{0,1\}$ and $t \xleftarrow{r} [q_d]$. Then, $\mathcal{B}$ sends $(pk_1, \cdots, pk_\ell)$ to $\mathcal{A}$. Finally, $\mathcal{B}$ sets $\mathbf{sk} = (sk_1, \cdots, sk_\ell)$ and $L_{kdm} = L_{\mathsf{H}} = L_{\mathsf{H}^*} = L_{\mathsf{G}} = L_{\mathsf{G}^*} = \emptyset$.

Hash queries If $\mathcal{A}$ queries (r, d) to H, $\mathcal{B}$ simulates H. Namely, $\mathcal{B}$ first checks whether there is an entry of the form $((r, d), R)$ in L_{H}. If so, $\mathcal{B}$ returns R to $\mathcal{A}$. Otherwise, $\mathcal{B}$ generates $R \xleftarrow{r} \{0,1\}^n$, adds $((r, d), R)$ to L_{H}, and returns R to $\mathcal{A}$. If $\mathcal{A}$ queries r to G, $\mathcal{B}$ simulates G analogously.

KDM queries For a KDM query (j, f) from $\mathcal{A}$, $\mathcal{B}$ computes $m_1 \leftarrow f^{\mathsf{HH}^*, \mathsf{GG}^*}(\mathbf{sk})$ and $m_0 \leftarrow 0^{|m_1|}$. Here, $\mathcal{B}$ correctly forms L_{H}, L_{H^*}, L_{G}, and L_{G^*} through to the end, and thus if f calls HH^* or GG^*, $\mathcal{B}$ can simulate them for f. Next, $\mathcal{B}$ generates $r \xleftarrow{r} \{0,1\}^\lambda$, $K \xleftarrow{r} \{0,1\}^\lambda$, and $R \xleftarrow{r} \{0,1\}^n$. Then, $\mathcal{B}$ computes $c \leftarrow \mathsf{Enc}(pk_j, r; R)$ and $d \leftarrow \mathsf{E}(K, m_b)$, and returns (c, d) to $\mathcal{A}$. Finally, $\mathcal{B}$ adds (r, K) to L_{G^*}, $((r, d), R)$ to L_{H^*}, and (j, c, d) to L_{kdm}.

Decryption queries For the i-th decryption query $(j, c, d) \notin L_{kdm}$ from $\mathcal{A}$, $\mathcal{B}$ responds as follows.

- In the case $i < t$, if there does not exist an entry $((r, d), R) \in L_{\mathsf{H}}$ which satisfies $c = \mathsf{Enc}(pk_j, r; R)$, $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{B}$ first checks whether there is an entry of the form (r, K) in $L_{\mathsf{G}} \cup L_{\mathsf{G}^*}$. If so, $\mathcal{B}$ returns $m \leftarrow \mathsf{D}(K, d)$ to $\mathcal{A}$. Otherwise, $\mathcal{B}$ generates $K \xleftarrow{r} \{0,1\}^\lambda$, adds (r, K) to L_{G}, and returns $m \leftarrow \mathsf{D}(K, d)$ to $\mathcal{A}$.
- In the case $i = t$, $\mathcal{B}$ first computes $r \leftarrow \mathsf{Dec}(sk_j, c)$. Then, if there exists an entry of the form $((r, d), \cdot)$ in $L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$, $\mathcal{B}$ aborts with output $\perp$. Otherwise, $\mathcal{B}$ outputs (j, r, c) and terminates.

$\mathcal{B}$ perfectly simulates Game 3 until $\mathcal{A}$ makes the t-th decryption query. We note that since t is chosen from $[q_d]$ uniformly at random and independently of $\mathcal{A}$, and is information-theoretically hidden from the view of $\mathcal{A}$, the choice of t does not affect the behavior of $\mathcal{B}$. When $\mathcal{A}$ makes the t-th decryption query (j, c, d), $\mathcal{B}$ first computes $r \leftarrow \mathsf{Dec}(sk_j, c)$, and if there does not exist an entry of the form $((r, d), \cdot)$ in $L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$, $\mathcal{B}$ outputs (j, r, c) and terminates. Otherwise, $\mathcal{B}$ outputs $\perp$ and terminates. We can see that $\mathcal{B}$ succeeds in breaking the smoothness of Π if and only if $c = \mathsf{Enc}(pk_j, r; R)$ holds for a fresh randomness R, which means that the event $\mathtt{SMTH}_3^t$ occurs in Game 3 which $\mathcal{B}$ simulates for $\mathcal{A}$. Therefore, we

can estimate the advantage of $\mathcal{B}$ $\mathsf{Adv}^{smth}_{\Pi,\mathcal{B}}(\lambda)$ as follows.

$$\begin{aligned}\mathsf{Adv}^{smth}_{\Pi,\mathcal{B}}(\lambda) &= \sum_{i\in[q_d]} \Pr[\mathtt{SMTH}^i_3 \wedge t = i] \\ &= \sum_{i\in[q_d]} \Pr[\mathtt{SMTH}^i_3] \cdot \Pr[t = i] = \frac{1}{q_d} \sum_{i\in[q_d]} \Pr[\mathtt{SMTH}^i_3]\end{aligned}$$

Therefore, we see that $\Pr[\mathtt{SMTH}_3] \leq q_d \cdot \mathsf{Adv}^{smth}_{\Pi,\mathcal{B}}(\lambda)$. Since Π is smooth and q_d is a polynomial of λ, we have $\Pr[\mathtt{SMTH}_3] = \mathsf{negl}(\lambda)$. □ (**Lemma 4**)

Proof of Lemma 5. Using the adversary $\mathcal{A}$ that attacks FO_2, we construct the adversaries $\mathcal{B}_{\mathtt{SUC}}$, $\mathcal{B}_{\mathtt{BHQ}}$, and $\mathcal{B}_{\mathtt{BDQ}}$ all of which attack the OT-CPA security of Σ. We first describe $\mathcal{B}_{\mathtt{SUC}}$ below.

Initialization On input security parameter 1^λ, $\mathcal{B}_{\mathtt{SUC}}$ first chooses $t \xleftarrow{r} [q_e]$. Then, $\mathcal{B}_{\mathtt{SUC}}$ generates ℓ key pairs $(pk_j, sk_j) \leftarrow \mathsf{KG}(1^\lambda)(j = 1, \cdots, \ell)$ and sends $(pk_1, \cdots, pk_\ell)$ to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathtt{SUC}}$ sets $\mathbf{sk} = (sk_1, \cdots, sk_\ell)$ and $L_{kdm} = L_{\mathsf{H}} = L_{\mathsf{H}^*} = L_{\mathsf{G}} = L_{\mathsf{G}^*} = \emptyset$.

Hash queries For a hash query from $\mathcal{A}$, $\mathcal{B}_{\mathtt{SUC}}$ responds in the same manner as $\mathcal{B}$ in the proof of Lemma 4.

KDM queries For the i-th KDM query (j, f) from $\mathcal{A}$, $\mathcal{B}_{\mathtt{SUC}}$ responds as follows.

- In the case $i < t$, $\mathcal{B}_{\mathtt{SUC}}$ first computes $m_1 \leftarrow f^{\mathsf{HH}^*,\mathsf{GG}^*}(\mathbf{sk})$ and $m_0 \leftarrow 0^{|f(\cdot)|}$. Since $\mathcal{B}_{\mathtt{SUC}}$ correctly forms L_{H}, L_{H^*}, L_{G}, and L_{G^*} up to this point, when f calls HH^* and GG^*, $\mathcal{B}_{\mathtt{SUC}}$ can simulate them for f. Next, $\mathcal{B}_{\mathtt{SUC}}$ generates $r \xleftarrow{r} \{0,1\}^\lambda$, $K \xleftarrow{r} \{0,1\}^\lambda$, and $R \xleftarrow{r} \{0,1\}^n$. Then $\mathcal{B}_{\mathtt{SUC}}$ computes $c \leftarrow \mathsf{Enc}(pk_j, r; R)$ and $d \leftarrow \mathsf{E}(K, m_b)$, and returns (c, d) to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathtt{SUC}}$ adds (r, K) to L_{G^*}, $((r, d), R)$ to L_{H^*}, and (j, c, d) to L_{kdm}.
- In the case $i = t$, $\mathcal{B}_{\mathtt{SUC}}$ first computes $m_1 \leftarrow f^{\mathsf{HH}^*,\mathsf{GG}^*}(\mathbf{sk})$ and $m_0 \leftarrow 0^{|f(\cdot)|}$. Since $\mathcal{B}_{\mathtt{SUC}}$ correctly forms L_{H}, L_{H^*}, L_{G}, and L_{G^*} up to this point, when f calls HH^* and GG^*, $\mathcal{B}_{\mathtt{SUC}}$ can simulate them for f. Next, $\mathcal{B}_{\mathtt{SUC}}$ sends (m_0, m_b) as a challenge query to the challenger to get the answer d. Then, $\mathcal{B}_{\mathtt{SUC}}$ generates $r \xleftarrow{r} \{0,1\}^\lambda$ and $R \xleftarrow{r} \{0,1\}^n$, computes $c \leftarrow \mathsf{Enc}(pk_j, r; R)$, and returns (c, d) to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathtt{SUC}}$ adds $(r, \perp)$ to L_{G^*}, $((r, d), R)$ to L_{H^*}, and (j, c, d) to L_{kdm}.
- In the case $i > t$, $\mathcal{B}_{\mathtt{SUC}}$ first generates $r \xleftarrow{r} \{0,1\}^\lambda$, $K \xleftarrow{r} \{0,1\}^\lambda$, and $R \xleftarrow{r} \{0,1\}^n$. Then, $\mathcal{B}_{\mathtt{SUC}}$ computes $c \leftarrow \mathsf{Enc}(pk_j, r; R)$ and $d \leftarrow \mathsf{E}(K, 0^{|f(\cdot)|})$, and returns (c, d) to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathtt{SUC}}$ adds (r, K) to L_{G^*}, $((r, d), R)$ to L_{H^*}, and (j, c, d) to L_{kdm}.

Decryption queries For a decryption query $(j, c, d) \notin L_{kdm}$ from $\mathcal{A}$, if there does not exist an entry $((r, d), R) \in L_{\mathsf{H}}$ which satisfies $c = \mathsf{Enc}(pk_j, r; R)$, $\mathcal{B}_{\mathtt{SUC}}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{B}_{\mathtt{SUC}}$ first checks whether there is an entry of the form (r, K) in L_{G}. If so, $\mathcal{B}_{\mathtt{SUC}}$ returns $m \leftarrow \mathsf{D}(K, d)$ to $\mathcal{A}$. Otherwise, $\mathcal{B}_{\mathtt{SUC}}$ generates $K \xleftarrow{r} \{0,1\}^\lambda$, adds (r, K) to L_{G}, and returns $m \leftarrow \mathsf{D}(K, d)$ to $\mathcal{A}$.

Final phase When $\mathcal{A}$ terminates with output b', $\mathcal{B}_{\mathtt{SUC}}$ outputs $\beta_{\mathtt{SUC}} = 1$ if $b = b'$. Otherwise, $\mathcal{B}_{\mathtt{SUC}}$ outputs $\beta_{\mathtt{SUC}} = 0$.

$\mathcal{B}_{\mathtt{BHQ}}$ runs in exactly the same way as $\mathcal{B}_{\mathtt{SUC}}$ except for how to determine the final output bit $\beta_{\mathtt{BHQ}}$. $\mathcal{B}_{\mathtt{BHQ}}$ determines $\beta_{\mathtt{BHQ}}$ as follows. $\mathcal{B}_{\mathtt{BHQ}}$ initially sets $\beta_{\mathtt{BHQ}} = 0$. When $\mathcal{A}$ queries r to G or queries (r, d) to H, $\mathcal{B}_{\mathtt{BHQ}}$ first responds in the same manner as $\mathcal{B}_{\mathtt{SUC}}$. In addition, $\mathcal{B}_{\mathtt{BHQ}}$ checks whether the query is a bad hash query or not. Namely, $\mathcal{B}_{\mathtt{BHQ}}$ checks whether there exists an entry in L_{G^*} whose first component is r. If so, $\mathcal{B}_{\mathtt{BHQ}}$ sets $\beta_{\mathtt{BHQ}} = 1$. When $\mathcal{A}$ terminates with output b', $\mathcal{B}_{\mathtt{BHQ}}$ outputs $\beta_{\mathtt{BHQ}}$.

$\mathcal{B}_{\mathtt{BDQ}}$ also runs in exactly the same way as $\mathcal{B}_{\mathtt{SUC}}$ except for how to determine the final output bit $\beta_{\mathtt{BDQ}}$. $\mathcal{B}_{\mathtt{BDQ}}$ determines $\beta_{\mathtt{BDQ}}$ as follows. $\mathcal{B}_{\mathtt{BDQ}}$ initially sets $\beta_{\mathtt{BDQ}} = 0$. When $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$, $\mathcal{B}_{\mathtt{BDQ}}$ first responds in the same manner as $\mathcal{B}_{\mathtt{SUC}}$. In addition, $\mathcal{B}_{\mathtt{BDQ}}$ checks whether the query is a bad decryption query or not. Namely, $\mathcal{B}_{\mathtt{BDQ}}$ checks whether the query satisfies the following condition: There exists an entry $((r, d), R) \in L_{\mathsf{H}} \cup L_{\mathsf{H}^*}$ which satisfies $c = \mathsf{Enc}(pk_j, r; R)$, and if so, for such r, $(r, \cdot) \in L_{\mathsf{G}^*}$ holds. If (j, c, d) satisfies the above condition, then $\mathcal{B}_{\mathtt{BDQ}}$ sets $\beta_{\mathtt{BDQ}} = 1$. When $\mathcal{A}$ terminates with output b', $\mathcal{B}_{\mathtt{BDQ}}$ outputs $\beta_{\mathtt{BDQ}}$.

Let β be the challenge bit in the game between the challenger and $\mathcal{B}_{\mathtt{SUC}}$. Then, the advantage of $\mathcal{B}_{\mathtt{SUC}}$ is estimated as follows.

$$\begin{aligned}\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{SUC}}}(\lambda) &= \frac{1}{2}|\Pr[\beta_{\mathtt{SUC}} = 1|\beta = 1] - \Pr[\beta_{\mathtt{SUC}} = 1|\beta = 0]| \\ &= \frac{1}{2}|\sum_{k\in[q_e]} \Pr[\beta_{\mathtt{SUC}} = 1 \wedge t = k|\beta = 1] \\ &\qquad - \sum_{k\in[q_e]} \Pr[\beta_{\mathtt{SUC}} = 1 \wedge t = k|\beta = 0]|\end{aligned}$$

Here, for any $k \in [q_e]$, we have the following two equations.

$$\begin{aligned}\Pr[\beta_{\mathtt{SUC}} = 1 \wedge t = k|\beta = 1] &= \Pr[t = k|\beta = 1]\Pr[\beta_{\mathtt{SUC}} = 1|\beta = 1 \wedge t = k] \\ \Pr[\beta_{\mathtt{SUC}} = 1 \wedge t = k|\beta = 0] &= \Pr[t = k|\beta = 0]\Pr[\beta_{\mathtt{SUC}} = 1|\beta = 0 \wedge t = k]\end{aligned}$$

We note that t is chosen from $[q_e]$ uniformly at random and independently of β. Hence, for all $k \in [q_e]$, we have $\Pr[t = k|\beta = 1] = \Pr[t = k|\beta = 0] = \frac{1}{q_e}$. Moreover, for every $k \in [q_e - 1]$, in the cases $\beta = 1 \wedge t = k$ and $\beta = 0 \wedge t = k + 1$, $\mathcal{B}_{\mathtt{SUC}}$ responds to KDM queries from $\mathcal{A}$ in exactly the same way. In the above two cases, the only difference is whether $\mathcal{B}_{\mathtt{SUC}}$ computes $f^{\mathsf{HH}^*,\mathsf{GG}^*}(\mathbf{sk})$ to responds to the $(k + 1)$-th KDM query from $\mathcal{A}$. Due to this difference, in the above two cases, the manner L_{H} and L_{G} are formed is different. However, since $\mathcal{A}$ cannot see the contents of L_{H} and L_{G}, this does not affect the behavior of $\mathcal{A}$. Therefore, we have $\Pr[\beta_{\mathtt{SUC}} = 1|\beta = 1 \wedge t = k] = \Pr[\beta_{\mathtt{SUC}} = 1|\beta = 0 \wedge t = k + 1]$ for any $k \in [q_e - 1]$. From these, we have the following equality.

$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{SUC}}}(\lambda) = \frac{1}{2q_e}|\Pr[\beta_{\mathtt{SUC}} = 1|\beta = 1 \wedge t = q_e] - \Pr[\beta_{\mathtt{SUC}} = 1|\beta = 0 \wedge t = 1]|$$

Since $\mathcal{B}_{\mathtt{BHQ}}$ and $\mathcal{B}_{\mathtt{BDQ}}$ run in exactly the same way as $\mathcal{B}_{\mathtt{SUC}}$ except for how to determine the final output bit, all of the above arguments also hold for $\mathcal{B}_{\mathtt{BHQ}}$ and

$\mathcal{B}_{\mathtt{BDQ}}$. Therefore, we also have the following equalities.

$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{BHQ}}}(\lambda) = \frac{1}{2q_e}|\Pr[\beta_{\mathtt{BHQ}} = 1|\beta = 1 \wedge t = q_e] - \Pr[\beta_{\mathtt{BHQ}} = 1|\beta = 0 \wedge t = 1]|$$
$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{BDQ}}}(\lambda) = \frac{1}{2q_e}|\Pr[\beta_{\mathtt{BDQ}} = 1|\beta = 1 \wedge t = q_e] - \Pr[\beta_{\mathtt{BDQ}} = 1|\beta = 0 \wedge t = 1]|$$

We note that, until $\mathcal{A}$ makes the t-th KDM query, $\mathcal{B}_{\mathtt{SUC}}$ correctly forms L_{H}, L_{H^*}, L_{G}, and L_{G^*}, and thus $\mathcal{B}_{\mathtt{SUC}}$ can compute KDM functions up to this point. On the other hand, $\mathcal{B}_{\mathtt{SUC}}$ cannot simulate the t-th entry of L_{G^*} because $\mathcal{B}$ simulates the security game for $\mathcal{A}$ so that the second component of the t-th entry of L_{G^*} is the value of the key the challenger generates, and thus $\mathcal{B}_{\mathtt{SUC}}$ does not know it. Therefore, when responding to the subsequent KDM queries, $\mathcal{B}_{\mathtt{SUC}}$ cannot compute KDM functions correctly. However, since $\mathcal{B}_{\mathtt{SUC}}$ only needs the output length of KDM functions to respond to the $(t+1)$-th and subsequent KDM queries, and KDM functions are length regular, $\mathcal{B}_{\mathtt{SUC}}$ need not compute f. Then, we see that when $\beta = 1 \wedge t = q_e$, $\mathcal{B}_{\mathtt{SUC}}$ perfectly simulates Game 4 for $\mathcal{A}$. On the other hand, when $\beta = 0 \wedge t = 1$, $\mathcal{B}_{\mathtt{SUC}}$ perfectly simulates Game 5 for $\mathcal{A}$. We note that t is information-theoretically hidden from the view of $\mathcal{A}$, and thus the choice of t does not affect the behavior of $\mathcal{A}$. Here, this argument also holds for $\mathcal{B}_{\mathtt{BHQ}}$ and $\mathcal{B}_{\mathtt{BDQ}}$.

In addition, $\mathcal{B}_{\mathtt{SUC}}$ outputs 1 only when $\mathcal{A}$ succeeds in guessing b, that is, $b = b'$ occurs, $\mathcal{B}_{\mathtt{BHQ}}$ outputs 1 only when $\mathcal{A}$ makes a bad hash query, and $\mathcal{B}_{\mathtt{BDQ}}$ outputs 1 only when $\mathcal{A}$ makes a bad decryption query. Therefore, we have following equalities.

$$\Pr[\beta_{\mathtt{SUC}} = 1|\beta = 1 \wedge t = q_e] = \Pr[\mathtt{SUC}_4],\ \Pr[\beta_{\mathtt{SUC}} = 1|\beta = 0 \wedge t = 1] = \Pr[\mathtt{SUC}_5]$$
$$\Pr[\beta_{\mathtt{BHQ}} = 1|\beta = 1 \wedge t = q_e] = \Pr[\mathtt{BHQ}_4],\ \Pr[\beta_{\mathtt{BHQ}} = 1|\beta = 0 \wedge t = 1] = \Pr[\mathtt{BHQ}_5]$$
$$\Pr[\beta_{\mathtt{BDQ}} = 1|\beta = 1 \wedge t = q_e] = \Pr[\mathtt{BDQ}_4],\ \Pr[\beta_{\mathtt{BDQ}} = 1|\beta = 0 \wedge t = 1] = \Pr[\mathtt{BDQ}_5]$$

Therefore, we get the following equalities.

$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{SUC}}}(\lambda) = \frac{1}{2q_e}|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]|$$
$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{BHQ}}}(\lambda) = \frac{1}{2q_e}|\Pr[\mathtt{BHQ}_4] - \Pr[\mathtt{BHQ}_5]|$$
$$\mathsf{Adv}^{otcpa}_{\Sigma,\mathcal{B}_{\mathtt{BDQ}}}(\lambda) = \frac{1}{2q_e}|\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]|$$

Since Σ is OT-CPA secure and q_e is a polynomial of λ, we see that $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| = \mathsf{negl}(\lambda)$, $|\Pr[\mathtt{BHQ}_4] - \Pr[\mathtt{BHQ}_5]| = \mathsf{negl}(\lambda)$, and $|\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]| = \mathsf{negl}(\lambda)$. □ (**Lemma 5**)

Proof of Lemma 6. Using the adversary $\mathcal{A}$ that attacks FO_2, we construct the following adversary $\mathcal{B}$ that attacks the List-OW-CPA security of Π. We note that since Π is OW-CPA secure, by Lemma 1, Π is also List-OW-CPA secure.

Initialization On input $(pk_1, \cdots, pk_\ell)$, $\mathcal{B}$ sends $(pk_1, \cdots, pk_\ell)$ to $\mathcal{A}$, and $\mathcal{B}$ sets $L_{kdm} = L_{\mathsf{H}} = L_{\mathsf{G}} = L_{ans} = \emptyset$.

Hash queries If $\mathcal{A}$ queries (r, d) to H, $\mathcal{B}$ simulates H. Namely, $\mathcal{B}$ first checks whether there is an entry of the form $((r, d), R)$ in L_{H}. If so, $\mathcal{B}$ returns R to $\mathcal{A}$. Otherwise, $\mathcal{B}$ generates $R \xleftarrow{r} \{0,1\}^n$, adds $((r, d), R)$ to L_{H}, and returns R to $\mathcal{A}$. Then, $\mathcal{B}$ adds r to L_{ans}. If $\mathcal{A}$ queries r to G, $\mathcal{B}$ simulates G analogously to H as above, and adds r to L_{ans}.

KDM queries For a KDM query (j, f) from $\mathcal{A}$, $\mathcal{B}$ first queries j to the challenger as an encryption query and gets the answer c. Then, $\mathcal{B}$ generates $K \xleftarrow{r} \{0,1\}^\lambda$ and computes $d \leftarrow \mathsf{E}(K, 0^{|f(\cdot)|})$. Finally, $\mathcal{B}$ adds (j, c, d) to L_{kdm}, and returns (c, d) to $\mathcal{A}$.

Decryption queries For a decryption query $(j, c, d) \notin L_{kdm}$ from $\mathcal{A}$, $\mathcal{B}$ responds in the same manner as $\mathcal{B}_{\mathtt{SUC}}$ in the proof of Lemma 5.

Final phase When $\mathcal{A}$ terminates with output b', $\mathcal{B}$ outputs L_{ans}.

In the List-OW-CPA game, the challenger maintains the list L_{enc} which stores plaintexts of the challenge ciphertexts. Then, the advantage of $\mathcal{B}$ is $\Pr[L_{enc} \cap L_{ans} \neq \emptyset]$. We see that $\mathcal{B}$ perfectly simulates Game 5 for $\mathcal{A}$. If $\mathcal{A}$ queries r as a G query or (r, d) as a H query, $\mathcal{B}$ adds r to L_{ans}. In addition, in Game 5, a new entry is added to L_{G} and L_{H} only when $\mathcal{A}$ makes a G query and H query, respectively. Therefore, we can write $L_{ans} = \{r | (r, \cdot) \in L_{\mathsf{G}} \vee ((r, \cdot), \cdot) \in L_{\mathsf{H}}\}$. Here, $(r, \cdot) \in L_{\mathsf{G}}$ (resp. $((r, \cdot), \cdot) \in L_{\mathsf{H}}$) indicates that there exists an entry in L_{G} (resp. L_{H}) whose first component is r.

On the other hand, every time $\mathcal{A}$ makes a KDM query (j, f), $\mathcal{B}$ sends j to the challenger as an encryption query and gets the answer c. Here, let c be an encryption of r. Then, by the above encryption query from $\mathcal{B}$, the challenger adds r to L_{enc}. On the other hand, in Game 5, the hash value of r is computed using G^* at this point, and thus an entry of the form $(r, \cdot)$ is added to L_{G^*}. Therefore, L_{enc} can be seen as the set of the first components of the entries in L_{G^*} in Game 5. (Actually, $\mathcal{B}$ does not make L_{G^*} by itself, but $\mathcal{B}$ does not need L_{G^*} to simulate Game 5 for $\mathcal{A}$.) From these, we see that $L_{enc} \cap L_{ans} \neq \emptyset$ holds if the event $\mathtt{BHQ}_5$ occurs in Game 5 which $\mathcal{B}$ simulates for $\mathcal{A}$. Therefore, we can see that $\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{B},\ell}(\lambda) \geq \Pr[\mathtt{BHQ}_5]$. Since Π is List-OW-CPA secure, we see that $\Pr[\mathtt{BHQ}_5] = \mathsf{negl}(\lambda)$. □ (**Lemma 6**)

Proof of Lemma 7. First, we define the following two events.

$\mathtt{BDQ1}_5$**:** In Game 5, $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$ satisfying the following condition: There exists an entry $((r, d), R) \in L_{\mathsf{H}}$ which satisfies $c = \mathsf{Enc}(pk_j, r; R)$ and $(r, \cdot) \in L_{\mathsf{G}^*}$.

$\mathtt{BDQ2}_5$**:** In Game 5, $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$ satisfying the following condition: There exists an entry $((r, d), R) \in L_{\mathsf{H}^*}$ which satisfies $c = \mathsf{Enc}(pk_j, r; R)$. (We note that if there is an entry of the form $((r, \cdot), \cdot) \in L_{\mathsf{H}^*}$, then there is an entry of the form $(r, \cdot) \in L_{\mathsf{G}^*}$.)

By the definition of the events, obviously, $\mathtt{BDQ}_5 = \mathtt{BDQ1}_5 \vee \mathtt{BDQ2}_5$ holds, and thus we have $\Pr[\mathtt{BDQ}_5] \leq \Pr[\mathtt{BDQ1}_5] + \Pr[\mathtt{BDQ2}_5]$. Here, regarding $\mathtt{BDQ2}_5$, when $\mathcal{A}$ makes

a decryption query $(j, c, d) \notin L_{kdm}$ satisfying the condition of $\mathtt{BDQ2}_5$, it holds that $r = r^*$, $R = R^*$, and $d = d^*$ for some entry $(j^*, c^*, d^*) \in L_{kdm}$, where $c^* = \mathsf{Enc}(pk_{j^*}, r^*; R^*)$. In addition, in this case, $j \neq j^*$ also holds. The reason is as follows. If $j = j^*$ holds, then $c = \mathsf{Enc}(pk_j, r; R) = \mathsf{Enc}(pk_{j^*}, r^*; R^*) = c^*$ holds, and thus we have $(j, c, d) = (j^*, c^*, d^*)$, which contradicts $(j, c, d) \notin L_{kdm}$. Therefore, $j \neq j^*$ holds. From these, the event $\mathtt{BDQ2}_5$ implies the following event.

$\mathtt{BDQ2}^*_5$**:** In Game 5, $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$ satisfying the following condition: For some entry $(j^*, c^*, d^*) \in L_{kdm}$, where $c^* = \mathsf{Enc}(pk_{j^*}, r^*; R^*)$, it holds that $c = \mathsf{Enc}(pk_j, r^*; R^*)$ and $j \neq j^*$.

Here, we have $\Pr[\mathtt{BDQ2}_5] \leq \Pr[\mathtt{BDQ2}^*_5]$.

Then, using the adversary $\mathcal{A}$ that attacks FO_2, we construct the following adversary $\mathcal{B}$ that attacks the List-OW-CPA security of Π. Since Π is OW-CPA secure, from Lemma 1, Π is also List-OW-CPA secure. Here, we note that $\mathcal{B}$ attacks the List-OW-CPA security of Π in the case the number of keys is $\ell - 1$.

Initialization On input $(pk^*_1, \cdots, pk^*_{\ell-1})$, $\mathcal{B}$ first chooses $s \xleftarrow{r} [\ell]$ and generates $(pk_s, sk_s) \leftarrow \mathsf{KG}(1^\lambda)$. Then, for $1 \leq j < s$, $\mathcal{B}$ sets $pk_j = pk^*_j$, and for $s < j \leq \ell$, $\mathcal{B}$ sets $pk_j = pk^*_{j-1}$. Finally, $\mathcal{B}$ sends $(pk_1, \cdots, pk_\ell)$ to $\mathcal{A}$, and sets $L_{kdm} = L_{\mathsf{H}} = L_{\mathsf{G}} = L^1_{ans} = L^2_{ans} = \emptyset$.

Hash queries If $\mathcal{A}$ queries (r, d) to H, $\mathcal{B}$ first simulates H. Namely, $\mathcal{B}$ first checks whether there is an entry of the form $((r, d), R)$ in L_{H}. If so, $\mathcal{B}$ returns R to $\mathcal{A}$. Otherwise, $\mathcal{B}$ generates $R \xleftarrow{r} \{0,1\}^n$, adds $((r, d), R)$ to L_{H}, and returns R to $\mathcal{A}$. Then, $\mathcal{B}$ adds r to L^1_{ans}. If $\mathcal{A}$ queries r to G, $\mathcal{B}$ just simulates G analogously to H. ($\mathcal{B}$ does not add r to L^1_{ans} in the case of G query.)

KDM queries For a KDM query (j, f) from $\mathcal{A}$, $\mathcal{B}$ responds as follows.

- In the case $j \neq s$, $\mathcal{B}$ first queries j if $j < s$ and $j - 1$ if $j > s$ to the challenger as an encryption query and gets the answer c. Then, $\mathcal{B}$ generates $K \xleftarrow{r} \{0,1\}^\lambda$ and computes $d \leftarrow \mathsf{E}(K, 0^{|f(\cdot)|})$. Finally, $\mathcal{B}$ adds (j, c, d) to L_{kdm}, and returns (c, d) to $\mathcal{A}$.
- In the case $j = s$, $\mathcal{B}$ first generates $r \xleftarrow{r} \{0,1\}^\lambda$, $K \xleftarrow{r} \{0,1\}^\lambda$, and $R \xleftarrow{r} \{0,1\}^\lambda$. Then, $\mathcal{B}$ computes $c = \mathsf{Enc}(pk_j, r; R)$ and $d \leftarrow \mathsf{E}(K, 0^{|f(\cdot)|})$. Finally, $\mathcal{B}$ adds (j, c, d) to L_{kdm}, and returns (c, d) to $\mathcal{A}$.

Decryption queries For a decryption query $(j, c, d) \notin L_{kdm}$ from $\mathcal{A}$, if $j = s$, $\mathcal{B}$ first computes $r \leftarrow \mathsf{Dec}(sk_s, c)$, and adds r to L^2_{ans} if $r \neq \perp$. Then, $\mathcal{B}$ responds in the same manner as $\mathcal{B}_{\mathtt{SUC}}$ in the proof of Lemma 5.

Final phase When $\mathcal{A}$ terminates with output b', $\mathcal{B}$ chooses $\gamma \xleftarrow{r} \{1, 2\}$ and outputs L^γ_{ans}.

We see that $\mathcal{B}$ perfectly simulates Game 5 for $\mathcal{A}$. In the initialization step, $\mathcal{B}$ chooses $s \xleftarrow{r} [\ell]$ and generates $(pk_s, sk_s) \leftarrow \mathsf{KG}(1^\lambda)$. Since s is information-theoretically hidden from the view of $\mathcal{A}$, this does not affect the behavior of $\mathcal{A}$. We note that, in the List-OW-CPA game, the challenger maintains the list L_{enc} which stores plaintexts of the challenge ciphertexts. Then, it holds that $\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{B},\ell-1}(\lambda) = \Pr[L_{enc} \cap L^\gamma_{ans} \neq \emptyset]$. Here, γ is a randomness over $\{1, 2\}$ which

$\mathcal{B}$ chooses in the final phase, and thus the following equality holds.

$$\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{B},\ell-1}(\lambda) = \frac{1}{2}\Pr[L_{enc} \cap L^1_{ans} \neq \emptyset] + \frac{1}{2}\Pr[L_{enc} \cap L^2_{ans} \neq \emptyset]$$

In the following, we first consider $\Pr[L_{enc} \cap L^1_{ans} \neq \emptyset]$. When the event $\mathtt{BDQ1}_5$ occurs in Game 5 which $\mathcal{B}$ simulates for $\mathcal{A}$, there exists an entry $((r^*, d), R) \in L_{\mathsf{H}}$ which satisfies $c = \mathsf{Enc}(pk_j, r^*, R)$ and $(r^*, \cdot) \in L_{\mathsf{G}^*}$ for some decryption query $(j, c, d) \notin L_{kdm}$ from $\mathcal{A}$. We note that only when $\mathcal{A}$ makes a H query, an entry is added to L_{H}. Thus, $((r^*, d), R) \in L_{\mathsf{H}}$ means that $\mathcal{A}$ has queries (r^*, d) as a H query. Therefore, in this case, L^1_{ans} contains r^*. Also, $(r^*, \cdot) \in L_{\mathsf{G}^*}$ means that r^* is generated to compute the answer to a KDM query from $\mathcal{A}$. (Actually, $\mathcal{B}$ does not make L_{G^*} by itself, but $\mathcal{B}$ need not make L_{G^*} to simulate Game 5 for $\mathcal{A}$.) Here, let r^* be generated to compute the answer (c^*, d^*) to a KDM query (j^*, f) from $\mathcal{A}$. In other words, let c^* be an encryption of r^* under pk_{j^*}. Then, if $j^* \neq s$, c^* is computed by the challenger, and thus L_{enc} contains r^*. On the other hand, if $j^* = s$, c^* is computed by $\mathcal{B}$, and thus L_{enc} does not contain r^*. Therefore, at least when $\mathcal{A}$ makes a decryption query satisfying the condition of the event $\mathtt{BDQ1}_5$, and $j^* \neq s$ holds for the above j^*, $L_{enc} \cap L^1_{ans} \neq \emptyset$ holds. Since s is chosen from $[\ell]$ uniformly at random, and is information-theoretically hidden from the view of $\mathcal{A}$, the choice of s is independent of the behavior of $\mathcal{A}$. Therefore, the probability that $j^* \neq s$ holds under the condition that $\mathcal{A}$ has made a decryption query satisfying the condition of $\mathtt{BDQ1}_5$ is $\frac{\ell-1}{\ell}$. Moreover, since $\mathcal{B}$ perfectly simulates Game 5 for $\mathcal{A}$, the probability that $\mathcal{A}$ makes a decryption query satisfying the condition of the event $\mathtt{BDQ1}_5$ is $\Pr[\mathtt{BDQ1}_5]$. From these, we have $\Pr[L_{enc} \cap L^1_{ans} \neq \emptyset] \geq \frac{\ell-1}{\ell}\Pr[\mathtt{BDQ1}_5]$.

Next, we consider $\Pr[L_{enc} \cap L^2_{ans} \neq \emptyset]$. When the event $\mathtt{BDQ2}^*_5$ occurs in Game 5 which $\mathcal{B}$ simulates for $\mathcal{A}$, for some decryption query $(j, c, d) \notin L_{kdm}$ from $\mathcal{A}$ and some entry $(j^*, c^*, d^*) \in L_{kdm}$, it holds that $c = \mathsf{Enc}(pk_j, r^*; R^*)$ and $j \neq j^*$, where $c^* = \mathsf{Enc}(pk_{j^*}, r^*; R^*)$. Then, if $j = s$, L^2_{ans} contains r^*. In addition, in this case, $j^* \neq j = s$ holds, and thus L_{enc} also contains r^*. Therefore, at least when $\mathcal{A}$ makes a decryption query $(j, c, d) \notin L_{kdm}$ satisfying the condition of the event $\mathtt{BDQ2}^*_5$, and $j = s$ holds, $L_{enc} \cap L^2_{ans} \neq \emptyset$ holds. Since the choice of s is independent of $\mathcal{A}$, the probability that $j = s$ holds under the condition that $\mathcal{A}$ has made a decryption query satisfying the condition of $\mathtt{BDQ2}^*_5$ is $\frac{1}{\ell}$. Moreover, since $\mathcal{B}$ perfectly simlates Game 5 for $\mathcal{A}$, the probability that $\mathcal{A}$ makes a decryption query satisfying the condition of the event $\mathtt{BDQ2}^*_5$ is $\Pr[\mathtt{BDQ2}^*_5]$. Therefore, we get $\Pr[L_{enc} \cap L^2_{ans} \neq \emptyset] \geq \frac{1}{\ell}\Pr[\mathtt{BDQ2}^*_5]$.

From these, we can estimate $\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{B},\ell-1}(\lambda)$ as follows.

$$\begin{aligned}
\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{B},\ell-1}(\lambda) &= \frac{1}{2}\Pr[L_{enc} \cap L^1_{ans} \neq \emptyset] + \frac{1}{2}\Pr[L_{enc} \cap L^2_{ans} \neq \emptyset] \\
&\geq \frac{1}{2} \cdot \frac{\ell-1}{\ell}\Pr[\mathtt{BDQ1}_5] + \frac{1}{2} \cdot \frac{1}{\ell}\Pr[\mathtt{BDQ2}^*_5] \\
&\geq \frac{1}{2\ell}(\Pr[\mathtt{BDQ1}_5] + \Pr[\mathtt{BDQ2}^*_5]) \\
&\geq \frac{1}{2\ell}(\Pr[\mathtt{BDQ1}_5] + \Pr[\mathtt{BDQ2}_5]) \geq \frac{1}{2\ell}\Pr[\mathtt{BDQ}_5]
\end{aligned}$$

From the above, we have $\Pr[\mathtt{BDQ}_5] \leq 2\ell \cdot \mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{B},\ell-1}(\lambda)$. Since Π is List-OW-CPA secure and ℓ is a polynomial of λ, we see that $\Pr[\mathtt{BDQ}_5] = \mathsf{negl}(\lambda)$. □ (**Lemma 7**)

From the inequality (1) and Lemmas 3 to 7, we have $\mathsf{Adv}^{kdmcca}_{\mathsf{FO}_2,\mathcal{A},\ell}(\lambda) = \mathsf{negl}(\lambda)$. Since the choice of ℓ and $\mathcal{A}$ is arbitrary, we see that FO_2 is KDM-CCA secure in the random oracle model. □ (**Theorem 4**)

References

1. IEEE standard specifications for public-key cryptography - amendment 1: Additional techniques. *IEEE Std 1363a-2004 (Amendment to IEEE Std 1363-2000)*, Sept 2004.
2. M. Abadi and P. Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *J. Cryptology*, 20(3):395, 2007.
3. P. Adão, G. Bana, J. Herzog, and A. Scedrov. Soundness and completeness of formal encryption: The cases of key cycles and partial information leakage. *Journal of Computer Security*, 17(5):737–797, 2009.
4. B. Applebaum. Key-dependent message security: Generic amplification and completeness. *EUROCRYPT 2011*, *LNCS* 6632, pp. 527–546. 2011.
5. B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. *CRYPTO 2009*, *LNCS* 5677, pp. 595–618. 2009.
6. M. Backes, M. Dürmuth, and D. Unruh. OAEP is secure under key-dependent messages. *ASIACRYPT 2008*, *LNCS* 5350, pp. 506–523. 2008.
7. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. *CRYPTO 1998*, *LNCS* 1462, pp. 26–45. 1998.
8. M. Bellare, V. Hoang, and P. Rogaway. Garbling schemes. *IACR Cryptology ePrint Archive*, 2012:265, 2012. (The proceedings version appears in ACMCCS 2012.)
9. M. Bellare, D. Hofheinz, and E. Kiltz. Subtleties in the definition of IND-CCA: when and how should challenge decryption be disallowed? *J. Cryptology*, 28(1):29–48, 2015.
10. M. Bellare and P. Rogaway. Optimal asymmetric encryption. *EUROCRYPT 1994*, *LNCS* 950, pp. 92–111. 1994.
11. J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. *SAC 2002*, *LNCS* 2595, pp. 62–75. 2002.
12. D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. *CRYPTO 2008*, *LNCS* 5157, pp. 108–125. 2008.
13. Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). *CRYPTO 2010*, *LNCS* 6223, pp. 1–20. 2010.
14. J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. *EUROCRYPT 2009*, *LNCS* 5479, pp. 351–368. 2009.
15. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. *EUROCRYPT 2001*, *LNCS* 2045, pp. 93–118. 2001.
16. D. Cash, M. Green, and S. Hohenberger. New definitions and separations for circular security. *PKC 2012*, *LNCS* 7293, pp. 540–557. 2012.

17. G. Davies and M. Stam. KDM security in the hybrid framework. *CT-RSA 2014*, *LNCS* 8366, pp. 461–480. 2014.
18. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography (extended abstract). *STOC 1991*, pp. 542–552. 1991.
19. E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. *PKC 1999*, *LNCS* 1560, pp. 53–68. 1999.
20. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *CRYPTO 1999*, *LNCS* 1666, pp. 537–554. 1999.
21. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *J. Cryptology*, 26(1):80–101, 2013.
22. D. Hofheinz. Circular chosen-ciphertext security with compact ciphertexts. *EUROCRYPT 2013*, *LNCS* 7881, pp. 520–536. 2013.
23. F. Kitagawa, T. Matsuda, G. Hanaoka, and K. Tanaka. Efficient key dependent message security amplification against chosen ciphertext attacks. *ICISC 2014*, *LNCS* 8949, pp. 84–100. 2014.
24. F. Kitagawa, T. Matsuda, G. Hanaoka, and K. Tanaka. Completeness of single-bit projection-kdm security for public key encryption. *CT-RSA 2015*, *LNCS* 9048, pp. 201–219. 2015.
25. T. Malkin, I. Teranishi, and M. Yung. Efficient circuit-size independent public key encryption with KDM security. *EUROCRYPT 2011*, pp. 507–526. 2011.
26. T. Okamoto and D. Pointcheval. REACT: rapid enhanced-security asymmetric cryptosystem transform. *CT-RSA 2001*, *LNCS* 2020, pp. 159–175. 2001.
27. T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. *EUROCRYPT 1998*, *LNCS* 1403, pp. 308–318. 1998.
28. V. Shoup. A proposal for an ISO standard for public key encryption. *IACR Cryptology ePrint Archive*, 2001:112, 2001.

A The proof of Lemma 1

Here, we define OW-CPA security for PKE schemes, and then prove Lemma 1.

Definition 8 (OW-CPA security). *Let Π be a PKE scheme whose message space is $\mathcal{M}$. We define the OW-CPA game between a challenger and an adversary $\mathcal{A}$ as follows.*

Initialization *First the challenger generates a key pair $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$. Then, the challenger generates $m \xleftarrow{r} \mathcal{M}$ and $c \leftarrow \mathsf{Enc}(pk, m)$, and sends (pk, c) to $\mathcal{A}$.*
Final phase *$\mathcal{A}$ outputs m'.*

In this game, we define the advantage of the adversary $\mathcal{A}$ as follows.

$$\mathsf{Adv}^{owcpa}_{\Pi,\mathcal{A}}(\lambda) = \Pr[m = m']$$

We say that Π is OW-CPA secure if for any PPT adversary $\mathcal{A}$, we have $\mathsf{Adv}^{owcpa}_{\Pi,\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

We give the proof of Lemma 1 below.

Proof of Lemma 1. Let $\mathcal{A}$ be an adversary for the List-OW-CPA security of Π which makes at most q encryption queries and outputs a list L_{ans} that contains at most p elements. Let $\ell = \ell(\lambda)$ be any polynomial. Then, we define the following event $S^{i,k}$ for any $i \in [q]$ and $k \in [p]$.

$S^{i,k}$**:** Let m_i be the i-th entry of L_{enc} and m'_k be the k-th entry of L_{ans}. Then, $m_i = m'_k$ holds.

Here, we have $\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{A},\ell}(\lambda) \leq \sum_{i\in[q]} \sum_{k\in[p]} \Pr[S^{i,k}]$.

Then, using $\mathcal{A}$, we construct the following adversary $\mathcal{B}$ which attacks the OW-CPA security of Π.

Initialization On the input (pk^*, c^*), $\mathcal{B}$ first chooses $s \xleftarrow{r} [\ell]$ and $t \xleftarrow{r} [q]$. Then, $\mathcal{B}$ sets $pk_s = pk^*$, generates $\ell - 1$ key pairs $(pk_j, sk_j) \leftarrow \mathsf{KG}(1^\lambda)(j = 1, \cdots, s-1, s+1, \cdots, \ell)$, and sends $(pk_1, \cdots, pk_\ell)$ to $\mathcal{A}$.

Encryption queries For the i-th encryption query $j \in [\ell]$ made by $\mathcal{A}$, $\mathcal{B}$ responds as follows.

- In the case $i \neq t$, $\mathcal{B}$ generates $m \xleftarrow{r} \mathcal{M}$, computes $c \leftarrow \mathsf{Enc}(pk_j, m)$, and returns c to $\mathcal{A}$.
- In the case $i = t$, if $j \neq s$, then $\mathcal{B}$ aborts with output $\perp$. Otherwise, $\mathcal{B}$ returns c^* to $\mathcal{A}$.

Final phase When $\mathcal{A}$ outputs L_{ans}, $\mathcal{B}$ first chooses $u \xleftarrow{r} [p]$, where p is the number of entries in L_{ans}. Then, $\mathcal{B}$ outputs the u-th entry of L_{ans}.

If $\mathcal{B}$ does not abort, $\mathcal{B}$ perfectly simulates the List-OW-CPA game for $\mathcal{A}$. We note that s, t, and u are chosen uniformly at random. In addition, if $\mathcal{B}$ does not abort, the choice of them is information-theoretically hidden from the view of $\mathcal{A}$, and thus is independent of $\mathcal{A}$. When $\mathcal{A}$ makes the t-th encryption query j_t, if $j_t = s$, $\mathcal{B}$ returns c^* which is the challenge ciphertext for $\mathcal{B}$ itself. Let c^* be an encryption of r^*. Then, the t-th entry of L_{enc} in the List-OW-CPA game which $\mathcal{B}$ simulates for $\mathcal{A}$ is r^*. In addition, in the final phase, $\mathcal{B}$ outputs the u-th entry of L_{ans} output by $\mathcal{A}$. From these, for any $i \in [q]$ and $k \in [p]$, $\mathcal{B}$ succeeds in breaking the OW-CPA security of Π if the event $S^{i,k}$ occurs in the List-OW-CPA game which $\mathcal{B}$ simulates for $\mathcal{A}$, and $t = i$, $j_t = s$, and $u = k$ hold. Therefore, we can estimate the advantage of $\mathcal{B}$ as follows.

$$\begin{aligned}
\mathsf{Adv}^{owcpa}_{\Pi,\mathcal{B}}(\lambda) &= \sum_{i\in[q]} \sum_{k\in[p]} \Pr[S^{i,k} \wedge t = i \wedge s = j_i \wedge u = k] \\
&= \sum_{i\in[q]} \sum_{k\in[q]} \Pr[S^{i,k}] \cdot \Pr[t = i] \cdot \Pr[s = j_i] \cdot \Pr[u = k] \\
&= \frac{1}{\ell pq} \sum_{i\in[q]} \sum_{k\in[p]} \Pr[S^{i,k}] \geq \frac{1}{\ell pq} \mathsf{Adv}^{mowcpa}_{\Pi,\mathcal{A},\ell}(\lambda)
\end{aligned}$$

Since Π is OW-CPA secure, and ℓ, p, and q are polynomials of λ, we see that $\mathsf{Adv}^{lowcpa}_{\Pi,\mathcal{A},\ell}(\lambda) \leq \ell pq \cdot \mathsf{Adv}^{owcpa}_{\Pi,\mathcal{B}}(\lambda) = \mathsf{negl}(\lambda)$. □ (**Lemma 1**)