Two-message Key Exchange with Strong Security from Ideal Lattices*

Zheng Yang[1], Yu Chen[2], and Song Luo[3]

[1] Department of Computer Science, University of Helsinki, Helsinki 00014, Finland
zheng.yang@rub.de
[2] State Key Laboratory of Information Security, Chinese Academy of Sciences, Beijing 100190, China
chenyu@iie.ac.cn
[3] School of Computer Science and Engineering, Chongqing University of Technology, Chongqing 400054, China
ratio@cqut.edu.cn

Abstract. In this paper, we first revisit the generic two-message key exchange (TMKE) scheme (which will be referred to as KF) introduced by Kurosawa and Furukawa (CT-RSA 2014). This protocol is mainly based on key encapsulation mechanism (KEM) which is assumed to be secure against chosen plaintext attacks (IND-CPA). However, we find out that the security of the KF protocol cannot be reduced to IND-CPA KEM. The concrete KF protocol instantiated from ElGamal KEM is even subject to key compromise impersonation (KCI) attacks. In order to overcome the flaws of the KF scheme, we introduce a new generic TMKE scheme from KEM. Instead, we require that the KEM should be secure against one-time adaptive chosen ciphertext attacks (OT-IND-CCA2). We call this class of KEM as OTKEM. In particular, we propose a new instantiation of OTKEM from Ring Learning with Errors (Ring-LWE) problem in the standard model. This yields a concrete post-quantum TMKE protocol with strong security. The security of our TMKE scheme is shown in the extended Canetti-Krawczyk model with perfect forward secrecy (eCK-PFS).

Keywords: KCI attack, two-message key exchange, standard model, lattice, ring-lwe

1 Introduction

Two-message key exchange (TMKE) stands for a special class of round-efficient protocols which allow two principles to agree on a shared key with only two protocol messages at all. If a TMKE protocol is secure against active adversaries, it is also categorized as an authenticated key exchange (AKE) protocol. Besides the communication efficiency, TMKE has some distinct properties that multi-pass key exchange protocols cannot provide. One prominent example of them is that a TMKE protocol can be used to provide security for asynchronous message systems. Note that in a TMKE session the participants may be not online simultaneously (in contrast to the multi-pass key exchange). A party (say Alice) could pre-generate her protocol messages and store them on a message server. Whenever another party (say Bob) wants to communicate with Alice (e.g., sending off-line messages), he can retrieve one of Alice's unused protocol messages from the server to generate his own protocol message based on Alice's message and complete the key exchange procedure.

Actually TMKE has a long research history that can be dated back to the seminal Diffie-Hellman key exchange [16]. A lot of famous Diffie-Hellman like protocols, such as HMQV [23] and NAXOS [25], are TMKE. However, we may require a TMKE protocol to be constructed in a more general fashion (with abstract cryptographic build blocks) and to be independent of specific computational hard problems. We could particularly obtain a wide range of protocol instantiations by substituting those generic blocks (in such a generic protocol) with arbitrary concrete algorithms (from different complexity hard problems), without substantially affecting

* A preliminary version of this paper appears in the proceedings of the RSA Conference Cryptographers' Track 2018 (CT-RSA 2018). This is the full version.

their overall structure or security analysis. In 2008, Boyd et al. [9] proposed an elegant one-round key exchange (ORKE) protocol[4] (which will be referred to as BCNP scheme) from key encapsulation mechanism (KEM). The BCNP scheme is shown to be secure in the Canetti-Krawczyk (CK) model [10] if the KEM scheme is secure against adaptive chosen ciphertext attacks (IND-CCA2). In the generic BCNP protocol, each party is assumed to have a long-term public key. Each party would contribute half of the session key which is encapsulated by its partner's public key. However, the generic BCNP protocol cannot provide weak perfect forward secrecy (wPFS) [23]. In order to improve the generic BCNP scheme, Fujioka et al. [19] proposed a KEM based scheme (which is referred to as FSXY scheme) which satisfies a stronger security in the CK+ model (which covers wPFS). To achieve wPFS, one more KEM scheme which is secure against chosen plaintext attacks (IND-CPA) is used (comparing to the BCNP scheme). Moreover, the NAXOS trick [25, 34] (which is known as twisted pseudo-random function (TPRF)) is applied in the FSXY scheme in order to satisfy the CK+ security (which is quite similar to the extended Canetti-Krawczyk (eCK) security [25]). The NAXOS trick works here as a function which takes as input both long-term and ephemeral secret keys, and outputs an intermediate secret which is assumed to be leakage free. Note that if one of its inputs is not exposed, then its output is still hidden from the adversary. Hence, the NAXOS trick is widely used in key exchange constructions to provide strong security. As pointed in [19], the FSXY scheme can be instantiated by many kinds of KEM including lattice based ones. In particular, we can easily obtain a post-quantum TMKE protocol, e.g. following another variant construction [20], by appropriately instantiating the KEM. These advantages make such generic TMKE to be more interesting. Moreover, the idea of KEM based key exchange is also widely used in real world protocols, such as Transport Layer Security (TLS) protocol v1.2 [14]. More works about TMKE construction are reviewed in Appendix E.

With respect to a generic TMKE protocol, it is remarkable that the security assumptions of underlying cryptographic building blocks are extremely important. A weaker assumption may allow a generic protocol to be more easily or even more efficiently implemented in practice. Note that the FSXY scheme requires both IND-CCA2 KEM and IND-CPA KEM. Building an IND-CCA2 KEM is notoriously more difficult, especially in the standard model. As an IND-CCA2 adversary is allowed to ask a polynomial number of queries to a decryption oracle. In order to weaken the security assumptions of KEM, Kurosawa and Furukawa [24] proposed TMKE protocols (which will be referred to as KF schemes) to provide the security in the CK model and the eCK model respectively. The KF schemes are designed relying on an IND-CPA KEM, and digital signature (SIG). The core idea of the KF scheme is that an initiator generates a fresh ephemeral public key, and the session key is encapsulated by a responder using this public key. The signature scheme here is used to sign the outgoing protocol message for authentication purpose, instead of the long-term public key based IND-CCA2 KEM used in the FSXY and BCNP schemes. Utilizing the IND-CPA KEM (as a building block) seems to be a breakthrough in KEM based TMKE constructions with eCK like strong security. However, is this true? Unfortunately, we are going to show that this result is false.

Our Contributions. In this work, we first revisit the security results of the KF scheme. We present a KCI attack against the concrete KF scheme, which is instantiated with the ElGamal KEM [17]. Our attack shows that the KF scheme cannot provide eCK security based on IND-CPA KEM. The authors have overlooked an important fact under the eCK model (wherein KCI attack is formulated): an initiator's session s' may receive a protocol message m' which is generated by the adversary on behalf of certain corrupted party. In particular, we observe that the session key of the concrete KF protocol can be manipulated by the adversary via her

[4] ORKE is a special class of TMKE. Two protocol messages can be generated and sent independently.

own message. Suppose that the target session under attacked has the session key K^*. Then, the adversary can easily lead another session (which is not the partner session of the target session) to have a related session key $K' = (K^*)^\beta$, where β is some value chosen by the adversary. Namely, the adversary can result in two non-partnered sessions to have related session key relying on the corrupted long-term secret key of the target session. Hence, the adversary can simply obtain K^* after revealing K'. This is possible in the eCK model via a session key reveal query. The details of this attack are illustrated in Section 4.

In order to overcome the design flaws of the KF scheme, we propose a generic construction for TMKE based on KEM, SIG, and pseudo-random function (PRF). In our construction, we particularly study the assumptions required by these building blocks. The security of the proposed scheme is proved without random oracles in the eCK-PFS model [13] which is strengthened from previous works, e.g., [5, 10, 23, 25]. The eCK-PFS model covers several important classes of attacks including: known session key (KSK) attacks, key compromise impersonation (KCI) attacks, chosen identity and public key (CIDPK) attacks, ephemeral secret key leakage (ESKL) attacks, and perfect forward secrecy (PFS) attacks. In order to resist with the quantum computer attacks, we introduce a new KEM scheme (for our TMKE construction) based on the presumed hardness of the Ring Learning with Error (Ring-LWE) problem.

Generic TMKE Scheme. Our construction is similar to the KF scheme (see Figure 1), but in our new KEM based TMKE construction, we need two kinds of KEM. The first KEM is required to satisfy IND-CPA and pair-generation-indistinguishability (PG-IND) introduced by Alawatugoda et al. [3]. This kind of KEM is used as a NAXOS trick as in [3]. The second KEM is used for session key generation, which should satisfy a weaker IND-CCA2 security, i.e., one-time IND-CCA2 (OT-IND-CCA2). Note that OT-IND-CCA2 is just a special case of q-bounded IND-CCA2 (q-IND-CCA2) security defined by Cramer et al. [12] when $q = 1$, where q is the number of allowed decryption oracle queries. In contrast to the regular notion of IND-CCA2, the adversary is only allowed to query at most one decryption oracle query to the challenge public key in the security experiment. This is important to solve the simulation problem of the KF scheme (see more details in Appendix A). One decryption oracle query is enough, because the public key is freshly chosen for each session. We may call an OT-IND-CCA2 secure KEM as OTKEM for short. Meanwhile, a signature scheme, which is strong existentially unforgeable under adaptive chosen messages attacks (SEUF-CMA). It is used to sign the initiator's ephemerally generated public key (of OTKEM) and all protocol messages of the receiver (including the ciphertext of OTKEM). PRF is used as a key derivation function to bind the session key material (the encapsulated key generated by the OTKEM) to specific session identifier sid. Here, the session identifier sid is determined by the whole transcript (T) of exchange protocol messages. Roughly speaking, we want to ensure that only two sessions having the same T would generate identical session keys. We highlight that PRF is useful to circumvent KCI attacks against the concrete KF protocol. Analogously, PRF here is only needed to resist with one-time chosen message attack.

OTKEM Construction from Ring-LWE. In [12], Cramer et al. proposed a generic q-IND-CCA2 secure public key encryption (PKE) scheme from IND-CPA PKE and q-cover-free family. Hence, we can obtain a number of OTKEM instantiations from various assumptions. However, for a security parameter κ and $q = 1$, the Cramer et al.'s scheme has to generate 16κ secret keys and 4κ ciphertexts of IND-CPA PKE scheme. This is quite inefficient. Since we only need to focus on 1-bounded IND-CCA2 security (instead of generic q-bounded one) for instantiating our TMKE scheme, we are motivated to build more efficient OTKEM. We here propose a new post-quantum OTKEM based on Ring-LWE [28, 29] in the standard model (in Section 6).

In order to achieve OT-IND-CCA2 security, the public key pk is generated with a tree-like structure, which consists of 2μ (for some integer μ) ring elements as the form $pk = \{S_{i,j} = a \cdot s_{i,j} + e_{i,j}\}_{(i,j)\in[\mu]\times\{0,1\}}$ where a is a public ring element, $s_{i,j}$ is a secret key and $e_{i,j}$ is a secret error.[5] We stress that such public key is one-time programmable by the μ bits hash value h of a target collision resistant hash function (TCRHF), i.e., $h = (h(1), h(2), \ldots, h(\mu)) := \mathsf{TCRHF}(m)$ where $h(i)$ is the i-th bit of h. The 'programmable' here means that the sub-public keys (i.e., $S_{i,j}$) selected within the encryption algorithm are determined by the bits of h. Namely, given $h = (h(1), h(2), \ldots, h(\mu))$, the set of $\{S_{i,h(i)}\}_{i\in[\mu]}$ will be chosen for encryption. Suppose that TCRHF is target collision resistant and the inputs are distinct among oracle queries (challenge or decryption). At least one of these ring elements can be used to embed the Ring-LWE challenge value, and the other keys can be simulated by the KEM challenger with her own secrets. Then, the challenge value is used only once to compute the challenge ciphertext and the session key. On the other hand, the KEM challenger knows all secrets used to answer the decryption oracle query. One remaining problem here is to determine what the input message of TCRHF should be. Note that the Ring-LWE problem may actually result in an approximate key agreement between sender and receiver. The reconciliation technique [36] can be applied to make the sender and receiver to reach exact agreement from their distinct noisy ring elements. However, the reconciliation function is run only after the session key is generated. Hence, we cannot take the output u of the reconciliation function into the hash function TCRHF. Because the hash value h should be used for session key computation. To prevent the adversary from manipulating u, we exploit an one-time signature scheme to build a faithful chain for the ciphertext. First of all, the ephemeral public key epk (in the ciphertext) of OTKEM and the verification key spk of OTS are generated by the receiver and included in the hash input $m = (pk, epk, spk)$. We have that m in the challenge query is unique with overwhelming probability, since epk and spk are freshly chosen at random. On the other hand, we sign the tuple (pk, epk, spk, u) with the OTS signing key. As long as the OTS scheme is SEUF-CMA secure, then the adversary is unable to manipulate the challenge ciphertext.

2 Preliminaries

General Notations. We let $\kappa \in \mathbb{N}$ be the security parameter and 1^κ be a string that consists of κ ones. For $x \in \mathbb{R}$, we define $\lfloor x \rceil = \lfloor x + 1/2 \rfloor \in \mathbb{Z}$. For any two subsets X, Y of some additive group, let $-X = \{-x : x \in X\}$ and $X + Y = \{x + y : x \in X, y \in Y\}$. We write $[n] = \{1, \ldots, n\} \subset \mathbb{N}$ to denote the set of integers between 1 and n. The radical of a positive integer m is denoted $rad(m)$ which is the product of all primes dividing m. The notation $a \xleftarrow{\$} S$ denotes the operation which samples a uniform random element from a set S. We let $\|$ denote the concatenation (operation) of two strings. We denote the binary representation of a value h with size μ as $h = (h(1), h(2), \ldots, h(\mu)) = \{0,1\}^\mu$.

Key Encapsulation Mechanism Schemes. Generally speaking, a KEM scheme consists of three polynomial time algorithms $\mathsf{KEM} = (\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$ with the following semantics:

- $(pk, sk) \xleftarrow{\$} \mathsf{KEM.Gen}(1^\kappa, rpg)$: a key generation algorithm which on input a security parameter 1^κ and a randomness $rpg \in \mathcal{RG}_{\mathsf{KEM}}$, outputs a pair of long-term encryption/decryption keys $(pk, sk) \in (\mathcal{PK}, \mathcal{SK})$, where $\mathcal{RG}_{\mathsf{KEM}}$ is a randomness space.
- $(K, C) \xleftarrow{\$} \mathsf{KEM.Enc}(pk, erk)$: an encryption algorithm which takes as input an encryption key pk and a randomness $erk \xleftarrow{\$} \mathcal{RK}_{\mathsf{KEM}}$, outputs a key $K \in \mathcal{K}_{\mathsf{KEM}}$ and a ciphertext

[5] A similar construction idea is concurrently applied to build a Decisional Diffie-Hellman (DDH) based one-round key exchange protocol [15].

$C \in \mathcal{C}_{\mathsf{KEM}}$, where $\mathcal{K}_{\mathsf{KEM}}$ is a session key space, $\mathcal{C}_{\mathsf{KEM}}$ is a ciphertext space and $\mathcal{RK}_{\mathsf{KEM}}$ is a randomness space.

– $(K) \leftarrow \mathsf{KEM.Dec}(sk, C)$: a decryption algorithm which takes as input a decryption key sk, a ciphertext $C \in \mathcal{C}_{\mathsf{KEM}}$, and outputs a key $K \in \mathcal{K}_{\mathsf{KEM}}$.

All 'spaces' for the corresponding values are parametrized with the security parameter κ. Let $\mathsf{ind\text{-}x} = \{\mathsf{ind\text{-}cca2}, \mathsf{ot\text{-}ind\text{-}cca2}, \mathsf{ind\text{-}cpa}\}$ ba a variable which stores some indicator of the security experiment, where ind-cca2 denotes the indistinguishability against adaptive chosen ciphertext attacks (IND-CCA2), ot-ind-cca2 denotes indistinguishability against one-time adaptive chosen ciphertext attacks (OT-IND-CCA2), ind-cpa denotes indistinguishability against chosen plaintext attacks (IND-CPA).

Definition 1. *For a key encapsulation mechanism scheme* KEM = (KEM.Gen, KEM.Enc, KEM.Dec) *and an adversary* $\mathcal{D}$*, we define the following experiment:*

$\mathsf{EXP}^{\mathsf{ind\text{-}x}}_{\mathsf{KEM},\mathcal{D}}(\kappa, q)$
$rpg \xleftarrow{\$} \mathcal{RG}_{\mathsf{KEM}}$;
$(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^\kappa, rpg)$;
$(K_0^*, C^*) \xleftarrow{\$} \mathsf{KEM.Enc}(pk, erk)$,
$K_1^* \xleftarrow{\$} \mathcal{K}_{\mathsf{KEM}}$, $b \xleftarrow{\$} \{0,1\}$;
$b' \leftarrow \mathcal{D}^{\mathcal{DEC}(sk,\cdot)}(pk, K_b^*, C^*)$;
if $b = b'$ *then return 1,*
otherwise return 0.

$\mathcal{DEC}(sk, C)$:
If $C = C^*$ *then return a failure* $\perp$,
Otherwise $K \leftarrow \mathsf{KEM.Dec}(sk, C)$
Return K

The number of decryption oracle $\mathcal{DEC}$ *queries is bound by the parameter* q*. We define the advantage of* $\mathcal{D}$ *in the above experiment as:*

$$\mathsf{Adv}^{\mathsf{ind\text{-}x}}_{\mathsf{KEM},\mathcal{D}}(\kappa, q) := \left| \Pr[\mathsf{EXP}^{\mathsf{ind\text{-}x}}_{\mathsf{KEM},\mathcal{D}}(\kappa, q) = 1] - \frac{1}{2} \right|.$$

We say that a key encapsulation mechanism scheme KEM *is secure, if for all PPT adversaries* $\mathcal{D}$ *the advantage* $\mathsf{Adv}^{\mathsf{ind\text{-}x}}_{\mathsf{KEM},\mathcal{D}}(\kappa, q)$ *is a negligible function in* κ*. If* $q = 1$ *then the* KEM *scheme is called as an OT-IND-CCA2 secure one-time key encapsulation mechanism (OTKEM) scheme. If* $q = 0$ *then the* KEM *scheme is IND-CPA secure KEM scheme.*

We also recall the notion regarding the pair-generation-indistinguishability introduced in [3]. We slightly change it for IND-CPA KEM. Let D_1 and D_2 be two distributions such that $D_1 = \{(K, C) : erk \xleftarrow{\$} \mathcal{RK}_{\mathsf{KEM}}, (K, C) \xleftarrow{\$} \mathsf{KEM.Enc}(ek, erk)\}$ and $D_2 := \{(K, C) : C \xleftarrow{\$} \mathcal{C}_{\mathsf{KEM}}, K \leftarrow \mathsf{KEM.Dec}(dk, C)\}$. The KEM is ϵ-pair-generation-indistinguishable (PG-IND) if for all $(ek, dk) \leftarrow \mathsf{KEM.Gen}(1^\kappa, rpg)$, the two distributions D_1 and D_2 are statistically indistinguishable with at most ϵ distance, where $\epsilon \leq \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{KEM},\mathcal{D}}(\kappa, 0)$. Such KEM will be referred to as PG-IND-CPA KEM.

(One-time) Digital Signature Schemes. We consider a digital signature scheme SIG that consists of three probabilistic polynomial time (PPT) algorithms SIG = (SIG.Gen, SIG.Sign, SIG.Vfy) which are described in Section 5. We denote one-time signature (OTS) scheme by OTS = (OTS.Gen, OTS.Sign, OTS.Vfy) representing a special case of SIG. Let ($\mathcal{PK}_{\mathsf{OTS}}$, $\mathrm{SK}_{\mathsf{OTS}}$ $\mathcal{M}_{\mathsf{OTS}}$, $\mathrm{S}_{\mathsf{OTS}}$) be the public, secret key, message and signature space of OTS respectively.

The signature scheme is associated with public and secret key spaces $\{\mathcal{PK}_{\mathsf{SIG}}, \mathcal{SK}_{\mathsf{SIG}}\}$, message space $\mathcal{M}_{\mathsf{SIG}}$, a secret randomness space $\mathcal{RS}_{\mathsf{SIG}}$ and signature space $\mathcal{S}_{\mathsf{SIG}}$ in the security parameter κ. The algorithms of SIG are defined as follows:

– $(sk, vk) \leftarrow \mathsf{SIG.Gen}(1^\kappa, rsg)$: This algorithm takes as input the security parameter κ and a randomness $rg \in \mathcal{RG}_{\mathsf{SIG}}$, and outputs a (public) verification key $vk \in \mathcal{PK}_{\mathsf{SIG}}$ and a secret signing key $sk \in \mathcal{SK}_{\mathsf{SIG}}$, where $\mathcal{RG}_{\mathsf{SIG}}$ is a randomness space.

– $\sigma \leftarrow \mathsf{SIG.Sign}(sk, m, rs)$: This is the signing algorithm that generates a signature $\sigma \in \mathcal{S}_{\mathsf{SIG}}$ for a message $m \in \mathcal{M}_{\mathsf{SIG}}$ with the signing key sk and a randomness $rs \in \mathcal{RS}_{\mathsf{SIG}}$, where $\mathcal{RS}_{\mathsf{SIG}}$ is a randomness space. We here assume that rs includes all randomnesses that are needed for signature generation. For a deterministic signature, rs is empty $\emptyset$.
– $\{0,1\} \leftarrow \mathsf{SIG.Vfy}(vk, m, \sigma)$: This is the verification algorithm that takes as input a verification key pk, a message m and a signature σ, outputs 1 if σ is a valid signature for m under vk, and 0 otherwise.

Let $\mathcal{SIG}(sk, \cdot)$ be a signing oracle which on input message m returns a signature $\sigma \leftarrow \mathsf{SIG.Sign}(sk, m, rs_i)$, where $rs_i \xleftarrow{\$} \mathcal{RS}_{\mathsf{SIG}}$. We use a list Slist to record all tuple (m_i, σ_i, rs_i) where m_i and σ_i are the input and output of i-th $\mathcal{SIG}$ oracle query respectively.

Definition 2. *For a signature scheme* $\mathsf{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Vfy})$ *and an adversary* $\mathcal{F}$*, we define the following experiment:*

$\mathsf{EXP}^{\mathsf{seuf\text{-}cma}}_{\mathsf{SIG},\mathcal{F}}(\kappa, q)$ *:*
$rsg \in \mathcal{RG}_{\mathsf{SIG}}$*;* $(sk, vk) \xleftarrow{\$} \mathsf{SIG.Gen}(1^\kappa, rsg)$*;*
$(\sigma^*, m^*) \leftarrow \mathcal{F}^{\mathcal{SIG}(sk,\cdot)}(vk)$*, which may make up to* q *queries to oracle* $\mathcal{SIG}(sk, m)$*; Return 1, if* $\mathsf{SIG.Vfy}(pk, m^*, \sigma^*) = 1$*, and* $(m^*, \sigma^*) \notin \mathsf{Slist}$*; Output 0 otherwise.*

We define the advantage of $\mathcal{F}$ *in the above experiment as:*

$$\mathsf{Adv}^{\mathsf{seuf\text{-}cma}}_{\mathsf{SIG},\mathcal{F}}(\kappa, q) := \Pr[\mathsf{EXP}^{\mathsf{seuf\text{-}cma}}_{\mathsf{SIG},\mathcal{F}}(\kappa, q) = 1].$$

We say that SIG *is secure against strong existential forgeries* $\mathcal{F}$ *under adaptive chosen-message attacks (SEUF-CMA), if for all PPT adversaries* $\mathcal{F}$ *the advantage* $\mathsf{Adv}^{\mathsf{seuf\text{-}cma}}_{\mathsf{SIG},\mathcal{F}}(\kappa, q)$ *is a negligible function in* κ*. If* $q = 1$ *then* SIG *is called as a SEUF-CMA secure OTS scheme.*

Target Collision-Resistant Hash Functions. Let $\mathsf{TCRHF} : \mathcal{K}_{\mathsf{TCRHF}} \times \mathcal{M}_{\mathsf{TCRHF}} \to \mathcal{Y}_{\mathsf{TCRHF}}$ be a family of keyed-hash functions where $\mathcal{K}_{\mathsf{TCRHF}}$ is the key space, $\mathcal{M}_{\mathsf{TCRHF}}$ is the message space and $\mathcal{Y}_{\mathsf{TCRHF}}$ is the hash value space. The public key $hk_{\mathsf{TCRHF}} \in \mathcal{K}_{\mathsf{TCRHF}}$ defines a hash function, denoted by $\mathsf{TCRHF}(hk_{\mathsf{TCRHF}}, \cdot)$. On input a message $m \in \mathcal{M}_{\mathsf{TCRHF}}$, this function $\mathsf{TCRHF}(hk_{\mathsf{TCRHF}}, m)$ generates a hash value $y \in \mathcal{Y}_{\mathsf{TCRHF}}$. If the hash key hk_{TCRHF} is obvious from the context, we write $\mathsf{TCRHF}(m)$ for $\mathsf{TCRHF}(hk_{\mathsf{TCRHF}}, m)$.

Definition 3. *For a function* $\mathsf{TCRHF} : \mathcal{K}_{\mathsf{TCRHF}} \times \mathcal{M}_{\mathsf{TCRHF}} \to \mathcal{Y}_{\mathsf{TCRHF}}$ *and an adversary* $\mathcal{H}$*, we define the advantage of* $\mathcal{H}$ *against* TCRHF *as:*

$$\mathsf{Adv}^{\mathsf{cr}}_{\mathsf{TCRHF},\mathcal{H}}(\kappa) := \Pr[hk_{\mathsf{TCRHF}} \xleftarrow{\$} \mathcal{K}_{\mathsf{TCRHF}}, m \leftarrow \mathcal{M}_{\mathsf{TCRHF}}, m' \leftarrow \mathcal{H}(hk_{\mathsf{TCRHF}}) : m \neq m' \wedge \mathsf{TCRHF}(m) = \mathsf{TCRHF}(m')].$$

We say that TCRHF *is* target-collision-resistant *if for all PPT adversaries* $\mathcal{H}$ *the advantage* $\mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{TCRHF},\mathcal{H}}(\kappa)$ *is a negligible function in* κ*.*

Pseudo-Random Functions. A pseudo-random function family is denoted by $\mathsf{PRF} : \mathcal{K}_{\mathsf{PRF}} \times \mathcal{D}_{\mathsf{PRF}} \to \mathcal{R}_{\mathsf{PRF}}$, where $\mathcal{K}_{\mathsf{PRF}}$ is the key space, $\mathcal{D}_{\mathsf{PRF}}$ is the domain and $\mathcal{R}_{\mathsf{PRF}}$ is the range of PRF for security parameter κ. Let PList be a list to store the messages queried in the following PRF oracle $\mathcal{FN}$.

Definition 4. *For a* $\mathsf{PRF} : \mathcal{K}_{\mathsf{PRF}} \times \mathcal{D}_{\mathsf{PRF}} \to \mathcal{R}_{\mathsf{PRF}}$ *and an adversary* $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$*, we define the following experiment:*

$\mathsf{EXP}^{\mathsf{ind\text{-}cma}}_{\mathsf{PRF},\mathcal{B}}(\kappa, q)$:	$\mathcal{FN}(k, x)$:
$b \xleftarrow{\$} \{0,1\}, k \xleftarrow{\$} \mathcal{K}_{\mathsf{PRF}}$;	*append* $x \to \mathsf{PList}$;
$(x^*, st) \leftarrow \mathcal{B}_1^{\mathcal{FN}(k,\cdot)}$, *s.t.* $x^* \notin \mathsf{PList}$;	*return* $\mathsf{PRF}(k, x)$;
if $x \in \mathsf{PList}$ *then return a failure* $\perp$;	
$V_1^* := \mathsf{PRF}(k, x^*), V_0^* \xleftarrow{\$} \mathcal{R}_{\mathsf{PRF}}$;	
$b' \leftarrow \mathcal{B}_2^{\mathcal{FN}(k,\cdot)}(st, V_b^*)$	
If $b = b'$ *then return 1;*	
Otherwise return 0;	

In the above experiment, the adversary $\mathcal{B}$ *is allowed to ask at most* q *oracle queries to* $\mathcal{FN}$*. We define the advantage of* $\mathcal{B}$ *in breaking the indistinguishability of* PRF *as:*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cma}}_{\mathsf{PRF},\mathcal{B}}(\kappa, q) := \left| \Pr[\mathsf{EXP}^{\mathsf{ind\text{-}cma}}_{\mathsf{PRF},\mathcal{B}}(\kappa, q) = 1] - \frac{1}{2} \right|.$$

The pseudo-random function PRF *is said to be a secure, if for all PPT adversaries* $\mathcal{B}$ *the advantage* $\mathsf{Adv}^{\mathsf{ind\text{-}cma}}_{\mathsf{PRF},\mathcal{B}}(\kappa, q)$ *is a negligible function in* κ*. If* $q = 1$ *then* PRF *scheme is called as a secure one-time* PRF *scheme.*

Subgaussian Distributions and Random Variables. We review the standard notion of subgaussian which is slightly relaxed as in [32]. For any $\delta > 0$, a random variable X (or its distribution) over $\mathbb{R}$ is said to be δ-subgaussian with parameter $z > 0$ if for all $r \in \mathbb{R}$, the (scaled) moment-generating function satisfies $\mathbb{E}[exp(2\pi r X)] \leq exp(\delta) \cdot exp(\pi z^2 r^2)$. In the light of Markov's inequality, for all $r \geq 0$, we have that

$$\Pr[|X| \geq r] \leq 2exp(\delta - \pi r^2/z^2). \tag{1}$$

It is a well known fact that any B-bounded centered random variable X (i.e., $|X| < B$ always) is 0-subgaussian with parameter $B\sqrt{2\pi}$. The notion of subgaussian can be generally extended to vectors. We say that a random real vector $\mathbf{x}$ is δ-subgaussian (of parameter z) if the inner product $\langle \mathbf{u}, \mathbf{x} \rangle \in \mathbb{R}$ is δ-subgaussian (of parameter z) for any real unit vector $\mathbf{u}$.

Fact 1 *If* X_1 *is* δ_1*-subgaussian with parameter* z_1*, and* X_2 *is* δ_2 *subgaussian with parameter* z_2*, and* X_1, X_2 *are independent, then* $X_1 + X_2$ *is* $(\delta_1 + \delta_2)$*-subgaussian with parameter* $\sqrt{z_1^2 + z_2^2}$*.*

Cyclotomic Rings. We briefly review the background knowledge of cyclotomic rings used in our works. Let m be a positive integer, and $F = \mathbb{Q}(\varsigma_m)$ be the mth cyclotomic field, where ς_m denotes an abstract field element of order m. We denote the cyclotomic ring by $R = \mathbb{Z}[X]/(\Phi_m(X)) \subset F$ where $\Phi_m(X)$ denotes the mth cyclotomic polynomial which is a monic and irreducible polynomial (with degree $n = \varphi(m)$) over rationals. The complex roots of $\Phi_m(X)$ are all the primitive mth roots of unity θ_m^i ($i \in \mathbb{Z}_m^*$) in $\mathbb{C}$, where $\theta_m = \mathsf{exp}(2\pi\sqrt{-1}/m)$. Let R_q denote the quotient ring R/qR, for any integer modulus $q \geq 1$. For any $p|m$, we let $\varsigma_p = \varsigma_m^{m/p} \in R$ (with order p) and define $g = \prod_{odd\ prime\ p|m}(1 - \varsigma_p)$ [36], where $\varsigma_m^{m/p}$ belongs to a $\mathbb{Z}$-basis of R, i.e., $\{1, \varsigma_m, \varsigma_m^2, \ldots, \varsigma_m^{n-1}\}$. We also define $\tilde{m} = m/2$ if m is even, and $\tilde{m} = m$ otherwise. Hence we have the fact that the element g divides $\tilde{m} \in R$, and is co-prime in R with all integer primes except the odd primes $p|m$.

Canonical Embedding. We here review the canonical embedding $\varrho : F \to \mathbb{C}^n$ for defining all geometric quantities. The canonical embedding is represented by the concatenation of n

ring embeddings $\varrho_i : F \to C$ that fix $\mathbb{Q}$ coordinate-wise, such that $\varrho(a) = (\varrho_i(a))_{i\in\mathbb{Z}_m^*}$. Note that, for each $i \in \mathbb{Z}_m^*$, there is an embedding ϱ defined by $\varrho_i(\varsigma_i) = \theta_m^i$, where $\theta_m \in \mathbb{C}$ is some fixed primitive mth root of unity. Clearly, the embeddings come in pairs of complex conjugates, i.e., $\varrho_i = \overline{\varrho_{m-i}}$. In the light of the conjugate pairs of embeddings, ϱ actually maps into the subspace $H \subseteq \mathbb{C}^n$ characterized by this conjugate symmetry. In addition, ℓ_2 and ℓ_∞ norms on F are defined by $\|e\|_2 := \|\varrho(e)\|_2 = (\sum_{i\in\mathbb{Z}_m^*} |\varrho_i(e)|^2)^{1/2}$ and $\|e\|_\infty := \|\varrho(e)\|_\infty = max_{i\in\mathbb{Z}_m^*}|\varrho_i(e)|$, respectively. We also have the expansion bound $\|e+e'||_2 \le \|e\|_2+\|e'\|_2$ and $\|e\cdot e'\|_2 \le \|e\|_2\cdot\|e'\|_\infty$.

Decoding Basis. We here recall some properties of a certain decoding basis [29, 36]. Let $R^\vee = (\hat{m}/g)^{-1}R \subset F$ be a codifferent fractional ideal [29]. Following the demonstration of [36], we map $R^\vee$ to R by multiplying $(\hat{m}/g)$. Then the decoding basis of R is the elements of the decoding basis of $R^\vee$ times $(\hat{m}/g)$. Any $e^\vee \in R^\vee$ that would normally belong to the codifferent by $e = (\hat{m}/g)e^\vee \in R$. While dealing with an error term e, we may always times it with g (i.e., $g \cdot e$) to undo the distortion caused by multiplying $(\hat{m}/g)$.

Lemma 1 ([36]). *Let $e \in F$ be such that $g \times e$ is δ-subgaussian with parameter $\hat{m} \times z$, and let $e' \in F$ be arbitrary. Then every decoding-basis coefficient of $e \times e'$ is δ-subgaussian with parameter $z \cdot \|e'\|_2$.*

Error Distributions. The Gaussian distribution D_z over $\mathbb{R}$ with parameter $z > 0$ is defined by a probability distribution function $\frac{exp(-\pi x^2/z^2)}{z}$. The error distributions are of the form $\psi = (\hat{m}/g) \times D_z$ over F. Such distributions are also discretized to the ring R, which result in a distribution $\mathcal{X} = \lfloor\psi\rceil$, i.e. sampling an elements $a \in F$ from ψ and then rounding each of its rational decoding-basis coefficients to their nearest integers. We have the following fact from [29].

Fact 2 *Let $e \leftarrow \mathcal{X}$ where $\mathcal{X} = \lfloor\psi\rceil$ and $\psi = (\hat{m}/g) \times D_z$. Then we have that (i) $g \cdot e$ is δ-subgaussian with parameter $\hat{m} \cdot \sqrt{z^2 + 2\pi rad(m)/m}$ for some $\delta \le 2^{-n}$; (ii) $\|g \cdot e||_2 \le \hat{m} \cdot (z + \sqrt{rad(m)/m}) \cdot \sqrt{n}$ except with probability at most 2^{-n}.*

Theorem 1 ([28]). *Let R be the m-th cyclotomic ring with dimension $n = \varphi(m)$. Let $\alpha = \alpha(n) < \sqrt{\log n/n}$, and let $q = q(n)$ be a poly(n)-bounded prime such that $q = 1 \mod m$ and $\alpha q \ge \varsigma(\sqrt{\log n})$. This is a poly$(n)$-time quantum reduction from $\tilde{O}(\sqrt{n}/\alpha)$-approximate* SIVP *(or* SVP*) on ideal lattices in R to solving Ring-LWE problem given only $\ell \ge 1$ samples, where $\mathcal{X} = \lfloor\psi\rceil$ and ψ is the Gaussian distribution $(\hat{m}/g) \cdot D_{\xi q}$ for $\xi = \alpha \cdot (n\ell/\log n\ell)^{1/4}$.*

Ring Learning with Errors. We consider the ring learning with error (Ring-LWE) problem based on a variant of cyclotomic ring with canonical embedding and decoding basis as in [29]. We denote D_z the Gaussian distribution over $\mathbb{R}$ with parameter $z > 0$, which is defined by a probability distribution function $exp(-\pi x^2/z^2)/z$. We here just let R_q denote the quotient ring R/qR for any integer modulus $q \ge 1$, and $\mathcal{X}$ be an error distribution.

We here review the decisional problem regarding Ring-LWE. Consider the ring R_q (or just R) defined above, and let the secret $s \xleftarrow{\$} \mathcal{X}$ be sampled from the discretized error distribution $\mathcal{X}$. The Ring-LWE distribution $A_{s,\mathcal{X}}$ over $R_q \times R_q$ is now generated by uniformly selecting $a \xleftarrow{\$} R_q$ and $e \xleftarrow{\$} \mathcal{X}$, and outputting $(a, b = a \cdot s + e)$.

Definition 5. *For a ring R_q and a discretized error distribution $\mathcal{X}$ and an adversary $\mathcal{E}$, we define the following experiment:*

$\mathsf{EXP}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa)$
$a \xleftarrow{\$} R_q,\ (s, e) \xleftarrow{\$} \mathcal{X}, V_0^* := a \cdot s + e \in A_{s,\mathcal{X}},\ V_1^* \xleftarrow{\$} R_q,\ b \xleftarrow{\$} \{0, 1\},$
$b' \leftarrow \mathcal{E}(R_q, \mathcal{X}, a, V_b^*);$ *if $b = b'$ then return 1, otherwise return 0.*

We define the advantage of $\mathcal{E}$ in the above experiment as:

$$\mathsf{Adv}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa) := \left|\Pr[\mathsf{EXP}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa) = 1] - \frac{1}{2}\right|.$$

We say that the decisional Ring-LWE problem is hard relative to R_q and $\mathcal{X}$, if for all PPT adversaries $\mathcal{E}$ the advantage $\mathsf{Adv}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa)$ is a negligible function in κ.

Reconciliation Mechanism. Now we recall the reconciliation mechanism used in [36] for transforming approximate agreement to exact agreement. This technique is one of the foundations of our one-time KEM scheme. For an integer p (e.g. $p = 2$) that divides q, we write $\lfloor\cdot\rceil_p : \mathbb{Z}_q \to \mathbb{Z}_p$ to denote the modular rounding function which works as $\lfloor v\rceil_p := \lfloor \frac{p}{q} \cdot v\rceil$, and $\lfloor\cdot\rfloor_p : \mathbb{Z}_q \to \mathbb{Z}_p$ to denote the modular rounding half down function which works as $\lfloor\cdot\rfloor_p := \lfloor \frac{p}{q} \cdot v\rfloor$. For $p = 2$ and the even modulus $q \geq 2$, we define two disjoint intervals $I_0 := \{0, 1, \ldots, \lfloor\frac{q}{4}\rceil - 1\}$, $I_1 := \{-\lfloor\frac{q}{4}\rfloor, \ldots, -1\} \mod q$ consisting of $\lfloor\frac{q}{4}\rceil$ and $\lfloor\frac{q}{4}\rfloor$ cosets in $\mathbb{Z}_q$ respectively. Note that these intervals split all elements $v \in \mathbb{Z}_q$ into two participations such that $\lfloor v\rceil_2 = 0$ ($v \in \{I_0, I_1\}$) and $\lfloor v\rceil_2 = 1$ ($v \in \{\frac{q}{2} + I_0, \frac{q}{2} + I_1\}$) respectively. We define the crossing-rounding function $\mathsf{HLP} : \mathbb{Z}_q \to \mathbb{Z}_2$ as $\mathsf{HLP}(v) = \lfloor \frac{4}{q} \cdot v\rfloor \mod 2$.

For two sufficiently close elements $v, v' \in \mathbb{Z}_q$ and the set $E := [-\frac{q}{8}, \frac{q}{8}] \cap \mathbb{Z}$, we define the reconciliation function $\mathsf{REC} : \mathbb{Z}_p \times \mathbb{Z}_2 \to \mathbb{Z}_2$ as:

$$\mathsf{REC}(v', b) = \begin{cases} 0 & if\ v' \in I_b + E\ (mod\ q) \\ 1 & otherwise \end{cases}$$

When q is odd, we define the randomized function $\mathsf{DBL} : \mathbb{Z}_q \to \mathbb{Z}_{2q}$. On input a $v \in \mathbb{Z}_q$, it outputs $\bar{v} = 2v - \bar{e} \in \mathbb{Z}_{2q}$ for some random $\bar{e} \in \mathbb{Z}$ which is uniformly random modulo two and independent of v, and small in magnitude. Note that if $v, v' \in \mathbb{Z}_q$ are close, then so are $2v', \mathsf{DBL}(v) \in \mathbb{Z}_{2q}$. If $v' = v + e \mod q$ for some small e, then $2v' = \bar{v} + (2e + \bar{e}) \mod 2q$, where $\bar{e}$ is a random element chosen by $\mathsf{DBL}(v)$ operation. To reconcile some $v' \in \mathbb{Z}_q$, we first transform it to an even element $2v'$ and apply REC to $2v' \in \mathbb{Z}_{2q}$ instead.

We review the security properties of the above functions, i.e. [36, Claim 3.1, Claim 3.2 and Claim 3.3], by the following Lemma 2.

Lemma 2. *For even q, if $v \in \mathbb{Z}_q$ is uniformly random, then $\lfloor v\rceil_2$ is uniformly random given $\mathsf{HLP}(v)$; if $v' = v + e \mod q$ for some $v \in \mathbb{Z}_q$ and $e \in E$, then $\mathsf{REC}(v', \mathsf{HLP}(v)) = \lfloor v\rceil_2$. For odd q, if $v \in \mathbb{Z}_q$ is uniformly random and $\bar{v} := \mathsf{DBL}(v) \in \mathbb{Z}_{2q}$, then $\lfloor\bar{v}\rceil_2$ is uniformly random given $\mathsf{HLP}(\bar{v})$.*

As demonstrated in [36], the above (cross-)rounding and reconciliation functions can be extended to cyclotomic rings R with the decoding basis, and a vector of ring elements. For a decoding basis $T = \{t_i\} \subset R$ and $v = \sum_j v_j \cdot t_j \in R_q$ for coefficients $v_j \in \mathbb{Z}_q$, we can re-write $\lfloor v\rceil_2 := \sum_j \lfloor v_j\rceil_2 \cdot t_j \in R_2$, and $\mathsf{HLP}(v) := \mathsf{HLP}(v_j) \cdot t_j \in R_2$. And the reconciliation function can be modified as $\mathsf{REC}(v', b) = \sum_j \mathsf{REC}(w_j, b_j) \cdot t_j$, where $v' = \sum_j w_j \cdot t_j$ and $b = \sum_j b_j \cdot d_j$.

3 Security Model

In this section, we review the eCK-PFS model which follows from [13, 6, 39]. In order to simulate the behaviors of a set of honest protocol principles in the real world, we may realize a collection of oracles $\{\pi^s_{\mathsf{id}_i} : i \in [\lambda], s \in [d]\}$ for $(\lambda, d) \in \mathbb{N}$. Each oracle $\pi^s_{\mathsf{id}_i}$ works as the s-th protocol instance (session) performed by party id_i. We stress that all oracles in this model can be run sequentially and concurrently. All identities and corresponding public keys $\{\mathsf{id}_i, pk_{\mathsf{id}_i} : i \in [\lambda]\}$

are stored in a public directory $\mathcal{PD}$ that can be accessed by all oracles. Furthermore, each oracle $\pi^s_{\mathsf{id}_i}$ is supposed to keep a list of internal state variables: (i) $\mathsf{pid}^s_{\mathsf{id}_i}$ storing the identities and public keys of session participants (which are sorted lexicographically in terms of identity), e.g., $\mathsf{pid}^s_{\mathsf{id}_i} = (ID_i, pk_{\mathsf{id}_i}, \mathsf{id}_j, pk_{\mathsf{id}_j})$ where id_j is the intended communication partner of id_i in the session $\pi^s_{\mathsf{id}_i}$; (ii) $ds^s_{\mathsf{id}_i} \in \{\mathtt{accept}, \mathtt{reject}\}$ denoting the final decision of a session; (iii) $K^s_{\mathsf{id}_i}$ storing the session key $K^s_{\mathsf{id}_i} \in \mathcal{K}_{\mathsf{ake}}$ (where $\mathcal{K}_{\mathsf{ake}}$ is a session key space); (iv) $sT^s_{\mathsf{id}_i}$ recording the transcript of messages sent by oracle $\pi^s_{\mathsf{id}_i}$; (v) $rT^s_{\mathsf{id}_i}$ recording the transcript of messages received by oracle $\pi^s_{\mathsf{id}_i}$; (vi) the role $\rho^s_{\mathsf{id}_i} \in \{Initiator(I), Responder(R)\}$ of id_i in the session $\pi^s_{\mathsf{id}_i}$.

Adversarial Model. We model an active adversary $\mathcal{A}$ as a probabilistic polynomial time (PPT) Turing Machine. Roughly speaking, active AKE adversaries may take full control of the communication networks, who are able to compromise the ephemeral or long-term secret from a party and to register some malicious parties of her choice, etc. To model these powers, we allow the adversary to ask the following queries:

- $\mathsf{Send}(\mathsf{id}_i, s, m)$: The adversary can use this query to send any message m of his own choice to the oracle $\pi^s_{\mathsf{id}_i}$. The oracle will respond with the next message m^* (if any) to be sent according to the protocol specification and its internal states. Oracle $\pi^s_{\mathsf{id}_i}$ would be initiated via sending the oracle the first message $m = (\top, \widetilde{\mathsf{id}_j})$ consisting of a special initialization symbol $\top$ and a value $\widetilde{\mathsf{id}_j}$. The $\widetilde{\mathsf{id}_j}$ is either the identity id_j of the intended partner or an empty string $\emptyset$. After answering a Send query, the variables $(\mathsf{pid}^s_{\mathsf{id}_i}, ds^s_{\mathsf{id}_i}, K^s_{\mathsf{id}_i}, sT^s_{\mathsf{id}_i}, rT^s_{\mathsf{id}_i,\cdot})$ will be updated depending on the specific protocol.
- $\mathsf{RevealKey}(\mathsf{id}_i, s)$: The oracle $\pi^s_{\mathsf{id}_i}$ responds with the contents of the variable $K^s_{\mathsf{id}_i}$ if and only if the oracle $\pi^s_{\mathsf{id}_i}$ has reached an internal state $ds^s_{\mathsf{id}_i} = \mathtt{accept}$.
- $\mathsf{RevealRand}(\mathsf{id}_i, s)$: The oracle $\pi^s_{\mathsf{id}_i}$ responds with the per-session randomness which is used to generate the protocol message of $\pi^s_{\mathsf{id}_i}$.
- $\mathsf{Corrupt}(\mathsf{id}_i, pk^*)$: This query is proceeded in terms of the following cases:
  - If $i \in [\lambda]$ and the public key of id_i is honest (i.e., it has not been modified by this query), then this query responds with the honest long-term secret key sk_{id_i} (corresponding to the original pk_{id_i}) of the party id_i, and the current public key pk_{id_i} stored in the public directory $\mathcal{PD}$ is replaced with the new pk^* of adversary's choice.
  - If $i \in [\lambda]$ and the current public key pk_{id_i} has been modified by the adversary via this query before, then this query just updates the pk_{id_i} with pk^*.
  - If $i \notin [\lambda]$, then a failure symbol $\perp$ is returned.

  After this query, the party id_i is called corrupted and all unstopped oracles of id_i can answer other queries using its old public/secret key pair. But if the honest public key pk_{id_i} is replaced with a dishonest public key pk^* by this query, then no more oracle of id_i can be initiated since then.
- $\mathsf{RegCorrupt}(\mathsf{id}_\imath, pk_{\mathsf{id}_\imath})$: This query allows the adversary to register an identity $\mathsf{id}_\imath$ ($\lambda < \imath$ and $\imath \in \mathbb{N}$) and a static public key $pk_{\mathsf{id}_\imath}$ on behalf of a party $\mathsf{id}_\imath$. Parties established by this query are called dishonest and are controlled by adversary.
- $\mathsf{Test}(\mathsf{id}_i, s)$: If the oracle has state $ds^s_{\mathsf{id}_i} \neq \mathtt{accept}$ or $K^s_{\mathsf{id}_i} = \emptyset$, then this query returns a failure symbol $\perp$. Otherwise it flips a fair coin $b \xleftarrow{\$} \{0,1\}$, samples a random key $K_0 \xleftarrow{\$} \mathcal{K}_{\mathsf{ake}}$, and sets $K_1 = K^s_{\mathsf{id}_i}$. Finally, the key K_b is returned. The oracle $\pi^s_{\mathsf{id}_i}$ selected by adversary in this query is called as *test oracle*.

Secure AKE Protocols. We first review the notions regarding the communication partnership of two oracles, i.e. *matching sessions* and *origin session* [13]. The definition of origin session here is a little different from [13]. Namely, the identity and the role of a party are considered in our definition.

Definition 6 (Origin Session). *An oracle* $\pi^s_{\mathsf{id}_i}$ *is said to have an* origin session *to an oracle* $\pi^t_{\mathsf{id}_j}$*, if* $\pi^s_{\mathsf{id}_i}$ *has sent all protocol messages,* $\mathsf{id}_i \in \mathsf{pid}^t_{\mathsf{id}_j}$*,* $\rho^s_{\mathsf{id}_i} \neq \rho^t_{\mathsf{id}_j}$ *and* $sT^s_{\mathsf{id}_i} = rT^t_{\mathsf{id}_j}$*. The oracle* $\pi^s_{\mathsf{id}_i}$ *is also said to be the origin oracle of* $\pi^t_{\mathsf{id}_j}$*.*

Definition 7 (Matching Sessions). *An oracle* $\pi^s_{\mathsf{id}_i}$ *is said to have a* matching session *to an oracle* $\pi^t_{\mathsf{id}_j}$*, if* $\pi^s_{\mathsf{id}_i}$ *is an origin oracle of* $\pi^t_{\mathsf{id}_j}$*, and* $\pi^t_{\mathsf{id}_j}$ *is also an origin oracle of* $\pi^s_{\mathsf{id}_i}$*. The oracle* $\pi^t_{\mathsf{id}_j}$ *is said to be the partner oracle of* $\pi^s_{\mathsf{id}_i}$*.*

Correctness. We say an AKE protocol Π is correct, if the oracles $\pi^s_{\mathsf{id}_i}$ and $\pi^t_{\mathsf{id}_j}$ accept with matching sessions, then both oracles should generate the same session key.

Oracle Freshness We now review the notion of *oracle freshness* that describes the active attacks which are allowed in the following security experiment. Let $\pi^s_{\mathsf{id}_i}$ be an accepted oracle with intended partner id_j. And let $\pi^t_{\mathsf{id}_j}$ be an oracle (if it exists), such that $\pi^s_{\mathsf{id}_i}$ has a matching session to $\pi^t_{\mathsf{id}_j}$. Let $\pi^z_{\mathsf{id}_j}$ be an oracle (if it exists), such that $\pi^z_{\mathsf{id}_j}$ has an origin session to $\pi^s_{\mathsf{id}_i}$. Then the oracle $\pi^s_{\mathsf{id}_i}$ is said to be fresh if none of the following conditions holds: (i) $\mathcal{A}$ queried $\mathsf{RegCorrupt}(\mathsf{id}_j, pk_{\mathsf{id}_j})$; (ii) $\mathcal{A}$ queried $\mathsf{RevealKey}(\mathsf{id}_i, s)$; (iii) If $\pi^t_{\mathsf{id}_j}$ exists, $\mathcal{A}$ queried $\mathsf{RevealKey}(\mathsf{id}_j, t)$; (iv) $\mathcal{A}$ queried both $\mathsf{Corrupt}(\mathsf{id}_i, pk_{\mathsf{id}_i})$ and $\mathsf{RevealRand}(\mathsf{id}_i, s)$; (v) If $\pi^z_{\mathsf{id}_j}$ exists, $\mathcal{A}$ queried both $\mathsf{Corrupt}(\mathsf{id}_j, pk_{\mathsf{id}_j})$ and $\mathsf{RevealRand}(\mathsf{id}_j, z)$; (vi) If $\pi^z_{\mathsf{id}_j}$ does not exist, $\mathcal{A}$ queried $\mathsf{Corrupt}(\mathsf{id}_j, pk_{\mathsf{id}_j})$ prior to the acceptance of $\pi^s_{\mathsf{id}_i}$.

Security Experiment $\mathsf{EXP}^{\mathsf{ake}}_{\Pi,\mathcal{A}}(\kappa)$: On input security parameter 1^κ, the security experiment is proceeded as a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ based on AKE protocol Π, where the following steps are performed:

1. $\mathcal{C}$ first implements the collection of oracles $\{\pi^s_{\mathsf{id}_i} : i \in [\lambda], s \in [d]\}$, and generates the long-term key pairs $(pk_{\mathsf{id}_i}, sk_{\mathsf{id}_i})$ for all honest parties id_i for $i \in [\lambda]$ where the identity id_i of each party is chosen uniquely from some identity space $\mathcal{IDS}$. $\mathcal{C}$ gives $\mathcal{A}$ all identities and public keys $\{(\mathsf{id}_1, pk_{\mathsf{id}_1}), \ldots, (\mathsf{id}_\lambda, pk_{\mathsf{id}_\lambda})\}$.
2. During the game, $\mathcal{A}$ may issue a polynomial number of queries regarding Send, RevealRand, Corrupt, RegCorrupt and RevealKey. At some point, $\mathcal{A}$ may ask one (and at most once) $\mathsf{Test}(\mathsf{id}_i, s)$ query.
3. At the end of the game, $\mathcal{A}$ may terminate and output a bit b' as its guess for b of the $\mathsf{Test}(\mathsf{id}_i, s)$ query. Then the experiment returns a failure symbol $\perp$ if one of the following conditions is held: (i) $\mathcal{A}$ has not issued a $\mathsf{Test}(\mathsf{id}_i, s)$ query, or (ii) the $\mathsf{Test}(\mathsf{id}_i, s)$ query returns a failure symbol $\perp$, or (iii) the test oracle is not fresh.
4. Finally, the experiment returns 1 if $b = b'$; Otherwise 0 is returned.

We call an adversary, which runs the above experiment without causing any failure, as a 'legal' adversary.

Definition 8 (Session Key Security). *We define the advantage of a legal adversary* $\mathcal{A}$ *running the above experiment against a correct AKE protocol* Π *as follows:*

$$\mathsf{Adv}^{\mathsf{ake}}_{\Pi,\mathcal{A}}(\kappa) := \left| 2\Pr[\mathsf{EXP}^{\mathsf{ake}}_{\Pi,\mathcal{A}}(\kappa) = 1] - 1 \right|$$

We say that a correct AKE protocol Π *is session-key-secure, if for all PPT legal adversaries* $\mathcal{A}$ *the advantage* $\mathsf{Adv}^{\mathsf{ake}}_{\Pi,\mathcal{A}}(\kappa)$ *is a negligible function in the security parameter* κ*.*

4 On the Insecurity of the KF Scheme

In this section, we present the problems which are overlooked in the KF scheme. We here mainly discuss the issues based on the eCK secure KF protocol [24, Appdenix A], i.e., the 2-pass-eCK protocol. The KF scheme is claimed to be secure in the eCK model relying on the IND-CPA KEM $\mathsf{KEM} = (\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$, a signature scheme $\mathsf{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Vfy})$ and a twisted pseudo-random function $\mathsf{TPRF} : \mathcal{K}_{\mathsf{PRF}} \times \{0,1\}^* \to \mathcal{R}_{\mathsf{PRF}}$ (We refer the reader to [24] for details of TPRF). As suggested in [24], TPRF can be just constructed from regular PRF. For example, $\mathsf{TPRF}((s,s'),(r,r')) = \mathsf{PRF}(s,r') \oplus \mathsf{PRF}(r,s')$ where (s,s') are long-term keys and (r,r') are ephemeral keys. One could just consider TPRF having the similar input and output of PRF, i.e., $k = (s,s')$ and $m = (r,r')$.

Here TPRF is served as a NAXOS trick to resist with the exposure of either ephemeral or long-term key of a session. However, TPRF does not affect our following attack. The core construction idea of the KF scheme is to establish a session key based on an ephemerally generated public key epk of the IND-CPA KEM, i.e., the session key chosen by the receiver is encapsulated by the ephemeral public key generated by the initiator. Meanwhile, the signature scheme used in the KF scheme is expected to authenticate the exchanged messages. Namely, the transported messages (e.g., public key and ciphertext of KEM) are signed by each corresponding party. However, the KF scheme ignores the fact that the protection of a signature becomes invalid if the signing key is corrupted, in such case, the attacker can choose arbitrary protocol message on behalf of the corrupted party.

The KF Scheme. We first review the KF scheme[6] (2-pass-eCK) between two parties id_1 and id_2 as follows:
Step 1. id_1 chooses two random values (r_1, r_2) and computes $R_1 := \mathsf{TPRF}(s_{\mathsf{id}_1}, r_1)$ and $R_2 := \mathsf{TPRF}(s_{\mathsf{id}_1}, r_2)$, where s_{id_1} is one of the long-term keys of id_1. Next id_1 generates $(esk_{\mathsf{id}_1}, epk_{\mathsf{id}_1}) \leftarrow \mathsf{KEM.Gen}(1^\kappa, R_1)$. id_1 sends $X = (\mathsf{id}_1, epk_{\mathsf{id}_1})$ and $\sigma_X := \mathsf{SIG.Sign}(ssk_{\mathsf{id}_1}, X, R_2)$ to id_2, where ssk_{id_1} is the signing key of id_1.
Step 2. If σ_X is invalid, then id_2 aborts. Otherwise id_2 chooses (r_3, r_4) randomly, and computes $R_3 = \mathsf{TPRF}(s_{\mathsf{id}_2}, r_3)$ and $R_4 = \mathsf{TPRF}(s_{\mathsf{id}_2}, r_4)$. id_2 computes $(K, C) := \mathsf{KEM.Enc}(epk_{\mathsf{id}_1}, R_3)$, and sends $Y = (\mathsf{id}_2, C)$ to id_1. id_2 also sends its signature $\sigma_{YX} = \mathsf{SIG.Sign}(ssk_{\mathsf{id}_2}, Y||X, R_4)$ to id_1. It then outputs the session key K.
Step 3. If σ_{YX} is invalid, then id_1 aborts. Otherwise id_1 computes
$K = \mathsf{KEM.Dec}(esk_{\mathsf{id}_1}, C)$, and outputs the session key K.

Note that Kurosawa and Furukawa suggested to instantiate the protocol with ElGamal KEM [17]. This yields a concrete KF protocol. Let $\mathbb{G}$ be a cyclic group with prime order p and group generator g. Then we could instantiate the values of KEM in the above scheme as $epk = g^a$, $C = g^r$ and $K := g^{ar}$, where $(a, r) \xleftarrow{\$} \mathbb{Z}_p^*$.

A KCI Attack against the Concrete KF Protocol. In the following, we show a KCI attack against the ElGamal KEM based concrete KF protocol. This attack could also support our observation on the reduction problem of the generic KF scheme.

We show how an adversary $\mathcal{A}$ violates the security of KF in the eCK model via the following attack:

1. $\mathcal{A}$ first executes the KF protocol instances between two oracles $\pi^s_{\mathsf{id}_1}$ and $\pi^{t^*}_{\mathsf{id}_2}$. $\mathcal{A}$ relays the message from $\pi^s_{\mathsf{id}_1}$ to $\pi^{t^*}_{\mathsf{id}_2}$ without any modification.

[6] The KF scheme here is described verbatim as in [24].

2. $\mathcal{A}$ corrupts id_2 (this is allowed due to the modeling of KCI attacks), and intercepts the signature $\sigma_{YX} := \mathsf{SIG.Sign}(ssk_{\mathsf{id}_2}, Y^*||X, R_4^*)$ from $\pi_{\mathsf{id}_2}^{t^*}$ and $C^* := g^{r^*}$.
3. $\mathcal{A}$ chooses a value β and computes $C_{\mathcal{A}} := g^{r^*\beta}$.
4. Then $\mathcal{A}$ generates another signature value $\sigma_{\mathcal{A}} = \mathsf{SIG.Sign}(ssk_{\mathsf{id}_2}, \mathsf{id}_2||C_{\mathcal{A}}||X, R_4')$, and sends $(C_{\mathcal{A}}, \sigma_{\mathcal{A}})$ to $\pi_{\mathsf{id}_1}^{s}$. The oracle $\pi_{\mathsf{id}_2}^{t^*}$ would accept the session but it is not partnered with $\pi_{\mathsf{id}_1}^{s}$.
5. $\mathcal{A}$ selects the oracle $\pi_{\mathsf{id}_2}^{t^*}$ as the test oracle which should generate the session key $K^* = g^{ar^*}$. $\mathcal{A}$ reveals the session key of $\pi_{\mathsf{id}_1}^{s}$, i.e. $K = g^{ar^*\beta}$. Note that we have the implication $K = (K^*)^{\beta}$. Then the adversary could win the game by extracting the session key of the oracle $\pi_{\mathsf{id}_2}^{t^*}$ as $K^* := (K)^{\beta^{-1}}$.

Thus, $\mathcal{A}$ succeeds in impersonating the honest party id_1 to id_2's oracle $\pi_{\mathsf{id}_2}^{t^*}$, since $\pi_{\mathsf{id}_2}^{t^*}$ is fresh but id_1 has no partner oracle to $\pi_{\mathsf{id}_2}^{t^*}$. The above attack is enough to prove that the KF construction is flawed in the eCK model. We also present the proof reduction problem of the generic KF scheme in Appendix A.

5 A Generic TMKE Construction from OTKEM

In this section, we propose a generic construction for eCK-PFS secure TMKE to overcome the problems of the KF scheme. Another motivation of our scheme is to achieve PFS which is an important security property and not satisfied by the KF scheme. The proposed generic TMKE protocol makes use of building blocks including: (i) OT-IND-CCA2 KEM OTKEM = ($\mathsf{OTKEM.Gen}$, $\mathsf{OTKEM.Enc}$, $\mathsf{OTKEM.Dec}$); (ii) PG-IND-CPA KEM wKEM = ($\mathsf{wKEM.Gen}$, $\mathsf{wKEM.Enc}$, $\mathsf{wKEM.Dec}$); (iii) pseudo-random function $\mathsf{PRF} : \mathcal{K}_{\mathsf{OTKEM}} \times \{0,1\}^* \rightarrow \mathcal{K}_{\mathsf{ake}}$; (iv) signature scheme SIG=($\mathsf{SIG.Gen}$, $\mathsf{SIG.Sign}$, $\mathsf{SIG.Vfy}$) which is strong existentially unforgeable against adaptive chosen message attacks (SEUF-CMA).

It is not hard to see that our KCI attack against the concrete KF scheme can be seen as a variant of chosen ciphertext attack against KEM. Hence, in order to fix the KF scheme, we particularly exploit OT-IND-CCA2 KEM as one of our cryptographic blocks. This is just based on our observation that the ephemeral public key of the initiator should be able to answer at least one decryption oracle query. Note that, in order to resist with the decryption query, a secure OTKEM needs to ensure that the session keys encapsulated by two distinct ciphertexts should be totally independent. This fact could thwart our KCI attack against the KF scheme (and fix the reduction problem shown in Figure 3). In contrast to the KF scheme, the pseudo-random function PRF is used as a key derivation function to bind all session related information (protocol messages and identities) into the corresponding session key. This is important to withstand active attacks, such as unknown key share attacks, and many others.

In our scheme, the PG-IND-CPA KEM wKEM is used as a NAXOS trick [3] to compute the input random valuees of other underlying building blocks. This NAXOS trick is only used as an alternative example to resist with the ephemeral key leakage from the test oracle. Moreover, one could obtain a leakage resilient TMKE protocol by appropriately instantiating wKEM as in [3]. However, we stress that, without such NAXOS trick, our scheme can still be secure in a rather strong security model called CK-PFS [40] which is just a variant of the CK model with PFS. Hence, one is free to obtain a simpler construction which is easier to realize, if achieving eCK-PFS like security is not a priority.

To provide the security in the eCK-PFS model, we require the signature scheme to meet one of the following *additional requirements* (which may be referred to as AR for short): (i) SIG is deterministic; (ii) each signing random value rs can be found within the corresponding signature σ, i.e., $rs \in \sigma$, where $\sigma \leftarrow \mathsf{SIG.Sign}(sk, m, rs)$ for some message m. These requirements

are implicitly given in the eCK-PFS secure BJS scheme [6]. Note that we do not generate the random value rs for signature generation via wKEM. Because, if we do so, we may be unable to reduce the security of our scheme to the IND-CPA security of wKEM. When the test oracle has no origin oracle (e.g., the adversary outputs a forgery based on an honest initiator's ephemeral public key), then the indented parter id_j of the test oracle $\pi^{s^*}_{\mathsf{id}_i}$ is allowed to be corrupted after $\pi^{s^*}_{\mathsf{id}_i}$ accepts. In this case, the adversary may know all secrets (ephemeral or long-term) of id_j's oracles. This is also why we need the strong unforgeability here (unlike EUF-CMA required in the KF scheme). However, during the security reduction to the SEUF-CMA security of SIG, each signing random value rs_{id_j} of id_j might be unknown to the challenger without AR.

Protocol Description. Our generic protocol is described as follows.

Initiation: At the beginning, a party id first chooses random value $rs_{\mathsf{id}} \xleftarrow{\$} \mathcal{RS}_{\mathsf{SIG}}$ and $rpg'_{\mathsf{id}} \xleftarrow{\$} \mathcal{RG}_{\mathsf{wKEM}}$. Then, it runs $(ssk_{\mathsf{id}}, spk_{\mathsf{id}}) \leftarrow \mathsf{SIG.Gen}(1^\kappa, rs_{\mathsf{id}})$ and $(dk_{\mathsf{id}}, ek_{\mathsf{id}}) \leftarrow \mathsf{wKEM.Gen}(1^\kappa, rpg'_{\mathsf{id}})$, where ek_{id} is discarded. The long-term secret key of id is $sk_{\mathsf{id}} = (ssk_{\mathsf{id}}, dk_{\mathsf{id}})$, and the corresponding public key is $pk_{\mathsf{id}} = spk_{\mathsf{id}}$.

Protocol Execution: The detail protocol executed between two parties id_1 and id_2 is shown by Figure 1.

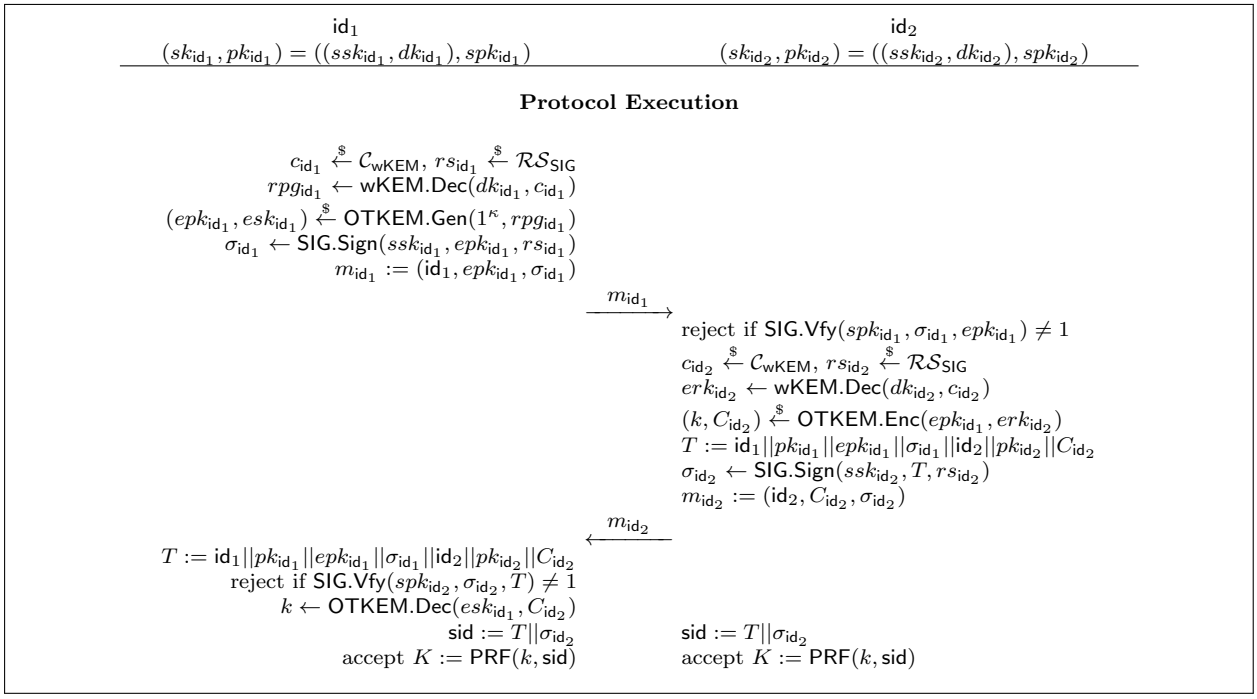

Fig. 1: Generic TMKE from OTKEM

Instantiations of Building Blocks. We hereby try to instantiate other underlying cryptographic building blocks which are able to resist with quantum computer attacks. With respect to wKEM, one could (for example) instantiate it using the Ring-LWE based scheme recently proposed by Peikert in [36] (or the one by Lyubashevsky et al. [29]). It is not hard to check that Peikert's scheme is ϵ-PG-IND with a negligible ϵ (otherwise it is not IND-CPA secure).

The lattice-based SEUF-CMA signature scheme proposed by Rückert [33] may be suitable for instantiating our scheme. An efficient Ring-LWE based PRF introduced by Banerjee et al. [4] can be used to realize our scheme. However, we stress that our TMKE scheme only needs PRF to withstand 'one chosen message query' in the security reduction. This property may lower the assumption when selecting concrete PRF schemes. We refer reader to [21, 35, 1, 2, 11] for LWE-based public-key cryptosystems which might be alternatively applicable to instantiate our generic protocol. Moreover, we compare our generic scheme with other KEM-based TMKE in Appendix B. In the next section, we will introduce a new concrete OTKEM which is suitable for our generic TMKE construction.

Security Analysis. The security result of our scheme is shown by the following theorem.

Theorem 2. *Suppose that the pseudo-random function* PRF *is secure, the key encapsulation mechanism* OTKEM *is OT-IND-CCA2 secure, the signature scheme* SIG *is SEUF-CMA secure and meets* AR*, and the key encapsulation mechanism* wKEM *is both ϵ-PG-IND and IND-CPA secure, with respect to the Definitions in Section 2. Then the proposed generic* TMKE *protocol is session-key-secure with* $\mathsf{Adv}^{\mathsf{ake}}_{\mathsf{TMKE},\mathcal{A}}(\kappa) \leq \lambda \cdot \mathsf{Adv}^{\mathsf{seuf\text{-}cma}}_{\mathsf{SIG},\mathcal{F}}(\kappa, d) + (4(d\lambda)^2) \cdot (4 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{wKEM},\mathcal{N}}(\kappa, 0) + 2 \cdot \mathsf{Adv}^{\mathsf{ot\text{-}ind\text{-}cca2}}_{\mathsf{OTKEM},\mathcal{D}}(\kappa, 1) + \mathsf{Adv}^{\mathsf{ind\text{-}cma}}_{\mathsf{PRF},\mathcal{B}}(\kappa, 1))$.

The full proof of this theorem can be found in Appendix C. We here only give some intuition for the proof of Theorem 2. The proof is basically proceeded in a sequence of games following the approach introduced by Shoup [38].

The first **Game 0** is the real security experiment. In **Game 1**, we show that no PPT adversary can forge the signature of any uncorrupted party. Otherwise the game is aborted. As a result, the test oracle always has an origin oracle.

In **Game 2**, we try to guess some important information regarding the test oracle and its origin oracle. The subsequent games are proceeded based on such correct guess.

We gradually change **Game 2** to **Game 3**, **Game 4** and **Game 5** by modifying the random values used by the test oracle and its origin oracle to be uniform random instead of generating them from wKEM. If the ephemeral key (i.e. the ciphertext c) is not exposed, then the output of $\mathsf{wKEM.Dec}(dk, c)$ is just a random value. When dk is not corrupted, the challenger just uses the encryption key ek to generate c instead. The security of wKEM can ensure that no PPT adversary is able to distinguish this change. Meanwhile, the security of OTKEM can ensure that each oracle generates a unique ephemeral public key epk.

We modify **Game 5** to **Game 6** by changing the PRF seed of the test oracle to be a random value. This change is used to reduce the security to that of OTKEM.

In the last game, i.e. **Game 7**, the session key of the test oracle is changed to be a random value. No PPT adversary can distinguish this change because of the security of PRF. Since the bit of Test query is not used any more. The adversary's advantage in this game is just zero.

6 An OTKEM from Ring-LWE

In this section, we introduce a new construction for OTKEM from Ring-LWE. The other building blocks include a collision resistant hash function $\mathsf{TCRHF} : hk_{\mathsf{TCRHF}} \times R_q \rightarrow \{0,1\}^{\mu}$ where $hk_{\mathsf{TCRHF}} \xleftarrow{\$} \mathcal{K}_{\mathsf{TCRHF}}$, and a SEUF-CMA one-time signature scheme OTS=(OTS.Gen, OTS.Sign, OTS.Vfy).A concrete solution for collision resistant hash function over rings can be found in [27]. The one-time signature scheme, for example proposed by Lyubashevsky and Micciancio [26] based on ideal lattice, could satisfy our requirement.

Construction. Let m be a positive integer specifying the m-th cyclotomic ring R of degree $n = \phi(m)$ and order q. Let q denote a positive odd modulus which is co-prime with every odd

prime dividing m and $q \equiv 1 \mod m$. Let $\mathcal{X} = \lfloor \psi \rceil$ be a discretized error distribution over R, where $\psi = (\hat{m}/g) \cdot D_z$ is over $\mathbb{F}$ for some parameter z. We also randomly choose $a \xleftarrow{\$} R_q$ as a public parameter. The main construction idea is inspired by the 'encoding procedure' in garbled circuits. And we rely on the possibility of homomorphic operations over ring elements. The concrete algorithms of our OTKEM are defined in Figure 2.

$\mathsf{OTKEM.Gen}(1^\kappa, rpg)$:	$\mathsf{OTKEM.Dec}(sk, C)$:
Parse $rpg = \{s_{\eta,\iota}, e_{\eta,\iota}\}_{(\eta,\iota)\in[\mu]\times\{0,1\}} \xleftarrow{\$} (\mathcal{X})^{2\mu}$; $sk = \{s_{\eta,\iota}\}_{(\eta,\iota)\in[\mu]\times\{0,1\}}$; $pk = \{S_{\eta,\iota}\}_{(\eta,\iota)\in[\mu]\times\{0,1\}}$ $:= \{a \cdot s_{\eta,\iota} + e_{\eta,\iota}\}_{(\eta,\iota)\in[\mu]\times\{0,1\}}$; Return (sk, pk).	$T = pk\|\|Y\|\|spk$; reject if $\mathsf{OTS.Vfy}(spk, \sigma, T\|\|u) \neq 1$; $h = (h(1), h(2), \ldots, h(\mu)) := \mathsf{TCRHF}(T)$; $v' = g \cdot Y \cdot \sum_{\eta=1}^{\mu} s_{\eta,h(\eta)}$ $= g \cdot (a \cdot r + e) \cdot \sum_{\eta=1}^{\mu} s_{\eta,h(\eta)}$ $K := \mathsf{REC}(2v', u)$; Return K.
$\mathsf{OTKEM.Enc}(pk, erk)$:	
Parse $erk = (r, e, f, rsg, rs) \xleftarrow{\$} (\mathcal{X})^3 \times \mathcal{RG}_{\mathsf{OTS}} \times \mathcal{RS}_{\mathsf{OTS}}$; $(ssk, spk) \leftarrow \mathsf{OTS.Gen}(1^\kappa, rsg)$, $Y := a \cdot r + e$, $T := pk\|\|Y\|\|spk$; $h = (h(1), h(2), \ldots, h(\mu)) := \mathsf{TCRHF}(T)$; $v := g \cdot r \cdot (\sum_{\eta=1}^{\mu} S_{\eta,h(\eta)}) + f = g \cdot r \cdot (\sum_{\eta=1}^{\mu} (a \cdot s_{\eta,h(\eta)} + e_{\eta,h(\eta)}) + f$; $\bar{v} = \mathsf{DBL}(v)$, $u := \mathsf{HLP}(\bar{v})$, $K := \lfloor \bar{v} \rceil \in R_2$; $\sigma := \mathsf{OTS.Sign}(sk, T\|\|u, rs), C := (Y, u, spk, \sigma)$; Return (K, C).	

Fig. 2: A Concrete OTKEM

Correctness. In order to show that both encryption and decryption algorithms compute the same session key, we first further expand the computations of v and v' as follows:
(i) $v = (\sum_{\eta=1}^{\mu} g \cdot r \cdot a \cdot s_{\eta,h(\eta)} + \sum_{\eta=1}^{\mu} g \cdot r \cdot e_{\eta,h(\eta)}) + f$; and
(ii) $v' = g \cdot (a \cdot r + e) \cdot \sum_{\eta=1}^{\mu} s_{\eta,h(\eta)} = \sum_{\eta=1}^{\mu} g \cdot a \cdot r \cdot s_{\eta,h(\eta)} + \sum_{\eta=1}^{\mu} g \cdot e \cdot s_{\eta,h(\eta)}$.

Let $\hat{g_1} = (\sum_{\eta=1}^{\mu} g \cdot r \cdot e_{\eta,h(\eta)}) + f$, $\hat{g_2} = \sum_{\eta=1}^{\mu} g \cdot e \cdot s_{\eta,h(\eta)}$, and $\hat{f} = \sum_{\eta=1}^{\mu} r \cdot s_{\eta,h(\eta)}$. Then we can rewrite v and v' as $v' = v + \hat{g_2} - \hat{g_1}$. We note that if v and v' are sufficiently close, then we have both encryption and decryption algorithms compute the same session key. Let $z' = \sqrt{z^2 + 2\pi \cdot rad(m)/m}$ and $\gamma = \hat{m} \cdot (z + \sqrt{rad(m)/m}) \cdot \sqrt{n}$.

Lemma 3. *Suppose* $\|g \cdot s_{\eta,h(\eta)}\|_2 \leq \gamma$ *and* $\|g \cdot e_{\eta,h(\eta)}\|_2 \leq \gamma$ *for* $\eta \in [\mu]$, *and* $(q/8)^2 \geq \omega^2 \cdot (z'^2 \cdot (2\mu\gamma^2 + n) + \pi/2)$, *for some* $\omega > 0$. *Then the proposed* $\mathsf{OTKEM.Dec}$ *decrypts correctly except with probability at most* $2n \cdot exp((2\mu + 1)\delta - \omega^2\pi)$ *for some* $\delta \leq 2^{-n}$.

Proof. Let $t = \hat{g_2} - \hat{g_1}$ and $\bar{e}$ be the random error chosen by $\mathsf{DBL}(v)$ on calculating $\bar{v} := 2v - \bar{e}$. By applying Lemma 2, it suffices to show that the decoding-basis coefficients of $2t + \bar{e}$ are all in $[-\frac{q}{4}, \frac{q}{4})$ with overwhelming probability as claimed. Due to the Fact 2, we have that $g \cdot e$ and $g \cdot r$ are δ-subgaussian with parameter $\hat{m} \cdot z'$. As $\mathsf{MAX}(\|g \cdot s_{\eta,h(\eta)}\|_2, \|g \cdot e_{\eta,h(\eta)}\|_2) \leq \gamma$ (for $\eta \in [\mu]$), the decoding-basis coefficients of $g \cdot e \cdot s_{\eta,h(\eta)}$ and $g \cdot r \cdot e_{\eta,h(\eta)}$ are all δ-subgaussian with parameter $z'\gamma$. By applying Lemma 1 and assuming $e' = 1$ (with $\|e'\|_2 = \sqrt{n}$), the decoding-basis coefficients of f are all δ-subgaussian with parameter $z'\sqrt{n}$. By the assumption the decoding-basis coefficients of $\bar{e}$ are all 0-subgaussian with parameter $\sqrt{2\pi}$. Because the elements r, f, e and $\bar{e}$ are all mutually independent, the decoding-basis coefficients of $2t + \bar{e}$ are all $(2\mu + 1)\delta$-subgaussian with parameter $2(z'^2 \cdot (2\mu\gamma^2 + n) + \frac{\pi}{2})^{1/2}$. The result of this lemma follows by applying equation 1 and the union bound over all n coefficients.

Security Result. Now we show the security of our proposed OTKEM via the following theorem.

Theorem 3. *Suppose the Ring-LWE assumption holds, the one-time signature scheme* OTS *is SEUF-CMA secure, and the hash function* TCRHF *is target-collision-resistant, then the proposed one-time key encapsulation mechanism* OTKEM *is secure with* $\mathsf{Adv}^{\mathsf{ot\text{-}ind\text{-}cca2}}_{\mathsf{OTKEM},\mathcal{D}}(\kappa,1) \leq \mathsf{Adv}^{\mathsf{seuf\text{-}cma}}_{\mathsf{OTS},\mathcal{F}}(\kappa,1) + \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{TCRHF},\mathcal{H}}(\kappa) + 4\mu \cdot \mathsf{Adv}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa)$.

The full proof is presented in Appendix D. We here give a general overview of the proof of Theorem 3. The proof is again shown by a number of games as the approach in [38]. Let $C^* := (Y^*, u^*, spk^*, \sigma^*)$ denote the ciphertext generated by challenge query.

Game 0 is the real security experiment. In **Game 1**, the challenger aborts if the adversary can generate a forge of the OTS scheme for the challenge OTS verification key spk^*. Due to the security of the OTS scheme, the adversary is unable to manipulate the value u^*.

In **Game 2**, we reduce the security to that of TCRHF. Therefore, there is no collision to $h^* = \mathsf{TCRHF}(pk^*||Y^*||spk^*)$ in the subsequent games.

In **Game 3**, we try to guess the τ^*-th bit in h^*, which is distinct to the τ^* bit of the hash value generated in the decryption oracle query.

In **Game 4**, we change the public key $S^*_{\tau^*,h^*(\tau^*)}$ to be a random ring element.

Finally, Y^*, v^* and K^* are changed to be random values in **Game 5**. These changes enable us to reduce the security to the hardness of the Ring-LWE problem.

Concrete Parameters. We now select the choices of the parameters for guaranteeing the asymptotic hardness (worst-case) of the Ring-LWE problem in our scheme. Suppose that $\mu \leq n$ and $\hat{m} = O(n)$. Since $rad(m)/m \leq 1$, we have that each $\|g \cdot s_{\eta,h(\eta)}\|_2 \leq \hat{m} \cdot (r+1) \cdot \sqrt{n}$ and $z' \leq z^2 + 2\pi$, except probability at most 2^{-n}. By taking $\omega = \sqrt{ln(2n/\epsilon)/\pi}$ (following [31, Lemma 3.3]) and $q \leq 8\omega\sqrt{(z^2+2\pi)(2\mu \cdot \hat{m}^2 \cdot (z+1)^2 + 1) \cdot n} = O(\hat{m} \cdot z^2 \cdot n) \cdot \omega$, the probability of a decryption failure is then bounded by ϵ. For instance, let $\epsilon = 2^{-128}$. Therefore, we may take $q = O(z^2 \cdot n^2 \log n)$. By applying Theorem 1 with $\ell = 2\mu + 1$, we specify that $z = \xi q$ and $\xi = \alpha \cdot ((2\mu+1)n/log((2\mu+1)n))^{1/4} \leq (2n^2/log(2n^2))^{1/4}$, where $z = (2n^2/log(2n^2))^{1/4} \cdot \omega(\sqrt{\log n})$ and $q = \tilde{O}(n^3)$. Then, the Ring-LWE problem is hard as long as the SVP problem on ideal lattices in R is hard to approximate to $\tilde{O}(\sqrt{n}/\alpha) = \tilde{O}(\sqrt{n} \cdot q) = \tilde{O}(n^{7/2})$.

References

1. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In Proc. of *EUROCRYPT'10*, Vol. 6110 of *LNCS*, pp. 553–572. Springer, Heidelberg, May 2010.
2. S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Proc. of *CRYPTO'10*, Vol. 6223 of *LNCS*, pp. 98–115. Springer, Heidelberg, August 2010.
3. J. Alawatugoda, D. Stebila, and C. Boyd. Modelling after-the-fact leakage for key exchange. In Proc. of *ASIACCS'14*, pp. 207–216. ACM Press, June 2014.
4. A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In Proc. of *EUROCRYPT'12*, Vol. 7237 of *LNCS*, pp. 719–737. Springer, April 2012.
5. M. Bellare and P. Rogaway. Entity authentication and key distribution. In Proc. of *CRYPTO'93*, Vol. 773 of *LNCS*, pp. 232–249. Springer, Heidelberg, August 1994.
6. F. Bergsma, T. Jager, and J. Schwenk. One-round key exchange with strong security: An efficient and generic construction in the standard model. In Proc. of *PKC'15*, Vol. 9020 of *LNCS*, pp. 477–494. Springer, March / April 2015.
7. J. W. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan, and D. Stebila. Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE. In Proc. of *ACM CCS'16*, pp. 1006–1018. ACM Press, 2016.
8. J. W. Bos, C. Costello, M. Naehrig, and D. Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In Proc. of *IEEE S&P 2015*, pp. 553–570. IEEE Computer Society Press, May 2015.
9. C. Boyd, Y. Cliff, J. G. Nieto, and K. G. Paterson. Efficient one-round key exchange in the standard model. In Proc. of *ACISP'08*, Vol. 5107 of *LNCS*, pp. 69–83. Springer, Heidelberg, July 2008.

10. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Proc. of *EUROCRYPT'01*, Vol. 2045 of *LNCS*, pp. 453–474. Springer, Heidelberg, May 2001.
11. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In Proc. of *EUROCRYPT'10*, Vol. 6110 of *LNCS*, pp. 523–552. Springer, Heidelberg, May 2010.
12. R. Cramer, G. Hanaoka, D. Hofheinz, H. Imai, E. Kiltz, R. Pass, A. Shelat, and V. Vaikuntanathan. Bounded CCA2-secure encryption. In Proc. of *ASIACRYPT'07*, Vol. 4833 of *LNCS*, pp. 502–518. Springer, Heidelberg, December 2007.
13. Cas J. F. Cremers and M. Feltz. Beyond eCK: Perfect forward secrecy under actor compromise and ephemeral-key reveal. In Proc. of *ESORICS'12*, Vol. 7459 of *LNCS*, pp. 734–751. Springer, Heidelberg, September 2012.
14. T. Dierks and E. Rescorla. The transport layer security (tls) protocol version 1.2. rfc 5246 (proposed standard), Auguest 2008.
15. Z. Yang, J. Lai. New constructions for (multiparty) one-round key exchange with strong security[J]. Sciece China Information Sciences, 2018, 61(5):059102.
16. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
17. T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, Jul 1985.
18. E. S. V. F., D. Hofheinz, E. Kiltz, and K. G. Paterson. Non-interactive key exchange. In Proc. of *PKC'13*, Vol. 7778 of *LNCS*, pp. 254–271. Springer, Heidelberg, February / March 2013.
19. A. Fujioka, K. Suzuki, K. Xagawa, and K. Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. In Proc. of *PKC'12*, Vol. 7293 of *LNCS*, pp. 467–484. Springer, Heidelberg, May 2012.
20. A. Fujioka, K. Suzuki, K. Xagawa, and K. Yoneyama. Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In Proc. of *ASIACCS'13*, pp. 83–94. ACM Press, May 2013.
21. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Proc. of ACM STOC'08, pp. 197–206. ACM Press, May 2008.
22. T. Jager, F. Kohlar, S. Schäge, and J. Schwenk. On the security of TLS-DHE in the standard model. In Proc. of *CRYPTO'12*, Vol. 7417 of *LNCS*, pp. 273–293. Springer, Heidelberg, August 2012.
23. H. Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In Proc. of *CRYPTO'05*, Vol. 3621 of *LNCS*, pp. 546–566. Springer, August 2005.
24. K. Kurosawa and J. Furukawa. 2-pass key exchange protocols from CPA-secure KEM. In Proc. of *CT-RSA'14*, Vol. 8366 of *LNCS*, pp. 385–401. Springer, Heidelberg, February 2014.
25. B. A. LaMacchia, K. Lauter, and A. Mityagin. Stronger security of authenticated key exchange. In Proc. of *ProvSec'07*, Vol. 4784 of *LNCS*, pp. 1–16. Springer, Heidelberg, November 2007.
26. V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures. In Proc. of *TCC'08*, Vol. 4948 of *LNCS*, pp. 37–54. Springer, Heidelberg, March 2008.
27. V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In Proc. of *FSE'08*, Vol. 5086 of *LNCS*, pp. 54–72. Springer, Heidelberg, February 2008.
28. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In Proc. of *EUROCRYPT'10*, Vol. 6110 of *LNCS*, pp. 1–23. Springer, Heidelberg, May 2010.
29. V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-LWE cryptography. In Proc. of *EUROCRYPT'13*, Vol. 7881 of *LNCS*, pp. 35–54. Springer, May 2013.
30. A. Menezes and B. Ustaoglu. Comparing the pre- and post-specified peer models for key agreement. In Proc. of *ACISP'08*, Vol. 5107 of *LNCS*, pp. 53–68. Springer, Heidelberg, July 2008.
31. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. In Proc. of FOCS'04, pp. 372–381, Oct 2004.
32. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In Proc. of *EUROCRYPT'12*, Vol. 7237 of *LNCS*, pp. 700–718. Springer, Heidelberg, April 2012.
33. M. Rückert. Strongly Unforgeable Signatures and Hierarchical Identity-Based Signatures from Lattices without Random Oracles. In Proc. of *PQCrypto*, Vol. 6061 of *LNCS*, pp.182-200. Springer, Heidelberg, May 2010.
34. T. Okamoto. Authenticated key exchange and key encapsulation in the standard model (invited talk). In Proc. of *ASIACRYPT'07*, Vol. 4833 of *LNCS*, pp. 474–484. Springer, Heidelberg, December 2007.
35. C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Proc. of ACM STOC'09, pp. 333–342. ACM Press, May / June 2009.
36. C. Peikert. *Lattice Cryptography for the Internet*, In Proc. of PQCrypto'14, pp. 197–219. Springer International Publishing, 2014.
37. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Proc. of ACM STOC'05, pp. 84–93. ACM Press, May 2005.

38. V. Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/.
39. Z. Yang, J. Lai, C. Liu, W. Liu, and S. Li. Simpler generic constructions for strongly secure one-round key exchange from weaker assumptions. *The Computer Journal*, 60(8):1145–1160, August 2017.
40. Z. Yang, J. Lai, C. Liu, W. Liu, and S. Luo. Signorke: Improving pairing-based one-round key exchange without random oracles. *IET Information Security*, 11 (5):243–249, September 2017.
41. J. Zhang, Z. Zhang, J. Ding, M. Snook, and Ö. Dagdelen. Authenticated key exchange from ideal lattices. In Proc. of *EUROCRYPT'15*, Vol. 9057 of *LNCS*, pp. 719–751. Springer, Heidelberg, April 2015.
42. B. Gong, Y. Zhao. Small Field Attack, and Revisiting RLWE-Based Authenticated Key Exchange from Eurocrypt'15. IACR Cryptology ePrint Archive, 2016/913.

A Incorrect Security Reduction of the KF Scheme

The reduction problem of the KF scheme is illustrated in the following, which is also informally shown in Figure 3. Consider the situation that there exists a distinguisher $\mathcal{D}$ who tries to make use of a successful eCK adversary $\mathcal{A}$ against the KF scheme to break the IND-CPA security of KEM. Given a KEM challenge instance (pk^*, C^*, K_b^*), the goal of $\mathcal{D}$ is to distinguish whether or not K_b^* is a real session key or a uniform random. Then $\mathcal{D}$ simulates the AKE security experiment for $\mathcal{A}$. Assume that the test oracle selected by $\mathcal{A}$ is $\pi_{\mathsf{id}_2}^{t^*}$ which has intended communication partner id_1. Note that, in the simulation, $\mathcal{D}$ needs to set one of the messages of $\pi_{\mathsf{id}_2}^{t^*}$ to be C^*, and the ephemeral public key of an oracle $\pi_{\mathsf{id}_1}^{s}$ to be pk^*. The other ephemeral secrets R_2 and R_4^* are chosen at random. The signatures are generated honestly as the protocol specification.

Note that $\mathcal{A}$ can replay the message from $\pi_{\mathsf{id}_1}^{s}$ to $\pi_{\mathsf{id}_2}^{t^*}$. Meanwhile, $\mathcal{A}$ corrupts id_2, and then generates a valid protocol message (in particular for the ciphertext $C_{\mathcal{A}}$) of her own choice using the knowledge of id_2's signing key, and sends such message (i.e., $C_{\mathcal{A}}$ and $\sigma_{\mathcal{A}}$) to $\pi_{\mathsf{id}_2}^{s}$. Recall that the ephemeral public key of $\pi_{\mathsf{id}_1}^{s}$ is set to be the KEM challenge value pk^* in order to do the security reduction. In this case, $\mathcal{D}$ cannot generate the session key for $\pi_{\mathsf{id}_1}^{s}$ without knowing both the ephemeral secret key for decrypting the received ciphertext $C_{\mathcal{A}}$ (chosen by adversary $\mathcal{A}$) and the secret key sk^* (related to pk^*). The ephemeral secret key esk^* and the ephemeral key used to compute $C_{\mathcal{A}}$ are both unknown to $\mathcal{D}$. But the adversary $\mathcal{A}$ can generate the session key of $\pi_{\mathsf{id}_1}^{s}$, since the ciphertext $C_{\mathcal{A}}$ is chosen by herself. In a nutshell, $\mathcal{D}$ is unable to correctly plug the KEM challenge instance into the simulation of the security experiment for $\mathcal{A}$.

B Comparison

In this section, we briefly summarize the comparison among our generic TMKE scheme and some other TMKE schemes which might be post-quantum secure. In Table 1, we mainly compare the building blocks and their security assumptions. Let 'EXT' denote the strong randomness extractor and 'H' be a cryptographic hash function (which is modeled as a random oracle). Let P_1 (C_1), P_2 (C_2) and P_3 (C_3) denote public key (ciphertext) of IND-CPA, IND-CCA and OT-IND-CCA2 KEM schemes respectively. Let R denote a ring element. And let S denote a SEUF-CMA regular signature. The efficiency here is not our main concern due to their similar (KEM-based) construction structures.

C Proof of Theorem 2: Session Key Security of the Proposed Generic TMKE Scheme

In this proof, we wish to show that there is no adversary $\mathcal{A}$ which can distinguish the real session key of the test oracle from a uniform random. Otherwise there must exist some adversary which

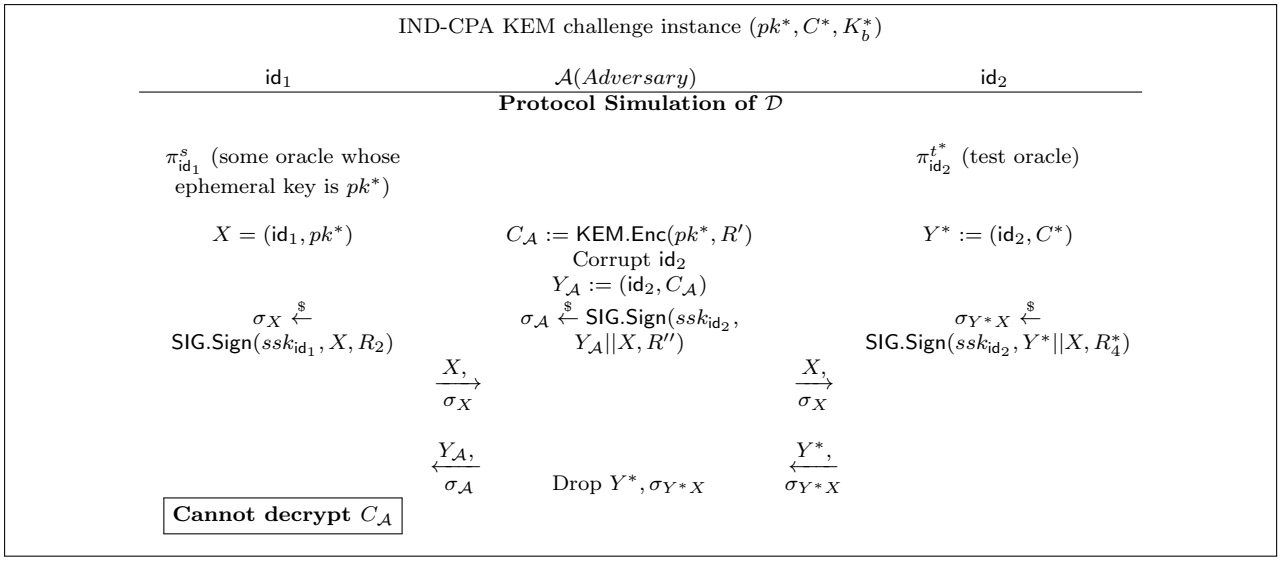

Fig. 3: Reduction Problem of the Generic KF Scheme.

	Peer Setting	NAXOS Trick?	Standard Model?	Security Model	Building Blocks & Assumptions	Ephemeral Key [Initiator][Responder]
BCNP [9]	Pre	No	Yes	CK	IND-CCA KEM EXT, PRF	$[1C_2][1C_2]$
FSXY [19]	Pre	Yes	Yes	CK+	IND-CPA wKEM IND-CCA KEM EXT, PRF	$[1P_1][1C_1]$ $[1C_2][1C_2]$
TMKE	Post	Yes	Yes	eCK-PFS	PG-IND-CPA wKEM OT-IND-CCA KEM SEUF-CMA SIG PRF	$[1P_3][1C_3]$ $[1S][1S]$

Table 1: Comparison.

can break the security of the underlying cryptographic building blocks. Let $\pi^{s^*}_{\mathsf{id}_i}$ be the test oracle and $\pi^{z^*}_{\mathsf{id}_j}$ be the origin oracle of $\pi^{s^*}_{\mathsf{id}_i}$. Since each protocol message contains the identity of the sender, the origin oracle must come from its intended partner (i.e. id_j).

To complete the proof, we need to examine all possible freshness cases formulated by Definition 3. Note that the adversary $\mathcal{A}$ is not allowed to ask RevealKey query to the test oracle or its partner oracle (if it exists). But there might different combinations regarding RevealRand and Corrupt queries. Hence, we here list all possible freshness cases related to RevealRand and Corrupt queries as follows:

- **E1** : If $\pi^{t^*}_{\mathsf{id}_j}$ exists :
  - **C1** – $\mathcal{A}$ did not query $\mathsf{RevealRand}(\mathsf{id}_i, s^*)$ nor $\mathsf{RevealRand}(\mathsf{id}_j, t^*)$.
  - **C2** : $\mathcal{A}$ did not query $\mathsf{Corrupt}(\mathsf{id}_i, pk^*)$ nor $\mathsf{RevealRand}(\mathsf{id}_j, t^*)$.
  - **C3** : $\mathcal{A}$ did not query $\mathsf{Corrupt}(\mathsf{id}_i, pk^*)$ nor $\mathsf{Corrupt}(\mathsf{id}_j, pk^*)$.
  - **C4** : $\mathcal{A}$ did not query $\mathsf{RevealRand}(\mathsf{id}_i, s^*)$ nor $\mathsf{Corrupt}(\mathsf{id}_j, pk^*)$.
- **E2** : If $\pi^{t^*}_{\mathsf{id}_j}$ does not exist :
  - **C5** : $\mathcal{A}$ did not query $\mathsf{Corrupt}(\mathsf{id}_i, pk^*)$ nor $\mathsf{Corrupt}(\mathsf{id}_j)$ prior to the acceptance of $\pi^{s^*}_{\mathsf{id}_i}$.
  - **C6** : $\mathcal{A}$ did not query $\mathsf{RevealRand}(\mathsf{id}_i, s^*)$ nor $\mathsf{Corrupt}(\mathsf{id}_j, pk^*)$ prior to the acceptance of $\pi^{s^*}_{\mathsf{id}_i}$.

However, only one of above cases would occur to the test oracle during the whole simulation of the security experiment. This means we only need to simulate the game under one of above freshness cases each time.

In the following, we proceed the proof via the game-based approach [38]. Let Adv_{ξ} denote the advantage of $\mathcal{A}$ in Game ξ.

Game 0. The first game is the real security experiment. Thus we have that

$$\mathsf{Adv}^{\mathsf{ake}}_{\mathsf{TMKE},\mathcal{A}}(\kappa) = \mathsf{Adv}_0.$$

Game 1. This game proceeds exactly as the previous game, but the challenger aborts if: there is a fresh oracle $\pi^s_{\mathsf{id}_i}$ received a message m ($m = (\mathsf{id}_j, epk_{\mathsf{id}_j}, \sigma_{\mathsf{id}_j})$ or $m = (\mathsf{id}_j, C_{\mathsf{id}_j}, \sigma_{\mathsf{id}_j})$) which is not sent by any oracle of id_j before id_j is corrupt, but $\mathsf{SIG.Vfy}(vk_{\mathsf{id}_j}, \sigma_{\mathsf{id}_j}, m) = 1$.

If the challenger aborts with overwhelming probability, then we could construct a signature forger $\mathcal{F}$ as follows. The forger $\mathcal{F}$ receives as input a public key pk^*, and runs the adversary $\mathcal{A}$ as a subroutine and simulates the challenger for $\mathcal{A}$. It first guesses a coordinate $j \xleftarrow{\$} [\lambda]$ pointing to the public key for which the adversary is able to forge, and sets $pk_{\mathsf{id}_j} = pk^*$. Next $\mathcal{F}$ generates all other long-term public/secret keys honestly as the challenger in the previous game. Then $\mathcal{F}$ proceeds as the challenger in Game 1, except that it uses its chosen-message oracle to generate a signature under pk_{id_j} for the oracles of id_j. While answering the $\mathsf{RevealRand}(\mathsf{id}_j, t)$ query, $\mathcal{F}$ includes rs_s (if any) extracted from the corresponding σ (in terms of AR) for this query.

When $\mathcal{A}$ has forged a signature on behalf of the guessed uncorrupted party id_j, then $\mathcal{F}$ can use it to break the SEUF-CMA security of SIG. Since there are ℓ honest parties, the probability that $\mathcal{F}$ guesses correctly on id_j is larger than $1/\lambda$. Therefore we have that

$$\mathsf{Adv}_0 \leq \mathsf{Adv}_1 + \lambda \cdot \mathsf{Adv}^{\mathsf{seuf\text{-}cma}}_{\mathsf{SIG},\mathcal{F}}(\kappa, d).$$

As a result, the test oracle $\pi^{s^*}_{\mathsf{id}_i}$ in this game always has an origin oracle $\pi^{z^*}_{\mathsf{id}_j}$ before id_j is corrupted. In particular, if the test oracle is initiator then it must have a matching session. Those facts imply that the freshness cases $C5$ and $C6$ never hold in this game.

Game 2. This game proceeds as before, but $\mathcal{C}$ tries to guess the test oracle and its origin oracle and one of the possible freshness cases (from $C1$ to $C4$). Technically, $\mathcal{C}$ aborts if it fails in this guess. Since there are 4 fresh cases and λ parties at all, and at most d oracles for each party, then the probability of a correct guess is at least $1/4(d\lambda)^2$. Thus we have that

$$\mathsf{Adv}_1 \leq 4(d\lambda)^2 \cdot \mathsf{Adv}_2.$$

Game 3. In this game, we first change the computation of the randomness ($rpg^{s^*}_{\mathsf{id}_i}$ or $erk^{s^*}_{\mathsf{id}_i}$) for the test oracle if its owner is guessed to be uncorrupted, i.e., the corresponding $dk^{s^*}_{\mathsf{id}_i}$ is not known to the adversary. Specifically, we do the following modifications. The ephemeral public key $ek^{s^*}_{\mathsf{id}_i}$ of $\pi^{s^*}_{\mathsf{id}_i}$ is not discarded. $\mathcal{C}$ chooses an internal randomness $erk' \xleftarrow{\$} \mathcal{RK}_{\mathsf{wKEM}}$, and computes the ephemeral key as $(c^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}}, K^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}}) := \mathsf{wKEM.Enc}(ek^{s^*}_{\mathsf{id}_i}, erk')$. Then $c^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}}$ will be returned by the $\mathsf{RevealRand}$ query, and K^*_{wKEM} will be parsed as the randomness of OTKEM. If the origin oracle is not corrupted before the acceptance of the test oracle, then we do the same change to it. Due to the ϵ-PG-IND security property of wKEM, we have that

$$\mathsf{Adv}_2 \leq \mathsf{Adv}_3 + 2 \cdot \epsilon \leq \mathsf{Adv}_3 + 2 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{wKEM},\mathcal{N}}(\kappa).$$

Note that the function $\mathsf{wKEM.Dec}$ is deterministic and its input ciphertext $c^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}}$ is chosen uniformly at random (in the origin protocol). If the ephemeral key (i.e., $c^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}}$) is not revealed by the adversary, then the randomness ($rpg^{s^*}_{\mathsf{id}_i}, erk^{s^*}_{\mathsf{id}_i}$) are just random values as the previous game. Thus in this game (and in the following games), we only concentrate on the

oracles whose owner is uncorrupted (but its ephemeral key is revealed).

Game 4. This game proceeds as before, but $\mathcal{C}$ replaces the randomness ($rpg^{s^*}_{\mathsf{id}_i}$ or $erk^{s^*}_{\mathsf{id}_i}$) of the test oracle $\pi^{s^*}_{\mathsf{id}_i}$ with a uniform random, if id_i is guessed to be uncorrupted before $\pi^{s^*}_{\mathsf{id}_i}$ accepts. If there exists an adversary $\mathcal{A}$ which can distinguish this game from the previous game , then we could construct an algorithm $\mathcal{N}$ to break the security of wKEM scheme by making use of $\mathcal{A}$.

In the following, we present the proof for the test party whose long-term secret key is uncorrupted. The algorithm $\mathcal{N}$ receives the challenge public key ek^*_{wKEM} and the challenge values $(K^*_{\mathsf{wKEM}}, C^*_{\mathsf{wKEM}})$ from the wKEM challenger. As for the test oracle, $\mathcal{N}$ parses K^*_{wKEM} as the randomnesses of OTKEM, i.e., $rpg^{s^*}_{\mathsf{id}_i} = K^*_{\mathsf{wKEM}}$ or $erk^{s^*}_{\mathsf{id}_i} = K^*_{\mathsf{wKEM}}$ (depending on the role of id_i). And C^*_{wKEM} is set as the ephemeral secret key of the test oracle, i.e., $c^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}} = C^*_{\mathsf{wKEM}}$. The rest of the simulation is identical to the previous game. Meanwhile, $\mathcal{N}$ answers the oracle queries as before.

If $b = 1$, then K^*_{wKEM} is the decryption of C^*_{wKEM} and the simulation constructed by $\mathcal{N}$ is identical to the previous game. Otherwise the simulation of $\mathcal{N}$ is the same as this game. Thus we have that

$$\mathsf{Adv}_3 \leq \mathsf{Adv}_4 + \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{wKEM},\mathcal{N}}(\kappa, 0).$$

Note that $rpg^{s^*}_{\mathsf{id}_i}$ or $erk^{s^*}_{\mathsf{id}_i}$ in this game is non-related to the ephemeral secret key of the test oracle. This implies the simulator in the following game can choose arbitrary ephemeral secret key $c^{s^*}_{\mathsf{id}_i,\mathsf{wKEM}}$ to answer the $\mathsf{RevealRand}$ query to the test oracle.

Game 5. We change this game by replacing the randomnesses (rpg^* or erk^*) of the uncorrupted origin oracle (of the test oracle), with a uniform random. With the similar argument as in the previous game, we have that

$$\mathsf{Adv}_4 \leq \mathsf{Adv}_5 + \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{wKEM},\mathcal{N}}(\kappa, 0).$$

Up to now, the randomness of each fresh oracle is just uniform random and hidden from the adversary.

Game 6. In this game, the challenger proceeds exactly like the previous game, except that it adds an additional abortion rule. Namely the challenger aborts, if a fresh oracle generates an ephemeral public key (or an ephemeral ciphertext of OTKEM) which has appeared before. If two oracles generate the same ephemeral public key or ephemeral ciphertext with overwhelming probability. This implies that the OTKEM is insecure at all, because of the high collision probability of key generation algorithm. Due to the security of OTKEM, the abortion event must occur with negligible probability. We have that

$$\mathsf{Adv}_5 \leq \mathsf{Adv}_6 + \mathsf{Adv}^{\mathsf{ot\text{-}ind\text{-}cca2}}_{\mathsf{OTKEM},\mathcal{D}}(\kappa, 1).$$

In this game, we can have the fact that the test oracle has a unique origin oracle.

Game 7. In this game, we would like to reduce the security to the OT-IND-CCA2 security of OTKEM. Hence, we change this game from the previous game by replacing the key material $k^{s^*}_{\mathsf{id}_i}$ (generated by the OTKEM scheme) with a random value $\widetilde{k^{s^*}_{\mathsf{id}_i}}$. If there exists an adversary $\mathcal{A}$ which can distinguish this game from the previous game. Then we can use it to build an efficient algorithm $\mathcal{D}$ to break the security of OTKEM.

Again $\mathcal{D}$ simulates the game for $\mathcal{A}$. Recall that, in order to simulate the game appropriately, $\mathcal{D}$ has to guess correctly in advance about the oracles that $\mathcal{A}$ will attack. $\mathcal{D}$ then obtains the challenge public key pk^*_{OTKEM}, challenge ciphertext C^*_{OTKEM} and challenge K^*_{OTKEM} from the

KEM challenger. The goal of $\mathcal{D}$ is to distinguish whether K^*_{OTKEM} is the true key or a random key. With respect to $\mathcal{A}$'s queries, $\mathcal{D}$ answers them as follows:

- $\mathsf{Send}(\mathsf{id}_u, t, m)$: This query is performed as defined in the security model according to the protocol specification. Meanwhile, $\mathcal{D}$ uses the values pk^*_{OTKEM} and C^*_{OTKEM} to simulate the test oracle or its origin oracle (depending on which one is initiator).
- $\mathsf{RevealKey}(\mathsf{id}_j, t)$: Note that this query would be never asked to the test oracle and its partner oracle. Otherwise the test oracle is not fresh. $\mathcal{D}$ can use the secrets chosen by herself to compute the session key honestly following the protocol specification.
  If the test oracle is initiator, then it must have a matching session. In this case, the test oracle and its partner oracle will use the same key material K^*_{OTKEM} to generate the final session key. All other session keys can be computed by the ephemeral secret keys chosen by $\mathcal{D}$. Once the test oracle is responder, the situation is more complicated. We need to correctly simulate the session key of the origin oracle $\pi^{z^*}_{\mathsf{id}_j}$. If $\pi^{z^*}_{\mathsf{id}_j}$ is the partner oracle of the test oracle, then $\mathcal{D}$ computes the same session key for them. If $\pi^{z^*}_{\mathsf{id}_j}$ is not the partner oracle of the test oracle, but it receives C^* as input. Then $\mathcal{D}$ uses the same challenge key K^*_{OTKEM} to generate the session key as $K^{z^*}_{\mathsf{id}_j} = \mathsf{PRF}(K^*_{\mathsf{OTKEM}}, \mathsf{sid}^{z^*}_{\mathsf{id}_j})$. Note that we must have $\mathsf{sid}^{z^*}_{\mathsf{id}_j} \neq \mathsf{sid}^{s^*}_{\mathsf{id}_i}$ in this case. If $\pi^{z^*}_{\mathsf{id}_j}$ is not the partner oracle of the test oracle and it receives C' such that $C' \neq C^*_{\mathsf{OTKEM}}$. Then $\mathcal{D}$ asks its decryption oracle $\mathcal{DEC}(C')$ to obtain the corresponding key k'. The session key is then generated as $K^{z^*}_{\mathsf{id}_j} = \mathsf{PRF}(k', \mathsf{sid}^{z^*}_{\mathsf{id}_j})$. In a nutshell, we can appropriately compute the session key of $\pi^{z^*}_{\mathsf{id}_j}$ in all cases.
- $\mathsf{RevealRand}(\mathsf{id}_i, s)$: $\mathcal{D}$ responds with the ephemeral keys chosen by herself.
- $\mathsf{Corrupt}(\mathsf{id}_i, pk_{\mathsf{id}_i})$ and $\mathsf{RegCorrupt}(\mathsf{id}_i, pk_{\mathsf{id}_i})$: $\mathcal{D}$ proceeds them as the original AKE challenger.
- $\mathsf{Test}(\mathsf{id}^*_i, s^*)$: $\mathcal{D}$ changes the returned key of this query as $K^{s^*}_{\mathsf{id}_i} = \mathsf{PRF}(k^*, \mathsf{sid}^{s^*}_{\mathsf{id}_i})$.

The simulation of $\mathcal{D}$ is perfect so far. If K^*_{OTKEM} is a random key then the game is equivalent to this game. Otherwise it is identical to the previous game. Applying the OT-IND-CCA2 security of OTKEM, we therefore obtain that

$$\mathsf{Adv}_6 \leq \mathsf{Adv}_7 + \mathsf{Adv}^{\mathsf{ot\text{-}ind\text{-}cca2}}_{\mathsf{OTKEM},\mathcal{D}}(\kappa, 1).$$

Game 8. In this game, the function $\mathsf{PRF}(\widetilde{k^{s^*}_{\mathsf{id}_i}}, \cdot)$ is replaced with a truly random function for the test oracle and its partner oracle (if it exists). Notice that the secret seed $\widetilde{k^{s^*}_{\mathsf{id}_i}}$ of the test oracle is changed to a truly random value due to the previous game. And the test oracle has a session identifier sid^* (i.e., the input of PRF) which is only shared with its partner oracle. Note that the origin oracle of the test oracle would deal with at most one chosen messages attack. When the origin oracle is not partnered to the test oracle but they share the same secret seed, we could ask a PRF oracle query $\mathcal{FN}(\mathsf{sid}^{z^*}_{\mathsf{id}_j})$ to compute the session key of the origin oracle.

Then any PPT adversary $\mathcal{A}$ distinguishing Game 8 from Game 7 can be used to break PRF. In terms of the security of PRF, we therefore have that

$$\mathsf{Adv}_7 \leq \mathsf{Adv}_8 + \mathsf{Adv}^{\mathsf{ind\text{-}cma}}_{\mathsf{PRF},\mathcal{B}}(\kappa, 1).$$

Note that in this game the session key returned by the Test query is totally a truly random value which is independent of the bit b chosen by the Test query and any messages. Thus, the advantage that the adversary wins in this game is $\mathsf{Adv}_8 = 0$.

Put altogether the probabilities from Game 0 to Game 8, we obtained the overall result of this theorem.

D Proof of Theorem 3: OT-IND-CCA2 Security of the Proposed OTKEM

In this proof, we are going to reduce the security of our proposed protocol to that of the underlying building blocks and the Ring-LWE problem. The proof is proceeded following the game-based approach. Let $C^* := (Y^*, u^*, spk^*, \sigma^*)$ denote the ciphertext generated by challenge query.

Game 0. The first game is the real security experiment. Thus we have that

$$\mathsf{Adv}^{\mathsf{ot\text{-}ind\text{-}cca2}}_{\mathsf{OTKEM},\mathcal{D}}(\kappa, 1) = \mathsf{Adv}_0.$$

Game 1. This game proceeds exactly as the previous game, but the challenger aborts if: the decryption oracle received a ciphertext $C' = (Y', u', spk^*, \sigma')$ such that $(Y', u') \neq (Y^*, u^*)$, but $\mathsf{SIG.Vfy}(spk^*, \sigma_{\mathsf{id}_j},$
$pk||Y'||spk^*||u) = 1$.

If the challenger aborts with overwhelming probability, then we could construct a signature forger $\mathcal{F}$ as follows. The forger $\mathcal{F}$ receives as input a public key vk^*, and runs the adversary $\mathcal{A}$ as a subroutine and simulates the challenger for $\mathcal{A}$. It will set $spk^* := vk^*$ in the challenge query. Then $\mathcal{F}$ proceeds as the challenger in the Game 1, except that it asks its signing oracle to generate a signature under vk^* for the challenge query. When $\mathcal{A}$ has forged a signature, then $\mathcal{F}$ can use it to break the SEUF-CMA security of OTS. Therefore we have that

$$\mathsf{Adv}_0 \leq \mathsf{Adv}_1 + \mathsf{Adv}^{\mathsf{seuf\text{-}cma}}_{\mathsf{OTS},\mathcal{F}}(\kappa, 1).$$

As a result, the challenge value u^* is tightly bound together to other values in the challenge ciphertext. Then the adversary is unable to replace u^* with another u' of her own choice.

Game 2. This game proceeds as the previous game, except that the simulator aborts if the decryption oracle receives a ciphertext which generates the same hash value of the challenge query. When the event does occur, we can easily construct an algorithm to break the target collision resistant hash function TCRHF. Hence, we have that

$$\mathsf{Adv}_1 \leq \mathsf{Adv}_2 + \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{TCRHF},\mathcal{H}}(\kappa).$$

Game 3. Recall that the hash value of the challenge query is unique due to the results of the previous games. The secret key sk^* and the public key $pk^* = \{S^*_{1,0}, S^*_{1,1}, \ldots, S^*_{\mu,0}, S^*_{\mu,1}\}$ (used to compute the session key) are programmed by the 'bits' $(h(1), \ldots, h(\mu))$ of a hash value. The hash value h' generated by the decryption oracle must be distinct to h^* of the challenge query, i.e., $h' \neq h^*$. In this case, we can embed the Ring-LWE challenge value into the public key and correctly answer the decryption oracle query. Since $h' \neq h^*$, there must exist one bit, say with coordinate τ^* in h^*, is distinct to the τ^*-th bit of h'. Hence, we require that the challenger should guess the coordinate τ^* and the exact value of $h^*(\tau^*)$. If it fails in the above guesses, then the challenger aborts. The successful probability of the guess is at least $\frac{1}{2\mu}$.

$$\mathsf{Adv}_2 \leq 2\mu \cdot \mathsf{Adv}_3.$$

Game 4. In this game, we replace the $(\tau^*, h^*(\tau^*))$-th public key with a random value, i.e., $S^*_{\tau^*,h^*(\tau^*)} \xleftarrow{\$} R_q$. If $\mathcal{D}$ is able to distinguish this game from the previous game, then we can make use of $\mathcal{D}$ to build a Ring-LWE solver $\mathcal{E}$. As the tuple $(a, S^*_{\tau^*,h^*(\tau^*)}$ is either a real sample from the distribution $A_{s_{\tau^*,h^*(\tau^*)},\mathcal{X}}$ for some random $s_{\tau^*,h^*(\tau^*)} \xleftarrow{\$} \mathcal{X}$ (which is unknown to $\mathcal{E}$), or a uniform random from $R_q \times R_q$. In the reduction, $\mathcal{E}$ can generate all other public key values honestly

with the secrets chosen by herself. Note that the public keys would be divided into two parts in terms of the hash value h^*. The first part contains the values that the adversary $\mathcal{E}$ knows all corresponding secrets of her own choice. The second part contains the key $S^*_{\tau^*,h^*(\tau^*)}$ which is only used by the challenge query. These facts imply that $\mathcal{E}$ is able to answer the decryption oracle using the secret keys of her own choice. But the session key of the challenge query would be generated involving $(a, S^*_{\tau^*,h^*(\tau^*)})$. Due to the computational complexity of Ring-LWE, we have that

$$\mathsf{Adv}_3 \leq \mathsf{Adv}_4 + \mathsf{Adv}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa).$$

Game 5. In this game, we replace Y^*, v^* and K^* with random values, i.e., $(Y^*, v^*, K^*) \xleftarrow{\$} R_q \times R_q \times R_2$. Again, if $\mathcal{D}$ is able to distinguish this game from the previous game, then we can make use of it to solve the Ring-LWE problem. Specifically, $\mathcal{E}$ takes as input two pairs $(a, w), (b, v) \in R_q \times R_q$, and sets $a^* := a$, $Y^* := w$ and $S_{\tau^*,\iota^*} := g^{-1}b$. For the challenge query, $\mathcal{E}$ computes $\bar{v^*} = \mathsf{HLP}(v)$, and $K^* = \lfloor \bar{v^*} \rceil$. Now if the inputs to $\mathcal{E}$ is sampled from $A_{r^*,\mathcal{X}}$, the output of $\mathcal{E}$ is distributed exactly as the previous game. Since (a, b) are independently random, and we have $Y^* = w = a \cdot r^* + e$ and $v = b \cdot r + g \cdot r \cdot (\sum^{\mu}_{\eta=1,\eta\neq\tau^*} S^*_{\eta,h^*(\eta)}) + f$ for independent f. On the other side, if the inputs given to $\mathcal{E}$ are uniform random over $R_q \times R_q$ and independent. Then the output of $\mathcal{E}$ is distributed exactly as in this game. As (a, b, w, v) are uniform random and independent, K^* is also uniform random. Meanwhile, the decryption oracle will be simulated as the previous game.

The simulation of $\mathcal{E}$ is perfect so far. If $w = a \cdot r^* + e$ then the game is equivalent to the previous game, otherwise the game is identical to this game. Due to the hardness of Ring-LWE problem, we therefore obtain that

$$\mathsf{Adv}_4 \leq \mathsf{Adv}_5 + \mathsf{Adv}^{\mathsf{rlwe}}_{R_q,\mathcal{X},\mathcal{E}}(\kappa).$$

Note that in this game the session key returned the challenge query is totally a truly random value which is independent of the bit b. Thus, the advantage that the adversary wins in this game is $\mathsf{Adv}_5 = 0$.

Put altogether the probabilities from Game 0 to Game 5, we obtained the overall result of this theorem.

E Other Related Work

Besides KEM, non-interactive key exchange (NIKE) [18] is recently applied to construct generic TMKE, e.g., [6, 39]. In PKC 2015, Bergsma et al. [6] introduced a very interesting generic ORKE scheme (which will be referred to as BJS scheme) from CKS-light NIKE scheme [18] and SEUF-CMA SIG scheme. The BJS scheme is the first eCK-PFS secure [13] generic ORKE construction in the standard model. In 2016, Yang et al. [39] proposed a simpler NIKE and SIG based ORKE scheme (which will be referred to as YLLLL scheme). In particular, the YLLLL scheme requires weaker assumptions on both NIKE and SIG in contrast to the BJS scheme. In particular, the SIG scheme only needs to provide strong existential unforgeability under weak chosen message attacks (SEUF-WCMA). Therefore, the YLLLL scheme turns out to have more protocol instantiations and to be much more efficient than the BJS scheme.

One important advantage of these NIKE based TMKE schemes (in contrast to the BCNP scheme and the FSXY scheme) is that they are truly TMKE which can work under the post-specified peer model [10]. In the post-specified peer model, a party does not know any cryptographic information (such as public key) about the receiver at the session initiation phase. As pointed out by Menezes and Ustaoglu [30], a protocol working under the pre-specified peer model

(such as the BCNP scheme and the FSXY scheme) may need an extra communication round to exchange identity and public keys. Moreover, we note that both the BJS protocol and the Yang's protocol are suggested to instantiate the NIKE scheme from [18] based on the traditional Diffie-Hellman based hard problems (We also refer reader to [34, 40] for more strongly secure Diffie-Hellman TMKE schemes without random oracles in the post-specified peer model). In this paper, we just focus on the generic TMKE construction which has post-quantum instantiations in the post-specified peer model.

Recently, there are several attempts have been made to build the lattice-based key exchange, e.g., [36, 8, 41, 7]. In PQCrypto 2014, Peikert [36] introduced an IND-CPA KEM based key exchange protocol in the post-specified peer modelbased on Ring-LWE. Peikert's construction [36] enjoys explicit authentication based on digital signature and message authentication code (MAC). Bos et al. [8] proposed an instantiation of Peikert's protocol, and integrated it into the TLS to provide post-quantum security. In CCS 2016, Bos et al. further developed key exchange for the TLS based on the Ring-LWE problem [37].

However, all the above protocols require multiple communication rounds (more than 2-pass) to exchange protocol messages. As introduced in Section 1, these protocols cannot be used for securing asynchronous communication in contrast to TMKE.

Moreover, the securities of Bos et al.'s schemes [8, 7] are shown in the ACCE security model [22] which is only developed from the Bellare-Rogaway (BR) model [5]. In 2015, Zhang et al.[41] proposed an efficient HMQV like TMKE protocol from Ring-LWE. However, this protocol is proved only with random oracles in a variant of BR model [5] with wPFS. In contrast, our constructions are primarily motivated to provide eCK-PFS security (which covers much more attacks than the BR security, such as KCI attacks, ephemeral key leakage attacks, UKS attacks, and PFS attacks, etc.) in the standard model. Although the instantiation of our generic TMKE in the standard model might be less efficient, we can also instantiate it with the IND-CCA2 secure KEM (e.g. [36]) in the random oracle model to obtain better performance. In addition, the security of the Zhang et al.'s scheme was recently challenged by Gong and Zhao [42].