Simple and Efficient KDM-CCA Secure Public Key Encryption

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

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

Abstract

We propose two efficient public key encryption (PKE) schemes satisfying key dependent message security against chosen ciphertext attacks (KDM-CCA security). The first one is KDM-CCA secure with respect to affine functions. The other one is KDM-CCA secure with respect to polynomial functions. Both of our schemes are based on the KDM-CPA secure PKE schemes proposed by Malkin, Teranishi, and Yung (EUROCRYPT 2011). Although our schemes satisfy KDM-CCA security, their efficiency overheads compared to Malkin et al.'s schemes are very small. Thus, efficiency of our schemes is drastically improved compared to the existing KDM-CCA secure schemes.

We achieve our results by extending the construction technique by Kitagawa and Tanaka (ASIACRYPT 2018). Our schemes are obtained via semi-generic constructions using an IND-CCA secure PKE scheme as a building block. We prove the KDM-CCA security of our schemes based on the decisional composite residuosity (DCR) assumption and the IND-CCA security of the building block PKE scheme.

Moreover, our security proofs are *tight* if the IND-CCA security of the building block PKE scheme is tightly reduced to its underlying computational assumption. By instantiating our schemes using existing tightly IND-CCA secure PKE schemes, we obtain the first tightly KDM-CCA secure PKE schemes whose ciphertext consists only of a constant number of group elements.

Keywords: key dependent message security, chosen ciphertext security

Contents

1 Introduction

1.1 Background

Key dependent message (KDM) security, introduced by Black, Rogaway, and Shrimpton [3], guarantees confidentiality of communication even if an adversary can get a ciphertext of secret keys. KDM security is defined with respect to a function family $\mathcal{F}$. Informally, a public key encryption (PKE) scheme is said to be $\mathcal{F}$-KDM secure if confidentiality of messages is protected even when an adversary can see a ciphertext of $f(\mathsf{sk}_1, \cdots, \mathsf{sk}_\ell)$ under the k-th public key for any $f \in \mathcal{F}$ and $k \in \{1, \cdots, \ell\}$, where ℓ denotes the number of keys. KDM security is useful for many practical applications including anonymous credential systems [8] and hard disk encryption systems (e.g., BitLocker [5]).

In this paper, we focus on constructing *efficient* PKE schemes that satisfy KDM security against chosen ciphertext attacks, namely *KDM-CCA* security, in the standard model. As pointed out by Camenisch, Chandran, and Shoup [7] who proposed the first KDM-CCA secure PKE scheme, KDM-CCA security is well motivated since it resolves key wrapping problems that arise in many practical applications. Moreover, in some applications of KDM secure schemes such as anonymous credential systems, we should consider active adversaries and need KDM-CCA security.

The first attempt to construct an efficient KDM secure PKE scheme was made by Applebaum, Cash, Peikert, and Sahai [1]. They proposed a PKE scheme that is KDM-CPA secure with respect to affine functions ($\mathcal{F}_{\mathsf{aff}}$-KDM-CPA secure) under a lattice assumption. Their scheme is as efficient as IND-CPA secure schemes based on essentially the same assumption.

Malkin, Teranishi, and Yung [23] later proposed a more efficient KDM-CPA secure PKE scheme under the decisional composite residuosity (DCR) assumption [25, 10]. Moreover, their scheme is KDM-CPA secure with respect to polynomial functions ($\mathcal{F}_{\mathsf{poly}}$-KDM-CPA secure), which is much richer than affine functions. A ciphertext of their scheme contains $d+1$ group elements, where d is the maximum degree of polynomial functions with respect to which their scheme is KDM-CPA secure. As a special case of $d = 1$, their scheme is an $\mathcal{F}_{\mathsf{aff}}$-KDM-CPA secure PKE scheme whose ciphertext consists of only two group elements.

Due to these works, we now have efficient KDM-CPA secure PKE schemes. As we can see, the above $\mathcal{F}_{\mathsf{aff}}$-KDM-CPA secure schemes are as efficient as PKE schemes that are IND-CPA secure under the same assumptions. However, the situation is somewhat unsatisfactory when considering KDM-CCA secure PKE.

Camenisch et al. [7] proposed the first KDM-CCA secure PKE scheme based on the Naor-Yung paradigm [24]. They showed that for any function class $\mathcal{F}$, an $\mathcal{F}$-KDM-CPA secure PKE scheme can be transformed into an $\mathcal{F}$-KDM-CCA secure one assuming a non-interactive zero knowledge (NIZK) proof system. They also showed a concrete instantiation based on the decisional Diffie-Hellman (DDH) assumption on bilinear groups. A ciphertext of their scheme contains $O(\lambda)$ group elements, where λ is the security parameter. Subsequently, Hofheinz [13] showed a more efficient KDM-CCA secure PKE scheme. His scheme is circular-CCA secure, relying on both the DCR and DDH assumptions, and decisional linear (DLIN) assumption on bilinear groups. A ciphertext of his scheme contains more than 50 group elements. Recently, Libert and Qian [21] improved the construction of Hofheinz based on the 3-party DDH (D3DH) assumption on bilinear groups, and shortened the ciphertext size by about 20 group elements.

The first KDM-CCA secure PKE scheme using neither NIZK proofs nor bilinear maps was proposed by Lu, Li, and Jia [22]. They claimed their scheme is $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure based on both the DCR and DDH assumptions. However, a flaw in their security proof was later pointed out by Han, Liu, and Lyu [12]. Han et al. also showed a new $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure scheme based on Lu et al.'s construction methodology, and furthermore constructed a $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA

secure PKE scheme. Their schemes rely on both the DCR and DDH assumptions. A ciphertext of their $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure scheme contains around 20 group elements. A ciphertext of their $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA secure scheme contains $O(d^9)$ group elements, where d is the maximum degree of polynomial functions.

Recently, Kitagawa and Tanaka [19] showed a new framework for constructing KDM-CCA secure schemes. Using the framework, they constructed an $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure PKE scheme based solely on the DDH assumption (without bilinear maps). However, their scheme is somewhat inefficient and its ciphertext consists of $O(\lambda)$ group elements.

The currently most efficient KDM-CCA secure PKE scheme is that of Han et al.. Their schemes are much efficient compared to other KDM-CCA secure schemes. However, there are still a large overhead compared to efficient KDM-CPA secure schemes. Especially, its overhead compared to Malkin et al.'s scheme is large even though Han et al.'s schemes are based on both the DDH and DCR assumptions while Malkin et al.'s scheme is based only on the DCR assumption.

In order to use a KDM-CCA secure PKE scheme in practical applications, we need a more efficient scheme.

1.2 Our Results

We propose two efficient KDM-CCA secure PKE schemes. The first one is $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure, and the other one is $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA secure. Both of our schemes are based on the KDM-CPA secure scheme proposed by Malkin et al. [23]. Although our schemes satisfy KDM-CCA security, its efficiency overheads compared to Malkin et al.'s schemes are very small. Thus, efficiency of our schemes is drastically improved compared to the previous KDM-CCA secure schemes.

We achieve our results by extending the construction technique by Kitagawa and Tanaka [19]. Our schemes are obtained via semi-generic constructions using an IND-CCA secure PKE scheme as a building block. By instantiating the underlying IND-CCA secure PKE scheme with the factoring-based scheme by Hofheinz and Kiltz [17] (and with some optimization techniques), we obtain KDM-CCA secure PKE schemes (with respect to affine functions and with respect to polynomials) such that the overhead of the ciphertext size of our schemes compared to Malkin et al.'s KDM-CPA secure scheme can be less than a single DCR-group element. (See Figures 1 and 2.)

Moreover, our security proofs are *tight* if the IND-CCA security of the building block PKE scheme is tightly reduced to its underlying computational assumption. By instantiating our schemes using existing tightly IND-CCA secure PKE schemes [14, 11], we obtain the first tightly KDM-CCA secure PKE schemes whose ciphertext consists only of a constant number of group elements. To the best of our knowledge, prior to our work, the only way to construct a tightly KDM-CCA secure PKE scheme is to instantiate the construction proposed by Camenisch et al. [7] using a tightly secure NIZK proof system such as the one proposed by Hofheinz and Jager [15]. A ciphertext of such schemes consists of $O(\lambda)$ group elements, where λ is the security parameter.

For a comparison of efficiency between our schemes and existing schemes, see Figures 1 and 2. In the figures, for reference, we include [23] on which our schemes are based but which is not KDM-CCA secure. In the figures, we also show concrete instantiations of our constructions. The details of these instantiations are explained in Section 7.

We note that the plaintext space of the schemes listed in Figures 1 and 2 except for our schemes and Malkin et al.'s [23], is smaller than the secret key space, and some modifications are needed for encrypting a whole secret key, which will result in a larger ciphertext size in

Scheme	Assumption	Ciphertext size	Tight?
[23] (not CCA)	DCR	$2\lvert\mathbb{Z}_{N^s}\rvert$	
[7] with [15, § 4]	DLIN	$O(\lambda)\lvert\mathbb{G}_{\mathsf{bi}}\rvert$	✓
[13] (Circular)	DCR+DDH$^{(\dagger)}$ & DLIN	$6\lvert\mathbb{Z}_{N^3}\rvert + 50\lvert\mathbb{G}_{\mathsf{bi}}\rvert + \mathsf{OH}_{\mathsf{ch\&sig}}$	
[21] (Circular)	DCR+DDH$^{(\dagger)}$ & D3DH	$6\lvert\mathbb{Z}_{N^3}\rvert + 31\lvert\mathbb{G}_{\mathsf{bi}}\rvert + \mathsf{OH}_{\mathsf{ch\&sig}}$	
[12]	DCR+DDH$^{(\ddagger)}$	$9\lvert\mathbb{Z}_{N^s}\rvert + 9\lvert\mathbb{Z}_{N^2}\rvert + 2\lvert\mathbb{Z}_{\bar{N}}\rvert + \lvert\mathbb{Z}_N\rvert + \mathsf{OH}_{\mathsf{ae}}$	
[19]	DDH	$O(\lambda)\lvert\mathbb{G}_{\mathsf{ddh}}\rvert$	
Ours (§ 5)	DCR & CCAPKE	$2\lvert\mathbb{Z}_{N^s}\rvert + \lvert\pi_{\mathsf{phf}}\rvert + \mathsf{OH}_{\mathsf{cca}}$	
with [17] & CRHF	DCR	$2\lvert\mathbb{Z}_{N^s}\rvert + 2\lvert\mathbb{Z}_{N'}\rvert + \mathsf{len}_{\mathsf{crhf}}$	
with [14]	DCR	$3\lvert\mathbb{Z}_{N^s}\rvert + 28\lvert\mathbb{Z}_{N'^2}\rvert + \mathsf{OH}_{\mathsf{ae}}$	✓
with [11]	DCR & DDH	$3\lvert\mathbb{Z}_{N^s}\rvert + 3\lvert\mathbb{G}_{\mathsf{ddh}}\rvert + \mathsf{OH}_{\mathsf{ae}}$	✓

Figure 1: Comparison of KDM-CCA secure PKE schemes with respect to affine functions. The last three rows are instantiation examples of our scheme. In the "Ciphertext size" column, we use the following notations: N and N' are RSA moduli, and $s \geq 2$ is the exponent of N in the DCR setting; $\bar{N} = 2N + 1$; For a group G, $|G|$ denotes the size of an element in G; $\mathbb{G}_{\mathsf{bi}}$ denotes a group equipped with a bilinear map, and $\mathbb{G}_{\mathsf{ddh}}$ denotes a DDH-hard group (without bilinear maps); $|\pi_{\mathsf{phf}}|$ denotes the output size of the underlying projective hash function; $\mathsf{OH}_{\mathsf{cca}}$ (resp. $\mathsf{OH}_{\mathsf{ae}}$) denotes the ciphertext overhead of the underlying IND-CCA secure PKE (resp. authenticated encryption) scheme; $\mathsf{OH}_{\mathsf{ch\&sig}}$ denotes an overhead caused by the underlying chameleon hash function and one-time signature scheme; $\mathsf{len}_{\mathsf{crhf}}$ denotes the output size of a collision resistant hash function; For λ-bit security, $\mathsf{OH}_{\mathsf{ae}} = \lambda$, $\mathsf{len}_{\mathsf{crhf}} = 2\lambda$, and $\mathsf{OH}_{\mathsf{ch\&sig}}$ can be smaller than $|\mathbb{Z}_N|$. $^{(\dagger)}$ DDH in the order-$\frac{\phi(N)}{4}$ subgroup of $\mathbb{Z}^*_{N^3}$. $^{(\ddagger)}$ DDH in $\mathbb{QR}_{\bar{N}} := \{a^2 \bmod \bar{N} | a \in \mathbb{Z}^*_{\bar{N}}\}$.

Scheme	Assumption	Ciphertext size	Tight?
[23] (not CCA)	DCR	$(d+1)\lvert\mathbb{Z}_{N^s}\rvert$	
[12]	DCR+DDH$^{(\ddagger)}$	$(8d^9+1)\lvert\mathbb{Z}_{N^s}\rvert + 9\lvert\mathbb{Z}_{N^2}\rvert + 2\lvert\mathbb{Z}_{\bar{N}}\rvert + \lvert\mathbb{Z}_N\rvert + \mathsf{OH}_{\mathsf{ae}}$	
Ours (§ 6)	DCR & CCAPKE	$(d+1)\lvert\mathbb{Z}_{N^s}\rvert + \lvert\pi_{\mathsf{phf}}\rvert + \mathsf{OH}_{\mathsf{cca}}$	
with [17] & CRHF	DCR	$(d+1)\lvert\mathbb{Z}_{N^s}\rvert + 2\lvert\mathbb{Z}_{N'}\rvert + \mathsf{len}_{\mathsf{crhf}}$	
with [14]	DCR	$(2d+1)\lvert\mathbb{Z}_{N^s}\rvert + 28\lvert\mathbb{Z}_{N'^2}\rvert + \mathsf{OH}_{\mathsf{ae}}$	✓
with [11]	DCR & DDH	$(2d+1)\lvert\mathbb{Z}_{N^s}\rvert + 3\lvert\mathbb{G}_{\mathsf{ddh}}\rvert + \mathsf{OH}_{\mathsf{ae}}$	✓

Figure 2: Comparison of KDM-CCA secure PKE schemes with respect to degree-d polynomial functions. We use the same notation as in Figure 1.

the resulting PKE schemes. On the other hand, our and Malkin et al.'s schemes can encrypt a whole secret key without any modification by setting $s \geq 3$. (We provide a more detailed explanation on the plaintext space of our scheme in Section 5.1.)

Organization. In Section 2, we give a technical overview behind our proposed PKE schemes. In Section 3, we review definitions of cryptographic primitives and assumptions. In Section 4, we introduce a new primitive that we call symmetric key encapsulation mechanism (SKEM) and provide concrete instantiations. In Section 5, we present our KDM-CCA secure PKE scheme with respect to affine functions, and in Section 6, we present our KDM-CCA secure PKE scheme with respect to polynomials. Finally, in Section 7, we give instantiation examples of KDM-CCA secure PKE schemes.

2 Technical Overview

We provide an overview of our construction. Our starting point is the construction of KDM-CPA secure PKE proposed by Malkin et al. [23]. Their scheme is highly efficient, but only KDM-CPA secure. Our basic idea is to construct KDM-CCA secure PKE by adopting a construction technique used in the recent work by Kitagawa and Tanaka [19] into Malkin et al.'s scheme.

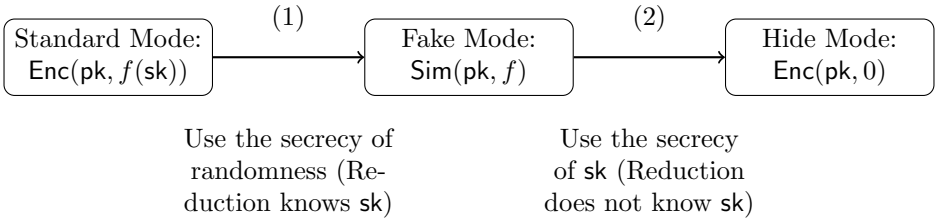

Figure 3: The triple mode proof. "XX Mode: YY" indicates that in XX Mode, the challenger returns YY as the answer to a KDM query from an adversary.

However, since a simple combination of them does not work, we introduce a new primitive that ties them together. We first review Malkin et al.'s scheme. Below, we explain the overview by focusing on constructing a PKE scheme that is $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure. The actual Malkin et al.'s scheme is $\mathcal{F}_{\mathsf{poly}}$-KDM-CPA secure, and we can construct a $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA secure scheme analogously.

2.1 KDM-CPA Secure Scheme by Malkin et al.

Malkin et al.'s scheme is secure under the DCR assumption and all procedures of their scheme are performed on $\mathbb{Z}^*_{N^s}$, where $N = PQ$ is an RSA modulus with safe primes P and Q of the same length, and $s \geq 2$ is an integer. Below, let $n = \frac{\phi(N)}{4}$. We can decompose $\mathbb{Z}^*_{N^s}$ as the internal direct product $G_{N^{s-1}} \otimes \langle -1 \rangle \otimes G_n \otimes G_2$, where $\langle -1 \rangle$ is the subgroup of $\mathbb{Z}^*_{N^s}$ generated by $-1 \bmod N^s$, and $G_{N^{s-1}}$, G_n, and G_2 are cyclic groups of order N^{s-1}, n, and 2, respectively. Note that $T := 1+N \in \mathbb{Z}^*_{N^s}$ has order N^{s-1} and it generates $G_{N^{s-1}}$. Moreover, we can efficiently compute discrete logarithms on $G_{N^{s-1}}$. In addition, we can generate a random generator of G_n.[1]

We can describe Malkin et al.'s scheme by using generators T and g of $G_{N^{s-1}}$ and G_n, respectively, and for simplicity we consider the single user setting for now. Below, all computations are done mod N^s unless stated otherwise, and we omit to write modN^s. When generating a key pair, we sample[2] a secret key as $x \xleftarrow{\mathsf{r}} \mathbb{Z}_n$ and compute a public key as $h = g^x$. When encrypting a message $m \in \mathbb{Z}_{N^{s-1}}$, we first sample $r \xleftarrow{\mathsf{r}} \mathbb{Z}_n$ and set a ciphertext as $(g^r, T^m \cdot h^r)$. If we have the secret key x, we can decrypt the ciphertext by computing the discrete logarithm of $(T^m \cdot h^r) \cdot (g^r)^{-x} = T^m$.

Triple mode proof framework. We say that a PKE scheme is KDM secure if an encryption of $f(\mathsf{sk})$ is indistinguishable from that of some constant message such as 0, where sk is a secret key and f is a function. Malkin et al. showed the $\mathcal{F}_{\mathsf{aff}}$-KDM-CPA security of their scheme based on the DCR assumption via the proof strategy that they call the *triple mode proof*.

In the triple mode proof framework, we prove KDM security using three main hybrid games. We let f be a function queried by an adversary as a KDM query. In the first hybrid called Standard Mode, the challenger returns an encryption of $f(\mathsf{sk})$. In the second hybrid called Fake Mode, the challenger returns a simulated ciphertext from f and the public key corresponding to sk. In the final hybrid called Hide Mode, the challenger returns an encryption of 0. See Figure 3.

If we can prove that the behavior of the adversary does not change between Standard Mode and Hide Mode, we see that the scheme is KDM secure. However, it is difficult to prove it

[1] This is done by generating $\mu \xleftarrow{\mathsf{r}} \mathbb{Z}^*_{N^s}$ and setting $g := \mu^{2N^{s-1}} \bmod N^s$. Then, g is a generator of G_n with overwhelming probability.

[2] In the actual scheme, we sample a secret key from $[\frac{N-1}{4}]$. We ignore this issue in this overview.

directly by relying on the secrecy of the secret key. This is because a reduction algorithm needs the secret key to simulate answers to KDM queries in Standard Mode. Then, we consider the intermediate hybrid, Fake Mode, and we try to prove the indistinguishability between Standard Mode and Fake Mode based on the secrecy of encryption randomness. We call this part Step (1). If we can do that, by showing the indistinguishability between Fake Mode and Hide Mode based on the secrecy of the secret key, we can complete the proof. We call this part Step (2). Note that a reduction for Step (2) does not need the secret key to simulate answers to KDM queries.

Using this framework, we can prove the KDM-CPA security of Malkin et al.'s scheme as follows. Let $f(x) = ax + b \bmod N^{s-1}$ be an affine function queried by an adversary, where $a, b \in \mathbb{Z}_{N^{s-1}}$. In Standard Mode, the adversary is given $(g^r, T^{ax+b} \cdot h^r)$. In Fake Mode, the adversary is given $(T^{-a} \cdot g^r, T^b \cdot h^r)$. We can prove the indistinguishability of these two hybrids using the indistinguishability of g^r and $T^{-a} \cdot g^r$. Namely, we use the DCR assumption and the secrecy of encryption randomness r in this step. Then, in Hide Mode, the adversary is given (g^r, h^r) that is an encryption of 0. We can prove the indistinguishability between Fake Mode and Hide Mode based on the interactive vector (IV) lemma [6] that is in turn based on the DCR assumption. The IV lemma says that for every constant $c_1, c_2 \in \mathbb{Z}_{N^{s-1}}$, $(T^{c_1} \cdot g^r, T^{c_2} \cdot h^r)$ is indistinguishable from (g^r, h^r) if in addition to r, x satisfying $h = g^x$ is hidden from the view of an adversary. This completes the proof of Malkin et al.'s scheme.

2.2 Problem When Proving KDM-CCA Security

Malkin et al.'s scheme is malleable thus is not KDM-CCA secure. In terms of the proof, Step (2) of the triple mode proof does not go through when considering KDM-CCA security. In Step (2), a reduction does not know the secret key and thus the reduction cannot simulate answers to decryption queries correctly.

On the other hand, we see that Step (1) of the triple mode proof goes through also when proving KDM-CCA security since a reduction algorithm knows the secret key in this step. Thus, to construct a KDM-CCA secure scheme based on Malkin et al's scheme, all we need is a mechanism that enables us to complete Step (2) of the triple mode proof.

2.3 The Technique by Kitagawa and Tanaka

To solve the above problem, we adopt the technique used by Kitagawa and Tanaka [19]. They constructed a KDM-CCA secure PKE scheme Π_{kdm} by combining projective hash functions PHF and PHF' and an IND-CCA secure PKE scheme Π_{cca}. Their construction is a double layered construction. Namely, when encrypting a message by their scheme, we first encrypt the message by the inner scheme constructed from PHF and PHF', and then encrypt the ciphertext again by Π_{cca}. The inner scheme is the same as the IND-CCA secure PKE scheme based on projective hash functions proposed by Cramer and Shoup [9] except that PHF used to mask a message is required to be *homomorphic* and on the other hand PHF' is required to be only universal (not 2-universal).

The security proof for this scheme can be captured by the triple mode proof framework. We first perform Step (1) of the triple mode proof based on the homomorphism of PHF and the hardness of a subset membership problem on the group behind projective hash functions. Then, we perform Step (2) of the triple mode proof using the IND-CCA security of Π_{cca}. In this step, a reduction algorithm can simulate answers to decryption queries. This is because the reduction algorithm can generate secret keys for PHF and PHF' by itself and access to the decryption oracle for Π_{cca}. When proving the CCA security of a PKE scheme based on projective hash functions, at some step in the proof, we need to estimate the probability that an adversary

makes an "illegal" decryption query. In the proof of the scheme by Kitagawa and Tanaka, this estimation can be done in Hide Mode of the triple mode proof. Due to this, the underlying PHF' needs to be only universal.

If the secret key csk of Π_{cca} is included as a part of the secret key of Π_{kdm}, to complete the proof, we need to change the security game so that csk is not needed to simulate answers to KDM queries in Step (1). It seems difficult unless we require an additional property for secret keys of Π_{cca} such as homomorphism. Instead, Kitagawa and Tanaka designed their scheme so that csk is included in the public key of Π_{kdm} after encrypting it by PHF. Then, by eliminating this encrypted csk from an adversary's view by using the security of PHF before Step (2) of the triple mode proof, the entire proof goes through. Note that, similarly to the proof for the construction by Cramer and Shoup [9], a reduction algorithm attacking the security of PHF can simulate answers to decryption queries due to the fact that the security property of PHF is statistical and an adversary for Π_{kdm} is required to make a proof that the query is "legal" using PHF'.

2.4 Adopting the Technique by Kitagawa and Tanaka

We now consider adopting the technique by Kitagawa and Tanaka into Malkin et al.'s scheme. Namely, we add a projective hash function for proving that an inner layer ciphertext of Malkin et al.'s scheme is well-formed, and also add an IND-CCA secure PKE scheme Π_{cca} as the outer layer. In order to prove the KDM-CCA security of this construction, we need to make the secret key csk of Π_{cca} as part of the public key of the resulting scheme after encrypting it somehow. Moreover, we have to eliminate this encrypted csk before Step (2) of the triple mode proof. However, this is not straightforward.

One naive way to do this is encrypting csk by the inner scheme based on the DCR assumption, but this idea does not work. Since the security of the inner scheme is computational unlike a projective hash function, a reduction algorithm attacking the inner scheme cannot simulate answers to decryption queries. One might think the problem is solved by modifying the scheme so that the security property of the inner scheme becomes statistical as a projective hash function, but this modification causes another problem. In order to do this, similarly to the DCR-based projective hash function by Cramer and Shoup [9], a secret key of the inner scheme needs to be sampled from a space whose size is as large as the order of $G_{N^{s-1}} \otimes G_n$ (that is, $N^{s-1} \cdot n$). However, the message space of this scheme is $\mathbb{Z}_{N^{s-1}}$, and thus we cannot encrypt such a large secret key by this scheme. The problem is more complicated when considering KDM-CCA security in the multi-user setting. Therefore, we need another solution to hide the secret key csk of Π_{cca}.

2.5 Solution: Symmetric Key Encapsulation Mechanism (SKEM)

To solve the above problem, we introduce a new primitive we call symmetric key encapsulation mechanism (SKEM). It is a key encapsulation mechanism in which we can use the same key for both the encapsulation algorithm Encap and decapsulation algorithm Decap. Moreover, it satisfies the following properties.

Encap can take an arbitrary integer $x \in \mathbb{Z}$ as an input secret key, but its computation is done by $x \bmod z$, where z is an integer determined in the setup. Then, for correctness, we require $\mathsf{Decap}(x \bmod z, \mathsf{ct}) = \mathsf{K}$, where $(\mathsf{ct}, \mathsf{K}) \leftarrow \mathsf{Encap}(x)$. Moreover, for security, the pseudorandomness of the session-time key K is required to hold as long as $x \bmod z$ is hidden from an adversary even if any other information of x is revealed.

Using SKEM $(\mathsf{Encap}, \mathsf{Decap})$ in addition to an IND-CCA secure PKE scheme Π_{cca} and a projective hash function PHF, we can construct a KDM-CCA secure PKE scheme based on

Malkin et al.'s scheme as follows. When generating a key pair, we first sample $x \xleftarrow{r} [n \cdot z]$ and compute $h \leftarrow g^x$, where z is an integer that is co-prime to n and satisfies $n \cdot z \leq N^{s-1}$. Then, we generate a key pair $(\mathsf{ppk}, \mathsf{psk})$ of PHF and $(\mathsf{cpk}, \mathsf{csk})$ of Π_{cca}, and $(\mathsf{ct}, \mathsf{K}) \leftarrow \mathsf{Encap}(x)$, and encrypt psk and csk to $\mathsf{ct}_{\mathsf{sk}}$ using the one-time key K. The resulting secret key is just x and public key is h, psk, cpk, and $(\mathsf{ct}, \mathsf{ct}_{\mathsf{sk}})$.[3] When encrypting a message m, we encrypt it in the same way as the Malkin et al.'s scheme and prove that those ciphertext components are included in G_n by using PHF. Then, we encrypt them by Π_{cca}. When decrypting the ciphertext, we first retrieve csk and psk from $(\mathsf{ct}, \mathsf{ct}_{\mathsf{sk}})$ and x using Decap, and decrypt the ciphertext using x, psk, and csk.

We can prove the $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA security of this scheme basically based on the triple mode proof framework. By doing the same process as Step (1) of the triple mode proof for Malkin et al.'s scheme, we can change the security game so that we can simulate answers to KDM queries using only x mod n. Moreover, due to the use of the projective hash function PHF, we can change the security game so that we can reply to decryption queries using only x mod n. Therefore, at this point, we do not need x mod z to simulate the security game, and thus we can use the security of the SKEM. We now delete csk and psk from $\mathsf{ct}_{\mathsf{sk}}$ using the security of the SKEM. Then, by using the security of Π_{cca}, we can accomplish Step (2) of the triple mode proof. Note that, similarly to the proof by Kitagawa and Tanaka [19], we estimate the probability that an adversary makes an "illegal" decryption query after Step (2) using the security of PHF.

2.6 Extension to the Multi-user Setting Using RKA Secure SKEM

The above overview of the proof considers KDM-CCA security in the single user setting. We can extend it to the multi-user setting. When considering KDM-CCA security in the multi-user setting, we modify the scheme so that we sample a secret key x from $[n \cdot z \cdot 2^{\xi}]$ such that $n \cdot z \cdot 2^{\xi} \leq N^{s-1}$. In the security proof, we sample a single x from $[n \cdot z]$ and generate the secret key x_i of the i-th user by sampling $\Delta_i \xleftarrow{r} [n \cdot z \cdot 2^{\xi}]$ and setting $x_i = x + \Delta_i$, where the addition is done over $\mathbb{Z}$. In this case, an affine function f of $x_1 \ldots, x_\ell$ is also an affine function of only x whose coefficients are determined by those of f and $\Delta_1, \ldots, \Delta_\ell$. Moreover, the statistical distance between a secret key generated in this way and that generated honestly is at most $2^{-\xi}$. Then, we can proceed the security proof in the same way as above, except for the part using the security of the SKEM.

The secret key x_i of the i-th user is now generated as $x + \Delta_i$ by using a single source x. Thus, each user's one-time key K_i used to hide the user's $(\mathsf{psk}, \mathsf{csk})$ is derived from a single source x and a "shift" value Δ_i. Standard security notations do not capture such a situation.

To address this problem, we require a *security property against related key attacks (RKA security)* for SKEM. However, a very weak form of RKA security is sufficient to complete the proof. We show that such an RKA secure SKEM can be constructed based only on the DCR assumption. Therefore, we can prove the KDM-CCA security in the multi-user setting of our scheme based only on the DCR assumption and the IND-CCA security of the underlying PKE scheme.

2.7 Differences in Usage of RKA Secure Primitive with Han et al.

We note that the previous most efficient KDM-CCA secure PKE schemes of Han et al. [12] (and the scheme of Lu et al. [22] on which the constructions of [12] are based), also use a "symmetric key" primitive that is "RKA secure". Specifically, Han et al. use a primitive called *authenticated encryption with auxiliary-input* (AIAE, for short), for which they define confidentiality and

[3] In the actual construction, we derive key pairs $(\mathsf{csk}, \mathsf{cpk})$ and $(\mathsf{ppk}, \mathsf{psk})$ using K as a random coin. This modification reduces the size of a public key.

integrity properties both under some appropriate forms of affine-RKA. Here, we highlight the differences between our proposed schemes and the schemes by Han et al. regarding the usage of a symmetric primitive with RKA security.

In our schemes, an RKA secure SKEM is used to derive the secret keys $(\mathsf{psk}, \mathsf{csk})$ of the underlying projective hash function and IND-CCA secure PKE scheme, and an SKEM ciphertext is put as part of a public key of the resulting scheme. In a modified security game considered in our security proofs, a KDM-CCA adversary sees multiple SKEM ciphertexts $\{\mathsf{ct}_i\}$ (contained in the public keys initially given to the adversary), where each ct_i is computed by using $x+\Delta_i \bmod z$ as a secret key, where $\Delta_i \in [n \cdot z \cdot 2^{\xi}]$ is chosen uniformly at random. Consequently, an SKEM used as a building block in our proposed schemes needs to be secure only against "passive" addition-RKA, in which the shift values $\{\Delta_i\}$ are chosen randomly by the challenger (rather than by an RKA adversary). Such an SKEM is easy to construct, and we will show several simple and efficient instantiations based on the DCR assumption, the DDH assumption, and hash functions with some appropriate form of "correlation-robustness" [18, 2].

On the contrary, in the Han et al.'s schemes, an AIAE ciphertext is directly contained as part of a ciphertext of the resulting scheme, and thus AIAE ciphertexts are exposed to a CCA. This is a main reason of the necessity of the integrity property for AIAE. Furthermore, in a modified security game considered in the security proofs of their schemes, a KDM-CCA adversary is able to observe multiple AIAE ciphertexts that are computed under secret keys that are derived via (some restricted from of) an affine function of a single (four-dimensional) vector of elements in $\mathbb{Z}_N$ through affine/poly-KDM queries, and thus their AIAE scheme needs to be secure under standard "active" affine-RKA (where key derivation functions are chosen by an RKA adversary, rather than the challenger). Han et al.'s instantiation of AIAE is essentially the Kurosawa-Desmedt encryption scheme [20] used as a symmetric encryption scheme, which is why they require the DDH assumption in addition to the DCR assumption.

2.8 Tightness of Our Construction

Our construction can be tightly instantiated by using a tightly IND-CCA secure PKE scheme as a building block. In our security proof, we can accomplish Step (1) of the triple mode proof by applying the DCR assumption only once via the IV lemma [6]. In Step (2), we need only a single application of the IND-CCA security of the outer scheme by requiring IND-CCA security in the multi-challenge multi-user setting. Thus, if the underlying IND-CCA secure scheme satisfies tight security in the setting, this step is also tight. In the estimation of the probability of "illegal" decryption queries, we only use a statistical property, and thus we do not lose any factor to the underlying assumption. The remaining part of our proof is eliminating secret keys of projective hash function and IND-CCA secure PKE encrypted by SKEM from an adversary's view. To make the entire proof tight, we have to accomplish this step tightly.

To achieve this, we show the RKA security of our SKEM can be tightly reduced to the underlying assumptions. Especially, in the proof of the DCR based construction, we show this using the IV lemma that is different from that we use in Step (1) of the triple mode proof. Namely, in this work, we use two flavors of the IV lemmas to make the security proof for the DCR-based instantiation tight.

To the best of our knowledge, prior to our work, the only way to construct tightly KDM-CCA secure PKE is instantiating the construction proposed by Camenisch et al. [7] using a tightly secure NIZK proof system such as that proposed by Hofheinz and Jager [15]. Schemes instantiated in such a way are not so practical and a ciphertext of them consists of $O(\lambda)$ group elements, where λ is the security parameter. We observe that the DDH-based construction of Kitagawa and Tanaka [19] can be tightly instantiated by using a tightly IND-CCA secure PKE

scheme as a building block, though they did not state that explicitly. However, its ciphertext also consists of $O(\lambda)$ group elements. Thus, our schemes are the first tightly KDM-CCA secure PKE scheme whose ciphertext consists of a constant number of group elements.

3 Preliminaries

In this section, we define some notations and cryptographic primitives.

3.1 Notations

In this paper, $x \xleftarrow{\mathsf{r}} X$ denotes choosing an element from a finite set X uniformly at random, and $y \leftarrow \mathsf{A}(x)$ denotes assigning to y the output of an algorithm A on an input x. For an integer $\ell > 0$, $[\ell]$ denote the set of integers $\{1, \ldots, \ell\}$. For a function f, $\mathsf{Sup}(f)$ denotes the support of f. For a finite set S, $|S|$ denotes its cardinality, and U_S denotes the uniform distribution over S.

λ denotes a security parameter. PPT stands for probabilistic polynomial time. A function $f(\lambda)$ is a negligible function if $f(\lambda)$ tends to 0 faster than $\frac{1}{\lambda^c}$ for every constant $c > 0$. We write $f(\lambda) = \mathsf{negl}(\lambda)$ to denote $f(\lambda)$ being a negligible function.

Let X be a distribution over a set S. The *min-entropy* of X, denoted by $\mathbf{H}_\infty(X)$, is defined by

$$\mathbf{H}_\infty(X) := -\log_2 \max_{z \in S} \Pr[X = z] \ .$$

Let X and Y be distributions over a set S. The *statistical distance* between X and Y, denoted by $\mathbf{SD}(X, Y)$, is defined by

$$\mathbf{SD}(X, Y) := \frac{1}{2} \sum_{z \in S} |\Pr[X = z] - \Pr[Y = z]| \ .$$

We say that X and Y are ϵ-close if $\mathbf{SD}(X, Y) \leq \epsilon$.

3.2 Leftover Hash Lemma

Here, we recall the leftover hash lemma. Recall that a hash family $\mathcal{H} = \{H : D \to R\}$ is said to be *universal* if for all distinct $x, x' \in D$, it holds that $\Pr_{H \xleftarrow{\mathsf{r}} \mathcal{H}}[H(x) = H(x')] \leq \frac{1}{|R|}$.

Lemma 1 (Leftover hash lemma) *Let $\mathcal{H} := \{H : D \to R\}$ be a universal hash family, and let X be a distribution over D. Then,*

$$\mathbf{SD}\left((H, H(X)), (H, \mathsf{U}_R)\right) \leq \frac{1}{2} \cdot \sqrt{2^{-\mathbf{H}_\infty(X)} \cdot |R|} \ ,$$

where $H \xleftarrow{\mathsf{r}} \mathcal{H}$.

3.3 Assumptions

We review the algebraic structure and assumptions used in this paper.

Let $N = PQ$ be an RSA modulus with len-bit safe primes $P = 2p+1$ and $Q = 2q+1$ where p and q are also primes. Let $n = pq$. Throughout the paper, we assume $\mathsf{len} \geq \lambda$, and we will frequently use the fact that $\mathbf{SD}(\mathsf{U}_{[n]}, \mathsf{U}_{\left[\frac{N-1}{4}\right]}) = \frac{P+Q-2}{N-1} = O(2^{-\mathsf{len}})$.

Let $s \geq 2$ be an integer and $T := 1 + N$. We can decompose $\mathbb{Z}_{N^s}^*$ as the internal direct product $G_{N^{s-1}} \otimes \langle -1 \rangle \otimes G_n \otimes G_2$, where $\langle -1 \rangle$ is the subgroup of $\mathbb{Z}_{N^s}^*$ generated by $-1 \bmod N^s$,

and $G_{N^{s-1}}$, G_n, and G_2 are cyclic groups of order N^{s-1}, n, and 2, respectively. Note that $T = 1 + N \in \mathbb{Z}_{N^s}^*$ has order N^{s-1} and it generates $G_{N^{s-1}}$. In addition, we can generate a random generator of G_n by generating $\mu \xleftarrow{\mathsf{r}} \mathbb{Z}_{N^s}^*$ and setting $g := \mu^{2N^{s-1}} \mod N^s$. Then, g is a generator of G_n with overwhelming probability. We also note that the discrete logarithm (base T) is easy to compute in $G_{N^{s-1}}$.

Let $\mathbb{QR}_{N^s} := \left\{x^2 \middle| x \in \mathbb{Z}_{N^s}^*\right\}$. Then, we have $\mathbb{QR}_{N^s} = G_{N^{s-1}} \otimes G_n$. We denote $\langle -1 \rangle \otimes \mathbb{QR}_{N^s}$ by $\mathbb{J}_{N^s}$. We can efficiently check the membership of $\mathbb{J}_{N^s}$ by computing the Jacobi symbol with respect to N, without P and Q.

Let GGen be an algorithm, which we call the DCR group generator, that given 1^λ and an integer $s \geq 2$, outputs $\mathsf{param} = (N, P, Q, T, g)$, where N, P, Q, and T are defined as above, and g is a random generator of G_n.

We adopt the definition of the DCR assumption [25, 10] used by Hofheinz [13].

Definition 1 (DCR assumption) *We say that the DCR assumption holds with respect to* GGen *if for any integer* $s \geq 2$ *and PPT adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{A}}(\lambda) = \left|\Pr[\mathcal{A}(N, g, g^r \bmod N^s) = 1] - \Pr[\mathcal{A}(N, g, T \cdot g^r \bmod N^s) = 1]\right| = \mathsf{negl}(\lambda) \ ,$$

where $(N, P, Q, T, g) \leftarrow \mathsf{GGen}\left(1^\lambda, s\right)$ *and* $r \xleftarrow{\mathsf{r}} [n]$.

We recall the *interactive vector game* introduced by Brakerski and Goldwasser [6].

Definition 2 (Interactive vector game) *Let* $s \geq 2$ *be an integer and* ℓ *be a polynomial of* λ*. We define the following* $\mathsf{IV}_{s,\ell}$ *game between a challenger and an adversary* $\mathcal{A}$*.*

1. *The challenger chooses a challenge bit* $b \xleftarrow{\mathsf{r}} \{0,1\}$ *and generates* $(N, P, Q, T, g) \leftarrow \mathsf{GGen}\left(1^\lambda, s\right)$*. If* $\ell = 1$*, the challenger sends* N *and* $g_1 := g$ *to* $\mathcal{A}$*. Otherwise, the challenger generates* $\alpha_i \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ *and computes* $g_i \leftarrow g^{\alpha_i} \bmod N^s$ *for every* $i \in [\ell]$*, and sends* N*,* g*, and* $g_1, \ldots, g_\ell$ *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *can adaptively make sample queries.*

   Sample queries $\mathcal{A}$ *sends* $(a_1, \ldots, a_\ell) \in \mathbb{Z}_{N^{s-1}}^\ell$ *to the challenger. The challenger generates* $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ *and computes* $e_i \leftarrow T^{b \cdot a_i} \cdot g_i^r \bmod N^s$ *for every* $i \in [\ell]$*. The challenger then returns* $(e_1, \ldots, e_\ell)$ *to* $\mathcal{A}$*.*
3. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$*.*

We say that $\mathsf{IV}_{s,\ell}$ *is hard if for any PPT adversary* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{IV}}_{s,\ell,\mathcal{A}}(\lambda) = 2 \cdot \left|\Pr[b = b'] - \frac{1}{2}\right|$ $= \mathsf{negl}(\lambda)$.

For any s and ℓ, $\mathsf{IV}_{s,\ell}$ is hard under the DCR assumption [6, 23]. We show the following lemmas related to $\mathsf{IV}_{s,\ell}$ that are useful to prove the tight security of our constructions.

Lemma 2 *Let* $s \geq 2$ *be an integer. Let* $\mathcal{A}$ *be a PPT adversary that plays the* $\mathsf{IV}_{s,1}$ *game and makes at most* q_{iv} *queries. Then, there exists a PPT adversary* $\mathcal{B}$ *satisfying*

$$\mathsf{Adv}^{\mathsf{iv}}_{s,1,\mathcal{A}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{O(q_{\mathsf{iv}})}{2^{\mathsf{len}}} \ .$$

Lemma 3 *Let* $s \geq 2$ *be an integer. Let* ℓ *be a polynomial of* λ*. Let* $\mathcal{A}$ *be a PPT adversary that plays the* $\mathsf{IV}_{s,\ell}$ *game and makes exactly one sample query. Then, there exists a PPT adversary* $\mathcal{B}$ *satisfying*

$$\mathsf{Adv}^{\mathsf{iv}}_{s,\ell,\mathcal{A}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{O(\ell)}{2^{\mathsf{len}}} \ .$$

The proofs of Lemmas 2 and 3 are given in Section B.

3.4 Projective Hash Function

We review the notion of *projective hash functions* (PHF) introduced by Cramer and Shoup [9] (which is also called *hash proof systems* in the literature). In this work, we will use PHFs defined with respect to the DCR group generator GGen.

Definition 3 (Projective hash function family) *A PHF family* PHF *with respect to* GGen *consists of a tuple* $(\mathsf{Setup}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ *with the following properties:*

- Setup *is a PPT algorithm that takes* $\mathsf{param} = (N, P, Q, T, g)$ *output by* $\mathsf{GGen}(1^\lambda, s)$ *(for some* $s \geq 2$*) as input, and outputs a public parameter* pp *that parameterizes the remaining components of* PHF*. (In the following, we always make the existence of* pp *implicit and suppress it from the notation.)*

- Π_{yes}, Π_{no}, $\mathcal{SK}$, $\mathcal{PK}$, *and* $\mathcal{K}$ *are sets parameterized by* pp *(and also by* param*).* Π_{yes} *and* Π_{no} *form an NP-language,*[4] *where for all* $c \in \Pi_{\mathsf{yes}}$*, there exists a witness* r *with which one can efficiently check the fact of* $c \in \Pi_{\mathsf{yes}}$*. An element in* Π_{yes} *(resp.* Π_{no}*) is called an yes (resp. no) instance.*

  Furthermore, it is required that given pp*, one can efficiently sample a uniformly random element from* $\mathcal{SK}$*.*

- Λ *is an efficiently computable (deterministic) hash function that takes a secret key* $\mathsf{sk} \in \mathcal{SK}$ *and an yes or no instance* $c \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$ *as input, and outputs a hash value* $\pi \in \mathcal{K}$*.*

- μ *is an efficiently computable (deterministic) projection map that takes a secret key* $\mathsf{sk} \in \mathcal{SK}$ *as input, and outputs a public key* $\mathsf{pk} \in \mathcal{PK}$*.*

- Pub *is an efficiently computable algorithm that takes a public key* $\mathsf{pk} \in \mathcal{PK}$*, an yes instance* $c \in \Pi_{\mathsf{yes}}$*, and a witness* r *that* $c \in \Pi_{\mathsf{yes}}$ *as input, and outputs a hash value* $\pi \in \mathcal{K}$*.*

- *Projective property: For all* $\mathsf{sk} \in \mathcal{SK}$*, the action of* $\Lambda_{\mathsf{sk}}(\cdot)$ *for yes instances* $c \in \Pi_{\mathsf{yes}}$ *is completely determined by* $\mathsf{pk} = \mu(\mathsf{sk})$*. Furthermore, for all* $c \in \Pi_{\mathsf{yes}}$ *and a corresponding witness* r*, it holds that* $\Lambda_{\mathsf{sk}}(c) = \mathsf{Pub}(\mu(\mathsf{sk}), c, r)$*.*

We next introduce the universal property for a PHF family. In this paper, we consider the statistical and computational variants. Our definition of the computational universal property is based on the "computational universal2" property for a hash proof system introduced by Hofheinz and Kiltz [16]. We adapt their definition to the "universal1" case, and also relax the notion so that we only require that guessing a hash value for a no instance is hard, rather than requiring that a hash value of a no instance is pseudorandom.

Definition 4 (Statistical/computational universal) *Let* $s \geq 2$*,* GGen *be the DCR group generator, and* $\mathsf{PHF} = (\mathsf{Setup}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ *be a PHF family with respect to* GGen*. We say that* PHF *is*

- ϵ-universal *if for any* param *output by* $\mathsf{GGen}(1^\lambda, s)$*, any* pp *output by* $\mathsf{Setup}(\mathsf{param})$*, any* $\mathsf{pk} \in \mathcal{PK}$*, any* $c \in \Pi_{\mathsf{no}}$*, and any* $\pi \in \mathcal{K}$*, we have*

$$\Pr_{\mathsf{sk} \leftarrow \mathcal{SK}}[\Lambda_{\mathsf{sk}}(c) = \pi | \mu(\mathsf{sk}) = \mathsf{pk}] \leq \epsilon \ . \tag{1}$$

[4] Strictly speaking, since Π_{yes} and Π_{no} may not cover the entire input space of the function $\Lambda_{\mathsf{sk}}(\cdot)$ introduced below, they form an NP-promise problem.

Furthermore, we simply say that PHF *is* universal *if it is ϵ-universal for some negligible function $\epsilon = \epsilon(\lambda)$.*

- computationally universal *if for any PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{A}}(\lambda)$ in the following game played by $\mathcal{A}$ and a challenger is negligible in λ:*

  1. *First, the challenger executes $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$ and $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{param})$. The challenger then chooses $\mathsf{sk} \xleftarrow{\mathsf{r}} \mathcal{SK}$, and computes $\mathsf{pk} \leftarrow \mu(\mathsf{sk})$. Then, the challenger sends $(N, T, g, \mathsf{pp}, \mathsf{pk})$ to $\mathcal{A}$.*
  2. *$\mathcal{A}$ can adaptively make evaluation queries.*

     Evaluation queries *$\mathcal{A}$ sends an yes or no instance $c \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$ to the challenger. If $c \in \Pi_{\mathsf{yes}}$, the challenger returns $\pi \leftarrow \Lambda_{\mathsf{sk}}(c)$ to $\mathcal{A}$. Otherwise (i.e. $c \in \Pi_{\mathsf{no}}$), the challenger returns $\perp$ to $\mathcal{A}$.*
  3. *$\mathcal{A}$ outputs a pair $(c^*, \pi^*) \in \Pi_{\mathsf{no}} \times \mathcal{K}$. The advantage of $\mathcal{A}$ is defined by $\mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{A}}(\lambda) := \Pr[\Lambda_{\mathsf{sk}}(c^*) = \pi^*]$.*

Remark 1 (Statistical implies computational) It is not hard to see that the (statistical) universal property implies the computational one (even against computationally unbounded adversaries). To see this, recall that the projective property ensures that the action of $\Lambda_{\mathsf{sk}}(\cdot)$ for yes instances is determined by pk. Thus, the evaluation results $\Lambda_{\mathsf{sk}}(c)$ for yes instances $c \in \Pi_{\mathsf{yes}}$ do not reveal the information of sk beyond the fact that $\mathsf{pk} = \mu(\mathsf{sk})$. Also, evaluation queries with no instances $c \in \Pi_{\mathsf{no}}$ are answered with $\perp$. These imply that throughout the game, the information of sk does not leak to an adversary beyond what is already leaked from pk. Thus, at the point of outputting (c^*, π^*), sk is uniformly distributed over the subset $\mathcal{SK}|_{\mathsf{pk}} := \{\mathsf{sk}' \in \mathcal{SK} | \mu(\mathsf{sk}') = \mathsf{pk}\}$ from an adversary's viewpoint, which is exactly the distribution of sk in the probability defining the universal property. Hence, if a PHF family is ϵ-universal, the probability that $\Lambda_{\mathsf{sk}}(c^*) = \pi^*$ occurs is upper bounded by ϵ.

3.5 Public Key Encryption

Here, we review the definitions for public key encryption (PKE).

Definition 5 (Public key encryption) *A PKE scheme* PKE *is a four tuple* (Setup, KG, Enc, Dec) *of PPT algorithms. Below, let $\mathcal{M}$ be the message space of* PKE.

- *The setup algorithm* Setup*, given a security parameter 1^λ, outputs a public parameter* pp*. For simplicity, we omit* pp *from inputs for* Enc *and* Dec.
- *The key generation algorithm* KG*, given a public parameter* pp*, outputs a public key* pk *and a secret key* sk.
- *The encryption algorithm* Enc*, given a public key* pk *and a message $m \in \mathcal{M}$, outputs a ciphertext* CT.
- *The decryption algorithm* Dec*, given a public key* pk*, a secret key* sk*, and a ciphertext c, outputs a message $\tilde{m} \in \{\perp\} \cup \mathcal{M}$.*

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

Next, we define key dependent message security against chosen ciphertext attacks (KDM-CCA security) for PKE.

Definition 6 (KDM-CCA security) *Let* PKE *be a PKE scheme,* $\mathcal{F}$ *function family, and* ℓ *the number of keys. We define the* $\mathcal{F}$*-KDM-CCA game between a challenger and an adversary* $\mathcal{A}$ *as follows. Let* $\mathcal{SK}$ *and* $\mathcal{M}$ *be the secret key space and message space of* PKE*, respectively.*

1. *First, the challenger chooses a challenge bit* $b \xleftarrow{\mathsf{r}} \{0,1\}$ *and generates* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*. Next, the challenger generates* ℓ *key pairs* $(\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KG}(\mathsf{pp})\,(k \in [\ell])$*. The challenger sets* $\mathbf{sk} := (\mathsf{sk}_1, \ldots, \mathsf{sk}_\ell)$ *and sends* $(\mathsf{pk}_1, \ldots, \mathsf{pk}_\ell)$ *to* $\mathcal{A}$*. Finally, the challenger prepares a list* L_{kdm} *which is initially empty.*

2. $\mathcal{A}$ *may adaptively make the following queries polynomially many times.*

   KDM queries $\mathcal{A}$ *sends* $\left(j, f^0, f^1\right) \in [\ell] \times \mathcal{F} \times \mathcal{F}$ *to the challenger. We require that* f^0 *and* f^1 *be functions such that* $f : \mathcal{SK}^\ell \to \mathcal{M}$*. The challenger returns* $\mathsf{CT} \leftarrow \mathsf{Enc}\left(\mathsf{pk}_j, f^b(\mathbf{sk})\right)$ *to* $\mathcal{A}$*. Finally, the challenger adds* (j, CT) *to* L_{kdm}*.*

   Decryption queries $\mathcal{A}$ *sends* (j, CT) *to the challenger. If* $(j, \mathsf{CT}) \in L_{\mathsf{kdm}}$*, the challenger returns* $\perp$ *to* $\mathcal{A}$*. Otherwise, the challenger returns* $m \leftarrow \mathsf{Dec}\left(\mathsf{pk}_j, \mathsf{sk}_j, \mathsf{CT}\right)$ *to* $\mathcal{A}$*.*

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

We say that PKE *is* $\mathcal{F}$*-KDM-CCA secure if for any polynomial* $\ell = \ell(\lambda)$ *and PPT adversary* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{kdmcca}}_{\mathsf{PKE},\mathcal{F},\ell,\mathcal{A}}(\lambda) = 2 \cdot \left|\Pr[b = b'] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$*.*

The above definition is slightly different from the standard definition where an adversary is required to distinguish encryptions of $f(\mathsf{sk}_1, \ldots, \mathsf{sk}_\ell)$ from encryptions of some fixed message. However, the two definitions are equivalent if the function class $\mathcal{F}$ contains a constant function, and this is the case for affine functions and polynomials treated in this paper.

The definition of IND-CCA security (in the multi-user/challenge setting) is recovered by restricting the functions used in KDM queries in the KDM-CCA game to constant functions, and thus we omit the description of the security game for it. We denote an adversary $\mathcal{A}$'s IND-CCA advantage by $\mathsf{Adv}^{\mathsf{indcca}}_{\mathsf{PKE},\ell,\mathcal{A}}(\lambda)$.

4 Symmetric KEM and Passive RKA Security

In our proposed PKE schemes, we will use a secret key variant of a key encapsulation mechanism (KEM) satisfying a weak form of RKA security with respect to addition, as one of the main building blocks. Since several instantiations for this building block from various assumptions are possible, in this section we formalize it as a stand-alone primitive called *symmetric KEM (SKEM)*, together with its RKA security in the form we use in the security proofs of the proposed PKE schemes. After giving the definitions in Section 4.1, we give concrete instantiations in Section 4.2 (and in Section C).

4.1 Definition

We first give the formal syntax and functional requirements of an SKEM, and then give some remarks.

Definition 7 (Symmetric key encapsulation mechanism) *An SKEM* SKEM *is a three tuple* $(\mathsf{Setup}, \mathsf{Encap}, \mathsf{Decap})$ *of PPT algorithms.*

- *The setup algorithm* Setup*, given a security parameter* 1^λ*, outputs a public parameter* pp *and a pair of natural numbers* $(z, \widetilde{z})$*, where* z *represents the size of the secret key space, and the secret key space is* $[z]$*, and* $\widetilde{z}$ *is an approximation of* z*. We assume that* $\widetilde{z}$ *(but not necessarily* z*) can be efficiently derived from* pp*. We also assume that* pp *specifies the session-key space* $\mathcal{K}$*.*

- *The encapsulation algorithm* Encap*, given a public parameter* pp *and a secret key* $\mathsf{sk} \in \mathbb{Z}$*, outputs a ciphertext* ct *and a session-key* $\mathsf{K} \in \mathcal{K}$*.*

- *The decapsulation algorithm* Decap*, given a public parameter* pp*, a secret key* $\mathsf{sk} \in \mathbb{Z}$*, and a ciphertext* ct*, outputs a session-key* $\mathsf{K} \in \mathcal{K}$*.*

As the functional (syntactical) requirements, we require the following three properties to hold for all $(\mathsf{pp}, z, \widetilde{z}) \leftarrow \mathsf{Setup}(1^\lambda)$*:*

1. *(Approximate samplability of secret keys:)* $\mathbf{SD}(\mathsf{U}_{[z]}, \mathsf{U}_{[\widetilde{z}]}) \leq O(2^{-\lambda})$ *holds.*
2. *(Correctness of decapsulation:)* $\mathsf{Decap}(\mathsf{pp}, \mathsf{sk} \bmod z, \mathsf{ct}) = \mathsf{K}$ *holds for every* $\mathsf{sk} \in \mathbb{Z}$ *and* $(\mathsf{ct}, \mathsf{K}) \leftarrow \mathsf{Encap}(\mathsf{pp}, \mathsf{sk})$*.*
3. *(Implicit modular-reduction in encapsulation:)* $\mathsf{Encap}(\mathsf{pp}, \mathsf{sk}; r) = \mathsf{Encap}(\mathsf{pp}, \mathsf{sk} \bmod z; r)$ *holds for every* $\mathsf{sk} \in \mathbb{Z}$ *and randomness* r *for* Encap*.*

Remark 2 (On the syntax and functional requirements)

- As mentioned above, when $(\mathsf{pp}, z, \widetilde{z})$ is output by $\mathsf{Setup}(1^\lambda)$, the secret key space under pp is $[z]$. For security reasons, however, in some constructions, the exact order z cannot be made public even for an entity executing Encap and Decap. (In particular, this is the case in our concrete instantiation from the DCR assumption, in which we set $z = \frac{\phi(N)}{4}$ and $\widetilde{z} = \frac{N-1}{4}$.) Hence, we instead require its approximation $\widetilde{z}$ to be public via pp.

- We allow Encap and Decap to take any integer $\mathsf{sk} \in \mathbb{Z}$ (rather than $\mathsf{sk} \in [z]$ or $\mathsf{sk} \in [\widetilde{z}]$) as a secret key, but their "correctness guarantees" expressed by the second and third items of the functional requirements, are with respect to the modular-reduced value $\mathsf{sk} \bmod z$. Such flexible interface is convenient when an SKEM is used as a building block in the proposed PKE schemes.

- The third item in the functional requirements ensures that a ciphertext/session-key pair $(\mathsf{ct}, \mathsf{K})$ generated by using $\mathsf{sk} \in \mathbb{Z}$ does not leak the information of sk beyond $\mathsf{sk} \bmod z$. This property plays an important role in the security proofs of our proposed PKE schemes.

- Note that an SKEM can satisfy our syntactical and functional requirements even if its ciphertext is empty. (Say, Encap and Decap output some deterministic function of pp and $\mathsf{sk} \bmod \widetilde{z}$.) In fact, our instantiation of an SKEM from a hash function is of this kind. See Section C.

In the following, we give the formalization of passive RKA security. It is essentially the definition of the same name defined for symmetric encryption by Applebaum, Harnik, and Ishai [2], with the slight difference that we allow an adversary to specify the upper bound B of the interval from which key-shifting values $\{\Delta_k\}$ are chosen randomly by the challenger.

Definition 8 (Passive RKA security) *Let* $\mathsf{SKEM} = (\mathsf{Setup}, \mathsf{Encap}, \mathsf{Decap})$ *be an SKEM, and let* ℓ *be a natural number. Consider the following game between a challenger and an adversary* $\mathcal{A}$*:*

1. *First, the challenger chooses a challenge bit* $b \xleftarrow{r} \{0,1\}$ *and generates* $(\mathsf{pp}, z, \widetilde{z}) \leftarrow \mathsf{Setup}(1^\lambda)$. *Then, the challenger sends* $\widetilde{z}$ *to* $\mathcal{A}$.

2. $\mathcal{A}$ *sends an integer* $B \geq \widetilde{z}$ *specifying the upper bound of the interval from which key-shifting values* $\{\Delta_k\}_{k\in[\ell]}$ *are chosen, to the challenger.*

3. *The challenger samples* $\mathsf{sk} \xleftarrow{r} [z]$ *and* $\Delta_k \xleftarrow{r} [B]$ *for every* $k \in [\ell]$. *Then, the challenger computes* $(\mathsf{ct}_k, \mathsf{K}_k^1) \leftarrow \mathsf{Encap}(\mathsf{pp}, \mathsf{sk} + \Delta_k)$[5] *and also samples* $\mathsf{K}_k^0 \leftarrow \mathcal{K}$ *for every* $k \in [\ell]$. *Finally, the challenger sends* pp, $(\Delta_k)_{k\in[\ell]}$, *and* $\left(\mathsf{ct}_k, \mathsf{K}_k^b\right)_{k\in[\ell]}$ *to* $\mathcal{A}$.

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

We say that SKEM *is* passively RKA secure, *if for any polynomial* $\ell = \ell(\lambda)$ *and PPT adversary* $\mathcal{A}$, *we have* $\mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{A}}(\lambda) = 2 \cdot \left|\Pr[b = b'] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$.

Remark 3 (Stretching a session-key with a pseudorandom generator) From the definition, it is easy to see that a session-key of an SKEM can be stretched by using a pseudorandom generator (PRG) while preserving its passive RKA security. More specifically, let $\mathsf{SKEM} = (\mathsf{Setup}, \mathsf{Encap}, \mathsf{Decap})$ be an SKEM with session-key space $\mathcal{K}$, and let $\mathsf{PRG} : \mathcal{K} \to \mathcal{K}'$ be a PRG such that $|\mathcal{K}| < |\mathcal{K}'|$. Let $\mathsf{SKEM}' = (\mathsf{Setup}, \mathsf{Encap}', \mathsf{Decap}')$ be the SKEM with session-key space $\mathcal{K}'$ that is obtained by naturally composing SKEM with PRG, namely, $\mathsf{Encap}'(\mathsf{pp}, \mathsf{sk})$ runs $(\mathsf{ct}, \mathsf{K}) \leftarrow \mathsf{Encap}(\mathsf{pp}, \mathsf{sk})$ and outputs $(\mathsf{ct}, \mathsf{PRG}(\mathsf{K}))$, and $\mathsf{Decap}'(\mathsf{pp}, \mathsf{sk}, \mathsf{ct}) := \mathsf{PRG}(\mathsf{Decap}(\mathsf{pp}, \mathsf{sk}, \mathsf{ct}))$. Then, if SKEM is passively RKA secure and PRG is a secure PRG, then SKEM' is also passively RKA secure. Moreover, if the passive RKA security of SKEM is tightly reduced to some assumption and the multi-instance version of the security of PRG is also tightly reduced to the same assumption, then so is the passive RKA security of SKEM'. (Since the proof is straightforward, we omit a formal proof of this simple fact.) Note that we can easily construct tightly secure PRG based on the DDH or DCR assumption.

4.2 Concrete Instantiations

Our definition of passive RKA security for an SKEM is sufficiently weak so that simple and efficient constructions are possible from the DCR or DDH assumption, which are essentially the symmetric-key version of the ElGamal KEM. We can also realize it from a hash function satisfying an appropriate form of "correlation robustness" [18, 2]. In this subsection, we give a concrete instantiation based on the DCR assumption. The other instantiations are given in Section C.

Let $s \geq 2$, GGen be the DCR group generator, and $\mathcal{H} = \left\{H : \{0,1\}^{2s\cdot\mathsf{len}} \to \mathcal{K}\right\}$ be a universal hash family. Then, we can construct an SKEM $\mathsf{SKEM} = (\mathsf{Setup}, \mathsf{Encap}, \mathsf{Decap})$ whose session-key space is $\mathcal{K}$, as described in Figure 4.[6]

It is obvious to see that SKEM satisfies the three functional requirements of SKEM. Specifically, let $(\mathsf{pp}, z, \widetilde{z})$ be output by Setup. Then, we have $\mathbf{SD}\left(\mathsf{U}_{[z]}, \mathsf{U}_{[\widetilde{z}]}\right) = \mathbf{SD}(\mathsf{U}_{\left[\frac{\phi(N')}{4}\right]}, \mathsf{U}_{\left[\frac{N'-1}{4}\right]}) = O(2^{-\mathsf{len}}) \leq O(2^{-\lambda})$. The other two properties of the functional requirements are also satisfied due to the fact that in Encap and Decap, a secret key is treated only in the exponent of elements in $G_{n'}$ (where $n' = (P'-1)(Q'-1)/4$, and $G_{n'}$ is the subgroup of $Z^*_{N's}$ of order n').

The passive RKA security of SKEM can be shown as follows.

[5] The addition $\mathsf{sk} + \Delta_k$ is done over $\mathbb{Z}$.

[6] Since the RSA modulus used in the SKEM has to be generated independently of that in the main constructions presented in Sections 5 and 6, here we use characters with a prime (e.g. N') for values in param.

$\mathsf{Setup}(1^\lambda)$:	$\mathsf{Encap}(\mathsf{pp}, \mathsf{sk} \in \mathbb{Z})$:	$\mathsf{Decap}(\mathsf{pp}, \mathsf{sk} \in \mathbb{Z}, \mathsf{ct})$:
$(N', P', Q', T', g') \leftarrow \mathsf{GGen}(1^\lambda, s)$	$(N', T', g', H) \leftarrow \mathsf{pp}$	$(N', T', g', H) \leftarrow \mathsf{pp}$
$H \xleftarrow{\mathsf{r}} \mathcal{H}$	$\alpha \xleftarrow{\mathsf{r}} [\frac{N'-1}{4}]$	$\mathsf{K} \leftarrow H(\mathsf{ct}^{\mathsf{sk}} \bmod N'^s)$
$\mathsf{pp} \leftarrow (N', T', g', H)$	$\mathsf{ct} \leftarrow g'^\alpha \bmod N'^s$	Return K.
Return $(\mathsf{pp}, z := \frac{\phi(N')}{4}, \widetilde{z} := \frac{N'-1}{4})$.	$\mathsf{K} \leftarrow H(\mathsf{ct}^{\mathsf{sk}} \bmod N'^s)$	
	Return $(\mathsf{ct}, \mathsf{K})$.	

Figure 4: The DCR-based instantiation of an SKEM.

Lemma 4 *If the DCR assumption holds with respect to* GGen*, and* $\epsilon_{\mathsf{LHL}} := \frac{1}{2} \cdot \sqrt{2^{-(s-1)\cdot(2\mathsf{len}-1)} \cdot |\mathcal{K}|}$ $= \mathsf{negl}(\lambda)$*, then* SKEM *is passively RKA secure.*

Specifically, for any polynomial $\ell = \ell(\lambda)$ *and PPT adversary* $\mathcal{A}$ *that attacks the passive RKA security of* SKEM*, there exists a PPT adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{A}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \ell \cdot \left(\epsilon_{\mathsf{LHL}} + O(2^{-\mathsf{len}})\right) \quad . \tag{2}$$

Proof of Lemma 4. Fix arbitrarily a polynomial $\ell = \ell(\lambda)$ and a PPT adversary $\mathcal{A}$. Consider the following sequence of games.

Game 0: This is the passive RKA security game in which $b = 1$. Note that in this game, sk is sampled as $\mathsf{sk} \xleftarrow{\mathsf{r}} [n']$, and each session-key K_k (corresponding to the k-th ciphertext ct_k) given to $\mathcal{A}$ is computed as

$$\mathsf{K}_k = H\left(\mathsf{ct}_k^{\mathsf{sk}+\Delta_k} \bmod N'^s\right) = H\left(g'^{\alpha_k(\mathsf{sk}+\Delta_i)} \bmod N'^s\right) \quad ,$$

where $\mathsf{pp} = (N', T', g', H)$ is a public parameter, α_k is the randomness used to generate the k-th ciphertext $\mathsf{ct}_k = g'^{\alpha_k} \bmod N'^s$, $\Delta_1, \ldots, \Delta_\ell$ are the key-shifting values chosen uniformly at random from $[B]$, and $B \geq \widetilde{z}$ is the bound of the interval chosen by $\mathcal{A}$.

Game 1: Same as Game 0, except that sk is sampled as $\mathsf{sk} \xleftarrow{\mathsf{r}} \left[\frac{N'-1}{4}\right]$.

Game 2: Same as Game 1, except that each K_k is computed by first choosing $\theta_k \xleftarrow{\mathsf{r}} [N'^{s-1}]$ and then computing

$$\mathsf{K}_k = H\left(T'^{\theta_k} \cdot \mathsf{ct}_k^{\mathsf{sk}+\Delta_k} \bmod N'^s\right) = H\left(T'^{\theta_k} \cdot g'^{\alpha_i(\mathsf{sk}+\Delta_k)} \bmod N'^s\right) \quad .$$

Game 3: Same as Game 2, except that sk is sampled as $\mathsf{sk} \xleftarrow{\mathsf{r}} [n']$.

Game 4: This is the passive RKA security game in which $b = 0$. Hence, each K_k is chosen uniformly at random from $\mathcal{K}$.

For $t \in \{0, \ldots, 4\}$, let $\mathtt{T}_t$ be the event that $\mathcal{A}$ outputs 1 in Game t. By the definition of the advantage and the triangle inequality, we have

$$\mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{A}}(\lambda) = |\Pr[\mathtt{T}_0] - \Pr[\mathtt{T}_4]| \leq \sum_{t \in \{0,1,2,3\}} |\Pr[\mathtt{T}_t] - \Pr[\mathtt{T}_{t+1}]| \quad . \tag{3}$$

Firstly, we have $|\Pr[\mathtt{T}_0] - \Pr[\mathtt{T}_1]| \leq O(2^{-\mathsf{len}})$ and $|\Pr[\mathtt{T}_2] - \Pr[\mathtt{T}_3]| \leq O(2^{-\mathsf{len}})$, since it holds that $\mathbf{SD}(\mathsf{U}_{\left[\frac{N'-1}{4}\right]}, \mathsf{U}_{[n']}) = \frac{P'+Q'-1}{N'-2} = O(2^{-\mathsf{len}})$. Below, we estimate $|\Pr[\mathtt{T}_1] - \Pr[\mathtt{T}_2]|$ and $|\Pr[\mathtt{T}_3] - \Pr[\mathtt{T}_4]|$.

Estimation of $|\Pr[\mathtt{T}_1] - \Pr[\mathtt{T}_2]|$**.** The view of $\mathcal{A}$ in Game 1 is indistinguishable from that in Game 2 due to the hardness of $\mathsf{IV}_{s,\ell}$. More specifically, there exists a PPT adversary $\mathcal{B}_{\mathsf{iv}}$ that makes a single query and satisfies $|\Pr[\mathtt{T}_1] - \Pr[\mathtt{T}_2]| = \mathsf{Adv}^{\mathsf{iv}}_{s,\ell,\mathcal{B}_{\mathsf{iv}}}(\lambda)$. We show the description of $\mathcal{B}_{\mathsf{iv}}$ below.

Given $(N', g', (g'_k = g'^{\alpha_k} \bmod N'^s)_{k\in[\ell]})$ from the challenger, $\mathcal{B}_{\mathsf{iv}}$ gives $\widetilde{z} := \frac{N'-1}{4}$ to $\mathcal{A}$, and receives from $\mathcal{A}$ the bound $B \geq \widetilde{z}$ of the interval for key-shifting values. $\mathcal{B}$ next chooses $H \xleftarrow{\mathsf{r}} \mathcal{H}$, and sets $\mathsf{pp} \leftarrow (N', T', g', H)$. Then, $\mathcal{B}_{\mathsf{iv}}$ samples $\theta_1, \ldots, \theta_\ell \xleftarrow{\mathsf{r}} [N'^{s-1}]$, submits a query $(\theta_1, \ldots, \theta_\ell)$ to the challenger, and receives a reply $(e_1, \ldots, e_\ell)$. $\mathcal{B}_{\mathsf{iv}}$ then regards g'_k as the k-th ciphertext ct_k, and samples a shift $\Delta_k \xleftarrow{\mathsf{r}} [B]$ and computes $\mathsf{K}_k \leftarrow H(e_k \cdot \mathsf{ct}_k^{\Delta_k} \bmod N'^s)$ for every $k \in [\ell]$. Then, $\mathcal{B}_{\mathsf{iv}}$ sends pp, $(\Delta_k)_{k\in[\ell]}$, and $(\mathsf{ct}_k, \mathsf{K}_k)_{k\in[\ell]}$ to $\mathcal{A}$. When $\mathcal{A}$ returns a guess bit b', $\mathcal{B}_{\mathsf{iv}}$ forwards it to the challenger and terminates.

By regarding the randomness $r \in [\frac{N'-1}{4}]$ chosen by $\mathcal{B}_{\mathsf{iv}}$'s challenger in response to $\mathcal{B}_{\mathsf{iv}}$'s query as a secret key sk for $\mathcal{A}$, it is straightforward to see that for each $\sigma \in \{0,1\}$, if $\mathcal{B}$'s challenge bit is σ, then $\mathsf{K}_k = H(T'^{\sigma\cdot\theta_k} \cdot \mathsf{ct}_k^{\mathsf{sk}+\Delta_k} \bmod N'^s)$ holds for every $k \in [\ell]$, and thus $\mathcal{B}_{\mathsf{iv}}$ simulates Game $(1+\sigma)$ perfectly for $\mathcal{A}$. Since $\mathcal{B}_{\mathsf{iv}}$ uses $\mathcal{A}$'s final guess bit directly, we have $\mathsf{Adv}^{\mathsf{iv}}_{s,\ell,\mathcal{B}_{\mathsf{iv}}}(\lambda) = |\Pr[\mathtt{T}_1] - \Pr[\mathtt{T}_2]|$.

Hence, due to Lemma 3, there exists another PPT adversary $\mathcal{B}$ such that $|\Pr[\mathtt{T}_1] - \Pr[\mathtt{T}_2]| \leq 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}} + O(\ell \cdot 2^{-\mathsf{len}})$.

Estimation of $|\Pr[\mathtt{T}_3] - \Pr[\mathtt{T}_4]|$**.** Due to the leftover hash lemma (Lemma 1), the view of $\mathcal{A}$ in Game 3 is $(\ell \cdot \epsilon_{\mathsf{LHL}})$-close to that in Game 4, and thus we have $|\Pr[\mathtt{T}_3] - \Pr[\mathtt{T}_4]| \leq \ell \cdot \epsilon_{\mathsf{LHL}}$. This can be seen as follows.

Consider Game 3 in which all the randomness except $H \in \mathcal{H}$ and $\{\theta_k\}_{k\in[\ell]}$ are fixed. Note that this determines $\mathsf{param} = (N', P', Q', T', g')$ output by GGen, $\mathcal{A}$'s randomness $r_{\mathcal{A}}$ and the bound $B \geq \widetilde{z}$ chosen by $\mathcal{A}$, the secret key sk, all key-shifting values $\{\Delta_k\}_{k\in[\ell]}$ and ciphertexts $\{\mathsf{ct}_k = g'^{\alpha_k} \bmod N'^s\}_{k\in[\ell]}$. Now, for each $k \in [\ell]$, define the distribution $X_k := \{\theta_k \xleftarrow{\mathsf{r}} [N'^{s-1}] : T'^{\theta_k} \cdot \mathsf{ct}_k^{(\mathsf{sk}+\Delta_k)} \bmod N'^s\}$, which is independent of the choice of H. (Recall that $\mathcal{A}$ determines the bound B for Δ_k's before it is given pp that contains H.) Then, consider the distribution $D_0 := \big(r_{\mathcal{A}}, \mathsf{pp}, (\Delta_k)_{k\in[\ell]}, (\mathsf{ct}_k)_{k\in[\ell]}, (H(X_k))_{k\in[\ell]}\big)$ and the distribution $D_1 := \big(r_{\mathcal{A}}, \mathsf{pp}, (\Delta_k)_{k\in[\ell]}, (\mathsf{ct}_k)_{k\in[\ell]}, (\mathsf{U}_{\mathcal{K}})^\ell\big)$, where $\mathsf{pp} = (N', T', g', H)$ and $H \xleftarrow{\mathsf{r}} \mathcal{H}$. Note that for each $\sigma \in \{0,1\}$, D_σ corresponds to the view of $\mathcal{A}$ in Game $(3+\sigma)$ in which everything except H and $(\mathsf{K}_k)_{k\in\ell}$ is fixed. Hence, $\mathbf{SD}(D_0, D_1)$ upper-bounds $|\Pr[\mathtt{T}_3] - \Pr[\mathtt{T}_4]|$. Also, notice that $\mathbf{SD}(D_0, D_1) = \mathbf{SD}\left((H, (H(X_k))_{k\in[\ell]}), (H, (\mathsf{U}_{\mathcal{K}})^\ell)\right)$ (where $H \xleftarrow{\mathsf{r}} \mathcal{H}$) due to the fixing of the values other than H and $\{\theta_k\}_{k\in[\ell]}$. Moreover, since $\mathbf{H}_\infty(X_k) = (s-1)\log N' \geq (s-1)\cdot(2\mathsf{len}-1)$ holds for every $k \in [\ell]$, the distribution of $(H, H(X_k))$ is ϵ_{LHL}-close to $(H, \mathsf{U}_{\mathcal{K}})$ by the leftover hash lemma (Lemma 1). Then, the triangle inequality implies $\mathbf{SD}\left((H, (H(X_k))_{k\in[\ell]}), (H, (\mathsf{U}_{\{0,1\}^n})^\ell)\right) \leq \ell \cdot \epsilon_{\mathsf{LHL}}$. Consequently, we have $|\Pr[\mathtt{T}_3] - \Pr[\mathtt{T}_4]| \leq \ell \cdot \epsilon_{\mathsf{LHL}}$.

Hence, using Equation 3, we can conclude that there exists a PPT adversary $\mathcal{B}$ satisfying Equation 2, as required. □ (**Lemma 4**)

5 KDM-CCA Secure PKE with respect to Affine Functions

In this section, we show a PKE scheme that is KDM-CCA secure with respect to affine functions based on the DCR assumption.

We first specify the *DCR language* with respect to which the underlying PHF family used in our proposed scheme is considered. Then, we give our proposed PKE scheme in Section 5.1. We also give two instantiations for the underlying PHF family, the first one in Section 5.2 and the second one in Section 5.3.

$\mathsf{Setup}_{\mathsf{aff}}(1^\lambda)$: $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$ $\mathsf{pp}_{\mathsf{phf}} \leftarrow \mathsf{Setup}_{\mathsf{phf}}(\mathsf{param})$ $(\mathsf{pp}_{\mathsf{skem}}, z, \widetilde{z}) \leftarrow \mathsf{Setup}_{\mathsf{skem}}(1^\lambda)$ $\mathsf{pp}_{\mathsf{cca}} \leftarrow \mathsf{Setup}_{\mathsf{cca}}(1^\lambda)$ $\mathsf{pp}_{\mathsf{aff}} \leftarrow (N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{pp}_{\mathsf{skem}}, \mathsf{pp}_{\mathsf{cca}})$ Return $\mathsf{pp}_{\mathsf{aff}}$.	$\mathsf{KG}_{\mathsf{aff}}(\mathsf{pp}_{\mathsf{aff}})$: $(N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{pp}_{\mathsf{skem}}, \mathsf{pp}_{\mathsf{cca}}) \leftarrow \mathsf{pp}_{\mathsf{aff}}$ $x \xleftarrow{\mathsf{r}} [\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi]$ $(\mathsf{ct}, \mathsf{K}) \leftarrow \mathsf{Encap}(\mathsf{pp}_{\mathsf{skem}}, x)$ Parse K as $(r^{\mathsf{KG}}, \mathsf{psk}) \in \mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$. $h \leftarrow g^{2x} \bmod N^s$ $\mathsf{ppk} \leftarrow \mu(\mathsf{psk})$ $(\mathsf{cpk}, \mathsf{csk}) \leftarrow \mathsf{KG}_{\mathsf{cca}}(\mathsf{pp}_{\mathsf{cca}}; r^{\mathsf{KG}})$ Return $\mathsf{PK} := (h, \mathsf{ct}, \mathsf{ppk}, \mathsf{cpk})$ and $\mathsf{SK} := x$.
$\mathsf{Enc}_{\mathsf{aff}}(\mathsf{PK}, m \in \mathbb{Z}_{N^{s-1}})$: $(h, \mathsf{ct}, \mathsf{ppk}, \mathsf{cpk}) \leftarrow \mathsf{PK}$ $r \xleftarrow{\mathsf{r}} [\frac{N-1}{4}]$ $u \leftarrow g^r \bmod N^s$ $v \leftarrow T^m \cdot h^r \bmod N^s$ $\pi \leftarrow \mathsf{Pub}(\mathsf{ppk}, u^2 \bmod N^s, 2r)$ $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}(\mathsf{cpk}, (u, v, \pi))$ Return CT.	$\mathsf{Dec}_{\mathsf{aff}}(\mathsf{PK}, \mathsf{SK}, \mathsf{CT})$: $(h, \mathsf{ct}, \mathsf{ppk}, \mathsf{cpk}) \leftarrow \mathsf{PK}$; $x \leftarrow \mathsf{SK}$ $\mathsf{K} \leftarrow \mathsf{Decap}(\mathsf{pp}_{\mathsf{skem}}, x, \mathsf{ct})$ Parse K as $(r^{\mathsf{KG}}, \mathsf{psk}) \in \mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$. $(\mathsf{cpk}, \mathsf{csk}) \leftarrow \mathsf{KG}_{\mathsf{cca}}(\mathsf{pp}_{\mathsf{cca}}; r^{\mathsf{KG}})$ $(u, v, \pi) \leftarrow \mathsf{Dec}_{\mathsf{cca}}(\mathsf{cpk}, \mathsf{csk}, \mathsf{CT})$ If $(u, v) \notin \mathbb{J}^2_{N^s}$ then return $\perp$. If $\pi \neq \Lambda_{\mathsf{psk}}(u^2 \bmod N^s)$ then return $\perp$. Return $m \leftarrow \log_T(v \cdot u^{-2x} \bmod N^s)$.

Figure 5: The proposed KDM-CCA secure PKE scheme Π_{aff} with respect to affine functions. (The public parameter $\mathsf{pp}_{\mathsf{aff}}$ is omitted from the inputs to $\mathsf{Enc}_{\mathsf{aff}}$ and $\mathsf{Dec}_{\mathsf{aff}}$.)

DCR language. Let $s \geq 2$, GGen be the DCR group generator, and $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$. The set of yes instances Π_{yes} is the subgroup G_n of $\mathbb{J}_{N^s}$, and the set of no instances Π_{no} is $G_{N^{s-1}} \otimes G_n \setminus G_n$. Note that we can represent any yes instance $c \in G_n$ as $c = g^r \bmod N^s$, where $r \in \mathbb{Z}$. Thus, such r works as a witness for $c \in \Pi_{\mathsf{yes}}$.

5.1 Proposed PKE Scheme

Let $s \geq 2$, and GGen be the DCR group generator. Let $\Pi_{\mathsf{cca}} = (\mathsf{Setup}_{\mathsf{cca}}, \mathsf{KG}_{\mathsf{cca}}, \mathsf{Enc}_{\mathsf{cca}}, \mathsf{Dec}_{\mathsf{cca}})$ be a PKE scheme such that the randomness space of $\mathsf{KG}_{\mathsf{cca}}$ is $\mathcal{R}^{\mathsf{KG}}$. Let $\mathsf{PHF} = (\mathsf{Setup}_{\mathsf{phf}}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ be a PHF family with respect to GGen for the DCR language (defined as above). Let $\mathsf{SKEM} = (\mathsf{Setup}_{\mathsf{skem}}, \mathsf{Encap}, \mathsf{Decap})$ be an SKEM whose session key space is $\mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$.[7] Finally, let $\xi = \xi(\lambda)$ be any polynomial such that $2^{-\xi} = \mathsf{negl}(\lambda)$. Using these building blocks, our proposed PKE scheme $\Pi_{\mathsf{aff}} = (\mathsf{Setup}_{\mathsf{aff}}, \mathsf{KG}_{\mathsf{aff}}, \mathsf{Enc}_{\mathsf{aff}}, \mathsf{Dec}_{\mathsf{aff}})$ is constructed as described in Figure 5. The plaintext space of Π_{aff} is $\mathbb{Z}_{N^{s-1}}$, where N is the modulus generated in $\mathsf{Setup}_{\mathsf{aff}}$.

The correctness of Π_{aff} follows from that of SKEM and Π_{cca}, and the projective property of PHF.

We note that although our scheme has correctness and can be proved secure for any $s \geq 2$, the plaintext space of our scheme is $\mathbb{Z}_{N^{s-1}}$, and thus if $s = 2$, then the plaintext space $\mathbb{Z}_N$ becomes smaller than the secret key space $\left[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi\right]$, in which case KDM security for affine functions does not even capture circular security. (Malkin et al.'s scheme [23] has exactly the same issue.) If $\widetilde{z} \cdot 2^\xi$ is smaller than N, then the secret key space can be contained in $\mathbb{Z}_{N^2}$, in which case $s \geq 3$ is sufficient in practice.[8]

[7] Strictly speaking, the concrete format of $\mathcal{SK}$ could be dependent on a public parameter $\mathsf{pp}_{\mathsf{phf}}$ of PHF. However, as noted in Remark 3, the session-key space of an SKEM can be flexibly adjusted by using a pseudorandom generator. Hence, for simplicity we assume that such an adjustment of the spaces is applied.

[8] Actually, if $s = 3$ and our DCR-based instantiation in Section 4.2 is used as the underlying SKEM, then the

We also note that even if the building block SKEM SKEM and/or PKE scheme Π_{cca} are instantiated also from the DCR assumption (or any other factoring-related assumption), the DCR groups formed by (N, T, g) in $\mathsf{pp}_{\mathsf{aff}}$ should not be shared with those used in SKEM and/or Π_{cca}. This is because in our security proof, the reduction algorithms for SKEM and Π_{cca} will use the information of P and Q behind N. (See our security proof below.) We also remark that in our construction, N has to be generated by a trusted party, or by users jointly via some secure computation protocol, so that no user knows its factorization. (The same applies to our DCR-based SKEM.) This is the same setting as in the previous DCR-based (KDM-)CCA secure PKE schemes [23, 12, 14].

Before proving the KDM-CCA security of Π_{aff}, we also note the difference between the "inner scheme" of Π_{aff} and Malkin et al.'s scheme [23]. Although these schemes are essentially the same, there is a subtle difference. Specifically, when generating h contained in PK of Π_{aff}, we generate it as $h \leftarrow g^{2x} \bmod N^s$ while it is generated as $h \leftarrow g^x \bmod N^s$ in Malkin et al.'s scheme. Moreover, such additional squarings are performed on u in the decryption procedure of our scheme. By these additional squarings, if it is guaranteed that an element u appearing in the decryption procedure belongs to $\mathbb{J}_{N^s} = G_{N^{s-1}} \otimes \langle -1 \rangle \otimes G_n$, it can be converted to an element in $G_{N^{s-1}} \otimes G_n$. Thus, we can consider a PHF family on $G_{N^{s-1}} \otimes G_n$ rather than $G_{N^{s-1}} \otimes \langle -1 \rangle \otimes G_n$, and as a result, we need not worry about a case that an adversary for Π_{aff} may learn $x \bmod 2$ through decryption queries. This helps us to simplify the security proof. Note that we cannot explicitly require that group elements contained in a ciphertext be elements in $G_{N^{s-1}} \otimes G_n$ since it is not known how to efficiently check the membership in $G_{N^{s-1}} \otimes G_n$ without the factorization of N, while we can efficiently check the membership in $\mathbb{J}_{N^s}$ using only N.

KDM-CCA security. Let ℓ be the number of keys in the security game. We will show that Π_{aff} is KDM-CCA secure with respect to the function family $\mathcal{F}_{\mathsf{aff}}$ consisting of functions described as

$$f(x_1, \ldots, x_\ell) = \sum_{k \in [\ell]} a_k x_k + a_0 \bmod N^{s-1} \ ,$$

where $a_0, \ldots, a_\ell \in \mathbb{Z}_{N^{s-1}}$. Formally, we prove the following theorem.

Theorem 1 *Assume that the DCR assumption holds with respect to* GGen, SKEM *is passively RKA secure,* PHF *is computationally universal, and* Π_{cca} *is IND-CCA secure. Then,* Π_{aff} *is* $\mathcal{F}_{\mathsf{aff}}$*-KDM-CCA secure.*

Specifically, for any polynomial $\ell = \ell(\lambda)$ *and PPT adversary* $\mathcal{A}$ *that attacks the* $\mathcal{F}_{\mathsf{aff}}$*-KDM-CCA security of* Π_{aff} *and makes* $q_{\mathsf{kdm}} = q_{\mathsf{kdm}}(\lambda)$ *KDM queries and* $q_{\mathsf{dec}} = q_{\mathsf{dec}}(\lambda)$ *decryption queries, there exist PPT adversaries* $\mathcal{B}_{\mathsf{dcr}}$, $\mathcal{B}_{\mathsf{rka}}$, $\mathcal{B}'_{\mathsf{rka}}$, $\mathcal{B}_{\mathsf{cca}}$, $\mathcal{B}'_{\mathsf{cca}}$, *and* $\mathcal{B}_{\mathsf{cu}}$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{kdmcca}}_{\Pi_{\mathsf{aff}},\mathcal{F}_{\mathsf{aff}},\ell,\mathcal{A}}(\lambda) \leq 2 \cdot \Big(& 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}_{\mathsf{dcr}}}(\lambda) + \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}_{\mathsf{rka}}}(\lambda) + \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}'_{\mathsf{rka}}}(\lambda) \\ & + \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}_{\mathsf{cca}}}(\lambda) + \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}'_{\mathsf{cca}}}(\lambda) + \ell \cdot (q_{\mathsf{dec}} \cdot \mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{B}_{\mathsf{cu}}}(\lambda) + 2^{-\xi}) \Big) \\ & + O(q_{\mathsf{kdm}} \cdot 2^{-\mathsf{len}}) + O(2^{-\lambda}) \ . \quad (4)\end{aligned}$$

Remark 4 (Tightness of the reduction) Note that our reductions to the DCR assumption and the security of the building blocks are tight, except for the reduction to the computational

RSA modulus N generated at the setup of our PKE construction has to be ξ-bit larger than the RSA modulus generated at the setup of SKEM to satisfy $[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^{\xi}] \subset \mathbb{Z}_{N^2}$. We do not need this special treatment if $s \geq 4$.

universal property of the underlying PHF family PHF, which has the factor $\ell \cdot q_{\mathsf{dec}}$. However, if PHF satisfies the *statistical* universal property, the term $\mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{B}_{\mathsf{cu}}}(\lambda)$ can be replaced with a negligible function that is independent of a computational assumption, and thus our reduction becomes fully tight. Hence, if we use an SKEM and an IND-CCA PKE scheme with a tight security reduction to the DCR assumption (or another assumption A), the overall reduction to the DCR(& A) assumption becomes fully tight as well.

Proof of Theorem 1. Let ℓ be the number of keys, and $\mathcal{A}$ be a PPT adversary that attacks the $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA security of Π_{aff} and makes at most q_{kdm} KDM and q_{dec} decryption queries. We proceed the proof via a sequence of games argument using 8 games (Game 0 to Game 7). For every $t \in \{0,\dots,7\}$, let $\mathtt{SUC}_t$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game t. Our goal is to bound every term appearing in the following Equation 5.

$$\begin{aligned}\mathsf{Adv}^{\mathsf{kdmcca}}_{\Pi_{\mathsf{aff}},\mathcal{F}_{\mathsf{aff}},\ell,\mathcal{A}}(\lambda) &= 2 \cdot \left|\Pr[\mathtt{SUC}_0] - \frac{1}{2}\right| \\ &\leq 2 \cdot \left(\sum_{t\in\{0,\dots,6\}} |\Pr[\mathtt{SUC}_t] - \Pr[\mathtt{SUC}_{t+1}]| + \left|\Pr[\mathtt{SUC}_7] - \frac{1}{2}\right| \right) \quad . \end{aligned} \tag{5}$$

Game 0**:** This is the original $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA game regarding Π_{aff}. By definition, we have $\mathsf{Adv}^{\mathsf{kdmcca}}_{\Pi_{\mathsf{aff}},\mathcal{F}_{\mathsf{aff}},\ell,\mathcal{A}}(\lambda) = 2 \cdot \left|\Pr[\mathtt{SUC}_0] - \frac{1}{2}\right|$.

The detailed description of the game is as follows.

1. The challenger chooses $b \xleftarrow{\mathsf{r}} \{0,1\}$. Then the challenger generates $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$, $\mathsf{pp}_{\mathsf{phf}} \leftarrow \mathsf{Setup}_{\mathsf{phf}}(\mathsf{param})$, $(\mathsf{pp}_{\mathsf{skem}}, z, \widetilde{z}) \leftarrow \mathsf{Setup}_{\mathsf{skem}}(1^\lambda)$, and $\mathsf{pp}_{\mathsf{cca}} \leftarrow \mathsf{Setup}_{\mathsf{cca}}(1^\lambda)$, and sets $\mathsf{pp}_{\mathsf{aff}} := \left(N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{pp}_{\mathsf{skem}}, \mathsf{pp}_{\mathsf{cca}}\right)$. Next, the challenger generates $(\mathsf{PK}_k, \mathsf{SK}_k)$ for every $k \in [\ell]$ as follows.
   (a) Generate $x_k \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi\right]$.
   (b) Generate $(\mathsf{ct}_k, \mathsf{K}_k) \leftarrow \mathsf{Encap}(\mathsf{pp}_{\mathsf{skem}}, x_k)$ and parse $\left(r^{\mathsf{KG}}_k, \mathsf{psk}_k\right) \leftarrow \mathsf{K}_k$.
   (c) Compute $h_k \leftarrow g^{2x_k} \bmod N^s$ and $\mathsf{ppk}_k \leftarrow \mu\left(\mathsf{psk}_k\right)$.
   (d) Generate $(\mathsf{cpk}_k, \mathsf{csk}_k) \leftarrow \mathsf{KG}_{\mathsf{cca}}\left(\mathsf{pp}_{\mathsf{cca}}; r^{\mathsf{KG}}_k\right)$.
   (e) Set $\mathsf{PK}_k := (h_k, \mathsf{ct}_k, \mathsf{ppk}_k, \mathsf{cpk}_k)$ and $\mathsf{SK}_k := x_k$.

   The challenger sends $\mathsf{pp}_{\mathsf{aff}}$ and $\{\mathsf{PK}_k\}_{k\in[\ell]}$ to $\mathcal{A}$ and prepares a list L_{kdm}.
2. The challenger responds to queries made by $\mathcal{A}$.
   For a KDM query $\left(j, \left(a^0_0, \dots, a^0_\ell\right), \left(a^1_0, \dots, a^1_\ell\right)\right)$ made by $\mathcal{A}$, the challenger responds as follows.
   (a) Set $m := \sum_{k\in[\ell]} a^b_k x_k + a^b_0 \bmod N^{s-1}$.
   (b) Generate $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ and compute $u \leftarrow g^r \bmod N^s$.
   (c) Compute $v \leftarrow T^m \cdot h^r_j \bmod N^s$ and $\pi \leftarrow \mathsf{Pub}\left(\mathsf{ppk}_j, u^2 \bmod N^s, 2r\right)$.
   (d) Return $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, (u, v, \pi)\right)$ and add (j, CT) to L_{kdm}.

   For a decryption query (j, CT) made by $\mathcal{A}$, the challenger returns $\perp$ to $\mathcal{A}$ if $(j, \mathsf{CT}) \in L_{\mathsf{kdm}}$, and otherwise responds as follows.
   (a) Compute $(u, v, \pi) \leftarrow \mathsf{Dec}_{\mathsf{cca}}(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT})$. If $(u, v) \notin \mathbb{J}^2_{N^s}$, return $\perp$. Otherwise, compute as follows.
   (b) Return $\perp$ if $\pi \neq \Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right)$and $m \leftarrow \log_T\left(v \cdot u^{-2x_j} \bmod N^s\right)$ otherwise.

Note that the above procedure is not exactly the same as that of the decryption algorithm $\mathsf{Dec}_{\mathsf{aff}}$, because the computations of Decap and $\mathsf{KG}_{\mathsf{cca}}$ for generating csk_j and psk_j are omitted. However, the answer to a decryption query computed by the above procedure is exactly the same as that computed by $\mathsf{Dec}_{\mathsf{aff}}$. Therefore, it does not affect the view of $\mathcal{A}$.

3. $\mathcal{A}$ outputs $b' \in \{0,1\}$.

Game 1: Same as Game 0, except for how KDM queries are replied. When $\mathcal{A}$ makes a KDM query $\left(j, \left(a_0^0, \ldots, a_\ell^0\right), \left(a_0^1, \ldots, a_\ell^1\right)\right)$, the challenger generates v and π respectively by $v \leftarrow T^m \cdot u^{2x_j} \bmod N^s$ and $\pi \leftarrow \Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right)$, instead of $v \leftarrow T^m \cdot h_j^r \bmod N^s$ and $\pi \leftarrow \mathsf{Pub}\left(\mathsf{ppk}_j, u^2 \bmod N^s, 2r\right)$, where $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ and $u = g^r \bmod N^s$.

v is generated identically in both games. Moreover, by the projective property of PHF, $\Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right) = \mathsf{Pub}\left(\mathsf{ppk}_j, u^2 \bmod N^s, 2r\right)$ holds, and thus π is also generated identically in both games. Hence, we have $|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| = 0$.

Game 2: Same as Game 1, except for how the challenger generates $\{x_k\}_{k\in[\ell]}$. The challenger first generates $x \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z}\right]$. Then, for every $k \in [\ell]$, the challenger generates $\Delta_k \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi\right]$ and computes $x_k \leftarrow x + \Delta_k$, where the addition is done over $\mathbb{Z}$.

$|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \leq \ell \cdot 2^{-\xi}$ holds since the distribution of x_k in Game 2 and that in Game 1 are $2^{-\xi}$-close for every $k \in [\ell]$.

Next, we will change the game so that we can respond to KDM queries made by $\mathcal{A}$ using only $x \bmod n = x \bmod \frac{\phi(N)}{4}$. To this end, we make some preparation. Observe that in Game 2, the answer to a KDM query $\left(j, \left(a_0^0, \ldots, a_\ell^0\right), \left(a_0^1, \ldots, a_\ell^1\right)\right)$ is $\mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, (u, v, \pi)\right)$, where

$$\begin{aligned} u &= g^r \bmod N^s \ , \\ v &= T^{\sum_{k\in[\ell]} a_k^b x_k + a_0^b} \cdot u^{2x_j} \bmod N^s \ , \\ \pi &= \Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right) \ , \end{aligned}$$

and $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$. We also have

$$\sum_{k\in[\ell]} a_k^b x_k + a_0^b = \sum_{k\in[\ell]} a_k^b \left(x + \Delta_k\right) + a_0^b = \left(\sum_{k\in[\ell]} a_k^b\right) x + \sum_{k\in[\ell]} a_k^b \Delta_k + a_0^b \ ,$$

where the addition is done over $\mathbb{Z}$. Thus, by defining

$$A^b = \sum_{k\in[\ell]} a_k^b \quad \text{and} \quad B^b = \sum_{k\in[\ell]} a_k^b \Delta_k + a_0^b \ , \tag{6}$$

we have $v = T^{A^b x + B^b} \cdot u^{2x_j} \bmod N^s = T^{A^b x + B^b} \cdot (g^r)^{2x_j} \bmod N^s$. Note that A^b and B^b are computed only from $\left(a_0^b, \ldots, a_\ell^b\right)$ and $\{\Delta_k\}_{k\in[\ell]}$.

Game 3: Same as Game 2, except that for a KDM query $\left(j, \left(a_0^0, \ldots, a_\ell^0\right), \left(a_0^1, \ldots, a_\ell^1\right)\right)$ made by $\mathcal{A}$, the challenger responds as follows. (The difference from Game 2 is only in Step 3.)

1. Compute A^b and B^b as in Equation 6.
2. Generate $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$.

3. Compute $u \leftarrow T^{-\frac{A^b}{2}} \cdot g^r \bmod N^s$.
4. Compute $v \leftarrow T^{A^b x + B^b} \cdot u^{2x_j} \bmod N^s$.
5. Compute $\pi \leftarrow \Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right)$.
6. Return $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, (u, v, \pi)\right)$ and add (j, CT) to L_{kdm}.

Under the hardness of $\mathsf{IV}_{s,1}$, the distributions of $g^r \bmod N^s$ and $T^{-\frac{A^b}{2}} \cdot g^r \bmod N^s$ are computationally indistinguishable. More specifically, there exists a PPT adversary $\mathcal{B}_{\mathsf{iv}}$ that makes q_{kdm} sample queries in the $\mathsf{IV}_{s,1}$ game and satisfies $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| = \mathsf{Adv}^{\mathsf{iv}}_{s,1,\mathcal{B}_{\mathsf{iv}}}(\lambda)$. Due to Lemma 2, this means that there exists another PPT adversary $\mathcal{B}_{\mathsf{dcr}}$ such that $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}_{\mathsf{dcr}}}(\lambda) + O(q_{\mathsf{kdm}} \cdot 2^{-\mathsf{len}})$.

In Game 3, the answer to a KDM query $\left(j, \left(a_0^0, \ldots, a_\ell^0\right), \left(a_0^1, \ldots, a_\ell^1\right)\right)$ is $\mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, (u, v, \pi)\right)$, where

$$
\begin{aligned}
u &= T^{-\frac{A^b}{2}} \cdot g^r \bmod N^s \ , \\
v &= T^{A^b x + B^b} \cdot u^{2x_j} \bmod N^s = T^{B^b - A^b \Delta_j} \cdot g^{2r(x \bmod n)} \cdot g^{2r\Delta_j} \bmod N^s \ , \\
\pi &= \Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right) \ ,
\end{aligned}
$$

$r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$, and A^b and B^b are computed as in Equation 6. Thus, we can reply to a KDM query made by $\mathcal{A}$ using only $x \bmod n = x \bmod \frac{\phi(N)}{4}$.

We next change how decryption queries made by $\mathcal{A}$ are replied.

Game 4: Same as Game 3, except for how the challenger responds to decryption queries made by $\mathcal{A}$. For a decryption query (j, CT) made by $\mathcal{A}$, the challenger returns $\perp$ to $\mathcal{A}$ if $(j, \mathsf{CT}) \in L_{\mathsf{kdm}}$, and otherwise responds as follows. (The difference from Game 3 is adding Step 2 to the procedure.)

1. Compute $(u, v, \pi) \leftarrow \mathsf{Dec}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT}\right)$. If $(u, v) \notin \mathbb{J}^2_{N^s}$, return $\perp$. Otherwise, compute as follows.
2. If $u \notin \langle -1 \rangle \otimes G_n$, return $\perp$. Otherwise, compute as follows.
3. Return $\perp$ if $\pi \neq \Lambda_{\mathsf{psk}_j}\left(u^2 \bmod N^s\right)$ and otherwise $m \leftarrow \log_T\left(v \cdot u^{-2x_j} \bmod N^s\right)$.

We define the following event in Game $i \in \{4, 5, 6, 7\}$.

$\mathtt{BDQ}_i$**:** $\mathcal{A}$ makes a decryption query $(j, \mathsf{CT}) \notin L_{\mathsf{kdm}}$ which satisfies the following conditions, where $(u, v, \pi) \leftarrow \mathsf{Dec}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT}\right)$.

- $(u, v) \in \mathbb{J}^2_{N^s}$.
- $u \notin \langle -1 \rangle \otimes G_n$. Note that $\mathbb{J}_{N^s} = \langle -1 \rangle \otimes G_{N^{s-1}} \otimes G_n$.
- $\pi = \Lambda_{\mathsf{psk}_j}(u^2 \bmod N^s)$.

We call such a decryption query a "**bad decryption query**".

Games 3 and 4 are identical unless $\mathcal{A}$ makes a bad decryption query in each game. Therefore, we have $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \Pr[\mathtt{BDQ}_4]$. Combining this with the triangle inequality, we will also bound the terms in the following Equation 7:

$$
|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \sum_{t \in \{4,5,6\}} |\Pr[\mathtt{BDQ}_t] - \Pr[\mathtt{BDQ}_{t+1}]| + \Pr[\mathtt{BDQ}_7] \ . \tag{7}
$$

Let (j, CT) be a decryption query made by $\mathcal{A}$ and $(u, v, \pi) \leftarrow \mathsf{Dec}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT}\right)$. If the query is not a bad decryption query and $u \in \mathbb{J}_{N^s}$, then $(u^2 \bmod N^s) \in G_n$. Thus,

$$u^{2x_j} \bmod N^s = (u^2)^{x+\Delta_j} \bmod N^s = (u^2 \bmod N^s)^{(x \bmod n)} \cdot u^{2\Delta_j} \bmod N^s \ .$$

Thus, if the query is not a bad decryption query, the answer to it can be computed by using only $x \bmod n$.

Furthermore, recall that due to the "implicit modular-reduction in encapsulation" property of SKEM, for every $k \in [\ell]$, the SKEM-ciphertext/session-key pair $(\mathsf{ct}_k, \mathsf{K}_k)$ computed for generating the k-th public key PK_k at the initial phase, can be generated by using only $x_k \bmod z = x + \Delta_k \bmod z$.

Hence, due to the change in Game 4, now we have done the preparation for "decomposing" x into its "mod n"-component and its "mod z"-component.

Game 5: Same as Game 4, except that the challenger generates $\widehat{x} \xleftarrow{\mathsf{r}} [n]$ and $\bar{x} \xleftarrow{\mathsf{r}} [z]$ and then uses them for $x \bmod n$ and $x \bmod z$, respectively.

Note that when $x \xleftarrow{\mathsf{r}} [\frac{N-1}{4} \cdot \widetilde{z}]$, the statistical distance between $(x \bmod n, x \bmod z)$ and $(\widehat{x} \bmod n, \bar{x} \bmod z)$ is bounded by $\mathbf{SD}(\mathsf{U}_{[\frac{N-1}{4} \cdot \widetilde{z}]}, \mathsf{U}_{[n \cdot z]})$, because if $x \xleftarrow{\mathsf{r}} [n \cdot z]$, then the distribution of $(x \bmod n, x \bmod z)$ and that of $(\widehat{x} \bmod n, \bar{x} \bmod z)$ are identical due to the Chinese remainder theorem.[9] Note also that $\mathbf{SD}(\mathsf{U}_{[\frac{N-1}{4} \cdot \widetilde{z}]}, \mathsf{U}_{[n \cdot z]}) \leq \mathbf{SD}(\mathsf{U}_{[\frac{N-1}{4}]}, \mathsf{U}_{[n]}) + \mathbf{SD}(\mathsf{U}_{[\widetilde{z}]}, \mathsf{U}_{[z]})$. Here, the former statistical distance is $\frac{P+Q-2}{N-1} = O(2^{-\mathsf{len}}) \leq O(2^{-\lambda})$, and the latter statistical distance is bounded by $O(2^{-\lambda})$ due to the "approximate samplability of a secret key" property of SKEM. Hence, we have $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| \leq O(2^{-\lambda})$ and $|\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]| \leq O(2^{-\lambda})$.

Game 6: Same as Game 5, except that for every $k \in [\ell]$, the challenger generates $\mathsf{K}_k \xleftarrow{\mathsf{r}} \mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$ from which $r_k^{\mathsf{KG}} \in \mathcal{R}^{\mathsf{KG}}$ and $\mathsf{psk}_k \in \mathcal{SK}$ are generated, instead of using K_k associated with ct_k.

By the passive RKA security of SKEM, the view of $\mathcal{A}$ in Game 6 is indistinguishable from that of Game 5. More specifically, there exist PPT adversaries $\mathcal{B}_{\mathsf{rka}}$ and $\mathcal{B}'_{\mathsf{rka}}$ that attack the passive RKA security of SKEM so that $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}_{\mathsf{rka}}}(\lambda)$ and $|\Pr[\mathtt{BDQ}_5] - \Pr[\mathtt{BDQ}_6]| = \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}'_{\mathsf{rka}}}(\lambda)$ hold, respectively. The description of $\mathcal{B}_{\mathsf{rka}}$ and $\mathcal{B}'_{\mathsf{rka}}$ is as follows.

We first show the description of $\mathcal{B}_{\mathsf{rka}}$. Given $\widetilde{z}$ from the challenger, $\mathcal{B}_{\mathsf{rka}}$ generates $(N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$, and sets the bound $B = \frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi$. $\mathcal{B}_{\mathsf{rka}}$ then sends B to the challenger, and receives $\mathsf{pp}_{\mathsf{skem}}$, key-shifting values $(\Delta_k)_{k \in [\ell]}$, and challenge ciphertext/session key pairs $(\mathsf{ct}_k, \mathsf{K}_k)_{k \in [\ell]}$. Then, using the values known to $\mathcal{B}_{\mathsf{rka}}$ so far, it generates all the remaining values of $\mathsf{pp}_{\mathsf{aff}}$ and $\{\mathsf{PK}_k\}_{k \in [\ell]}$ where $\mathcal{B}_{\mathsf{rka}}$ regards sk in $\mathcal{B}_{\mathsf{rka}}$'s passive RKA security game as $\bar{x}$ in the game simulated by $\mathcal{B}_{\mathsf{rka}}$, and then interacts with $\mathcal{A}$ in exactly the same way as the challenger in Game 5 does. (Note that $\mathcal{B}_{\mathsf{rka}}$ has/generates all the information (e.g. P, Q, $\widehat{x}$) necessary to respond to KDM and decryption queries from $\mathcal{A}$.) Finally, $\mathcal{B}_{\mathsf{rka}}$ outputs 1 if $\mathcal{A}$ succeeds in guessing the challenge bit b (which was chosen by $\mathcal{B}_{\mathsf{rka}}$), otherwise outputs 0, and terminates. It is straightforward to see that if $\mathcal{B}_{\mathsf{rka}}$'s challenge bit is 1 (resp. 0), $\mathcal{B}_{\mathsf{rka}}$ simulates Game 5 (resp. Game 6) perfectly for $\mathcal{A}$. In particular, if $\mathcal{B}_{\mathsf{rka}}$'s challenge bit is 1, then every pair

[9] Here, we are implicitly assuming that $n = pq$ and z are relatively prime. This occurs with overwhelming probability due to the DCR assumption. We thus ignore the case of n and z are not relatively prime in the proof for simplicity. Note that the tightness of the reduction to the DCR assumption is not affected even if we take into account the possibility of this undesirable event.

$(\mathsf{K}_k, \mathsf{ct}_k)$ is a real ciphertext/session-key pair generated by using a shifted secret key $\bar{x} + \Delta_k$, which is exactly as in Game 5. On the other hand, if $\mathcal{B}_{\mathsf{rka}}$'s challenge bit is 0, then every K_k is chosen uniformly at random from $\mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$, which is exactly as in Game 6. Thus, we have $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}_{\mathsf{rka}}}(\lambda)$.

The design of $\mathcal{B}'_{\mathsf{rka}}$ is exactly as $\mathcal{B}_{\mathsf{rka}}$, except that $\mathcal{B}'_{\mathsf{rka}}$ outputs 1 if and only if $\mathcal{A}$ has made a bad decryption query (which can be checked by $\mathcal{B}'_{\mathsf{rka}}$ because it owns P and Q). Such $\mathcal{B}'_{\mathsf{rka}}$ satisfies $|\Pr[\mathtt{BDQ}_5] - \Pr[\mathtt{BDQ}_6]| = \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}'_{\mathsf{rka}}}(\lambda)$.

Game 7: Same as Game 6, except that the challenger responds to KDM queries (j, CT) made by $\mathcal{A}$ with $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, (0,0,0)\right)$.

We can consider straightforward reductions to the security of the underlying PKE scheme Π_{cca} for bounding $|\Pr[\mathtt{SUC}_6] - \Pr[\mathtt{SUC}_7]|$ and $|\Pr[\mathtt{BDQ}_6] - \Pr[\mathtt{BDQ}_7]|$. Note that the reduction algorithms can check whether $\mathcal{A}$ makes a bad decryption query or not by using decryption queries for Π_{cca}, and $\phi(N)$ and $\{\mathsf{psk}_k\}_{k\in[\ell]}$ that could be generated by the reductions themselves. Thus, there exist PPT adversaries $\mathcal{B}_{\mathsf{cca}}$ and $\mathcal{B}'_{\mathsf{cca}}$ such that $|\Pr[\mathtt{SUC}_6] - \Pr[\mathtt{SUC}_7]| = \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}_{\mathsf{cca}}}(\lambda)$ and $|\Pr[\mathtt{BDQ}_6] - \Pr[\mathtt{BDQ}_7]| = \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}'_{\mathsf{cca}}}(\lambda)$.

In Game 7, the challenge bit b is information-theoretically hidden from the view of $\mathcal{A}$. Thus, we have $\left|\Pr[\mathtt{SUC}_7] - \frac{1}{2}\right| = 0$.

Finally, $\Pr[\mathtt{BDQ}_7]$ is bounded by the computational universal property of PHF. More specifically, there exists a PPT adversary $\mathcal{B}_{\mathsf{cu}}$ such that $\Pr[\mathtt{BDQ}_7] \leq \ell \cdot q_{\mathsf{dec}} \cdot \mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{B}_{\mathsf{cu}}}(\lambda) + O(2^{-\mathsf{len}})$. To see this, consider the following PPT adversary $\mathcal{B}_{\mathsf{cu}}$. Given $(N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{ppk})$ from the challenger, $\mathcal{B}_{\mathsf{cu}}$ picks $j^* \in [\ell]$ uniformly at random, and regards the given ppk as the projection key ppk_{j^*} of the j^*-th user (and thus implicitly regards the corresponding secret key psk as psk_{j^*}). Then, $\mathcal{B}_{\mathsf{cu}}$ generates all the remaining values of $\mathsf{pp}_{\mathsf{aff}}$ and $\{\mathsf{PK}_k\}_{k\in[\ell]}$ in exactly the same way as the challenger in Game 7 does, with one exception that $\mathcal{B}_{\mathsf{cu}}$ picks $\widehat{x}$ from $\left[\frac{N-1}{4}\right]$, instead of $[n]$. Then, it starts interacting with $\mathcal{A}$. $\mathcal{B}_{\mathsf{cu}}$ responds to the KDM queries from $\mathcal{A}$ in exactly the same way as the challenger in Game 7 does. $\mathcal{B}_{\mathsf{cu}}$ responds to the decryption queries (j, CT) from $\mathcal{A}$ in the same way as the challenger in Game 7 does, except for the following points:

- To check whether $u \in \langle -1\rangle \otimes G_n$, $\mathcal{B}_{\mathsf{cu}}$ submits $u^2 \bmod N^s$ as an evaluation query to the challenger. If the answer π to the query is not $\perp$, $\mathcal{B}_{\mathsf{cu}}$ decides that $u \in \langle -1\rangle \otimes G_n$, and otherwise $u \notin \langle -1\rangle \otimes G_n$. Note that if $u \in \mathbb{J}_{N^s}$, then $u^2 \bmod N^s$ is guaranteed to be either an yes or no instance, and $u \in \langle -1\rangle \otimes G_n$ if and only if $u^2 \bmod N^s$ is an yes instance.

- Furthermore, if $j = j^*$ and the challenger's response π is not $\perp$, then $\mathcal{B}_{\mathsf{cu}}$ uses the answer π to the query as $\Lambda_{\mathsf{psk}_{j^*}}(u^2 \bmod N^s)$ in the decryption procedure.

When $\mathcal{A}$ terminates, $\mathcal{B}_{\mathsf{cu}}$ picks one of $\mathcal{A}$'s decryption queries uniformly at random, which we denote by (j', CT'). If $j' = j^*$, $\mathsf{Dec}_{\mathsf{cca}}(\mathsf{cpk}_{j^*}, \mathsf{csk}_{j^*}, \mathsf{CT}') = (u', v', \pi') \in \mathbb{J}^2_{N^s} \times \{0,1\}^*$, and $u'^2 \bmod N^s \in G_{N^{s-1}} \otimes G_n \setminus G_n$ hold simultaneously, $\mathcal{B}_{\mathsf{cu}}$ returns $(u'^2 \bmod N^s, \pi')$ and terminates (where $u^2 \bmod N^s \in G_{N^{s-1}} \otimes G_n \setminus G_n$ can be checked by making an evaluation query). Otherwise, $\mathcal{B}_{\mathsf{cu}}$ simply gives up and aborts.

Note that other than picking $\widehat{x}$ from $\left[\frac{N-1}{4}\right]$ (instead from $[n]$), $\mathcal{B}_{\mathsf{cu}}$ perfectly simulates Game 7 for $\mathcal{A}$, and thus the difference between the probability that $\mathcal{A}$ makes a bad decryption query in the game simulated by $\mathcal{B}_{\mathsf{cu}}$ and that in Game 7 are bounded by $\mathbf{SD}(\mathsf{U}_{\left[\frac{N-1}{4}\right]}, \mathsf{U}_{[n]}) = O(2^{-\mathsf{len}})$. Note also that the position j^* is information-theoretically hidden from $\mathcal{A}$'s view, and $\mathcal{B}_{\mathsf{cu}}$ finally picks a query (j', CT') uniformly from $\mathcal{A}$'s q_{dec} decryption queries. Hence, conditioned on the event that $\mathcal{A}$ has made a bad decryption query, the probability that $j' = j^*$ and $\mathcal{B}_{\mathsf{cu}}$

picks it as (j', CT') in its final step is at least $\frac{1}{\ell \cdot q_{\mathsf{dec}}}$. Consequently, we have $\mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{B}_{\mathsf{cu}}}(\lambda) \geq \frac{1}{\ell \cdot q_{\mathsf{dec}}} \cdot (\Pr[\mathtt{BDQ}_7] - O(2^{-\mathsf{len}}))$.

From the above arguments and Equations 5 and 7, we can conclude that there exist PPT adversaries $\mathcal{B}_{\mathsf{dcr}}$, $\mathcal{B}_{\mathsf{rka}}$, $\mathcal{B}'_{\mathsf{rka}}$, $\mathcal{B}_{\mathsf{cca}}$, $\mathcal{B}'_{\mathsf{cca}}$, and $\mathcal{B}_{\mathsf{cu}}$ satisfying Equation 4, as required. □ (**Theorem 1**)

5.2 Basic Construction of Projective Hash Function

For the PHF family for the DCR language used in our construction Π_{aff}, we provide two instantiations: the basic construction $\mathsf{PHF}_{\mathsf{aff}}$ that achieves the statistical universal property in this subsection, and its "space-efficient" variant $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ that achieves only the computational universal property in the next subsection.

Let $s \geq 2$, and GGen be the DCR group generator. The basic construction $\mathsf{PHF}_{\mathsf{aff}} = (\mathsf{Setup}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ is as follows. (The construction here is basically the universal PHF family for the DCR setting by Cramer and Shoup [9], extended for general $s \geq 2$.) Recall that $\Pi_{\mathsf{yes}} = G_n$ and $\Pi_{\mathsf{no}} = G_{N^{s-1}} \otimes G_n \setminus G_n$ for the DCR language. Given param output from $\mathsf{GGen}(1^\lambda, s)$, Setup outputs a public parameter pp that concretely specifies $(\mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ defined as follows. We define $\mathcal{SK} := \left[N^{s-1} \cdot \frac{N-1}{4}\right]$, $\mathcal{PK} := G_n$, and $\mathcal{K} := G_{N^{s-1}} \otimes G_n$. For every $\mathsf{sk} \in \left[N^{s-1} \cdot \frac{N-1}{4}\right]$ and $c \in G_{N^{s-1}} \otimes G_n$, we also define μ and Λ as

$$\mu(\mathsf{sk}) := g^{\mathsf{sk}} \bmod N^s \quad \text{and} \quad \Lambda_{\mathsf{sk}}(c) := c^{\mathsf{sk}} \bmod N^s \ .$$

Projective property. Let $\mathsf{sk} \in \left[N^{s-1} \cdot \frac{N-1}{4}\right]$, $\mathsf{pk} = g^{\mathsf{sk}} \bmod N^s$, and $c = g^r \bmod N^s$, where $r \in \mathbb{Z}$ is regarded as a witness for $c \in G_n$. We define the public evaluation algorithm Pub as

$$\mathsf{Pub}(\mathsf{pk}, c, r) := \mathsf{pk}^r \bmod N^s \ .$$

We see that

$$\mathsf{pk}^r \equiv \left(g^{\mathsf{sk}}\right)^r \equiv (g^r)^{\mathsf{sk}} \equiv \Lambda_{\mathsf{sk}}(c) \bmod N^s \ ,$$

and thus $\mathsf{PHF}_{\mathsf{aff}}$ satisfies the projective property.

Universal property. We now show that $\mathsf{PHF}_{\mathsf{aff}}$ satisfies the statistical universal property. Let $\mathsf{pk} \in G_n$, $c \in G_{N^{s-1}} \otimes G_n \setminus G_n$, and $\pi \in G_{N^{s-1}} \otimes G_n$. We can write c and π as $c = T^\theta \cdot g^r \bmod N^s$ and $\pi = T^{\theta'} \cdot g^{r'} \bmod N^s$, where $\theta, \theta' \in \left[N^{s-1}\right]$ such that $\theta \neq N^{s-1}$ and $r, r' \in [n]$. Consider the following probability

$$\begin{aligned}
\Pr_{\mathsf{sk} \xleftarrow{\mathsf{r}} [N^{s-1} \cdot n]}[\Lambda_{\mathsf{sk}}(c) = \pi | \mu(\mathsf{sk}) = \mathsf{pk}] &= \Pr_{\mathsf{sk} \xleftarrow{\mathsf{r}} [N^{s-1} \cdot n]} \left[c^{\mathsf{sk}} \equiv \pi \bmod N^s \middle| g^{\mathsf{sk}} \equiv \mathsf{pk} \bmod N^s \right] \\
&\leq \Pr_{\mathsf{sk} \xleftarrow{\mathsf{r}} [N^{s-1} \cdot n]} \left[T^{\theta \cdot \mathsf{sk}} \equiv T^{\theta'} \bmod N^s \middle| \mathsf{sk} \equiv \log_g \mathsf{pk} \bmod n \right] \\
&= \Pr_{\mathsf{sk}^* \xleftarrow{\mathsf{r}} [N^{s-1}]} \left[\theta \cdot \mathsf{sk}^* \equiv \theta' \bmod N^{s-1} \right] \\
&\leq \Pr_{\mathsf{sk}^* \xleftarrow{\mathsf{r}} [P^{s-1}]} \left[\theta \cdot \mathsf{sk}^* \equiv \theta' \bmod P^{s-1} \right] \ , \qquad (8)
\end{aligned}$$

where the last inequality is true even if P is replaced with Q. Since $\theta \neq N^{s-1}$, we have either $\theta \not\equiv 0 \bmod P^{s-1}$ or $\theta \not\equiv 0 \bmod Q^{s-1}$. Without loss of generality, we assume that $\theta \not\equiv 0 \bmod P^{s-1}$. (Otherwise, switch the role of P and Q.) Below, we prove that the probability in Equation 8 is bounded by $O(2^{-\mathsf{len}})$.

First, we consider the case of $\mathsf{GCD}(P^{s-1}, \theta) = 1$.[10] In this case, there exists a multiplicative

[10] If $s = 2$, this always holds.

inverse of θ mod P^{s-1}, and Equation 8 is at most $\frac{1}{P^{s-1}} \leq O(2^{-\mathsf{len}})$.

Next, we consider the case of $\mathsf{GCD}(P^{s-1}, \theta) \neq 1$. In this case, there exist $\zeta \in [s-2]$ and θ_0 that is co-prime to P such that $\theta = P^{\zeta} \cdot \theta_0$. Then, if θ' is not a multiple of P^{ζ}, then Equation 8 is 0. Otherwise (i.e. θ' is a multiple of P^{ζ}), Equation 8 is equal to

$$\Pr_{\mathsf{sk}^* \xleftarrow{r} [P^{s-\zeta-1}]} \left[\theta_0 \cdot \mathsf{sk}^* \equiv \frac{\theta'}{P^{\zeta}} \bmod P^{s-\zeta-1} \right] .$$

Since θ_0 is co-prime to P, there exists a multiplicative inverse of θ_0 mod $P^{s-\zeta-1}$. Then, the above probability is bounded by $\frac{1}{P^{s-\zeta-1}} \leq O(2^{-\mathsf{len}})$.

Therefore, the probability in Equation 8 is bounded by $O(2^{-\mathsf{len}})$ in any case. Moreover, $\mathbf{SD}\left(\mathsf{U}_{\mathcal{SK}}, \mathsf{U}_{[N^{s-1} \cdot n]}\right) = \mathbf{SD}\left(\mathsf{U}_{\left[N^{s-1} \cdot \frac{N-1}{4}\right]}, \mathsf{U}_{[N^{s-1} \cdot n]}\right) \leq O\left(2^{-\mathsf{len}}\right)$. Thus, $\mathsf{PHF}_{\mathsf{aff}}$ is $O(2^{-\mathsf{len}})$-universal.

5.3 Space-Efficient Construction of Projective Hash Function

The second instantiation is a "space-efficient" variant of the first construction. Specifically, it is obtained from $\mathsf{PHF}_{\mathsf{aff}}$ by "compressing" the output of the function Λ in $\mathsf{PHF}_{\mathsf{aff}}$ with a collision resistant hash function.

More formally, let $\mathcal{H} = \left\{H : \{0,1\}^* \to \{0,1\}^{\mathsf{len}_{\mathsf{crhf}}}\right\}$ be a collision resistant hash family. Then, consider the "compressed"-version of the PHF family $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}} = (\mathsf{Setup}', \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}' := \{0,1\}^{\mathsf{len}_{\mathsf{crhf}}}, \Lambda', \mu, \mathsf{Pub}')$, in which Setup' picks $H \xleftarrow{r} \mathcal{H}$ in addition to generating $\mathsf{pp} \leftarrow \mathsf{Setup}$, Λ' is defined simply by composing Λ and H by $\Lambda'_{\mathsf{sk}}(\cdot) := H(\Lambda_{\mathsf{sk}}(\cdot))$, Pub' is defined similarly by composing Pub and H, and the remaining components are unchanged from $\mathsf{PHF}_{\mathsf{aff}}$. $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ preserves the projective property of $\mathsf{PHF}_{\mathsf{aff}}$ and it is possible to show that the "compressed" construction $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ satisfies the computational universal property.

This "compressing technique" is applicable to not only the specific instantiation $\mathsf{PHF}_{\mathsf{aff}}$, but also more general PHF families PHF, so that if the underlying PHF is (statistically) universal and satisfies some additional natural properties (that are satisfied by our instantiation in Section 5.2) and $\mathcal{H}$ is collision resistant, then the resulting "compressed" version $\mathsf{PHF}^{\mathsf{hash}}$ is computationally universal. In Section A, we formally show the additional natural properties, and the formal statement for the compressing technique as well as its proof.

The obvious merit of using $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ instead of $\mathsf{PHF}_{\mathsf{aff}}$ is its smaller output size. The disadvantage is that unfortunately, the computational universal property of $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ is only loosely reduced to the collision resistance of $\mathcal{H}$. Specifically, the advantage of a computational universal adversary is bounded only by the square root of the advantage of the collision resistance adversary (reduction algorithm). For the details, see Section A.

6 KDM-CCA Secure PKE with respect to Polynomials

In this section, we show a PKE scheme that is KDM-CCA secure with respect to polynomials based on the DCR assumption. More specifically, our scheme is KDM-CCA secure with respect to modular arithmetic circuits (MAC) defined by Malkin et al. [23].

Our scheme is based on the *cascaded ElGamal encryption* scheme used by Malkin et al., and uses a PHF family for a language that is associated with it, which we call the *cascaded ElGamal language*. Furthermore, for considering a PHF family for this language, we need to make a small extension to the syntax of the functions μ, and thus we also introduce it here as well.

After introducing the cascaded ElGamal language as well as the extension to a PHF family below, we will show our proposed PKE scheme in Section 6.1, and give the instantiations of the underlying PHF family in Section 6.2.

Augmenting the syntax of PHFs. For our construction in this section, we use a PHF family whose syntax is slightly extended from Definition 3. Specifically, we introduce an auxiliary key $\mathsf{ak} \in \mathcal{AK}$ that is used as part of a public parameter pp output by Setup, where $\mathcal{AK}$ itself could also be parameterized by param output by GGen. Then, we allow this ak to (1) affect the structure of the witnesses for Π_{yes}, and (2) be taken as input by the projection map μ so that it takes $\mathsf{ak} \in \mathcal{AK}$ and $\mathsf{sk} \in \mathcal{SK}$ as input. We simply refer to a PHF family with such augmentation as an augmented PHF family.

For an augmented PHF family, we have to slightly adapt the definition of the statistical/computational universal property from Definition 4. Specifically,

- for the definition of the ϵ-universal property, in addition to param, pp, $\mathsf{pk} \in \mathcal{PK}$, $c \in \Pi_{\mathsf{no}}$, and $\pi \in \mathcal{K}$, we also take the universal quantifier for all $\mathsf{ak} \in \mathcal{AK}$ for considering the probability in Equation 1.
- for the definition of the computational universal property, we change the initial phase (Step 1) of the game to allow an adversary to choose $\mathsf{ak} \in \mathcal{AK}$ in the following way:

  1. First, the challenger executes $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$, and sends (N, T, g) to $\mathcal{A}$. $\mathcal{A}$ sends $\mathsf{ak} \in \mathcal{AK}$ to the challenger. The challenger then executes $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{param})$, chooses $\mathsf{sk} \xleftarrow{\mathsf{r}} \mathcal{SK}$, and computes $\mathsf{pk} \leftarrow \mu(\mathsf{ak}, \mathsf{sk})$. Then, the challenger sends $(\mathsf{pp}, \mathsf{pk})$ to $\mathcal{A}$.

  The remaining description of the game and the definition of the adversary's advantage are unchanged.

We note that the implication of the statistical universal property to the computational one, is also true for an augmented PHF family.

Cascaded ElGamal language. Let $s \geq 2$, GGen be the DCR group generator, and $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}\left(1^\lambda, s\right)$. Let $d = d(\lambda)$ be a polynomial. Let the auxiliary key space $\mathcal{AK}$ be defined as G_n, and let $\mathsf{ak} \in \mathcal{AK}$ (which will be a public key of the underlying cascaded ElGamal encryption scheme in our concrete instantiations of PHFs). The set of yes instances Π_{yes} is G_n^d, and the set of no instances is $(G_{N^{s-1}} \otimes G_n)^d \setminus G_n^d$. Any yes instance $c \in G_n^d$ can be expressed in the form $c = (c_1, \ldots, c_d)$ such that $c_d = g^{r_d} \bmod N^s$ and $c_i = g^{r_i} \cdot \mathsf{ak}^{r_{i+1}} \bmod N^s$ for every $i \in [d-1]$, where $r = (r_1, \ldots, r_d) \in \mathbb{Z}^d$. Thus, such r works as a witness for $c \in \Pi_{\mathsf{yes}}$ under $\mathsf{ak} \in \mathcal{AK}$.

6.1 Proposed PKE Scheme

Let $s \geq 2$, and GGen be the DCR group generator. Let $d = d(\lambda)$ be a polynomial. Let $\Pi_{\mathsf{cca}} = (\mathsf{Setup}_{\mathsf{cca}}, \mathsf{KG}_{\mathsf{cca}}, \mathsf{Enc}_{\mathsf{cca}}, \mathsf{Dec}_{\mathsf{cca}})$ be a PKE scheme such that the randomness space of $\mathsf{KG}_{\mathsf{cca}}$ is $\mathcal{R}^{\mathsf{KG}}$. Let $\mathsf{PHF} = (\mathsf{Setup}_{\mathsf{phf}}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \mu, \Lambda, \mathsf{Pub})$ be an augmented PHF family with respect to GGen for the cascaded ElGamal language (defined as above). Let $\mathsf{SKEM} = (\mathsf{Setup}_{\mathsf{skem}}, \mathsf{Encap}, \mathsf{Decap})$ be an SKEM whose session-key space is $\mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$.[11] Finally, let $\xi = \xi(\lambda)$ be any polynomial such that $2^{-\xi} = \mathsf{negl}(\lambda)$. Using these building blocks, our proposed

[11] The same format adjustment as in Π_{aff} can be applied. See the footnote in Section 5.1.

$\mathsf{Setup}_{\mathsf{poly}}(1^\lambda)$: $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$ $\mathsf{pp}_{\mathsf{phf}} \leftarrow \mathsf{Setup}_{\mathsf{phf}}(\mathsf{param})$ $(\mathsf{pp}_{\mathsf{skem}}, z, \widetilde{z}) \leftarrow \mathsf{Setup}_{\mathsf{skem}}(1^\lambda)$ $\mathsf{pp}_{\mathsf{cca}} \leftarrow \mathsf{Setup}_{\mathsf{cca}}(1^\lambda)$ $\mathsf{pp}_{\mathsf{poly}} \leftarrow (N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{pp}_{\mathsf{skem}}, \mathsf{pp}_{\mathsf{cca}})$ Return $\mathsf{pp}_{\mathsf{poly}}$.	$\mathsf{KG}_{\mathsf{poly}}(\mathsf{pp}_{\mathsf{poly}})$: $(N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{pp}_{\mathsf{skem}}, \mathsf{pp}_{\mathsf{cca}}) \leftarrow \mathsf{pp}_{\mathsf{poly}}$ $x \xleftarrow{\mathsf{r}} [\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi]$ $(\mathsf{ct}, \mathsf{K}) \leftarrow \mathsf{Encap}(\mathsf{pp}_{\mathsf{skem}}, x)$ Parse K as $(r^{\mathsf{KG}}, \mathsf{psk}) \in \mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$. $h \leftarrow g^{2x} \bmod N^s$ $\mathsf{ppk} \leftarrow \mu(h, \mathsf{psk})$ //h is used as an aux. key $(\mathsf{cpk}, \mathsf{csk}) \leftarrow \mathsf{KG}_{\mathsf{cca}}(\mathsf{pp}_{\mathsf{cca}}; r^{\mathsf{KG}})$ Return $\mathsf{PK} := (h, \mathsf{ct}, \mathsf{ppk}, \mathsf{cpk})$ and $\mathsf{SK} := x$.
$\mathsf{Enc}_{\mathsf{poly}}(\mathsf{PK}, m \in \mathbb{Z}_{N^s})$: $(h, \mathsf{ct}, \mathsf{ppk}, \mathsf{cpk}) \leftarrow \mathsf{PK}$ $\forall i \in [d]$: $r_i \xleftarrow{\mathsf{r}} [\frac{N-1}{4}]$; $y_i \leftarrow g^{r_i} \bmod N^s$ $u_d \leftarrow y_d$ $\forall i \in [d-1]$: $u_i \leftarrow y_i \cdot h^{r_{i+1}} \bmod N^s$ $r \leftarrow (2r_1, \ldots, 2r_d)$ $u \leftarrow (u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s)$ $v \leftarrow T^m \cdot h^{r_1} \bmod N^s$ $\pi \leftarrow \mathsf{Pub}(\mathsf{ppk}, u, r)$ $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}(\mathsf{cpk}, (\{u_i\}_{i\in[d]}, v, \pi))$ Return CT.	$\mathsf{Dec}_{\mathsf{poly}}(\mathsf{PK}, \mathsf{SK}, \mathsf{CT})$: $(h, \mathsf{ct}, \mathsf{ppk}, \mathsf{cpk}) \leftarrow \mathsf{PK}$; $x \leftarrow \mathsf{SK}$ $\mathsf{K} \leftarrow \mathsf{Decap}(\mathsf{pp}_{\mathsf{skem}}, x, \mathsf{ct})$ Parse K as $(r^{\mathsf{KG}}, \mathsf{psk}) \in \mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$. $(\mathsf{cpk}, \mathsf{csk}) \leftarrow \mathsf{KG}_{\mathsf{cca}}(\mathsf{pp}_{\mathsf{cca}}; r^{\mathsf{KG}})$ $(\{u_i\}_{i\in[d]}, v, \pi) \leftarrow \mathsf{Dec}_{\mathsf{cca}}(\mathsf{cpk}, \mathsf{csk}, \mathsf{CT})$ If $(\{u\}_{i\in[d]}, v) \notin \mathbb{J}_{N^s}^{d+1}$ then return $\perp$. $u \leftarrow (u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s)$ If $\pi \neq \Lambda_{\mathsf{psk}}(u)$ then return $\perp$. $y_d \leftarrow u_d$ $\forall i \in [d-1]$: $y_i \leftarrow u_i \cdot (y_{i+1})^{-2x} \bmod N^s$ Return $m \leftarrow \log_T(v \cdot y_1^{-2x} \bmod N^s)$.

Figure 6: The proposed KDM-CCA secure PKE scheme Π_{poly} with respect to polynomials. (The public parameter $\mathsf{pp}_{\mathsf{poly}}$ is omitted from the inputs to $\mathsf{Enc}_{\mathsf{poly}}$ and $\mathsf{Dec}_{\mathsf{poly}}$.)

PKE scheme $\Pi_{\mathsf{poly}} = (\mathsf{Setup}_{\mathsf{poly}}, \mathsf{KG}_{\mathsf{poly}}, \mathsf{Enc}_{\mathsf{poly}}, \mathsf{Dec}_{\mathsf{poly}})$ is constructed as described in Figure 6. The plaintext space of Π_{poly} is $\mathbb{Z}_{N^{s-1}}$, where N is the RSA modulus generated in $\mathsf{Setup}_{\mathsf{poly}}$.

For the scheme Π_{poly}, the same remarks as those for Π_{aff} apply. Namely, the correctness and the security proof work for any $s \geq 2$, while to capture circular security, we should use $s \geq 3$. Furthermore, if we use a statistically universal PHF family, the KDM-CCA security of Π_{poly} is tightly reduced to the DCR assumption and the security properties of the building blocks Π_{cca} and SKEM.

For the scheme Π_{poly}, the same remarks as those for Π_{aff} apply. Namely, the correctness and the security proof work for any $s \geq 2$, while to capture circular security, we should use $s \geq 3$. Furthermore, if we use a statistically universal PHF family, the KDM-CCA security of Π_{poly} is tightly reduced to the DCR assumption and the security properties of the building blocks Π_{cca} and SKEM.

KDM-CCA security. Π_{poly} is KDM-CCA secure with respect to the class of circuits $\mathcal{MAC}_d$, consisting of circuits satisfying the following conditions.

- Inputs are variables and constants of $\mathbb{Z}_{N^{s-1}}$.
- Gates are $+$, $-$, or $\cdot$ over $\mathbb{Z}_{N^{s-1}}$ and the number of gates is polynomial in λ.
- Each circuit in $\mathcal{MAC}_d$ computes a polynomial whose degree is at most d. For a circuit $C \in \mathcal{MAC}_d$, we denote the polynomial computing C by f_C.

The formal statement for the security of Π_{poly} is as follows.

Theorem 2 *Assume that the DCR assumption holds with respect to* GGen, SKEM *is passively RKA secure,* PHF *is computationally universal, and* Π_{cca} *is IND-CCA secure. Then,* Π_{poly} *is* $\mathcal{MAC}_d$*-KDM-CCA secure.*

Specifically, for any polynomial $\ell = \ell(\lambda)$ *and PPT adversary* $\mathcal{A}$ *that attacks the* $\mathcal{MAC}_d$*-KDM-CCA security of* Π_{poly} *and makes* $q_{\mathsf{kdm}} = q_{\mathsf{kdm}}(\lambda)$ *KDM queries and* $q_{\mathsf{dec}} = q_{\mathsf{dec}}(\lambda)$ *decryption queries, there exist PPT adversaries* $\mathcal{B}_{\mathsf{dcr}}$, $\mathcal{B}_{\mathsf{rka}}$, $\mathcal{B}'_{\mathsf{rka}}$, $\mathcal{B}_{\mathsf{cca}}$, $\mathcal{B}'_{\mathsf{cca}}$, *and* $\mathcal{B}_{\mathsf{cu}}$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{kdmcca}}_{\Pi_{\mathsf{poly}},\mathcal{MAC}_d,\ell,\mathcal{A}}(\lambda) \leq 2 \cdot \Big(& 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}_{\mathsf{dcr}}}(\lambda) + \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}_{\mathsf{rka}}}(\lambda) + \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}'_{\mathsf{rka}}}(\lambda) \\ & + \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}_{\mathsf{cca}}}(\lambda) + \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}'_{\mathsf{cca}}}(\lambda) + \ell \cdot (q_{\mathsf{dec}} \cdot \mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{B}_{\mathsf{cu}}}(\lambda) + 2^{-\xi})\Big) \\ & + O(d \cdot q_{\mathsf{kdm}} \cdot 2^{-\mathsf{len}}) + O(2^{-\lambda}) \ . \end{aligned} \tag{9}$$

Before proving Theorem 2, we introduce the following lemma shown by Malkin et al. [23].

Lemma 5 *Let* d *and* ℓ *be polynomials of* λ*. Let* $K \in \mathbb{N}$*. There exists a PPT algorithm* Coeff *that, given* K, $j \in [\ell]$, $\Delta_1, \ldots, \Delta_\ell \in \mathbb{Z}$, *and a polynomial* f_C *computing* $C \in \mathcal{MAC}_d$*, outputs* $a_0, \ldots, a_d$ *such that*

$$f_C(x + \Delta_1, \ldots, x + \Delta_\ell) = \sum_{i \in [d]} a_i \, (x + \Delta_j)^i + a_0 \bmod K \ .$$

Proof of Theorem 2. In a high-level, the structure of the proof is the same as the proof for Π_{aff}.

Let ℓ be the number of keys, and $\mathcal{A}$ be a PPT adversary that attacks the $\mathcal{MAC}_d$-KDM-CCA security of Π_{poly} and makes at most q_{kdm} KDM and q_{dec} decryption queries. We proceed the proof via a sequence of games argument using 8 games (Game 0 to Game 7). For every $t \in \{0, \ldots, 7\}$, let $\mathtt{SUC}_t$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game t. Our goal is to bound every term appearing in the following Equation 10.

$$\begin{aligned}\mathsf{Adv}^{\mathsf{kdmcca}}_{\Pi_{\mathsf{aff}},\mathcal{MAC}_d,\mathcal{A},\ell}(\lambda) &= 2 \cdot \left| \Pr[\mathtt{SUC}_0] - \frac{1}{2} \right| \\ &\leq 2 \cdot \left(\sum_{t \in \{0,\ldots,6\}} |\Pr[\mathtt{SUC}_t] - \Pr[\mathtt{SUC}_{t+1}]| + \left| \Pr[\mathtt{SUC}_7] - \frac{1}{2} \right| \right) \ . \end{aligned} \tag{10}$$

Game 0**:** This is the original $\mathcal{MAC}_d$-KDM-CCA game regarding Π_{poly}. By definition, we have $\mathsf{Adv}^{\mathsf{kdmcca}}_{\Pi_{\mathsf{poly}},\mathcal{MAC}_d,\ell,\mathcal{A}}(\lambda) = 2 \cdot \left| \Pr[\mathtt{SUC}_0] - \frac{1}{2} \right|$.

The detailed description of the game is as follows.

1. The challenger chooses $b \xleftarrow{\mathsf{r}} \{0,1\}$. Then, the challenger generates $\mathsf{param} \leftarrow \mathsf{GGen}(1^\lambda, s)$, $\mathsf{pp}_{\mathsf{phf}} \leftarrow \mathsf{Setup}_{\mathsf{phf}}(\mathsf{param})$, $(\mathsf{pp}_{\mathsf{skem}}, z, \widetilde{z}) \leftarrow \mathsf{Setup}_{\mathsf{skem}}(1^\lambda)$, and $\mathsf{pp}_{\mathsf{cca}} \leftarrow \mathsf{Setup}_{\mathsf{cca}}(1^\lambda)$, and sets $\mathsf{pp}_{\mathsf{poly}} := \left(N, T, g, \mathsf{pp}_{\mathsf{phf}}, \mathsf{pp}_{\mathsf{skem}}, \mathsf{pp}_{\mathsf{cca}}\right)$. Next, the challenger generates $(\mathsf{PK}_k, \mathsf{SK}_k)$ for every $k \in [\ell]$ as follows.
   (a) Generate $x_k \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi\right]$.
   (b) Generate $(\mathsf{ct}_k, \mathsf{K}_k) \leftarrow \mathsf{Encap}(\mathsf{pp}_{\mathsf{skem}}, x_k)$ and parse $\left(r^{\mathsf{KG}}_k, \mathsf{psk}_k\right) \leftarrow \mathsf{K}_k$.
   (c) Compute $h_k \leftarrow g^{2x_k} \bmod N^s$ and $\mathsf{ppk}_k \leftarrow \mu\left(h_k, \mathsf{psk}_k\right)$.
   (d) Generate $(\mathsf{cpk}_k, \mathsf{csk}_k) \leftarrow \mathsf{KG}_{\mathsf{cca}}\left(\mathsf{pp}_{\mathsf{cca}}; r^{\mathsf{KG}}_k\right)$.
   (e) Set $\mathsf{PK}_k := (h_k, \mathsf{ct}_k, \mathsf{ppk}_k, \mathsf{cpk}_k)$ and $\mathsf{SK}_k := x_k$.

   The challenger sends $\mathsf{pp}_{\mathsf{poly}}$ and $\{\mathsf{PK}_k\}_{k \in [\ell]}$ to $\mathcal{A}$ and prepares a list L_{kdm}.

2. The challenger responds to queries made by $\mathcal{A}$.
   For a KDM query $\left(j, C^0, C^1\right)$ made by $\mathcal{A}$, the challenger responds as follows.
   (a) Set $m := f_{C^b}\left(x_1, \ldots, x_\ell\right)$.
   (b) Generate $r_i \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ and compute $y_i \leftarrow g^{r_i} \bmod N^s$ for every $i \in [d]$.
   (c) Set $u_d := y_d$ and compute $u_i \leftarrow y_i \cdot h_j^{r_{i+1}} \bmod N^s$ for every $i \in [d-1]$.
   (d) Set $r = (2r_1, \ldots, 2r_d)$ and $u = \left(u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s\right)$.
   (e) Compute $v \leftarrow T^m \cdot h_j^{r_1} \bmod N^s$ and $\pi \leftarrow \mathsf{Pub}(\mathsf{ppk}_j, u, r)$.
   (f) Return $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \left(\left\{u_i\right\}_{i\in[d]}, v, \pi\right)\right)$ and add (j, CT) to L_{kdm}.

   For a decryption query (j, CT) made by $\mathcal{A}$, the challenger returns $\perp$ to $\mathcal{A}$ if $(j, \mathsf{CT}) \in L_{\mathsf{kdm}}$, and otherwise responds as follows.
   (a) Compute $\left(\left\{u_i\right\}_{i\in[d]}, v, \pi\right) \leftarrow \mathsf{Dec}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT}\right)$. If $\left(\left\{u_i\right\}_{i\in[d]}, v\right) \notin \mathbb{J}_{N^s}^{d+1}$, return $\perp$. Otherwise, compute as follows.
   (b) Set $u := \left(u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s\right)$ and return $\perp$ if $\pi \neq \Lambda_{\mathsf{psk}_j}(u)$. Otherwise, compute as follows.
   (c) Set $y_d := u_d$ and compute $y_i \leftarrow u_i \cdot \left(y_{i+1}^{2x_j}\right)^{-1} \bmod N^s$ for every $i \in [d-1]$.
   (d) Return $m \leftarrow \log_T\left(v \cdot \left(y_1^{2x_j}\right)^{-1} \bmod N^s\right)$.

   Note that the above procedure is not exactly the same as that of the decryption algorithm $\mathsf{Dec}_{\mathsf{poly}}$, because the computations of Decap and $\mathsf{KG}_{\mathsf{cca}}$ for generating csk_j and psk_j are omitted. However, the answer to a decryption query computed by the above procedure is exactly the same as that computed by $\mathsf{Dec}_{\mathsf{poly}}$. Therefore, it does not affect the view of $\mathcal{A}$.
3. $\mathcal{A}$ outputs $b' \in \{0, 1\}$.

Game 1: Same as Game 0, except for how the challenger generates $\{u_i\}_{i\in[d-1]}$, v, and π when $\mathcal{A}$ makes a KDM query $\left(j, C^0, C^1\right)$. The challenger generates $u_i \leftarrow y_i \cdot y_{i+1}^{2x_j} \bmod N^s$ for every $i \in [d-1]$. Moreover, the challenger generates v and π respectively by $v \leftarrow T^m \cdot y_1^{2x_j} \bmod N^s$ and $\pi \leftarrow \Lambda_{\mathsf{psk}_j}(u)$, instead of $v \leftarrow T^m \cdot h_j^{r_1} \bmod N^s$ and $\pi \leftarrow \mathsf{Pub}\left(\mathsf{ppk}_j, u, r\right)$. ($r$ and u are generated in the same way as in Game 0.)

Clearly, $\{u_i\}_{i\in[d]}$ and v are generated identically in both games. Furthermore, by the projective property of PHF, $\Lambda_{\mathsf{psk}_j}(u) = \mathsf{Pub}\left(\mathsf{ppk}_j, u, r\right)$ holds, and thus π is also generated identically in both games. Hence, we have $|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| = 0$.

Game 2: Same as Game 1, except for how the challenger generates $\{x_k\}_{k\in[\ell]}$. The challenger first generates $x \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z}\right]$. Then, for every $k \in [\ell]$, the challenger generates $\Delta_k \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi\right]$ and computes $x_k \leftarrow x + \Delta_k$, where the addition is done over $\mathbb{Z}$.

$|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \leq \ell \cdot 2^{-\xi}$ holds since the statistical distance between the distribution of x_k in Game 2 and that in Game 1 is at most $2^{-\xi}$ for every $k \in [\ell]$.

Next, we will change the game so that we can respond to KDM queries made by $\mathcal{A}$ using only $x \bmod n = x \bmod \frac{\phi(N)}{4}$.

Game 3: Same as Game 2, except that for a KDM query $\left(j, C^0, C^1\right)$ made by $\mathcal{A}$, the challenger responds as follows. (The difference from Game 2 is only in Step 3.)

1. Compute $(a_0^b, \ldots, a_d^b) \leftarrow \mathsf{Coeff}\left(N^{s-1}, j, \Delta_1, \ldots, \Delta_\ell, f_{C^b}\right)$.
2. Set $m = f_{C^b}(x_1, \ldots, x_\ell) = a_0^b + \sum_{i\in[d]} a_i^b x_j^i \bmod N^{s-1}$.
3. Generate $r_i \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ and compute $y_i \leftarrow T^{A_i} \cdot g^{r_i} \bmod N^s$ for every $i \in [d]$, where $A_i = \left(-\frac{1}{2}\right)^i \cdot \sum_{i'=i}^{d} a_{i'}^b x_j^{i'-i}$.
4. Set $u_d := y_d$ and compute $u_i \leftarrow y_i \cdot y_{i+1}^{2x_j} \bmod N^s$ for every $i \in [d-1]$.
5. Set $u = \left(u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s\right)$.
6. Compute $v \leftarrow T^m \cdot y_1^{2x_j} \bmod N^s$ and $\pi \leftarrow \Lambda_{\mathsf{psk}_j}(u)$.
7. Return $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \left(\{u_i\}_{i\in[d]}, v, \pi\right)\right)$ and add (j, CT) to L_{kdm}.

Under the hardness of $\mathsf{IV}_{s,1}$, the distributions of $g^{r_i} \bmod N^s$ and $T^{A_i} \cdot g^{r_i} \bmod N^s$ are computationally indistinguishable for every $i \in [d]$. More specifically, there exists a PPT adversary $\mathcal{B}_{\mathsf{iv}}$ that makes $d \cdot q_{\mathsf{kdm}}$ sample queries in the $\mathsf{IV}_{s,1}$ game and satisfies $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| = \mathsf{Adv}^{\mathsf{iv}}_{s,1,\mathcal{B}_{\mathsf{iv}}}(\lambda)$. Due to Lemma 2, this means that there exists another PPT adversary $\mathcal{B}_{\mathsf{dcr}}$ such that $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}_{\mathsf{dcr}}}(\lambda) + O(d \cdot q_{\mathsf{kdm}} \cdot 2^{-\mathsf{len}})$.

In Game 3, the answer to a KDM query (j, C^0, C^1) is $\mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \left(\{u_i\}_{i\in[d]}, v, \pi\right)\right)$, where

$$
\begin{aligned}
u_d &= T^{\left(-\frac{1}{2}\right)^d \cdot a_d^b} \cdot g^{r_d} \bmod N^s \ , \\
u_i &= y_i \cdot y_{i+1}^{2x_j} \bmod N^s \\
&= T^{\left(-\frac{1}{2}\right)^i \cdot a_i^b} \cdot g^{2(r_{i+1})(x \bmod n)} \cdot g^{r_i + 2(r_{i+1})\Delta_j} \bmod N^s \quad \text{for every } i \in [d-1] \ , \\
v &= T^{a_0^b + \sum_{i\in[d]} a_i^b x_j^i} \cdot y_1^{2x_j} \bmod N^s \\
&= T^{a_0^b} \cdot g^{2r_1(x \bmod n)} \cdot g^{2r_1\Delta_j} \bmod N^s \ , \\
\pi &= \Lambda_{\mathsf{psk}_j}(u) \ ,
\end{aligned}
$$

where $r_i \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ for every $i \in [d]$, $u = (u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^2)$, and $(a_0^b, \ldots, a_d^b) \leftarrow \mathsf{Coeff}\left(N^{s-1}, j, \Delta_1, \ldots, \Delta_\ell, f_{C^b}\right)$. Thus, we can reply to a KDM query made by $\mathcal{A}$ using only $x \bmod n = x \bmod \frac{\phi(N)}{4}$.

We next change how decryption queries are replied.

Game 4: Same as Game 3, except for how the challenger responds to decryption queries made by $\mathcal{A}$. For a decryption query (j, CT) made by $\mathcal{A}$, the challenger returns $\perp$ to $\mathcal{A}$ if $(j, \mathsf{CT}) \in L_{\mathsf{kdm}}$, and otherwise responds as follows. (The difference from Game 3 is adding Step 2 to the procedure.)

1. Compute $\left(\{u_i\}_{i\in[d]}, v, \pi\right) \leftarrow \mathsf{Dec}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT}\right)$. If $\left(\{u_i\}_{i\in[d]}, v\right) \notin \mathbb{J}^{d+1}_{N^s}$, return $\perp$. Otherwise, compute as follows.
2. If $u_i \notin \langle -1 \rangle \otimes G_n$ for some $i \in [d]$, return $\perp$. Otherwise, compute as follows.
3. Set $u := \left(u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s\right)$ and return $\perp$ if $\pi \neq \Lambda_{\mathsf{psk}_j}(u)$. Otherwise, compute as follows.
4. Set $y_d := u_d$ and compute $y_i \leftarrow u_i \cdot \left(y_{i+1}^{2x_j}\right)^{-1} \bmod N^s$ for every $i \in [d-1]$.
5. Return $m \leftarrow \log_T \left(v \cdot \left(y_1^{2x_j}\right)^{-1} \bmod N^s\right)$.

We define the following event in Game $i \in \{4,5,6,7\}$.

$\mathtt{BDQ}_i$**:** $\mathcal{A}$ makes a decryption query $(j, \mathsf{CT}) \notin L_{\mathsf{kdm}}$ which satisfies the following conditions, where $\left(\{u_i\}_{i\in[d]}, v, \pi\right) \leftarrow \mathsf{Dec}_{\mathsf{cca}}\left(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT}\right)$.

- $\left(\{u_i\}_{i\in[d]}, v\right) \in \mathbb{J}^{d+1}_{N^s}$.
- $u_i \notin \langle -1\rangle \otimes G_n$ for some $i \in [d]$. Note that $\mathbb{J}_{N^s} = \langle -1\rangle \otimes G_{N^{s-1}} \otimes G_n$.
- $\pi = \Lambda_{\mathsf{psk}_j}(u)$, where $u = \left(u_1^2 \bmod N^s, \ldots, u_d^2 \bmod N^s\right)$.

We call such a decryption query a "**bad decryption query**".

Games 3 and 4 are identical unless $\mathcal{A}$ makes a bad decryption query in each game. Therefore, we have $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \Pr[\mathtt{BDQ}_4]$. Combining this with the triangle inequality, we will also bound the terms in the following Equation 11:

$$|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \sum_{t\in\{4,5,6\}} |\Pr[\mathtt{BDQ}_t] - \Pr[\mathtt{BDQ}_{t+1}]| + \Pr[\mathtt{BDQ}_7] \ . \tag{11}$$

Let (j, CT) be a decryption query made by $\mathcal{A}$. Furthermore, let $\left(\{u_i\}_{i\in[d]}, v, \pi\right) \leftarrow \mathsf{Dec}_{\mathsf{cca}}(\mathsf{cpk}_j, \mathsf{csk}_j, \mathsf{CT})$, $y_d \leftarrow u_d$, and $y_i \leftarrow u_i \cdot y_{i+1}^{-2x_j} \bmod N^s$ for every $i \in [d-1]$. Note that if the query is not a bad decryption query and $u_i \in \mathbb{J}_{N^s}$ for every $i \in [d]$, then $(u_i^2 \bmod N^s) \in G_n$ for every $i \in [d]$. Using this, for every $i = d, \ldots, 1$, we can inductively show that (1) $(y_i^{2x_j} \bmod N^s) \in G_n$ and (2) it can be computed only using $x \bmod n$. Specifically, for the base case of $i = d$, we have

$$\begin{aligned} y_d^{2x_j} \bmod N^s &= (u_d^2)^{x+\Delta_j} \bmod N^s \\ &= (u_d^2 \bmod N^s)^{(x \bmod n)} \cdot (u_d^2 \bmod N^s)^{\Delta_j} \bmod N^s \ , \end{aligned}$$

which also implies $(y_d^{2x_j} \bmod N^s) \in G_n$. For $i \in [d-1]$, if $(y_{i+1}^{2x_j} \bmod N^s) \in G_n$ has been already computed, then we have

$$\begin{aligned} y_i^{2x_j} \bmod N^s &= (u_i \cdot y_{i+1}^{-2x_j})^{2x_j} \bmod N^s \\ &= (u_i^2)^{x_j} \cdot (y_{i+1}^{2x_j})^{-2x_j} \bmod N^s \\ &= (u_i^2 \bmod N^s)^{(x \bmod n)} \cdot (u_i^2 \bmod N^s)^{\Delta_j} \\ &\quad \cdot (y_{i+1}^{2x_j} \bmod N^s)^{-2(x \bmod n)} \cdot (y_{i+1}^{2x_j} \bmod N^s)^{-2\Delta_j} \bmod N^s \ , \end{aligned}$$

which shows that $(y_i^{2x_j} \bmod N^2) \in G_n$ and can be computed only using $x \bmod n$. Hence, if the query is not a bad decryption query, the answer to it can be computed by using only $x \bmod n$.

Furthermore, recall that due to the "implicit modular-reduction in encapsulation" property of SKEM, for every $k \in [\ell]$, the SKEM-ciphertext/session-key pair $(\mathsf{ct}_k, \mathsf{K}_k)$ computed for generating the k-th public key PK_k at the initial phase, can be generated by using only $x_k \bmod z = x + \Delta_k \bmod z$.

Hence, due to the change in Game 4, now we have done the preparation for "decomposing" x into its "mod n"-component and its "mod z"-component.

Game 5: Same as Game 4, except that the challenger generates $\widehat{x} \xleftarrow{\mathsf{r}} [n]$ and $\bar{x} \xleftarrow{\mathsf{r}} [z]$ and then uses them for $x \bmod n$ and $x \bmod z$, respectively.

With exactly the same argument as in the corresponding step in the proof of Theorem 1, we have $|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| \leq O(2^{-\lambda})$ and $|\Pr[\mathtt{BDQ}_4] - \Pr[\mathtt{BDQ}_5]| \leq O(2^{-\lambda})$.

Game 6: Same as Game 5, except that for every $k \in [\ell]$, the challenger generates $\mathsf{K}_k \xleftarrow{\mathsf{r}} \mathcal{R}^{\mathsf{KG}} \times \mathcal{SK}$, from which $r_k^{\mathsf{KG}} \xleftarrow{\mathsf{r}} \mathcal{R}^{\mathsf{KG}}$ and $\mathsf{psk}_k \xleftarrow{\mathsf{r}} \mathcal{SK}$ are generated, instead of using K_k associated with ct_k.

Again, with exactly the same argument as in the corresponding step in the proof of Theorem 1, we can show that there exist PPT adversaries $\mathcal{B}_{\mathsf{rka}}$ and $\mathcal{B}'_{\mathsf{rka}}$ that use $\mathcal{A}$ as a subroutine and attack the passive RKA security of SKEM so that $|\Pr[\mathtt{SUC}_5] - \Pr[\mathtt{SUC}_6]| = \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}_{\mathsf{rka}}}(\lambda)$ and $|\Pr[\mathtt{BDQ}_5] - \Pr[\mathtt{BDQ}_6]| = \mathsf{Adv}^{\mathsf{rka}}_{\mathsf{SKEM},\ell,\mathcal{B}'_{\mathsf{rka}}}(\lambda)$ hold, respectively.

Game 7: Same as Game 6, except that the challenger responds to KDM queries made by $\mathcal{A}$ with $\mathsf{CT} \leftarrow \mathsf{Enc}_{\mathsf{cca}}\left(\mathsf{pk}_j, 0^{d+2}\right)$.

Again, with the same arguments as in the corresponding steps in the proof of Theorem 1, we have that

- There exist PPT adversaries $\mathcal{B}_{\mathsf{cca}}$ and $\mathcal{B}'_{\mathsf{cca}}$ such that $|\Pr[\mathtt{SUC}_6] - \Pr[\mathtt{SUC}_7]| = \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}_{\mathsf{cca}}}(\lambda)$ and $|\Pr[\mathtt{BDQ}_6] - \Pr[\mathtt{BDQ}_7]| = \mathsf{Adv}^{\mathsf{indcca}}_{\Pi_{\mathsf{cca}},\ell,\mathcal{B}'_{\mathsf{cca}}}(\lambda)$.
- $\left|\Pr[\mathtt{SUC}_7] - \frac{1}{2}\right| = 0$.
- There exists a PPT adversary $\mathcal{B}_{\mathsf{cu}}$ such that $\Pr[\mathtt{BDQ}_7] \leq \ell \cdot q_{\mathsf{dec}} \cdot \mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF},\mathcal{B}_{\mathsf{cu}}}(\lambda) + O(2^{-\mathsf{len}})$.

In fact, the description of $\mathcal{B}_{\mathsf{cu}}$ has to be slightly modified from the one we had in the proof of Theorem 1, to take into account that PHF is now an augmented PHF family. Given (N, T, g) from the challenger, $\mathcal{B}_{\mathsf{cu}}$ picks $j^* \xleftarrow{\mathsf{r}} [\ell]$, executes $(\mathsf{pp}_{\mathsf{skem}}, z, \widetilde{z}) \leftarrow \mathsf{Setup}_{\mathsf{skem}}(1^\lambda)$, samples $\widehat{x} \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ and $\Delta_{j^*} \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4} \cdot \widetilde{z} \cdot 2^\xi\right]$, and generates $h_{j^*} = g^{2(\widehat{x}+\Delta_{j^*})} \bmod N^s$. Then, $\mathcal{B}_{\mathsf{cu}}$ submits $h_{j^*} \in G_n$ as an auxiliary key, and receives $\mathsf{pp}_{\mathsf{phf}}$ and ppk from the challenger. Now, $\mathcal{B}_{\mathsf{cu}}$ regards ppk as ppk_{j^*} in the j^*-th user's public key. From here on $\mathcal{B}_{\mathsf{cu}}$ proceeds in exactly the same way as $\mathcal{B}_{\mathsf{cu}}$ in the proof of Theorem 1 does. It is straightforward to see that the same analysis as before applies.

From the above arguments and Equations 10 and 11, we can conclude that there exist PPT adversaries $\mathcal{B}_{\mathsf{dcr}}$, $\mathcal{B}_{\mathsf{rka}}$, $\mathcal{B}'_{\mathsf{rka}}$, $\mathcal{B}_{\mathsf{cca}}$, $\mathcal{B}'_{\mathsf{cca}}$, and $\mathcal{B}_{\mathsf{cu}}$ satisfying Equation 9, as required. □ (**Theorem 2**)

6.2 Instantiations of Projective Hash Function

For our construction Π_{poly}, we use an augmented PHF family for the cascaded ElGamal language. As in the case for Π_{aff}, we have two instantiations: the basic construction $\mathsf{PHF}_{\mathsf{poly}}$ and the space-efficient construction $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{poly}}$ using a collision resistant hash function. Since the construction of $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{poly}}$ from $\mathsf{PHF}_{\mathsf{poly}}$ is exactly the same as the construction of $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ from $\mathsf{PHF}_{\mathsf{aff}}$ given in Section 5, in this subsection we only show the basic construction $\mathsf{PHF}_{\mathsf{poly}} = (\mathsf{Setup}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \mu, \Lambda, \mathsf{Pub})$.

Let $s \geq 2$, GGen be the DCR group generator, and $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$. Recall that for the cascaded ElGamal language, $\Pi_{\mathsf{yes}} = G_n^d$, $\Pi_{\mathsf{no}} = (G_{N^{s-1}} \otimes G_n)^d \setminus G_n^d$, and the auxiliary key space is $\mathcal{AK} = G_n$.

Given param, Setup outputs pp that concretely specifies $(\mathcal{SK}, \mathcal{PK}, \mathcal{K}, \mu, \mathsf{Pub})$ as follows: We define $\mathcal{SK} := \left[N^{s-1} \cdot \frac{N-1}{4}\right]$, $\mathcal{PK} := G_n^2$, and $\mathcal{K} := (G_{N^{s-1}} \times G_n)^d$. For every $\mathsf{sk} \in \left[\frac{N-1}{4}\right]$, $\mathsf{ak} \in G_n$, and $c = (c_1, \ldots, c_d) \in (G_{N^{s-1}} \otimes G_n)^d$, we also define μ and Λ as

$$\mu(\mathsf{ak}, \mathsf{sk}) := \left(g^{\mathsf{sk}} \bmod N^s, \mathsf{ak}^{\mathsf{sk}} \bmod N^s\right) \quad \text{and}$$
$$\Lambda_{\mathsf{sk}}(c) := \left(c_1^{\mathsf{sk}} \bmod N^s, \ldots, c_d^{\mathsf{sk}} \bmod N^s\right) .$$

Projective property. Let $\mathsf{sk} \in \left[\frac{N-1}{4}\right]$, $\mathsf{ak} \in G_n$, $\mathsf{pk} = (y, z) = (g^{\mathsf{sk}} \bmod N^s, \mathsf{ak}^{\mathsf{sk}} \bmod N^s)$, and $c = (c_1, \ldots, c_d)$, where $c_d = g^{r_d} \bmod N^s$, $c_i = g^{r_i} \cdot \mathsf{ak}^{r_{i+1}} \bmod N^s$ for every $i \in [d-1]$, and $r = (r_1, \ldots, r_d) \in \mathbb{Z}^d$. We define the public evaluation algorithm Pub as

$$\mathsf{Pub}(\mathsf{ppk}, c, r) := (y^{r_1} \cdot z^{r_2} \bmod N^s, \ldots, y^{r_{d-1}} \cdot z^{r_d} \bmod N^s, y^{r_d} \bmod N^s) \quad .$$

We see that

$$y^{r_d} \equiv (g^{r_d})^{\mathsf{sk}} \bmod N^s \quad , \quad \text{and}$$
$$y^{r_i} \cdot z^{r_{i+1}} \equiv \left(g^{\mathsf{sk}}\right)^{r_i} \cdot \left(\mathsf{ak}^{\mathsf{sk}}\right)^{r_{i+1}} \equiv (g^{r_i} \cdot \mathsf{ak}^{r_{i+1}})^{\mathsf{sk}} \bmod N^s$$

for every $i \in [d-1]$. Therefore, we have $\mathsf{Pub}(\mathsf{pk}, c, r) = \Lambda_{\mathsf{sk}}(c)$, and thus $\mathsf{PHF}_{\mathsf{poly}}$ satisfies the projective property.

Universal property. We now show that $\mathsf{PHF}_{\mathsf{poly}}$ satisfies the statistical universal property. Let $\mathsf{ak} \in G_n$, $\mathsf{pk} = (y, z) \in \mathsf{Sup}(\mu(\mathsf{ak}, \cdot))$, $c = (c_1, \ldots, c_d) \in (G_{N^{s-1}} \otimes G_n)^d \setminus G_n^d$, where $c_{i^*} \in G_{N^{s-1}} \otimes G_n \setminus G_n$ for some $i^* \in [d]$. (Our analysis below focuses on this i^*, and proceeds identically to the analysis of the universal property for $\mathsf{PHF}_{\mathsf{aff}}$.) Let $\pi = (\pi_1, \ldots, \pi_d) \in (G_{N^{s-1}} \otimes G_n)^d$. We can write $c_{i^*} = T^{\theta} \cdot g^r \bmod N^s$ and $\pi_{i^*} = T^{\theta'} \cdot g^{r'} \bmod N^s$, where $\theta, \theta' \in \left[N^{s-1}\right]$ such that $\theta \neq N^{s-1}$ and $r, r' \in [n]$. Consider the following probability

$$\begin{aligned}
&\Pr_{\mathsf{sk} \xleftarrow{r} [N^{s-1} \cdot n]} [\Lambda_{\mathsf{sk}}(c) = \pi | \mu(\mathsf{ak}, \mathsf{sk}) = \mathsf{pk}] \\
&\leq \Pr_{\mathsf{sk} \xleftarrow{r} [N^{s-1} \cdot n]} \left[c_{i^*}^{\mathsf{sk}} \equiv \pi_{i^*} \bmod N^s \middle| g^{\mathsf{sk}} \equiv y \bmod N^s \wedge \mathsf{ak}^{\mathsf{sk}} \equiv z \bmod N^s \right] \\
&\leq \Pr_{\mathsf{sk} \xleftarrow{r} [N^{s-1} \cdot n]} \left[T^{\theta \cdot \mathsf{sk}} \equiv T^{\theta'} \bmod N^s \middle| \mathsf{sk} \equiv \log_g y \equiv \log_{\mathsf{ak}} z \bmod n \right] \\
&= \Pr_{\mathsf{sk}^* \xleftarrow{r} [N^{s-1}]} \left[\theta \cdot \mathsf{sk}^* \equiv \theta' \bmod N^{s-1} \right] \\
&\leq \Pr_{\mathsf{sk}^* \xleftarrow{r} [P^{s-1}]} \left[\theta \cdot \mathsf{sk}^* \equiv \theta' \bmod P^{s-1} \right] \quad , \qquad (12)
\end{aligned}$$

where the last inequality is true even if P is replaced with Q. Since $\theta \neq N^{s-1}$, we have either $\theta \not\equiv 0 \bmod P^{s-1}$ or $\theta \not\equiv 0 \bmod Q^{s-1}$. Without loss of generality, we assume that $\theta \not\equiv 0 \bmod P^{s-1}$. (Otherwise, switch the role of P and Q.) Below, we prove the probability in Equation 12 is bounded by $O(2^{-\mathsf{len}})$.

First, we consider the case of $\mathsf{GCD}(P^{s-1}, \theta) = 1$.[12] In this case, there exists a multiplicative inverse of $\theta \bmod P^{s-1}$, and Equation 12 is at most $\frac{1}{P^{s-1}} \leq O(2^{-\mathsf{len}})$.

Next, we consider the case of $\mathsf{GCD}(P^{s-1}, \theta) \neq 1$. In this case, there exist $\zeta \in [s-2]$ and θ_0 that is co-prime to P such that $\theta = P^{\zeta} \cdot \theta_0$. Then, if θ' is not a multiple of P^{ζ}, then Equation 12 is 0. Otherwise (i.e. θ' is a multiple of P^{ζ}), Equation 12 is equal to

$$\Pr_{\mathsf{sk}^* \xleftarrow{r} \left[P^{s-\zeta-1}\right]} \left[\theta_0 \cdot \mathsf{sk}^* \equiv \frac{\theta'}{P^{\zeta}} \bmod P^{s-\zeta-1} \right] \quad .$$

Since θ_0 is co-prime to P, there exists a multiplicative inverse of $\theta_0 \bmod P^{s-\zeta-1}$. Then, the above probability is bounded by $\frac{1}{P^{s-\zeta-1}} \leq O(2^{-\mathsf{len}})$.

Therefore, the probability in Equation 12 is bounded by $O(2^{-\mathsf{len}})$ in any case. Moreover, $\mathbf{SD}(\mathsf{U}_{\mathcal{SK}}, \mathsf{U}_{[N^{s-1} \cdot n]}) = \mathbf{SD}\left(\mathsf{U}_{\left[N^{s-1} \cdot \frac{N-1}{4}\right]}, \mathsf{U}_{[N^{s-1} \cdot n]}\right) \leq O\left(2^{-\mathsf{len}}\right)$. Therefore, $\mathsf{PHF}_{\mathsf{poly}}$ is $O(2^{-\mathsf{len}})$-universal.

[12] If $s = 2$, this always holds.

7 Instantiations

We give some instantiation examples of $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure PKE schemes and $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA secure PKE schemes from our proposed schemes Π_{aff} in Section 5 and Π_{poly} in Section 6. These instantiations are summarized in Figures 1 and 2 in Section 1.2. In all of the following instantiations, the plaintext space of the resulting schemes is $\mathbb{Z}_{N^{s-1}}$, where N is the RSA modulus generated in the setup algorithm and $s \geq 3$, and we assume that the underlying SKEM is instantiated with the one presented in Section 4.2.

The first instantiations are obtained by instantiating the underlying PHF family with the "space-efficient" PHF families ($\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{aff}}$ for Π_{aff} and $\mathsf{PHF}^{\mathsf{hash}}_{\mathsf{poly}}$ for Π_{poly}), and the underlying IND-CCA secure PKE scheme with the scheme based on the factoring assumption proposed by Hofheinz and Kiltz [17]. The KDM-CCA security of the resulting PKE schemes is not tightly reduced to the DCR assumption, but a ciphertext of the $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure scheme consists of only two elements of $\mathbb{Z}_{N^s}$, two elements of $\mathbb{Z}_{N'}$ (caused by the Hofheinz-Kiltz scheme), and a hash value output by a collision-resistant hash function, where N' is the RSA modulus generated in the Hofheinz-Kiltz scheme. Note that if $s \geq 3$, the size of two elements of $\mathbb{Z}_{N'}$ plus the size of a hash value is typically (much) smaller than one element of $\mathbb{Z}_{N^s}$! Furthermore, the improvement on the ciphertext size of $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA secure scheme from the previous works is much more drastic. For KDM security with respect to degree-d polynomials, a ciphertext of our instantiation consists of $(d+1)$ elements of $\mathbb{Z}_{N^s}$, two elements of $\mathbb{Z}_{N'}$, and a hash value, and its size overhead compared to Malkin et al.'s scheme [23] is independent of d. In contrast, the ciphertext size of the previous best construction of Han et al. [12] is $O(d^9)$ elements of $\mathbb{Z}_{N^s}$ and more (and in addition its security relies on both the DCR and DDH assumptions).

The second instantiations are PKE schemes obtained by instantiating the underlying PHF family with the "basic" PHF families ($\mathsf{PHF}_{\mathsf{aff}}$ for Π_{aff} and $\mathsf{PHF}_{\mathsf{poly}}$ for Π_{poly}), and the underlying IND-CCA secure PKE scheme with the scheme proposed by Hofheinz [14]. Hofheinz' scheme is tightly IND-CCA secure under the DCR assumption, and its ciphertext overhead is 28 group elements plus the ciphertext overhead caused by authenticated encryption. The advantage of the second instantiations is that we obtain the first tightly $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure PKE scheme and a tightly $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA PKE scheme based solely on the DCR assumption. The disadvantage is the relatively large ciphertext size.

The third instantiations are obtained by replacing the underlying PKE scheme in the second ones with the PKE scheme proposed by Gay, Hofheinz, and Kohl [11]. Gay et al.'s scheme is tightly IND-CCA secure under the DDH assumption, and its ciphertext overhead is just three group elements of a DDH-hard group plus the ciphertext overhead caused by authenticated encryption. By the third instantiations, relying on both the DCR and DDH assumptions, we obtain a tightly $\mathcal{F}_{\mathsf{aff}}$-KDM-CCA secure PKE scheme whose ciphertext consists of essentially only three elements of $\mathbb{Z}_{N^s}$ and three elements of the DDH-hard group. We also obtain a tightly $\mathcal{F}_{\mathsf{poly}}$-KDM-CCA secure PKE scheme with much smaller ciphertexts than our second instantiation achieving the same security.

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

References

[1] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. *CRYPTO 2009*, pp. 595–618.

[2] B. Applebaum, D. Harnik, and Y. Ishai. Semantic security under related-key attacks and applications. *ICS 2011*, pp. 45–60.

[3] J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. *SAC 2002*, pp. 62–75.

[4] F. Böhl, G. T. Davies, and D. Hofheinz. Encryption schemes secure under related-key and key-dependent message attacks. *PKC 2014*, pp. 483–500.

[5] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. *CRYPTO 2008*, pp. 108–125.

[6] Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). *CRYPTO 2010*, pp. 1–20.

[7] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. *EUROCRYPT 2009*, pp. 351–368.

[8] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. *EUROCRYPT 2001*, pp. 93–118.

[9] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. *EUROCRYPT 2002*, pp. 45–64.

[10] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. *PKC 2001*, pp. 119–136.

[11] R. Gay, D. Hofheinz, and L. Kohl. Kurosawa-desmedt meets tight security. *CRYPTO 2017, Part III*, pp. 133–160.

[12] S. Han, S. Liu, and L. Lyu. Efficient KDM-CCA secure public-key encryption for polynomial functions. *ASIACRYPT 2016, Part II*, pp. 307–338.

[13] D. Hofheinz. Circular chosen-ciphertext security with compact ciphertexts. *EUROCRYPT 2013*, pp. 520–536.

[14] D. Hofheinz. Adaptive partitioning. *EUROCRYPT 2017, Part III*, pp. 489–518.

[15] D. Hofheinz and T. Jager. Tightly secure signatures and public-key encryption. *CRYPTO 2012*, pp. 590–607.

[16] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. *CRYPTO 2007*, pp. 553–571.

[17] D. Hofheinz and E. Kiltz. Practical chosen ciphertext secure encryption from factoring. *EUROCRYPT 2009*, pp. 313–332.

[18] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. Extending oblivious transfers efficiently. *CRYPTO 2003*, pp. 145–161.

[19] F. Kitagawa and K. Tanaka. A framework for achieving KDM-CCA secure public-key encryption. *ASIACRYPT 2018, Part II*, pp. 127–157.

[20] K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. *CRYPTO 2004*, pp. 426–442.

[21] B. Libert and C. Qian. Lossy algebraic filters with short tags. *PKC 2019, Part I*, pp. 34–65.

[22] X. Lu, B. Li, and D. Jia. KDM-CCA security from RKA secure authenticated encryption. *EUROCRYPT 2015, Part I*, pp. 559–583.

[23] T. Malkin, I. Teranishi, and M. Yung. Efficient circuit-size independent public key encryption with KDM security. *EUROCRYPT 2011*, pp. 507–526.

[24] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pp. 427–437.

[25] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. *EUROCRYPT'99*, pp. 223–238.

A Compressing Projective Hash Functions

In this section, we formally show that if a statistically universal PHF family satisfies some additional natural property (that is in particular satisfied by our concrete instantiations in Sections 5.2 and 6.2) and we compress its output by using a collision resistant hash function, then the resulting PHF family still satisfies the computational version of the universal property. Although our definitions and formal statements in this section are for PHF families with respect to the DCR group generator GGen, we stress that it is only for concreteness. The compressing technique is applicable to any (augmented) PHF family satisfying a natural "trapdoor" property. (See Remark 5 in the end of this section.)

We first recall the formal definition of a collision resistant hash family.

Definition 9 (Collision resistant hash functions) *Let $\mathcal{H} = \{H : \{0,1\}^* \to R\}$ be a hash family. We say that $\mathcal{H}$ is* collision resistant *if for any PPT adversary $\mathcal{A}$, we have*

$$\mathsf{Adv}^{\mathsf{crhf}}_{\mathcal{H},\mathcal{A}}(\lambda) = \Pr[H(m) = H(m') \wedge m \neq m'] = \mathsf{negl}(\lambda) \ ,$$

where $H \xleftarrow{\mathsf{r}} \mathcal{H}$ and $(m, m') \leftarrow \mathcal{A}(1^\lambda, H)$.

Next, we introduce additional and yet natural properties for a PHF family that allow us to apply the compressing technique.

Definition 10 (Compression-friendliness) *Let $s \geq 2$, GGen be the DCR group generator, and $\mathsf{PHF} = (\mathsf{Setup}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ be a PHF family with respect to GGen. We say that PHF is* compression-friendly *if the following properties are satisfied: Let $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$ and $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{param})$. Then,*

- *(Checkability of yes instances:) Given param, pp, and $c \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$, whether $c \in \Pi_{\mathsf{yes}}$ or not is efficiently checkable.*
- *(Conditional resampling of secret keys:) Given param, pp, $\mathsf{sk} \in \mathcal{SK}$, and $\mathsf{pk} = \mu(\mathsf{sk})$, it is possible to efficiently sample a fresh secret key $\widetilde{\mathsf{sk}} \in \mathcal{SK}$ according to a distribution that is $O(2^{-\lambda})$-close to the uniform distribution over the subset $\mathcal{SK}|_{\mathsf{pk}} := \{\mathsf{sk}' \in \mathcal{SK} | \mu(\mathsf{sk}') = \mathsf{pk}\}$.*

It is straightforward to see that the concrete instantiation of the PHF family for the DCR language in Section 5.2 and that for the cascaded ElGamal language in Section 6.2 are compression-friendly.

To see that the former PHF family is compression-friendly, recall that $\Pi_{\mathsf{yes}} = G_n$ and $\Pi_{\mathsf{no}} = G_{N^{s-1}} \otimes G_n \setminus G_n$. Thus, given $\mathsf{param} = (N, P, Q, T, g)$ and an instance $c \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$, one can efficiently check the membership of Π_{yes} using P and Q. Furthermore, given param, $\mathsf{sk} \in \left[N^s \cdot \frac{N-1}{4}\right]$, and $\mathsf{pk} = \mu(\mathsf{sk}) = g^{\mathsf{sk}} \bmod N^s$, we can sample a fresh secret key $\widetilde{\mathsf{sk}} \in \mathcal{SK}$ by first sampling $\theta \xleftarrow{\mathsf{r}} [N^{s-1}]$ and then letting $\widetilde{\mathsf{sk}}$ be the element satisfying $\widetilde{\mathsf{sk}} \equiv \mathsf{sk} \bmod n$ and $\widetilde{\mathsf{sk}} \equiv \theta \bmod N^{s-1}$. (Note that such element is uniquely determined by the Chinese remainder theorem.) Since $g \in G_n$, it is immediate to see that

$$\mu(\widetilde{\mathsf{sk}}) = g^{\widetilde{\mathsf{sk}}} \bmod N^s = g^{\widetilde{\mathsf{sk}} \bmod n} \bmod N^s = g^{\mathsf{sk} \bmod n} \bmod N^s = g^{\mathsf{sk}} \bmod N^s = \mu(\mathsf{sk}) \ ,$$

and the statistical distance between the distribution of $\widetilde{\mathsf{sk}}$ sampled as above and the uniform distribution over $\mathcal{SK}|_{\mathsf{pk}} = \left\{\mathsf{sk}' \in \left[N^{s-1} \cdot \frac{N-1}{4}\right] | g^{\mathsf{sk}'} \bmod N^s = \mathsf{pk}\right\}$, is bounded by $O(2^{-\mathsf{len}}) \leq O(2^{-\lambda})$.

The fact that the PHF family for the cascaded ElGamal language in Section 6.2 is compression-friendly can be seen analogously, and thus we omit the detail.

We now proceed to formally showing the compressing technique. Let $s \geq 2$, GGen be the DCR group generator, and $\mathsf{PHF} = (\mathsf{Setup}, \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}, \Lambda, \mu, \mathsf{Pub})$ be a PHF family with respect to GGen. Let $\mathcal{H} = \left\{H : \{0,1\}^* \to \{0,1\}^{\mathsf{len}_{\mathsf{crhf}}}\right\}$ be a hash family for some polynomial $\mathsf{len}_{\mathsf{crhf}} = \mathsf{len}_{\mathsf{crhf}}(\lambda)$. Consider the "compressed"-version of a PHF family $\mathsf{PHF}^{\mathsf{hash}} = (\mathsf{Setup}', \Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}}, \mathcal{SK}, \mathcal{PK}, \mathcal{K}' := \{0,1\}^{\mathsf{len}_{\mathsf{crhf}}}, \Lambda', \mu, \mathsf{Pub}')$, in which Setup', Λ', and Pub' are defined as follows (and the remaining components are unchanged between $\mathsf{PHF}^{\mathsf{hash}}$ and PHF):

- $\mathsf{Setup}'(\mathsf{param})$ runs $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{param})$, picks $H \xleftarrow{\mathsf{r}} \mathcal{H}$, and outputs $\mathsf{pp}' := (\mathsf{pp}, H)$;
- For $\mathsf{pp}' = (\mathsf{pp}, H)$, $\mathsf{sk} \in \mathcal{SK}$, and $c \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$, we define $\Lambda'_{\mathsf{sk}}(c) := H(\Lambda_{\mathsf{sk}}(c))$;
- Similarly to Λ', Pub' is defined by composing Pub and H straightforwardly.

It is obvious that $\mathsf{PHF}^{\mathsf{hash}}$ preserves the projective property of the underlying PHF. We now formally show that the "compressed" PHF $\mathsf{PHF}^{\mathsf{hash}}$ satisfies the computational universal property if the underlying PHF is compression-friendly and statistically universal, and $\mathcal{H}$ is collision resistant.

Lemma 6 *If* PHF *is compression-friendly and* ϵ*-universal for some negligible* $\epsilon = \epsilon(\lambda)$*, and* $\mathcal{H}$ *is collision resistant, then* $\mathsf{PHF}^{\mathsf{hash}}$ *is computationally universal.*

Specifically, for any PPT adversary $\mathcal{A}$ *that attacks the computational universal property of* $\mathsf{PHF}^{\mathsf{hash}}$*, there exists a PPT adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF}^{\mathsf{hash}},\mathcal{A}}(\lambda) \leq \sqrt{\mathsf{Adv}^{\mathsf{crhf}}_{\mathcal{H},\mathcal{B}}(\lambda) + \epsilon + O(2^{-\lambda})} \ . \tag{13}$$

Proof of Lemma 6. Let $\mathcal{A}$ be any PPT adversary that attacks the computational universal property of $\mathsf{PHF}^{\mathsf{hash}}$. Suppose the computational universal game of $\mathsf{PHF}^{\mathsf{hash}}$ is played by $\mathcal{A}$ and the challenger, and $\mathcal{A}$ finally outputs $(c^*, \pi^*) \in \Pi_{\mathsf{no}} \times \{0,1\}^{\mathsf{len}_{\mathsf{crhf}}}$. We denote this process by "$(c^*, \pi^*) \leftarrow \mathsf{G}^{\mathsf{cu}}$". From here on, the values without any mention (such as pk, sk, and H) are those generated in the game. We then let the challenger choose two fresh keys $\widetilde{\mathsf{sk}}_1$ and $\widetilde{\mathsf{sk}}_2$ uniformly and independently from $\mathcal{SK}|_{\mathsf{pk}}$. Then, consider the following events:

$\mathtt{SUC}_0$: $H(\Lambda_{\mathsf{sk}}(c^*)) = \pi^*$ holds. (That is, $\mathcal{A}$ wins the computational universal game.)

$\mathtt{SUC}_1$: $H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^*$ holds.

$\mathtt{SUC}_2$: $H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^*$ and $H(\Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)) = \pi^*$ hold simultaneously.

$\mathtt{COL}$: $\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*) = \Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)$ holds.

Unless mentioned otherwise, the probabilities in the following are over the process "$(c^*, \pi^*) \leftarrow \mathsf{G}^{\mathsf{cu}}$; $\widetilde{\mathsf{sk}}_1, \widetilde{\mathsf{sk}}_2 \xleftarrow{\mathsf{r}} \mathcal{SK}|_{\mathsf{pk}}$".

By definition, we have $\mathsf{Adv}^{\mathsf{cu}}_{\mathsf{PHF}^{\mathsf{hash}},\mathcal{A}}(\lambda) = \Pr[\mathtt{SUC}_0]$. To complete the proof, we will show the following four items:

- $\Pr[\mathtt{SUC}_0] = \Pr[\mathtt{SUC}_1]$.
- $(\Pr[\mathtt{SUC}_1])^2 \leq \Pr[\mathtt{SUC}_2]$.
- $\Pr[\mathtt{COL}] \leq \epsilon$.
- There exists a PPT adversary $\mathcal{B}$ satisfying $\mathsf{Adv}^{\mathsf{crhf}}_{\mathcal{H},\mathcal{B}}(\lambda) \geq \Pr[\mathtt{SUC}_2] - \Pr[\mathtt{COL}] - O(2^{-\lambda})$.

Note that the above (in)equalities imply Equation 13, and thus complete the proof.

Firstly, it is immediate to see that $\Pr[\mathtt{SUC}_0] = \Pr[\mathtt{SUC}_1]$ holds, because the distributions of sk and $\widetilde{\mathsf{sk}}_1$ are identical.

Secondly, $(\Pr[\mathtt{SUC}_1])^2 \leq \Pr[\mathtt{SUC}_2]$ can be seen as follows:

$$
\begin{aligned}
\Pr[\mathtt{SUC}_2] &= \Pr_{(c^*,\pi^*)\leftarrow\mathsf{G}^{\mathsf{cu}};\ \widetilde{\mathsf{sk}}_1,\widetilde{\mathsf{sk}}_2\xleftarrow{\mathsf{r}}\mathcal{SK}|_{\mathsf{pk}}}\left[\ H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^* \wedge H(\Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)) = \pi^*\ \right] \\
&= \mathop{\mathbf{E}}_{(c^*,\pi^*)\leftarrow\mathsf{G}^{\mathsf{cu}}}\left[\ \Pr_{\widetilde{\mathsf{sk}}_1,\widetilde{\mathsf{sk}}_2\xleftarrow{\mathsf{r}}\mathcal{SK}|_{\mathsf{pk}}}\left[\ H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^* \wedge H(\Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)) = \pi^*\ \right]\ \right] \\
&\overset{(*)}{=} \mathop{\mathbf{E}}_{(c^*,\pi^*)\leftarrow\mathsf{G}^{\mathsf{cu}}}\left[\ \left(\Pr_{\widetilde{\mathsf{sk}}_1\xleftarrow{\mathsf{r}}\mathcal{SK}|_{\mathsf{pk}}}\left[\ H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^*\ \right]\right)^2\ \right] \\
&\overset{(\dagger)}{\geq} \left(\ \mathop{\mathbf{E}}_{(c^*,\pi^*)\leftarrow\mathsf{G}^{\mathsf{cu}}}\left[\ \Pr_{\widetilde{\mathsf{sk}}_1\xleftarrow{\mathsf{r}}\mathcal{SK}|_{\mathsf{pk}}}\left[\ H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^*\ \right]\ \right]\ \right)^2 \\
&= \left(\ \Pr_{(c^*,\pi^*)\leftarrow\mathsf{G}^{\mathsf{cu}};\ \mathsf{sk}_1\xleftarrow{\mathsf{r}}\mathcal{SK}|_{\mathsf{pk}}}\left[\ H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^*\ \right]\ \right)^2 = (\Pr[\mathtt{SUC}_1])^2\ ,
\end{aligned}
$$

where the inequality (*) is because the events $H(\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)) = \pi^*$ and $H(\Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)) = \pi^*$ become independent once (c^*, π^*) and H are fixed, and $\widetilde{\mathsf{sk}}_1$ and $\widetilde{\mathsf{sk}}_2$ are distributed identically; The inequality (†) is due to the Jensen inequality[13].

Thirdly, it is also immediate to see that $\Pr[\mathtt{COL}] \leq \epsilon$ holds due to the ϵ-universal property of the underlying PHF. Indeed, we can regard $\widetilde{\mathsf{sk}}_1$ as the sk in the probability defining the universal property, then the ϵ-universal property ensures that the probability that $\Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)$ hits $\Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)$ is at most ϵ, regardless of the value of $\widetilde{\mathsf{sk}}_2$.

Finally, we show that there exists a PPT adversary $\mathcal{B}$ against the collision resistance of the hash family $\mathcal{H}$ satisfying $\mathsf{Adv}^{\mathsf{crhf}}_{\mathcal{H},\mathcal{B}}(\lambda) \geq \Pr[\mathtt{SUC}_2] - \Pr[\mathtt{COL}] - O(2^{-\lambda})$. To show this, we use the assumption that PHF is compression-friendly. The description of $\mathcal{B}$ is as follows:

[13] $\mathbf{E}[f(X)] \geq f(\mathbf{E}[X])$ holds for any convex function f.

- Given 1^λ and $H \in \mathcal{H}$ from the collision-resistance challenger, $\mathcal{B}$ generates $\mathsf{param} = (N, P, Q, T, g) \leftarrow \mathsf{GGen}(1^\lambda, s)$ and $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{phf}}(\mathsf{param})$, and sets $\mathsf{pp}' \leftarrow (\mathsf{pp}, H)$. $\mathcal{B}$ then chooses $\mathsf{sk} \xleftarrow{\mathsf{r}} \mathcal{SK}$, and computes $\mathsf{pk} \leftarrow \mu(\mathsf{sk})$. Then, $\mathcal{B}$ gives $(N, T, g, \mathsf{pp}', \mathsf{pk})$ to $\mathcal{A}$.
- For the evaluation queries $c \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$ from $\mathcal{A}$, $\mathcal{B}$ checks whether c is an yes instance by using param and pp, which is possible due to the "checkability of yes instances" property of the compression-friendliness of PHF. If $c \in \Pi_{\mathsf{yes}}$, $\mathcal{B}$ computes $\pi \leftarrow H(\Lambda_{\mathsf{sk}}(c))$ and returns it to $\mathcal{A}$. Otherwise (i.e. $c \in \Pi_{\mathsf{no}}$), $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$.
- When $\mathcal{A}$ outputs $(c^*, \pi^*) \in \Pi_{\mathsf{no}} \times \{0,1\}^{\mathsf{len}_{\mathsf{crhf}}}$, $\mathcal{B}$ picks two fresh keys $\widetilde{\mathsf{sk}}_1, \widetilde{\mathsf{sk}}_2 \in \mathcal{SK}|_{\mathsf{pk}}$ using param and pp, with the method guaranteed by the "conditional resampling of secret keys" property of the compression-friendliness of PHF. (Note that the distributions of $\widetilde{\mathsf{sk}}_1$ and $\widetilde{\mathsf{sk}}_2$ are $O(2^{-\lambda})$-close to the uniform distribution over $\mathcal{SK}|_{\mathsf{pk}}$.) Then, $\mathcal{B}$ computes $\widetilde{\pi}_1 \leftarrow \Lambda_{\widetilde{\mathsf{sk}}_1}(c^*)$ and $\widetilde{\pi}_2 \leftarrow \Lambda_{\widetilde{\mathsf{sk}}_2}(c^*)$. Finally, $\mathcal{B}$ outputs $(\widetilde{\pi}_1, \widetilde{\pi}_2)$ as a candidate of a collision pair for H, and terminates.

$\mathcal{B}$'s collision resistance advantage can be estimated as follows (where the notation "$\Pr_{\mathcal{B}}$" is to make it explicit that the probability is over $\mathcal{B}$'s collision resistance game):

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{crhf}}_{\mathcal{H},\mathcal{B}}(\lambda) &= \Pr_{\mathcal{B}}[H(\widetilde{\pi}_1) = H(\widetilde{\pi}_2) \wedge \widetilde{\pi}_1 \neq \widetilde{\pi}_2] \\
&= \Pr_{\mathcal{B}}[H(\widetilde{\pi}_1) = H(\widetilde{\pi}_2)] - \Pr_{\mathcal{B}}[\widetilde{\pi}_1 = \widetilde{\pi}_2] \\
&\geq \Pr_{\mathcal{B}}[H(\widetilde{\pi}_1) = \pi^* \wedge H(\widetilde{\pi}_2) = \pi^*] - \Pr_{\mathcal{B}}[\widetilde{\pi}_1 = \widetilde{\pi}_2] \ .
\end{aligned}$$

It is straightforward to see that due to the compression-friendliness of PHF, $\mathcal{B}$ perfectly simulates the computational universal game for $\mathcal{A}$, and thus (c^*, π^*) output by $\mathcal{A}$ in the game simulated by $\mathcal{B}$ is identically distributed to that generated as $(c^*, \pi^*) \leftarrow \mathsf{G}^{\mathsf{cu}}$. Also, the distributions of $\widetilde{\mathsf{sk}}_1$ and $\widetilde{\mathsf{sk}}_2$ are $O(2^{-\lambda})$-close to the uniform distribution over $\mathcal{SK}|_{\mathsf{pk}}$. Thus, we have

$$\left|\Pr_{\mathcal{B}}[H(\widetilde{\pi}_1) = \pi^* \wedge H(\widetilde{\pi}_2) = \pi^*] - \Pr[\mathtt{SUC}_2]\right| \leq O(2^{-\lambda}) \ , \quad \text{and}$$

$$\left|\Pr_{\mathcal{B}}[\widetilde{\pi}_1 = \widetilde{\pi}_2] - \Pr[\mathtt{COL}]\right| \leq O(2^{-\lambda}) \ .$$

Hence, we finally obtain $\mathsf{Adv}^{\mathsf{crhf}}_{\mathcal{H},\mathcal{B}}(\lambda) \geq \Pr[\mathtt{SUC}_2] - \Pr[\mathtt{COL}] - O(2^{-\lambda})$, as desired. □ (**Lemma 6**)

Remark 5 (On the generality of the compressing technique) Note that in the proof of Lemma 6, we did not use a specific property of the DCR group generator GGen. Hence, it is straightforward to see that the compressing technique is applicable to PHFs for other languages (not necessarily associated with the DCR group generator GGen), as long as the setup algorithm of a PHF family, in addition to pp, outputs a trapdoor that allows the owner to (1) check whether or not a given instance c is an yes instance (e.g. P and Q in the case of the PHF family for the DCR language), and (2) given additionally sk and $\mathsf{pk} = \mu(\mathsf{sk})$, sample a fresh secret key $\widetilde{\mathsf{sk}}$ conditioned on $\mu(\widetilde{\mathsf{sk}}) = \mathsf{pk}$ from a distribution that is statistically close to the uniform distribution over the subset $\mathcal{SK}|_{\mathsf{pk}}$. If a PHF family supports such a trapdoor, we can conduct essentially the same proof as our proof of Lemma 6.

B Proofs of Lemmas 2 and 3

Proof of Lemma 2. We proceed the proof via a sequence of games. For every $t \in \{0, \ldots, 5\}$, let $\mathtt{SUC}_t$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game t.

Game 0: This is the original IV_1 game. Then, we have $\mathsf{Adv}^{\mathsf{IV}}_{s,\mathcal{A},1}(\lambda) = 2 \cdot \left|\Pr[\mathtt{SUC}_0] - \frac{1}{2}\right|$.

Game 1: Same as Game 0, except that the challenger generates $r \xleftarrow{\mathsf{r}} [n]$ instead of $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ when responding to a sample query.

$|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| \leq \frac{q_{\mathsf{iv}}(P+Q-1)}{N-2}$ holds since the distribution of r in Game 0 is $\frac{(P+Q-1)}{N-2}$-close to that in Game 0 and $\mathcal{A}$ makes at most q_{iv} queries.

Game 2: Same as Game 1, except that the challenger generates $w \xleftarrow{\mathsf{r}} [n]$ at the beginning of the game and when responding to a sample query a made by $\mathcal{A}$, the challenger generates $r \xleftarrow{\mathsf{r}} [n]$ and returns $e = T^{b \cdot a} \cdot g_1^{wr} \bmod N^s$.

If w is co-prime to n, then $wr \bmod n$ is distributed uniformly over $[n]$. Thus, we can bound $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]|$ by the probability that w is not co-prime to n. Therefore, we have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \leq \frac{p+q-1}{n-2}$.

Game 3: Same as Game 2, except that the challenger generates $r \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot \frac{N-1}{4}\right]$ instead of $r \xleftarrow{\mathsf{r}} [n]$ when responding to a sample query.

The only information of r that $\mathcal{A}$ can obtain is $r \bmod n$ through $g_1^{wr} \bmod N^s$. The distribution of $r \bmod n$ in Game 2 is $\frac{(P+Q-1)}{N-2}$-close to that in Game 3 and $\mathcal{A}$ makes at most q_{iv} queries. Therefore, we obtain $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq \frac{q_{\mathsf{iv}}(P+Q-1)}{N-2}$.

Game 4: Same as Game 3, except that when $\mathcal{A}$ makes a sample query $a \in \mathbb{Z}_{N^{s-1}}$, the challenger returns $T^{b \cdot a} \cdot (T \cdot g_1^w)^r \bmod N^s$ instead of $T^{b \cdot a} \cdot g_1^{wr} \bmod N^s$.

By the DCR assumption, the distributions $g_1^w \bmod N^s$ and $T \cdot g_1^w \bmod N^s$ are computationally indistinguishable, where $w \xleftarrow{\mathsf{r}} [n]$. In other words, there exists a PPT adversary $\mathcal{B}$ that satisfies $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| = \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda)$.

Game 5: Same as Game 4, except that the challenger generates $r \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot n\right]$ instead of $r \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot \frac{N-1}{4}\right]$ when responding to a sample query.

$|\Pr[\mathtt{SUC}_4] - \Pr[\mathtt{SUC}_5]| \leq \frac{q_{\mathsf{iv}}(P+Q-1)}{N-2}$ holds since the distribution of r in Game 5 is $\frac{(P+Q-1)}{N-2}$-close to that in Game 4 and $\mathcal{A}$ makes at most q_{iv} queries.

In Game 5, the answer to a sample query a made by $\mathcal{A}$ is

$$e = T^{b \cdot a} \cdot (T \cdot g_1^w)^r \bmod N^s = T^{b \cdot a + (r \bmod N^{s-1})} \cdot g_1^{wr \bmod n} \bmod N^s \ ,$$

where $r \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot n\right]$. Therefore, by $(r \bmod N^{s-1})$, the value of b is information-theoretically hidden from the view of $\mathcal{A}$ in Game 5. Thus, we have $\left|\Pr[\mathtt{SUC}_5] - \frac{1}{2}\right| = 0$.

From the above arguments, there exists a PPT adversary $\mathcal{B}$ that satisfies

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{iv}}_{s,1,\mathcal{A}}(\lambda) &\leq 2 \cdot \left(\mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{3q_{\mathsf{iv}}(P+Q-1)}{N-2} + \frac{(p+q-1)}{n-2} \right) \\
&\leq 2 \cdot \left(\mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{(6q_{\mathsf{iv}}+4)}{2^{\mathsf{len}}} \right) \\
&= 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{O\left(q_{\mathsf{iv}}\right)}{2^{\mathsf{len}}} \ .
\end{aligned}$$

□ (**Lemma 2**)

Proof of Lemma 3. We proceed the proof via a sequence of games. For every $t \in \{0, \ldots, 4\}$, let $\mathtt{SUC}_t$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game t.

Game 0: This is the original $\mathsf{IV}_{s,\ell}$ game. Then, we have $\mathsf{Adv}^{\mathsf{IV}}_{s,\ell,\mathcal{A}}(\lambda) = 2 \cdot \left|\Pr[\mathtt{SUC}_0] - \frac{1}{2}\right|$.

Game 1: Same as Game 1, except that the challenger generates $\alpha_i \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot \frac{N-1}{4}\right]$ instead of $\alpha_i \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$ for every $i \in [\ell]$.

$|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| \leq \frac{2\ell(P+Q-1)}{N-2}$ holds since the distribution of g_i in Game 0 is $\frac{2(P+Q-1)}{N-2}$-close to that in Game 1 for every $i \in [\ell]$.

Game 2: Same as Game 1, except that the challenger generates $r \xleftarrow{\mathsf{r}} [n]$ instead of $r \xleftarrow{\mathsf{r}} \left[\frac{N-1}{4}\right]$.

$|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \leq \frac{P+Q-1}{N-2}$ holds since the distribution of r in Game 1 is $\frac{P+Q-1}{N-2}$-close to that in Game 2 and $\mathcal{A}$ make at most one sample query.

Game 3: Same as Game 2, except that when $\mathcal{A}$ makes a sample query $(a_1, \ldots, a_\ell) \in \mathbb{Z}^\ell_{N^{s-1}}$, the challenger compute e_i by $T^{b \cdot a_i} \cdot (T \cdot g^r)^{\alpha_i} \bmod N^s$ instead of $T^{b \cdot a_i} \cdot g_i^r \bmod N^s = T^{b \cdot a_i} \cdot (g^r)^{\alpha_i} \bmod N^s$.

By the DCR assumption, the distributions $g^r \bmod N^s$ and $T \cdot g^r \bmod N^s$ are computationally indistinguishable, where $r \xleftarrow{\mathsf{r}} [n]$. In other words, there exists a PPT adversary $\mathcal{B}$ that satisfies $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| = \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda)$.

Game 4: Same as Game 3, except that the challenger generates $\alpha_i \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot n\right]$ instead of $\alpha_i \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot \frac{N-1}{4}\right]$ for every $i \in [\ell]$.

$|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \frac{\ell(P+Q-1)}{N-2}$ holds since the distribution of α_i in Game 3 is $\frac{(P+Q-1)}{N-2}$-close to that in Game 4 for every $i \in [\ell]$.

In Game 4, the answer to a sample query $(a_1, \ldots, a_\ell)$ made by $\mathcal{A}$ is $(e_1, \ldots, e_d)$, where

$$e_i = T^{b \cdot a_i} \cdot (T \cdot g^r)^{\alpha_i} \bmod N^s = T^{b \cdot a_i + (\alpha_i \bmod N^{s-1})} \cdot g^{r(\alpha_i \bmod n)} \bmod N^s \ ,$$

where $\alpha_i \xleftarrow{\mathsf{r}} \left[N^{s-1} \cdot n\right]$. At the beginning of the game, $\mathcal{A}$ is given $g_i = g^{\alpha_i} \bmod N^s = g^{\alpha_i \bmod n} \bmod N^s$ for every $i \in [\ell]$. On the other hand, for every $i \in [\ell]$, the information of $\alpha_i \bmod N^{s-1}$ does not appear except for e_i throughout the game. Therefore, by $(\alpha_i \bmod N^{s-1})$ for every $i \in [\ell]$, the value of b is information-theoretically hidden from the view of $\mathcal{A}$ in Game 4. Thus, we have $\left|\Pr[\mathtt{SUC}_4] - \frac{1}{2}\right| = 0$.

From the above arguments, there exists a PPT adversary $\mathcal{B}$ that satisfies

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{iv}}_{s,\ell,\mathcal{A}}(\lambda) &\leq 2 \cdot \left(\mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{(3\ell+1)(P+Q-1)}{N-2}\right) \\
&\leq 2 \cdot \left(\mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{(6\ell+2)}{2^{\mathsf{len}}}\right) \\
&= 2 \cdot \mathsf{Adv}^{\mathsf{dcr}}_{s,\mathcal{B}}(\lambda) + \frac{O(\ell)}{2^{\mathsf{len}}} \ .
\end{aligned}$$

□ (**Lemma 3**)

C Other Instantiations of SKEM

Here, we explain other instantiations of an SKEM satisfying passive RKA security.

Instantiation from the DDH assumption. As mentioned in Section 4.2, the symmetric-key version of the ElGamal KEM yields an SKEM satisfying our passive RKA security notion. Specifically, let $\mathbb{G}$ be a cyclic group with prime order $p = \Omega(2^\lambda)$. Then, the construction of the DDH-based SKEM is as follows:

$\mathsf{Setup}(1^\lambda)$: Pick $g \xleftarrow{\mathsf{r}} \mathbb{G}$, and return $\mathsf{pp} := (\mathbb{G}, p, g)$ and $z = \widetilde{z} := p$. The session key space is $\mathbb{G}$.

$\mathsf{Encap}(\mathsf{pp}, \mathsf{sk} \in \mathbb{Z})$: Pick $\mathsf{ct} \xleftarrow{\mathsf{r}} \mathbb{G}$, compute $\mathsf{K} = \mathsf{ct}^{\mathsf{sk}}$, and return $(\mathsf{ct}, \mathsf{K})$.

$\mathsf{Decap}(\mathsf{pp}, \mathsf{sk} \in \mathbb{Z}, \mathsf{ct})$: Return $\mathsf{K} = \mathsf{ct}^{\mathsf{sk}}$.

It is immediate to see that this SKEM satisfies the functional requirements. Specifically, we have $\mathsf{ct}^{\mathsf{sk}} = \mathsf{ct}^{\mathsf{sk} \bmod p}$ for any $\mathsf{sk} \in \mathbb{Z}$, and thus the modular-reduction of sk is naturally performed in the exponent, and the correctness follows from that of the ElGamal KEM.

It is also straightforward to see that this SKEM is passively RKA secure under the DDH assumption. Specifically, fix any polynomial $\ell = \ell(\lambda)$ and a PPT adversary $\mathcal{A}$, and suppose $\mathcal{A}$ chooses the bound $B \geq p$. Let $g \xleftarrow{\mathsf{r}} \mathbb{G}$, $\Delta_1, \ldots, \Delta_\ell \xleftarrow{\mathsf{r}} [B]$, $\mathsf{sk} \xleftarrow{\mathsf{r}} \mathbb{Z}_p$, and $\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell \xleftarrow{\mathsf{r}} \mathbb{G}$. Then, $\mathcal{A}$'s view in the passive RKA security game with $b = 1$ is distributed as $\{(\Delta_k, \mathsf{ct}_k, \mathsf{K}_k = \mathsf{ct}_k^{\mathsf{sk}+\Delta_k})_{k\in[\ell]}\}$, which is computationally indistinguishable from $\{R_1, \ldots, R_\ell \xleftarrow{\mathsf{r}} \mathbb{G} : (\Delta_k, \mathsf{ct}_k, \mathsf{K}_k = R_k \cdot \mathsf{ct}_k^{\Delta_k})_{k\in[\ell]}\}$ due to the DDH assumption. (This step can be tightly reduced to the DDH assumption due to its random self-reducibility.) The latter distribution is identical to $\{\mathsf{K}_1, \ldots, \mathsf{K}_\ell \xleftarrow{\mathsf{r}} \mathbb{G} : (\Delta_k, \mathsf{ct}_k, \mathsf{K}_k)_{k\in[\ell]}\}$, and is exactly the distribution of $\mathcal{A}$'s view in the passive RKA security game with $b = 0$.

Instantiations with "empty ciphertext" from hash functions. If we assume an appropriate form of "correlation robustness" [18, 2], we can obtain even simpler instantiations with "empty" ciphertext. (As noted in Section 4.1, such a construction is not excluded.)

Specifically, given a hash function $H : \mathbb{Z}_z \to \mathcal{K}$ for some $z = \Omega(2^\lambda)$, consider an SKEM whose encapsulation and decapsulation algorithms both take $\mathsf{sk} \in \mathbb{Z}$ as input, and outputs $\mathsf{K} = H(\mathsf{sk} \bmod z)$.

If H satisfies the property that for any polynomial $\ell = \ell(\lambda)$ and any $B \geq z$ (that could be chosen maliciously by a PPT adversary), the distribution $\{\mathsf{sk} \xleftarrow{\mathsf{r}} \mathbb{Z}_z; \Delta_1, \ldots, \Delta_\ell \xleftarrow{\mathsf{r}} [B] : (\Delta_k, H(\mathsf{sk}+\Delta_k \bmod z))_{k\in[\ell]}\}$ is computationally indistinguishable from the uniform distribution over $([B] \times \mathcal{K})^\ell$, then it directly implies that the above "empty-ciphertext" construction is a passively RKA secure SKEM. This assumption on H is essentially the correlation robustness [18, 2][14], with the slight difference that we consider a slightly stronger adversarial behavior in the sense that an adversary has the power to specify the bound $B \geq z$ of the interval from which the "shifts" $\{\Delta_k\}$ are chosen.

As observed in [18], it is reasonable to heuristically assume that practical cryptographic hash functions such as SHA-256 and SHA-3, satisfy this property. Moreover, Applebaum et al. [2] showed that a hash function $H : \mathbb{Z}_p \to \mathbb{G}$ defined by $H(x) := g^x$ (where $\mathbb{G}$ is a cyclic group with prime order p and $g \in \mathbb{G}$ is a fixed group element) can be shown to be correlation robust under the *ℓ-power DH assumption* in the underlying group $\mathbb{G}$, which states that $\{\alpha \xleftarrow{\mathsf{r}} \mathbb{Z}_p : (g^{\alpha^k})_{k\in[\ell]}\}$

[14] Correlation robustness was first introduced in [18] for a function with one-bit output, where the authors of [18] considered not "addition mod z" but "XOR of bitstrings" for the correlations of inputs. The notion was generalized in [2] for a hash function with a more general form, and a general linear relation for inputs.

is computationally indistinguishable from the uniform distribution over $\mathbb{G}^\ell$. It is immediate to see that their analysis carries over to the case in which an adversary can specify the bound $B \geq z := p$ of the interval from which the "shifts" $\{\Delta_k\}$ are chosen.

Finally, we remark that we can consider a minor variant of the above construction where H is not a fixed hash function but chosen from a family of hash functions and put in a public parameter. This "hash family" version of SKEM can be similarly shown to satisfy passive RKA security from a hash-family version of correlation robustness.

Instantiations from RKA secure encryption schemes. Observe that in general, a KEM is a simpler primitive than an encryption scheme, because the former can always be instantiated by encrypting a random value. Although our syntactical and functional requirements exclude constructions whose secret key is a vector of elements (such as the constructions based on the learning with errors (LWE) and learning parities with noise (LPN) assumptions in [2]), from the existing encryption schemes satisfying passive RKA security with respect to addition or any stronger form of RKA security, we can also obtain concrete instantiations of an SKEM. In particular, this "encrypting a random session-key" approach allows us to obtain RKA secure SKEMs from the DDH assumption [2], the DCR assumption [4], and a correlation robust hash function [2], to name a few.

However, constructing SKEMs in this way is typically redundant and/or overkill, in the sense that the existing RKA secure encryption schemes already implicitly contains an SKEM as its internal structure. In particular, in retrospect, our DCR-based SKEM in Section 4.2 can be understood as obtained from the KEM-part of the encryption scheme of Böhl et al. [4] (which is in turn based on the cascaded ElGamal scheme of Malkin et al. [23] used as symmetric encryption), where we additionally apply a universal hash function to the message-masking component and regard it as a session-key; the DDH-based and hash-based SKEMs can be seen to be obtained by regarding the "message-masking" component in the RKA secure encryption schemes in [2] as a session-key. Furthermore, other instantiations obtained in this way do not have a merit (in terms of efficiency or assumption) compared to the DCR/DDH/hash-based instantiations in Section 4.2 and in this section. Thus, we do not further elaborate this direction.