Understanding and Constructing AKE via Double-Key Key Encapsulation Mechanism

Haiyang Xue[1,2,3], Xianhui Lu[1,2,3(✉)], Bao Li[1,2,3], Bei Liang[4], and Jingnan He[1,2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{hyxue12,xhlu}@is.ac.cn
[2] Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, Beijing, China
[3] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[4] Chalmers University of Technology, Gothenburg, Sweden

Abstract. Motivated by abstracting the common idea behind several implicitly authenticated key exchange (AKE) protocols, we introduce a primitive that we call double-key key encapsulation mechanism (2-key KEM). It is a special type of KEM involving two pairs of secret-public keys and satisfying some function and security property. Such 2-key KEM serves as the core building block and provides alternative approaches to simplify the constructions of AKE. To see the usefulness of 2-key KEM, we show how several existing constructions of AKE can be captured as 2-key KEM and understood in a unified framework, including widely used HMQV, NAXOS, Okamoto-AKE, and FSXY12-13 schemes. Then, we show (1) how to construct 2-key KEM from concrete assumptions, (2) how to adapt the classical Fujisaki-Okamoto transformation and KEM combiner to achieve the security requirement of 2-key KEM, (3) an elegant Kyber-AKE over lattice using the improved Fujisaki-Okamoto technique.

Keywords: Authenticated key exchange · CK model
Key encapsulation mechanism

1 Introduction

Key exchange (KE), which enables two parties to securely establish a common session key while communicating over an insecure channel, is one of the most important and fundamental primitives in cryptography. After the introduction of Diffie-Hellman key exchange in [12], cryptographers have devised a wide selection of the KE with various use-cases. One important direction is authenticated key exchange (AKE). The main problems that the following works focus on are

T. Peyrin and S. Galbraith (Eds.): ASIACRYPT 2018, LNCS 11273, pp. 158–189, 2018.
https://doi.org/10.1007/978-3-030-03329-3_6

specified as security models [5–7,15,24], efficient and provably-secure realizations [1–3,7,15,16,22,24–27,29,34,35].

In an AKE protocol, each party has a pair of secret-public keys, a *static/long-term public key* and the corresponding *static/long-term secret key*. The static public key is interrelated with a party's identity, which enables the other parties to verify the authentic binding between them. A party who wants to share information with another party generates ephemeral one-time randomness which is known as *ephemeral secret keys*, computes *session state* (which is originally not explicitly defined [7], but nowadays it is generally agreed [15,24] that the session state should at least contain ephemeral secret keys) from ephemeral and static secret keys and incoming message, then outputs corresponding *ephemeral public outgoing message*. Then each party uses their static secret keys and the ephemeral secret keys along with the transcripts of the session to compute a shared *session key*.

Many studies have investigated the security notion of AKE including BR model and Canetti-Krawczyk (CK) model [7]. Fujioka *et al.* [15] re-formulated the *desirable* security notion of AKE in [23], including resistance to KCI (key compromise impersonation attack), wPFS (weak perfect forward attack) and MEX (maximal exposure attack), as well as provable security in the CK model, and called it the CK^+ security model. LaMacchia *et al.* [24] also proposed a very strong security model, called the eCK model. The CK model and the eCK model are incomparable [6], and the eCK model is not stronger than the CK model while the CK^+ model is [15]. However, each of these two models, eCK and CK^+ can be theoretically seen as a strong version of the AKE security model.

To achieve a secure AKE in one of the above security models (CK, CK^+, eCK), the solutions are divided into two classes: explicit AKE and implicit AKE. The solution of explicit AKE is to explicitly authenticate the exchanged messages between the involved parties by generally using additional primitives *i.e.,* signature or MAC to combine with the underlying KE, such as IKE [8], SIGMA [22], TLS [2,21] etc.; while the solution of implicit AKE initiated by [25], is to implicitly authenticate each party by its unique ability so as to compute the resulted session key. These kinds of implicit AKE schemes include (H)MQV [23,26], Okamoto [27,28], NAXOS [24], OAKE [34], FSXY variants [1,15,16,33], and AKE from lattice assumptions [3,35].

Motivation. In this paper, we focus on the second class, *i.e.,* constructions of *implicit AKE*. Based on different techniques and assumptions, many implicit AKE protocols have been proposed in recent years [15,16,23,24,27,28,34].

However, the constructing techniques and methods of the existing implicit AKE protocols are somewhat separate and the study on the highly accurate analysis of AKE's requirement for the building block is critically in a shortage, especially for the exact underlying primitives that serve as fundamental building blocks and capture the common idea and technique behind the constructions and security proofs of AKE. On the contrary, with respect to explicit AKE Canetti and Krawcayk [8,22] gave the frame of "SIGin-and-MAc" (later extended by [29]) which provides a good guideline for designing explicit AKE.

In fact, Boyd *et al.* [1] and Fujioka *et al.* [15,16] initiated the research on studying frameworks of implicit AKE. Boyd *et al.* firstly noticed the connection between AKE and key encapsulation mechanism (KEM), then Fujioka *et al.* provided CK^+ secure AKE protocols from chosen ciphertext (CCA) secure KEM in the random oracle and standard models. Although the paradigm of connecting the AKE with KEM is of great significance, it can not be applied to explain many widely-used and well-known constructions of AKE such as HMQV and its variant [23,34] which are built on the challenge-respond signature; AKE protocol in [27] which results from universal hash proof [10]; as well as NAXOS [24].

Hence, one of the important problems on AKE is to give an even more general framework for constructing AKE that is able to not only unify the existing structures of AKE protocol as much as possible, but also to systemize and simplify the construction and analysis methods of AKE protocol. It will be useful and helpful for understanding the existing works and future studying on formalization of the AKE construction under a unified framework with some well-studied and simple cryptographic primitive as building block.

1.1 Our Contributions

- Based on the above motivations and observations, we introduce *double-key key encapsulation mechanism* (2-key KEM) and its secure notions, *i.e.,* [IND/OW-CCA, IND/OW-CPA] security. We also show its distinction with previous similar notions.
- Based on the [IND/OW-CCA, IND/OW-CPA] secure 2-key KEM, we present unified frames of CK^+ secure AKE, which in turn conceptually capture the common pattern for the existing constructions and security proof of AKE, including well-known HMQV [23], NAXOS [24], Okamoto-AKE [27,28], and FSXY12 [15], FSXY13 [16].
- We investigate the constructions of 2-key KEM based on concrete assumptions. We also show the failure of implying [IND/OW-CCA, IND/OW-CPA] secure 2-key KEM from KEM combiner and the classical Fujisaki-Okamoto (FO) transformation. Hence, with a slight but vital modification by taking public key as input to the hash step we provide improved KEM combiner and improved FO to adapt them in our 2-key KEM setting.
- Equipped with 2-key KEM and our frame above, we propose a post-quantum AKE based on Module-LWE assumption, which consumes less communications than Kyber [3] using frame of FSXY13 [16].

2-Key Key Encapsulation Mechanism. Generally, the 2-key KEM scheme is a public key encapsulation with two pairs of public and secret keys, but the main distinctions are the functionality and security.

The encapsulation and decapsulation algorithms: instead of taking as input single public key to generate a random key K and a ciphertext C and single secret key to decapsulate ciphertext C, each algorithm takes two public keys (pk_1, pk_0) to generate (C, K) and only with both two secret keys (sk_1, sk_0) the decapsulation algorithm can decapsulate C.

We define the security notion of 2-key KEM/PKE in the attacking model [IND/OW-CCA, IND/OW-CPA] which captures the idea that the 2-key KEM is secure under one secret-public key pair even if another pair of secret-public key is generated by the adversary. Informally, the [IND/OW-CCA, $\cdot$] denotes the security model where adversary $\mathcal{A}$ aims to attack the ciphertext under pk_1 and pk_0^* (with its control over the generation of pk_0^*), and it is allowed to query a strong decapsulation oracle that will decapsulate the ciphertext under pk_1 and arbitrary pk_0' (generated by challenger); while [$\cdot$, IND/OW-CPA] denotes the security model where adversary $\mathcal{B}$ aims to attack the ciphertext under pk_0 and pk_1^* (with its control over the generation of pk_1^*). We say a 2-key KEM is [IND/OW-CCA, IND/OW-CPA] secure if it is both [IND/OW-CCA, $\cdot$] and [$\cdot$, IND/OW-CPA] secure.

Compared with classical definition of CCA security, the [CCA, $\cdot$] adversary of 2-key KEM has two main enhancements: (1) one of the challenge public keys pk_0^*, under which the challenge ciphertext is computed, is generated by the adversary; (2) the adversary is allowed to query a strong decryption oracle, and get decapsulation of the ciphertext under arbitrary public keys (pk_1^*, pk_0') where pk_0' is generated by the challenger.

AKE from 2-Key KEM. Equipped with [IND/OW-CCA, IND/OW-CPA] 2-key KEM, by taking pk_1 as static public key and pk_0 as ephemeral public key, we give several general frames of CK^+ secure AKE, AKE, $\mathsf{AKE}_{\mathsf{ro\text{-}pkic\text{-}lr}}$ and $\mathsf{AKE}_{\mathsf{std}}$, depending on different tricks. The CK^+ security of our AKE is decomposed to the [IND/OW-CCA, $\cdot$] security (corresponding to KCI and MEX security) and [$\cdot$, IND/OW-CPA] security (corresponding to wPFS) of 2-key KEM. Furthermore, to resist the leakage of partial randomness, a function $f(ssk_B, esk_B)$ is required so that if one of ssk_B and esk_B is leaked $f(ssk_B, esk_B)$ is still computationally indistinguishable with a random string.

In Table 1 we summarize which one of our general frames is used to explain which one of the existing AKE protocols by employing the specific tricks and assumptions. Our general protocols capture the common idea of constructing CK^+ secure AKE. And depending on 2-key KEM and different tricks, it facilitates a number of instantiations, including HMQV [23], NAXOS [24], Okamoto [27], FSXY12 [15], and FSXY13 [16].

By considering an AKE protocol in such a framework based on 2-key KEM, the complicated security proofs of existing AKE is decomposed into several smaller cases each of which is easier to work with. Moreover, this general scheme not only explains previous constructions, but also yields efficient AKE from lattice problems. After giving [IND-CPA, IND-CPA] twin-kyber under Module-LWE assumption, we obtain a *post-quantum AKE* with less communications.

Constructions of 2-Key KEM. In addition to show that existing AKEs imply [CCA, CPA] secure 2-key KEM, we investigate the general constructions.

Putting Public Key in the Hashing or PRF step. The Fujisaki-Okamoto (FO) [14,18] transformation and KEM combiner are general techniques of classical CCA security for one-key KEM. We show the failure of implying [IND/OW-

Table 1. The unification of AKEs. Comb. is the abbreviation for combiner. GDH is the Gap-DH assumption. RO is the notion of random oracle. Std is the shortened form of standard model. πPRF means the pairwise-independent random source PRF [28]

Frameworks	Models	Concrete AKEs	Assumptions	Tricks
AKE	RO	FSXY13 [16], Kyber [3]	OW-CCA	Modified KEM Comb.
	RO	AKE-2Kyber (Sect.7)	M-LWE	Modified FO
$\mathsf{AKE}_{\text{ro-pkic-lr}}$	RO	HMQV [23] OAKE [34]	GDH, KEA1	Remarks 1–3
	RO	NAXOS [24]	GDH	Remarks 1, 2
$\mathsf{AKE}_{\text{std}}$	Std	FSXY12 [15]	IND-CCA	Modified KEM Comb.
	Std	Okamoto [28]	DDH, πPRF	Twisted PRF

CCA, IND/OW-CPA] secure 2-key KEM from KEM combiner and the classical FO transformation by giving particular attacks on concrete schemes. Hence, we show that with a slight but vital modification, when extracting encapsulated key, by taking public key as input to the hash or PRF step, the modified KEM combiner and FO transformation work for 2-key KEM.

1.2 Strong Point of the AKE via 2-Key KEM

The main advantage of our contributions is that we use a non-interactive primitive to handle the complex requirement of interactive protocols. The functionality and security requirements of [CCA, CPA] secure 2-key KEM are relatively easier to work with and understand. As it is known, in AKE we have to consider complex and diverse adversaries. However, when considering the AKE under our unified framework based on 2-key KEM, all the attacking strategies in CK^+ model can be simplified to the singular security of 2-key KEM.

The non-interactive 2-key KEM helps us to highly simplify the constructions for AKE as well as to understand the essential working mechanism. In fact, KEM is relatively well-studied and intensively analyzed. Following the first practical CCA secure PKE [9], there have been a number of CCA secure PKE/KEM schemes based on both concrete assumptions [3,9,30,31] and general cryptographic primitives [11,19,30]. Therefore, it is possible for us to employ the established and nature technique of classical KEM to construct 2-key KEM, and further AKE.

2 Preliminary

For a variable x, if x is a bit string, denote $[x]_i$ as the i-th bit of x; if x is a polynomial, denote $[x]_i$ as the i-th coefficient of x; if x is a sets of vectors (with string or number) denote $[x]_i$ as the sets of all i-th element of vectors in x;

2.1 CK$^+$ Security Model

We recall the CK$^+$ model introduced by [23] and later refined by [15,16], which is a CK [7] model integrated with the weak PFS, resistance to KCI and MEX properties. Since we focus on **two-pass protocols** in this paper, for simplicity, we show the model specified to two pass protocols.

In AKE protocol, U_i denotes a party indexed by i, who is modeled as probabilistic polynomial time (PPT) interactive Turing machines. We assume that each party U_i owns a static pair of secret-public keys (ssk_i, spk_i), where the static public key is linked to U_i's identity, using some systems i.e. PKI, such that the other parties can verify the authentic binding between them. We do not require the well-formness of static public key, in particular, a corrupted party can adaptively register any static public key of its choice.

Session. Each party can be activated to run an instance called a *session*. A party can be activated to initiate the session by an incoming message of the forms $(\Pi, \mathcal{I}, U_A, U_B)$ or respond to an incoming message of the forms $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, where Π is a protocol identifier, $\mathcal{I}$ and $\mathcal{R}$ are role identifiers corresponding to *initiator* and *responder*. Activated with $(\Pi, \mathcal{I}, U_A, U_B)$, U_A is called the session *initiator*. Activated with $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, U_B is called the session *responder*.

According to the specification of AKE, the party creates randomness which is called *ephemeral secret key*, computes and maintains a *session state*, and completes the session by outputting a session key and erasing the session state. Note that Canetti-Krawczyk [7] defines session state as session-specific secret information but leaves it up to a protocol to specify which information is included in session state; LaMacchia et al. [24] explicitly set all random coins used by a party in a session as session-specific secret information and call it *ephemeral secret key*. We require that the session state at least contains the ephemeral secret key.

A session may also be aborted without generating a session key. The initiator U_A creates a session state and outputs X_A, then may receive an incoming message of the forms $(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ from the responder U_B, then may computes the session key SK. On the contrary, the responder U_B outputs X_B, and may compute the session key SK. We say that a session is *completed* if its owner computes the session key.

A session is associated with its owner, a peer, and a session identifier. If U_A is the initiator, the session identifier is $\mathsf{sid} = (\Pi, \mathcal{I}, U_A, U_B, X_A)$ or $\mathsf{sid} = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$, which denotes U_A as an owner and U_B as a peer. If U_B is the responder, the session is identified by $\mathsf{sid} = (\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$, which denotes U_B as an owner and U_A as a peer. The *matching session* of $(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ is $(\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$ and vice versa.

Adversary. The adversary $\mathcal{A}$ is modeled in the following to capture real attacks in open networks.

- Send(message): $\mathcal{A}$ could send message in one of the forms: $(\Pi, \mathcal{I}, U_A, U_B)$, $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, or $(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$, and obtains the response.

- $\mathsf{SessionKeyReveal}(\mathsf{sid})$: if the session sid is completed, $\mathcal{A}$ obtains the session key SK for sid.
- $\mathsf{SessionStateReveal}(\mathsf{sid})$: The adversary $\mathcal{A}$ obtains the session state of the owner of sid if the session is not completed. The session state includes all ephemeral secret keys and intermediate computation results except for immediately erased information but does not include the static secret key.
- $\mathsf{Corrupt}(U_i)$: By this query, $\mathcal{A}$ learns all information of U_A (including the static secret, session states and session keys stored at U_A); in addition, from the moment U_A is corrupted all its actions may be controlled by $\mathcal{A}$.

Freshness. Let $\mathsf{sid}^* = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ or $(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ be a completed session between honest users U_A and U_B. If the matching session of sid^* exists, denote it by $\overline{\mathsf{sid}^*}$. We say session sid^* is fresh if $\mathcal{A}$ does not queries: (1) $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$, $\mathsf{SessionKeyReveal}(\mathsf{sid}^*)$, and $\mathsf{SessionStateReveal}(\overline{\mathsf{sid}^*})$, $\mathsf{SessionKeyReveal}(\overline{\mathsf{sid}^*})$ if $\overline{\mathsf{sid}^*}$ exists; (2) $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$ and $\mathsf{SessionKeyReveal}(\mathsf{sid}^*)$ if $\overline{\mathsf{sid}^*}$ does not exist.

Security Experiment. The adversary $\mathcal{A}$ could make a sequence of the queries described above. During the experiment, $\mathcal{A}$ makes the query of $\mathsf{Test}(\mathsf{sid}^*)$, where sid^* must be a fresh session. $\mathsf{Test}(\mathsf{sid}^*)$ select random bit $b \in_U \{0, 1\}$, and return the session key held by sid^* if $b = 0$; and return a random key if $b = 1$.

The experiment continues until $\mathcal{A}$ returns b' as a guess of b. The adversary $\mathcal{A}$ wins the game if the test session sid^* is still fresh and $b' = b$. The advantage of the adversary $\mathcal{A}$ is defined as $\mathsf{Adv}_{\Pi}^{ck+}(\mathcal{A}) = \Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}$.

Definition 1. *We say that a AKE protocol* Π *is secure in the* CK^+ *model if the following conditions hold:*
(Correctness:) if two honest parties complete matching sessions, then they both compute the same session key except with negligible probability.
(Soundness:) for any PPT $\mathcal{A}$, $\mathsf{Adv}_{\Pi}^{ck+}(\mathcal{A})$ *is negligible for the test session* sid^*,

1. *the static secret key of the owner of* sid^* *is given to* $\mathcal{A}$, *if* $\overline{\mathsf{sid}^*}$ *does not exist.*
2. *the ephemeral secret key of the owner of* sid^* *is given to* $\mathcal{A}$, *if* $\overline{\mathsf{sid}^*}$ *does not exist.*
3. *the static secret key of the owner of* sid^* *and the ephemeral secret key of* $\overline{\mathsf{sid}^*}$ *are given to* $\mathcal{A}$, *if* $\overline{\mathsf{sid}^*}$ *exists.*
4. *the ephemeral secret key of* sid^* *and the ephemeral secret key of* $\overline{\mathsf{sid}^*}$ *are given to* $\mathcal{A}$, *if* $\overline{\mathsf{sid}^*}$ *exists.*
5. *the static secret key of the owner of* sid^* *and the static secret key of the peer of* sid^* *are given to* $\mathcal{A}$, *if* $\overline{\mathsf{sid}^*}$ *exists.*
6. *the ephemeral secret key of* sid^* *and the static secret key of the peer of* sid^* *are given to* $\mathcal{A}$, *if* $\overline{\mathsf{sid}^*}$ *exists.*

As indicated in Table 2, the CK^+ model captures all non-trivial patterns of exposure of static and ephemeral secret keys listed in Definition 1, and these ten cases cover wPFS, resistance to KCI, and MEX.

Table 2. The behavior of AKE adversary in CK$^+$ model. $\overline{\mathsf{sid}^*}$ is the matching session of sid^*, if it exists. "Yes" means that there exists $\overline{\mathsf{sid}^*}$, "No" means do not. $ssk_A(ssk_B)$ means the static secret key of $A(B)$. $esk_A(esk_B)$ is the ephemeral secret key of $A(B)$ in sid^* or $\overline{\mathsf{sid}^*}$ if there exists. "$\surd$" means the secret key may be revealed to adversary, "$\times$" means is not. "-" means the secret key does not exist

Event	Case	sid^*	$\overline{\mathsf{sid}^*}$	ssk_A	esk_A	esk_B	ssk_B	Security
E_1	1	A	No	$\surd$	$\times$	-	$\times$	KCI
E_2	2	A	No	$\times$	$\surd$	-	$\times$	MEX
E_3	2	B	No	$\times$	-	$\surd$	$\times$	MEX
E_4	1	B	No	$\times$	-	$\times$	$\surd$	KCI
E_5	5	A or B	Yes	$\surd$	$\times$	$\times$	$\surd$	wPFS
E_6	4	A or B	Yes	$\times$	$\surd$	$\surd$	$\times$	MEX
$E_{7\text{-}1}$	3	A	Yes	$\surd$	$\times$	$\surd$	$\times$	KCI
$E_{7\text{-}2}$	3	B	Yes	$\times$	$\surd$	$\times$	$\surd$	KCI
$E_{8\text{-}1}$	6	A	Yes	$\times$	$\surd$	$\times$	$\surd$	KCI
$E_{8\text{-}2}$	6	B	Yes	$\surd$	$\times$	$\surd$	$\times$	KCI

3 2-Key Key Encapsulation Mechanism and Basic Results

3.1 2-Key Key Encapsulation Mechanism

Generally, a double-key (2-key) KEM is a public key encapsulation with two pairs of public and secret keys. Formally, a 2-key KEM $\mathsf{2KEM} = (\mathsf{KeyGen1}, \mathsf{KeyGen0}, \mathsf{Encaps}, \mathsf{Decaps})$ is a quadruple of PPT algorithms together with a key space $\mathcal{K}$.

- $\mathsf{KeyGen1}(\lambda, pp)$: on inputs security parameter λ, and public parameters pp, output a pair of public-secret keys (pk_1, sk_1). In order to show the randomness that is used, we denote key generation algorithm as $\mathsf{KeyGen1}(\lambda, pp; r)$. For simplicity, sometimes we omit the input security parameter λ and public parameter pp and denote it as $\mathsf{KeyGen1}(r)$ directly.
- $\mathsf{KeyGen0}(\lambda)$: on inputs security parameter λ output a pair of public and secret keys (pk_0, sk_0).
- $\mathsf{Encaps}(pk_1, pk_0; \mathsf{aux_e})$: on input public keys pk_1, pk_0 and auxiliary input $\mathsf{aux_e}$ (if there is), output ciphertext c and encapsulated key k in key space $\mathcal{K}$. Sometimes, we explicitly add the randomness r and denote it as $\mathsf{Encaps}(pk_1, pk_0, r; \mathsf{aux_e})$.
- $\mathsf{Decaps}(sk_1, sk_0, c; \mathsf{aux_d})$: on input secret keys sk_0, sk_1, auxiliary input $\mathsf{aux_d}$ (if there is) and c, output key k.

Correctness. For $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}(\lambda, pp)$, $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}(\lambda, pp)$ and $(c, k) \leftarrow \mathsf{Encaps}(pk_1, pk_0)$, we require that $\mathsf{Decaps}(sk_1, sk_0, c) = k$ holds with all but negligible probability.

Security. We consider two kinds of security *i.e.,* indistinguishability and one-wayness in the attacking model $[\mathsf{ATK}_1, \mathsf{ATK}_0]$. More precisely, in our $[\mathsf{ATK}_1, \mathsf{ATK}_0]$ security model for 2KEM, we consider two adversaries, *i.e.,* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ attacking pk_1 (controlling the generation of pk_0^*) and $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ attacking pk_0 (controlling the generation of pk_1^*). In Fig. 1 below we show the security games of one-wayness and indistinguishable security corresponding to $[\mathsf{IND/OW\text{-}ATK}_1, \cdot]$ and $[\cdot, \mathsf{IND/OW\text{-}ATK}_0]$ respectively.

To be clear, the auxiliary inputs $\mathsf{aux_e}$ and $\mathsf{aux_d}$ may contain public part, called public auxiliary input, and secret part, called secret auxiliary input. In the security games, both the challenger and adversary have public auxiliary input, while only the challenger has the secret auxiliary input. For simplicity, we do not explicitly show $\mathsf{aux_e}$ and $\mathsf{aux_d}$ in the security games.

Game [IND-ATK1, ·] on pk_1	**Game [·, IND-ATK0] on pk_0**
01 $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}(pp)$;	14 $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}(pp)$
02 $L_0 = \{(-,-,-)\}$	15 $L_1 = \{(-,-,-)\}$
03 $(state, pk_0^*) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{ATK}_1}, \mathcal{O}_{\mathsf{leak}_0}}(pk_1)$	16 $(state, pk_1^*) \leftarrow \mathcal{B}_1^{\mathcal{O}_{\mathsf{ATK}_0}, \mathcal{O}_{\mathsf{leak}_1}}(pk_0)$;
04 $b \leftarrow \{0,1\}$;	17 $b \leftarrow \{0,1\}$;
05 $(c^*, k_0^*) \leftarrow \mathsf{Encaps}(pk_1, pk_0^*), k_1^* \leftarrow \mathcal{K}$;	18 $(c^*, k_0^*) \leftarrow \mathsf{Encaps}(pk_1^*, pk_0), k_1^* \leftarrow \mathcal{K}$;
06 $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{ATK}_1}, \mathcal{O}_{\mathsf{leak}_0}}(state, c^*, k_b^*)$;	19 $b' \leftarrow \mathcal{B}_2^{\mathcal{O}_{\mathsf{ATK}_0}, \mathcal{O}_{\mathsf{leak}_1}}(state, c^*, k_b^*)$;
07 return $b' \stackrel{?}{=} b$	20 return $b' \stackrel{?}{=} b$
Game [OW-ATK1, ·] on pk_1	**Game [·, OW-ATK0] on pk_0**
08 $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}(pp)$;	21 $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}(pp)$
09 $L_0 = \{(-,-,-)\}$	22 $L_1 = \{(-,-,-)\}$
10 $(state, pk_0^*) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{ATK}_1}, \mathcal{O}_{\mathsf{leak}_0}}(pk_1)$	23 $(state, pk_1^*) \leftarrow \mathcal{B}_1^{\mathcal{O}_{\mathsf{ATK}_0}, \mathcal{O}_{\mathsf{leak}_1}}(pk_0)$;
11 $(c^*, k^*) \leftarrow \mathsf{Encaps}(pk_1, pk_0^*)$;	24 $(c^*, k^*) \leftarrow \mathsf{Encaps}(pk_1^*, pk_0)$;
12 $k' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{ATK}_1}, \mathcal{O}_{\mathsf{leak}_0}}(state, c^*)$;	25 $k' \leftarrow \mathcal{B}_2^{\mathcal{O}_{\mathsf{ATK}_0}, \mathcal{O}_{\mathsf{leak}_1}}(state, c^*)$;
13 return $k' \stackrel{?}{=} k^*$	26 return $k' \stackrel{?}{=} k^*$

Fig. 1. The $[\mathsf{ATK1}, \cdot]$, and $[\cdot, \mathsf{ATK0}]$ games of 2KEM for adversaries $\mathcal{A}$ and $\mathcal{B}$. The oracles $\mathcal{O}_{\mathsf{leak}_0}$, $\mathcal{O}_{\mathsf{ATK}_1}$, $\mathcal{O}_{\mathsf{leak}_1}$, and $\mathcal{O}_{\mathsf{ATK}_0}$ are defined in the following

On the i-th query of $\mathcal{O}_{\mathsf{leak}_0}$, the challenger generates $(pk_0^i, sk_0^i) \leftarrow \mathsf{KeyGen0}(r_0^i)$, sets $L_0 = L_0 \cup \{(pk_0^i, sk_0^i, r_0^i)\}$ and returns (pk_0^i, sk_0^i, r_0^i) to adversary $\mathcal{A}$. On the i-th query of $\mathcal{O}_{\mathsf{leak}_1}$, the challenger generates $(pk_1^i, sk_1^i) \leftarrow \mathsf{KeyGen1}(r_1^i)$, sets $L_1 = L_1 \cup \{(pk_1^i, sk_1^i, r_1^i)\}$ and returns (pk_1^i, sk_1^i, r_1^i) to adversary $\mathcal{B}$.

Depending on the definition of oracle $\mathcal{O}_{\mathsf{ATK}_1}$ the adversary $\mathcal{A}$ accesses, and $\mathcal{O}_{\mathsf{ATK}_0}$ that the adversary $\mathcal{B}$ accesses, we get CPA and CCA notions respectively.

- if $\mathcal{O}_{\mathsf{ATK}_1(pk_0', c')} = -$, it implies CPA notion;
- if $\mathcal{O}_{\mathsf{ATK}_1(pk_0', c')} \neq -$, it works as following: If $pk_0' \in [L_0]_1 \wedge (c' \neq c^* \vee pk_0' \neq pk_0^*)$, compute $k' \leftarrow \mathsf{Decaps}(sk_1, sk_0', c')$, and return the corresponding k', otherwise return $\perp$. This case implies CCA notion.

– if $\mathcal{O}_{\mathsf{ATK}_0(pk'_1,c')} = -$, it implies CPA notion;
– if $\mathcal{O}_{\mathsf{ATK}_0(pk'_1,c')} \neq -$, it works as following: If $pk'_1 \in [L_1]_1 \wedge (c' \neq c^* \vee pk'_1 \neq pk^*_1)$, compute $k' \leftarrow \mathsf{Decaps}(sk'_1, sk_0, c')$, and return the corresponding k', otherwise return $\perp$. This case implies CCA notion.

Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary against pk_1 of 2KEM. We define the advantage of $\mathcal{A}$ winning in the game IND-ATK1 and OW-ATK1 respectively as: $\mathrm{Adv}^{[\mathsf{IND\text{-}ATK1},\cdot]}_{\mathsf{2KEM}}(\mathcal{A}) = \left|\Pr[\mathsf{IND\text{-}ATK1}^{\mathcal{A}} \Rightarrow 1] - \frac{1}{2}\right|$, and $\mathrm{Adv}^{[\mathsf{OW\text{-}ATK1},\cdot]}_{\mathsf{2KEM}}(\mathcal{A}) = \Pr[\mathsf{OW\text{-}ATK1}^{\mathcal{A}} \Rightarrow 1]$, where game $[\mathsf{IND\text{-}ATK1}, \cdot]$ and $[\mathsf{OW\text{-}ATK1}, \cdot]$ are described in Fig. 1.

We say that 2KEM is $[\mathsf{IND\text{-}ATK1}, \cdot]$ secure, if $\mathrm{Adv}^{[\mathsf{IND\text{-}ATK1},\cdot]}_{\mathsf{2KEM}}(\mathcal{A})$ is negligible; that 2KEM is $[\mathsf{OW\text{-}ATK1}, \cdot]$ secure, if $\mathrm{Adv}^{[\mathsf{OW\text{-}ATK1},\cdot]}_{\mathsf{2KEM}}(\mathcal{A})$ is negligible, for any PPT adversary $\mathcal{A}$. The $[\cdot, \mathsf{IND\text{-}ATK0}]$ and $[\cdot, \mathsf{OW\text{-}ATK0}]$ security can be defined in the same way. Here to avoid repetition we omit their description.

[ATK1, ATK0] Security. The scheme 2KEM is called $[\mathsf{ATK1}, \mathsf{ATK0}]$ secure if it is both $[\mathsf{ATK1}, \cdot]$ and $[\cdot, \mathsf{ATK0}]$ secure for any PPT algorithms $\mathcal{A}$ and $\mathcal{B}$. By the combination of adversaries $\mathcal{A}$ and $\mathcal{B}$ attacking different security (*i.e.,* indistinguishability and one-wayness), we could get 16 different definitions of security for 2-key KEM. What we concern in this paper is the [CCA, CPA] security in both indistinguishability and one-wayness setting. For simplicity, we abbreviate the security model as [IND/OW-CCA, IND/OW-CPA].

3.2 Differences Between [CCA, ·] Security and Previous Definitions

In order to avoid confusion, we re-clarify the definition of $[\mathsf{IND/OW\text{-}CCA}, \cdot]$ security and analyze its difference with previous similar notions, including classical CCA security, KEM combiner [17], and completely non-malleable scheme [13].

Compared with classical CCA adversary, the $[\mathrm{CCA}, \cdot]$ adversary of 2-key KEM (1) has the capability of choosing one of the challenge public key pk^*_0; (2) could query a strong decryption oracle, which decapsulates the ciphertext under several public keys (pk^*_1, pk'_0) where pk'_0 is generated by the challenger. While in the classical definition of decapsulation oracle the adversary could only query decapsulation oracle with ciphertext under the challenge public keys (pk^*_1, pk^*_0).

Very recently, Giacon *et. al* [17] study combiners for KEMs. That is, given a set of KEMs, an unknown subset of which might be arbitrarily insecure, Giacon *et. al* investigate how they can be combined to form a single KEM that is secure if at least one ingredient KEM is. The KEM combiners treated by Giacon *et al.* have a parallel structure: If the number of KEMs to be combined is n, a public key of the resulting KEM consists of a vector of n public keys; likewise for secret keys. The encapsulation procedure performs n independent encapsulations, one for each combined KEM. The ciphertext of the resulting KEM is simply the concatenation of all generated ciphertexts. The session key is obtained as a function of keys and ciphertexts. Although from the literature our 2-key KEM looks like the two KEM combiner, the security requirement and concrete constructions between them are completely different. Since the two KEM combiner considers

the problem that if one of two KEMs is insecure and the other one is CCA secure, how to combine them to obtain a CCA secure single KEM. In fact, the adversary of KEM combiner security model is the classical CCA adversary (it can only query the decryption oracle under certain public keys). Actually, in Sect. 6.1, we show there exists $[\mathrm{CCA}, \cdot]$ adversary to attack a CCA secure two KEM combiner.

Aiming to construct non-malleable commitments, Fischlin [13] considered completely non-malleable (NM) schemes. The complete NM scheme is later extended to indistinguishability setting by Barbosa and Farshim [4] with a strong decryption oracle, which allows the adversary to queries with ciphertext under *arbitrary public key of its choice*. Note that our $[\mathrm{CCA}, \cdot]$ is also modeled to allow the adversary to query a strong (but weaker than complete NM) decapsulation oracle with ciphertext under several public keys that are chosen by challenger instead of by adversary. On the other hand, the complete NM adversary is not allowed to choose part of challenge public key, while $[\mathrm{CCA}, \cdot]$ is.

Based on the above observations, we give comparison among these different definitions by considering two public keys in Table 3. For convenience, we consider classical CCA and complete NM schemes in which public keys are expressed as two public keys (pk_1, pk_0) and let KEM combiner be two combiner of KEM. The differences among security requirements are the capability of adversary, namely, whether the adversary is allowed to choose part of the challenge public keys, or under which public keys the ciphertexts that adversary is allowed to query decryption oracle with are computed.

Table 3. The differences of related definitions. "Cha." is the abbreviation of "challenge". $\mathcal{C}$ denote the challenger and $\mathcal{A}$ denote the adversary. We use $\mathcal{A}(sk_0^*)$ to denote that $\mathcal{A}$ breaks the KEM under pk_0^*. In both Classical CCA and KEM combiner the decapsulation oracle only returns when $(pk_1, pk_0) = (pk_1^*, pk_0^*)$, while in Complete NM (pk_1, pk_0) could be arbitrary public keys chosen by adversary, and in $[\mathsf{CCA}, \cdot]$, pk_0 could be arbitrary public key chosen by challenger.

Definitions	Cha. PK (pk_1^*, pk_0^*)	Cha. ciphertext c^*	$\mathcal{O}_{\mathsf{Dec}}((pk_1, pk_0), c')$
Classical CCA	$(pk_1^*, pk_0^*) \leftarrow \mathcal{C}$	c^* under (pk_1^*, pk_0^*)	$(pk_1, pk_0) = (pk_1^*, pk_0^*)$
KEM Combiner [17]	$(pk_1^*, pk_0^*) \leftarrow \mathcal{C}$, $\mathcal{A}(sk_0^*)$	$c_1^* \| c_0^*$, c_i^* under pk_i^*	$(pk_1, pk_0) = (pk_1^*, pk_0^*)$
Complete NM [13]	$(pk_1^*, pk_0^*) \leftarrow \mathcal{C}$	c^* under (pk_1^*, pk_0^*)	$(pk_1, pk_0) \leftarrow \mathcal{A}$
$[\mathsf{CCA}, \cdot]$	$pk_1^* \leftarrow \mathcal{C}, pk_0^* \leftarrow \mathcal{A}$	c^* under (pk_1^*, pk_0^*)	$pk_1 = pk_1^*$, $pk_0 \leftarrow \mathcal{C}$

3.3 Basic Definitions and Results Related to 2-Key KEM

$[\mathsf{CCA}, \cdot]$ Security with Non-adaptive Adversary. We can define a weak $[\mathsf{CCA}, \cdot]$ adversary, who is not able to adaptively choose the challenge public key. In this case, taking the adversary $\mathcal{A}$ attacking pk_1 as an example, the challenge public key pk_0^* is generated by challenger instead of $\mathcal{A}$, which means $pk_0^* \in [L_0]_1$.

Public Key Independent Ciphertext. The concept of public-key-independent-ciphertext (PKIC) was first proposed in [33]. We extend it to 2-key

KEM setting. The PKIC 2-key KEM allows a ciphertext to be generated independently from one of two public keys, while the encapsulated key underlay in such ciphertext to be generated with the randomness and both two public keys. More precisely, algorithm $(c, k) \leftarrow \mathsf{Encaps}(pk_1, pk_0, r)$ can be realized in two steps: in step 1, ciphertex c is generated from pk_1 and randomness r. We precisely denote it as $c \leftarrow \mathsf{Encaps0}(pk_1, \text{-}, r)$; in step 2, the encapsulated key k in c is generated from r, pk_1, and pk_0. We precisely denote it as $k \leftarrow \mathsf{Encaps1}(pk_1, pk_0, r)$.

Classical One-Key KEM and 2-Key KEM. Note that given a concrete 2-key KEM, usually it is not obvious and natural to regress to one-key KEM by setting $pk_0 = \text{-}$. However given any classical one-key KEM, it can be seen as a 2-key KEM with $\mathsf{KeyGen0}$ not in use and $pk_0 = \text{-}$. At that time, the $[\mathsf{OW/IND\text{-}CCA}, \cdot]$ security of this 2-key KEM return to the classical $\mathsf{OW/IND\text{-}CCA}$ security of the underlying KEM.

Min-Entropy. In case of 2-key KEM with PPT adversary $\mathcal{A}$, for $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}$ and $pk_0 \leftarrow \mathcal{A}$ or $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}$ and $pk_1 \leftarrow \mathcal{A}$, we define the *min-entropy* of $\mathsf{Encaps}(pk_1, pk_0)$ by $\gamma(pk_1, pk_0, \mathcal{A}) = -\log \max_{c \in \mathcal{C}} \Pr[c = \mathsf{Encaps}(pk_1, pk_0)]$. We say that KEM is γ-*spread* if for every $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}$ and $pk_0 \leftarrow \mathcal{A}$ or $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}$ and $pk_1 \leftarrow \mathcal{A}$, $\gamma(pk_1, pk_0, \mathcal{A}) \geq \gamma$, which means for every ciphertext $c \in \mathcal{C}$, it has $\Pr[c = \mathsf{Encaps}(pk_1, pk_0)] \leq 2^{-\gamma}$.

4 Authenticated Key Exchange from 2-Key KEM

In this section, we propose CK^+ secure AKEs from $[\mathsf{CCA}, \mathsf{CPA}]$ secure 2-key KEM in both random oracle and standard models. Before showing our AKEs, we need a primitive of random function with half of leakage, that is used by several existing AKEs.

Definition 2 (Random Function with half of leakage (hl-RF)). *Let $f : D_{sk} \times D_b \rightarrow R$ be a function from domain $D_{sk} \times D_b$ to R. Denote $\mathsf{KeyGen} \rightarrow D_{sk} \times D_{pk}$ as key generation algorithm for some KEM. Let $\mathcal{D}_b, \mathcal{R}$ be the uniformly distributions over D_b, R. It is called $(\varepsilon_1, \varepsilon_2)$ hl-RF with respect to KeyGen, if for $(pk, sk) \leftarrow \mathsf{KeyGen}$, the following distributions are computational indistinguishable with advantage $\varepsilon_1, \varepsilon_2$.*

$$\{(pk, sk, f(sk, b)) | b \leftarrow \mathcal{D}_b\} =_{\varepsilon_1} \{(pk, sk, U) | U \leftarrow \mathcal{R}\};$$
$$\{(pk, b, f(sk, b)) | b \leftarrow \mathcal{D}_b\} =_{\varepsilon_2} \{(pk, b, U) | b \leftarrow \mathcal{D}_b, U \leftarrow \mathcal{R}\}.$$

The hk-RF can be achieved in both random oracle model and standard model.

- In the random oracle model, if f is a hash function, without the knowledge of b, the output of f is totally random; if KEM with respect to KeyGen is secure, without the knowledge of sk the output of f is computational indistinguishable with a random string (otherwise the adversary must query random oracle with sk which against the security of KEM) given pk. Then Eq. 2 holds. This structure is known as NAXOS trick [24].

– Let $F' : D_b \times \{0,1\}^\lambda \to R$ and $F'' : D_{sk} \times D_b \to R$ be two pseudo random functions (PRFs). If assume KeyGen outputs an additional string $s \leftarrow \{0,1\}^\lambda$, after obtaining (pk, sk), set $sk = (sk||s)$. If $f(sk, b) = F'_b(1^\lambda) \oplus F''_s(b)$, then even given pk, without the knowledge of s or b, $f(sk, b)$ is computational indistinguishable with random distribution over R. This is known as twisted PRF trick [15,27].

4.1 AKE from 2-Key KEM in Random Oracle Model

Roadmap: We first give a basic AKE from two [OW-CCA, OW-CPA] secure 2-key KEMs. Utilizing extra properties of 2-key KEM, like PKIC or resistance of leakage of partial randomness, we then present two elegant AKEs based on 2-key KEM with different property.

Let $\mathsf{2KEM} = (\mathsf{KeyGen1}, \mathsf{KeyGen0}, \mathsf{Encaps}, \mathsf{Decaps})$ be a [OW-CCA, OW-CPA] secure 2-key KEM with secret key space $D_{sk_1} \times D_{sk_0}$, random space R. Let $H : \{0,1\}^* \to \{0,1\}^\lambda$ be hash function, $f_A : D_{sk_1} \times \{0,1\}^* \to R$ and $f_B : D_{sk_1} \times \{0,1\}^* \to R$ be hl-RFs. The CK^+ secure AKE is presented in Fig. 2.

Stage 0: static secret-public key pair and public parameters. Each user's static secret-public key pair is generated using $\mathsf{KeyGen1}$. Sample one pair of key $(cpk_0, csk_0) \leftarrow \mathsf{KeyGen0}$ (which need not to be randomly generated). Set cpk_0 as the predetermined ephemeral public key which will be used by initiator afterwards and csk_0 as the predetermined ephemeral secret key that will be used by responder. Let (cpk_0, csk_0) be parts of public parameters.

Stage 1: Initiator U_A generates two randomness r_A, r_{A0}; it computes (C_B, K_B) under public key pk_B and predetermined cpk_0 with randomness $f_A(sk_A, r_A)$, and generates ephemeral secret-public key $(pk_{A0}, sk_{A0}) \leftarrow \mathsf{KeyGen0}(r_{A0})$. Then it sends C_B, pk_{A0} to U_B.

Stage 2: Responder U_B generates randomness r_B; it computes (C_A, K_A) under public keys pk_A and pk_{A0} with randomness $f_B(sk_B, r_B)$; U_B sends C_A to U_A and de-encapsulates C_B using sk_B and predetermined csk_0 to obtain K'_B; it then computes $SK = H(U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A, K_A, K'_B)$, and erases K'_B.

Stage 3: U_A de-encapsulates C_A using sk_A and sk_{A0} to obtain K'_A and computes $SK = H(U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A, K'_A, K_B)$.

The session state of sid owned by U_A consists of ephemeral secret key r_{A0}, r_A, decapsulated key K'_A and encapsulated key K_B; The session state of sid owned by U_B consists of ephemeral secrete key r_B and encapsulated key K_A.

Theorem 1. *If the underlying 2KEM is* [OW-CCA, OW-CPA] *secure and* γ*-spread,* f_A, f_B *are* $(\varepsilon_1, \varepsilon_2)$ *hl-RFs, and there are* N *users in the AKE protocol and the upbound of sessions between two users is* l*, for any PPT adversary* $\mathcal{A}$ *against AKE with totally* q *times of* CK^+ *queries, there exists* $\mathcal{S}$ *s.t.,*

$$Adv^{ck+}_{\mathsf{AKE}}(\mathcal{A}) \leq \frac{1}{2} + \min \left\{ \begin{array}{l} N^2 l \cdot Adv^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}(\mathcal{S}) + N^2 lq \cdot (\varepsilon_1 + \varepsilon_2 + 2^{-\gamma}), \\ N^2 l \cdot Adv^{[\cdot,\mathsf{OW\text{-}CPA}]}_{\mathsf{2KEM}}(\mathcal{S}) + N^2 lq \cdot \varepsilon_2 \end{array} \right\}.$$

U_A		U_B
$(pk_A, sk_A) \leftarrow \mathsf{KeyGen1}$		$(pk_B, sk_B) \leftarrow \mathsf{KeyGen1}$
$r_A \leftarrow \{0,1\}^*, r_{A0} \leftarrow \{0,1\}^*$		$r_B \leftarrow \{0,1\}^*$
$R_A := f_A(sk_A, r_A)$		
$(C_B, K_B) \leftarrow \mathsf{Encaps}(pk_B, cpk_0, R_A)$		$R_B := f_B(sk_B, r_B)$
$(pk_{A0}, sk_{A0}) \leftarrow \mathsf{KeyGen0}(r_{A0})$	$\xrightarrow{C_B, pk_{A0}}$	$(C_A, K_A) \leftarrow \mathsf{Encaps}(pk_A, pk_{A0}, R_B)$
$K'_A \leftarrow \mathsf{Decaps}(sk_A, sk_{A0}, C_A)$	$\xleftarrow{C_A}$	$K'_B \leftarrow \mathsf{Decaps}(sk_B, csk_0, C_B)$
$SK = H(si, K'_A, K_B)$		$SK = H(si, K_A, K'_B)$

Fig. 2. AKE from [OW-CCA, OW-CPA] secure 2KEM in random oracle model. cpk_0, csk_0 are predetermined and default ephemeral keys and they are part of the public parameters. si here is $(U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A)$

Proof of Theorem 1. Let Succ be the event that the guess of $\mathcal{A}$ against freshed test session is correct. Let AskH be the event that $\mathcal{A}$ poses $(U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A, K_A, K_B)$ to H, where C_B, pk_{A0}, C_A are the views of the test session and K_A, K_B are the keys encapsulated in the test session. Let $\overline{\mathsf{AskH}}$ be the complement of AskH. Then,

$$\Pr[\mathsf{Succ}] = \Pr[\mathsf{Succ} \wedge \overline{\mathsf{AskH}}] + \Pr[\mathsf{Succ} \wedge \mathsf{AskH}] \leq \Pr[\mathsf{Succ} \wedge \overline{\mathsf{AskH}}] + \Pr[\mathsf{AskH}],$$

where the probability is taken over the randomness used in CK^+ experiment.

We then show that $\Pr[\mathsf{Succ} \wedge \overline{\mathsf{AskH}}] \leq 1/2$ (as in Lemma 1) and $\Pr[\mathsf{AskH}]$ is negligible (as in Lemma 2) in all the events (listed in Table 2) of CK^+ model. Followed by Lemmas 1 and 2, we acheive the security of AKE in CK^+ model. Thus, we only need to prove Lemmas 1 and 2.

Lemma 1. *If H is modeled as a random oracle, we have $Pr[\mathsf{Succ} \wedge \overline{\mathsf{AskH}}] \leq 1/2$.*

Proof of Lemma 1. If $\Pr[\overline{\mathsf{AskH}}] = 0$ then the claim is straightforward, otherwise we have $\Pr[\mathsf{Succ} \wedge \overline{\mathsf{AskH}}] = \Pr[\mathsf{Succ}|\overline{\mathsf{AskH}}]\Pr[\overline{\mathsf{AskH}}] \leq \Pr[\mathsf{Succ}|\overline{\mathsf{AskH}}]$. Let sid be any completed session owned by an honest party such that $\mathsf{sid} \neq \mathsf{sid}^*$ and $\overline{\mathsf{sid}}$ is not matching sid^*. The inputs to sid are different from those to sid^* and $\overline{\mathsf{sid}^*}$ (if there exists the matching session of sid^*). If $\mathcal{A}$ does not explicitly query the view and keys to oracle, then $H(U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A, K_A, K_B)$ is completely random from $\mathcal{A}$'s point of view. Therefore, the probability that $\mathcal{A}$ wins when AskH does not occur is exactly $1/2$.

Lemma 2. *If the underlying 2KEM is [OW-CCA, OW-CPA] secure, the probability of event AskH defined above is negligible. Precisely,*

$$\Pr[\mathsf{AskH}] \leq \min \left\{ \begin{array}{l} N^2 l \cdot Adv_{\mathsf{2KEM}}^{[\mathsf{OW\text{-}CCA},\cdot]}(\mathcal{S}) + N^2 lq \cdot (\varepsilon_1 + \varepsilon_2 + 2^{-\gamma}), \\ N^2 l \cdot Adv_{\mathsf{2KEM}}^{[\cdot,\mathsf{OW\text{-}CPA}]}(\mathcal{S}) + N^2 lq \cdot \varepsilon_2 \end{array} \right\}.$$

Please refer the full version [32] for the formal proof. we give a sketch of proof here. In the following, to bound $\Pr[\mathsf{AskH}]$, we work with the events in Table 4.

Table 4. The bounds of $\mathsf{AskH} \wedge \overline{\mathsf{Askh}}$ in the proof of Lemma 2. Refer Table 2 for the meanings of notions.

Events	sid^*	$\overline{\mathsf{sid}^*}$	ssk_A	esk_A	esk_B	ssk_B	Bounds
$\mathsf{AskH} \wedge E_1$	A	No	$\surd$	$\times$	-	$\times$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_B$, $pk_0^* = cpk_0$
$\mathsf{AskH} \wedge E_2$	A	No	$\times$	$\surd$	-	$\times$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_B$, $pk_0^* = cpk_0$
$\mathsf{AskH} \wedge E_3$	B	No	$\times$	-	$\surd$	$\times$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_A$, $pk_0^* \leftarrow \mathcal{A}$
$\mathsf{AskH} \wedge E_4$	B	No	$\times$	-	$\times$	$\surd$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_A$, $pk_0^* \leftarrow \mathcal{A}$
$\mathsf{AskH} \wedge E_5$	A/B	Yes	$\surd$	$\times$	$\times$	$\surd$	$\mathrm{Adv}^{[\cdot,\mathsf{OW\text{-}CPA}]}_{\mathsf{2KEM}}$, $pk_0 = pk_0(\mathsf{sid}^*)$ $pk_1^* \in [L_1]_1$
$\mathsf{AskH} \wedge E_6$	A/B	Yes	$\times$	$\surd$	$\surd$	$\times$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_A, pk_0^* \in [L_0]_1$
$\mathsf{AskH} \wedge E_{7\text{-}1}$	A	Yes	$\surd$	$\times$	$\surd$	$\times$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_B$ $pk_0^* = cpk_0$
$\mathsf{AskH} \wedge E_{7\text{-}2}$	B	Yes	$\times$	$\surd$	$\times$	$\surd$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_A$, $pk_0^* \in [L_0]_1$
$\mathsf{AskH} \wedge E_{8\text{-}1}$	A	Yes	$\times$	$\surd$	$\times$	$\surd$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_A$, $pk_0^* \in [L_0]_1$
$\mathsf{AskH} \wedge E_{8\text{-}2}$	B	Yes	$\surd$	$\times$	$\surd$	$\times$	$\mathrm{Adv}^{[\mathsf{OW\text{-}CCA},\cdot]}_{\mathsf{2KEM}}$, $pk_1 = pk_B$, $pk_0^* = cpk_0$

Due to the $[\mathsf{OW\text{-}CCA}, \cdot]$ security of 2KEM with $pk_1 = pk_A$ and pk_0^* generated by $\mathcal{A}$, the probability of events $\mathsf{AskH} \wedge E_3$ and $\mathsf{AskH} \wedge E_4$ is negligible; Due to the $[\mathsf{OW\text{-}CCA}, \cdot]$ security of KEM with $pk_1 = pk_B$ and $pk_0^* = cpk_0$, the probability of events $\mathsf{AskH} \wedge E_1$, $\mathsf{AskH} \wedge E_2$, $\mathsf{AskH} \wedge E_{7\text{-}1}$ and $\mathsf{AskH} \wedge E_{8\text{-}2}$ is negligible; Due to the $[\mathsf{OW\text{-}CCA}, \cdot]$ security of 2KEM with $pk_1 = pk_A$ and $pk_0^* \in [L_0]_1$, the probability of events $\mathsf{AskH} \wedge E_6$, $\mathsf{AskH} \wedge E_{7\text{-}2}$ and $\mathsf{AskH} \wedge E_{8\text{-}1}$ is negligible. Due to the $[\cdot, \mathsf{OW\text{-}CPA}]$ security with $pk_1^* \in [L_1]_1$, the probability of event $\mathsf{AskH} \wedge E_5$ is negligible.

Here, we only take $\mathsf{AskH} \wedge E_3$ as an example to explain in detail. For the other cases we can deal with them in a similar way. In the event E_3, the test session sid^* has no matching session, and the ephemeral secret keys r_B of U_B is given to $\mathcal{A}$. In case of $\mathsf{AskH} \wedge E_3$, the $[\mathsf{OW\text{-}CCA}, \cdot]$ adversary $\mathcal{S}$ performs as follows. It simulates the CK^+ games, and transfers the probability that the event AskH performed by $\mathcal{A}$ occurs to the advantage of attacking $[\mathsf{OW\text{-}CCA}, \cdot]$ security.

In order to simulate the random oracles, $\mathcal{S}$ maintains two lists for H oracle and SessionKeyReveal respectively. H-oracle and SessionKeyReveal are related, which means the adversary may ask SessionKeyReveal without the encapsulated keys at first, and then may ask H-oracle with the encapsulated keys. Thus, the reduction must ensure consistency with the random oracle queries to H and SessionKeyReveal. The decryption oracle for $[\mathsf{OW\text{-}CCA}, \cdot]$ game would help to maintain the consistency of H-oracle and SessionKeyReveal.

On receiving the public key pk_1 from the $[\mathsf{OW\text{-}CCA}, \cdot]$ challenger, to simulate the CK^+ game, $\mathcal{S}$ randomly chooses two parties U_A, U_B and the i-th session as a guess of the test session with success probability $1/N^2l$. $\mathcal{S}$, picks one preset $(cpk_0, csk_0) \leftarrow \mathsf{KeyGen0}$ as public parameters, runs KeyGen1 to set all the static secret and public key pairs (pk_P, sk_P) for all N users U_P except for U_A. Specially, $\mathcal{S}$ sets the static secret and public key pairs (pk_B, sk_B) for U_B, and sets $pk_A = pk_1$.

Without knowing the secret key of U_A, $\mathcal{S}$ chooses totally random r_A as part of ephemeral secret key and totally random R_A for Encaps. Since f_A is $(\varepsilon_1, \varepsilon_2)$ hl-RF, the difference between simulation with modification of r_A and real game

is bounded by ε_1. When a ephemeral public key pk_{P0} is needed, $\mathcal{S}$ queries $(pk_0^i, sk_0^i, r_0^i) \leftarrow \mathcal{O}_{\mathsf{leak}_0}$ and sets $pk_{P0} = pk_0^i$. When a session state revealed to a session owned by U_A, is queried, $\mathcal{S}$ returns r_A and r_0^i of this session as part of ephemeral secret key.

On receiving the i-th session (C'_B, pk_0^*) from U_A (that is sent by $\mathcal{A}$ in the CK^+ games), $\mathcal{S}$ returns pk_0^* to the $[\mathsf{OW\text{-}CCA}, \cdot]$ challenger and receives the challenge ciphertext C^* under public key pk_1 and pk_0^*. Then $\mathcal{S}$ returns C^* to U_A as the response of i-th session from U_B. $\mathcal{S}$ chooses a totally independent randomness r_B as the ephemeral secret key of U_B for C^* and leaks it to adversary $\mathcal{A}$. Since f_B is $(\varepsilon_1, \varepsilon_2)$ hl-RF, the difference between simulation with modification of r_B and real game is bounded by ε_2.

$\mathcal{S}$ simulates the oracle queries of $\mathcal{A}$ and maintains the hash lists. Specially, when AskH happens, which means $\mathcal{A}$ poses $(U_A, U_B, pk_A, pk_B, C'_B, pk_0^*, C^*, K_A, K_B)$ to H, where C'_B, pk_0^*, C^* are the views of the test session and K_B is the key encapsulated in C'_B, $\mathcal{S}$ returns K_A as the guess of K^* encapsulated in C^*, which contradicts with the $[\mathsf{OW\text{-}CCA}, \cdot]$ security for $pk_1 = pk_A$, $pk_0^* \leftarrow \mathcal{A}$. □

4.1.1 If 2-Key KEM Is PKIC

As we notice in AKE, the session state of sid owned by U_B does not contain decapsulated key K'_B. If the underlying 2-key KEM is PKIC (which is defined in Sect. 3.3), and U_B also sends ephemeral public key pk_{B0} out in every session, K'_B is encapsulated under two public keys pk_B and pk_{B0}, then K'_B could be included in session state, and the predetermined ephemeral public key cpk_0 can be omitted. Let $\mathsf{2KEM_{pkic}} = (\mathsf{KeyGen1}, \mathsf{KeyGen0}, \mathsf{Encaps0}, \mathsf{Encaps1}, \mathsf{Decaps})$ be PKIC and $[\mathsf{OW\text{-}CCA}, \mathsf{OW\text{-}CPA}]$ secure 2-key KEM. The AKE can be modified to include K'_B as session state by (1) replacing $\mathsf{2KEM}$ with $\mathsf{2KEM_{pkic}}$; (2) requiring U_B to generate a fresh $(pk_{B0}, sk_{B0}) \leftarrow \mathsf{KeyGen0}$ and send out ephemeral public key pk_{B0}; (3) encapsulating and separating $(C_B, K_B) \leftarrow \mathsf{Encaps}(pk_B, pk_{B0}, R_A)$ in two steps and computing $C_B \leftarrow \mathsf{Encaps0}(pk_B, \text{-}, R_A)$ and $K_B \leftarrow \mathsf{Encaps1}(pk_B, pk_{B0}, R_A)$. The modified protocol $\mathsf{AKE_{ro\text{-}pkic}}$ is shown in Fig. 3.

Note that the encapsulation algorithm of PKIC 2-key KEM can be split into two steps. Since the generation of ciphertext C_B does not require pk_{B0}, we denote it as $C_B \leftarrow \mathsf{Encaps0}(pk_B, \text{-}, R_A)$. The computation of encapsulated key K_B requires pk_{B0}, and we denote it as $K_B \leftarrow \mathsf{Encaps1}(pk_B, pk_{B0}, R_A)$.

Since the proof mainly follows that of Theorem 1, we only show the difference here. The main difference is the analysis of $\Pr[\mathsf{AskH}]$ in Lemma 2. Now, the probability of events $\mathsf{AskH} \wedge E_1$, $\mathsf{AskH} \wedge E_2$, $\mathsf{AskH} \wedge E_{7\text{-}1}$, $\mathsf{AskH} \wedge E_{8\text{-}2}$ is bounded by the $[\mathsf{OW\text{-}CCA}, \cdot]$ security of $\mathsf{2KEM_{pkic}}$ with pk_0^* chosen by $\mathcal{A}$ rather than the predetermined cpk_0. Precisely, in those events, when the adversary queries the session state of U_B whose secret key is unknown to simulator $\mathcal{S}$, in AKE, $\mathcal{S}$ queries the decryption oracle of $\mathsf{2KEM}$ with cpk_0 and C_B (when adversary queries $\mathsf{Send}(\Pi, R, U_B, U_P, C_B, pk_{A0})$), while in $\mathsf{AKE_{pkic}}$, $\mathcal{S}$ queries the decryption oracle of $\mathsf{2KEM_{pkic}}$ with (pk_{B0}, C_B) chosen by $\mathcal{A}$. This modification does not affect the proof of security.

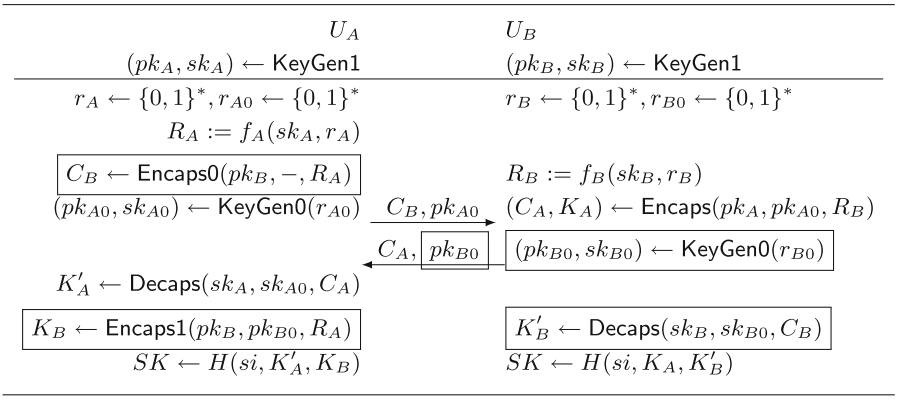

Fig. 3. $\mathsf{AKE}_{\mathsf{ro\text{-}pkic}}$ from PKIC [OW-CCA, OW-CPA] secure 2KEM. Here $si = (U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A, pk_{B0})$. The boxed argument is the difference with AKE

4.1.2 If PKIC 2-Key KEM Is Even Secure with Leakage of Partial Randomness

We can further refine the framework $\mathsf{AKE}_{\mathsf{ro\text{-}pkic}}$ based on two observations: (1) From the proof of Theorem 1 (especially Lemma 2), we can see that the only purpose of f_A and f_B is to preserve the $[\mathsf{OW\text{-}CCA}, \cdot]$ security with $pk_1 = pk_A$ and the $[\cdot, \mathsf{OW\text{-}CPA}]$ security with $pk_0 = pk_{A0}$ even if part of randomness, r_B or sk_B is leaked to the adversary. If the underlying 2-key KEM itself is strong enough to preserve the [OW-CCA, OW-CPA] security with respect to *some* function $f_A(sk_A, r_A)$ (resp. $f_B(sk_B, r_A)$), and leakage of sk_A or r_A for fixed pk_A (resp. sk_B or r_B for fixed pk_B), the functions f_A and f_B don't have to be hl-RFs. (2) if the 2-key KEM is strong enough to preserve security even when the randomness r_{B0} used to generate pk_{B0} is generated from $f_{B0}(sk_B, r_B)$ for some function f_{B0}, then we could regard $f_{B0}(sk_B, r_B)$ as a random string using to compute pk_{B0}. The same holds when $(pk_{A0}, sk_{A0}) \leftarrow \mathsf{KeyGen0}(r_{A0})$ where $r_{A0} = f_{A0}(sk_A, r_A)$ for some function f_{A0}.

Therefore, the problem comes down to study the security of 2-key KEM when C_A (under public keys pk_A and pk_{A0}) shares the randomness of pk_B and pk_{B0}.

Definition 3. *We say 2-key KEM is leakage resistant of partial randomness with respect to f_B and f_{B0} (they need not to be hl-RFs), if the following property holds. Under public key pk_A and pk_{A0}, the $[\mathsf{OW\text{-}CCA}, \mathsf{OW\text{-}CPA}]$ security still holds where the ciphertext is computed as $\mathsf{Encaps}(pk_A, pk_{A0}, f_B(sk_B, r_B))$ for some fixed pk_B (where $(pk_B, sk_B) \leftarrow \mathsf{KeyGen1}$), when either r_B and pk_{B0} or sk_B and pk_{B0} are given to adversary, where $(pk_{B0}, sk_{B0}) \leftarrow \mathsf{KeyGen0}(f_{B0}(sk_B, r_B))$.*

Equipped with PKIC 2-key KEM that resists to the leakage of partial randomness with respect to f_B and f_{B0}, we set $f_{A0}(sk_A, r_A)$ and $f_{B0}(sk_B, r_B)$ as the randomness for KeyGen0, and denote the result AKE as $\mathsf{AKE}_{\mathsf{ro\text{-}pkic\text{-}lr}}$ in Fig. 4. The session state of sid owned by U_A consists of r_A, K'_A and K_B, the session state of sid owned by U_B consists of r_B, K_A and K'_B.

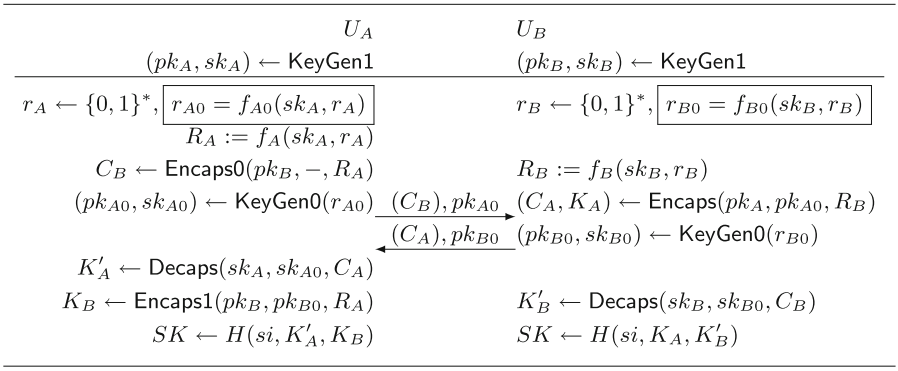

Fig. 4. $\mathsf{AKE}_{\mathsf{ro\text{-}pkic\text{-}lr}}$. Here $si = (U_A, U_B, pk_A, pk_B, C_B, pk_{A0}, C_A, pk_{B0})$. The boxed argument is the main difference with $\mathsf{AKE}_{\mathsf{ro\text{-}pkic}}$

Remark 1. As in the definition of 2-key KEM, both Encaps and Decaps allow to have auxiliary input $\mathsf{aux_e}$ or $\mathsf{aux_d}$. In $\mathsf{AKE}_{\mathsf{ro\text{-}pkic\text{-}lr}}$ (AKE and $\mathsf{AKE}_{\mathsf{ro\text{-}pkic}}$), the static public keys are generated by KeyGen1 during the registration phase (i.e., Stage 0) and publicly available. Thus, in the protocol, it makes sense that Encaps and Decaps algorithms take the static public keys as public auxiliary input. And for user U_A (resp. U_B), it is also reasonable that Encaps executed by U_A (resp. U_B) takes his static secret key sk_A (resp. sk_B) as auxiliary input. In this sense, one couple of 2KEM is really "coupled" with each other.

Remark 2. Since C_A share the randomness of pk_{B0} and secret key of pk_B, if the 2-key KEM and function f_B/f_{B0} further satisfy that C_A is publicly computable from pk_B and pk_{B0}, we can omit C_A in the communications. The same holds for C_B, if it is publicly computable from pk_A and pk_{A0}, we can omit C_B.

Remark 3. Note that the computation of f_B is part of $\mathsf{Encaps}(pk_A, pk_{A0}, R_B)$ algorithm. f_B may take pk_A as input. At that time, to be clear, we denote $f_B(sk_B, r_B)$ as $f_B(sk_B, r_B, pk_A)$. It is similar in the case of f_A.

With these modifications, we should handle the proof more carefully. The main challenge is that the ciphertext C_A, static public key pk_B, ephemeral public key pk_{B0} are correlated (the same holds for C_B, pk_A, and pk_{A0}). We should handle the problem that, since C_A shares the randomness with pk_{B0} and secret key of pk_B, when applying the $[\mathsf{OW\text{-}CCA}, \cdot]$ security of 2-key KEM with $pk_1 = pk_A$ in event $\mathsf{AskH} \wedge E_3$, $\mathsf{AskH} \wedge E_6$, not only sk_A but also sk_B is unknown to simulator $\mathcal{S}$. (The same situation occurs when applying $[\mathsf{OW\text{-}CCA}, \cdot]$ security of 2-key KEM with $pk_1 = pk_B$ in event $\mathsf{AskH} \wedge E_2$).

The way to solving this problem is to bring in another $[\mathsf{OW\text{-}CCA}, \cdot]$ challenge. As an example, we sketch the proof of event $\mathsf{AskH} \wedge E_3$ to show how this resolves the above problem. The main modification is for the proof of Lemma 2. In case of $\mathsf{AskH} \wedge E_3$, the $[\mathsf{OW\text{-}CCA}, \cdot]$ adversary $\mathcal{S}$ performs as follows. On receiving the public key pk_1 from the $[\mathsf{OW\text{-}CCA}, \cdot]$ challenger, to simulate the CK^+ game, $\mathcal{S}$

randomly chooses two parties U_A, U_B and the i-th session as a guess of the test session. $\mathcal{S}$ runs $\mathsf{KeyGen1}$ to generate all static public keys except U_A and U_B. $\mathcal{S}$ queries the first $[\mathsf{OW\text{-}CCA}, \cdot]$ challenger to get pk_1, and sets $pk_A = pk_1$. $\mathcal{S}$ queries the second $[\mathsf{OW\text{-}CCA}, \cdot]$ challenger again to get another pk_1' and sets $pk_B = pk_1'$.

Note that now $\mathcal{S}$ does not know the secret key of both pk_A and pk_B. Here $\mathcal{S}$ generates (pk^*_{B0}, sk^*_{B0}) by itself. $\mathcal{S}$ sends pk^*_{B0} to the second challenge to get challenge ciphertext C^*_B and keeps both pk^*_{B0} and C^*_B secret to CK^+ adversary $\mathcal{A}$. On receiving the i-th session (C'_B, pk^*_{A0}) from U_A (that is sent by $\mathcal{A}$ in the CK^+ games), $\mathcal{S}$ queries the first $[\mathsf{OW\text{-}CCA}, \cdot]$ challenger with pk^*_{A0} and obtains C^*_A, pk_{B0} and its randomness r_{B0}. $\mathcal{S}$ returns C^*_A and pk_{B0} to U_A as the response of i-th session from U_B, and sets pk^*_{A0} as the public key under which C^*_A is encrypted. $\mathcal{S}$ also leaks r_{B0} to adversary as the ephemeral secret key.

With the first $[\mathsf{OW\text{-}CCA}, \cdot]$ challenge, $\mathcal{S}$ could partially maintain the hash list and $\mathsf{SessionStateReveal}$ and $\mathsf{SessionKeyReveal}$ with strong decapsulation oracle when U_B is not involved. When U_B is involved, the second $[\mathsf{OW\text{-}CCA}, \cdot]$ challenge is needed. Note that since 2-key KEM is γ-spread, the probability that $\mathcal{A}$ queries a message with $C_B = C^*_B$ is bounded by $q \times 2^{-\gamma}$. The simulation is perfect and the other part of proof proceeds the same with Lemma 2.

4.2 AKE from 2-Key KEM in Standard Model

The protocol $\mathsf{AKE}/\mathsf{AKE}_{\mathsf{ro\text{-}pkic}}$ in random oracle model can be easily extended to one that is secure in the standard model, denoted by $\mathsf{AKE}_{\mathsf{std}}/\mathsf{AKE}_{\mathsf{std\text{-}pkic}}$, via the following steps:

1. replacing the $[\mathsf{OW\text{-}CCA}, \mathsf{OW\text{-}CPA}]$ secure 2-key KEM in random oracle model with the $[\mathsf{IND\text{-}CCA}, \mathsf{IND\text{-}CPA}]$ secure 2-key KEM in standard model;
2. instantiating the hl-RF functions f_A, f_B in standard model instead of the random oracle model. As noted after the definition, the instantiation of hl-RF in standard model require PRF and extra randomness. Thus every user holds extra random secret $s_P \leftarrow \{0,1\}^\lambda$ as part of the static secret key and $R_A = f_A(sk_A||s_A, r_A)$, $R_B = f_B(sk_B||s_B, r_B)$.
3. replacing the random oracle $H(si, K_A, K_B)$ with $F_{K_A}(si) \oplus \hat{F}_{K_B}(si)$, to extract session key, where F and $\hat{F}$ are PRFs.

Actually, converting a scheme in the random oracle model into that in the standard model is generally not trivial, and there are many negative results. However, without taking advantage of strong property of random oracle, our step 2 and 3 just use the property that if the input is unknown then the output is totally random. The difficult part is step 1. Once the 2-key KEM in random oracle model is replaced by $[\mathsf{IND\text{-}CCA}, \mathsf{IND\text{-}CPA}]$ secure 2-key KEM in standard model, the proof of security for AKE in standard model is straightforward.

5 Unification of Prior Works

In this section, we show that existing AKEs, HMQV [23], NAXOS [24], Okamoto [27], and FSXY framework [15,16], can be explained in our unified frameworks.

5.1 HMQV-AKE

In HMQV [23], the 2-key KEM is initiated by $\mathsf{2KEM_{HMQV}}$ in Fig. 5. Let h and $\hat{H}$ be hash functions. Let G be a group of prime order p with g as a generator. Both Encaps and Decaps algorithms have auxiliary input $\mathsf{aux_e} = (B, b)$ where $B = g^b$ and $\mathsf{aux_d} = B$. Note that, here, B is the public auxiliary input and b is the secret auxiliary input. By applying $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$, Remarks 1–3, we present how the HMQV scheme is integrated in our unified framework of AKE and how it is built from the view of 2-key KEM in Fig. 6.

$\mathsf{KeyGen1}(\lambda)$	$\mathsf{KeyGen0}(\lambda)$	$\mathsf{Encaps}(pk_1, pk_0; \mathsf{aux}_e(B, b))$	$\mathsf{Decaps}(sk_1, sk_0, c; \mathsf{aux}_d(B))$
$a \leftarrow \mathbb{Z}_p$;	$x \leftarrow \mathbb{Z}_p$	$y \leftarrow \mathbb{Z}_p$, $Y = g^y$,	$YB^e \leftarrow c$;
$A = g^a$	$X = g^x$	$e = h(Y, A), d = h(X, B)$	$e = h(Y, A), d = h(X, B)$
$pk_1 = A$	$pk_0 = X$;	$k = \hat{H}((XA^d)^{y+eb})$	$k' = \hat{H}((YB^e)^{x+da})$
$sk_1 = a$	$sk_0 = x$.	Return $k, c = YB^e$.	Return k'

Fig. 5. The [OW-CCA, OW-CCA] secure $\mathsf{2KEM_{HMQV}}$ implied by HMQV.

Theorem 2. *Under the Gap-DH and KEA1 assumptions*[1]*,* $\mathsf{2KEM_{HMQV}}$ *in Fig. 5 is [OW-CCA, OW-CCA] secure with the resistance to the leakage of partial randomness with respect to* $f_B(b, y, A) = y + b \cdot h(g^y, A)$ *and* $f_{B0}(b, y) = y$ *in the random oracle model.*

Please refer the full version [32] for the proof of Theorem 2.

As said in Remark 3, f_B takes A as input and $f_B(b, y, A) = y + b \cdot h(g^y, A)$. By Theorem 2, $\mathsf{2KEM_{HMQV}}$ is [OW-CCA, OW-CCA] secure even if partial randomness (b or y) is leaked with respect to $f_B(b, y, A) = y + b \cdot h(g^y, A)$ and $f_{B0}(b, y) = y$. By changing the role of A and B, X and Y, we also get a dual scheme of $\mathsf{2KEM_{HMQV}}$, with respect to $f_A(a, x, B) = x + a \cdot h(g^x, B)$ and $f_{A0}(a, x) = x$. Obviously, $\mathsf{2KEM_{HMQV}}$ is PKIC, which means that the ciphertext is independent of the public key pk_0. Thus the Encaps algorithm can be split into two steps Encaps0 and Encaps1. However, when integrating $\mathsf{2KEM_{HMQV}}$ into $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$ to reproduce HMQV, one may doubt that whether $\mathsf{aux}_e = (B, b)$ or (A, a) required by Encaps and $\mathrm{aux}_d = B$ or A required by Decaps influence the reconstruction. As explained in Remark 2, since B and A are the static public keys and generated during the registration phase, they can be used as the public auxiliary input by any user during the execution phase. As a static secret key, b can be used by U_B as secret auxiliary input during the execution phase. Based on the above analysis, applying $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$ and Remarks 1–3 to $\mathsf{2KEM_{HMQV}}$, HMQV is reconstructed in Fig. 6.

Moreover, A, B are static public keys, and d, e are publicly computable, C_A, C_B can be publicly computed from $pk_{B0} = Y$ and $pk_{A0} = X$. Thus, we can apply Remark 1 to omit $C_B = XA^d$ and $C_A = YB^d$ in the communications.

[1] For formal definitions of Gap-DH and KEA1 assumptions, please refer HMQV.

$U_A : A = g^a, a$		$U_B : B = g^b, b$
$x \leftarrow \mathbb{Z}_p, X = g^x$		$y \leftarrow \mathbb{Z}_p, Y = g^y$
$d = h(X, B), C_B = XA^d$	$\xrightarrow{(C_B = XA^d,)pk_{A0} = X}$	$e = h(Y, A), C_A = YB^e$
$e = h(Y, A)$	$\xleftarrow{(C_A = YB^e,)pk_{B0} = Y}$	$d = h(X, B)$
$K_B = K'_A = \hat{H}\left((YB^e)^{x+ad}\right)$		$K_A = K'_B = \hat{H}\left((XA^d)^{y+be}\right)$
$SK \leftarrow H(si, K_B)$		$SK \leftarrow H(si, K_A)$

Fig. 6. Understanding HMQV with $\mathsf{2KEM_{HMQV}}$ in the frame $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$ where $si = (U_A, U_B, A, B, C_B, X, C_A, Y)$.

5.2 NAXOS-AKE

In [24], the 2-key KEM is initiated by $\mathsf{2KEM_{NAXOS}}$ in Fig. 7. Let G be a group of prime order p with g as a generator. Let $h : \mathbb{Z}_p \times \mathbb{Z}_p \to \mathbb{Z}_p$ and $\hat{H} : \mathbb{Z}_p \times \mathbb{Z}_p \to \{0,1\}^\lambda$ be hash functions. By applying $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$ and Remarks 1–2, in Fig. 8, we present how the NAXOS scheme is integrated in our unified framework of AKE and how it is built from the view of 2-key KEM.

$\mathsf{KeyGen1}(\lambda)$	$\mathsf{KeyGen0}(\lambda)$	$\mathsf{Encaps}(pk_1, pk_0; \mathsf{aux}_e(B, b))$;	$\mathsf{Decaps}(sk_1, sk_0, c)$
$a \leftarrow \mathbb{Z}_p$;	$x \leftarrow \mathbb{Z}_p$	$y_0 \leftarrow \mathbb{Z}_p$, $y = h(y_0, b)$	$Y \leftarrow c$;
$A = g^a$	$X = g^x$	$Y = g^y$	$x = h(x_0, a)$
$pk_1 = A$	$pk_0 = X$;	$k = \hat{H}(A^y, X^y)$	$k' = \hat{H}(Y^a, Y^x)$
$sk_1 = a$	$sk_0 = x$.	Return $k, c = Y$.	Return k'

Fig. 7. The [OW-CCA, OW-CCA] secure $\mathsf{2KEM_{NAXOS}}$ implied by NAXOS

Theorem 3. *Under the Gap-DH assumption,* $\mathsf{2KEM_{NAXOS}}$ *is* [*OW-CCA*, *OW-CCA*] *secure even with the leakage of one of* y_0 *and* b *where* $f_B(b, y_0) = h(b, y_0)$ *and* $f_{B0}(b, y_0) = h(b, y_0)$ *in the random oracle model.*

By Theorem 3, $\mathsf{2KEM_{NAXOS}}$ is [OW-CCA, OW-CCA] secure even if partial randomness (b or y_0) is leaked with respect to $f_B(b, y_0) = h(b, y_0)$ and $f_{B0}(b, y_0) = h(b, y_0)$. Obviously, $\mathsf{2KEM_{NAXOS}}$ is PKIC. We split Encaps algorithm into two steps $\mathsf{Encaps0}$ and $\mathsf{Encaps1}$. As explained in Remark 2, since b is static secret key and generated by U_B, in the execution phase U_B takes it as secret auxiliary input. Based on the above analysis, applying $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$ and Remarks 1–2 to $\mathsf{2KEM_{NAXOS}}$, NAXOS is reconstructed in Fig. 8.

Moreover, C_A is equal to $pk_{B0} = Y$ and C_B is equal to $pk_{A0} = X$. Thus we can apply Remark 2 to omit $C_B = X$ and $C_A = Y$ in the communications.

5.3 Okamoto-AKE

In Okamoto-AKE [27], the 2-key KEM is initiated by $\mathsf{2KEM_{Oka}}$ in Fig. 9. In $\mathsf{2KEM_{Oka}}$, the computation is proceeded over group G of prime order p with

$U_A : A = g^a, a$		$U_B : B = g^b, b$
$x_0 \leftarrow \mathbb{Z}_p, x = h(x_0, a)$		$y_0 \leftarrow \mathbb{Z}_p, y = h(y_0, b)$
$C_B = pk_{A0} = X = g^x$	$\xrightarrow{(C_B = X), pk_{A0} = X}$	$C_A = pk_{B0} = Y = g^y$
$K_B = \hat{H}(B^x, Y^x)$	$\xleftarrow{(C_A = Y), pk_{B0} = Y}$	$K_A = \hat{H}(A^y, X^y)$
$K'_A = \hat{H}(Y^a, Y^x)$		$K'_B = \hat{H}(X^b, X^y)$
$SK \leftarrow H(si, K'_A, K_B)$		$SK \leftarrow H(si, K_A, K'_B)$

Fig. 8. Understanding NAXOS with $\mathsf{2KEM_{naxos}}$ in the frame $\mathsf{AKE_{ro\text{-}pkic\text{-}lr}}$ where $si = (U_A, U_B, A, B, X, Y)$.

generator g, h_{tcr} is a target-collision resistant (TCR) hash function and $\bar{F}$ is a pairwise-independent random source PRF. (Please refer [27] for the formal definition of pairwise-independent random source PRFs.)

$\mathsf{2KEM_{Oka}.KeyGen1}(\lambda)$	$\mathsf{2KEM_{Oka}.KeyGen0}(\lambda)$
$a_1, a_2, a_3, a_4 \leftarrow \mathbb{Z}_p^4$, $A_1 = g_1^{a_1} g_2^{a_2}$, $A_2 = g_1^{a_3} g_2^{a_4}$	$x_3 \leftarrow \mathbb{Z}_p$, $X_3 = g_1^{x_3}$
$pk_1 = (A_1, A_2)$, $sk_1 = (a_1, a_2, a_3, a_4)$	$pk_0 = X_3$, $sk_0 = x_3$
$\mathsf{2KEM_{Oka}.Encaps}(pk_0, pk_1)$;	$\mathsf{2KEM_{Oka}.Decaps}(sk_0, sk_1, C)$
$y, y_3 \leftarrow \mathbb{Z}_p^2$, $Y_1 = g_1^y, Y_2 = g_2^y$, $Y_3 = g_1^{y_3}$	$C \in G^3$, $(Y_1, Y_2, Y_3) \leftarrow C$;
$C = (Y_1, Y_2, Y_3), c = h_{tcr}(A_1, A_2, C)$	$c = h_{tcr}(A_1, A_2, C)$
$\sigma = X_3^{y_3}(A_1 A_2^c)^y$	$\sigma' = Y_3^{x_3} Y_1^{a_1 + ca_3} Y_2^{a_2 + ca_4}$
$K = \bar{F}_\sigma(pk_0, C)$	$K' = \bar{F}_{\sigma'}(pk_0, C)$

Fig. 9. The [IND-CCA, IND-CPA] secure $\mathsf{2KEM_{Oka}}$ implied by Okamato-AKE.

Let G be a group of order p with the generator g. Let $1_G = g^p$ be the identity element. The DDH assumption states that $\{(G, g^a, g^b, g^{ab})\}_\lambda$ is computationally indistinguishable from $\{(G, g^a, g^b, g^c)\}_\lambda$, where a, b, c are randomly and independently chosen in $\mathbb{Z}_p$. If $c = ab$, (g, g^a, g^b, g^c) is called a DDH tuple, otherwise it's called a non-DDH tuple. Denote the advantage of any PPT algorithm $\mathcal{B}$ solving DDH problem as $\mathrm{Adv}^{\mathsf{ddh}}_{\mathcal{B}} = |\Pr[\mathcal{B}(g^a, g^b, g^{ab}) = 1] - \Pr[\mathcal{B}(g^a, g^b, g^c) = 1]|$.

Theorem 4. *Under the DDH assumption, if h_{tcr} is a TCR hash function and $\bar{F}$ is a pairwise-independent random source PRF, then $\mathsf{2KEM_{Oka}}$ in Fig. 9 is [IND-CCA, IND-CPA] secure in the standard model.*

Please refer the full version [32] for the formal proof of Theorem 4.

By applying $\mathsf{AKE_{std}}$, in Fig. 10, we present how the Okamato scheme is integrated in our unified framework of AKE and how it is built from the view of 2-key KEM. Let $F' : \{0,1\}^\lambda \times \{0,1\}^\lambda \rightarrow \mathbb{Z}_p$ and $F'' : \mathbb{Z}_p \times \{0,1\}^\lambda \rightarrow \mathbb{Z}_p$ be PRFs. In the frame of $\mathsf{AKE_{std}}$, by setting $s_A = a_0, s_B = b_0$, $r_A = x'_1 || x'_2, r_{A0} = x_3$, $r_B = y'_1 || y'_2$, choosing $cpk_0 = 1_G, csk_0 = p$, initiating f_A and f_B as $F'_{x'_1}(1^k) \oplus F''_{\sum_0^4 a_i}(x'_2)$ and $F'_{y'_1}(1^k) \oplus F''_{\sum_0^4 b_i}(y'_2)$, and applying $\mathsf{2KEM_{Oka}}$ as 2-key KEM, we will get Okamoto AKE in Fig. 10.

$U_A : A_1, A_2, a_1, a_2, a_3, a_4, a_0 \leftarrow \mathbb{Z}_p$		$U_B : B_1, B_2, b_1, b_2, b_3, b_4, b_0 \leftarrow \mathbb{Z}_p$
$x'_1, x'_2 \leftarrow \{0,1\}^\lambda$		$y'_1, y'_2 \leftarrow \{0,1\}^\lambda$
$(x, x_3) = F'_{x'_1}(1^k) + F''_{\sum_0^4 a_i}(x'_2)$		$(y, y_3) = F'_{y'_1}(1^k) + F''_{\sum_0^4 b_i}(y'_2)$
$X_1 = g_1^x, X_2 = g_2^x, X_3 = g_1^{x_3}$		$Y_1 = g_1^y, Y_2 = g_2^y, Y_3 = g_1^{y_3}$
$C_B = (X_1, X_2, 1_G), pk_{A0} = X_3$	$\xrightarrow{C_B, X_3}$	$C_A = (Y_1, Y_2, Y_3)$
$d = h_{tcr}(U_B, X_1, X_2)$	$\xleftarrow{C_A}$	$c = h_{tcr}(U_A, Y_1, Y_2, Y_3)$
$\sigma_B = (B_1 B_2^d)^x, K_B = \bar{F}_{\sigma_B}(1_G, C_B)$		$\sigma_A = X_3^{y_3}(A_1 A_2^c)^y, K_A = \bar{F}_{\sigma_A}(X_3, C_A)$
$c = h_{tcr}(U_A, C_A)$		$d = h_{tcr}(U_B, C_B)$
$\sigma'_A = X_3^{y_3} Y_1^{a_1+ca_3} Y_2^{a_2+ca_4}$		$\sigma'_B = X_1^{b_1+db_3} X_2^{b_2+db_4}$
$K'_A = \bar{F}_{\sigma'_A}(X_3, C_A)$		$K'_A = \bar{F}_{\sigma'_B}(1_G, C_B)$
$SK \leftarrow F_{K_B}(si) \oplus \hat{F}_{K'_A}(si)$		$SK \leftarrow F_{K'_B}(si) \oplus \hat{F}_{K_A}(si)$

Fig. 10. Understanding Okamoto-AKE from $\mathsf{2KEM_{Oka}}$ where $si = (U_A, U_B, C_B, X_3, C_A)$ in frame $\mathsf{AKE_{std}}$. Some notions are borrowed from $\mathsf{2KEM_{Oka}}$

5.4 FSXY12-AKE and FSXY13-AKE

Fujioka *et al.* in PKC 12 (called FSXY12 [15]) proposed a construction of AKE from IND-CCA secure KEM and IND-CPA secure KEM in the standard model. In FSXY12 [15], U_B sends a ciphertext of IND-CCA secure KEM and a ciphertext of IND-CPA secure KEM, and the session key is computed from these two encapsulated keys, public key of IND-CPA secure KEM, and ciphertext in the PRF functions. As we point out in Sect. 6.1, the FSXY12 scheme implies a trivial [IND-CCA, IND-CPA] secure 2-key KEM from the improved KEM combiner in the standard model. More precisely, in $\mathsf{AKE_{std}}$, cpk_0 and csk_0 is set to be empty; C_B is just $c_{B1}||$-, where c_{B1} is the ciphertext of IND-CCA secure one-key KEM under pk_B; C_A is replaced by the concatenation of $c_{A1}||c_{A0}$, where c_{A1} is the ciphertext of IND-CCA secure one-key KEM under pk_A with encapsulated key k_{A1} and c_{A0} is the ciphertext of IND-CPA secure one-key KEM under pk_{A0} with encapsulated key k_{A0}; and K_A is replaced by $F_{k_{A1}}(pk_{A0}, c_{A1}||c_{A0}) \oplus F_{k_{A0}}(pk_{A0}, c_{A1}||c_{A0})$. To make it clearer, in Sect. 6.1 we explain why we should put public key in PRFs when combining two KEMs. Note that FSXY12 implicitly did it in the same way by putting sid in PRF. Thus, due to this observation, our frame of $\mathsf{AKE_{std}}$ with improved KEM combiner can be used to explain the FSXY12 scheme.

Considering efficiency, Fujioka *et al.* in AsiaCCS 13 (called FSXY13 [16]) proposed AKE from OW-CCA secure KEM and OW-CPA secure KEM in the random oracle model. In FSXY13 [16], U_B sends a ciphertext of OW-CCA secure KEM and a ciphertext of OW-CPA secure KEM. The session key is computed from these two encapsulated keys, public key of CPA secure KEM, and ciphertext in the hashing step. As we point out in Sect. 6.1, the FSXY13 scheme implies a trivial [OW-CCA, OW-CPA] secure 2-key KEM from the improved KEM combiner in the random oracle model. Precisely, in AKE, cpk_0 and csk_0 is set to be empty; C_B is just $c_{B1}||$-, where c_{B1} is the ciphertext of OW-CCA secure one-key KEM under pk_B; C_A is replaced by the concatenation of $c_{A1}||c_{A0}$, where c_{A1} is the

ciphertext of OW-CCA secure one-key KEM under pk_A with encapsulated key k_{A1} and c_{A0} is the ciphertext of OW-CPA secure one-key KEM under pk_{A0} with encapsulated key k_{A0}; and K_A is replaced by $\hat{H}(pk_{A0}, k_1||k_{A0}, c_{A1}||c_{A0})$. In Sect. 6.1 we explain why we should put public key in hashing step when combining two KEMs. Note that FSXY13 implicitly did it in the same way by putting sid in hashing step. Thus, our frame of AKE with improved KEM combiner works for explaining the FSXY13 scheme.

6 More General Constructions for 2-Key KEM

In this section we investigate how to improve the KEM combiner [17] and Fujisaki-Okamoto transformation [14,18] so as to yield more general constructions of 2-key KEM, which are much more well-suited for lattice assumptions.

6.1 Improved Combiner of Two KEMs

Giacon *et al.* [17] propose two KEM combiner and yield a new single KEM that is classical CCA secure as long as one of the ingredient KEMs is. We show that the simple KEM combiner does not work for our 2-key KEM. Furthermore, we show that with a slight but vital modification the combiner could work.

6.1.1 The Failure to Imply [OW-CCA, ·] Secure 2key KEM from KEM Combiner

We give a scheme that is a CCA secure two KEM combiner but is not $[\mathsf{OW\text{-}CCA}, \cdot]$ secure.

Let h and H be hash functions. Let $G =< g >$ be a group with prime order p. Let $pk_1 = (g_1, g_2 = g_1^a), sk_1 = a$, the ciphertext be $c_1 = (g_1^r, g_2^r \cdot m)$ where $r = h(m)$ and the encapsulated key be $k_1 = H(m)$. By the FO transformation [14] and DDH assumption, the first KEM is one-way-CCA secure. Let $pk_0 = (h_1, h_2 = h_1^b), sk_0 = b$, the ciphertext be $c_0 = h_1^x$ and the encapsulated key be $k_0 = H(h_2^x)$; and obviously the second KEM is IND-CPA secure.

Let the combined ciphertext be $(c_1||c_0)$ and combined encapsulated key be $K = \hat{H}(k_1||k_0, c_1||c_0)$, by the KEM combiner [17] (Lemma 6 and Example 3 in [17]), the combined KEM is CCA secure. However, such combined KEM is not $[\mathsf{OW\text{-}CCA}, \cdot]$ secure which means there exists an adversary $\mathcal{A}$ that can break $[\mathsf{OW\text{-}CCA}, \cdot]$ game.

Note that $c_0 = h_1^x$ encapsulates the key $k_0^* = H(h_2^x)$ under public key $pk_0 = (h_1, h_2)$ while it encapsulates the same key $k_0^* = H(h_2^x)$ under public key $pk_0 = (h_1^c, h_2^c)$ for some $c \in \mathbb{Z}_p$. The $[\mathsf{OW\text{-}CCA}, \cdot]$ adversary $\mathcal{A}$ first queries the $\mathcal{O}_{\mathsf{leak}}$ oracle and gets $pk_0 = (h_1, h_2)$. Then it randomly chooses $c \in \mathbb{Z}_p$ and sets $pk_0^* = (h_1^c, h_2^c)$. After receiving $c_1^*||c_0^*$ under public keys pk_1 and pk_0^*, $\mathcal{A}$ queries the decryption oracle with $(pk_1, pk_0, c_1^*||c_0^*)$, and would receive exactly $K^* = \hat{H}(k_1^*||k_0^*, c_1^*||c_0^*)$.

6.1.2 Improvement on KEM Combiner to Achieve [CCA, CPA] Secure 2-Key KEM

Inspired by the attacks above, we propose a improved combiner of CCA secure and CPA secure KEMs to achieve [CCA, CPA] secure 2-key KEM. Let $\mathsf{KEM}_{\mathsf{cca}} = (\mathsf{KeyGen}_{\mathsf{cca}}, \mathsf{Encaps}_{\mathsf{cpa}}, \mathsf{Decaps}_{\mathsf{cca}})$ be IND-CCA secure KEM, $\mathsf{KEM}_{\mathsf{cpa}} = (\mathsf{KeyGen}_{\mathsf{cpa}}, \mathsf{Encaps}_{\mathsf{cpa}}, \mathsf{Decaps}_{\mathsf{cpa}})$ be IND-CPA secure KEM. Let $\hat{H}$ be a hash function and F be a PRF. The improved combiner is shown in Fig. 11, where function $f(pk_0, k_1||k_0, c)$ can be initiated by $\hat{H}(pk_0, k_1||k_0, c)$ or $F_{k_1}(pk_0, c) \oplus F_{k_0}(pk_0, c)$. Our main modification is to take public key as input to the hash function or PRF when generating encapsulated key.

$\mathsf{KeyGen1}(\lambda)$	$\mathsf{KeyGen0}(\lambda)$	$\mathsf{Enc}(pk_1, pk_0)$;	$\mathsf{Dec}(sk_1, sk_0, c_1\|\|c_0)$
$(pk_1, sk_1) \leftarrow$	$(pk_0, sk_0) \leftarrow$	$(c_1, k_1) \leftarrow \mathsf{Encaps}_{\mathsf{cca}}(pk_1)$	$k_1 \leftarrow \mathsf{Decaps}_{\mathsf{cca}}(sk_1, c_1)$
$\mathsf{KeyGen}_{\mathsf{cca}}$	$\mathsf{KeyGen}_{\mathsf{cpa}}$	$(c_0, k_0) \leftarrow \mathsf{Encaps}_{\mathsf{cpa}}(pk_0)$	$k_0 \leftarrow \mathsf{Decaps}_{\mathsf{cpa}}(sk_0, c_0)$
		$c = c_1\|\|c_0,\ k = f(pk_0, k_1\|\|k_0, c)$	$k = f(pk_0, k_1\|\|k_0, c)$

Fig. 11. The [CCA, CPA] secure $2\mathsf{KEM}_f$ in random oracle or standard model depending on the instantiation of $f(pk_0, k_1||k_0, c)$.

Theorem 5. *Let the underlying two KEMs be IND-CCA and IND-CPA secure. If $f(pk_0, k_1||k_0, c) = \hat{H}(pk_0, k_1||k_0, c)$ for a hash function $\hat{H}$, $2\mathsf{KEM}_f$ in Fig. 11 is [OW-CCA, OW-CCA] secure in random oracle model; if $f(pk_0, k_1||k_0, c) = F_{k_1}(pk_0, c) \oplus F_{k_0}(pk_0, c)$ for PRF F, $2\mathsf{KEM}_f$ in Fig. 11 is [IND-CCA, IND-CPA] secure in standard model.*

Please refer the full version [32] for the proof.

6.2 Modified FO Transformation

In this section, we investigate the constructions of passively 2-key PKE and give a modified FO transformation which can be used to transform a passively secure 2-key PKE to an adaptively secure 2-key KEM.

6.2.1 Passively Secure 2-Key PKE

As the preparation for realizing adaptively secure 2-key KEM and the modified FO transformation, similar to the notion of 2-key KEM, we can also provide the notion of 2-key (public key encryption) PKE.

Informally, a 2-key PKE $2\mathsf{PKE}$=(KeyGen0, KeyGen1, Enc, Dec) is a quadruple of PPT algorithms together with a plaintext space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$, where KeyGen1 outputs a pair of public and secret keys (pk_1, sk_1), KeyGen0 outputs a pair of keys (pk_0, sk_0), $\mathsf{Enc}(pk_1, pk_0, m)$ outputs the ciphertext $C \in \mathcal{C}$, and $\mathsf{Dec}(sk_1, sk_0, C)$ outputs a plaintext m. Sometimes, we explicitly add the randomness r to Enc and denote it as $\mathsf{Enc}(pk_1, pk_0, m, r)$. Here we only describe

Game IND-CPA on pk_1	**Game IND-CPA on** pk_0
01 $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}(pp)$	15 $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}(pp)$
02 $L_0 = \{(-,-,-)\}$	16 $L_1 = \{(-,-,-)\}$
03 $(state, pk_0^*, m_1, m_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{leak}_0}}(pk_1)$	17 $(state, pk_1^*, m_0, m_1) \leftarrow \mathcal{B}_1^{\mathcal{O}_{\mathsf{leak}_1}}(pk_0)$
04 $b \leftarrow \{0,1\}$;	18 $b \leftarrow \{0,1\}$
05 $c^* \leftarrow \mathsf{Enc}(pk_1, pk_0^*, m_b)$;	19 $c^* \leftarrow \mathsf{Enc}(pk_1^*, pk_0, m_b)$;
06 $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{leak}_0}}(state, c^*)$	20 $b' \leftarrow \mathcal{B}_2^{\mathcal{O}_{\mathsf{leak}_1}}(state, c^*)$
07 return $b' \stackrel{?}{=} b$	21 return $b' \stackrel{?}{=} b$

Fig. 12. The $[\mathsf{IND\text{-}CPA}, \cdot]$, and $[\cdot, \mathsf{IND\text{-}CPA}]$ games of 2PKE for adversaries $\mathcal{A}$ and $\mathcal{B}$.

the $[\mathsf{IND\text{-}CPA}, \mathsf{IND\text{-}CPA}]$ security game in Fig. 12. For more concrete and full definition of 2-key PKE please refer the full version [32].

Passively Secure Twin-ElGamal from DDH Assumption. Our construction is actually a conjoined ElGamal encryption. Let's call it twin-ElGamal. The $[\mathsf{IND\text{-}CPA}, \mathsf{IND\text{-}CPA}]$ secure twin-ElGamal $\mathsf{2PKE}_{\mathsf{cpaddh}} = (\mathsf{KeyGen1}, \mathsf{KeyGen0}, \mathsf{Enc}, \mathsf{Dec})$ is presented in detail in Fig. 13.

$\mathsf{KeyGen1}(\lambda)$	$\mathsf{KeyGen0}(\lambda)$	$\mathsf{Enc}(pk_1, pk_0, m)$;	$\mathsf{Dec}(sk_0, sk_1, C)$
$a_1 \leftarrow \mathbb{Z}_p$, $h_1 = g^{a_1}$;	$a_0 \leftarrow \mathbb{Z}_p$, $h_0 = g^{a_0}$;	$r_1, r_0 \leftarrow \mathbb{Z}_p$	$(c_1, c_2, c_3) \leftarrow C$
$pk_1 = (g, h_1)$, $sk_1 = a_1$	$pk_0 = (g, h_0)$, $sk_0 = a_0$	$c = g^{r_1}, g^{r_0}, h_1^{r_1} h_0^{r_0} \cdot m$	$m' = c_3 / c_1^{a_1} c_2^{a_0}$

Fig. 13. The $[\mathsf{IND\text{-}CPA}, \mathsf{IND\text{-}CPA}]$ secure $\mathsf{2PKE}_{\mathsf{cpaddh}}$ under DDH assumption

Theorem 6. *Under the DDH assumption, the twin-ElGamal* $\mathsf{2PKE}_{\mathsf{cpaddh}}$ *scheme shown in Fig. 13 is* $[\mathsf{IND\text{-}CPA}, \mathsf{IND\text{-}CPA}]$ *secure.*

Please refer the full version [32] for the proof.

6.2.2 Modified FO Transformation from Passive to Adaptive Security

In the random oracle model, the FO [14,18] technique is able to transform a passively secure one-key encryption scheme to an adaptively secure scheme. We show that the classical FO transformation does not work for our 2-key encryption scheme. Then we show that with a slight but vital modification the FO transformation could work.

The Failure of Classical FO Transform on 2-key KEM. We give a novel twin-ElGamal scheme by injecting redundant public keys, and show that such twin-ElGamal scheme after FO transformation is still OW-CCA secure, but not $[\mathsf{OW\text{-}CCA}, \cdot]$ secure.

The $\mathsf{KeyGen0}$ algorithm of $\mathsf{2PKE}_{\mathsf{cpaddh}}$ chooses a random $z \leftarrow \mathbb{Z}_p$, and sets $pk_0 = (g, h_0, g_0 = g^z), sk_0 = (a_0, z)$. The algorithm $\mathsf{KeyGen1}, \mathsf{Enc}, \mathsf{Dec}$ are the

same as in $\mathsf{2PKE}_{\mathsf{cpaddh}}$. Obviously this novel twin-ElGamal scheme is IND-CPA secure under DDH assumption. Let $\mathsf{2PKE}^{\mathsf{fo}}_{\mathsf{cpaddh}}$ be the scheme by applying classical FO transform on the novel twin-Elgamal. It is OW-CCA secure. Note that the encapsulated key is $K = H(m, c)$ where H is a hash function.

However, there exists an $[\mathsf{IND\text{-}CCA}, \cdot]$ attacker $\mathcal{A}$ of $\mathsf{2PKE}^{\mathsf{fo}}_{\mathsf{cpaddh}}$ that works as follows: $\mathcal{A}$ first queries the $\mathcal{O}_{\mathsf{leak}_0}$ and gets $pk_0^1 = (g, h_0, g_0 = g^z), sk_0^1 = (a_0, z)$. Then $\mathcal{A}$ chooses $g_0' \neq g_0 \in \mathbb{G}$, and sets $pk_0^* = (g, h_0, g_0')$ as challenge public key. On receiving challenge ciphertext c^* under (pk_1, pk_0^*), $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{ow\text{-}cca}}$ with (pk_0^1, c^*). Since $pk_0^1 \neq pk_0^*$, $\mathcal{O}_{\mathsf{ow\text{-}cca}}$ would return K'. $\mathcal{A}$ just outputs K'. Since c^* encapsulated the same key $K^* = H(m, c^*)$ under both public keys (pk_1, pk_0^1) and (pk_1, pk_0^*). $\mathcal{A}$ will succeed with probability 1.

Modification on FO Transform to Achieve [IND-CCA, IND-CCA] Secure 2-Key KEM from 2-Key PKE. Motivated by the above attacks, we give a modified FO transform by a slight but vital modification from "*Hashing*" in [18] to "*Hashing with public key as input*". Actually, taking the public keys as input to hash function is also motivated by the fact that: from the perspective of proof, "Hashing with public key as input" would help to preserve the consistency of strong decryption oracle and hashing list.

Since we take the decryption failure into account, let's firstly recall and adapt the definition of correctness for decryption in [18] to our 2-key setting. When $\mathsf{2PKE} = \mathsf{2PKE}^G$ is defined with respect to a random oracle G, it is said to be δ_{q_G}-correct if for adversary $\mathcal{A}$ making at most q_G queries to random oracle G, it holds that $\Pr[\mathsf{COR\text{-}RO}^{\mathcal{A}_{\mathsf{2PKE}}} \Rightarrow 1] \leq \delta_{q_G}$, where the correctness game COR-RO is defined as following: $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}(pp)$, $(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}(pp)$, $m \leftarrow \mathcal{A}^{G(\cdot)}(pk_1, sk_1, pk_0, sk_0)$, $c \leftarrow \mathsf{Enc}(pk_1, pk_0, m)$. Return $\mathsf{Dec}(sk_1, sk_0, c) \stackrel{?}{=} m$.

Let $\mathsf{2PKE} = (\mathsf{KeyGen1}', \mathsf{KeyGen0}', \mathsf{Enc}, \mathsf{Dec})$ be a [IND-CPA, IND-CPA] secure 2-key PKE with message space $\mathcal{M}$. The [IND-CCA, IND-CCA] secure $\mathsf{2KEM} = (\mathsf{KeyGen1}, \mathsf{KeyGen0}, \mathsf{Encaps}, \mathsf{Decaps})$ are described as in Fig. 14.

$\mathsf{KeyGen1}(\lambda)$	$\mathsf{KeyGen0}(\lambda)$
$(pk_1', sk_1') \leftarrow \mathsf{KeyGen1}'$, $s_1 \leftarrow \{0,1\}^l$;	$(pk_0', sk_0') \leftarrow \mathsf{KeyGen0}'$, $s_0 \leftarrow \{0,1\}^l$
$sk_1 = (sk_1', s_1)$, $pk_1 = pk_1'$	$sk_0 = (sk_0', s_0)$, $pk_0 = pk_0'$;
$\mathsf{Encaps}(pk_1, pk_0)$;	$\mathsf{Decaps}(sk_1, sk_0, c)$
$m \leftarrow \mathcal{M}$	$sk_1 = (sk_1', s_1)$, $sk_0 = (sk_0', s_0)$
$c \leftarrow \mathsf{Enc}(pk_1, pk_0, m; G(m))$	$m' = \mathsf{Dec}(sk_1', sk_0', c)$
$K = H(pk_1, pk_0, m, c)$;	$c' = \mathsf{Enc}(pk_1, pk_0, m'; G(m'))$
return (K, c)	if $m' = \perp$ or $c \neq c'$, let $m' = s_1 \| s_0$
	return $K = H(pk_1, pk_0, m', c)$

Fig. 14. The [IND-CCA, IND-CCA] secure 2-key KEM 2KEM by modified FO

Theorem 7. *For any* $[\mathsf{IND\text{-}CCA}, \cdot]$ *adversary* $\mathcal{C}$, *or* $[\cdot, \mathsf{IND\text{-}CCA}]$ *adversary* $\mathcal{D}$ *against* 2KEM *with at most* q_D *queries to decapsulation oracle* DECAPS, q_H

(resp. q_G) queries to random oracle H (resp. G), there are $[\mathsf{IND\text{-}CPA}, \cdot]$ *adversary* $\mathcal{A}$*, or* $[\cdot, \mathsf{IND\text{-}CPA}]$ *adversary* $\mathcal{B}$ *against 2PKE, that make at most q_H (resp. q_G) queries to random oracle H (resp. G) s.t.*

$$Adv_{2KEM}^{[\mathsf{IND\text{-}CCA},\cdot]}(\mathcal{C}) \leq \frac{q_H}{2^l} + \frac{q_H + 1}{|M|} + q_G \cdot \delta + 4Adv_{2PKE}^{[\mathsf{IND\text{-}CPA},\cdot]}(\mathcal{A}).$$

Please refer the full version [32] for the proof.

7 Efficient Post-quantum AKE from Module-LWE

With the above analysis and tools, we give a more compact AKE from Module-LWE assumption with less communications than Kyber [3]. The roadmap is that we first give a [IND-CPA, IND-CPA] secure 2-key PKE from Module-LWE, by applying the modified FO transform in Sect. 6.2.2 and the AKE in Sect. 4.1 step by step, and we finally obtain a AKE scheme.

Let q be a prime and R_q denote the ring $\mathbb{Z}_q[x]/(x^n+1)$. Define the centered binomial distribution B_η for positive integer η as: sample $(a_1, \cdots, a_\eta, b_1, \cdots, b_\eta)$ uniformly from $\{0,1\}$, and output $\sum_{i=1}^{\eta}(a_i - b_i)$. Denote $\mathbf{s} \leftarrow \beta_\eta$ as that each of $\mathbf{s}$'s coefficient is generated according to B_η. Let k, m be a positive integer parameter. For PPT adversary $\mathcal{A}$, the advantage $\mathbf{Adv}_{m,k,\eta}^{mlwe}(\mathcal{A})$ of solving Module-LWE problem is the advantage of distinguishing two distributions $\{(\mathbf{A} \leftarrow R_q^{m\times k}, \mathbf{As}+\mathbf{e}) | (\mathbf{s},\mathbf{e}) \leftarrow \beta_\eta^k \times \beta_\eta^k\}$ and $\{(\mathbf{A} \leftarrow R_q^{m\times k}, \mathbf{b} \leftarrow R_q^m)\}$.

Let $d_{t_1}, d_{t_0}, d_{u_1}, d_{u_0}, d_v$ be positive numbers, depending on the special choice of the parameters settings, and $n = 256$. Every message in $\mathcal{M} = \{0,1\}^n$ can be seen as a polynomial in R_q with coefficients in $\{0,1\}$. Let $\mathbf{A}$ be a random $k \times k$ matrix in R_q. Let $\lceil x \rfloor$ be the rounding of x to the closest integer. For distribution X, let $\sim X = \mathsf{Samp}(r)$ be sample algorithm with randomness r according to distribution X.

For an even (resp. odd) positive integer α, we define $r' = r \mod^{\pm} \alpha$ to be the unique element r' in the range $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (resp. $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$) such that $r' = r \mod \alpha$. For any positive integer α, define $r' = r \mod^{+} \alpha$ to be the unique element r' in the range $0 < r' < \alpha$ such that $r' = r \mod \alpha$. When the exact representation is not important, we simplify it as $r \mod \alpha$. For $x \in \mathbb{Q}$, $d \leq \log_2 q$, define the compress function as $\mathsf{Comp}_q(x,d) = \lceil (2^d)/q \cdot x \rfloor \mod^{+} 2^d$, and the decompress function as $\mathsf{Decomp}_q(x,d) = \lceil q/(2^d) \cdot x \rfloor$. And when applying the Comp and Decomp function to $\mathbf{x}$, the procedure is applied to coefficient.

Twin-Kyber. Our construction, called twin-kyber, is an extension of kyber scheme [3] in the same conjoined way for our twin-ElGamal scheme. With the parameters above, twin-kyber $\mathsf{2PKE}_{\mathsf{mlwe}} = (\mathsf{KeyGen1}, \mathsf{KeyGen0}, \mathsf{Enc}, \mathsf{Dec})$ is shown in Fig. 15.

Theorem 8. *If there is a PPT adversary $\mathcal{A}$ against* $[\mathsf{IND\text{-}CPA}, \mathsf{IND\text{-}CPA}]$ *security of* $\mathsf{2PKE}_{\mathsf{mlwe}}$*, there exists $\mathcal{B}$ such that,* $\boldsymbol{Adv}_{\mathsf{2PKE}_{\mathsf{mlwe}}}^{[\mathsf{IND\text{-}CPA},\mathsf{IND\text{-}CPA}]}(\mathcal{A}) \leq 2\boldsymbol{Adv}_{k+1,k,\eta}^{mlwe}(\mathcal{B})$.

$\mathsf{KeyGen1}(\lambda)$	$\mathsf{KeyGen0}(\lambda)$
01 $\sigma_1 \leftarrow \{0,1\}^{256}$	05 $\sigma_0 \leftarrow \{0,1\}^{256}$
02 $(\mathbf{s_1}, \mathbf{e_1}) \sim \beta_\eta^k \times \beta_\eta^k = \mathsf{Sam}(\sigma_1)$	06 $(\mathbf{s_0}, \mathbf{e_0}) \sim \beta_\eta^k \times \beta_\eta^k = \mathsf{Sam}(\sigma_0)$
03 $\boldsymbol{t_1} = \mathsf{Comp}_q(\mathbf{As_1} + \mathbf{e_1}, d_{t_1} = \lceil \log q \rceil)$	07 $\boldsymbol{t_0} = \mathsf{Comp}_q(\mathbf{As_0} + \mathbf{e_0}, d_{t_0} = \lceil \log q \rceil)$
04 $(pk_1 = \mathbf{t_1}, sk_1 = \mathbf{s_1})$	08 $(pk_0 = \mathbf{t_0}, sk_0 = \mathbf{s_0})$
$\mathsf{Enc}(pk_1 = \mathbf{t_1}, pk_0 = \mathbf{t_0}, m \in \mathcal{M})$	$\mathsf{Dec}(sk_1 = \mathbf{s_1}, sk_0 = \mathbf{s_0}, c = (\mathbf{u_1}, \mathbf{u_0}, v)$
09 $r', r \leftarrow \{0,1\}^{256}$	15 $\mathbf{u_1} = \mathsf{Decomp}_q(\mathbf{u_1}, d_{u_1})$
10 $(\mathbf{r_1}, \mathbf{r_0}, \mathbf{e_3}, \mathbf{e_4}, e) \sim (\beta_\eta^k)^4 \times \beta_\eta = \mathsf{Sam}(r)$	16 $\mathbf{u_0} = \mathsf{Decomp}_q(\mathbf{u_0}, d_{u_0})$
11 $\mathbf{u_1} = \mathsf{Comp}_q(\mathbf{A}^T\mathbf{r_1} + \mathbf{e_3}, d_{u_1})$	17 $v = \mathsf{Decomp}_q(v, d_v)$
12 $\mathbf{u_0} = \mathsf{Comp}_q(\mathbf{A}^T\mathbf{r_0} + \mathbf{e_4}, d_{u_0})$	18 $m' = \mathsf{Comp}_q(v - \mathbf{s_1}^T\mathbf{u_1} - \mathbf{s_0}^T\mathbf{u_0}, 1)$
13 $v = \mathsf{Comp}_q(\mathbf{t_1}^T\mathbf{r_1} + \mathbf{t_0}^T\mathbf{r_0} + e + \lceil \frac{q}{2} \rfloor m, d_v)$	
14 $c = (\mathbf{u_1}, \mathbf{u_0}, v)$	

Fig. 15. The [IND-CPA, IND-CPA] secure $\mathsf{2PKE_{mlwe}}$ under Module-LWE assumption.

Please refer the full version [32] for the analysis of decryption failure and proof. By applying the modified FO transformation to $\mathsf{2PKE_{mlwe}}$, we obtain a [OW-CCA, OW-CCA] secure $\mathsf{2KEM_{mlwe}}$. Then by setting $cpk_0 = (0)^k$ and $csk_0 = (0)^k$, and integrating $\mathsf{2KEM_{mlwe}}$ to AKE in Sect. 4, a novel and efficient post-quantum AKE from Module-LWE assumption is constructed.

The parameter setting and comparison are given in Tables 5 and 6. Note that by setting $d_{t_1} = d_{t_0} = \lceil \log q \rceil$ we actually do not apply compress on public keys. (which fix one bug of the security proof in [3]). One may doubt that with $q = 3329$ we can not apply NTT technique to accelerate the multiplications of two polynomials $f(x) \times g(x)$ over R_q, since $512 \nmid 3328$. Actually, we can fix this gap. Separate $f(x) = f_B(x^2) + xf_A(x^2)$, $g(x) = g_2(x^2) + xg_1(x^2)$ into a series of odd power and a series of even power, then $f(x) \times g(x) = f_B(x^2)g_2(x^2) + (f_A(x^2)g_2(x^2) + f_B(x^2)g_1(x^2))x + f_A(x^2)g_1(x^2)x^2$. Then we can apply NTT to $f_i(y)g_j(y)$ over $Z_q[y]/(y^{128} + 1)$ by setting $y = x^2$ since $256|3328$.

Table 5. The parameters for $\mathsf{2KEM_{mlwe}}$. δ is the decryption failure.

Scheme	n	k	q	η	$(d_{t_1}, d_{t_0}, d_{u_1}, d_{u_0}, d_v)$	δ	Security Level
$\mathsf{2KEM_{mlwe}}$	256	4	3329	1	$(12, 12, 9, 9, 5)$	$2^{-174.3}$	256

Table 6. The message size for Kyber in frame of FSXY13 and ours in frame of AKE.

AKEs	Assumptions	Sec	$U_A \rightarrow U_B$ (Bytes)	$U_B \rightarrow U_A$ (Bytes)
Kyber.AKE	$\mathbf{Adv}^{mlwe}_{5,4,5}$	256	2912	3008
AKE from $\mathsf{2KEM_{mlwe}}$	$\mathbf{Adv}^{mlwe}_{5,4,5}$	256	2838	2464

Acknowledgments. Haiyang Xue was supported by the National Natural Science Foundation of China 61602473, 61672019, 61772522, and the National Cryptography Development Fund MMJJ20170116. Xianhui Lu was supported by the National Natural Science Foundation of China 61572495. Bao Li was supported by the National Natural Science Foundation of China 61772515. Jingnan He was supported by the National Natural Science Foundation of China 61672030. Bei Liang was partially supported by the STINT grant (no 3720596). This work was supported by the National 973 Program of China under Grant 2014CB340603 and the Fundamental theory and cutting edge technologyResearch Program of Institute of Information Engineering, CAS (Grant No. Y7Z0291103).

References

1. Boyd, C., Cliff, Y., Gonzalez Nieto, J., Paterson, K.G.: Efficient one-round key exchange in the standard model. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 69–83. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70500-0_6
2. Bos, J.W., Costello, C., Naehrig, M., Stebila, D.: Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In: 2015 IEEE Symposium on Security and Privacy, pp. 553–570 (2015)
3. Bos, J., et al.: CRYSTALS - Kyber: a CCA-secure module-lattice-based KEM. In: 2018 IEEE Symposium on Security and Privacy, pp. 353–367. Code is available in https://github.com/pq-crystals/kyber
4. Barbosa, M., Farshim, P.: Relations among notions of complete non-malleability: indistinguishability characterisation and efficient construction without random oracles. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 145–163. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14081-5_10
5. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_21
6. Cremers, C.J.F.: Session-state reveal is stronger than ephemeral key reveal: attacking the NAXOS authenticated key exchange protocol. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 20–33. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01957-9_2
7. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44987-6_28
8. Canetti, R., Krawczyk, H.: Security analysis of IKE's signature-based key-exchange protocol. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 143–161. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_10
9. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055717
10. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_4

11. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. SIAM J. Comput. **30**, 391–437 (2000)
12. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
13. Fischlin, M.: Completely non-malleable schemes. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 779–790. Springer, Heidelberg (2005). https://doi.org/10.1007/11523468_63
14. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_34
15. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly secure authenticated key exchange from factoring, codes, and lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 467–484. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_28
16. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In: AsiaCCS, pp. 83–94 (2013)
17. Giacon, F., Heuer, F., Poettering, B.: KEM combiners. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10769, pp. 190–218. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_7
18. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 341–371. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12
19. Kiltz, E.: Chosen-ciphertext security from tag-based encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_30
20. Kiltz, E., Pietrzak, K., Stam, M., Yung, M.: A new randomness extraction paradigm for hybrid encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 590–609. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_34
21. Krawczyk, H.: The order of encryption and authentication for protecting communications (or: How secure is SSL?). In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 310–331. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_19
22. Krawczyk, H.: SIGMA: the 'SIGn-and-MAc' approach to authenticated Diffie-Hellman and its use in the IKE protocols. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 400–425. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_24
23. Krawczyk, H.: HMQV: a high-performance secure Diffie-Hellman protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_33
24. LaMacchia, B., Lauter, K., Mityagin, A.: Stronger security of authenticated key exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75670-5_1
25. Matsumoto, T., Takashima, Y., Imai, H.: On seeking smart public-key distribution systems. Trans. IECE Jpn. **E69**(2), 99–106 (1986)
26. Menezes, A., Qu, M., Vanstone, S.: Some new key agreement protocols providing mutual implicit authentication. In: SAC 1995, pp. 22–32 (1995)

27. Okamoto, T.: Authenticated Key Exchange and Key Encapsulation Without Random Oracles. IACR ePrint report 2007/473, full version of [28]
28. Okamoto, T.: Authenticated key exchange and key encapsulation in the standard model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76900-2_29
29. Peikert, C.: Lattice cryptography for the internet. In: Mosca, M. (ed.) PQCrypto 2014. LNCS, vol. 8772, pp. 197–219. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11659-4_12
30. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: STOC 2008, pp. 187–196 (2008)
31. Wee, H.: Efficient chosen-ciphertext security via extractable hash proofs. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 314–332. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_17
32. Xue, H., Lu, X., Li, B., Liang, B., He, J.: Understanding and Constructing AKE via Double-key Key Encapsulation Mechanism IACR ePrint report 2018/817
33. Yoneyama, K.: One-round authenticated key exchange with strong forward secrecy in the standard model against constrained adversary. In: Hanaoka, G., Yamauchi, T. (eds.) IWSEC 2012. LNCS, vol. 7631, pp. 69–86. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34117-5_5
34. Yao, A.C.C., Zhao, Y.: OAKE: a new family of implicitly authenticated Diffie-Hellman protocols. In: CCS 2013, pp. 1113–1128 (2013)
35. Zhang, J., Zhang, Z., Ding, J., Snook, M., Dagdelen, Ö.: Authenticated key exchange from ideal lattices. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 719–751. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_24