KDM Security for the Fujisaki-Okamoto Transformations in the QROM

Fuyuki Kitagawa [1]   Ryo Nishimaki [1]

[1] NTT Corporation, Tokyo, Japan
{fuyuki.kitagawa.yh,ryo.nishimaki.zk}@hco.ntt.co.jp

Abstract

Key dependent message (KDM) security is a security notion that guarantees confidentiality of communication even if secret keys are encrypted. KDM security has found a number of applications in practical situations such as hard-disk encryption systems, anonymous credentials, and bootstrapping of fully homomorphic encryption. Recently, it also found an application in quantum delegation protocols as shown by Zhang (TCC 2019).

In this work, we investigate the KDM security of existing practical public-key encryption (PKE) schemes proposed in the quantum random oracle model (QROM). Concretely, we study a PKE scheme whose KEM is constructed by using Fujisaki-Okamoto (FO) transformations in the QROM. FO transformations are applied to IND-CPA secure PKE schemes and yield IND-CCA secure key encapsulation mechanisms (KEM). Then, we show the following results.

- We can reduce the KDM-CPA security in the QROM of a PKE scheme whose KEM is derived from any of the FO transformations proposed by Hofheinz et al. (TCC 2017) to the IND-CPA security of the underlying PKE scheme, without square root security loss. For this result, we use one-time-pad (OTP) as DEM to convert KEM into PKE.
- We can reduce the KDM-CCA security in the QROM of a PKE scheme whose KEM is derived from a single variant of the FO transformation proposed by Hofheinz et al. (TCC 2017) to the IND-CPA security of the underlying PKE scheme, without square root security loss. For this result, we use OTP-then-MAC construction as DEM to convert KEM into PKE. Also, we require a mild injectivity assumption for the underlying IND-CPA secure PKE scheme.

In order to avoid square root security loss, we use a double-sided one-way to hiding (O2H) lemma proposed by Kuchta et al. (EUROCRYPT 2020). In the context of KDM security, there is a technical hurdle for using double-sided O2H lemma due to the circularity issue. Our main technical contribution is to overcome the hurdle.

Keywords: Fujisaki-Okamoto transformations, quantum random oracle model, key dependent message security

Contents

1 Introduction

1.1 Background

Post-quantum security is emerging as a de facto standard since quantum technology has been making rapid progress. In particular, since the NIST post-quantum cryptography standardization project started, IND-CCA security in the quantum random oracle model (QROM) have been extensively studied to design practical and post-quantum secure public-key encryption (PKE) [BHH$^+$19, AHU19, HKSU20, JZM19a, HHK17, JZC$^+$18, SXY18, TU16, KSS$^+$20]. IND-CCA [RS92, DDN00] is the gold standard security notion for PKE since chosen-ciphertext attacks are realistic in many practical applications [Ble98]. The random oracle model (ROM) [BR93] is an idealized model where hash functions are modeled as ideal random functions in security proofs. This idealized model helps us to design extremely efficient cryptographic primitives. In the QROM [BDF$^+$11], a random oracle query is a superposition query since adversaries are modeled as quantum polynomial-time algorithms and hash functions are locally computable.

Although IND-CCA is suitable for many practical applications, a stronger security goal than standard confidentiality is required in some settings. Key-dependent message (KDM) security [BRS03] is such an example. KDM security guarantees that adversaries cannot distinguish encryption of $f_0(\mathsf{sk})$ from encryption of $f_1(\mathsf{sk})$ where sk is a secret key and f_0, f_1 are arbitrary functions. The KDM situation is realistic in hard disk encryption systems like BitLocker [BHHO08] and bootstrapping fully homomorphic encryption [Gen09]. We also use KDM secure encryption as a building block of cryptographic primitives and protocols such as anonymous credentials [CL01]. In particular, (non-adaptive) KDM secure secret-key encryption (SKE) against quantum adversaries is used to achieve delegation of quantum computation [Zha19]. The KDM situation also naturally arises in formal verification of cryptographic protocols [AR02].

Thus, a natural question is:

Can we achieve practical KDM-CPA/CCA secure PKE in the QROM?
or
Do existing practical IND-CPA/CCA secure PKE satisfy KDM security in the QROM?

The difficulty of this question depends on what level of security and efficiency we achieve.

Security analysis in the QROM usually deviates from one in the classical ROM. One significant issue is that, in the QROM, we cannot directly use the observability of the classical ROM, which says reduction algorithms can observe input points where adversaries make random oracle queries. In the QROM, reduction algorithms need to measure superposition queries to observe random oracle queries, but this prevents reduction since adversaries can detect measurement. Superposition queries also prevent us from straightforwardly applying the adaptive programming technique. These problems make it more challenging to achieve CCA and KDM security in the QROM since each property is one of the crucial properties in the proofs for CCA and KDM [FO13, KMHT16]. New techniques have been proposed to solve the security-proof problems in the QROM. The one-way to hiding (O2H) lemma [Unr15] and its variants [AHU19, BHH$^+$19, KSS$^+$20] are the most well-known useful tools to solve the problem above and achieve secure encryption in the QROM.

Roughly speaking, the (original) O2H lemma is as follows. A quantum distinguisher $\mathcal{A}$ is given oracle access to an oracle $\mathcal{O}$, which is either a random function $H : X \to Y$ or $G : X \to Y$ such that $\forall x \notin S$, $H(x) = G(x)$. Let z be a random classical string or quantum state ((G, H, S, z) may have an arbitrary distribution). Let $\mathcal{D}$ be a quantum algorithm that is given input z and oracle access to H, measures $\mathcal{A}$'s query, and outputs the result. The distinguishing advantage of $\mathcal{A}$, $\epsilon_{\mathcal{A}}$, is bounded by the *square root* of the search advantage of $\mathcal{D}$, $\epsilon_{\mathcal{D}}$, that finds an element in S.[1] All O2H lemmas except the

[1] Here, we ignore security loss by the number of queries and constants for simplicity.

variant by Kuchta, Sakzad, Stehlé, Steinfeld, and Sun [KSS+20] incur a square root security loss. A square root security loss significantly degrades the performance of cryptographic primitives since we need to use much longer security parameters for building blocks to guarantee a reasonable security level, say, 128-bit security.[2] Thus, to achieve practical KDM secure PKE schemes, we should avoid a square root loss. When we focus on tight security, both security advantages and the running time of reductions are crucial factors. However, in most PKE schemes (and all our schemes), the overhead of running time of reductions is only additive and is not a dominant factor. Thus, we focus on security loss.

At first glance, the O2H lemma by Kuchta et al. [KSS+20] (denoted by O2H with MRM) seems to immediately answer our question since it does not incur a square root security loss. However, this is not the case. O2H with MRM is a variation of the *double-sided* O2H lemma by Bindel, Hamburg, Hövelmanns, Hülsing, and Persichetti [BHH+19], where $\mathcal{D}$ is given oracle access to *both* H *and* G. Thus, in O2H with MRM, $\mathcal{D}$ is given oracle access to a random oracle H and *a modified random oracle* G. This is not an issue for proving IND-CPA/CCA security. However, it is a serious issue for proving KDM security because correlated information about secret keys could remain in the modified random oracle G in known proofs for KDM in the classical ROM. See Section 1.4 for the detail. Kuchta et al. [KSS+20] left relaxing their double-sided O2H with MRM to a single-sided variant as an open question. However, that question remains elusive. In the KDM setting, we cannot directly apply a double-sided type O2H lemma. Achieving KDM security with a double-sided O2H lemma is of independent interest. Thus, our question is more precisely described as follows.

Can we achieve practical KDM-CPA/CCA secure PKE without a square root security loss in the QROM?

or

Do existing practical IND-CPA/CCA secure PKE satisfy KDM security without a square root security loss in the QROM?

1.2 Our Result

In this work, we affirmatively answer the question above. We prove the following.

- We can obtain KDM-CPA secure PKE without a square root security loss by applying a Fujisaki-Okamoto transformation (denoted by FO) [FO13, HHK17] to IND-CPA secure PKE and combining one-time pad (OTP) as DEM.
- We can obtain KDM-CCA secure PKE without a square root security loss by applying an FO [FO13, BHH+19] to IND-CPA secure PKE and combining OTP and strong one-time MAC[3] (that is, OTP-then-MAC) as DEM.

Note that our goal is PKE (not KEM) since we can consider the KDM setting only in PKE. We need OTP to achieve PKE since FO yields KEM [FO13, HHK17]. Our results are extremely versatile since we can convert IND-CPA secure PKE to KDM-CPA/CCA secure PKE by the well-known general transformations. FO yields practical KEM/PKE schemes and is employed in many candidates of the NIST PQC standardization to achieve CCA security. Note that we do not need the perfect correctness of the building block PKE. However, for the result on KDM-CCA secure PKE, we require that a derandomized version of the building block PKE is injective as in the CCA schemes in some previous works [BHH+19, KSS+20]. Bindel et al. argue that injectivity is commonly satisfied by many practical IND-CPA secure lattice based schemes [BHH+19]. We also note that we use PKE in the multi-user

[2] Saito, Xagawa, and Yamakawa [SXY18] estimate that we need 376-bit security of underlying trapdoor functions for 128-bit security of the IND-CCA KEM scheme by Boneh et al. [BDF+11] if the number of queries is 2^{60} due to a square root security loss.

[3] Strong one-time MAC unconditionally exists.

setting [BBM00] as the building block PKE in the transformation since the KDM setting is the multi-user setting by default.[4]

To explain our result more precisely, we recall that an FO can be decomposed into two transformations T and U. This was first observed by Hofheinz, Hövelmanns, and Kiltz [HHK17]. In this work, we adopt variants of T and U defined by Bindel et al. [BHH$^+$19]. The only difference between the transformations by Hofheinz et al. and those by Bindel et al. is that the validity check by encryption in the decryption algorithm is performed as a part of T in the former while it is performed as a part of U in the latter. Thus, the resulting FO is the same regardless of which definitions of T and U we use.

T transformation transforms an IND-CPA secure PKE scheme into an OW-CPA secure deterministic PKE scheme. U transformation transforms an OW-CPA secure deterministic PKE scheme into an IND-CCA secure KEM. Regarding U, there are six variants, $\mathsf{U}^{\perp}$, $\mathsf{U}^{\not\perp}$, $\mathsf{U}^{\perp,\mathtt{keyconf}}$, $\mathsf{U}^{\perp}_{m}$, $\mathsf{U}^{\not\perp}_{m}$, and $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$. Here, $\perp$ and $\not\perp$ mean explicit and implicit rejection in decryption, respectively, and no subscript and subscript m mean a hash function takes a ciphertext as a part of the input or not. Superscript $\mathtt{keyconf}$ (key confirmation) means that we add a hash value of a plaintext to a ciphertext and check the hash value in decryption. Bindel et al. [BHH$^+$19] prove that $\mathsf{U}^{\perp}$, $\mathsf{U}^{\not\perp}$, and $\mathsf{U}^{\perp,\mathtt{keyconf}}$ yield IND-CCA KEM if and only if $\mathsf{U}^{\perp}_{m}$, $\mathsf{U}^{\not\perp}_{m}$, and $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ yield IND-CCA KEM, respectively. It does not matter whether a hash function takes a ciphertext as the input or not. This is also the case in the context of KDM security since the prove can be done via simple mappings between random functions. Thus, in this work, we focus on $\mathsf{U}^{\perp}_{m}$, $\mathsf{U}^{\not\perp}_{m}$, and $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$.

To solve the correlated information problem above, we introduce a new security notion called *seed-dependent message one-wayness against related seed attacks (SDM-OW-RSA)*. This notion is a technical contribution and plays a crucial role in this work (defined in Section 2.3). Then, we show that if we apply the $\mathsf{U}^{\perp}_{m}$ transformation to SDM-OW-RSA deterministic PKE, the resulting scheme is KDM-CPA secure by combining OTP as DEM. We also show that if we apply $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ to SDM-OW-RSA secure deterministic PKE with injectivity, the resulting scheme is KDM-CCA secure by combining OTP-then-MAC as DEM. Although we need O2H with MRM in this part to avoid a square root security loss, we can overcome the double-sided oracle issue due to SDM-OW-RSA security.

In order to complete the proof for the KDM security of FO transformations, we go to the following path. We first introduce a variant of T that we call T transformation with hash key generation $\mathsf{T}_{\mathtt{HKG}}$, and show that if we apply $\mathsf{T}_{\mathtt{HKG}}$ to IND-CPA PKE, the resulting deterministic PKE scheme satisfies SDM-OW-RSA without square root security loss. Combined with the above, we see that $\mathsf{U}^{\perp}_{m}$ (resp. $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$) together with $\mathsf{T}_{\mathtt{HKG}}$ can be used to obtain a KDM-CPA (resp. KDM-CCA) secure PKE scheme from an IND-CPA secure PKE scheme without square root loss. Finally, we show that $\mathsf{T}_{\mathtt{HKG}}$ in those constructions can be replaced with T, thus prove the KDM security of FO transformations.

Although we omit in this paper, we can see that we can prove the KDM-CPA security without a square root security loss even if we use $\mathsf{U}^{\not\perp}_{m}$ instead of $\mathsf{U}^{\perp}_{m}$. Interestingly, if we use $\mathsf{U}^{\not\perp}_{m}$ instead of $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$, it is not clear whether we can prove the KDM-CCA security without a square root loss. In the IND-CCA case, $\mathsf{U}^{\not\perp}_{m}$ provides us with IND-CCA security without a square root security loss [KSS$^+$20, BHH$^+$19]. See Section 1.4 for the detail. We summarize these results in Table 1.

1.3 Related Work

Our work is the first study on KDM secure *PKE in the QROM*. Our work also focuses on *tighter reductions*. Zhang constructs a non-adaptive KDM-CPA *SKE* scheme in the QROM to achieve delegation of quantum computation [Zha19].

Backes, Dürmuth, and Unruh [BDU08] study the KDM security of the OAEP transformation [BR95] in the classical ROM. They prove that OAEP is KDM-secure in the classical ROM if the underlying

[4] We can achieve PKE in the ℓ-user setting with advantage ϵ' from standard PKE with advantage ϵ such that $\epsilon' \approx \ell \cdot \epsilon$.

Table 1: Summary of our results. Here, $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ and $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}}$ denote $\mathsf{U}^{\perp}_{m}$ with OTP and $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ with OTP-then-MAC, respectively. Let ϵ_{Σ} and d_F be the attacker advantage in scheme Σ and the query depth of queries to random oracle F, respectively. Note that $d_F \leq q_F$ where q_F is the number of random oracle queries. We use PKE in the multi-user setting for the building block PKE (denoted by PKE). Open Q. means that it is an open question whether we can achieve KDM-CCA security by using $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}[\mathsf{PKE}_1, H]$ transfromation.

Transformation	Security implication	Security bound	Condition
$\mathsf{PKE}_1 := \mathsf{T}_{\mathtt{HKG}}[\mathsf{PKE}, G]$ (§ 5)	IND-CPA ⇒ SDM-OW-RSA	$O(d_G \cdot \epsilon_{\mathsf{PKE}})$	none
$\mathsf{U}^{\perp}_{m,\mathtt{OTP}}[\mathsf{PKE}_1, H]$ (§ 4)	SDM-OW-RSA ⇒ KDM-CPA	$O(d_H \cdot \epsilon_{\mathsf{PKE}_1})$	none
$\mathsf{U}^{\perp}_{m,\mathtt{OTP}}[T[\mathsf{PKE}, G], H]$ (§ 6)	IND-CPA ⇒ KDM-CPA	$O(d_H \cdot d_G \cdot \epsilon_{\mathsf{PKE}})^{\mathrm{a}}$	none
$\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}[T[\mathsf{PKE}, G], H]$	IND-CPA ⇒ KDM-CPA	$O(d_H \cdot d_G \cdot \epsilon_{\mathsf{PKE}})^{\mathrm{a}}$	none
$\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}}[\mathsf{PKE}_1, H]$ (§ B)	SDM-OW-RSA ⇒ KDM-CCA	$O(d_H \cdot \epsilon_{\mathsf{PKE}_1})$	injectivity
$\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}}[T[\mathsf{PKE}, G], H]$ (§ B&6)	IND-CPA ⇒ KDM-CCA	$O(d_H \cdot d_G \cdot \epsilon_{\mathsf{PKE}})^{\mathrm{a}}$	injectivity
$\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}[\mathsf{PKE}_1, H]$	open Q. ⇒ KDM-CCA	open Q.	open Q.

[a] This is a simplified bound. See Section 6 for the detail.

trapdoor permutation is partial-domain one-way. Note that there is no post-quantum secure trapdoor permutation so far. Davies and Stam [DS14] study the KDM security in the KEM/DEM framework. They prove that if a key derivation function (KDF) is used in between the KEM and DEM part and the KDF function is modelled as a classical random oracle, the resulting PKE scheme is KDM-secure. See the reference for security requirements. Kitagawa, Matsuda, Hanaoka, and Tanaka [KMHT16] prove that the FO transformation [FO13] satisfies KDM-CCA security in the classical ROM.[5] These works studied KDM security in the classical ROM basically prove KDM security by eliminating key dependency of plaintexts by random oracle programming.

We also briefly introduce previous works on IND-CCA secure PKE/KEM in the QROM. Let ϵ and ϵ_{bb} be the advantages of IND-CCA PKE/KEM and the building block, respectively. Let q_H be the number of random oracle queries (and we set $d_H := q_H$ for simplicity). Below, we omit "IND-CCA" and "in the QROM" since all results are about them. We also ignore the differences between FO and FO variants.

Boneh et al. [BDF+11] use a KEM variant of Bellare-Rogaway transformation [BR93] to obtain their KEM from trapdoor functions and $\epsilon \approx q_H \sqrt{\epsilon_{\mathsf{bb}}}$. Targhi and Unruh [TU16] use FO to obtain their PKE from OW-CPA PKE and $\epsilon \approx q_H^{1.5} \sqrt[4]{\epsilon_{\mathsf{bb}}}$. They also use an OAEP variant to obtain their PKE from partial domain trapdoor injective OWFs and $\epsilon \approx \mathrm{poly}(q_H) \sqrt[8]{\epsilon_{\mathsf{bb}}}$. Hofheinz et al. [HHK17] present modular analysis for FO, but their KEM does not improve the construction by Targhi and Unruh. Saito et al. [SXY18] use FO to obtain their KEM from disjoint simulatable deterministic PKE and $\epsilon \approx \epsilon_{\mathsf{bb}}$. They also obtain their KEM from IND-CPA PKE with perfect correctness and $\epsilon \approx q_H \sqrt{\epsilon_{\mathsf{bb}}}$. Jiang, Zhang, Chen, Wang, and Ma [JZC+18] use FO and obtain their KEM from OW-CPA PKE and $\epsilon \approx q_H \sqrt{\epsilon_{\mathsf{bb}}}$. Jiang, Zhang, and Ma [JZM19a] achieve the same bound as those by Jiang et al. [JZC+18] and Saito et al. [SXY18] by using the same assumptions and FO with explicit rejection. Ambainis, Hamburg, and Unruh [AHU19] prove an improved variant of the original O2H lemma (semi-classical O2H lemma) and its bound is $\epsilon_{\mathcal{A}} \approx \sqrt{q_H} \sqrt{\epsilon_{\mathcal{D}}}$ (the query loss is improved). The semi-classical O2H lemma leads to KEM with improved bounds in the query part [AHU19, HKSU20, JZM19b]. Bindel et al. [BHH+19] prove the double-sided O2H lemma whose bound is $\epsilon_{\mathcal{A}} \approx \sqrt{\epsilon_{\mathcal{D}}}$. They use FO to obtain their KEM from IND-CPA PKE with injectivity, but its bound is essentially the same as that of schemes using the semi-classical O2H lemma. Kuchta et al. [KSS+20] prove O2H with MRM and obtain their KEM from

[5] Precisely speaking, the FO transformations studied in the context of QROM are somewhat different from the original FO transformation [FO13].

IND-CPA PKE with injectivity via FO, and $\epsilon \approx q_H^2 \epsilon_{\mathsf{bb}}$.

1.4 Technical Overview

We provide the technical overview of this work. Our goal here is to show that the KDM security in the QROM of the PKE scheme $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{T}(\mathsf{PKE}, G_{\mathsf{enc}}), H)$[6] can be reduced to the IND-CPA security of the underlying PKE without square root security loss. Roughly speaking, the difficulty is that in the setting of KDM security, double-sided O2H lemmas [BHH+19, KSS+20] cannot be applied straightforwardly, which is currently the only tool that enables us to circumvent square root security loss in the QROM.

We first explain how we circumvent square root security loss and prove the KDM security in the QROM of the PKE scheme $\mathsf{U}_{m,\mathtt{OTP}}^{\perp} = \mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{dPKE}, H)$ whose ciphertext is described as

$$(\mathsf{dEnc}(\mathsf{pk}, s), H(s) \oplus m),$$

where dEnc is the encryption algorithm of a deterministic PKE scheme dPKE with the message space $\mathcal{M}$, $s \leftarrow \mathcal{M}$, and H is a random oracle. We identify that the KDM security in the QROM of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}$ can be reduced without square root loss to the security notion of dPKE that we call seed-dependent message one-wayness (SDM-OW security). Then, we explain that the SDM-OW security in the QROM of a tweaked version of $\mathsf{T} = \mathsf{T}(\mathsf{PKE}, G_{\mathsf{enc}})$ can be reduced to the IND-CPA security of the underlying PKE scheme PKE without square root security loss. We call the tweaked version T transformation with hash key generation $\mathsf{T}_{\mathtt{HKG}} = \mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathsf{kg}}, G_{\mathsf{enc}}))$ where G_{kg} and G_{enc} are random oracles. From these facts, we see that the KDM security in the QROM of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathsf{enc}}, G_{\mathsf{kg}})), H)$ can be reduced to the IND-CPA security of PKE without square root security loss. Finally, we state that the KDM security of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{T}(\mathsf{PKE}, G_{\mathsf{enc}}), H)$ immediately follows from the KDM security of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathsf{enc}}, G_{\mathsf{kg}})), H)$.

Below, we start with how to prove the KDM security of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}$ in the classical ROM. For simplicity, in this overview, we consider the following simplified KDM security. Given a ciphertext of $f_b(\mathsf{sk})$, any adversary cannot predict b correctly better than random guessing, where $b \leftarrow \{0, 1\}$ is the challenge bit and f_0 and f_1 are any a-priori fixed two functions. The actual KDM security requires indistinguishability holds for multiple pairs of functions adaptively chosen by an adversary under multiple public and secret key pairs.

KDM security of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}$ in the classical ROM. Let $\mathcal{A}$ be an adversary. $\mathcal{A}$ is given the challenge ciphertext and the random oracle access, which are described as

$$CT : (\mathsf{dEnc}(\mathsf{pk}, s), H(s) \oplus f_b(\mathsf{sk})) \quad \text{and} \quad RO : H(x).$$

We first make a conceptual change to the security game so that the challenge ciphertext and the random oracle are described as

$$CT : (\mathsf{dEnc}(\mathsf{pk}, s), u) \quad \text{and} \quad RO : V(x) = \begin{cases} u \oplus f_b(\mathsf{sk}) & (\text{if } x = s) \\ H(x) & (\text{otherwise}), \end{cases}$$

where u is a uniformly chosen value independent of H and $f_b(\mathsf{sk})$. We can confirm that this is a purely conceptual change since V behaves as a random function and the challenge ciphertext is computed as $(\mathsf{dEnc}(\mathsf{pk}, s), V(s) \oplus f_b(\mathsf{sk})) = (\mathsf{dEnc}(\mathsf{pk}, s), u)$. Therefore, it does not change $\mathcal{A}$'s advantage. Then, we further change the security game so that $\mathcal{A}$ gets access to H instead of V, but the challenge ciphertext is still generated using V. Thus, the challenge ciphertext is not changed from $(\mathsf{dEnc}(\mathsf{pk}, s), u)$. In other

[6] We again note that we use variants of T and U transformations defined by [BHH+19] in this work.

words, except for the generation of the challenge ciphertext, we program the output value of the random oracle at point s from $V(s) = u \oplus f_b(\mathsf{sk})$ into $H(s)$. The view of $\mathcal{A}$ is now

$$CT = (\mathsf{dEnc}(\mathsf{pk}, s), u) \quad \text{and} \quad RO : H(x).$$

We see that in the final game, the challenge bit b is completely hidden from the view of $\mathcal{A}$, and thus $\mathcal{A}$'s advantage is 0. Therefore, we must estimate how much the advantage of $\mathcal{A}$ is changed by the above programming of the random oracle. From the difference lemma[7], this can be bounded by the probability that $\mathcal{A}$ queries s to H in the final security game. In the final game, information of $f_b(\mathsf{sk})$ is completely eliminated from the view of $\mathcal{A}$. Thus, we can use the security of dPKE in order to estimate the probability. Concretely, the probability is estimated by using the OW-CPA security of dPKE. This completes the proof. Of course, square root security loss does not occur in this proof.

KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ in the QROM? When we try to prove KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ in the QROM, we need a different tool from the difference lemma. This is because "the probability that $\mathcal{A}$ queries s to H" is not well-defined in this case since $\mathcal{A}$ can make a query to the random oracle in super-position. In the QROM, in many cases, we can use one-way to hiding (O2H) lemma [Unr15] and its variants [AHU19, BHH$^+$19, KSS$^+$20] as drop-in replacements of the difference lemma in the security proof done in the classical ROM. Roughly speaking, the O2H lemma guarantees that there exists an extractor $\mathcal{D}$ such that the distinguishing gap caused by a programming of a quantumly-accessible random oracle can be bounded by the probability that $\mathcal{D}$ extracts the programmed point. O2H lemma is classified into two categories. The first one is a single-sided O2H lemma where $\mathcal{D}$ gets access to either pre-programmed or post-programmed random oracles. The other one is a double-sided O2H lemma where $\mathcal{D}$ gets access to both of them. In order to circumvent the square root security loss, we currently need to use double-sided O2H lemma proposed in [KSS$^+$20] called O2H with measure-rewind-measure (MRM) lemma.

Suppose to prove KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ in the QROM, we follow the same strategy as the case of the classical ROM (i.e., make a conceptual change and program V into H) and use O2H lemma instead of the difference lemma. Since our goal here is to prove the KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ in the QROM without square root security loss, we use O2H lemma with MRM. By doing so, we can say that there exists a QPT extractor $\mathcal{D}$ such that

$$\left|\Pr\left[1 \leftarrow \mathcal{A}^{|V\rangle}(z)\right] - \Pr\left[1 \leftarrow \mathcal{A}^{|H\rangle}(z)\right]\right| \leq 4d \cdot \Pr\left[s \leftarrow \mathcal{D}^{|V,H\rangle}(z)\right],$$

where $z = (\mathsf{dEnc}(\mathsf{pk}, s), u)$ and d is the query depth of $\mathcal{A}$ to the random oracle.[8] Thus, if we can in turn bound the probability $\Pr\left[s \leftarrow \mathcal{D}^{|V,H\rangle}(z)\right]$ by using the security of the underlying dPKE, we can complete the entire security proof. However, it turns out that it cannot be done straightforwardly using the OW-CPA security of dPKE as before. The reason is that since $\mathcal{D}$ has access to not only H but also V that has information of $f_b(\mathsf{sk})$, it is not clear whether we can use the OW-CPA security of dPKE. Recall that in the proof in the classical ROM case, when estimating "the probability that $\mathcal{A}$ queries s to H" using the OW-CPA security of dPKE, information of $f_b(\mathsf{sk})$ is eliminated from the view of $\mathcal{A}$ since $\mathcal{A}$ does not have access to V.

In summary, in the proof in the classical ROM, we can successfully reduce the KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ to the OW-CPA security of dPKE by eliminating information of $f_b(\mathsf{sk})$ using programming of the random oracle. However, in the case of the QROM, if we use O2H with MRM lemma, it seems difficult to eliminate the information of $f_b(\mathsf{sk})$ by programming the random oracle. This is because we finally

[7] The lemma states that if $\Pr[A \wedge \neg C] = \Pr[B \wedge \neg C]$, $|\Pr[A] - \Pr[B]| \leq \Pr[C]$ holds for any events A,B, and C.

[8] The notation $\mathcal{A}^{|O\rangle}$ indicates that $\mathcal{A}$ is allowed to make a query to O in super-position. Also, for the definition of query depth, see Section 3.

need to handle the extractor $\mathcal{D}$ who gets access to both pre-programmed and post-programmed random oracles.

Note that even if V does not have information of $f_b(\mathsf{sk})$, it might not be clear whether an OW-CPA adversary can simulate two random oracles V and H at the same time for $\mathcal{D}$. The reason is that the differing point s of the two random oracles is the solution of the OW-CPA game itself. This problem can be handled by using the correctness of dPKE. As shown by [LW21], the correctness of dPKE implies that under a randomly generated key $(\mathsf{pk}, \mathsf{sk})$, a randomly generated message m does not have a collision, that is another message m' such that $\mathsf{dEnc}(\mathsf{pk}, m) = \mathsf{dEnc}(\mathsf{pk}, m')$, with overwhelming probability. If $\mathsf{ct} = \mathsf{dEnc}(\mathsf{pk}, s)$ has unique pre-image s, the OW-CPA adversary can check the condition "if $x = s$" by checking "if $\mathsf{dEnc}(\mathsf{pk}, x) = \mathsf{ct}$" (in super-position), thus can simulate V and H at the same time if V does not have information of $f_b(\mathsf{sk})$.

Reduction to SDM-OW security. Although it seems difficult to bound the probability $\Pr\left[s \leftarrow \mathcal{D}^{|V,H\rangle}(z)\right]$ using the OW-CPA security of dPKE, we show that it can be bounded if dPKE satisfies *SDM-OW security* introduced in this work. Hereafter, we assume that the message space $\mathcal{M}$ of dPKE is an abelian group with the operation "+" and the random coin space of the key generation algorithm dKG of dPKE is contained in $\mathcal{M}$. Then, SDM-OW security is a security notion that guarantees that given $(s, \mathsf{dEnc}(\mathsf{pk}, r+s))$, an adversary cannot compute $r+s$, where $s \leftarrow \mathcal{M}$, and $r \in \mathcal{M}$ is the random coin used to generate $(\mathsf{pk}, \mathsf{sk})$ (i.e., $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda; r)$).

The estimation is done after adding the following changes to z and V that do not affect the view of $\mathcal{D}$. First, we replace s in z and V with $r+s$, where $r \in \mathcal{M}$ is the random coin used to generate $(\mathsf{pk}, \mathsf{sk})$. Namely, we change z and V as

$$z = (\mathsf{dEnc}(\mathsf{pk}, r+s), u) \quad \text{and} \quad V(x) = \begin{cases} u \oplus f_b(\mathsf{sk}) & (\text{if } x = r+s) \\ H(x) & (\text{otherwise}). \end{cases} \tag{1}$$

This change does not affect the view of $\mathcal{D}$ since s is chosen uniformly at random and independently of r. Then, we further replace V with the following

$$V(x) = \begin{cases} u \oplus \widehat{f_b}(x) & (\text{if } x = r+s) \\ H(x) & (\text{otherwise}), \end{cases} \tag{2}$$

where $\widehat{f_b}$ is a function that is given x as an input, computes $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda; x-s)$, and outputs $f_b(\mathsf{sk})$. We can check that V in Equation (1) and V in Equation (2) are functionally equivalent. Thus, this change also does not affect the view of $\mathcal{D}$. Moreover, we finally replace the condition "if $x = s + r$" in V with "if $\mathsf{dEnc}(\mathsf{pk}, x) = \mathsf{dEnc}(\mathsf{pk}, r+s)$". As noted before, this can be justified from the correctness of dPKE.

We see that by the above changes, z and V (i.e., the entire view of $\mathcal{D}$) can now be simulated by an SDM-OW adversary $\mathcal{B}$ who is given $(s, \mathsf{dEnc}(\mathsf{pk}, r+s))$. Moreover, $\mathcal{B}$ can break the SDM-OW security if the simulated $\mathcal{D}$ successfully extracts the differing point of V and H, that is, $r+s$. This means that $\Pr\left[s \leftarrow \mathcal{D}^{|V,H\rangle}(z)\right]$ can be bounded by using the SDM-OW security of dPKE.

From the above arguments, we see that the KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ in the QROM can be reduced to the SDM-OW security of dPKE without square root security loss.

SDM-OW security of a variant of T. We next explain the SDM-OW security of $\mathsf{T}_{\mathtt{HKG}} = \mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathsf{kg}}, G_{\mathsf{enc}}))$ can be reduced to the IND-CPA security of the underlying PKE scheme PKE without square roof security loss, where G_{kg} and G_{enc} are random oracles. $\mathsf{T}_{\mathtt{HKG}}$ is a tweaked version of $T = T(\mathsf{PKE}, G_{\mathsf{enc}})$ transformation. T transformation converts a (randomized) IND-CPA secure PKE scheme into an OW-CPA secure deterministic PKE scheme. The encryption algorithm of T is described as $\mathsf{Enc}(\mathsf{pk}, m; G_{\mathtt{enc}}(m))$, where Enc is the encryption algorithm of the underlying PKE. The key generation and decryption algorithms of T are those of PKE themselves. In $\mathsf{T}_{\mathtt{HKG}}$, we also generate a key pair $(\mathsf{pk}, \mathsf{sk})$ by using a random coin generated by the random oracle $G_{\mathtt{kg}}$, that is, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^{\lambda}; G_{\mathtt{kg}}(r))$, where $r \leftarrow \mathcal{M}$.

Bindel et al. [BHH$^+$19] showed that the OW-CPA security of T can be reduced to the IND-CPA security of PKE without square root security loss. The important thing is that the target security notion is one-wayness (not indistinguishability) here. Essentially, Bindel et al. avoided the square root security loss by relying on the fact that if the target security notion is one-wayness and the starting security notion is indistinguishability, we can avoid square root security loss by using *single-sided* O2H lemma called semi-classical O2H lemma [AHU19]. In this work, we show that such a reduction to IND-CPA security without square root loss is possible even when we prove $\mathsf{T}_{\mathtt{HKG}}$'s SDM-OW security, which can be seen as one-wayness for a kind of key dependent messages. In fact, there is no difficulty based on the circularity issue as before since we use single-sided O2H lemma in this step, *not double-sided* one. Roughly speaking, when we use single-sided O2H lemma, we can eliminate correlations between keys, encryption random coins, and plaintexts by random oracle programming in the security proof even in the context of QROM. We give the overview of this proof in Section 5.2. More specifically, we provide a high-level idea of how to solve the correlations after we describe a few hybrid games for the proof, and complete the proof.

The KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}(\mathsf{PKE}, G_{\mathtt{enc}}), H)$. From the discussions so far, we see that the KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathtt{kg}}, G_{\mathtt{enc}})), H)$ can be reduced to the IND-CPA security of PKE without square root security loss. This immediately implies the same holds for $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}(\mathsf{PKE}, G_{\mathtt{enc}}), H)$. This is because adversaries cannot detect whether the public and secret key pair is generated using a random oracle or not. The KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}(\mathsf{PKE}, G_{\mathtt{enc}}), H)$ can be reduced to that of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathtt{kg}}, G_{\mathtt{enc}})), H)$.

Remarks.

- In the actual security game of KDM security, an adversary can choose a pair of functions (f_0, f_1) adaptively and obtain a ciphertext of $f_b(\mathsf{sk})$ multiple times under the existence of multiple key pairs. Also, to capture a wide range of usage scenarios, we allow those functions to get access to random oracles. We handle these issues by using the adaptive reprogramming technique for QROM [Unr14] and introducing a security notion we call SDM-OW-RSA security which is an extension of SDM-OW security.

- Our proof technique is also compatible with KDM-CCA security. Concretely, we can prove the KDM-CCA security of a PKE scheme constructed by using $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m} = \mathsf{U}^{\perp,\mathtt{keyconf}}_{m}(\mathsf{dPKE}, H)$ [BHH$^+$19] as KEM and OTP-then-MAC as DEM without square root security loss. We assume the underlying dPKE is SDM-OW-RSA secure and additionally satisfies injectivity. The security proof is a combination of our proof for the KDM security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ and the proof for the IND-CCA security of $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ by [BHH$^+$19, KSS$^+$20]. Thus, we mainly focus on KDM-CPA security in the main body, and we provide the results on KDM-CCA security in Appendix B.

  As shown by [BHH$^+$19], $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ and $\mathsf{U}^{\not\perp}_{m}$ are IND-CCA secure KEMs that are compatible with double-sided O2H lemma such as O2H lemma with MRM. To use $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ as the KEM part in the above construction is essential. If we use $\mathsf{U}^{\not\perp}_{m}$ as the KEM part, it seems difficult to prove the KDM-CCA security of the construction. $\mathsf{U}^{\not\perp}_{m}$ returns a random value generated by using pseudo-random functions (PRF) if the decryption algorithm detects a given ciphertext is not valid to make it possible to simulate the decryption oracle without using secret keys. In the KDM-CCA security game of a PKE scheme whose KEM part is $\mathsf{U}^{\not\perp}_{m}$, the keys of PRF are also encrypted. In that case, we cannot use the security of PRF and cannot simulate the decryption oracle. It is an interesting open problem to prove KDM-CCA security of a PKE scheme whose KEM part is $\mathsf{U}_m^{\not\perp}$ without square root security loss.

- Our proof strategy explained so far can be realized more easily for SKE where the secret key is used for encryption. A ciphertext of a simple SKE scheme is $(s, H(\mathsf{sk}\|s) \oplus m)$, where H is a random oracle. The simple scheme has a good structure to apply our proof strategy because the secret key sk can be recovered from the differing point $\mathsf{sk}\|s$ when programming the random oracle in the security proof. Zhang [Zha19] showed the non-adaptive KDM security of the SKE scheme with security bound $\sqrt{\frac{\mathrm{poly}(q, q_{\mathtt{kdm}}, q_f, \ell)}{2^\lambda}}$, where q is the number of random oracle queries, $q_{\mathtt{kdm}}$ is the number of KDM queries, q_f is the number of random oracle queries by KDM functions, ℓ is the number of secret keys, and λ is the length of sk. Using our proof strategy, we can prove the non-adaptive KDM security of the SKE scheme with security bound roughly $\frac{\mathrm{poly}(q, q_{\mathtt{kdm}}, q_f, \ell)}{2^\lambda}$. We formally prove it in Appendix C. The proof of this is much easier than the proof of our main construction $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}$. The former can be a warming-up for the latter.
- We do not directly prove the KDM security of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{T}(\mathsf{PKE}, G_{\mathtt{enc}}), H)$, and first prove that of $\mathsf{U}_{m,\mathtt{OTP}}^{\perp}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, (G_{\mathtt{kg}}, G_{\mathtt{enc}})), H)$. If we directly prove the former in a modular way, we think we would need to introduce a more complicated security notion for deterministic PKE schemes. We believe that the introduction of $\mathsf{T}_{\mathtt{HKG}}$ makes our presentation simpler and more modular.
- In this work, we focus on PKE schemes whose DEM is OTP for a technical reason. As we saw above, for our strategy, it is important that DEM has a non-committing property in the sense that we can move an encrypted plaintext from the ciphertext to the key. Although our technique can be used to not only OTP but also any DEM with non-committing property, it is an interesting open question to prove KDM security of FO transformation with any DEM without square root security loss.

2 Preliminaries

2.1 Notations

In this paper, for a finite set X and a distribution D, $x \leftarrow X$ denotes selecting an element from X uniformly at random, $x \leftarrow D$ denotes sampling an element x according to D. Let $y \leftarrow \mathsf{A}(x)$ denotes assigning to y the output of a probabilistic or deterministic algorithm A on an input x. When we explicitly show that A uses randomness r, we write $y \leftarrow \mathsf{A}(x; r)$. When A is allowed to get access to an oracle O, we write $y \leftarrow \mathsf{A}^O(x)$. Let $[a]$ and $[a, b]$ denote the sets of integers $\{1, \cdots, a\}$ and $\{a, \cdots, b\}$, respectively. λ denote a security parameter. PPT and QPT algorithms stand for probabilistic polynomial-time algorithms and polynomial-time quantum algorithms, respectively. Let negl denote a negligible function.

2.2 Public-Key Encryption

A public-key encryption (PKE) scheme PKE is a three tuple $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ of PPT algorithms. Let $\mathcal{M}$ be the message space of PKE. 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 message $m \in \mathcal{M}$, outputs a ciphertext CT. The decryption algorithm Dec, given a secret key sk and ciphertext CT, outputs a message $\tilde{m} \in \{\perp\} \cup \mathcal{M}$.

Definition 2.1 (Correctness of PKE). *We say that* PKE *is* δ*-correct if*

$$\mathbb{E}\left[\max_{m\in\mathcal{M}} \Pr[\mathsf{Dec}(\mathsf{sk},\mathsf{Enc}(\mathsf{pk},m;r)) \neq m] \middle| (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda), r \leftarrow \mathcal{R}\right] \leq \delta \ ,$$

where $\mathcal{R}$ *is the random coin space of* Enc*. If* PKE *is constructed in the random oracle model, the expectation is taken over the choice of* $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$ *and the random oracle.*

We say that PKE is deterministic PKE if $\mathsf{Enc}(\mathsf{pk},\cdot)$ is a deterministic function. We introduce the correctness notion that is specific to deterministic PKE. In addition to the ordinary correctness above, it requires that under a randomly generated key $(\mathsf{pk},\mathsf{sk})$, a randomly generated message m does not have a collision, that is another message m' such that $\mathsf{dEnc}(\mathsf{pk},m) = \mathsf{dEnc}(\mathsf{pk},m')$. This correctness notion is useful when we use double-sided O2H lemmas [BHH+19, KSS+20].

Definition 2.2 (Correctness of deterministic PKE). *We say that a deterministic PKE scheme* $\mathsf{dPKE} = (\mathsf{dKG},\mathsf{dEnc},\mathsf{dDec})$ *with the message space* $\mathcal{M}$ *is* (δ_1,δ_2)*-correct if it is* δ_1*-correct and it holds that*

$$\Pr\left[\exists m' \in \mathcal{M} \ : \ \mathsf{dEnc}(\mathsf{pk},m') = \mathsf{dEnc}(\mathsf{pk},m) | (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda), m \leftarrow \mathcal{M}\right] \leq \delta_2 \ .$$

If dPKE *is constructed in the random oracle model, the probability is taken over the choice of* $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)$*,* $m \leftarrow \mathcal{M}$*, and the random oracle.*

We introduce a multi-instance and multi-challenge version of IND-CPA security for PKE that we denote as IND-m-CPA security.

Definition 2.3 (IND-m-CPA security for PKE). *Let* $\mathsf{PKE} = (\mathsf{KG},\mathsf{Enc},\mathsf{Dec})$ *be a PKE scheme. We define* $\mathsf{Exp}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}}(1^\lambda)$ *for an adversary* $\mathcal{A}$ *as follows.*

Initialize: *First, the challenger chooses a challenge bit* $b \leftarrow \{0,1\}$*. Next, the challenger generates* $(\mathsf{pk}^k,\mathsf{sk}^k) \leftarrow \mathsf{KG}(1^\lambda)$ *for every* $k \in [\ell]$*. The challenger executes* $b' \leftarrow \mathcal{A}^{O_{\mathtt{IND}}}((\mathsf{pk}^k)_{k\in[\ell]})$*.*

$O_{\mathtt{IND}}$**:** *On the i-th call with input* $(k_i, \mathsf{m}_{i,0}, \mathsf{m}_{i,1})$*, where* $k_i \in [\ell]$ *and* $|\mathsf{m}_{i,0}| = |\mathsf{m}_{i,1}|$*, it returns* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}^{k_i}, \mathsf{m}_{i,b})$*.*

Finalize: *The challenger outputs* 1 *if* $b = b'$ *and* 0 *otherwise.*

We say that PKE *is IND-m-CPA secure if for any polynomial* $\ell = \ell(\lambda)$ *and QPT adversary* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}}(1^\lambda)\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$*.*

We introduce the definition of KDM-CPA security for PKE. In the main body of the paper, we mainly focus on KDM-CPA security. We provide the definition of KDM-CCA security in Appendix B.

Definition 2.4 (KDM-CPA security for PKE). *Let* $\mathsf{PKE} = (\mathsf{KG},\mathsf{Enc},\mathsf{Dec})$ *be a PKE scheme. We define* $\mathsf{Exp}^{\mathsf{kdm\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}}(1^\lambda)$ *for an adversary* $\mathcal{A}$ *as follows.*

Initialize: *First, the challenger chooses a challenge bit* $b \xleftarrow{\mathsf{r}} \{0,1\}$*. Next, the challenger generates* $(\mathsf{pk}^k,\mathsf{sk}^k) \leftarrow \mathsf{KG}(1^\lambda)$ *for every* $k \in [\ell]$*. The challenger sets* $\mathbf{sk} := (\mathsf{sk}^1,\ldots,\mathsf{sk}^\ell)$*, and executes* $b' \leftarrow \mathcal{A}^{O_{\mathtt{KDM}}}((\mathsf{pk}^k)_{k\in[\ell]})$*.*

$O_{\mathtt{KDM}}$**:** *On the i-th call with input* $(k_i, f_{i,0}, f_{i,1})$*, where* $k_i \in [\ell]$ *and* $f_{i,0}$ *and* $f_{i,1}$ *are efficiently computable functions with the same output length, it returns* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}^{k_i}, f_{i,b}(\mathbf{sk}))$*.*

Finalize: *The challenger outputs* 1 *if* $b = b'$ *and* 0 *otherwise.*

We say that PKE *is KDM-CPA secure if for any polynomial* $\ell = \ell(\lambda)$ *and QPT adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{kdm\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{kdm\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}}(1^\lambda)\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

Remark 2.5 (KDM security in QROM). In order to capture a wide variety of situations, we allow KDM functions to get access to random oracles if the scheme is constructed in the (quantum) random oracle model. We allow only classical access random oracles for KDM functions, while adversaries get access to random oracles in super-position. This setting is sufficient when honest entities are classical.

2.3 SDM-OW-RSA Security

We introduce a new security notion *seed-dependent message one-wayness against related seed attacks (SDM-OW-RSA security).* This notion plays a crucial role in achieving KDM security from IND-m-CPA security in the QROM without square roof security loss.

Definition 2.6 (SDM-OW-RSA security for PKE). *Let* $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme such that the message space* $\mathcal{M}$ *is an abelian group with the operation* $+$*, and the random coin space of* KG *is* $\mathcal{M}$*. We define* $\mathsf{Exp}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{PKE},\ell,q_{\mathtt{sdm}},\mathcal{A}}(1^\lambda)$ *for an adversary* $\mathcal{A}$ *as follows.*

Initialize: *The challenger first generates* $r \leftarrow \mathcal{M}$*. The challenger then generates* $\Delta^k \leftarrow \mathcal{M}$ *and* $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{KG}(1^\lambda; r + \Delta^k)$ *for every* $k \in [\ell]$*. Next, for every* $k \in [\ell]$ *and* $i \in [q_{\mathtt{sdm}}]$*, the challenger generates* $s_{i,k} \leftarrow \mathcal{M}$ *and computes* $\mathsf{ct}_{i,k} \leftarrow \mathsf{Enc}\left(\mathsf{pk}^k, r + s_{i,k}\right)$*. Finally, the challenger executes* $T \leftarrow \mathcal{A}((\mathsf{pk}^k, \Delta^k)_{k\in[\ell]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i\in[q_{\mathtt{sdm}}],k\in[\ell]})$*.*

Finalize: *The challenger outputs* 1 *if and only if* T *contains* r' *such that* $r' = r + s_{i,k}$ *holds for some* $i \in [q_{\mathtt{sdm}}]$ *and* $k \in [\ell]$*.*

We say that PKE *is SDM-OW-RSA secure if for any polynomial* $\ell = \ell(\lambda)$ *and* $q_{\mathtt{sdm}} = q_{\mathtt{sdm}}(\lambda)$ *and QPT adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{PKE},\ell,\mathcal{A}}(\lambda) = \Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{PKE},\ell,\mathcal{A}}(1^\lambda)\right] = \mathsf{negl}(\lambda).$$

3 Quantum Random Oracle and Useful Lemmas

Given a function $H : X \rightarrow Y$, a quantum-accessible oracle O of H is modeled by a unitary transformation U_H operating on two registers *in* and *out*, in which $|x\rangle\,|y\rangle$ is mapped to $|x\rangle\,|y \oplus H(x)\rangle$, where $\oplus$ denotes XOR group operation on Y. Following [AHU19, BHH+19, KSS+20], we model a quantum algorithm $\mathcal{A}$ making parallel queries to a quantum oracle O as a quantum algorithm making $d \leq q$ queries to an oracle $O^{\otimes n}$ consisting of $n = q/d$ parallel copies of oracle O. Given an input state of n pairs of *in*/*out* registers $|x_1\rangle\,|y_1\rangle \cdots |x_n\rangle\,|y_n\rangle$, the oracle $O^{\otimes n}$ maps it to the state $|x_1\rangle\,|y_1 \oplus H(x_1)\rangle \cdots |x_n\rangle\,|y_n \oplus H(x_n)\rangle$. We call d the algorithm's query depth, n the parallelization factor, and $q = n \cdot d$ the total number of oracle queries. We write $\mathcal{A}^{|O\rangle}$ to denote that the algorithm $\mathcal{A}$'s oracle O is a quantum-accessible oracle.

Simulation of quantum random oracles. In this paper, following many previous works in the QROM, we give quantum-accessible random oracles to reduction algorithms if needed. This is just a convention. We can efficiently simulate quantum-accessible random oracles perfectly by using $2q$-wise independent hash function [Zha12], where q is the number of queries to the quantum-accessible random oracles by an adversary.

3.1 One-Way to Hiding (O2H) Lemma

Definition 3.1 (Punctured oracle). *Let $F : X \to Y$ be any function, and $S \subset X$ be a set. The oracle $F \setminus S$("F punctured by S") takes as input a value $x \in X$. It first computes whether $x \in S$ into an auxiliary register and measures it. Then it computes $F(x)$ and returns the result. Let* $\mathtt{Find}$ *be the event that any of the measurements returns* 1.

Lemma 3.2 (Semi-classical O2H [AHU19, Theorem 1]). *Let $G, H : X \to Y$ be random functions, z be a random value, and $S \subseteq X$ be a random set such that $G(x) = H(x)$ for every $x \notin S$. The tuple (G, H, S, z) may have arbitrary joint distribution. Furthermore, let $\mathcal{A}$ be a quantum oracle algorithm. Let* $\mathtt{Ev}$ *be any classical event. Then we have*

$$\left|\sqrt{\Pr[\mathtt{Ev} : \mathcal{A}^{|G\rangle}(z)]} - \sqrt{\Pr[\mathtt{Ev} \wedge \neg\mathtt{Find} : \mathcal{A}^{|H\setminus S\rangle}(z)]}\right| \leq \sqrt{(d+1) \cdot \Pr[\mathtt{Find} : \mathcal{A}^{|H\setminus S\rangle}(z)]} \ ,$$

where d is the query depth of $\mathcal{A}$ for G and $H \setminus S$.

Lemma 3.3 (Search in semi-classical oracle [AHU19, Theorem 2]). *Let $H : X \to Y$ be a random function, let z be a random value, and let $S \subset X$ be a random set. (H, S, z) may have arbitrary joint distribution. Let $\mathcal{A}$ be a quantum oracle algorithm. If for each $x \in X$, $\Pr[x \in S] \leq \epsilon$ (conditioned on H and z), then we have*

$$\Pr\left[\mathtt{Find} : \mathcal{A}^{|H\setminus S\rangle}(z)\right] \leq 4q\epsilon \ ,$$

where q is the number of queries to $H \setminus S$ by $\mathcal{A}$.

Note that the above lemma is originally introduced in [AHU19], but we use a variant that is closer to Lemma 4 in [BHH$^+$19].

Lemma 3.4 (Adapted version of O2H with MRM [KSS$^+$20, Lemma 3.3]). *Let $G, H : X \to Y$ be functions, and $S \subseteq X$ be a set such that $G(x) = H(x)$ for every $x \notin S$. Also, let z be a value and $O_{\mathtt{aux}}$ be a function. The tuple $(G, H, S, z, O_{\mathtt{aux}})$ may have arbitrary joint distribution. Furthermore, let $\mathcal{A}$ be a quantum oracle algorithm. Then we can construct an algorithm $\mathcal{D}$ such that*

- *The running time of $\mathcal{D}$ is roughly three times longer than that of $\mathcal{A}$. Moreover, if $\mathcal{A}$ makes at most q queries to G and H with query depth d, $\mathcal{D}$ makes at most $O(q)$ queries to each of those oracles with query depth $O(d)$, and outputs a list $T \subseteq X$ of size at most $O(q)$.*
- *It holds that*

$$\left|\Pr\left[1 \leftarrow \mathcal{A}^{|G,O_{\mathtt{aux}}\rangle}(z)\right] - \Pr\left[1 \leftarrow \mathcal{A}^{|H,O_{\mathtt{aux}}\rangle}(z)\right]\right| \leq 4d \cdot \Pr\left[T \cap S \neq \emptyset : T \leftarrow \mathcal{D}^{|G,H,O_{\mathtt{aux}}\rangle}(z)\right] \ ,$$

where d is the query depth of $\mathcal{A}$ for the first oracle.

Remark 3.5 (On the difference from the original version). There are some differences between Lemma 3.4 and the original O2H lemma with MRM [KSS$^+$20, Lemma 3.3]. First, in Lemma 3.4, we allow the algorithm $\mathcal{A}$ to get access to an additional oracle $O_{\mathtt{aux}}$, which is not explicitly appeared in the original version. Second, in Lemma 3.4, we explicitly state the size of $\mathcal{D}$'s output T is at most $O(q)$ while the original lemma does not refer to the size of T. For the first one, it is easy to see that even if we introduce such an additional oracle, the lemma still holds. (This extension is used in also [LW21].) For the second, the concrete extractor $\mathcal{D}$ constructed in [KSS$^+$20] satisfies this condition. Since we need the upper bound on the size of T in order to estimate the security bound in our proof, we place the requirement.

3.2 Additional Lemma

Lemma 3.6 ([Unr14, Lemma 13 in the eprint version]). *Let $\delta_x(\cdot)$ be a point function that outputs 1 if and only if it is given x. Let $\delta_\perp(\cdot)$ be the constant function that outputs 0 for all inputs. Let $\mathcal{A}$ be an oracle QPT algorithm making at most q queries. Let ρ_0 denote the final state of $\mathcal{A}$ together with x in the following experiment: Pick $x \leftarrow \mathcal{M}$ and run $\mathcal{A}^{|\delta_x\rangle}$. Let ρ_1 denote the final state of $\mathcal{A}$ together with x in the following experiment: Pick $x \leftarrow \mathcal{M}$ and run $\mathcal{A}^{|\delta_\perp\rangle}$. Then, we have $\|\rho_0 - \rho_1\|_{\mathsf{tr}} \leq \frac{2q}{\sqrt{|\mathcal{M}|}}$.*

Using Lemma 3.6, we can prove the following lemma which is a multi-point version of adaptive reprogramming of QRO used in the proof of adaptive O2H lemma [Unr14, Lemma 14 in the eprint version]. The following lemma is needed to handle KDM queries that are made adaptively.

Lemma 3.7 (Adaptive reprogramming of QRO). *We consider the following $\mathsf{Exp}^{\mathsf{adp\text{-}prog}}_{q_{\mathtt{prog}},\mathcal{A}}(1^\lambda)$.*

Initialization *The challenger first generates the challenge bit $b \leftarrow \{0,1\}$ and a fresh random oracle $V_0 : X \to Y$. Then, the challenger executes $b' \leftarrow \mathcal{A}^{|V_0\rangle, O_{\mathtt{prog}}}(1^\lambda)$, where $O_{\mathtt{prog}}$ is defined as follows.*

$O_{\mathtt{prog}}$: *On the i-th call, it first generates $s_i \leftarrow X$. If $b = 0$, it just returns $(s_i, V_0(s_i))$. Otherwise, it generates $u_i \leftarrow Y$, updates the random oracle $\mathcal{A}$ gets access into V_i defined as*

$$V_i(x) = \begin{cases} u_j & (\textit{if } \ x = s_j \ \textit{ holds for some } \ j \leq i) \\ H(x) & (\textit{otherwise}), \end{cases}$$

and returns $(s_i, V_i(s_i)) = (s_i, u_i)$.

Finalization *The challenger outputs 1 if $b = b'$ and 0 otherwise.*

Then, for any integer $q_{\mathtt{prog}}$ and an oracle algorithm $\mathcal{A}$ that makes at most q queries to O_{b}, we have $\left|\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{adp\text{-}prog}}_{q_{\mathtt{prog}},\mathcal{A}}(1^\lambda)\right] - \frac{1}{2}\right| \leq \frac{2q \cdot q_{\mathtt{prog}}}{\sqrt{|X|}}$.

Proof. We consider the following intermediate games for $i^* \in [0, q_{\mathtt{prog}}]$.

Game i^*: This is the same as $\mathsf{Exp}^{\mathsf{adp\text{-}prog}}_{\mathcal{A}}(1^\lambda)$ where $b = 0$ except that $O_{\mathtt{prog}}$ behaves as follows.

$O_{\mathtt{prog}}$: On the i-th call, it generates $s_i \leftarrow X$. If $i > i^*$, it returns $(s_i, V_{i^*}(s_i))$. Otherwise, it also generates $u_i \leftarrow Y$, updates the oracle $\mathcal{A}$ gets access into V_i defined as

$$V_i(x) = \begin{cases} u_j & (\text{if } \ \exists j \leq i \ : \ x = s_j) \\ V_0(x) & (\text{otherwise}) \end{cases}$$

and returns $(s_i, V_i(s_i)) = (s_i, u_i)$.

$\overline{\textbf{Game}}$ i^*: This game is the same as Game i^* except that for $i \leq i^*$, V_i is replaced with $V_i\{s_{i^*+1}\}$ defined as

$$V_i\{s_{i^*+1}\}(x) = \begin{cases} \perp & (\text{if } \ x = s_{i^*+1}) \\ u_j & (\text{if } \ \exists j \leq i \ : \ x = s_j) \\ V_0(x) & (\text{otherwise}) \end{cases} .$$

$\widehat{\textbf{Game}}$ i^***:** This game is the same as $\overline{\text{Game}}$ i^* except that on the $i^* + 1$-th call, $O_{\texttt{prog}}$ updates the oracle $\mathcal{A}$ gets access into V_{i^*+1} defined as

$$V_{i^*+1}(x) = \begin{cases} u_j & (\text{if } \exists j \le i^* + 1 \ : \ x = s_j) \\ V_0(x) & (\text{otherwise}) \end{cases}$$

and returns $(s_{i^*+1}, V_{i^*+1}(s_{i^*+1})) = (s_{i^*+1}, u_{i^*+1})$, where $u_{i^*+1} \leftarrow Y$. Also, on the i-th call for $i > i^* + 1$, $O_{\texttt{prog}}$ returns $(s_i, V_{i^*+1}(s_i))$.

Let $\texttt{ONE}_X, \overline{\texttt{ONE}}_X, \widehat{\texttt{ONE}}_X$ be the event that $\mathcal{A}$ outputs 1 as the final output in Game X, $\overline{\text{Game}}$ X, and $\widehat{\text{Game}}$ X, respectively. Game 0 (resp. Game $q_{\texttt{prog}}$) is exactly the same as $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{adp\text{-}prog}}(1^\lambda)$ where $b = 0$ (resp. $b = 1$). Thus, we have

$$\begin{aligned}
&\left|\Pr\left[1 \leftarrow \mathsf{Exp}_{q_{\texttt{prog}},\mathcal{A}}^{\mathsf{adp\text{-}prog}}(1^\lambda)\right] - \frac{1}{2}\right| \\
&\le \frac{1}{2}\left|\Pr[\texttt{ONE}_0] - \Pr\left[\texttt{ONE}_{q_{\texttt{prog}}}\right]\right| \\
&\le \frac{1}{2} \sum_{i \in [0, q_{\texttt{prog}} - 1]} \left|\Pr[\texttt{ONE}_i] - \Pr[\texttt{ONE}_{i+1}]\right| \\
&\le \frac{1}{2} \sum_{i \in [0, q_{\texttt{prog}} - 1]} \left|\Pr[\texttt{ONE}_i] - \Pr[\overline{\texttt{ONE}}_i]\right| + \left|\Pr[\overline{\texttt{ONE}}_i] - \Pr\left[\widehat{\texttt{ONE}}_i\right]\right| + \left|\Pr\left[\widehat{\texttt{ONE}}_i\right] - \Pr[\texttt{ONE}_{i+1}]\right| \ .
\end{aligned}$$

We show that $\left|\Pr[\texttt{ONE}_{i^*}] - \Pr[\overline{\texttt{ONE}}_{i^*}]\right| \le \frac{2q}{\sqrt{|X|}}$ for every $i^* \in [0, q_{\texttt{prog}} - 1]$ using Lemma 3.6. Consider the following oracle algorithm $\mathcal{B}$ that has oracle access to a function $F : X \to \{0, 1\}$.

$\mathcal{B}$**:** $\mathcal{B}$ first generates a fresh random oracle $V_0 : X \to Y$. Then, $\mathcal{B}$ executes $\mathcal{A}^{|V_0'\rangle, O_{\texttt{prog}}}(1^\lambda)$ just before $\mathcal{A}$ makes the $i^* + 1$-th query to $O_{\texttt{prog}}$, where V_0' is defined as

$$V_0'(x) = \begin{cases} \bot & (\text{if } F(x) = 1) \\ V_0(x) & (\text{otherwise}) \end{cases}$$

and $O_{\texttt{prog}}$ is simulated as follows.

$O_{\texttt{prog}}$**:** On the i-th call, it generates $s_i \leftarrow X$. It also generates $u_i \leftarrow Y$, updates the oracle $\mathcal{A}$ can get access into V_i' defined as

$$V_i'(x) = \begin{cases} \bot & (\text{if } F(x) = 1) \\ u_j & (\text{if } \exists j \le i \ : \ x = s_j) \\ V_0(x) & (\text{otherwise}) \end{cases}$$

and returns $(s_i, V_i(s_i)) = (s_i, u_i)$.

If F equals $\delta_\bot$ (that is, a constant function that outputs 0 for all inputs), then V_i' is functionally equivalent to V_i. If F equals $\delta_{s_{i^*+1}}$ (that is, a point function that outputs 1 only for s_{i^*}), then V_i' is functionally equivalent to $V_i\{s_{i^*+1}\}$. $\mathcal{B}$ simulates Game i^* (resp. $\overline{\text{Game}}$ i^*) for $\mathcal{A}$ just before $\mathcal{A}$ makes the $i^* + 1$-th query to $O_{\texttt{prog}}$ if $F = \delta_\bot$ (resp. $F = \delta_{s_{i^*+1}}$). Thus, from Lemma 3.6, the trace distance between the internal state of $\mathcal{A}$ in Game i^* and that in $\overline{\text{Game}}$ i^* at the point just before $\mathcal{A}$ makes $i^* + 1$-th query to $O_{\texttt{prog}}$ can be bounded by $\frac{2q}{\sqrt{|X|}}$. Moreover, the remaining procedures of Game i^* and $\overline{\text{Game}}$ i^* are exactly the same. Therefore, we obtain $\left|\Pr[\texttt{ONE}_{i^*}] - \Pr[\overline{\texttt{ONE}}_{i^*}]\right| \le \frac{2q}{\sqrt{|X|}}$.

Similarly to $|\Pr[\mathtt{ONE}_{i^*}] - \Pr[\overline{\mathtt{ONE}}_{i^*}]|$, we can obtain $\left|\Pr\left[\widehat{\mathtt{ONE}}_{i^*}\right] - \Pr[\mathtt{ONE}_{i^*+1}]\right| \leq \frac{2q}{\sqrt{|X|}}$. Also, we can see that the difference between $\overline{\mathsf{Game}}\ i^*$ and $\widehat{\mathsf{Game}}\ i^*$ is only conceptual, and thus we have $\left|\Pr[\overline{\mathtt{ONE}}_{i^*}] - \Pr\left[\widehat{\mathtt{ONE}}_{i^*}\right]\right| = 0$.

Overall, we obtain $\left|\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{adp\text{-}prog}}_{q_{\mathrm{prog}},\mathcal{A}}(1^\lambda)\right] - \frac{1}{2}\right| \leq \frac{2q \cdot q_{\mathrm{prog}}}{\sqrt{|X|}}$. □ (**Lemma 3.7**)

4 KDM-CPA Security of $\mathsf{U}^{\not\perp}_m$ with OTP as DEM

In this section, we show that the KDM-CPA security in the QROM of a PKE scheme $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}} = \mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}(\mathsf{dPKE}, H)$ can be reduced to the SDM-OW-RSA security of the underlying dPKE without square root security loss. $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}$ is constructed by using $\mathsf{U}^{\not\perp}_m(\mathsf{dPKE}, H)$ [BHH+19] as KEM and OTP as DEM. Since we focus on KDM-*CPA* security here, $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}$ omits the ciphertext validity check by re-encryption in the decryption algorithm, which is performed in $\mathsf{U}^{\not\perp}_m$. For the construction of $\mathsf{U}^{\not\perp}_m$, see Appendix A.

4.1 Construction

Construction 4.1. Let $\mathsf{dPKE} = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ be a deterministic PKE scheme whose message space is $\mathcal{M}$. We assume that $\mathcal{M}$ is an abelian group and denote the operation in $\mathcal{M}$ as $+$. Let $H : \mathcal{M} \to \{0,1\}^*$ be a hash function. We construct $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ as follows.

$\mathsf{KG}(1^\lambda)$: Return $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)$.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{m})$: Generate $s \leftarrow \mathcal{M}$ and compute $\mathsf{ct} \leftarrow \mathsf{dEnc}(\mathsf{pk}, s)$ and $t = H(s) \oplus \mathsf{m}$. Return $\mathsf{CT} = (\mathsf{ct}, t)$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}')$: Parse $\mathsf{CT}' = (\mathsf{ct}', t')$, compute $s' \leftarrow \mathsf{dDec}(\mathsf{sk}, \mathsf{ct}')$, and return $\perp$ if $s' = \perp$. Otherwise, return $t' \oplus H(s')$.

We see that if dPKE is (δ_1, δ_2)-correct, then $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}$ is δ_1-correct for any δ_1.

4.2 Security Proof

We prove the following theorem.

Theorem 4.2. *Let $\ell = \ell(\lambda)$ be a polynomial and dPKE be a (δ_1, δ_2)-correct deterministic PKE. Let $\mathcal{A}$ be a QPT adversary against the KDM-CPA security of $\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}} = \mathsf{U}^{\not\perp}_{m,\mathtt{OTP}}(\mathsf{dPKE}, H)$ making q (superposition) random oracle queries to H with query depth d and $q_{\mathtt{kdm}}$ (classical) queries to $\mathsf{O}_{\mathtt{KDM}}$. Also, let q_f be the upper bound of the total number of (classical) random oracle queries made by KDM functions. Then, there exists a QPT adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{kdm\text{-}cpa}}_{\mathsf{U}^{\not\perp}_{m,\mathtt{OTP}},\ell,\mathcal{A}}(1^\lambda) \leq 4d \cdot \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE},\ell,q_{\mathtt{kdm}},\mathcal{B}}(1^\lambda) + \frac{4(q+q_f)q_{\mathtt{kdm}}}{\sqrt{|\mathcal{M}|}} + (4d+1) \cdot q_{\mathtt{kdm}} \cdot \delta_2\ . \tag{3}$$

Proof. We complete the proof using hybrid games. Let $\mathtt{SUC}_X$ be the event that the final output is 1 in Game X. We assume that $\mathcal{A}$ makes at least one KDM query before the first set of random oracle queries and between d^*-th set of random oracle queries and (d^*+1)-th set of random oracle queries for every $d^* \in [d-1]$. This assumption is without loss of generality in the sense that any adversary can be transformed into one satisfying this condition without changing the number and depth of random oracle queries.

Game 1: This is $\mathsf{Exp}^{\mathsf{kdm\text{-}cpa}}_{\mathsf{U}^{\perp}_{m,\mathtt{OTP}},\ell,\mathcal{A}}(1^\lambda)$.

> **Initialize:** First, the challenger chooses a challenge bit $b \leftarrow \{0,1\}$. The challenger also generates a fresh random oracle H. Next, the challenger generates $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda)$ for every $k \in [\ell]$. The challenger sets $\mathbf{sk} := (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$ and $\mathbf{pk} := (\mathsf{pk}^1, \ldots, \mathsf{pk}^\ell)$, and executes $b' \leftarrow \mathcal{A}^{|H\rangle, O_{\mathtt{KDM}}}(\mathbf{pk})$. $O_{\mathtt{KDM}}$ behaves as follows.
>
> $O_{\mathtt{KDM}}$**:** On the i-th call with input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.
>
> 1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, s_i)$.
> 2. Compute $t_i = H(s_i) \oplus f_{i,b}^H(\mathbf{sk})$.
> 3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.
>
> **Finalize:** The challenger outputs 1 if $b = b'$ and 0 otherwise.

Game 2: This is the same as Game 1 except the behavior of $O_{\mathtt{KDM}}$. In this game, $O_{\mathtt{KDM}}$ adaptively reprograms the random oracle that $\mathcal{A}$ (and functions queried by $\mathcal{A}$) gets access every time it is invoked. The detailed description is as follows.

> $O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.
>
> 1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, s_i)$.
> 2. $\underline{\text{Generate } u_i \leftarrow \{0,1\}^* \text{ and compute } t_i = u_i \oplus f_{i,b}^{V_{i-1}}(\mathbf{sk})}$.
> 3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.
>
> Also, it updates the random oracle into
>
> $$V_i(x) = \begin{cases} \underline{u_j} & \underline{(\text{if } \exists j \leq i \ : \ x = s_j)} \\ H(x) & (\text{otherwise}), \end{cases}$$

From Lemma 3.7, we have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| = \frac{4(q+q_f)q_{\mathtt{kdm}}}{\sqrt{\mathcal{M}}}$.

Game 3: This game is the same as Game 2 except that u_i is replaced with $u_i \oplus f_{i,b}^{V_{i-1}}(\mathbf{sk})$ for every $i \in [q_{\mathtt{kdm}}]$. More concretely, the behavior of $O_{\mathtt{KDM}}$ is changed as follows.

> $O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.
>
> 1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, s_i)$.
> 2. Generate $u_i \leftarrow \{0,1\}^*$ and $\underline{\text{set } t_i \leftarrow u_i}$.
> 3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.
>
> Also, it updates the random oracle into
>
> $$V_i(x) = \begin{cases} \underline{u_j \oplus f_{j,b}^{V_{j-1}}(\mathbf{sk})} & (\text{if } \exists j \leq i \ : \ x = s_j) \\ H(x) & (\text{otherwise}), \end{cases}$$

This change does not affect the view of $\mathcal{A}$ since u_i is chosen uniformly at random and independently of $f_{i,b}^{V_{i-1}}(\mathbf{sk})$ for every $i \in [q_{\mathtt{kdm}}]$. Thus, we have $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| = 0$.

Game 4: This game is the same as Game 3 except for the following. The challenger first generates $r \leftarrow \mathcal{M}$. The challenger then generates $\Delta^1, \ldots, \Delta^\ell \leftarrow \mathcal{M}$ and generates $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda; r + \Delta^k)$ for every $k \in [\ell]$.

```
$\widehat{f}^H_{i,b}\left[(s_j, u_j, f_{j,b})_{j\in[i]}, (\Delta^k)_{k\in[\ell]}\right](x)$ :

Hardwired: $(s_j, u_j, f_{j,b})_{j\in[i]}, (\Delta^k)_{k\in[\ell]}$.
Oracle $H$.
Input: $x \in \mathcal{M}$.
    1. Compute $w = x - s_i$ and $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda; w + \Delta^k)$ for every $k \in [\ell]$, and set $\mathbf{sk} = (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$.
    2. Repeat the following from $j = 1$ to $i - 1$, where $\widehat{V}_0 = H$.
        (a) Compute $v_j = u_j \oplus f_{j,b}^{\widehat{V}_{j-1}}(\mathbf{sk})$.
        (b) Set $\widehat{V}_j$ as
            $$\widehat{V}_j(x') = \begin{cases} v_{j'} & (\text{if } \exists j' \in [q_{\mathtt{kdm}}] : j' \le j \text{ and } x' = w + s_{j'}) \\ H(x') & (\text{otherwise}). \end{cases}$$
    3. Return $f_{i,b}^{\widehat{V}_{i-1}}(\mathbf{sk})$.
```

Figure 1: The description of $\widehat{f}^H_{i,b}$.

The above change does not affect the view of $\mathcal{A}$ since the distribution of $(\mathsf{pk}^k, \mathsf{sk}^k)_{k\in[\ell]}$ does not change. Thus, we have $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| = 0$.

Game 5: This game is the same as Game 4 except that s_i is replaced with $r + s_i$. More concretely, the challenger generates ct_i as $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, \underline{r + s_i})$ for every $i \in [q_{\mathtt{kdm}}]$. Also, the challenger sets V_i as

$$V_i(x) = \begin{cases} u_j \oplus f_{j,b}^{V_{j-1}}(\mathbf{sk}) & (\text{if } \exists j \le i : \underline{x = r + s_j}) \\ H(x) & (\text{otherwise}) \end{cases}$$

for every $i \in [q_{\mathtt{kdm}}]$.

We have $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| = 0$ since this change also does not affect the view of $\mathcal{A}$.

From the next game, we use the function $\widehat{f}_{i,b}$ described in Figure 1. $\widehat{f}_{i,b}$ is designed so that it computes $f_{i,b}^{V_{i-1}}(\mathbf{sk})$ if it has oracle access to H and is given $r + s_i$ as an input. For this aim, $\widehat{f}^H_{i,b}$ sequentially computes V_j from $V_1, V_2, ..., V_{i-1}$ using H. They are denoted as $\widehat{V}_j$ in the description of $\widehat{f}^H_{i,b}$. Here, the computation of $\widehat{V}_j$ by $\widehat{f}^H_{j,b}$ is local, and thus $\widehat{f}^H_{j,b}$ does not perform the updates of the random oracle that $\mathcal{A}$ gets access.

Game 6: For every $i \in [q_{\mathtt{kdm}}]$, we define a function . Then, Game 6 is the same as Game 5 except that the challenger sets V_i as

$$V_i(x) = \begin{cases} u_j \oplus \underline{\widehat{f}^H_{j,b}(x)} & (\text{if } \exists j \le i : x = r + s_j) \\ H(x) & (\text{otherwise}) \end{cases}$$

for every $i \in [q_{\mathtt{kdm}}]$.

Since $\widehat{f}_{i,b}$ correctly computes $f_{i,b}^{V_{i-1}}(\mathbf{sk})$ if it has oracle access to H and is given $r + s_i$ as an input for every $i \in [q_{\mathtt{kdm}}]$, the functionality of V_i does not change between Game 5 and 6 for every $i \in [q_{\mathtt{kdm}}]$. Therefore, we have $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = 0$.

Game 7: This game is the same as Game 6 except that for every $i \in [q_{\mathtt{kdm}}]$, V_i is defined as

$$V_i(x) = \begin{cases} u_j \oplus \widehat{f}^H_{j,b}(x) & (\text{if } \exists j \leq i \ : \ \underline{\mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j}) \\ H(x) & (\text{otherwise}). \end{cases}$$

If ct_i has a unique pre-image $r + s_i$ under pk^{k_i} for every $i \in [q_{\mathtt{kdm}}]$, the functionality of V_i does not change for every $i \in [q_{\mathtt{kdm}}]$ between Game 6 and 7. Thus, from the correctness of dPKE, we have $|\Pr[\mathtt{SUC}_6] - \Pr[\mathtt{SUC}_7]| \leq q_{\mathtt{kdm}} \cdot \delta_2$.

At Game 7, $\mathcal{A}$ can obtain information of the challenge bit b only through d sets of random oracle queries. Below, we use d more hybrid games and remove information of b from those d sets of random oracle queries one by one.

Game $7 + d^*$ $(d^* = 1, \ldots, d)$**:** This is the same game as Game 7 except $O_{\mathtt{KDM}}$ defers updating the random oracle. Concretely, $O_{\mathtt{KDM}}$ does not update the random oracle until $\mathcal{A}$ makes the d^*-th set of random oracle queries. The detailed description of $O_{\mathtt{KDM}}$ is as follows.

$O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, r + s_i)$.
2. Generate $u_i \leftarrow \{0,1\}^*$ and set $t_i \leftarrow u_i$.
3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.

Also, if $\mathcal{A}$ already makes d^*-th set of queries to the random oracle, it updates the random oracle into

$$V_i(x) = \begin{cases} u_j \oplus \widehat{f}^H_{j,b}(x) & (\text{if } \exists j \leq i \ : \ \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ H(x) & (\text{otherwise}). \end{cases}$$

We have $\left|\Pr[\mathtt{SUC}_{7+d}] - \frac{1}{2}\right| = 0$ since in Game $7 + d$, the view of $\mathcal{A}$ is completely independent of b. In order to estimate $|\Pr[\mathtt{SUC}_{7+d^*-1}] - \Pr[\mathtt{SUC}_{7+d^*}]|$ for every $d^* \in [d]$, we consider the following procedure Setup_{d^*}.

Setup_{d^*}**:** First, the challenger chooses a challenge bit $b \leftarrow \{0,1\}$. The challenger also generates a fresh random oracle H. Next, the challenger generates $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda; r + \Delta^k)$, where $r \leftarrow \mathcal{M}$ and $\Delta^k \leftarrow \mathcal{M}$ for every $k \in [\ell]$. The challenger sets $\mathbf{pk} := (\mathsf{pk}^1, \ldots, \mathsf{pk}^\ell)$, and executes $\mathcal{A}^{|H\rangle, O_{\mathtt{KDM}}}(\mathbf{pk})$ just before $\mathcal{A}$ makes the d^*-th set of random oracle queries. $O_{\mathtt{KDM}}$ behaves as follows.

$O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, r + s_i)$.
2. Generate $u_i \leftarrow \{0,1\}^*$ and set $t_i \leftarrow u_i$.
3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.

Let $\mathcal{A}$ makes i^* KDM queries before d^*-th set of random oracle queries. Then, the challenger sets V_{i^*} as

$$V_{i^*}(x) = \begin{cases} u_j \oplus \widehat{f}^H_{j,b}(x) & (\text{if } \exists j \leq i^* \ : \ \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ H(x) & (\text{otherwise}) \end{cases}$$

and $S_{i^*} = \{x | \exists j \in [i^*] \ : \ \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j\}$. The challenger also generates $s_{i,k} \leftarrow \mathcal{M}$ and generates $\mathsf{ct}_{i,k} \leftarrow \mathsf{dEnc}(\mathsf{pk}^k, r + s_{i,k})$ for every $i \in [i^* + 1, q_{\mathtt{kdm}}]$ and $k \in [\ell]$. The challenger then sets

$$z = (|st\rangle, b, \mathbf{pk}, (\Delta^k)_{k \in [\ell]}, (k_i, f_{i,b}, s_i, \mathsf{ct}_i, u_i)_{i \in [i^*]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i \in [i^*+1, q_{\mathtt{kdm}}], k \in [\ell]}) \ , \tag{4}$$

where $|st\rangle$ is the internal state of $\mathcal{A}$ at this point. The challenger outputs $(V_{i^*}, H, S_{i^*}, z, O_{\mathtt{aux}} = H)$.

Also, we consider the following QPT algorithm $\mathcal{A}_{d^*}$ that has oracle access to $O \in \{V_{i^*}, H\}$ and $O_{\mathsf{aux}} = H$.

$\mathcal{A}_{d^*}$**:** Given an input z, $\mathcal{A}_{d^*}$ parse it as Equation (4) and executes $\mathcal{A}^{|O\rangle, O_{\mathtt{KDM}}}$ from $\mathcal{A}$'s d^*-th set of random oracle queries using $|st\rangle$ as the internal state of $\mathcal{A}$ at that point. $\mathcal{A}_{d^*}$ simulates $O_{\mathtt{KDM}}$ as follows.

$O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Set $\mathsf{ct}_i \leftarrow \mathsf{ct}_{i,k_i}$ (and set $s_i \leftarrow s_{i,k_i}$).
2. Generate $u_i \leftarrow \{0,1\}^*$ and set $t_i \leftarrow u_i$.
3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.

Also, it updates the random oracle that $\mathcal{A}$ gets access into

$$V_i(x) = \begin{cases} u_j \oplus \widehat{f}_{j,b}^H(x) & (\text{if } \exists j \leq i \ : \ \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ H(x) & (\text{otherwise}). \end{cases}$$

When $\mathcal{A}$ terminates with output b', $\mathcal{A}_{d^*}$ outputs 1 if $b = b'$ and 0 otherwise.

Suppose we execute Setup_{d^*} and $\mathcal{A}_{d^*}$ successively. They simulate the view of $\mathcal{A}$ in Game $7 + d^* - 1$ (resp. Game $7 + d^*$) if $O = V_{i^*}$ (resp. $O = H$). Also, $\mathcal{A}_{d^*}$ outputs 1 if and only if the output of the simulated games is 1. Thus, we have $\Pr[\mathtt{SUC}_{7+d^*-1}] = \Pr\left[1 \leftarrow \mathcal{A}_{d^*}^{|O=V_{i^*}, O_{\mathsf{aux}}=H\rangle}(z) : \mathsf{Setup}_{d^*}\right]$ and $\Pr[\mathtt{SUC}_{7+d^*}] = \Pr\left[1 \leftarrow \mathcal{A}_{d^*}^{|O=H, O_{\mathsf{aux}}=H\rangle}(z) : \mathsf{Setup}_{d^*}\right]$. From Lemma 3.4, there exists a QPT algorithm $\mathcal{D}_{d^*}$ such that

$$|\Pr[\mathtt{SUC}_{7+d^*-1}] - \Pr[\mathtt{SUC}_{7+d^*}]| \leq 4 \cdot \Pr\left[T \cap S_{i^*} \neq \emptyset \,\middle|\, T \leftarrow \mathcal{D}_{d^*}^{|V_{i^*}, H, O_{\mathsf{aux}}=H\rangle}(z), \mathsf{Setup}_{d^*}\right].$$

Note that $\mathcal{A}_{d^*}$ makes queries to $O \in \{V_{i^*}, H\}$ with depth 1 by the following reason. $\mathcal{A}_{d^*}$ is supposed to simulate Game $7 + d^* - 1$ (resp. Game $7 + d^*$) for $\mathcal{A}$ from the point that $\mathcal{A}$ makes d^*-th set of random oracle queries when $\mathcal{A}_{d^*}$ gets access to $O = V_{i^*}$ (resp. $O = H$). The answers to $\mathcal{A}$'s $(d^* + 1)$ to d-th set of random oracle queries are identical between Game $7 + d^* - 1$ and $7 + d^*$. (Here, $\mathcal{A}$ makes at least one KDM query between the d^*-th and $(d^* + 1)$-th set of random oracle queries due to the assumption. Thus, they are answered using an updated random oracle.) $\mathcal{A}_{d^*}$ can simulate them by using $O_{\mathsf{aux}} = H$ and information included in z. Therefore, $\mathcal{A}_{d^*}$ uses its oracle O only for answering to $\mathcal{A}$'s d^*-th set of random oracle queries, and thus $\mathcal{A}_{d^*}$'s query depth to O is 1.

We bound the right-hand side probability. In order to bound it, using $\mathcal{D}_{d^*}$, we construct the following adversary $\mathcal{B}_{d^*}$ against the SDM-OW-RSA security of dPKE.

$\mathcal{B}_{d^*}$**:** Given $\mathbf{pk} = (\mathsf{pk}^1, \ldots, \mathsf{pk}^\ell)$, $(\Delta^k)_k$, and $(s_{i,k}, \mathsf{ct}_{i,k})_{i \in [q_{\mathtt{kdm}}], k \in [\ell]}$, $\mathcal{B}_{d^*}$ first simulates Setup_{d^*}. $\mathcal{B}_{d^*}$ chooses a challenge bit $b \leftarrow \{0,1\}$ and prepares a fresh random oracle H. $\mathcal{B}_{d^*}$ then executes $\mathcal{A}^{|H\rangle, O_{\mathtt{KDM}}}(\mathbf{pk})$ just before $\mathcal{A}$ makes the d^*-th set of random oracle queries, where $O_{\mathtt{KDM}}$ is simulated as follows.

$O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Set $\mathsf{ct}_i \leftarrow \mathsf{ct}_{i,k_i}$ (and set $s_i \leftarrow s_{i,k_i}$).
2. Generate $u_i \leftarrow \{0,1\}^*$ and set $t_i \leftarrow u_i$.
3. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, t_i)$.

Let $\mathcal{A}$ makes i^* KDM queries before d^*-th set of random oracle queries. Then, $\mathcal{B}_{d^*}$ sets V_{i^*} as

$$V_{i^*}(x) = \begin{cases} u_j \oplus \widehat{f}^H_{j,b}(x) & (\text{if } \exists j \leq i^* \ : \ \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ H(x) & (\text{otherwise}). \end{cases}$$

$\mathcal{B}_{d^*}$ also sets

$$z = (\left|st\right\rangle, b, \mathbf{pk}, (\Delta^k)_{k \in [\ell]}, (k_i, f_{i,b}, s_i, \mathsf{ct}_i, u_i)_{i \in [i^*]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i \in [i^*+1, q_{\mathtt{kdm}}], k \in [\ell]}) \ ,$$

where $\left|st\right\rangle$ is the internal state of $\mathcal{A}$ at this point. Finally, $\mathcal{B}_{d^*}$ outputs $T \leftarrow \mathcal{D}_{d^*}^{|V_{i^*}, H, O_{\mathsf{aux}} = H\rangle}(z)$.

$\mathcal{B}_{d^*}$ perfectly simulates a successive execution of Setup_{d^*} and $\mathcal{D}_{d^*}$. Also, in the simulated execution, if $T \cap S_{i^*} \neq \emptyset$ occurs and ct_i has a unique pre-image $r + s_i$ under pk^{k_i} for every $i \in [q_{\mathtt{kdm}}]$, $\mathcal{B}_{d^*}$ wins. Thus, we have

$$Pr[T \cap S_{i^*} \neq \emptyset : T \leftarrow \mathcal{D}_{d^*}^{|V_{i^*}, H, O_{\mathsf{aux}} = H\rangle}(z), \mathsf{Setup}_{d^*}] \leq \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE}, \ell, q_{\mathtt{kdm}}, \mathcal{B}_{d^*}}(1^\lambda) + q_{\mathtt{kdm}} \cdot \delta_2.$$

From the discussions so far, by setting $\mathcal{B}$ as $\mathcal{B}_{d^*}$ such that $\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE}, \ell, q_{\mathtt{kdm}}, \mathcal{B}_{d^*}}(1^\lambda) \leq \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE}, \ell, q_{\mathtt{kdm}}, \mathcal{B}}(1^\lambda)$ for every $d^* \in [d]$, we see that there exists a QPT $\mathcal{B}$ that satisfies Equation (3). □ (**Theorem 4.2**)

5 SDM-OW-RSA Secure Deterministic PKE

In this section, we show that the SDM-OW-RSA security in the QROM of a tweaked version of T transformation [BHH+19] can be reduced to the IND-CPA security of the underlying PKE scheme. For the construction of the original T transformation, see Appendix A.

5.1 Construction

Construction 5.1. Let $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme whose message space is an abelian group $\mathcal{M}$ with the operation $+$. We also let the random coin space of KG and Enc be $\mathcal{R}_{\mathsf{kg}}$ and $\mathcal{R}_{\mathsf{enc}}$, respectively. Let $G = (G_{\mathsf{kg}}, G_{\mathsf{enc}})$ be a pair of hash functions, where $G_{\mathsf{kg}} : \mathcal{M} \to \mathcal{R}_{\mathsf{kg}}$ and $G_{\mathsf{enc}} : \mathcal{M} \to \mathcal{R}_{\mathsf{enc}}$. We construct T transformation with hash key generation $\mathsf{T}_{\mathtt{HKG}} = \mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G) = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ as follows.

$\mathsf{dKG}(1^\lambda; r)$**:** Return $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda; G_{\mathsf{kg}}(r))$.

$\mathsf{dEnc}(\mathsf{pk}, \mathsf{m})$**:** Return $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; G_{\mathsf{enc}}(\mathsf{m}))$.

$\mathsf{dDec}(\mathsf{sk}, \mathsf{CT})$**:** Return $\mathsf{m} \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$.

Recall that we define a deterministic PKE scheme is (δ_1, δ_2)-correct if it is δ_1-correct, and under a randomly generated key $(\mathsf{pk}, \mathsf{sk})$, the probability that a randomly generated message m has a collision, that is, another message m' such that $\mathsf{dEnc}(\mathsf{pk}, m) = \mathsf{dEnc}(\mathsf{pk}, m')$ is bounded by δ_2. Under this definition, as shown by [LW21, Lemma 4], $T(\mathsf{PKE}, G_{\mathsf{enc}})$ is $(\delta, 2\delta)$-correct if PKE is δ-correct for any δ. We can easily see that the correctness of $\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G)$ can be reduced to that of $\mathsf{T}(\mathsf{PKE}, G_{\mathsf{enc}})$, and thus $\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G)$ is $(\delta, 2\delta)$-correct if PKE is δ-correct for any δ.

5.2 Security Proof

We prove the following theorem.

Theorem 5.2. *Let $\ell = \ell(\lambda)$ and $q_{\mathtt{sdm}} = q_{\mathtt{sdm}}(\lambda)$ be polynomials and* PKE *be a PKE scheme. Let $\mathcal{A}$ be a QPT adversary against SDM-OW-RSA security of* $\mathsf{T}_{\mathtt{HKG}} = \mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G)$ *making total q (superposition) random oracle queries to $G_{\mathtt{kg}}$ and $G_{\mathtt{enc}}$ with query depth d, and outputs a list of size at most t as the final output. Then, there exists a QPT adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{T}_{\mathtt{HKG}},\ell,q_{\mathtt{sdm}},\mathcal{A}}(\lambda) \leq (d+2) \cdot \left(2 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{B}}(1^\lambda) + \frac{4(q+t)\ell(q_{\mathtt{sdm}}+1)}{|\mathcal{M}|}\right) + \frac{\ell q_{\mathtt{sdm}}(\ell q_{\mathtt{sdm}}-1)}{2|\mathcal{M}|} \ . \tag{5}$$

Proof. Without loss of generality, we assume that $\mathcal{A}$ makes random oracle queries to a single random oracle $G = G_{\mathtt{kg}} \times G_{\mathtt{enc}}$ instead of separate two random oracles $G_{\mathtt{kg}}$ and $G_{\mathtt{enc}}$ in the security games. Let $\widehat{\mathcal{A}}$ be a QPT adversary that runs in the same way as $\mathcal{A}$ except that before it terminates, $\widehat{\mathcal{A}}$ computes and discards $G(r')$ for all r' contained in $\mathcal{A}$'s final output T. Then, $\widehat{\mathcal{A}}$ makes at most $q+t$ queries to G with query depth $d+1$, and we have $\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{T}_{\mathtt{HKG}},\ell,q_{\mathtt{sdm}},\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{T}_{\mathtt{HKG}},\ell,q_{\mathtt{sdm}}\widehat{\mathcal{A}}}(\lambda)$. We estimate the latter using hybrid games. Let $\mathtt{SUC}_X$ be the event that the final output is 1 in Game X.

Game 1: This is $\mathsf{Exp}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{T}_{\mathtt{HKG}},\ell,q_{\mathtt{sdm}},\widehat{\mathcal{A}}}(1^\lambda)$.

> **Initialize:** The challenger generates $r \leftarrow \mathcal{M}$ and generates $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{KG}(1^\lambda; G_{\mathtt{kg}}(r + \Delta^k))$, where $\Delta^k \leftarrow \mathcal{M}$ for every $k \in [\ell]$. Then, for every $k \in [\ell]$ and $i \in [q_{\mathtt{sdm}}]$, the challenger generates $s_{i,k} \leftarrow \mathcal{M}$ and computes $\mathsf{ct}_{i,k} \leftarrow \mathsf{Enc}(\mathsf{pk}^k, r + s_{i,k}; G_{\mathtt{enc}}(r + s_{i,k}))$. The challenger executes $T \leftarrow \widehat{\mathcal{A}}^{|G\rangle}((\mathsf{pk}^k, \Delta^k)_{k\in[\ell]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i\in[q_{\mathtt{sdm}}],k\in[\ell]})$.
>
> **Finalize:** The challenger outputs 1 if and only if T contains r' such that $r' = r + s_{i,k}$ holds for some $i \in [q_{\mathtt{sdm}}]$ and $k \in [\ell]$.

Game 2: This game is the same as Game 1 except the followings. First, if there exists a pair $(s_{i,k}, s_{i',k'})$ such that $s_{i,k} = s_{i',k'}$, the challenger immediately outputs 0 as the final output of the game. Also, $G = G_{\mathtt{kg}} \times G_{\mathtt{enc}}$ is replaced with

$$V(x) = \begin{cases} \underline{u^k} & \underline{(\text{if } \exists k \in [\ell] \ : \ x = r + \Delta^k)} \\ \underline{v_{i,k}} & \underline{(\text{if } \exists i \in [q_{\mathtt{kdm}}] \text{ and } k \in [\ell] \ : \ x = r + s_{i,k})} \\ G(x) & (\text{otherwise}), \end{cases}$$

where $u^k, v_{i,k} \leftarrow \mathcal{R}_{\mathtt{kg}} \times \mathcal{R}_{\mathtt{enc}}$ for every $k \in [\ell]$ and $i \in [q_{\mathtt{kdm}}]$.

We have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| = \frac{\ell q_{\mathtt{sdm}}(\ell q_{\mathtt{sdm}}-1)}{2|\mathcal{M}|}$ since Game 1 and 2 are identical unless there exists a pair $(s_{i,k}, s_{i',k'})$ such that $s_{i,k} = s_{i',k'}$. Below, we let $S = \{r + \Delta^k\}_{k\in[\ell]} \cup \{r + s_{i,k}\}_{i\in[q_{\mathtt{sdm}}],k\in[\ell]}$.

Before proceeding the hybrid games, We provide the high level overview of the rest of games. In Game 2, the key generation randomness $G_{\mathtt{kg}}(r + \Delta^k)$ and encryption randomness $G_{\mathtt{enc}}(r + s_{i,k})$ correlate with the encrypted plaintexts $r + s_{i,k}$. Thus, next, at transition from Game 2 to 3, we eliminate the correlation by programming the random oracle. Concretely, in Game 3, the above randomnesses are generated by using V, but $\widehat{\mathcal{A}}$ gets access to only the punctured oracle $G \setminus S$, not V. In order to justify the programming, we use semi-classical O2H lemma (Lemma 3.2). By doing so, we can justify the programming without square root security loss, and obtain $\Pr[\mathtt{SUC}_2] \leq (d+2)\Pr[\mathtt{Find}_3]$, where $\mathtt{Find}_X$ be the event that the punctured oracle $G \setminus S$ returns 1 in Game X. Thus, all we have to do is to bound

$\Pr[\texttt{Find}_3]$. At Game 3, from the view of $\mathcal{A}$, the key generation randomness and encryption randomness are uniformly random strings that are independent of r, that is, u^k and $v_{i,k}$. Namely, the correlation issue above are solved. Thus, at transition from Game 3 to 4, we use the IND-m-CPA security of PKE, and eliminate information of r from $\mathsf{ct}_{i,k}$. In Game 4, except the punctured oracle $G \setminus S$, r is completely hidden from the view of $\widehat{\mathcal{A}}$. Therefore, by using Lemma 3.3, we can bound $\Pr[\texttt{Find}_4]$ and complete the proof.

Game 3: This game is the same as Game 2 except that $\widehat{\mathcal{A}}$ gets access to the punctured oracle $G \setminus S$. $(\mathsf{pk}^k, \mathsf{sk}^k)$ and $\mathsf{ct}_{i,k}$ are still generated using V for every $k \in [\ell]$ and $i \in [q_{\texttt{sdm}}]$.

Let $\texttt{Find}_X$ be the event that the punctured oracle $G \setminus S$ returns 1 in Game X. From the definition of $\widehat{\mathcal{A}}$, we have $\Pr\left[\texttt{SUC}_3 \wedge \neg\texttt{Find}_3\right] = 0$. Thus, we have

$$\sqrt{\Pr[\texttt{SUC}_2]} = \left|\sqrt{\Pr[\texttt{SUC}_2]} - \sqrt{\Pr\left[\texttt{SUC}_3 \wedge \neg\texttt{Find}_3\right]}\right| .$$

By applying Lemma 3.2, we obtain

$$\left|\sqrt{\Pr[\texttt{SUC}_2]} - \sqrt{\Pr\left[\texttt{SUC}_3 \wedge \neg\texttt{Find}_3\right]}\right| \leq \sqrt{(d+2) \cdot \Pr[\texttt{Find}_3]} .$$

Therefore, we also obtain $\Pr[\texttt{SUC}_2] \leq (d+2)\Pr[\texttt{Find}_3]$.

Game 4: This game is the same as Game 3 except that $\mathsf{ct}_{i,k}$ is generated as $\mathsf{ct}_{i,k} \leftarrow \mathsf{Enc}(\mathsf{pk}^k, 0)$ for every $k \in [\ell]$ and $i \in [q_{\texttt{sdm}}]$.

In order to estimate $|\Pr[\texttt{Find}_3] - \Pr[\texttt{Find}_4]|$, using $\widehat{\mathcal{A}}$, we construct the following QPT adversary $\mathcal{B}$ against the IND-m-CPA security of PKE. In the description, a function Test takes a value x and a set X as inputs and outputs 1 if $x \in X$ and 0 otherwise.

Initialize: Given $(\mathsf{pk}^k)_k$, $\mathcal{B}$ first generates $r \leftarrow \mathcal{M}$. $\mathcal{B}$ then generates $\Delta^k \leftarrow \mathcal{M}$ for every $k \in [\ell]$, $s_{i,k} \leftarrow \mathcal{M}$ for every $i \in [q_{\texttt{sdm}}]$ and $k \in [\ell]$, and a fresh random oracle G. If there exists a pair $(s_{i,k}, s_{i',k'})$ such that $s_{i,k} = s_{i',k'}$, $\mathcal{B}$ outputs 0 and terminates. Next, for every $i \in [q_{\texttt{sdm}}]$ and $k \in [\ell]$, $\mathcal{B}$ queries $(k, r + s_{i,k}, 0)$ to its oracle $O_{\texttt{IND}}$ and obtains $\mathsf{ct}_{i,k}$. Finally, $\mathcal{B}$ sets $b' = 0$ and executes $T \leftarrow \widehat{\mathcal{A}}^{|G \setminus S\rangle}((\mathsf{pk}^k, \Delta^k)_{k \in [\ell]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i \in [q_{\texttt{sdm}}], k \in [\ell]})$, where $G \setminus S$ is simulated as follows.

$G \setminus S$**:** When $\widehat{\mathcal{A}}$ makes a (superposition) query $|x\rangle|y\rangle$ to $G \setminus S$, $\mathcal{B}$ first computes $|x\rangle|y\rangle|\mathsf{Test}(x,S)\rangle$ and measures $|\mathsf{Test}(x,S)\rangle$. If the result is 0, $\mathcal{B}$ just returns $|x\rangle|y \oplus G(x)\rangle$ to $\widehat{\mathcal{A}}$. Otherwise, $\mathcal{B}$ set the value of b' to 1, and returns $|x\rangle|y \oplus G(x)\rangle$ to $\widehat{\mathcal{A}}$.

Finalize: If $\widehat{\mathcal{A}}$ terminates, $\mathcal{B}$ terminates with output b'.

Let the challenge bit in $\mathsf{Exp}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{B}}$ be b. $\mathcal{B}$ perfectly simulates Game 3 and 4 for $\mathcal{A}$ when $b = 0$ and $b = 1$, respectively. Also, $\mathcal{B}$ outputs $b' = 1$ if and only if $\texttt{Find}_3$ and $\texttt{Find}_4$ occur in the simulated Games. Thus, we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{B}}(1^\lambda) &= \left|\Pr[b = b'] - \frac{1}{2}\right| \\
&= \frac{1}{2}\left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right| \\
&= \frac{1}{2}\left|\Pr[\texttt{Find}_3] - \Pr[\texttt{Find}_4]\right| .
\end{aligned}$$

Finally, we bound $\Pr[\mathtt{Find}_4]$. In Game 4, conditioned on $(\mathsf{pk}^k, \Delta^k)_{k\in[\ell]}$ and $(s_{i,k}, \mathsf{ct}_{i,k})_{i\in[q_{\mathtt{sdm}}],k\in[\ell]}$, we have $\Pr_{r\leftarrow\mathcal{M}}[m \in S] \leq \frac{\ell(q_{\mathtt{sdm}}+1)}{|\mathcal{M}|}$ for any $m \in \mathcal{M}$. Thus, from Lemma 3.3, we obtain $\Pr[\mathtt{Find}_4] \leq \frac{4(q+t)\ell(q_{\mathtt{sdm}}+1)}{|\mathcal{M}|}$.

Overall, we see that there exists a QPT $\mathcal{B}$ that satisfies Equation (5). □ (**Theorem 5.2**)

6 Conclusion: KDM Security of FO Transformations

In the conclusion, we show that the KDM security in the QROM of FO transformations can be reduced to the IND-CPA security of the underlying PKE scheme without square root security loss.

We first provide the security bound for the KDM-CPA security of the PKE scheme $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G), H)$ in terms of the IND-m-CPA security of the underlying PKE. In order to capture the most general setting, we allow adversaries for the KDM-CPA security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G), H)$ and KDM functions queried by them to get access to not only H but also G. The access to G by an adversary does not affect the security proof provided in Section 4.2 since H and G are independent random oracles. Then, the following theorem holds.

Theorem 6.1. *Let $\ell = \ell(\lambda)$ be a polynomial and* PKE *be a δ-correct PKE scheme. Let $\mathcal{A}_{\mathtt{kdm}}$ be an adversary for the KDM-CPA security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G), H)$ making $q_{\mathtt{kdm}}$ KDM queries. Suppose $\mathcal{A}_{\mathtt{kdm}}$ makes at most q^G (resp. q^H) super-position random oracle queries to G (resp. H) with query depth d^G (resp. d^H). Also, suppose KDM functions queried by $\mathcal{A}_{\mathtt{kdm}}$ makes at most q^G_f (resp. q^H_f) classical random oracle queries to G (resp. H). Then, there exists a QPT adversary $\mathcal{A}_{\mathtt{ind}}$ such that*

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{kdm\text{-}cpa}}_{\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE},G),H),\ell,\mathcal{A}_{\mathtt{kdm}}}(1^\lambda) \\
&\leq 4d^H \cdot O(d^G + d^H \cdot q^G_f)\left(2\cdot\mathsf{Adv}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}_{\mathtt{ind}}}(1^\lambda) + \frac{O(q^G + q^H\cdot(\ell + q^G_f))\cdot\ell\cdot(q_{\mathtt{kdm}}+1)}{|\mathcal{M}|}\right) \\
&\qquad + \frac{2d^H\ell q_{\mathtt{kdm}}(\ell q_{\mathtt{kdm}}-1)}{|\mathcal{M}|} + \frac{4(q^H + q^H_f)q_{\mathtt{kdm}}}{\sqrt{|\mathcal{M}|}} + 2(4d^H+1)\cdot q_{\mathtt{kdm}}\cdot\delta\ . \qquad (6)
\end{aligned}$$

Proof. We estimate the number of queries to G made by $\mathcal{B}_{d^*}$ appeared in the proof of Theorem 4.2 when $\mathcal{A}_{\mathtt{kdm}}$ is used inside of it. First, $\mathcal{B}_{d^*}$ make $O(q^G)$ queries with depth $O(d^G)$ in order to simulate queries to G made by $\mathcal{D}_{d^*}$. Also, every time $\mathcal{D}_{d^*}$ makes a query to V_{i^*}, $\mathcal{B}_{d^*}$ needs to make at most $O(\ell + q^G_f)$ queries to G with depth $O(q^G_f)$ in order for the computation of $\widehat{f_{i,b}}$. Since $\mathcal{D}_{d^*}$ makes at most $O(q^H)$ queries to V_{i^*} with depth $O(d^H)$, to simulate $\mathcal{D}_{d^*}$'s queries to V_{i^*}, $\mathcal{B}_{d^*}$ needs to make at most $O(q^H \cdot (\ell + q^G_f))$ queries to G with query depth $O(d^H \cdot q^G_f)$. Therefore, $\mathcal{B}_{d^*}$ makes at most $O(q^G + q^H \cdot (\ell + q^G_f))$ queries to G with query depth $O(d^G + d^H \cdot q^G_f)$. This holds for every $d^* \in [d]$. Also, Since $\mathcal{D}_{d^*}$ outputs a list of size $O(q^H)$, so does $\mathcal{B}_{d^*}$ for every $d^* \in [d]$. From this fact and Theorems 4.2 and 5.2, we see that there exists a QPT $\mathcal{A}_{\mathtt{ind}}$ that satisfies Equation (6). □ (**Theorem 6.1**)

Remark 6.2 (On the value of q^G_f and q^H_f.). Note that the values of q^G_f and q^H_f are determined depending on usage scenarios and independent of the adversary's behavior. For example, in the usage scenario where we need only circular security such as anonymous credential [CL01], we can set $q^G_f = q^H_f = 0$. In that case, the multiplicative term of $\mathsf{Adv}^{\mathsf{ind\text{-}m\text{-}cpa}}_{\mathsf{PKE},\ell,\mathcal{A}_{\mathtt{ind}}}(1^\lambda)$ in Equation (6) is roughly the square of the query depth of $\mathcal{A}_{\mathtt{kdm}}$ to the random oracles. It is asymptotically the same as the multiplicative term appeared in the proof of IND-CCA secure KEM using O2H lemma with MRM [KSS+20]. In order to capture a wide range of applications, we allow KDM functions to get access to the random oracles in this work, but we think q^G_f and q^H_f are not large in many applications.

Let $\mathsf{FO}^{\perp}_{m,\mathtt{OTP}}(\mathsf{PKE}, G_{\mathtt{enc}}, H)$ be a PKE scheme constructed by combining the KEM $\mathsf{U}^{\perp}_{m}(\mathsf{T}(\mathsf{PKE}, G_{\mathtt{enc}}), H)$ with OTP as DEM. We provide the formal description of T and $\mathsf{U}^{\perp}_{m}$ in Appendix A. From Theorem 6.1, we can show that $\mathsf{FO}^{\perp}_{m}(\mathsf{PKE}, G_{\mathtt{enc}}, H)$ satisfies KDM-CPA security with asymptotically the same security loss with respect to the underlying IND-m-CPA secure PKE as Equation (6). Concretely, we have the following theorem.

Theorem 6.3. *Let $\ell = \ell(\lambda)$ be a polynomial and* PKE *be a PKE scheme. Let $\mathcal{A}_{\mathtt{kdm}}$ be an adversary for the KDM-ATK security of* $\mathsf{FO}^{\perp}_{m,\mathtt{OTP}}(\mathsf{PKE}, G_{\mathtt{enc}}, H)$ *where ATK $\in \{CPA, CCA\}$. Then, for* $\mathsf{atk} \in \{\mathsf{cpa}, \mathsf{cca}\}$, *there exists an adversary $\mathcal{A}'_{\mathtt{kdm}}$ such that*

$$\mathsf{Adv}^{\mathsf{kdm\text{-}atk}}_{\mathsf{FO}^{\perp}_{m,\mathtt{OTP}}(\mathsf{PKE},G_{\mathtt{enc}},H),\ell,\mathcal{A}_{\mathtt{kdm}}}(1^{\lambda}) \leq \mathsf{Adv}^{\mathsf{kdm\text{-}atk}}_{\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE},G),H),\ell,\mathcal{A}'_{\mathtt{kdm}}}(1^{\lambda}) + \frac{\ell(\ell-1)}{2|\mathcal{M}|} \ .$$

Proof. Suppose we modify the security game $\mathsf{Exp}^{\mathsf{kdm\text{-}atk}}_{\mathsf{FO}^{\perp}_{m,\mathtt{OTP}},\ell,\mathcal{A}_{\mathtt{kdm}}}(1^{\lambda})$ so that the k-th key pair $(\mathsf{pk}^{k}, \mathsf{sk}^{k})$ is generated by using $G_{\mathtt{kg}}(r^{k})$ as the random coin for KG for every $k \in [\ell]$, where $G_{\mathtt{kg}} : \mathcal{M} \to \mathcal{R}_{\mathtt{kg}}$ is a random oracle and $r^{k} \leftarrow \mathcal{M}$ for every $k \in [\ell]$. If $r^{1}, \ldots, r^{\ell}$ are mutually different, then the distribution of ℓ key pairs does not change from the view of $\mathcal{A}_{\mathtt{kdm}}$ by this modification. We emphasize that $\mathcal{A}_{\mathtt{kdm}}$ does not have access to $G_{\mathtt{kg}}$. By the modification, $\mathcal{A}_{\mathtt{kdm}}$'s advantage is changed at most $\frac{\ell(\ell-1)}{2|\mathcal{M}|}$. We can see that we can easily construct an adversary $\mathcal{A}'_{\mathtt{kdm}}$ such that $\mathsf{Adv}^{\mathsf{kdm\text{-}atk}}_{\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE},G),H),\ell,\mathcal{A}'_{\mathtt{kdm}}}(1^{\lambda})$ is exactly the same as $\mathcal{A}_{\mathtt{kdm}}$' advantage in the modified game. Therefore, we obtain the theorem. □ (**Theorem 6.3**)

Thus, we see that the KDM-CPA security of $\mathsf{FO}^{\perp}_{m,\mathtt{OTP}}(\mathsf{PKE}, G_{\mathtt{enc}}, H)$ is reduced to that of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}(\mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G), H)$ with additional security loss $\frac{\ell(\ell-1)}{2|\mathcal{M}|}$ which is absorbed by the additive term of Equation (6).

Extension to KDM-CCA security. In the main body of this paper, we focused on KDM-CPA security. Our proof technique is also compatible with KDM-CCA security. Concretely, we can prove the KDM-CCA security of a PKE scheme constructed by using a variant of $\mathsf{U}^{\perp}_{m}$ called $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m} = \mathsf{U}^{\perp,\mathtt{keyconf}}_{m}(\mathsf{dPKE}, H)$ as KEM and OTP-then-MAC as DEM without square root security loss if the underlying dPKE is SDM-OW-RSA secure and additionally satisfies injectiveness. The security proof is a combination of our proof for the KDM-CPA security of $\mathsf{U}^{\perp}_{m,\mathtt{OTP}}$ and the proof for the IND-CCA security of $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m}$ by [BHH+19, KSS+20]. We provide the formal description of this construction and security proof for the KDM-CCA security of it in Appendix B.

By following a similar argument as the case of KDM-CPA security, we can show that the KDM-CCA security of the KEM $\mathsf{FO}^{\perp,\mathtt{keyconf}}_{m}(\mathsf{PKE}, G_{\mathtt{enc}}, H) = \mathsf{U}^{\perp,\mathtt{keyconf}}_{m}(T(\mathsf{PKE}, G_{\mathtt{enc}}), H)$ combined with OTP-then-MAC as DEM, can be reduced to the IND-CPA security of PKE. The multiplicative term in the security bound with respect to the underlying PKE is roughly the same as Equation (6) though some additive terms are added to the security bound.

Acknowledgments

The authors thank Takashi Yamakawa for helpful comments.

References

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 269–295. Springer, Heidelberg, August 2019. (Cited on page 1, 4, 6, 8, 11, 12.)

[AR02] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 15(2):103–127, March 2002. (Cited on page 1.)

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000. (Cited on page 3.)

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011. (Cited on page 1, 2, 4.)

[BDU08] Michael Backes, Markus Dürmuth, and Dominique Unruh. OAEP is secure under key-dependent messages. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 506–523. Springer, Heidelberg, December 2008. (Cited on page 3.)

[BHH$^+$19] Nina Bindel, Mike Hamburg, Kathrin Hövelmanns, Andreas Hülsing, and Edoardo Persichetti. Tighter proofs of CCA security in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 61–90. Springer, Heidelberg, December 2019. (Cited on page 1, 2, 3, 4, 5, 6, 8, 10, 11, 12, 15, 20, 24, 27, 29, 30.)

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 108–125. Springer, Heidelberg, August 2008. (Cited on page 1.)

[Ble98] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 1–12. Springer, Heidelberg, August 1998. (Cited on page 1.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993. (Cited on page 1, 4.)

[BR95] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111. Springer, Heidelberg, May 1995. (Cited on page 3.)

[BRS03] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *SAC 2002*, volume 2595 of *LNCS*, pages 62–75. Springer, Heidelberg, August 2003. (Cited on page 1.)

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001. (Cited on page 1, 23.)

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. (Cited on page 1.)

[DS14] Gareth T. Davies and Martijn Stam. KDM security in the hybrid framework. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 461–480. Springer, Heidelberg, February 2014. (Cited on page 4.)

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013. (Cited on page 1, 2, 4.)

[Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`. (Cited on page 1.)

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Heidelberg, November 2017. (Cited on page 1, 2, 3, 4, 27.)

[HKSU20] Kathrin Hövelmanns, Eike Kiltz, Sven Schäge, and Dominique Unruh. Generic authenticated key exchange in the quantum random oracle model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 389–422. Springer, Heidelberg, May 2020. (Cited on page 1, 4.)

[JZC$^+$18] Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 96–125. Springer, Heidelberg, August 2018. (Cited on page 1, 4.)

[JZM19a] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Key encapsulation mechanism with explicit rejection in the quantum random oracle model. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 618–645. Springer, Heidelberg, April 2019. (Cited on page 1, 4.)

[JZM19b] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 227–248. Springer, Heidelberg, 2019. (Cited on page 4.)

[KMHT16] Fuyuki Kitagawa, Takahiro Matsuda, Goichiro Hanaoka, and Keisuke Tanaka. On the key dependent message security of the Fujisaki-Okamoto constructions. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 99–129. Springer, Heidelberg, March 2016. (Cited on page 1, 4.)

[KSS$^+$20] Veronika Kuchta, Amin Sakzad, Damien Stehlé, Ron Steinfeld, and Shifeng Sun. Measure-rewind-measure: Tighter quantum random oracle model proofs for one-way to hiding and CCA security. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 703–728. Springer, Heidelberg, May 2020. (Cited on page 1, 2, 3, 4, 5, 6, 8, 10, 11, 12, 23, 24, 29.)

[LW21] Xu Liu and Mingqiang Wang. QCCA-secure generic key encapsulation mechanism with tighter security in the quantum random oracle model. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 3–26. Springer, Heidelberg, May 2021. (Cited on page 7, 12, 20.)

[RS92] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 433–444. Springer, Heidelberg, August 1992. (Cited on page 1.)

[SXY18] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 520–551. Springer, Heidelberg, April / May 2018. (Cited on page 1, 2, 4.)

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 192–216. Springer, Heidelberg, October / November 2016. (Cited on page 1, 4.)

[Unr14] Dominique Unruh. Quantum position verification in the random oracle model. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 1–18. Springer, Heidelberg, August 2014. (Cited on page 8, 13.)

[Unr15] Dominique Unruh. Revocable quantum timed-release encryption. *J. ACM*, 62(6):49:1–49:76, 2015. (Cited on page 1, 6.)

[Zha12] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, Heidelberg, August 2012. (Cited on page 11.)

[Zha19] Jiayu Zhang. Delegating quantum computation in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 30–60. Springer, Heidelberg, December 2019. (Cited on page 1, 3, 9, 37.)

A T, $\mathsf{U}_m^{\perp}$, and $\mathsf{U}_m^{\perp,\mathtt{keyconf}}$ Transformations

We Recall the two transformations T and U that together yield FO. T and U are first introduced by Hofheinz et al. [HHK17]. In this work, we adopt variants of T and U defined by Bindel et al. [BHH$^+$19]. The only difference between the transformations by Hofheinz et al. and those by Bindel et al. is that the validity check by encryption in the decryption algorithm is performed as a part of T in the former while it is performed as a part of U in the latter. Thus, the resulting FO is the same regardless of which definitions of T and U we use.

Construction A.1 (T transformation). Let $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme whose message space is $\mathcal{M}$. We also let the random coin space of Enc be $\mathcal{R}_{\mathsf{enc}}$. Let $G_{\mathsf{enc}} : \mathcal{M} \to \mathcal{R}_{\mathsf{enc}}$ be a hash function. T transformation $\mathsf{T} = \mathsf{T}(\mathsf{PKE}, G_{\mathsf{enc}}) = (\mathsf{KG}_{\mathsf{T}}, \mathsf{Enc}_{\mathsf{T}}, \mathsf{Dec}_{\mathsf{T}})$ is described as follows.

$\mathsf{KG}_{\mathsf{T}}(1^{\lambda})$: Return $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^{\lambda})$.

$\mathsf{Enc}_{\mathsf{T}}(\mathsf{pk}, \mathsf{m})$: Return $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; G_{\mathsf{enc}}(\mathsf{m}))$.

$\mathsf{Dec}_{\mathsf{T}}(\mathsf{sk}, \mathsf{CT})$: Return $\mathsf{m} \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$.

Construction A.2 ($\mathsf{U}_m^{\perp}$ transformation). Let $\mathsf{dPKE} = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ be a deterministic PKE scheme whose message space is $\mathcal{M}$. Let $H : \mathcal{M} \to \{0,1\}^*$ be a hash function. A KEM scheme $\mathsf{U}_m^{\perp} = \mathsf{U}_m^{\perp}(\mathsf{dPKE}, H) = (\mathsf{KG}_{\mathsf{U}_m^{\perp}}, \mathsf{Enc}_{\mathsf{U}_m^{\perp}}, \mathsf{Dec}_{\mathsf{U}_m^{\perp}})$ is described as follows.

$\mathsf{KG}_{\mathsf{U}_m^{\perp}}(1^{\lambda})$: Return $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^{\lambda})$.

$\mathsf{Enc}_{\mathsf{U}_m^{\perp}}(\mathsf{pk})$: Generate $s \leftarrow \mathcal{M}$ and compute $\mathsf{ct} \leftarrow \mathsf{dEnc}(\mathsf{pk}, s)$. Return ct as a ciphertext and $H(s)$ as a session key.

$\mathsf{Dec}_{\mathsf{U}_m^{\perp}}(\mathsf{sk}, \mathsf{ct}')$: Compute $s' \leftarrow \mathsf{dDec}(\mathsf{sk}, \mathsf{ct}')$ and return $\perp$ if $s' = \perp$ or $\mathsf{ct} \neq \mathsf{dEnc}(\mathsf{pk}, s')$. Otherwise, return $H(s')$.

Construction A.3 ($\mathsf{U}_m^{\perp,\mathtt{keyconf}}$ transformation). Let $\mathsf{dPKE} = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ be a deterministic PKE scheme whose message space is $\mathcal{M}$. Let $H : \mathcal{M} \to \{0,1\}^*$ be a hash function. A KEM scheme $\mathsf{U}_m^{\perp,\mathtt{keyconf}} = \mathsf{U}_m^{\perp,\mathtt{keyconf}}(\mathsf{dPKE}, H) = (\mathsf{KG}_{\mathsf{U}_m^{\perp,\mathtt{keyconf}}}, \mathsf{Enc}_{\mathsf{U}_m^{\perp,\mathtt{keyconf}}}, \mathsf{Dec}_{\mathsf{U}_m^{\perp,\mathtt{keyconf}}})$ is described as follows.

$\mathsf{KG}_{\mathsf{U}_m^{\perp,\mathtt{keyconf}}}(1^\lambda)$: Return $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)$.

$\mathsf{Enc}_{\mathsf{U}_m^{\perp,\mathtt{keyconf}}}(\mathsf{pk})$: Generate $s \leftarrow \mathcal{M}$ and compute $\mathsf{ct} \leftarrow \mathsf{dEnc}(\mathsf{pk}, s)$ and $\mathsf{seskey} \| \mathsf{kc} \leftarrow H(s)$. Return $(\mathsf{ct}, \mathsf{kc})$ as a ciphertext and seskey as a session key.

$\mathsf{Dec}_{\mathsf{U}_m^{\perp,\mathtt{keyconf}}}(\mathsf{sk}, (\mathsf{ct}', \mathsf{kc}'))$: Compute $s' \leftarrow \mathsf{dDec}(\mathsf{sk}, \mathsf{ct}')$ and return $\perp$ if $s' = \perp$ or $\mathsf{ct}' \neq \mathsf{dEnc}(\mathsf{pk}, s')$. Otherwise, compute $\mathsf{seskey}' \| \mathsf{kc}'' \leftarrow H(s')$ and returns seskey' if $\mathsf{kc}' = \mathsf{kc}''$ and $\perp$ otherwise.

B KDM-CCA Security of $\mathsf{U}_m^{\perp,\mathtt{keyconf}}$ with OTP-then-MAC as DEM

In this section, we show that a PKE scheme that we denote $\mathsf{U}_{m,\mathtt{OTP+MAC}}^{\perp,\mathtt{keyconf}} = \mathsf{U}_{m,\mathtt{OTP+MAC}}^{\perp,\mathtt{keyconf}}(\mathsf{dPKE}, H)$ satisfies KDM-CCA security in the QROM without square root security loss if the underlying dPKE satisfies SDM-OW-RSA security and injectiveness.

B.1 Definitions

Definition B.1 (KDM-CCA security for PKE). *Let* $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme. We define* $\mathsf{Exp}_{\mathsf{PKE},\ell,\mathcal{A}}^{\mathsf{kdm\text{-}cca}}(1^\lambda)$ *for an adversary* $\mathcal{A}$ *as follows.*

Initialize: *First, the challenger chooses a challenge bit* $b \leftarrow \{0,1\}$. *Next, the challenger generates* $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{KG}(1^\lambda)$ *for every* $k \in [\ell]$. *The challenger sets* $\mathbf{sk} := (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$, *and executes* $b' \leftarrow \mathcal{A}^{O_{\mathtt{KDM}}, O_{\mathtt{Dec}}}((\mathsf{pk}^k)_{k \in [\ell]})$.

$O_{\mathtt{KDM}}$: *On the i-th call with input* $(k_i, f_{i,0}, f_{i,1})$, *where* $k_i \in [\ell]$ *and* $f_{i,0}$ *and* $f_{i,1}$ *are efficiently computable functions with the same output length, it returns* $\mathsf{CT}_i \leftarrow \mathsf{Enc}(\mathsf{pk}^{k_i}, f_{i,b}(\mathbf{sk}))$.

$O_{\mathtt{Dec}}$: *On input* (k', CT'), *it returns* $\perp$ *if* $(k', \mathsf{CT}') = (k_j, \mathsf{CT}_j)$ *for* $j \leq i$, *where i is the number of KDM queries already made at this point. Otherwise, it returns* $\mathsf{Dec}(\mathsf{sk}^{k'}, \mathsf{CT}')$.

Finalize: *The challenger outputs* 1 *if* $b = b'$ *and* 0 *otherwise.*

We say that PKE *is KDM-CCA secure if for any polynomial* $\ell = \ell(\lambda)$ *and QPT adversary* $\mathcal{A}$, *we have* $\mathsf{Adv}_{\mathsf{PKE},\ell,\mathcal{A}}^{\mathsf{kdm\text{-}cca}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Exp}_{\mathsf{PKE},\ell,\mathcal{A}}^{\mathsf{kdm\text{-}cca}}(1^\lambda)\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$.

Definition B.2 (Injectivity of deterministic PKE). *We say that a deterministic PKE scheme* $\mathsf{dPKE} = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ *is* η*-injective if*

$$\Pr\left[\mathsf{dEnc}(\mathsf{pk}, \cdot) \text{ is not injective } | (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)\right] \leq \eta \ .$$

If dPKE *is constructed in the random oracle model, the probability is taken over the choice of* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)$ *and the random oracle.*

Definition B.3 (Finding failing ciphertext). *Let* $\mathsf{dPKE} = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ *be a deterministic PKE scheme. We define* $\mathsf{Expt}^{\mathsf{ffc}}_{\mathsf{dPKE},\mathcal{A}}(1^\lambda)$ *for an adversary* $\mathcal{A}$ *as follows.*

Initialize: *First, the challenger generates* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)$. *The challenger executes* $L_{\mathtt{ffc}} \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{sk})$.

Finalize: *The challenger outputs* 1 *if there exists* $(m, \mathsf{CT}) \in \mathcal{M} \times L_{\mathtt{ffc}}$ *such that* $\mathsf{CT} = \mathsf{dEnc}(\mathsf{pk}, m)$ *and* $\mathsf{dDec}(\mathsf{sk}, \mathsf{CT}) \neq m$. *Otherwise, the challenger outputs* 0.

We define $\mathsf{Adv}^{\mathsf{ffc}}_{\mathsf{dPKE},\mathcal{A}}(\lambda) = \Pr\left[1 \leftarrow \mathsf{Expt}^{\mathsf{ffc}}_{\mathsf{dPKE},\mathcal{A}}(1^\lambda)\right]$.

The above definition is slightly different from the original definition used in [BHH+19, KSS+20]. In the original definition, $\mathcal{A}$ is given only the public key pk, but in Definition B.3, $(\mathsf{pk}, \mathsf{sk})$ is given to $\mathcal{A}$. As shown below, we bound $\mathsf{Adv}^{\mathsf{ffc}}_{\mathsf{PKE},\mathcal{A}}(\lambda)$ statistically, and thus the difference is not a big issue.

Lemma B.4. *Let* $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a* δ*-correct PKE scheme whose message space is an abelian group* $\mathcal{M}$ *with the operation* $+$. *We also let the random coin space of* KG *and* Enc *be* $\mathcal{R}_{\mathtt{kg}}$ *and* $\mathcal{R}_{\mathtt{enc}}$, *respectively. Let* $G = (G_{\mathtt{kg}}, G_{\mathtt{enc}})$ *be a pair of hash functions, where* $G_{\mathtt{kg}} : \mathcal{M} \to \mathcal{R}_{\mathtt{kg}}$ *and* $G_{\mathtt{enc}} : \mathcal{M} \to \mathcal{R}_{\mathtt{enc}}$. *Suppose* $\mathsf{T}_{\mathtt{HKG}} = \mathsf{T}_{\mathtt{HKG}}(\mathsf{PKE}, G)$ *constructed as Construction 5.1 is* η*-injective. Let* $\mathcal{A}$ *be an adversary that runs in* $\mathsf{Expt}^{\mathsf{ffc}}_{\mathsf{T}_{\mathtt{HKG}},\mathcal{A}}(1^\lambda)$ *and makes at most* q *queries to* G *with query depth* d *and returns* $L_{\mathtt{ffc}}$ *of size at most* $q_{\mathtt{dec}}$. *Then, we have* $\mathsf{Adv}^{\mathsf{ffc}}_{\mathsf{T}_{\mathtt{HKG}},\mathcal{A}}(\lambda) \leq ((4d+1) \cdot \delta + \sqrt{3\eta}) \cdot (q + q_{\mathtt{dec}}) + \eta$.

This lemma can be proven in almost the same way as Lemma 6 in [BHH+19] that guarantees the same bound for T. We omit the formal proof.

Definition B.5 (Strong OT-MAC). *A strong OT-MAC* MAC *is a three tuple* $(\mathsf{MGen}, \mathsf{Tag}, \mathsf{Vrfy})$ *of PPT algorithms. Below, let* D_{mac} *be the domain of* MAC.

- $\mathsf{MGen}(1^\lambda)$: *Given a security parameter* 1^λ, *outputs a key* mk.
- $\mathsf{Tag}(\mathsf{mk}, m)$: *Given a key* mk *and a message* $m \in D_{\mathsf{mac}}$, *outputs* mac.
- $\mathsf{Vrfy}(\mathsf{mk}, m, \mathsf{mac})$: *Given a key* mk, *message* $m \in D_{\mathsf{mac}}$, *and* mac, *outputs* $\top$ *or* $\bot$.

We require the following properties.

Correctness: *For every* $m \in D_{\mathsf{mac}}$ *and* $\mathsf{mk} \leftarrow \mathsf{MGen}(1^\lambda)$, *we have* $\mathsf{Vrfy}(\mathsf{mk}, m, \mathsf{Tag}(\mathsf{mk}, m)) = \top$.

Security: *For any QPT adversary* $\mathcal{A}$, *it holds that*

$$\mathsf{Adv}^{\mathsf{sot\text{-}mac}}_{\mathsf{MAC},\mathcal{A}}(1^\lambda) = \Pr\left[\begin{array}{c|c} \mathsf{Vrfy}(\mathsf{mk}, m, \mathsf{mac}) = \top \wedge & \mathsf{mk} \leftarrow \mathsf{MGen}(1^\lambda) \\ (m, \mathsf{mac}) \neq (m_1, \mathsf{mac}_1) & (m, \mathsf{mac}) \leftarrow \mathcal{A}(1^\lambda)^{\mathsf{Tag}(\mathsf{mk},\cdot)} \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $\mathcal{A}$ *gets access to the oracle only once and* m_1 *is the query from* $\mathcal{A}$ *and* mac_1 *is the response.*

We have the following theorem.

Theorem B.6. *There exists an information-theoretically secure strong OT-MAC.*

B.2 Construction

Construction B.7. Let $\mathsf{MAC} = (\mathsf{MGen}, \mathsf{Tag}, \mathsf{Vrfy})$ be a strong OT-MAC. Let $\mathsf{dPKE} = (\mathsf{dKG}, \mathsf{dEnc}, \mathsf{dDec})$ be a deterministic PKE scheme whose message space is $\mathcal{M}$. We assume that $\mathcal{M}$ is an abelian group and denote the operation in $\mathcal{M}$ as $+$. Let H be a hash function. We construct $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}} = \mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}}(\mathsf{dPKE}, H) = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ as follows.

$\mathsf{KG}(1^\lambda)$**:** Return $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{dKG}(1^\lambda)$.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{m})$**:** Generate $s \leftarrow \mathcal{M}$ and compute $\mathsf{ct} \leftarrow \mathsf{dEnc}(\mathsf{pk}, s)$ and $\mathsf{otp}\|\mathsf{mk}\|\mathsf{kc} \leftarrow H(\mathsf{pk}\|s)$. Compute $t = \mathsf{otp} \oplus \mathsf{m}$ and $\mathsf{mac} \leftarrow \mathsf{Tag}(\mathsf{mk}, t)$. Return $\mathsf{CT} = (\mathsf{ct}, \mathsf{kc}, t, \mathsf{mac})$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}')$**:** Parse $\mathsf{CT}' = (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}')$, compute $s' \leftarrow \mathsf{dDec}(\mathsf{sk}, \mathsf{ct}')$ and return $\perp$ if $s' = \perp$ or $\mathsf{ct}' \neq \mathsf{dEnc}(\mathsf{pk}, s')$. Otherwise, compute $\mathsf{otp}'\|\mathsf{mk}'\|\mathsf{kc}'' \leftarrow H(\mathsf{pk}\|s')$ and return $\perp$ if $\mathsf{kc}' \neq \mathsf{kc}''$. Otherwise, return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', t', \mathsf{mac}')$ and $\perp$ otherwise.

We see that if dPKE is (δ_1, δ_2)-correct, then $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}}$ is δ_1-correct for any δ_1.

Remark B.8 (On hashing pk with s). In the above construction, pk is fed into H together with s. As stated by [BHH+19], we usually need to do this to prove a security notion defined in the security game where there are multiple public and secret key pairs, especially in the case of CCA security.

B.3 Security Proof

We prove the following theorem.

Theorem B.9. *Let $\ell = \ell(\lambda)$ be any polynomial, dPKE be a (δ_1, δ_2)-correct and η-injective deterministic PKE scheme, and MAC be a strong OT-MAC. Let $\mathcal{A}$ be a QPT adversary against the KDM-CCA security of $\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}} = \mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}}(\mathsf{dPKE}, H)$ making q (superposition) random oracle queries with query depth d to H, $q_{\mathtt{kdm}}$ (classical) queries to $O_{\mathtt{KDM}}$, and $q_{\mathtt{dec}}$ (classical) queries to $O_{\mathtt{Dec}}$. Also, let q_f be the upper bound of the total number of (classical) random oracle queries made by KDM functions. Then, there exists QPT adversaries $\mathcal{B}$, $\mathcal{B}_{\mathtt{ffc}}$, and $\mathcal{B}_{\mathtt{mac}}$ such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{kdm\text{-}cca}}_{\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}},\ell,\mathcal{A}}(1^\lambda) \leq{}& 8d \cdot \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE},\ell,q_{\mathtt{kdm}},\mathcal{B}}(1^\lambda) + \ell \cdot \mathsf{Adv}^{\mathsf{ffc}}_{\mathsf{dPKE},\mathcal{B}_{\mathtt{ffc}}}(1^\lambda) + q_{\mathtt{kdm}} \cdot \mathsf{Adv}^{\mathsf{sot\text{-}mac}}_{\mathsf{MAC},\mathcal{B}_{\mathtt{mac}}}(1^\lambda) \\ &+ \frac{4(q+q_f)q_{\mathtt{kdm}}}{\sqrt{|\mathcal{M}|}} + \frac{q_{\mathtt{dec}}}{2^{|\mathsf{kc}|}} + \ell \cdot \eta + q_{\mathtt{kdm}} \cdot \delta_1 + 2(4d+1) \cdot q_{\mathtt{kdm}} \cdot \delta_2 \ . \qquad (7)\end{aligned}$$

Proof. We complete the proof using hybrid games. Let $\mathtt{SUC}_X$ be the event that the final output is 1 in Game X. We assume that $\mathcal{A}$ makes at least one KDM query before the first set of random oracle queries and between d^*-th set of random oracle queries and (d^*+1)-th set of random oracle queries for every $d^* \in [d-1]$. This assumption is without loss of generality in the sense that any adversary can be transformed into one satisfying this condition without changing the number and depth of random oracle queries.

Game 1: This is $\mathsf{Exp}^{\mathsf{kdm\text{-}cca}}_{\mathsf{U}^{\perp,\mathtt{keyconf}}_{m,\mathtt{OTP+MAC}},\ell,\mathcal{A}}(1^\lambda)$.

Initialize: First, the challenger chooses a challenge bit $b \leftarrow \{0,1\}$. The challenger also generates a fresh random oracle H. Next, the challenger generates $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda)$ for every $k \in [\ell]$. The challenger sets $\mathbf{sk} := (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$ and $\mathbf{pk} := (\mathsf{pk}^1, \ldots, \mathsf{pk}^\ell)$, and executes $b' \leftarrow \mathcal{A}^{|H\rangle, O_{\mathtt{KDM}}, O_{\mathtt{Dec}}}(\mathbf{pk})$. $O_{\mathtt{KDM}}$ and $O_{\mathtt{Dec}}$ behave as follows.

$O_{\mathtt{KDM}}$**:** On the i-th call with input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, s_i)$.
2. Compute $\mathsf{otp}_i \| \mathsf{mk}_i \| \mathsf{kc}_i \leftarrow H(\mathsf{pk}^{k_i} \| s_i)$.
3. Compute $t_i = \mathsf{otp}_i \oplus f_{i,b}^{H}(\mathbf{sk})$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

$O_{\mathtt{Dec}}$: On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\perp$ if $(k', \mathsf{CT}') = (k_j, \mathsf{CT}_j)$ for $j \leq i$, where i is the number of KDM queries already made at this point. Otherwise, it responds as follows.

1. Compute $s' \leftarrow \mathsf{dDec}(\mathsf{sk}^{k'}, \mathsf{ct}')$.
2. Return $\perp$ if $\perp = s'$ or $\mathsf{ct}' \neq \mathsf{dEnc}(\mathsf{pk}^{k'}, s')$. Otherwise, compute $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow H(\mathsf{pk}^{k'} \| s')$. Return $\perp$ if $\mathsf{kc}' \neq \mathsf{kc}''$.
3. Return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', t', \mathsf{mac}')$ and $\perp$ otherwise.

Finalize: The challenger outputs 1 if $b = b'$ and 0 otherwise.

Game 2: This is the same as Game 1 except the behavior of $O_{\mathtt{KDM}}$. In this game, $O_{\mathtt{KDM}}$ adaptively reprograms the random oracle that $\mathcal{A}$ (and functions queried by $\mathcal{A}$) gets access every time it is invoked. The detailed description is as follows.

$O_{\mathtt{KDM}}$: On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, s_i)$.
2. Generate $u_i \leftarrow \{0,1\}^*$ and parse it as $\mathsf{otp}_i \| \mathsf{mk}_i \| \mathsf{kc}_i$.
3. Compute $t_i = \mathsf{otp}_i \oplus f_{i,b}^{V_{i-1}}(\mathbf{sk})$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

Also, it updates the random oracle into

$$V_i(\mathsf{pk} \| x) = \begin{cases} u_j & (\text{if } \exists j \leq i \ : \ \mathsf{pk} \| x = \mathsf{pk}^{k_j} \| s_j) \\ H(\mathsf{pk} \| x) & (\text{otherwise}), \end{cases}$$

We can show that $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| = \frac{4(q+q_f)q_{\mathtt{kdm}}}{\sqrt{\mathcal{M}}}$ by using a modified version of Lemma 3.7. The reason why we cannot use Lemma 3.7 directly is that the programmed point here is $\mathsf{pk}^{k_i} \| s_i$, but Lemma 3.7 requires a programmed point be chosen uniformly at random. However, even if we allow a programmed point to be the form of $z \| x$, where z is an adversarially chosen value and x is a uniformly at random value, we can have the same bound as Lemma 3.7. We omit the formal proof.

Game 3: This is the same as Game 2 except $O_{\mathtt{Dec}}$ behaves as follows, where R is a random oracle.

$O_{\mathtt{Dec}}$: On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\perp$ if $(k', \mathsf{CT}') = (k_j, \mathsf{CT}_j)$ for $j \leq i$, where i is the number of KDM queries already made at this point. Otherwise, it responds as follows.

1. Compute $s' \leftarrow \mathsf{dDec}(\mathsf{sk}^{k'}, \mathsf{ct}')$.
2. Compute $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow R(\mathsf{pk}^{k'} \| \mathsf{ct}')$ if $s' = \perp$ or $\mathsf{ct}' \neq \mathsf{dEnc}(\mathsf{pk}^{k'}, s')$. Otherwise, computes $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow H(\mathsf{pk}^{k'} \| s')$. Return $\perp$ if $\mathsf{kc}' \neq \mathsf{kc}''$.
3. Return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', \mathsf{mac}')$ and $\perp$ otherwise.

R is used only inside $O_{\mathtt{Dec}}$. Then, for the first decryption query $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$ such that $s' = \perp$ or $\mathsf{ct}' \neq \mathsf{dEnc}(\mathsf{pk}^{k'}, s')$, the probability that $\mathsf{kc}' = \mathsf{kc}''$ holds is $\frac{1}{2^{|\mathsf{kc}|}}$, where $s' \leftarrow \mathsf{dDec}(\mathsf{sk}^{k'}, \mathsf{ct}')$ and $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow R(\mathsf{pk}^{k'} \| \mathsf{ct}')$. Thus, $O_{\mathtt{Dec}}$ returns $\perp$ for the decryption query without the probability $\frac{1}{2^{|\mathsf{kc}|}}$. By repeating this argument, we obtain $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq \frac{q_{\mathsf{dec}}}{2^{|\mathsf{kc}|}}$.

Game 4: This game is the same as Game 3 except that H is replaced with

$$V_0(\mathsf{pk}\|x) = \begin{cases} \underline{R(\mathsf{pk}\|\mathsf{dEnc}(\mathsf{pk}, x))} & \underline{(\text{if } \exists k \in [\ell] \ : \ \mathsf{pk} = \mathsf{pk}^k)} \\ H(\mathsf{pk}\|x) & (\text{otherwise}), \end{cases} \tag{8}$$

where R is the random oracle introduced in the previous game.

We define an event $\mathtt{FFC}_X$ as follows.

$\mathtt{FFC}_X$**:** In Game X, $\mathcal{A}$ makes a decryption query $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$ satisfying the condition that there exists $m' \in \mathcal{M}$ such that $\mathsf{ct}' = \mathsf{dEnc}(\mathsf{pk}^{k'}, m')$ but $m' \neq \mathsf{dDec}(\mathsf{sk}^{k'}, \mathsf{ct}')$.

If $\mathtt{FFC}_3$ and $\mathtt{FFC}_4$ does not occur, R is used inside $O_{\mathtt{Dec}}$ only for $\mathsf{pk}^k\|\mathsf{ct}'$ such that there does not exists $m' \in \mathcal{M}$ satisfying $\mathsf{ct}' = \mathsf{dEnc}(\mathsf{pk}^k, m')$ for every $k \in [\ell]$. On the other hand, inside V_0, R is used for $\mathsf{pk}^k\|\mathsf{ct}$ such that there exists $m \in \mathcal{M}$ satisfying $\mathsf{ct} = \mathsf{dEnc}(\mathsf{pk}^k, m)$. Moreover, if $\mathsf{dEnc}(\mathsf{pk}^k, \cdot)$ is injective for every $k \in [\ell]$, $R(\mathsf{pk}^k\|\mathsf{dEnc}(\mathsf{pk}^k, x))$ and $R(\mathsf{pk}^k\|\mathsf{dEnc}(\mathsf{pk}^k, x'))$ are independent random values for any different x and x'. Thus, we have $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \Pr[\mathtt{FFC}_4] + \ell \cdot \eta$.

We see that there exists a QPT adversary $\mathcal{B}_{\mathtt{ffc}}$ such that $\Pr[\mathtt{FFC}_4] \leq \ell \cdot \mathsf{Adv}^{\mathsf{ffc}}_{\mathsf{dPKE}, \mathcal{B}_{\mathtt{ffc}}}(1^\lambda)$.

Game 5: This is the same as Game 4 except $O_{\mathtt{Dec}}$ behaves as follows.

$O_{\mathtt{Dec}}$**:** On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\bot$ if there exists $j \leq i$ such that $\underline{k' = k_j \text{ and } \mathsf{ct}' = \mathsf{ct}_j}$, where i is the number of KDM queries already made at that point. Otherwise, it responds in the same way as Game 4.

We define an event $\mathtt{BD}_X$ as follows.

$\mathtt{BD}_X$**:** In Game X, $\mathcal{A}$ makes a decryption query $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$ satisfying the following conditions for some $j \leq i$, where i is the number of KDM queries already made at that point.

- It holds that $k' = k_j$ and $\mathsf{ct}' = \mathsf{ct}_j$.
- $(t', \mathsf{mac}') \neq (t_j, \mathsf{mac}_j)$.
- $\top = \mathsf{Vrfy}(\mathsf{mk}_j, t', \mathsf{mac}')$.

Suppose ct_i is correctly decrypted to s_i using sk^{k_i} for every $i \in [q_{\mathtt{kdm}}]$. In Game 4, for a decryption query $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$ such that $k' = k_j$ and $\mathsf{ct}' = \mathsf{ct}_j$ holds for some $j \in [q_{\mathtt{kdm}}]$, $O_{\mathtt{dec}}$ returns $\bot$ unless $\top = \mathsf{Vrfy}(\mathsf{mk}_j, t', \mathsf{mac}')$ (regardless of whether $\mathsf{kc}' = \mathsf{kc}_j$ holds or not). Thus, we have $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| \leq \Pr[\mathtt{BD}_5] + q_{\mathtt{kdm}} \cdot \delta_1$.

We see that now $O_{\mathtt{Dec}}$ behave as follows without using $(\mathsf{sk}^k)_{k \in [\ell]}$.

$O_{\mathtt{Dec}}$**:** On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\bot$ if there exists $j \leq i$ such that $k' = k_j$ and $\mathsf{ct}' = \mathsf{ct}_j$, where i is the number of KDM queries already made at this point. Otherwise, it responds as follows.

1. Compute $\mathsf{otp}'\|\mathsf{mk}'\|\mathsf{kc}'' \leftarrow R(\mathsf{pk}^{k'}\|\mathsf{ct}')$. Return $\bot$ if $\mathsf{kc}' \neq \mathsf{kc}''$.
2. Return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', t', \mathsf{mac}')$ and $\bot$ otherwise.

Game 6: This game is the same as Game 5 except that u_i is replaced with $u_i \oplus f_{i,b}^{V_{i-1}}(\mathbf{sk})\|0^L$ for every $i \in [q_{\mathtt{kdm}}]$, where $L = |\mathsf{mk}| + |\mathsf{kc}|$. More concretely, the behavior of $O_{\mathtt{KDM}}$ is changed as follows.

$O_{\mathtt{KDM}}$**:** On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, s_i)$.
2. Generate $u_i \leftarrow \{0,1\}^*$ and parse it as $\mathsf{otp}_i \| \mathsf{mk}_i \| \mathsf{kc}_i$.
3. Compute $t_i = \mathsf{otp}_i$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

Also, it updates the random oracle into

$$V_i(\mathsf{pk}\|x) = \begin{cases} \underline{u_j \oplus f_{j,b}^{V_{j-1}}(\mathbf{sk})\|0^L} & (\text{if } \exists j \leq i \ : \ \mathsf{pk}\|x = \mathsf{pk}^{k_j}\|s_j) \\ V_0(\mathsf{pk}\|x) & (\text{otherwise}), \end{cases}$$

This change does not affect the view of $\mathcal{A}$ since u_i is chosen uniformly at random and independently of $f_{i,b}^{V_{i-1}}(\mathbf{sk})$ for every $i \in [q_{\mathtt{kdm}}]$. Thus, we have $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = 0$ and $|\Pr[\mathtt{BD}_5] - \Pr[\mathtt{BD}_6]| = 0$.

Game 7: This game is the same as Game 6 except for the following. The challenger first generates $r \leftarrow \mathcal{M}$. The challenger then generates $\Delta^1, \ldots, \Delta^\ell \leftarrow \mathcal{M}$ and generates $(\mathsf{pk}^k, \mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda; r + \Delta^k)$ for every $k \in [\ell]$.

The above change does not affect the view of $\mathcal{A}$ since the distribution of $(\mathsf{pk}^k, \mathsf{sk}^k)_{k \in [\ell]}$ does not change. Thus, we have $|\Pr[\mathtt{SUC}_6] - \Pr[\mathtt{SUC}_7]| = 0$ and $|\Pr[\mathtt{BD}_6] - \Pr[\mathtt{BD}_7]| = 0$.

Game 8: This game is the same as Game 7 except that s_i is replaced with $r + s_i$. More concretely, the challenger generates ct_i as $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, \underline{r + s_i})$ for every $i \in [q_{\mathtt{kdm}}]$. Also, the challenger sets V_i as

$$V_i(\mathsf{pk}\|x) = \begin{cases} u_j \oplus f_{j,b}^{V_{j-1}}(\mathbf{sk})\|0^L & (\text{if } \exists j \leq i \ : \ \mathsf{pk}\|x = \mathsf{pk}^{k_j}\|\underline{r + s_j}) \\ V_0(\mathsf{pk}\|x) & (\text{otherwise}), \end{cases}$$

for every $i \in [q_{\mathtt{kdm}}]$.

We have $|\Pr[\mathtt{SUC}_7] - \Pr[\mathtt{SUC}_8]| = 0$ and $|\Pr[\mathtt{BD}_7] - \Pr[\mathtt{BD}_8]| = 0$ since this change also does not affect the view of $\mathcal{A}$.

Game 9: For every $i \in [q_{\mathtt{kdm}}]$, we define a function $\widehat{f}_{i,b}$ as described in Figure 2. Then, Game 9 is the same as Game 8 except that the challenger sets V_i as

$$V_i(\mathsf{pk}\|x) = \begin{cases} u_j \oplus \underline{\widehat{f}_{j,b}^{V_0}(x)}\|0^L & (\text{if } \exists j \leq i \ : \ \mathsf{pk}\|x = \mathsf{pk}^{k_j}\|r + s_j) \\ V_0(\mathsf{pk}\|x) & (\text{otherwise}) \end{cases}$$

for every $i \in [q_{\mathtt{kdm}}]$.

We see that for every $i \in [q_{\mathtt{kdm}}]$, $\widehat{f}_{i,b}$ correctly computes $f_{i,b}^{V_{i-1}}(\mathbf{sk})$ if it has oracle access to V_0 and is given $r + s_i$ as an input. Therefore, the functionality of V_i does not change between Game 8 and 9 for every $i \in [q_{\mathtt{kdm}}]$, and thus we have $|\Pr[\mathtt{SUC}_8] - \Pr[\mathtt{SUC}_9]| = 0$ and $|\Pr[\mathtt{BD}_8] - \Pr[\mathtt{BD}_9]| = 0$.

Game 10: This is the same as Game 9 except that the challenger sets V_i as

$$V_i(\mathsf{pk}\|x) = \begin{cases} u_j \oplus \widehat{f}_{j,b}^{V_0}(x)\|0^L & (\text{if } \exists j \leq i \ : \ \mathsf{pk} = \mathsf{pk}^{k_j} \wedge \underline{\mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j}) \\ V_0(\mathsf{pk}\|x) & (\text{otherwise}) \end{cases}$$

for every $i \in [q_{\mathtt{kdm}}]$.

$$\widehat{f}_{i,b}^{V_0}\left[(s_j,u_j,f_{j,b})_{j\in[i]},(\Delta^k)_{k\in[\ell]}\right](x):$$

```
Hardwired: $(s_j,u_j,f_{j,b})_{j\in[i]},(\Delta^k)_{k\in[\ell]}$.
Oracle $V_0$.
Input: $x \in \mathcal{M}$.
    1. Compute $w = x - s_i$ and $(\mathsf{pk}^k,\mathsf{sk}^k) \leftarrow \mathsf{dKG}(1^\lambda; w+\Delta^k)$ for every $k \in [\ell]$, and set $\mathbf{sk} = (\mathsf{sk}^1,\ldots,\mathsf{sk}^\ell)$.
    2. Repeat the following from $j = 1$ to $i-1$, where $\widehat{V}_0 = V_0$.
        (a) Compute $v_j = u_j \oplus f_{j,b}^{\widehat{V}_{j-1}}(\mathbf{sk})\|0^L$.
        (b) Set $\widehat{V}_j$ as
            $\widehat{V}_j(\mathsf{pk}\|x') = \begin{cases} v_{j'} & (\text{if } \exists j' \in [q_{\mathtt{kdm}}] : j' \le j \text{ and } \mathsf{pk}\|x' = \mathsf{pk}^{k_{j'}}\|w+s_{j'}) \\ V_0(\mathsf{pk}\|x') & (\text{otherwise}). \end{cases}$
    3. Return $f_{i,b}^{\widehat{V}_{i-1}}(\mathbf{sk})$.
```

Figure 2: The description of $\widehat{f}_{i,b}^{V_0}$.

If ct_i has a unique pre-image $r+s_i$ under pk^{k_i} for every $i \in [q_{\mathtt{kdm}}]$, the functionality of V_i does not change for every $i \in [q_{\mathtt{kdm}}]$ between Game 9 and 10. Thus, from the correctness of dPKE, we have $|\Pr[\mathtt{SUC}_9] - \Pr[\mathtt{SUC}_{10}]| \le q_{\mathtt{kdm}} \cdot \delta_2$ and $|\Pr[\mathtt{BD}_9] - \Pr[\mathtt{BD}_{10}]| \le q_{\mathtt{kdm}} \cdot \delta_2$.

Game $10+d^*$ $(d^* = 1,\ldots,d)$**:** This is the same game as Game 9 except $O_{\mathtt{KDM}}$ defers updating the random oracle. Concretely, $O_{\mathtt{KDM}}$ does not update the random oracle until $\mathcal{A}$ makes the d^*-th set of random oracle queries. The detailed description of $O_{\mathtt{KDM}}$ is as follows.

$O_{\mathtt{KDM}}$**:** On input $(k_i,f_{i,0},f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, r+s_i)$.
2. Generate $u_i \leftarrow \{0,1\}^*$ and parse it as $\mathsf{otp}_i\|\mathsf{mk}_i\|\mathsf{kc}_i$.
3. Compute $t_i = \mathsf{otp}_i$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

Also, if $\mathcal{A}$ already makes d^*-th set of queries to the random oracle, it updates the random oracle into V_i.

We have $\left|\Pr[\mathtt{SUC}_{10+d}] - \frac{1}{2}\right| = 0$ since in Game $10+d$, the view of $\mathcal{A}$ is completely independent of b. Also, we can see that there exists a QPT adversary $\mathcal{B}_{\mathtt{mac}}$ such that $\Pr[\mathtt{BD}_{10+d}] \le q_{\mathtt{kdm}} \cdot \mathsf{Adv}_{\mathsf{MAC},\mathcal{B}_{\mathtt{mac}}}^{\mathsf{sot\text{-}mac}}(1^\lambda)$.

In order to estimate $|\Pr[\mathtt{SUC}_{10+d^*-1}] - \Pr[\mathtt{SUC}_{10+d^*}]|$ for every $d^* \in [d]$, we consider the following procedure Setup_{d^*}.

Setup_{d^*}**:** First, the challenger chooses a challenge bit $b \leftarrow \{0,1\}$. The challenger also generates a fresh random oracle H and R. Next, the challenger generates ℓ key pairs $(\mathsf{pk}^k,\mathsf{sk}^k) \leftarrow \mathsf{KG}(1^\lambda; r+\Delta^k)$, where $r \leftarrow \mathcal{M}$ and $\Delta^k \leftarrow \mathcal{M}$ for every $k \in [\ell]$. The challenger sets $\mathbf{pk} := (\mathsf{pk}^1,\ldots,\mathsf{pk}^\ell)$ and V_0 as Equation (8), and executes $\mathcal{A}^{|V_0\rangle, O_{\mathtt{KDM}}, O_{\mathtt{Dec}}}(\mathbf{pk})$ just before $\mathcal{A}$ makes the d^*-th set of random oracle queries. $O_{\mathtt{KDM}}$ and $O_{\mathtt{Dec}}$ behave as follows.

$O_{\mathtt{KDM}}$**:** On the i-th call with input $(k_i,f_{i,0},f_{i,1})$, it returns CT_i generated as follows.

1. Generate $s_i \leftarrow \mathcal{M}$ and compute $\mathsf{ct}_i \leftarrow \mathsf{dEnc}(\mathsf{pk}^{k_i}, r+s_i)$.
2. Generate $u_i \leftarrow \{0,1\}^*$ and parse it as $\mathsf{otp}_i\|\mathsf{mk}_i\|\mathsf{kc}_i$.

3. Compute $t_i = \mathsf{otp}_i$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

$O_{\mathtt{Dec}}$: On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\bot$ if there exists $j \leq i$ such that $k' = k_j$ and $\mathsf{ct}' = \mathsf{ct}_j$, where i is the number of KDM queries already made at this point. Otherwise, it responds as follows.

1. Compute $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow R(\mathsf{pk}^{k'} \| \mathsf{ct}')$. Return $\bot$ if $\mathsf{kc}' \neq \mathsf{kc}''$.
2. Return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', t', \mathsf{mac}')$ and $\bot$ otherwise.

Let $\mathcal{A}$ makes i^* KDM queries before d^*-th set of random oracle queries. Then, the challenger sets V_{i^*} as

$$V_{i^*}(\mathsf{pk} \| x) = \begin{cases} u_j \oplus \widehat{f}_{j,b}^{V_0}(x) \| 0^L & (\text{if } \exists j \leq i^* \ : \ \mathsf{pk} = \mathsf{pk}^{k_j} \wedge \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ V_0(\mathsf{pk} \| x) & (\text{otherwise}). \end{cases}$$

and $S_{i^*} = \{\mathsf{pk} \| x \mid \exists j \in [i^*] \ : \ \mathsf{pk} = \mathsf{pk}^{k_j} \wedge \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j\}$. The challenger also generates $s_{i,k} \leftarrow \mathcal{M}$ and generates $\mathsf{ct}_{i,k} \leftarrow \mathsf{dEnc}(\mathsf{pk}^k, r + s_{i,k})$ for every $i \in [i^* + 1, q_{\mathtt{kdm}}]$ and $k \in [\ell]$. The challenger then sets

$$z = (|st\rangle, b, \mathbf{pk}, (\Delta^k)_{k \in [\ell]}, (k_i, f_{i,b}, s_i, \mathsf{ct}_i, u_i)_{i \in [i^*]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i \in [i^*+1, q_{\mathtt{kdm}}], k \in [\ell]}) \ , \tag{9}$$

where $|st\rangle$ is the internal state of $\mathcal{A}$ at this point. The challenger outputs $(V_{i^*}, V_0, S_{i^*}, z, O_{\mathtt{aux}} = (V_0, R))$.

Also, we consider the following QPT algorithm $\mathcal{A}_{d^*}$ that has oracle access to $O \in \{V_{i^*}, V_0\}$ and $O_{\mathtt{aux}} = (V_0, R)$.

$\mathcal{A}_{d^*}$: Given an input z, $\mathcal{A}_{d^*}$ parse it as Equation (4) and executes $\mathcal{A}^{|O\rangle, O_{\mathtt{KDM}}, O_{\mathtt{Dec}}}$ from $\mathcal{A}$'s d^*-th set of random oracle queries using $|st\rangle$ as the internal state of $\mathcal{A}$ at that point. $\mathcal{A}_{d^*}$ simulates $O_{\mathtt{KDM}}$ and $O_{\mathtt{Dec}}$ as follows.

$O_{\mathtt{KDM}}$: On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Set $\mathsf{ct}_i \leftarrow \mathsf{ct}_{i,k_i}$ (and set $s_i \leftarrow s_{i,k_i}$).
2. Generate $u_i \leftarrow \{0,1\}^*$ and parse it as $\mathsf{otp}_i \| \mathsf{mk}_i \| \mathsf{kc}_i$.
3. Compute $t_i = \mathsf{otp}_i$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

Also, it updates the random oracle that $\mathcal{A}$ gets access into

$$V_i(\mathsf{pk} \| x) = \begin{cases} u_j \oplus \widehat{f}_{j,b}^{V_0}(x) \| 0^L & (\text{if } \exists j \leq i \ : \ \mathsf{pk} = \mathsf{pk}^{k_j} \wedge \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ V_0(\mathsf{pk} \| x) & (\text{otherwise}). \end{cases}$$

$O_{\mathtt{Dec}}$: On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\bot$ if there exists $j \leq i$ such that $k' = k_j$ and $\mathsf{ct}' = \mathsf{ct}_j$, where i is the number of KDM queries already made at this point. Otherwise, it responds as follows.

1. Compute $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow R(\mathsf{pk}^{k'} \| \mathsf{ct}')$. Return $\bot$ if $\mathsf{kc}' \neq \mathsf{kc}''$.
2. Return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', t', \mathsf{mac}')$ and $\bot$ otherwise.

When $\mathcal{A}$ terminates with output b', $\mathcal{A}_{d^*}$ outputs 1 if $b = b'$ and 0 otherwise.

Suppose we execute Setup_{d^*} and $\mathcal{A}_{d^*}$ successively. They simulate the view of $\mathcal{A}$ in Game $10 + d^* - 1$ (resp. Game $10 + d^*$) if $O = V_{i^*}$ (resp. $O = V_0$). Also, $\mathcal{A}_{d^*}$ outputs 1 if and only if the output of the simulated games is 1. Thus, we have $\Pr[\mathtt{SUC}_{10+d^*-1}] = \Pr\left[1 \leftarrow \mathcal{A}_{d^*}^{|O=V_{i^*}, O_{\mathrm{aux}}=(V_0,R)\rangle}(z) : \mathsf{Setup}_{d^*}\right]$ and $\Pr[\mathtt{SUC}_{10+d^*}] = \Pr\left[1 \leftarrow \mathcal{A}_{d^*}^{|O=V_0, O_{\mathrm{aux}}=(V_0,R)\rangle}(z) : \mathsf{Setup}_{d^*}\right]$. From Lemma 3.4, there exists a QPT algorithm $\mathcal{D}_{d^*}$ such that

$$\left|\Pr[\mathtt{SUC}_{10+d^*-1}] - \Pr[\mathtt{SUC}_{10+d^*}]\right| \leq 4 \cdot \Pr\left[T \cap S_{i^*} \neq \emptyset : T \leftarrow \mathcal{D}_{d^*}^{|V_{i^*}, V_0, O_{\mathrm{aux}}=(V_0,R)\rangle}(z), \mathsf{Setup}_{d^*}\right] . \tag{10}$$

Note that $\mathcal{A}_{d^*}$ uses its oracle $O \in \{V_{i^*}, V_0\}$ only for simulating $\mathcal{A}$'s d^*-th set of random oracle queries. Thus, $\mathcal{A}_{d^*}$ make queries to O with depth 1.

We bound the right-hand side probability. In order to bound it, using $\mathcal{D}_{d^*}$, we construct the following adversary $\mathcal{B}_{d^*}$ against the SDM-OW-RSA security of dPKE.

$\mathcal{B}_{d^*}$: Given $\mathbf{pk} = (\mathsf{pk}^1, \ldots, \mathsf{pk}^\ell)$, $(\Delta^k)_k$, and $(s_{i,k}, \mathsf{ct}_{i,k})_{i \in [q_{\mathtt{kdm}}], k \in [\ell]}$, $\mathcal{B}_{d^*}$ first simulates Setup_{d^*}. $\mathcal{B}_{d^*}$ chooses a challenge bit $b \leftarrow \{0,1\}$, prepares a fresh random oracles H and R, and set V_0 as Equation (8). $\mathcal{B}_{d^*}$ then executes $\mathcal{A}^{|V_0\rangle, O_{\mathtt{KDM}}, O_{\mathtt{Dec}}}(\mathbf{pk})$ just before $\mathcal{A}$ makes the d^*-th set of random oracle queries, where $O_{\mathtt{KDM}}$ and $O_{\mathtt{Dec}}$ are simulated as follows.

$O_{\mathtt{KDM}}$: On input $(k_i, f_{i,0}, f_{i,1})$, it returns CT_i generated as follows.

1. Set $\mathsf{ct}_i \leftarrow \mathsf{ct}_{i,k_i}$ (and set $s_i \leftarrow s_{i,k_i}$).
2. Generate $u_i \leftarrow \{0,1\}^*$ and parse it as $\mathsf{otp}_i \| \mathsf{mk}_i \| \mathsf{kc}_i$.
3. Compute $t_i = \mathsf{otp}_i$ and $\mathsf{mac}_i \leftarrow \mathsf{Tag}(\mathsf{mk}_i, t_i)$.
4. Set $\mathsf{CT}_i \leftarrow (\mathsf{ct}_i, \mathsf{kc}_i, t_i, \mathsf{mac}_i)$.

$O_{\mathtt{Dec}}$: On input $(k', \mathsf{CT}') = (k', (\mathsf{ct}', \mathsf{kc}', t', \mathsf{mac}'))$, it returns $\perp$ if there exists $j \leq i$ such that $k' = k_j$ and $\mathsf{ct}' = \mathsf{ct}_j$, where i is the number of KDM queries already made at this point. Otherwise, it responds as follows.

1. Compute $\mathsf{otp}' \| \mathsf{mk}' \| \mathsf{kc}'' \leftarrow R(\mathsf{pk}^{k'} \| \mathsf{ct}')$. Return $\perp$ if $\mathsf{kc}' \neq \mathsf{kc}''$.
2. Return $t' \oplus \mathsf{otp}'$ if $\top = \mathsf{Vrfy}(\mathsf{mk}', t', \mathsf{mac}')$ and $\perp$ otherwise.

Let $\mathcal{A}$ makes i^* KDM queries before d^*-th set of random oracle queries. Then, $\mathcal{B}_{d^*}$ sets V_{i^*} as

$$V_{i^*}(\mathsf{pk}\|x) = \begin{cases} u_j \oplus \widehat{f}_{j,b}^{V_0}(x) \| 0^L & (\text{if } \exists j \leq i^* \ : \ \mathsf{pk} = \mathsf{pk}^{k_j} \land \mathsf{dEnc}(\mathsf{pk}^{k_j}, x) = \mathsf{ct}_j) \\ V_0(\mathsf{pk}\|x) & (\text{otherwise}). \end{cases}$$

$\mathcal{B}_{d^*}$ also sets

$$z = \left(|st\rangle, b, \mathbf{pk}, (\Delta^k)_{k \in [\ell]}, (k_i, f_{i,b}, s_i, \mathsf{ct}_i, u_i)_{i \in [i^*]}, (s_{i,k}, \mathsf{ct}_{i,k})_{i \in [i^*+1, q_{\mathtt{kdm}}], k \in [\ell]}\right) ,$$

where $|st\rangle$ is the internal state of $\mathcal{A}$ at this point. Finally, $\mathcal{B}_{d^*}$ outputs $T \leftarrow \mathcal{D}_{d^*}^{|V_{i^*}, V_0, O_{\mathrm{aux}}=(V_0,R)\rangle}(z)$.

$\mathcal{B}_{d^*}$ perfectly simulates a successive execution of Setup_{d^*} and $\mathcal{D}_{d^*}$. Also, in the simulated execution, if $T \cap S_{i^*} \neq \emptyset$ occurs and ct_i has a unique pre-image $r + s_i$ under pk^{k_i} for every $i \in [q_{\mathtt{kdm}}]$, $\mathcal{B}_{d^*}$ wins. Thus, we have

$$\Pr\left[T \cap S_{i^*} \neq \emptyset : T \leftarrow \mathcal{D}_{d^*}^{|V_{i^*}, V_0, O_{\mathrm{aux}}=(V_0,R)\rangle}(z), \mathsf{Setup}_{d^*}\right] \leq \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE}, \ell, q_{\mathtt{kdm}}, \mathcal{B}_{d^*}}(1^\lambda) + q_{\mathtt{kdm}} \cdot \delta_2.$$

From the discussions so far, by setting $\mathcal{B}'$ as $\mathcal{B}_{d^*}$ such that $\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE},\ell,q_{\mathtt{kdm}},\mathcal{B}_{d^*}}(1^\lambda) \le \mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE},\ell,q_{\mathtt{kdm}},\mathcal{B}'}(1^\lambda)$ for every $d^* \in [d]$, we see that there exists a QPT $\mathcal{B}'$ such that $|\Pr[\mathtt{SUC}_{10}] - \Pr[\mathtt{SUC}_{10+d}]| \le 4d \cdot (\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE},\ell,q_{\mathtt{kdm}},\mathcal{B}'}(1^\lambda) + q_{\mathtt{kdm}} \cdot \delta_2)$.

Similarly, we can show that there exists a QPT $\mathcal{B}''$ such that $|\Pr[\mathtt{BD}_{10}] - \Pr[\mathtt{BD}_{10+d}]| \le 4d \cdot (\mathsf{Adv}^{\mathsf{sdm\text{-}ow\text{-}rsa}}_{\mathsf{dPKE},\ell,q_{\mathtt{kdm}},\mathcal{B}''}(1^\lambda) + q_{\mathtt{kdm}} \cdot \delta_2)$. Note that we can efficiently check whether $\mathtt{BD}_X$ occurs or not without using $(\mathsf{sk}^k)_{k\in[\ell]}$.

From the discussions so far, by setting $\mathcal{B}$ appropriately, we see that there exists $\mathcal{B}$, $\mathcal{B}_{\mathtt{ffc}}$, and $\mathcal{B}_{\mathtt{mac}}$ satisfying Equation (7). □ **(Theorem B.9)**

C Non-Adaptively KDM Secure SKE

Zhang [Zha19] showed that a simple random oracle based SKE scheme satisfies non-adaptive KDM-CPA security with security bound roughly $\sqrt{\frac{\mathrm{poly}(q,q_{\mathtt{kdm}},q_f,\ell)}{2^\lambda}}$, where q and $q_{\mathtt{kdm}}$ are the number of (superposition) random queries and (classical) KDM queries made by adversaries, q_f is the number of (classical) random oracle queries made by KDM functions, ℓ be the number of secret keys, and λ is the length of secret keys. In this section, we show the construction's KDM-CPA security with better security bound $\frac{\mathrm{poly}(q,q_{\mathtt{kdm}},q_f,\ell)}{2^\lambda}$ using our proof strategy.

C.1 Definition

A secret-key encryption (SKE) scheme SKE is a three tuple $(\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ of PPT algorithms. Let $\mathcal{M}$ be the message space of SKE. The key generation algorithm Gen, given a security parameter 1^λ, outputs a secret key sk. The encryption algorithm E, given a secret key sk and message $m \in \mathcal{M}$, outputs a ciphertext ct. The decryption algorithm D, given a secret key sk and ciphertext ct, outputs a message $\tilde{m} \in \{\bot\} \cup \mathcal{M}$. As correctness, we require $\mathsf{D}(\mathsf{sk}, \mathsf{E}(\mathsf{sk}, m)) = m$ for every $m \in \mathcal{M}$ and $\mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$.

Definition C.1 (Non-adaptive KDM-CPA security for SKE). *Let $\mathsf{SKE} = (\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ be an SKE scheme. We define $\mathsf{Exp}^{\mathsf{na\text{-}kdm\text{-}cpa}}_{\mathsf{SKE},\ell,(k_i,f_{i,0},f_{i,1})_{i\in[q_{\mathtt{kdm}}]},\mathcal{A}}(1^\lambda)$ for an adversary $\mathcal{A}$ as follows, where $k_i \in [\ell]$ and $f_{i,0}$ and $f_{i,1}$ are efficiently computable functions of the same output length for every $i \in [q_{\mathtt{kdm}}]$.*

Initialize: *The challenger chooses the challenge bit $b \leftarrow \{0,1\}$, generates $\mathsf{sk}^k \leftarrow \mathsf{Gen}(1^\lambda)$ for every $k \in [\ell]$, and sets $\mathbf{sk} = (\mathsf{sk}^1, \dots, \mathsf{sk}^\ell)$. The challenger generates $\mathsf{ct}_i \leftarrow \mathsf{E}(\mathsf{sk}^{k_i}, f_{i,b}(\mathbf{sk}))$ for every $i \in [q_{\mathtt{kdm}}]$. Then, the challenger executes $b' \leftarrow \mathcal{A}((\mathsf{ct}_i)_{i\in[q_{\mathtt{kdm}}]})$.*

Finalize: *The challenger outputs 1 if $b = b'$ and 0 otherwise.*

We say that SKE is non-adaptively KDM-CPA secure if for any polynomial $\ell = \ell(\lambda)$ and $q_{\mathtt{kdm}} = q_{\mathtt{kdm}}(\lambda)$, tuples $(k_i, f_{i,0}, f_{i,1})$, and QPT adversary $\mathcal{A}$, we have

$$\mathsf{Adv}^{\mathsf{na\text{-}kdm\text{-}cpa}}_{\mathsf{SKE},\ell,(k_i,f_{i,0},f_{i,1})_{i\in[q_{\mathtt{kdm}}]},\mathcal{A}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{na\text{-}kdm\text{-}cpa}}_{\mathsf{SKE},\ell,(k_i,f_{i,0},f_{i,1})_{i\in[q_{\mathtt{kdm}}]},\mathcal{A}}(1^\lambda)\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

C.2 Additional Lemma

The following lemma is used to prove the non-adaptive KDM security of the SKE scheme in Appendix C.3.

Lemma C.2 (Inverting QRO with correlated inputs). *Let $\ell = \ell(\lambda)$ be a polynomial and $\mathcal{A}$ be an oracle QPT algorithm that makes at most q queries with query depth d, and outputs a list T of size at most t as the final output. We consider the following $\mathsf{Exp}^{\mathsf{ci\text{-}inv}}_{\ell,\mathcal{A}}(1^\lambda)$.*

Initialization *The challenger first generate a fresh random oracle $H : X \to Y$. Then, the challenger also generates $s \leftarrow X$ and $\Delta^k \leftarrow X$ for every $k \in [\ell]$. Then, the challenger executes $T \leftarrow \mathcal{A}^{|H\rangle}((\Delta^k, H(s + \Delta^k))_{k\in[\ell]})$.*

Finalization *The challenger outputs* 1 *if T contains z such that $z = s + \Delta^k$ holds for some $k \in [\ell]$ and* 0 *otherwise.*

Then, we have $\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{ci\text{-}inv}}_{\ell,\mathcal{A}}(1^\lambda)\right] \leq \frac{4(d+2)(q+t)\ell}{|X|}$.

Proof. Let $\widehat{\mathcal{A}}$ be a QPT adversary that runs in the same way as $\mathcal{A}$ except that before it terminates, $\widehat{\mathcal{A}}$ computes and discards $H(z)$ for every $z \in T$, where T is the final output of $\mathcal{A}$. Then, $\mathcal{A}$ makes at most $q + t$ queries to H with the query depth $d + 1$, and we have $\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{ci\text{-}inv}}_{\ell,\mathcal{A}}(1^\lambda)\right] = \Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{ci\text{-}inv}}_{\ell,\widehat{\mathcal{A}}}(1^\lambda)\right]$. We estimate the latter.

We complete the proof using hybrid games. We define Game 1 as $\mathsf{Exp}^{\mathsf{ci\text{-}inv}}_{\ell,\widehat{\mathcal{A}}}(1^\lambda)$. Let $\mathtt{SUC}_X$ be the event that the challenger outputs 1 as the final output in Game X. We also let $S = \{s + \Delta^1, \ldots, s + \Delta^\ell\}$.

Game 2: This game is the same as Game 1 except that V defined as

$$V(x) = \begin{cases} y^k & (\text{if } \exists k \in [\ell] \ : \ x = s + \Delta_k) \\ H(x) & (\text{otherwise}), \end{cases}$$

is used instead of H, where $y^k \leftarrow Y$ for every $k \in [\ell]$.

We have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| = 0$ since Game 1 and 2 are exactly the same from the view of $\widehat{\mathcal{A}}$.

Game 3: This game is the same as Game 2 except that $\widehat{\mathcal{A}}$ gets access to the punctured oracle $H \setminus S$. In other words, the challenger executes $T \leftarrow \mathcal{A}^{|H\setminus S\rangle}((\Delta^k, y^k)_{k\in[\ell]})$ at the end of the initialization step.

Let $\mathtt{Find}_X$ be the event that the punctured oracle $H \setminus S$ returns 1 in Game X. From the definition of $\widehat{\mathcal{A}}$, we have $\Pr\left[\mathtt{SUC}_3 \wedge \neg\mathtt{Find}_3\right] = 0$. Thus, we have

$$\sqrt{\Pr[\mathtt{SUC}_2]} = \left|\sqrt{\Pr[\mathtt{SUC}_2]} - \sqrt{\Pr\left[\mathtt{SUC}_3 \wedge \neg\mathtt{Find}_3\right]}\right| .$$

By applying Lemma 3.2, we obtain

$$\left|\sqrt{\Pr[\mathtt{SUC}_2]} - \sqrt{\Pr\left[\mathtt{SUC}_3 \wedge \neg\mathtt{Find}_3\right]}\right| \leq \sqrt{(d+2)\cdot\Pr[\mathtt{Find}_3]} .$$

Therefore, we also obtain $\Pr[\mathtt{SUC}_2] \leq (d+2)\Pr[\mathtt{Find}_3]$.

Finally, we bound $\Pr[\mathtt{Find}_3]$. In Game 3, conditioned on $(\Delta^k, y^k)_k$, we have $\Pr_{s\leftarrow X}[x \in S] \leq \frac{\ell}{|X|}$ for any $x \in X$. Thus, from Lemma 3.3, we obtain $\Pr[\mathtt{Find}_3] \leq \frac{4(q+t)\ell}{|X|}$.

From the discussions so far, we obtain $\Pr\left[1 \leftarrow \mathsf{Exp}^{\mathsf{ci\text{-}inv}}_{\ell,\mathcal{A}}(1^\lambda)\right] \leq \frac{4(d+2)(q+t)\ell}{|X|}$. □ (**Lemma C.2**)

C.3 Construction

Construction C.3. Let $H : \{0,1\}^{2\lambda} \to \{0,1\}^*$ be a hash function. We construct the following $\mathsf{SKE}_{\mathtt{kdm}} = (\mathsf{Gen}, \mathsf{E}, \mathsf{D})$.

$\mathsf{Gen}(1^\lambda)$**:** Return $\mathsf{sk} \leftarrow \{0,1\}^\lambda$.

$\mathsf{E}(\mathsf{sk}, \mathsf{m})$**:** Generate $s \leftarrow \{0,1\}^\lambda$, compute $t = H(\mathsf{sk} \| s) \oplus \mathsf{m}$, and return (s,t).

$\mathsf{D}(\mathsf{sk}, \mathsf{ct})$**:** Parse $\mathsf{ct} = (s,t)$ and return $t \oplus H(\mathsf{sk} \| s)$.

The construction clearly satisfies correctness.

C.4 Security Proof

We prove the following theorem.

Theorem C.4. *Let $\ell = \ell(\lambda)$ and $q_{\mathtt{kdm}} = q_{\mathtt{kdm}}(\lambda)$ be polynomials. Let $k_i \in [\ell]$ and $f_{i,0}$ and $f_{i,1}$ are efficiently computable functions of the same output length for every $i \in [q_{\mathtt{kdm}}]$. Let $\mathcal{A}$ be any (possibly computationally unbounded) adversary against the non-adaptive KDM-CPA security of* $\mathsf{SKE}_{\mathtt{kdm}}$ *making q (superposition) random oracle queries to H with query depth d. Also, let q_f be the upper bound of the total number of (classical) random oracle queries made by KDM functions queried by $\mathcal{A}$. Then, it holds that*

$$\mathsf{Adv}^{\mathsf{na\text{-}kdm\text{-}cpa}}_{\mathsf{SKE}_{\mathtt{kdm}}, \ell, (k_i, f_{i,0}, f_{i,1})_{i \in [q_{\mathtt{kdm}}]}, \mathcal{A}}(1^\lambda) \leq \frac{O(q_f \cdot q_{\mathtt{kdm}} + d^2 \cdot q \cdot \ell)}{2^\lambda} .$$

Proof. We prove this theorem using hybrid games. Let $\mathtt{SUC}_X$ be the event that $\mathcal{A}$ wins in Game X.

Game 1**:** This is $\mathsf{Exp}^{\mathsf{na\text{-}kdm\text{-}cpa}}_{\mathsf{SKE}_{\mathtt{kdm}}, \ell, (k_i, f_{i,0}, f_{i,1})_{i \in [q_{\mathtt{kdm}}]}, \mathcal{A}}(1^\lambda)$.

Initialize: The challenger chooses the challenge bit $b \leftarrow \{0,1\}$, generates $\mathsf{sk}^k \leftarrow \{0,1\}^\lambda$ for every $k \in [\ell]$, and sets $\mathbf{sk} = (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$. The challenger generates $\mathsf{ct}_i = (s_i, t_i)$ for every $i \in [q_{\mathtt{kdm}}]$ as follows.

1. Generate $s_i \leftarrow \{0,1\}^\lambda$.
2. Compute $t_i = H(\mathsf{sk}^{k_i} \| s_i) \oplus f^H_{i,b}(\mathbf{sk})$.

Then, the challenger executes $b' \leftarrow \mathcal{A}^{|H\rangle}((\mathsf{ct}_i)_{i \in [q_{\mathtt{kdm}}]})$.

Finalize: The challenger outputs 1 if $b = b'$ and 0 otherwise.

Game 2**:** This game is the same as Game 1 except that H is replaced with

$$V(x \| w) = \begin{cases} \underline{u_i} & \underline{(\text{if } \exists i \in [q_{\mathtt{kdm}}] \ : \ x \| w = \mathsf{sk}^{k_i} \| s_i)} \\ H(x \| w) & (\text{otherwise}), \end{cases}$$

where $u_i \leftarrow \{0,1\}^*$ for every $i \in [q_{\mathtt{kdm}}]$.

Game 1 and 2 are exactly the same from the view of $\mathcal{A}$. Thus, we have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| = 0$.

Game 3**:** This game is the same as Game 2 except that KDM functions $f_{i,b}$ gets access to $\underline{H}$ instead of V for every $i \in [q_{\mathtt{kdm}}]$.

$$\widehat{f}^H_{i,b}\left[f_{i,b}, k_i, (\Delta^k)_{k\in[\ell]}\right](x):$$

Hardwired: $f_{i,b}, k_i, (\Delta^k)_{k\in[\ell]}$.
Oracle H.
Input: $x \in \{0,1\}^\lambda$.

1. Compute $\mathsf{sk}^k = x \oplus \Delta^{k_i} \oplus \Delta^k$ for every $k \in [\ell]$, and set $\mathbf{sk} = (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$.
2. Return $f^H_{i,b}(\mathbf{sk})$.

Figure 3: The description of $\widehat{f}^H_{i,b}$.

Game 2 and 3 differs only when $f_{i,b}$ calls one of $\mathsf{sk}^{k_1}\|s_1, \ldots, \mathsf{sk}^{k_{q_{\mathtt{kdm}}}}\|s_{q_{\mathtt{kdm}}}$ for some $i \in [q_{\mathtt{kdm}}]$. Since $s_1, \ldots, s_{q_{\mathtt{kdm}}}$ are chosen uniformly at random and independently from $(f_{i,b})_{i\in[q_{\mathtt{kdm}}]}$ and $(\mathsf{sk}^k)_{k\in[\ell]}$, we have $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq \frac{q_f \cdot q_{\mathtt{kdm}}}{2^\lambda}$.

Game 4: This game is the same as Game 3 except that u_i is replaced with $u_i \oplus f^H_{i,b}(\mathbf{sk})$ for every $i \in [q_{\mathtt{kdm}}]$. Concretely, t_i is set as $t_i \leftarrow u_i$ for every $i \in [q_{\mathtt{kdm}}]$, and V is defined as

$$V(x\|w) = \begin{cases} \underline{u_i \oplus f^H_{i,b}(\mathbf{sk})} & (\text{if } \exists i \in [q_{\mathtt{kdm}}] \ : \ x\|w = \mathsf{sk}^{k_i}\|s_i) \\ H(x\|w) & (\text{otherwise}). \end{cases}$$

This change does not affect the view of $\mathcal{A}$ since u_i is chosen uniformly at random and completely independent of $f^H_{i,b}(\mathbf{sk})$ for every $i \in [\ell]$. Thus, we have $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| = 0$.

Game 5: This game is the same as Game 4 except how the challenger generates $\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell$. Concretely, the challenger first generates $\mathsf{sk} \leftarrow \{0,1\}^\lambda$ and $\Delta^k \leftarrow \{0,1\}^\lambda$ for every $k \in [\ell]$. Then, the challenger sets $\underline{\mathsf{sk}^k = \mathsf{sk} \oplus \Delta^k \text{ for every } k \in [\ell]}$.

We have $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| = 0$ since the change does not affect the view of $\mathcal{A}$.

From the next game, we use the function $\widehat{f}_{i,b}$ described in Figure 3. $\widehat{f}_{i,b}$ is designed so that it computes $f^H_{i,b}(\mathbf{sk})$ if it has oracle access to H and is given sk^{k_i} as an input.

Game 6: This game is the same as Game 5 except that V is defined as

$$V(x\|w) = \begin{cases} u_i \oplus \underline{\widehat{f}^H_{i,b}(x)} & (\text{if } \exists i \in [q_{\mathtt{kdm}}] \ : \ x\|w = \mathsf{sk}^{k_i}\|s_i) \\ H(x\|w) & (\text{otherwise}). \end{cases}$$

Since for every $i \in [q_{\mathtt{kdm}}]$, $\widehat{f}_{i,b}$ correctly computes $f^H_{i,b}(\mathbf{sk})$ if it has oracle access to H and is given sk^{k_i} as an input, the functionality of V does not change between Game 5 and 6. Therefore, we have $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = 0$.

Game 7: This game is the same as Game 6 except that V is defined as

$$V(x\|w) = \begin{cases} u_i & (\text{if } \exists i \in [q_{\mathtt{kdm}}] \ : \ \underline{P(x)\|w = P(\mathsf{sk}^{k_i})\|s_i}) \\ H(x\|w) & (\text{otherwise}), \end{cases}$$

where $P : \{0,1\}^\lambda \to \{0,1\}^{3\lambda}$ is a random function.

If P is injective, Game 6 and 7 are exactly the same from the view of $\mathcal{A}$. Thus, we have $|\Pr[\mathtt{SUC}_6] - \Pr[\mathtt{SUC}_7]| \leq \frac{2^\lambda(2^\lambda-1)}{2} \cdot \frac{1}{2^{3\lambda}} \leq \frac{1}{2^\lambda}$.

Game 8**:** This game is the same as Game 7 except that $\mathcal{A}$ gets access to H instead of V though $\mathsf{ct}_1, \ldots, \mathsf{ct}_{q_{\mathtt{kdm}}}$ are generated by using V.

We see that $\Pr[\mathtt{SUC}_8] = 1/2$ since the information of b is completely hidden from the view of $\mathcal{A}$. In order to estimate $|\Pr[\mathtt{SUC}_7] - \Pr[\mathtt{SUC}_8]|$, we consider the following procedure Setup.

Setup**:** The challenger chooses the challenge bit $b \leftarrow \{0,1\}$ and fresh random oracles H and P. The challenger then generates $\mathsf{sk} \leftarrow \{0,1\}^\lambda$ and $\Delta^k \leftarrow \{0,1\}^\lambda$ for every $k \in [\ell]$, and sets $\mathsf{sk}^k = \mathsf{sk} \oplus \Delta^k$ for every $k \in [\ell]$ and $\mathbf{sk} = (\mathsf{sk}^1, \ldots, \mathsf{sk}^\ell)$. The challenger also computes $\mathsf{pk}_k \leftarrow P(\mathsf{sk}_k)$ for every $k \in [\ell]$. Next, the challenger generates $s_i \leftarrow \{0,1\}^\lambda$ and $u_i \leftarrow \{0,1\}^*$, and sets $\mathsf{ct}_i \leftarrow (s_i, u_i)$ for every $i \in [q_{\mathtt{kdm}}]$. The challenger sets V as

$$V(x\|w) = \begin{cases} u_i \oplus \widehat{f}_{i,b}^H(x) & (\text{if } \exists i \in [q_{\mathtt{kdm}}] \ : \ P(x)\|w = \mathsf{pk}^{k_i}\|s_i) \\ H(x\|w) & (\text{otherwise}) \end{cases} ,$$

$S = \{x\|w | \exists i \in [q_{\mathtt{kdm}}] \ : \ P(x)\|w = \mathsf{pk}^{k_i}\|s_i\}$, and $z = (b, (\mathsf{ct}_i)_{i \in [q_{\mathtt{kdm}}]})$. The challenger outputs (V, H, S, z).

Also, for $O \in \{V, H\}$, we consider a QPT algorithm $\mathcal{A}'^{|O\rangle}$ that is given $z = (b, (\mathsf{ct}_i)_{i \in [q_{\mathtt{kdm}}]})$ as input, executes $b' \leftarrow \mathcal{A}^{|O\rangle}((\mathsf{ct}_i)_{i \in [q_{\mathtt{kdm}}]})$, and outputs 1 if $b = b'$ and 0 otherwise.

Suppose we execute Setup and $\mathcal{A}'$ successively. They simulate the view of $\mathcal{A}$ in Game 7 (resp. Game 8) if $O = V$ (resp. $O = H$). Also, $\mathcal{A}'$ outputs 1 if and only if the output of the simulated games is 1. Thus, we have $\Pr[\mathtt{SUC}_7] = \Pr\left[1 \leftarrow \mathcal{A}'^{|V\rangle}(z) : \mathsf{Setup}\right]$ and $\Pr[\mathtt{SUC}_8] = \Pr\left[1 \leftarrow \mathcal{A}'^{|H\rangle}(z) : \mathsf{Setup}\right]$. From Lemma 3.4, there exists a QPT algorithm $\mathcal{D}$ such that

$$|\Pr[\mathtt{SUC}_7] - \Pr[\mathtt{SUC}_8]| \leq 4d \cdot \Pr\left[T \cap S \neq \emptyset : T \leftarrow \mathcal{D}^{|V,H\rangle}(z), \mathsf{Setup}\right] .$$

We estimate the right-hand side probability of the above inequality. This can be done by using Lemma C.2. Consider the following adversary $\mathcal{B}$ run in $\mathsf{Exp}_{\ell,\mathcal{B}}^{\mathsf{ci\text{-}inv}}(1^\lambda)$.

$\mathcal{B}$**:** $\mathcal{B}$ has oracle access to P. Given $(\Delta^k, \mathsf{pk}^k)_{k \in [\ell]}$ as an input, $\mathcal{B}$ first chooses the challenge bit $b \leftarrow \{0,1\}$ and fresh random oracles H. Next, $\mathcal{B}$ generates $s_i \leftarrow \{0,1\}^\lambda$ and $u_i \leftarrow \{0,1\}^*$, and sets $\mathsf{ct}_i \leftarrow (s_i, u_i)$ for every $i \in [q_{\mathtt{kdm}}]$. $\mathcal{B}$ sets V as

$$V(x\|w) = \begin{cases} u_i \oplus \widehat{f}_{i,b}^H(x) & (\text{if } \exists i \in [q_{\mathtt{kdm}}] \ : \ P(x)\|w = \mathsf{pk}^{k_i}\|s_i) \\ H(x\|w) & (\text{otherwise}) \end{cases} ,$$

and $z = (b, (\mathsf{ct}_i)_{i \in [q_{\mathtt{kdm}}]})$. $\mathcal{B}$ outputs $T \leftarrow \mathcal{D}^{|V,H\rangle}(z)$.

$\mathcal{B}$ perfectly simulates a successive execution of Setup and $\mathcal{D}$. In the simulated execution, if $T \cap S \neq \emptyset$ occurs and P is injective, $\mathcal{B}$ wins. Thus, from Lemma C.2, we have $\Pr\left[T \cap S \neq \emptyset : T \leftarrow \mathcal{D}^{|V,H\rangle}(z), \mathsf{Setup}\right] \leq \frac{O(d \cdot q \cdot \ell)}{2^\lambda}$.

From the discussions so far, we obtain

$$\begin{aligned} \mathsf{Adv}^{\mathsf{na\text{-}kdm\text{-}cpa}}_{\mathsf{SKE}_{\mathtt{kdm}},\ell,(k_i,f_{i,0},f_{i,1})_{i \in [q_{\mathtt{kdm}}]},\mathcal{A}}(1^\lambda) &\leq \frac{q_f \cdot q_{\mathtt{kdm}}}{2^\lambda} + \frac{1}{2^\lambda} + 4d \cdot \frac{O(d \cdot q \cdot \ell)}{2^\lambda} \\ &= \frac{O(q_f \cdot q_{\mathtt{kdm}} + d^2 \cdot q \cdot \ell)}{2^\lambda} . \end{aligned}$$