Compact Authenticated Key Exchange in the Quantum Random Oracle Model

Haiyang Xue[1,2,3] ⋆, Man Ho Au[2], Rupeng Yang[2], Bei Liang[4], Haodong Jiang[5]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
haiyangxc@gmail.com
[2] Department of Computer Science, The University of Hong Kong, Hong Kong
allen.au@gmail.com, orbbyrp@gmail.com
[3] Data Assurance and Communications Security Research Center, Chinese Academy of Sciences, Beijing, China
[4] Chalmers University of Technology, Gothenburg, Sweden
lbei@chalmers.se
[5] State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, Henan, China
hdjiang13@gmail.com

Abstract. We propose a generic construction of two-message authenticated key exchange (AKE) in the quantum random oracle model (QROM). It can be seen as a QROM-secure version of X3LH-AKE [Xue et al. ASIACRYPT 2018], a generic AKE based on double-key PKE. We prove that, with some modification, the security of X3LH-AKE in QROM can be reduced to the one-way security of double-key PKE. In addition to answering several open problems on the QROM security of prior works, such as SIAKE [Xu et al. ASIACRYPT 2019], FSXY-AKE and 2Kyber-AKE, we propose a new construction, CSIAKE, based on commutative supersingular isogenies.
Our frame enjoys the following desirable features. First of all, it supports PKEs with non-perfect correctness. Secondly, the security reduction is relatively tight. In addition, the basic building block is weak and compact. Finally, the resulting AKE achieves the security in CK^+ model as strong as in X3LH-AKE, and the transformation overhead is low.

1 Introduction

Authenticated Key exchange. Since the introduction of authenticated key exchange (AKE), there has been a series of works, which are mainly specified in the following directions, security models, provably-secure instantiations based on classical or quantum-resistant assumptions. Among these constructions, some are explicitly authenticated (which generally use additional primitives, i.e., signature or MAC), while some are implicitly authenticated.

Recently, one of the most important and appealing directions is to construct strongly secure AKE against quantum attacks. Although one could achieve this goal by combining quantum-secure PKE/KEM with quantum-resistant signatures [32], the resulting explicitly authenticated schemes incur considerable overhead. An alternative approach is to construct implicit AKE from quantum-secure PKE/KEM based on established generic frameworks. For example, the framework of [16] yields to AKEs from IND-CCA secure KEMs and pseudo-random functions (PRFs) in the standard model.

Recognizing the inefficiency of [16], Fujioka et al. [17] generically construct AKE (denoted as FSXY-AKE hereafter) relying on the random oracle. Particularly, FSXY-AKE requires an adaptively secure (OW-CCA) KEM, and a parallel execution of one

⋆ This work was done while the author was in the Department of Computing, The Hong Kong Polytechnic University.

OW-CCA secure KEM and one passively-secure KEM, all of which are implied by OW-CPA secure PKE in the classical random oracle model (ROM) [16, Sec. 4]. There have been several instantiations of FSXY-AKE. Among them, those based on Module-LWE assumption [6] and supersingular isogeny [30] are the most attractive ones.

Another noteworthy framework is due to Xue et al. [37] (denoted as X3LH-AKE hereafter). Abstracted from many well-known ad-hoc implicit AKEs, X3LH-AKE generically devises AKE from a new primitive called adaptively secure double-key (2-key) KEM which could be obtained from a passively secure 2-key PKE [37, Sec. 6.2]. Observing that in FSXY the responder returns two independent ciphertexts under initiator's static and ephemeral public keys, Xue et al. [37] made use of a 2-key PKE to incorporate two individual encryptions to a single encryption under two public keys. They also formalized the security requirement for the underlying 2-key PKE, namely [OW-CPA, OW-CPA]. More precisely, in the [OW-CPA, $\cdot$] (resp. [$\cdot$, OW-CPA]) game, the adversary attempts to invert the ciphertext of a random message which is encrypted under a pair of public keys, where the first (resp. second) public key is generated honestly by the challenger, while the second (resp. first) public key is chosen by adversary.

FSXY-AKE could be taken as a special case of X3LH-AKE where 2-key PKE is the combination of two independent PKEs. The prominent advantage of X3LH-AKE is that there are several compact constructions of 2-key PKE except for the direct combination of two PKEs. Thus, following their framework, several compact AKEs are given, such as 2Kyber-AKE [37] based on Module-LWE assumption and SIAKE [38] based on supersingular isogeny. As shown in Table 1, SIAKE is more compact than that presented in [30] by following FSXY-AKE.

The Quantum ROM. Since quantum computers could execute all the off-line primitives, including hash functions, Boneh et al. [5] introduced the quantum ROM (QROM), in which the adversary has the capability to query random oracle with arbitrary superpositions. It is widely believed that proofs in the *quantum* ROM rather than *classical* ROM fulfill the security requirements against quantum adversaries.

Motivation. Although X3LH-AKE (SIAKE, 2Kyber-AKE) and FSXY-AKE lead to efficient and quantum-resistant AKEs, their security analysis are conducted in the classical ROM, leaving the studies of their securities in QROM as open problems, which motivate us to focus on in this paper.

Hövelmanns et al. [21] proposed a modified FSXY-AKE (denoted as HKSU-AKE hereafter) by eliminating intermediate hash and proved its QROM security. Since the 2-key system of FSXY-AKE consists of two independent 1-key KEMs/PKEs, the challenge falls down on the Fujisaki-Okamoto (FO, i.e., CPA PKE to CCA KEM) transformation. Thus, several recent progresses and techniques [20,23,34,2] on QROM security of FO could be used. Hövelmanns et al. [21] also re-examines the puncture technique in [34] by considering decryption error and applies it to FSXY-AKE. The drawback of puncture trick is that the underlying 1-key PKE should be IND-CPA secure (cf. OW-CPA secure PKE is sufficient in the original FSXY-AKE).

However, HKSU's technique can not be applied to X3LH-AKE and its compact instantiations, i.e., SIAKE and 2Kyber-AKE, due to the following reasons. First, compact instantiations of 2-key PKE, such as that used in SIAKE, provide only one-wayness rather than IND security. Secondly, HKSU's proof (and several others for QROM security of FO) is built indispensably on the so-called injective mapping with encryption under *fixed* public key in order to decouple the decapsulation-like oracle with secret key. However, in X3LH-AKE (specifically, when 2-key PKE is not the direct combination of two PKEs), the adversary could query decapsulation-like oracle under *many* public keys. Worse still, the adversary has the capability to choose part of the challenge public key.

Thus, the main motivation of this work is to prove the security of X3LH-AKE (and thus SIAKE, FSXY, and 2Kyber-AKE) in QROM and reduce their securities to a *weak-*

er (i.e., one-wayness) primitive, thus supplementing the state-of-the-art of quantum-resistant AKEs.

1.1 Our Contributions

We prove that, with a slight modification, X3LH-AKE converts any [OW-CPA, OW-CPA] secure 2-key PKE into a CK^+ secure AKE in QROM. Our results imply that several prior schemes, including SIAKE [38], 2Kyber-AKE [37] and FSXY [17], can be adapted to achieve QROM security. We also propose a new AKE, namely, CSIAKE, based on commutative supersingular isogenies [15].

To the best of our knowledge, SIAKE, CSIAKE and 2Kyber-AKE represent the most efficient QROM-secure AKEs under their corresponding assumptions. Fig. 1 and Table 1 summarize ours and existing transformations.

- We prove that X3LH (with a slight modification) indeed transforms 2-key PKE into an adaptively secure AKE in QROM. Our transformation is practical, not only because it relies on a weak building block, i.e., [OW-CPA, OW-CPA] secure 2-key PKE, which supports more compact instantiations, but also because it is as efficient as X3LH.
- Our result answers several open problems on the securities of previous AKEs in QROM, such as SIAKE, 2Kyber-AKE and FSXY. For FSXY, the security requirement of underlying PKE is the same as FSXY, i.e., OW-CPA, contrary to the results in [21] which require IND-CPA. Furthermore, our reduction is relatively tighter than that in HKSU [21].
- Our result also provides an alternative approach to design new post-quantum AKEs in QROM. With such guide, CSIAKE, a new construction based on commutative supersingular isogenies [15] is given.
- Our last contribution is the proof technique, namely, domain separation and *injective mapping with encryption under many public keys*. We believe that this technique, which is of independent interest, is useful for multi-user security of FO in QROM.

AKEs	Assumptions	QROM	Tool's Security	Comm. (Bytes)	Computation
FSXY [17,6]	M-LWE		OW-CPA	5920	
X3LH [37]			[OW-CPA,OW-CPA]	5302	
HKSU [21]		✓	IND-CPA	5920	
Ours		✓	[OW-CPA,OW-CPA]	5302	
FSXY [17,30]	SIDH		OW-CPA	2352	6+6 Isog.
SIAKE [38]			[OW-CPA,OW-CPA]	1788	6+5 Isog.
HKSU [21]		✓	IND-CPA	2352	6+6 Isog.
Ours		✓	[OW-CPA,OW-CPA]	1788	6+5 Isog.
HKSU [21]	CSIDH	✓	IND-CPA	2688	6+6 Isog.
Ours		✓	[OW-CPA,OW-CPA]	2028	6+5 Isog.

Table 1. Comparison of AKEs. The parameters for M-LWE and SIDH(-p751) are chosen to have the same security as AES-256, while that for CSIDH(-5280), recommended by [11], has the same security as AES-128. We do not count the computation under M-LWE assumption since it is not a bottleneck. "$x + y$ Isog." in the computation column means that the initiator (resp. responder) has to perform x (resp. y) isogeny computation.

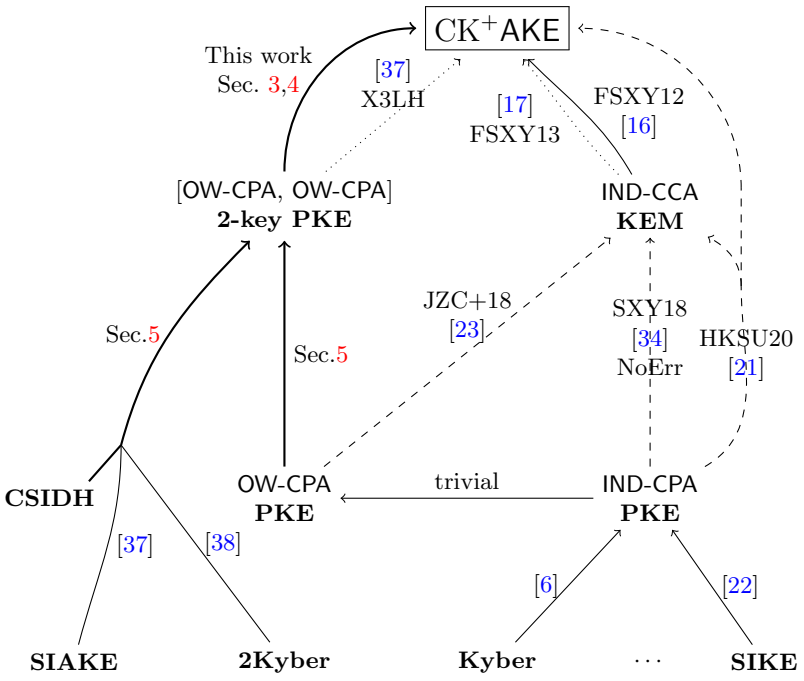


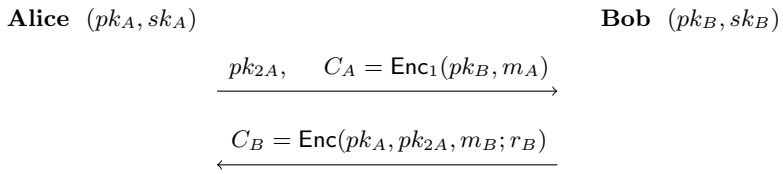
Fig. 1. Illustration of existing works from probabilistic PKE and our contributions. "NoErr" indicates the work does not consider decryption error. The dotted (resp. dashed and solid) lines indicate works in the classical ROM (resp. QROM or standard) model. The thick solid lines are our contributions.

1.2 Technique Overview

We first review the frame of X3LH-AKE in detail and discuss the challenges that arise when proving its security in QROM. Then, we present our solution. We conclude with a discussion on the applicability of our framework.

Review of X3LH-AKE and Challenges.

The core idea of X3LH-AKE from 2-key PKE[6] is illustrated in the following figure. A 2-key PKE accommodates two keys in a single encryption, and its encryption algorithm Enc takes as input the public keys pk_1, pk_2, plaintext m and randomness r, and outputs ciphertext $C = \mathsf{Enc}(pk_1, pk_2, m; r)$.



Initiator Alice, with static public key pk_A, first generates an ephemeral key pk_{2A} and sends it to Bob. Bob returns a 2-key encryption C_B to Alice. The process is somewhat symmetric in the sense that in the first move, Alice sends to Bob a 1-key encryption

[6] In [37], it is a two-step process: firstly from passively secure 2-key PKE to adaptively secure 2-key KEM, and then to X3LH-AKE.

under Bob's public key, say $C_A = \mathsf{Enc}_1(pk_B, m_A)$. The final session key is extracted from K_A (encapsulated in C_A) and K_B (encapsulated in C_B) together with the session identifier.

The $[\mathsf{OW\text{-}CPA}, \mathsf{OW\text{-}CPA}]$ security of underlying 2-key PKE is defined by $[\mathsf{OW\text{-}CPA}, \cdot]$ and $[\cdot, \mathsf{OW\text{-}CPA}]$ games. In $[\mathsf{OW\text{-}CPA}, \cdot]$ game, the adversary aims to recover m, where pk_1 is chosen by challenger and pk_2 is chosen by adversary; while in the $[\cdot, \mathsf{OW\text{-}CPA}]$ game, pk_1 (resp. pk_2) is chosen by the adversary (resp. challenger).

The AKE adversary in CK^+ game could send any message he wants, and make $\mathsf{SessionKeyReveal}$ query on any non-test session to obtain the corresponding session key. For example, on receiving an ephemeral public key from the initiator, the adversary may reply a 2-key ciphertext and query $\mathsf{SessionKeyReveal}$ to extract the corresponding session key. Furthermore, in the test session, the adversary may choose the ephemeral public key and try to guess the session key after receiving a challenge ciphertext.

To fill up the gap between CK^+ secure AKE and $[\mathsf{OW\text{-}CPA}, \mathsf{OW\text{-}CPA}]$ secure 2-key PKE, two issues should be resolved in both classical ROM and QROM. First of all, how to answer the $\mathsf{SessionKeyReveal}$ queries without the first secret key. Secondly, how to decouple the challenge session key K^* from the challenge ciphertext C^* and claim its randomness. To simplify the analysis, in the following we just consider the authentication of the initiator.

Classical ROM. With the help of hash list, two issues above could be solved by setting $r_B = G(m_B)$ and $K_B = h(pk_{2A}, m_B)$. For any $\mathsf{SessionKeyReveal}$ query on a session in which pk_1 is the owner's static key, pk_2' is the ephemeral public key and C' is chosen by adversary, the simulator returns a session key by setting the encapsulated key either as $h' = h(pk_2', m')$ or a random value, depending on whether there exists some (pk_2', m', h') in the hash list such that $\mathsf{Enc}(pk_1, pk_2', m', G(m')) = C'$. Secondly, if the plaintext m^* of C^* is in the hash list, we could solve the $[\mathsf{OW\text{-}CPA}, \cdot]$ problem; otherwise, we could claim the randomness of K^*. This is how the original X3LH [37] does.

Quantum ROM. Although searching hash list operates very well in classical ROM, it is not easy to employ this trick in QROM. For FO transform (i.e., 1-key PKE to KEM), two techniques have been proposed to address similar problems. At first, encryption is embedded into the random oracle in order to render the Decapsulation oracle simulatable without knowledge of the secret key. Secondly, the challenge key is decoupled from the challenge ciphertext by a query extraction argument (One Way-to-Hiding lemma).

If we limit 2-key PKE as a parallel execution of two independent PKEs, these two techniques in FO could be directly applied, as HKSU [21] does. However, several compact 2-key PKEs in SIAKE [38], 2Kyber [37] and CSIAKE in Sec.5 do not satisfy the premise. On the other hand, if we restrict AKE adversaries's capability (e.g., both the static and ephemeral public keys in the test session are honestly generated and fixed, and $\mathsf{SessionKeyReveal}$ can only be queried on this fixed public key.), these techniques may also be applicable. However, a realistic adversary not only might query $\mathsf{SessionKeyReveal}$ on many public keys, but also might choose one of the challenge public keys (i.e., the ephemeral public key in test session).

Thus, the compact structure of 2-key PKE brings in new challenges. We propose our techniques step by step to achieve above two main targets.

Preparation for answering the $\mathsf{SessionKeyReveal}$ query without secret key. In FO transform that turns 1-key PKE $(\mathsf{KeyGen}, \mathsf{Enc}_1, \mathsf{Dec}_1)$ into CCA KEM, randomness used in Enc_1 is replaced by $G(m)$ and the encapsulated key is set as $h(m)$. Introduced in [5] and utilized by many recent works [23,34,2,7,21,28], injective mapping with encryption is a useful technique to answer decapsalution oracle without the knowledge of secret key. As illustrated in Fig. 2, if $\mathsf{Enc}_1(pk, \cdot; G(\cdot))$ is an injective mapping (with the assumption of perfect correctness), $h(m)$ could be modeled as $h_q \circ \mathsf{Enc}_1(pk, m; G(m))$ where h_q is a private oracle. Then, the decapsulation oracle could be simulated by using h_q without the help of secret key, since we have $h(\mathsf{Dec}_1(sk, C)) = h_q \circ \mathsf{Enc}_1 \circ \mathsf{Dec}_1(sk, C) = h_q(C)$.

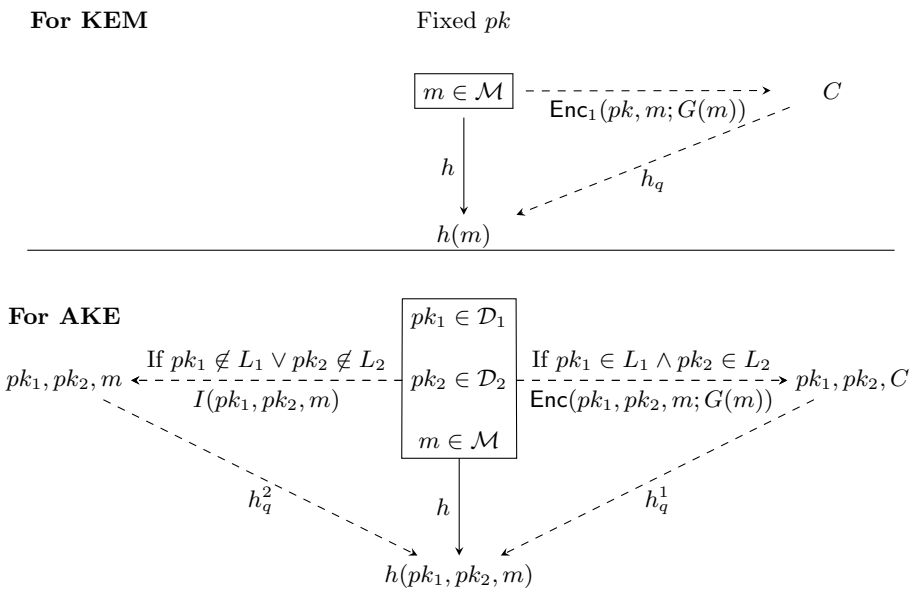


Fig. 2. The injective mapping by Enc_1 under *fixed* public key and the injective mapping by Enc under *many* public keys. pk is a fixed public key. L_1 (resp. L_2) is the list of honestly generated first public keys (resp. second public keys). I is the identical map.

Note that in the above analysis, public key should be fixed. However, in the scenario of X3LH-AKE, the adversary could query $\mathsf{SessionKeyReveal}$ on many public keys. To apply the injective mapping with encryption, public keys should be embedded in h to specify under which public keys Enc is applied, i.e., $K = h(pk_1, pk_2, m)$. However, the adversary could query quantum random oracle h with any pk_1 and pk_2 it wants, which means $\mathsf{Enc}(pk_1, pk_2, m; G(m))$ may not be *injective*, i.e., several messages may map to one ciphertext. One may come up with an idea of checking the validity of public key, however, the validity of public key is itself a much bigger challenge [27] (e.g., lattice and supersingular isogeny).

Our observation is that the $\mathsf{SessionKeyReveal}$ query applies to many but bounded public keys, i.e., those honestly generated static public keys and ephemeral public keys. Thus, although maintaining hash lists is not an easy task in QROM, the list of bounded public keys could be prepared at the very beginning. Concretely, let N be the number of users and l the number of sessions between two users. Let $L_1 := \{(pk_{1,i}, sk_{1,i})_{1 \le i \le N}\}$ be the list of honest static public-secret keys, and $L_2 := \{(pk_{2,i}^j, sk_{2,i}^j)_{1 \le i \le N, 1 \le j \le Nl}\}$ be the list prepared for the ephemeral public-secret keys, where $pk_{2,i}^j$ is used by U_i as ephemeral public key in its j-th session.

With L_1 and L_2, the domain of h, i.e., $\mathcal{D}_1 \times \mathcal{D}_2 \times \mathcal{M}$ could be separated as $L_1 \times L_2 \times \mathcal{M}$ and its complement. With such a domain separation, our technique, i.e., injective mapping with encryption under many public keys, is illustrated in Fig. 2. $h(pk_1, pk_2, m)$ is defined according to the domain separation. Concretely, if $pk_1 \in L_1 \wedge pk_2 \in L_2$, $h(pk_1, pk_2, m) = h_q^1(pk_1, pk_2, \mathsf{Enc}(pk_1, pk_2, m; G(m)))$; otherwise let $h(pk_1, pk_2, m) = h_q^2(pk_1, pk_2, m)$, where h_q^1, h_q^2 are private oracles. Such defined function $h(pk_1, pk_2, m)$ is still an injective mapping when there is no decryption error. With this replacement, we could answer $\mathsf{SessionKeyReveal}$ query on $(pk_1 \in L_1, pk_2 \in L_2, C, \cdots)$ by using $h_q^1(pk_1, pk_2, C)$ as the key encapsulated in C, i.e., without the knowledge of secret key.

Decoupling Session Key with Challenge Ciphertext. The One Way to Hiding (OW2H) lemma [36] and its variants play essential roles to decouple encapsulated key from challenge ciphertext in FO. Informally, OW2H lemma states that: if a quantum

distinguisher, issuing queries to quantum random oracle $\mathcal{O}_1$ or $\mathcal{O}_2$ which only differ on a set S, could distinguish them from each other, then there exists a one-wayness attacker to find some element in S. When applying OW2H, the way of how to use the OW2H relates to the security requirement of underlying primitive, and the choice of S is effected by the capability and aim of the adversary.

The Method of Applying OW2H. So far, there are two effective methods of applying OW2H, one is the puncture technique [34,21], and another is the unified oracle trick [23], which yield different security requirements of underlying primitives. With puncture technique that removes 0 from the message space, once applying OW2H to G on m^*, challenge ciphertext C^* can be replaced with an encryption of 0, if the underlying PKE is IND-CPA secure, thereby decoupling C^* with K^*. Whereas, unified oracle trick is to take G, h as an unified oracle $G \times h$, and apply OW2H lemma to $G \times h$ on m^*. Finding out any element in S would render to solving the OW-CPA problem; otherwise, K^* has been decoupled from C^*.

Several instantiations of 2-key PKE in X3LH (such as those of SIAKE and CSIAKE in Sec. 5) only provide one-wayness, thus, we use the unified oracle trick and take G, h as $G \times h$. To this end, we replace $G(m_B)$ with $G(pk_{2A}, m_B)$ in the protocol. Looking ahead, after the guess of the static public key in test session, say pk_1^*, any query (pk_2, m) to G could be handled as a query with (pk_1^*, pk_2, m).

The Choice of S. For the 1-key encryption and fixed pk, S could be a set consisting of a single point, i.e., the challenge message. However, in X3LH-AKE, one of the challenge public key, i.e., ephemeral public key pk_2^* in test session is chosen by the adversary. S should be carefully chosen to make sure that, on the one hand, it is large enough such that pk_2^* is covered, but on the other hand, it is not too large such that the answer for SessionKeyReveal will not be affected.

Let $L_{\mathsf{2after}} \subset L_2$ be the list of ephemeral public keys that are prepared for those sessions after the test one. $S = \{pk_1^*\} \times \mathcal{D}_2 \setminus L_{\mathsf{2after}} \times \{m_B^*\}$, where m_B^* is the challenge message, exactly is the set satisfying all requirements. 1) If ephemeral public key has high entropy, with overwhelming probability it holds that $pk_2^* \in \mathcal{D}_2 \setminus L_{\mathsf{2after}}$, which satisfies the first requirement. 2) Since m_B^* is randomly chosen by simulator, any SessionKeyReveal query before the test session meets m_B^* with negligible probability. Furthermore, by the definition of L_{2after}, any ephemeral public key in SessionKeyReveal query after test session will be in L_{2after}.Thus, the second requirement is satisfied.

Putting them together and the applicability of our framework. According to above analysis, the X3LH is modified by embedding both static and ephemeral public keys into h and embedding ephemeral public key into G. The cost of adding public keys into the hash function is negligible, while the gain is QROM security. To explain the practicality of this framework, we give several intuitions on the constructions of 2-key PKE. To illustrate the idea, we use the classical ElGamal encryption as an example. Let g be a common parameter and (g_1, h_1), (g_2, h_2) be the public keys of two ElGamal PKEs. A 2-key PKE of message $m := m_1||m_2$ can be generated under these two keys using randomness r_1, r_2 in following three approaches.

Types	Intuition	Schemes
Type 1	$[g^{r_1}, h_1^{r_1} \oplus m_1, h_2^{r_1} \oplus m_2]$	SIAKE [38], CSIAKE Sec. 5
Type 2	$[g_1^{r_1}, g_2^{r_2}, h_1^{r_1} h_2^{r_2} \oplus m]$	2Kyber [37]
Type 3	$[g_1^{r_1}, h_1^{r_1} \oplus m_1]\|\|[g_2^{r_2}, h_2^{r_2} \oplus m_2]$	FSXY [17]

Additional information about these instantiations, SIAKE, CSIAKE, 2Kyber and FSXY, are given in Sec. 5. We stress that the IND-like security of Type 1 scheme, i.e., (C)SIAKE, can not be reduced to standard assumption. Compared with [21], our transformation offers better concrete efficiency in two aspects. Firstly, the size of a 2-key PKE ciphertext is usually smaller than double sizes of a standard PKE ciphertext.

Secondly, in our transformation only one-wayness is required while theirs needs IND-CPA security.

1.3 Related Works and Open Problem

1-KEY PKE-TO-KEM. Several works [20,23,24,34,2,7,40,28] have re-examined the FO transform in QROM. They utilized the injective mapping or additional hash to avoid recording queries and also proposed different variants of OW2H lemma. Please refer to Appendix A for more details. Zhandry [40] showed a possibility for lazy-sampling and recording queries. As he commented, his proof might be looser than those using OW2H.

HASHING WITH PUBLIC KEY. The technique of hashing with public key has been used to analyze the multi-user security of Schnorr signature [3]. Recently, several submissions for the NIST Post-Quantum Cryptography Standardization, including Kyber [6], have also employed such technique. The Kyber proposed heuristic analysis from the perspective of multi-target attacks utilizing decryption failure. The necessity of putting public key into hashing is still heavily debated [10]. Our analysis in this work shows that hashing with public key seems necessary to prove the multi-user security of FO in QROM.

HKSU IN QROM. Hövelmanns et al. [21] proposed a modular framework $\mathsf{FO}_{\mathsf{AKE}}$, i.e., HKSU, from IND-CPA secure 1-key PKE in QROM. We note that when applying to FSXY, our framework in this paper needs one more re-encryption than HKSU. We take it as a compromise in order to include more compact instantiations. Firstly, there exist more compact constructions of 2-key PKE except for two parallel executions of 1-key PKEs. For example, based on (commutative) supersingular isogeny, the initiator and responder in HKSU need 6 and 6 isogeny computation respectively, while those in ours need 6 and 5. (A same computation comparison of SIAKE and FSXY has been given in [38].) Secondly, our framework relies on the one-wayness, which is weaker than indistinguishability that is relied by HKSU [21]. Furthermore, our technique has relatively tighter reduction for the factors N and l, where N is the number of users, l is the number of sessions between two users. In [21], they represent the number of sessions that the adversary established, which is bounded by N^2l, as an entirety "S" in their notation. For decryption error, ours is N and theirs is N^4l^2; for the underlying scheme, ours is N^2l and theirs is N^4l^2; and for the entropy of public keys, ours is N^2l while theirs is N^6l^3.

OPEN PROBLEM. However, in both [21] and our work, the reductions are not tight in a strict sense, since the reduction loss depends on N, l, q (the number of possible queries to random oracles), and a square root. Thus, one **open problem** is: *does there exist tighter reductions for compact AKEs in QROM?*

The reduction loss related to N and l already exists in classical ROM. Two exceptions are [8][19]. However, their models are weak, and constructions rely on stronger tools which introduce high overhead. Cohn-Gordon et al. [12] only eliminates l in their reduction. In QROM, Jiang et al. [25] found out that the square-root advantage loss in the OW2H is unavoidable, if the one-wayness attacker runs the distinguisher only once and involves no rewinding. Although, recently, double-sided OW2H lemma [7] and a new rewinding technique [28] have been proposed to reduce q and the square-root advantage loss, the reduction algorithm needs to know both $\mathcal{O}_1$ and $\mathcal{O}_2$, which can not be satisfied when the underlying probabilistic scheme is one-way secure.

2 Preliminary

In this section, we recall the definition of double-key PKE and the CK^+ security model. At last, preliminary knowledge of quantum random oracle model is given.

2.1 Double-Key PKE and Security

We revisit the definition of 2-key PKE [37]. A 2-key PKE 2PKE=(KGen1, KGen2, Enc, Dec) is a quadruple of PPT algorithms together with two public key space $\mathcal{D}_{pk_1}$, $\mathcal{D}_{pk_2}$, a plaintext space $\mathcal{M}$, a randomness space $\mathcal{R}$ and a ciphertext space $\mathcal{C}$. We want to highlight that the set membership problem of $\mathcal{D}_{pk_1}$ and $\mathcal{D}_{pk_2}$ is generally hard. Let $\mathcal{D}_1$ (resp. $\mathcal{D}_2$) be some extension of $\mathcal{D}_{pk_1}$ (resp. $\mathcal{D}_{pk_2}$) such that the set membership problem of $\mathcal{D}_1$ (resp. $\mathcal{D}_2$)is easy.

- $\mathsf{KGen1}$: on input security parameter, output public-secret key (pk_1, sk_1).
- $\mathsf{KGen2}$: on input security parameter, output public-secret key (pk_2, sk_2).
- $\mathsf{Enc}(pk_1, pk_2, m; r)$: on input public keys pk_1, pk_2, plaintext $m \in \mathcal{M}$, and randomness $r \in \mathcal{R}$, output the ciphertext $C \in \mathcal{C}$. Sometimes, we eliminate the randomness r and denote it as $\mathsf{Enc}(pk_1, pk_2, m)$ for simplicity.
- $\mathsf{Dec}(sk_1, sk_2, C)$: on input secret keys sk_1, sk_2 and cipheretext $C \in \mathcal{C}$, output a plaintext m.

Correctness and Decryption failure. The decryption failure is defined as $\delta_2 := \mathrm{E}\left(\max_{m\in\mathcal{M}} \Pr\left[\mathsf{Dec}(sk_1, sk_2, \mathsf{Enc}(pk_1, pk_2, m; r)) \neq m\right]\right)$, where the probability is taken over the randomness used in Enc and the expectation is taken over $(pk_1, sk_1) \leftarrow \mathsf{KGen1}$ and $(pk_2, sk_2) \leftarrow \mathsf{KGen2}$.

Entropy of second public key. For any $pk_2' \in \mathcal{D}_2$, $\Pr[pk_2 = pk_2' | (pk_2, sk_2) \leftarrow \mathsf{KGen2}] \leq \varepsilon_{\mathsf{pk2}}$.

One-way Security. We recall the definition of [OW-CPA, OW-CPA] security of 2-key PKE in [37]. To define [OW-CPA, OW-CPA] security for 2PKE, two adversaries, *i.e.*, $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ attacking pk_1 and $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ attacking pk_2, are taken into account. The [OW-CPA, $\cdot$] and [$\cdot$, OW-CPA] security games are shown in Fig. 3 from left to right respectively[7].

Game [OW-CPA, $\cdot$]	Game [$\cdot$, OW-CPA]
$(pk_1, sk_1) \leftarrow \mathsf{KGen1}$	$(pk_2, sk_2) \leftarrow \mathsf{KGen2}$
$(state; pk_2^*) \leftarrow \mathcal{A}_1(pk_1)$	$(state; pk_1^*) \leftarrow \mathcal{B}_1(pk_2)$
$m \leftarrow \mathcal{M}, c^* \leftarrow \mathsf{Enc}(pk_1, pk_2^*, m)$	$m \leftarrow \mathcal{M}, c^* \leftarrow \mathsf{Enc}(pk_1^*, pk_2, m)$
$m' \leftarrow \mathcal{A}_2(state, c^*)$	$m' \leftarrow \mathcal{B}_2(state, c^*)$
return $m' \stackrel{?}{=} m$	return $m' \stackrel{?}{=} m$

Fig. 3. The one-way security games for 2-key PKE.

The advantage of $\mathcal{A}$ winning in the game [OW-CPA, $\cdot$] is

$$\mathrm{Adv}_{\mathsf{2PKE}}^{[\mathsf{OW\text{-}CPA},\cdot]}(\mathcal{A}) = \Pr\left[[\mathsf{OW\text{-}CPA}, \cdot]^{\mathcal{A}} \Rightarrow 1\right].$$

We say that 2PKE is [OW-CPA, $\cdot$] secure, if for any PPT adversary $\mathcal{A}$, the advantage $\mathrm{Adv}_{\mathsf{2PKE}}^{[\mathsf{OW\text{-}CPA},\cdot]}(\mathcal{A})$ is negligible. The advantage $\mathrm{Adv}_{\mathsf{2PKE}}^{[\cdot,\mathsf{OW\text{-}CPA}]}(\mathcal{B})$ and [$\cdot$, OW-CPA] security can be defined in the same manner. If 2PKE is [OW-CPA, $\cdot$] and [$\cdot$, OW-CPA] secure, we call it is [OW-CPA, OW-CPA] secure.

[7] In the original definition [37, Sec. 6.2], a list of honestly generated public-secret keys is given to adversary. When the public key has high entropy, the definition with this list is equivalent to that without the list.

1-key PKE Let $\mathsf{PKE} = (\mathsf{KeyGen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ be a 1-key PKE with randomness space $\mathcal{R}_1$, message space $\mathcal{M}_1$ and ciphertext space $\mathcal{C}_1$. It can be taken as a special 2-key PKE where $\mathsf{KGen2}$ does nothing (such as $(-,-) \leftarrow \mathsf{KGen2}$), $\mathsf{KGen1}$, Enc, and Dec do as what $\mathsf{KeyGen}_1, \mathsf{Enc}_1$, and Dec_1 do. The decryption failure for the underlying 1-key PKE is the same as that for this 2-key PKE. The $\mathsf{OW\text{-}CPA}$ advantage $\mathrm{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{PKE}}$ of PKE is exactly the $[\mathsf{OW\text{-}CPA}, \cdot]$ advantage $\mathrm{Adv}^{[\mathsf{OW\text{-}CPA},\cdot]}$ for the specified 2-key PKE.

2.2 CK$^+$ Security Model

Here, we recall the CK$^+$ model introduced by [16,17], which is a modified CK model [13] integrated with the weak PFS security, resistant to KCI and MEX attacks. Please refer to Appendix B for a brief discussion on security models.

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 spk_i is linked to U_i's identity such that the other parties can verify the authentic binding between them. We do not require the well-formedness 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 form $(\Pi, \mathcal{I}, U_A, U_B)$ or respond to an incoming message of the form $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, 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 session specified randomness which is generally called *ephemeral secret key*, computes and maintains a *session state*, generates outgoing messages, and completes the session by outputting a session key and erasing the session state. Here 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 compute the session key SK. On the contrary, the responder U_B outputs X_B, and may compute the session key SK. We state 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. Adversary $\mathcal{A}$ is modeled as following to capture real attacks, including the control of communication and access to some secret information.

- Send(message): $\mathcal{A}$ sends messages in one of the following 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.
- $\mathsf{SessionStateReveal}(\mathsf{sid})$: $\mathcal{A}$ obtains the session state of the owner of sid if the session is not completed. The session state should be specified by the concrete protocols. We require it returns the ephemeral secret keys and some intermediate computation results except for immediately erased information.
- $\mathsf{Corrupt}(U_i)$: this query allows the adversary to learn the static secret key of user U_i. Atter this query U_i is said to be corrupted.

Freshness. Let $\mathsf{sid}^* = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ or $(\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$ be a completed session between 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 query: 1) $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$, $\mathsf{SessionKeyReveal}(\mathsf{sid}^*)$, $\mathsf{SessionStateReveal}(\overline{\mathsf{sid}^*})$, or $\mathsf{SessionKeyReveal}(\overline{\mathsf{sid}^*})$ when $\overline{\mathsf{sid}^*}$ exists; 2) $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$ or $\mathsf{SessionKeyReveal}(\mathsf{sid}^*)$ when $\overline{\mathsf{sid}^*}$ does not exist.

Security Experiment. (Quantum) adversary $\mathcal{A}$ could make a sequence of queries described above. During the experiment, $\mathcal{A}$ makes the query of $\mathsf{Test}(\mathsf{sid}^*)$. $\mathsf{Test}(\mathsf{sid}^*)$ select random bit $b \in \{0,1\}$, 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'. The advantage of adversary $\mathcal{A}$ is defined as $\mathsf{Adv}_{\Pi}^{ck+}(\mathcal{A}) = \Pr[b' = 1|b = 1] - \Pr[b' = 1|b = 0]$.

Definition 1. *We state 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 adversary $\mathcal{A}$, $\mathsf{Adv}_{\Pi}^{ck+}(\mathcal{A})$ is negligible for the test session sid^*, for any one of the cases listed in the following and Table 2. Note that in these cases except 5, when it is allowed, the ephemeral secret key or static secret key of sid^* or $\overline{\mathsf{sid}^*}$ is given to $\mathcal{A}$ directly once it is determined (for case 5, once the the test session ends).

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 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.*

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

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

2.3 The Quantum Random Oracle Model

Boneh et al. [5] introduced the quantum random oracle (QRO) model. Zhandary [39] proved that any $2q$-wise independent random function can be used to simulate the QRO allowing at most q queries. The one way-to-hiding (OW2H) lemma, initially proposed

by Unruh [36], is a useful tool for security analysis in QROM. Recently, Ambainis et al. [2] introduced the semi-classical OW2H, which is very generic and flexible. The OW2H lemma is revisited in Theorem 3 of [2] (say as revisited OW2H lemma) and it is implied by semi-classical OW2H. Such revisited OW2H lemma is more suitable for this work. The difference is that [2] considers the query depth d, while we use the number of queries q.

Lemma 1 (OW2H, Probabilities [2]). *Let $S \subseteq X$ be random. Let $\mathcal{O}_1, \mathcal{O}_2 : X \to Y$ be random functions satisfying $\forall x \notin S, \mathcal{O}_1(x) = \mathcal{O}_2(x)$. Let z be a random bitstring. $(S, \mathcal{O}_1, \mathcal{O}_2, z$ may have arbitrary joint distribution.) Let U_A be an oracle algorithm with query number q. Let $B^{\mathcal{O}_1}$ be an oracle algorithm that on input z does the following: pick $i \leftarrow 1, \cdots, q$, run $A^{\mathcal{O}_1}(z)$ until (just before) the i-th query, measure all query input registers in the computational basis, and output the set T of measurement outcomes. Let*

$$P_{left} := \Pr[b = 1 : b \leftarrow A^{\mathcal{O}_1}(z)], P_{right} := \Pr[b = 1 : b \leftarrow A^{\mathcal{O}_2}(z)],$$
$$P_{guess} := \Pr[S \cap T \neq \emptyset : T \leftarrow B^{\mathcal{O}_1}(z)].$$

Then we have $|P_{left} - P_{right}| \leq 2q\sqrt{P_{guess}}$ and $|\sqrt{P_{left}} - \sqrt{P_{right}}| \leq 2q\sqrt{P_{guess}}$.

Lemma 2 ([34]). *Let $H : \{0,1\}^l \times \mathcal{X} \to \mathcal{Y}$ and $H' : \mathcal{X} \to \mathcal{Y}$ be two independent random oracles, where l is an integer. For any unbounded time quantum adversary $\mathcal{A}$ with at most q_H times queries to H, we have*

$$\left|\Pr[\mathcal{A}^{H,H(s,\cdot)}() \to 1 | s \leftarrow \{0,1\}^l] - \Pr[\mathcal{A}^{H,H'}() \to 1]\right| \leq q_H \cdot 2^{\frac{-l+1}{2}}.$$

Let X be a finite set, and $F : X \to \{0,1\}$ be a quantum accessible oracle. Let B_{λ_x} be a Bernoulli distribution that depends on $x \in X$, that is, for each x, $\Pr[F(x) = 1] = \lambda_x$. Let λ be the upper bound of λ_x for every $x \in X$.

Lemma 3 (Generic Distinguishing Problem, [21]). *Let X be a finite set, and $\lambda \in [0,1]$. Then for any unbounded quantum algorithm $\mathcal{A}$ issuing at most q quantum queries, $\left|\Pr[\mathcal{A}^F() \to 1 | F(x) \leftarrow B_{\lambda_x}] - \Pr[\mathcal{A}^F() \to 1 | F(x) = 0]\right| \leq 8(q+1)^2\lambda$.*

3 Authenticated Key Exchange in QROM

Let $\mathsf{2PKE} = (\mathsf{KGen1}, \mathsf{KGen2}, \mathsf{Enc}, \mathsf{Dec})$ be a [OW-CPA, OW-CPA] secure 2-key PKE with public key space $\mathcal{D}_{pk_1}$ and $\mathcal{D}_{pk_2}$, randomness space $\mathcal{R} = \{0,1\}^r$, message space $\mathcal{M} = \{0,1\}^n$ and ciphertext space $\mathcal{C}$. Let $\mathsf{PKE} = (\mathsf{KeyGen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ be a OW-CPA secure 1-key PKE with randomness space $\mathcal{R}_1 = \{0,1\}^{r_1}$, message space $\{0,1\}^{n_1}$ and ciphertext space $\mathcal{C}_1$. We further require that KeyGen_1 works the same as $\mathsf{KGen1}$. This is not a strong requirement, whereas as shown in Sec.5 it is inherent.

Let $\mathcal{D}_1$ (resp. $\mathcal{D}_2$) be extension set of $\mathcal{D}_{pk_1}$ (resp. $\mathcal{D}_{pk_2}$) such that its set membership problem is easy. Let uI be the space of user's id. Let

$$\begin{aligned}
&f : \{0,1\}^{2n} \to \{0,1\}^n, && f_1 : \{0,1\}^{2n} \to \{0,1\}^{n_1},\\
&G : \mathcal{D}_2 \times \{0,1\}^n \to \{0,1\}^r, && G_1 : \mathcal{D}_1 \times \{0,1\}^{n_1} \to \{0,1\}^{r_1}\\
&h : \mathcal{D}_1 \times \mathcal{D}_2 \times \{0,1\}^n \to \{0,1\}^n, && h_1 : \mathcal{D}_1 \times \{0,1\}^{n_1} \to \{0,1\}^n,\\
&h' : \mathcal{D}_1 \times \mathcal{D}_2 \times \{0,1\}^n \times \mathcal{C} \to \{0,1\}^n, && h'_1 : \mathcal{D}_1 \times \{0,1\}^n \times \mathcal{C}_1 \to \{0,1\}^n,\\
&H : \{0,1\}^{2n} \times \mathcal{C}_1 \times \mathcal{C} \times uI^2 \to \{0,1\}^n
\end{aligned}$$

be hash functions.

Setup: Each user's static public-secret key pair is generated using $\mathsf{KGen1}$. Let $s_P, s'_P \leftarrow \{0,1\}^n$ be the static secret information for user U_P.

Protocol and Specifications: With the setup, the protocol $\mathsf{AKE}_{\mathsf{QRO}}$ between U_A and U_B is presented in Figure 4. The session state owned by U_A consists of r_{2A}, r_A. The session state owned by B consists of r_B.

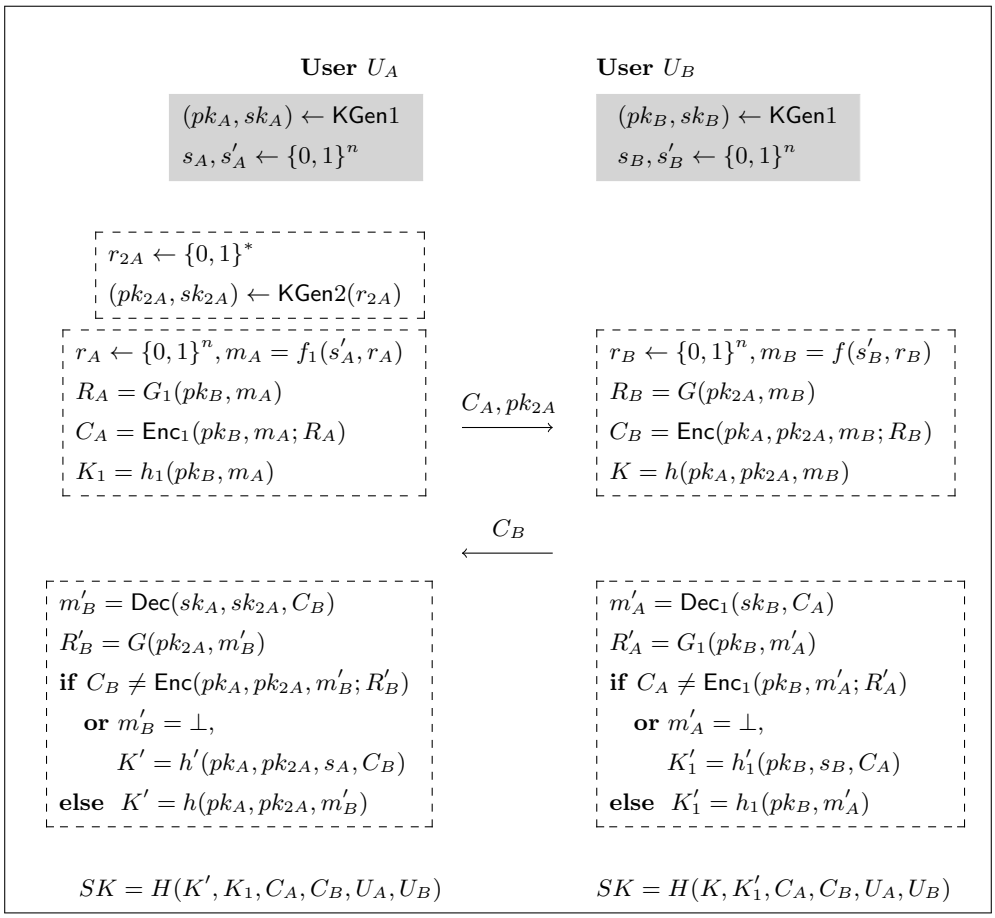


Fig. 4. $\mathsf{AKE}_{\mathsf{QRO}}$ in the QROM.

Theorem 1. *Assume 2PKE is* [*OW-CPA, OW-CPA*] *secure with decryption error* δ_2 *and PKE is OW-CPA secure with decryption error* δ_1. *N users are involved and there are at most l sessions between two users. For any quantum adversary* $\mathcal{A}$ *against* AKE_{QRO} *with at most q AKE queries, and* q_h *(resp.* q_G, q_{G_1}, q_f, q_{f_1}, q_{h_1}, $q_{h'}$, $q_{h'_1}$, q_H*) quantum queries to RO h (resp.* G, G_1 f, f_1, h_1, h', h'_1, H*), there exist* [*OW-CPA, OW-CPA*] *solvers* $\mathcal{D}$ *or* $\mathcal{C}$, *or OW-CPA solver* $\mathcal{B}$, *such that,*

$$
\begin{aligned}
\mathsf{Adv}^{ck+}_{\mathsf{AKE}_{QRO}}(\mathcal{A}) \leq\ & 4N^2 l(q_G + q_h + 2q)\sqrt{Adv^{[\mathsf{OW\text{-}CPA},\cdot]}_{\mathsf{2PKE}}(\mathcal{D})} \\
& + 2N^2 l \cdot (2q + q_H + q_f + 4)2^{\frac{-n+1}{2}} + 2N \cdot (q_{h'} + q_{h'_1} + l^2)2^{\frac{-n+1}{2}} \\
& + 16N(q_G + 2q)^2\delta_2 + 16N(q_{G_1} + 2q)^2\delta_1 + 2N^2 l\varepsilon_{pk2}, \\
\mathsf{Adv}^{ck+}_{\mathsf{AKE}_{QRO}}(\mathcal{A}) \leq\ & 4N^2 l(q_{G_1} + q_{h_1} + 2q)\sqrt{Adv^{\mathsf{OW\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{B})} \\
& + 2N^2 l \cdot (2q + q_H + q_{f_1} + 4)2^{\frac{-n+1}{2}} + 2N \cdot (q_{h'} + q_{h'_1})2^{\frac{-n+1}{2}} \\
& + 16N(q_G + 2q)^2\delta_2 + 16N(q_{G_1} + 2q)^2\delta_1, \\
\mathsf{Adv}^{ck+}_{\mathsf{AKE}_{QRO}}(\mathcal{A}) \leq\ & 4N^2 l(q_G + q_h + 2q)\sqrt{Adv^{[\cdot,\mathsf{OW\text{-}CPA}]}_{\mathsf{2PKE}}(\mathcal{C})} + 2N^2 l \cdot (q + q_H)2^{\frac{-n+1}{2}} \\
& + 2N \cdot (2q + q_{h'} + q_f)2^{\frac{-n+1}{2}}.
\end{aligned}
$$

Proof of Theorem 1 (Sketch). Here, we give a sketch of the proof to illustrate the core idea. For detailed proof please refer to sec. 4.

First of all, we assume the adversary does not make any SessionKeyReveal or SessionStateReveal query. As in the definition, the adversary falls into one of the cases from E_1 to $E_{8\text{-}2}$ in table 2. Take event E_3 as example, where the adversary sends pk^*_{2A} in the test session and he/she knows r_B but does not know sk_A, s_A, s'_A and sk_B, s_B, s'_B used in this session. From the argument for case E_3, we can easily extend the proof to other cases. By Lemma 2 which says that quantum random oracle could be used as a pseudorandom function, $m^*_B = f(s'_B, r_B)$, i.e., the message, is randomly chosen. The OW2H lemma and $[\mathsf{OW\text{-}CPA}, \cdot]$ security would guarantee the randomness of K^* and session key.

Now we consider the case that adversary may make the SessionKeyReveal queries as well. For E_5 the analysis is still the same. However, for other cases like E_3, the simulator does not hold the static-secret key sk_A of U_A. If the adversary makes SessionKeyReveal queries that involve U_A, the simulator fails to compute the encapsulated key and session key. In the classical ROM, it is easy to overcome this obstacle by searching the hash lists and taking pk_{2A} as input of h, which is how X3LH-AKE handles [37].

Whereas, to simulate SessionKeyReveal queries in QROM, we should embed the encryption under many public keys into h. Thus, both static and ephemeral public keys should be included as the inputs of h, which makes new issues arise, in particular, that public keys may not be honestly generated and the encryption may not be injective. Nevertheless, with solutions highlighted in our technique overview, we could build two lists of (honestly generated) static public keys and ephemeral public keys at the very beginning. Then, we could apply encryption-then-hash and decouple the static secret key of test session with the SessionKeyReveal oracle. Afterwards, with a careful choice of S, the OW2H lemma can be applied to argue the randomness of the session key in test session.

Concretely, for E_3, the security is argued with a sequence of games as shown in Table 3. At first, we generate two lists L_1, L_2 of honest static keys and ephemeral keys for all users and their sessions at the very beginning. Then both G and G_1 are simulated such that there is no decryption failure under all the static and ephemeral public keys in L_1, L_2. The session key for the invalid ciphertext (that involves U_A, the owner of test session) is computed with private oracles of ciphertext. Then, oracle query to h with an input (pk_1, pk_2, m) is conceptually replaced by the encryption-then-hash $h_q(pk_1, pk_2, \mathsf{Enc}(pk_1, pk_2, m; G(pk_2, m)))$ when $pk_1 \times pk_2 \in L_1 \times L_2$. We do the encryption-then-hash for $h_1(pk_1, m_1)$ with another private random oracle. Conceptually, the encapsulated key in valid ciphertext is computed with the private oracles. By integrating the decapsulation for both the valid and invalid ciphertexts, we could avoid using static secret key of U_A when answering SessionKeyReveal. In the Game 6, G and G_1 are switched back. After randomizing plaintext m^*_B in Game 7, we could replace $G \times h$ with a new oracle that differs with $G \times h$ on a set S. The set S should be carefully chosen such that the randomness and encapsulated key in test session are altered and the answer for SessionKeyReveal on any other session remains. Then, we can apply OW2H lemma and argue the distinguisher with a square root of the advantage to solve the one-wayness game of underlying 2-key PKE. Finally, since the quantum random oracle is a pseudorandom function, the session key in test session is pseudorandom as well. For all other cases the analyses are similar.

4 Formal Security Proof

To prove Theorem 1, we should handle every case in Table 2. The reduction algorithm reduces the advantage of CK^+ adversary to that of attacking $[\mathsf{OW\text{-}CPA}, \cdot]$, $[\cdot, \mathsf{OW\text{-}CPA}]$ of 2PKE or OW-CPA of PKE depending on which case we cope with. For cases E_3, E_4, E_6, $E_{7\text{-}1}$ and $E_{8\text{-}2}$, their sequences of games proceed similarly. For cases E_1, E_2, $E_{7\text{-}2}$ and $E_{8\text{-}1}$ the game sequences proceed similarly. And for case E_5 (wPFS), the sequence of games is much simpler.

Games				
	I	h (for encapsulated key)	K of valid C	R_B^*
	II	h_1 (for encapsulated key)	**Justification**	m_B^*/K^*
	III	G/G_1 (for randomness)	K of invalid C	SK^* (session key)
Games 0-1	I	$h(pk_1, pk_2, m)$	$h(pk_A, pk_{2,A}^j, m)/h_1(pk_A, m_1)$	$G(pk_{2A}^*, m_B^*)$
	II	$h_1(pk_1, m_1)$	– – – – –	$f(s'_B, r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$G(pk_2, m)/G_1(pk_1, m_1)$	$h'(pk_A, pk_{2,A}^j, s_A, C)/h'_1(pk_A, s_A, C)$	$H(K^*, \cdots)$
Game 2	I	$h(pk_1, pk_2, m)$	$h(pk_A, pk_{2,A}^j, m)/h_1(pk_A, m_1)$	$G(pk_{2A}^*, m_B^*)$
	II	$h_1(pk_1, m_1)$	**Lemma 4**	$f(s'_B, r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$\tilde{G}(pk_2, m)/\tilde{G}_1(pk_1, m_1)$	$h'(pk_A, pk_{2,A}^j, s_A, C)/h'_1(pk_A, s_A, C)$	$H(K^*, \cdots)$
Game 3	I	$h(pk_1, pk_2, m)$	$h(pk_A, pk_{2,A}^j, m)/h_1(pk_A, m_1)$	$G(pk_{2A}^*, m_B^*)$
	II	$h_1(pk_1, m_1)$	**Lemma 5**	$f(s'_B, r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$\tilde{G}(pk_2, m)/\tilde{G}_1(pk_1, m_1)$	$h'_q(pk_A, pk_{2,A}^j, C)/h'_{1q}(pk_A, C)$	$H(K^*, \cdots)$
Game 4	I	h_q^1 or h_q^2	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$G(pk_{2A}^*, m_B^*)$
	II	h_q^3 or h_q^4	**Conceptual**	$f(s'_B, r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$\tilde{G}(pk_2, m)/\tilde{G}_1(pk_1, m_1)$	$h'_q(pk_A, pk_{2,A}^j, C)/h'_{1q}(pk_A, C)$	$H(K^*, \cdots)$
Game 5	I	h_q^1 or h_q^2	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$G(pk_{2A}^*, m_B^*)$
	II	h_q^3 or h_q^4	**Conceptual**	$f(s'_B, r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$\tilde{G}(pk_2, m)/\tilde{G}_1(pk_1 m_1)$	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$H(K^*, \cdots)$
Game 6	I	h_q^1 or h_q^2	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$G(pk_{2A}^*, m_B^*)$
	II	h_q^3 or h_q^4	**Lemma 4**	$f(s'_B, r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$G(pk_2, m)/G_1(pk_1, m_1)$	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$H(K^*, \cdots)$
Game 7	I	h_q^1 or h_q^2	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$G(pk_{2A}^*, m_B^*)$
	II	h_q^3 or h_q^4	**Lemma 6**	$f_r(r_B)/h(pk_A, pk_{2A}^*, m_B^*)$
	III	$G(pk_2, m)/G_1(pk_1, m_1)$	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$H(K^*, \cdots)$
Game 8	I	h_q^1 or h_q^2	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$R_B^* \leftarrow \mathcal{R}$
	II	h_q^3 or h_q^4	**Lemma 7/OW2H**	$f_r(r_B)/K^* \leftarrow \{0,1\}^n$
	III	$G(pk_2, m)/G_1(pk_1, m_1)$	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$H(K^*, \cdots)$
Game 9	I	h_q^1 or h_q^2	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$R_B^* \leftarrow \mathcal{R}$
	II	h_q^3 or h_q^4	**Lemma 2**	$f_r(r_B)/K^* \leftarrow \{0,1\}^n$
	III	$G(pk_2, m)/G_1(pk_1, m_1)$	$h_q^1(pk_A, pk_{2,A}^j, C)/h_q^3(pk_A, C)$	$SK^* \leftarrow \{0,1\}^n$

Table 3. Overview of games for the proof of Theorem 1 w.r.t case E_3. Some details are not shown in this table, such as building lists, the guess of test session, the abort events, and some replacements of random oracles. Please refer to the Games for details. "valid C" and "invalid C" are those ciphertexts received by U_A, the owner of test session. m_B^*, R_B^*, K^* indicate the message, randomness, encapsulated key corresponding to the ciphertext in test session. SK^* is the session key of test session.

Here we take E_3 for example and show the game sequence of proof in detail, which is illustrated in Table 3. For all the other cases, we will highlight the differences of proof with E_3's proof. Let Adv_i be $|\Pr[\mathcal{A} \Rightarrow 1|b = 1 \text{ in Game } i] - \Pr[\mathcal{A} \Rightarrow 1|b = 0 \text{ in Game } i]|$.

Game 0. This is the original CK$^+$ game as defined in Section 2.2.

In this game, $\mathcal{A}$ could query Send, $\mathsf{Corrupt}$, $\mathsf{SessionKeyReveal}$ and $\mathsf{SessionStateReveal}$ oracles whenever he wants. Note that $\mathcal{A}$ is also given access to quantum ROs for $f, f_1, G, G_1, h, h_1, h', h'_1$ and H. At some point, $\mathcal{A}$ chooses a test session, and he may send messages or passively keep track of messages of test session belonging to one of the cases in Table 2. As said before, we take E_3 as an example. Then $\mathcal{A}$ receives the test session key SK^* or a totally random key depending on $b = 1$ or 0. After more queries to Send etc. oracles and quantum ROs, $\mathcal{A}$ finally outputs a bit b'. Let $\mathsf{Adv}^{ck+}_{\mathsf{AKE}_{\mathsf{QRO}}}(\mathcal{A}) = \mathsf{Adv}_0$.

Game 1. In this game, the challenger prepares honestly generated static keys and ephemeral keys for all users at the very beginning of the CK$^+$ game, in the state of on-the-fly in Game 0.

Concretely, let $L_1 := \{(pk_{1,i}, sk_{1,i})_{1\le i\le N}\}$ be the list prepared for honest static public-secret keys. Let $L_2 := \{(pk^j_{2,i}, sk^j_{2,i})_{1\le i\le N, 1\le j\le 2Nl}\}$ be the list prepared for the ephemeral public-secret keys. Specially, $(pk_{1,i}, sk_{1,i})$ is the static public-secret keys prepared for U_i and $pk^j_{2,i}$ is used by U_i as ephemeral public key in its j-th session when it's initiator[8].

Since this is only a conceptual change, we have $\mathsf{Adv}_0 = \mathsf{Adv}_1$.

Game 2. In this game, we impose a requirement that no decryption failure for both Enc and Enc_1 will occur with respect to public key pairs from $L_1 \times L_2$. The random oracle G (resp. G_1) is replaced with $\tilde{G}$ (resp. $\tilde{G}_1$) that only samples good randomness (which will be defined later) for all public keys in $L_1 \times L_2$ (resp. L_1).

For any fixed public key pairs $(pk_{1,i}, sk_{1,i}) \in L_1$, $(pk^j_{2,i}, sk^j_{2,i}) \in L_2$, $pk_2 \in \mathcal{D}_2$, and $m \in \{0,1\}^n$, define $\mathcal{R}_{\mathsf{bad}}(i, j; pk_2, m)$ as

$$\begin{cases} \{r \in \mathcal{R} | \mathsf{Dec}(sk_{1,i}, sk^j_{2,i}, \mathsf{Enc}(pk_{1,i}, pk^j_{2,i}, m; r)) \neq m\} & \text{if } pk_2 = pk^j_{2,i} \\ \emptyset & \text{o.w.} \end{cases}$$

For fixed L_1 and L_2, let $\mathcal{R}_{\mathsf{bad}}(L_1, L_2; pk_2, m) := \cup_{i\in[1,N], j\in[1,2Nl]}\mathcal{R}_{\mathsf{bad}}(i, j; pk_2, m)$ be the set of bad randomness for the encryption Enc, and let $\mathcal{R}_{\mathsf{good}}(L_1, L_2; pk_2, m) := \mathcal{R} \setminus \mathcal{R}_{\mathsf{bad}}(L_1, L_2; pk_2, m)$ be the set of good randomness accordingly.

For a fixed public key pk_1, and $m_1 \in \{0,1\}^{n_1}$, define $\mathcal{R}_{1\mathsf{bad}}(pk_1, m_1)$ as

$$\begin{cases} \{r_1 \in \mathcal{R}_1 | \mathsf{Dec}_1(sk_{1,i}, \mathsf{Enc}_1(pk_{1,i}, m_1; r_1)) \neq m_1\} & \text{if } \exists i \text{ s.t. } pk_1 = pk_{1,i} \\ \emptyset & \text{o.w.} \end{cases}$$

For a fixed L_1 and m_1, denote $\mathcal{R}_{1\mathsf{bad}}(pk_1, m_1)$ as the set of bad randomness for Enc_1 and $\mathcal{R}_{1\mathsf{good}}(pk_1, m_1) = \mathcal{R}_1 \setminus \mathcal{R}_{1\mathsf{bad}}(pk_1, m_1)$ as the set of good randomness.

Concretely, we choose internal $2(q_G + q_{G_1} + 2q)$-wise independent random functions g_q and g_{1q}. On receiving $pk_2 \times m \in \mathcal{D}_2 \times \{0,1\}^n$, $\tilde{G}$ samples and outputs an element from set $\mathcal{R}_{\mathsf{good}}(L_1, L_2; pk_2, m)$ using randomness $g_q(pk_2, m)$. On input $pk_1 \times m_1 \in \mathcal{D}_1 \times \{0,1\}^{n_1}$, $\tilde{G}_1$ samples and outputs an element from set $\mathcal{R}_{1\mathsf{good}}(pk_1, m_1)$ using randomness $g_{1q}(pk_1, m_1)$.

Lemma 4. $\mathit{Adv}_1 - \mathit{Adv}_2 \le 16(q_G + 2q)^2\delta_2 + 16(q_{G_1} + 2q)^2\delta_1$.

[8] Note that this does not mean the prepared keys are used in the real game, as the adversary may arbitrarily register an *invalid* public key for U_i, then $(pk_{1,i}, sk_{1,i})$ is not used. This also may happen for ephemeral keys, since the adversary may make Send queries. Fortunately, we do not need to answer the $\mathsf{SessionKeyReveal}$ query on those sessions.

Note that from Game 2 to 6, since the set of good randomness should be identified, simulating $\tilde{G}$ and $\tilde{G}_1$ requires unbounded power, which implies that the simulator is an unbounded algorithm. It makes sense because the differences between these games are analyzed in the information-theoretical perspective.

Game 3. In this game, we guess the owner of test session, denoted by U_A, and change the way to compute encapsulated keys for invalid ciphertexts received by U_A. Assume that pk_A is the static public key, and (sk_A, s_A, s'_A) are the static secret keys of U_A.

When U_A is an initiator. U_A receives an invalid ciphertext C_A^j and uses $pk_{2,A}^j$ as an ephemeral public key in this session, $h'(pk_A, pk_{2,A}^j, s_A, C_A^j)$ is replaced by

$$h'_q(pk_A, pk_{2,A}^j, C_A^j); \tag{1}$$

and when U_A is a responder and receives an invalid ciphertext C_A^j, $h'_1(pk_A, s_A, C_A^j)$ is replaced by

$$h'_{1q}(pk_A, C_A^j), \tag{2}$$

where $h'_q : \mathcal{D}_1 \times \mathcal{D}_2 \times \mathcal{C} \to \{0,1\}^n$ and $h'_{1q} : \mathcal{D}_1 \times \mathcal{C}_1 \to \{0,1\}^n$ are internal hash functions.

Lemma 5. $\mathsf{Adv}_2 \leq N \cdot \mathsf{Adv}_3 + 2N(2Nl + q_{h'} + q_{h'_1})2^{\frac{-n+1}{2}}$.

Game 4. We change the way to answer queries to h (resp. h_1), and also change the way to compute K (resp. K_1) from the valid ciphertext received by U_A. This game is to make preparation for getting rid of the usage of secret key sk_A during $\mathsf{SessionKeyReveal}$ queries that involve U_A.

Firstly, h (resp. h_1) is answered using two internal random oracles according to the domain separation:

$$h(pk_1, pk_2, m) = \begin{cases} h_q^1(pk_1, pk_2, \mathsf{Enc}(pk_1, pk_2, m; \tilde{G}(pk_2, m))) & \text{if } pk_1 \times pk_2 \in L_1 \times L_2 \\ h_q^2(pk_1, pk_2, m) & \text{o.w.} \end{cases}$$

$$h_1(pk_1, m_1) = \begin{cases} h_q^3(pk_1, \mathsf{Enc}_1(pk_1, m_1; \tilde{G}_1(m_1))) & \text{if } pk_1 \in L_1 \\ h_q^4(pk_1, m_1) & \text{o.w.} \end{cases}$$

where $h_q^1 : \mathcal{D}_1 \times \mathcal{D}_2 \times \mathcal{C} \to \{0,1\}^n$, $h_q^3 : \mathcal{D}_1 \times \mathcal{C}_1 \to \{0,1\}^n$, $h_q^2 : \mathcal{D}_1 \times \mathcal{D}_2 \times \{0,1\}^n \to \{0,1\}^n$, and $h_q^4 : \mathcal{D}_1 \times \{0,1\}^{n_1} \to \{0,1\}^n$ are internal oracles. Note that since $\tilde{G}$ and $\tilde{G}_1$ output good randomness, both the derandomized Enc in h_q^1 and Enc_1 in h_q^3 are injective functions on messages. Thus h and h_1 are still uniformly random. This is only a conceptual change.

Secondly, when U_A, as an initiator, receives a valid ciphertext C_A^j and uses $pk_{2,A}^j$ as ephemeral public key, $K = h(pk_A, pk_{2,A}^j, \mathsf{Dec}(sk_A, sk_{2,A}^j, C_A^j))$ is replaced by

$$h_q^1(pk_A, pk_{2,A}^j, C_A^j). \tag{3}$$

When U_A is a responder and receives a valid ciphertext C_A^j, $h_1(pk_A, \mathsf{Dec}_1(sk_A, C_A^j))$ is replaced by

$$h_q^3(pk_A, C_A^j). \tag{4}$$

This is also a conceptual replacement. And by checking the cases one by one, the replacements for encapsulated keys of valid ciphertexts are consistent with the replacements of h and h_1. The view of $\mathcal{A}$ in Game 3 and Game 4 is identical even for unbounded (quantum) adversary. Thus, $\mathsf{Adv}_3 = \mathsf{Adv}_4$.

Game 5. Now we are ready to get rid of using the secret key sk_A during $\mathsf{SessionKeyReveal}$ queries. We incorporate the ways to decapsulate K for both valid and invalid ciphertexts received by U_A. For an invalid ciphertext sent to U_A, the encapsulated key K is computed

the same as for a valid ciphertext. Concretely, Equ.(1) of Game 3 is replaced by Equ.(3), and Equ.(2) is replaced by Equ.(4).

Since h_q^1 and h_q^3 are internal oracles, the adversary can only access to h_q^1 and h_q^3 by querying h. As $\tilde{G}$ and $\tilde{G}_1$ only sample good randomness, the ciphertexts on which $\mathcal{A}$ could query to h_q^1 and h_q^3 are all valid. However, the ciphertexts on which $\mathcal{A}$ queries to h'_q, h'_{1q} are all invalid. That is, the domain consisting of all the ciphertexts on which $\mathcal{A}$ could query to h_q^1 (resp. h_q^3) disjoints with that of h'_q (resp. h'_{1q}). Switching the internal oracles when receiving invalid ciphertexts dose not change the view of an unbounded (quantum) adversary. Thus, $\mathsf{Adv}_4 = \mathsf{Adv}_5$.

Game 6. We switch back to G (resp. G_1) from $\tilde{G}$ (resp. $\tilde{G}_1$). The argument is similar to that in Game 2. $\mathsf{Adv}_5 - \mathsf{Adv}_6 \leq 16(q_G + 2q)^2\delta_2 + 16(q_{G_1} + 2q)^2\delta_1$.

Note that in Game 6 and the subsequent games, the secret key sk_A is not used any more. We are ready to decouple K^* from the challenge ciphertext in the test session.

Game 7. Let $L_{2\mathsf{after}}$ be $\{(pk_{2,A}^j, sk_{2,A}^j)_{Nl+1\leq j\leq 2Nl}\}$, a subset of L_2. After the test session, all the ephemeral public keys used by U_A will be chosen from $L_{2\mathsf{after}}$. If any public key in $L_{2\mathsf{after}}$ is equal to pk_{2A}^*, abort. Secondly, we guess the responder of test session and denote it as U_B, and also guess which session between U_A and U_B is the test session at the very beginning. If the guess fails, just abort. Thirdly, we change the generation of m_B as $m_B := f_r(r_B)$ with an internal random oracle f_r, which is identical to $m_B \leftarrow \mathcal{M}$. Particularly, in the test session $m_B^* \leftarrow \mathcal{M}$. Finally, if there exists a message used by $\mathcal{A}$ (to interacts with U_A) before the test session, which is equal to m_B^*, the game also aborts.

Lemma 6. $Adv_6 - Nl \cdot Adv_7 \leq 2Nl \cdot (q + q_f)2^{\frac{-n+1}{2}} + 2Nl \cdot \varepsilon_{pk2} + Nl^2 2^{-n+1}$.

Now, random oracles G and h can be regarded as a single oracle $G \times h$. As shown in [23], if G and h have the same domain, we can use $G \times h$ to simulate both G and h. Here, we could easily construct a G' that can simulate G with the same domain as h. For example, $G'(pk_A, pk_2, m) = G(pk_2, m)$. Then, a query (pk_2, m) to G can be directly converted to another query (pk_A, pk_2, m) to G' for fixed pk_A. Therefore, we can use $G' \times h$ to simulate both G and h. For simplicity, in the context, we stick to using $G \times h$ instead of $G' \times h$. We take $\{pk_A\} \times \mathcal{D}_2 \times \mathcal{M}$ as the domain of G. The same holds for $\ddot{G}$ and $\ddot{h}$.

Game 8. Define set $S := \{pk_A\} \times \mathcal{D}_2 \setminus L_{2\mathsf{after}} \times \{m_B^*\}$. Let $\ddot{h}$ (resp. $\ddot{G}$) be a function such that the function values on S (resp. $\mathcal{D}_2 \setminus L_{2\mathsf{after}} \times \{m_B^*\}$) are totally random, and $\ddot{h} = h$ (resp. $\ddot{G} = G$) everywhere else. In this game, $G \times h$ is replaced by $\ddot{G} \times \ddot{h}$.

A equivalent description of this game is that: $G \times h$ is still the same, but now their values on S that we provide to $\mathcal{A}$ in the CK$^+$ games are totally random. Specially, the randomess $R_B^* = G(pk_{2A}^*, m_B^*)$ for C_B^* and encapsulated key $K^* = h(pk_A, pk_{2A}^*, m_B^*)$ are replaced by random strings.

Lemma 7. $Adv_7 - Adv_8 \leq 2(q_G + q_h + 2q)\sqrt{Adv_{2PKE}^{[\mathsf{OW\text{-}CPA},\cdot]}(\mathcal{D})}$.

The formal proof for Lemma 7 is provided in the subsection below. We give a sketch of proof here. S is carefully chosen such that any $\mathsf{SessionKeyReveal}$ query on the non-test session will not need the outcome of $G \times h$ on S. Thus, even $G \times h$ is replaced by $\ddot{G} \times \ddot{h}$, h_q^1 and h_q^3 still can be used to answer the $\mathsf{SessionKeyReveal}$ queries. By applying OW2H lemma, the upper bound is a square root of the probability that one could measure some query to find some value in S. And finding out a value in S in could solve the onewayness of the underlying $\mathsf{2PKE}$.

Game 9. We change the session key SK^* to a totally random key. With a similar argument in Game 7 , by Lemma 2, $\mathsf{Adv}_8 - \mathsf{Adv}_9 \leq 2(q + q_H)2^{\frac{-n+1}{2}}$, since K^* is totally

random from the view of $\mathcal{A}$. Note that in this game, SK^* is random no mater $b = 1$ or not. Thus $\mathsf{Adv}_8 = 0$.

To sum up, we give the upper bound of AKE adversary for the case E_3 as the first in-equation in Theorem 1.

For case E_4, the proof of the game sequences is almost the same as for E_3, except that in Game 7 the AKE adversary $\mathcal{A}$ does not know r_B for E_4, while he does not know s'_B for E_3 instead. For case E_2, the difference with proof of case E_3 lies in that the role of U_A and U_B is exchanged, and the challenge ciphertext is under 1-key public key encryption in E_2, while in case E_3 it is under 2-key PKE instead. For case E_1, the proof of its game sequences is almost the same as for E_2, except that in E_1 the AKE adversary $\mathcal{A}$ does not know r_A, while he does not know s'_A in E_2 instead.

For cases E_6, E_{7-1}, E_{7-2}, E_{8-1}, E_{8-2}, the proof is almost the same as for E_3, E_1, E_4, E_2, E_3 respectively.

For case E_5, the proof is much simpler since we only need to deal with the weak perfect forward security, which means no decapsulation oracle is needed, and the injective mapping with encryption under many public keys technique can be left out. □

4.1 Proof of Lemmas

4.1.1 Proof of Lemma 4: $\mathsf{Adv}_1 - \mathsf{Adv}_2 \leq 16(q_G + 2q)^2\delta_2 + 16(q_{G_1} + 2q)^2\delta_1$.

We first define an internal Game 1-1 in which only G is replaced.

By the definition of $\mathcal{R}_{\mathsf{bad}}(i, j; pk_2, m)$ in Game 2, when $pk_2 \in L_2$, there exists i^*, j^* such that $\mathcal{R}_{\mathsf{bad}}(i, j; pk_2, m)$ is non-empty only when $i = i^*$ and $j = j^*$; when $pk_2 \notin L_2$, $\mathcal{R}_{\mathsf{bad}}(i, j; pk_2, m)$ is always empty.

Define $\delta(i, j; pk_2, m) = \frac{|\mathcal{R}_{\mathsf{bad}}(i,j;pk_2,m)|}{|\mathcal{R}|}$ and $\delta(i, j) = \max_{m\in\{0,1\}^n} \delta(i, j; pk_2, m)$. By the definition of correctness, $\mathrm{E}(\delta(i, j)) = \delta_2$ or 0, depending on $pk_2 \in L_2$ or not, where the expectation is taken over $(pk_{1,i}, sk_{1,i}) \leftarrow \mathsf{KGen1}, (pk^j_{2,i}, sk^j_{2,i}) \leftarrow \mathsf{KGen2}$. Thus, $\exists$ i^*, j^*, such that

$$\begin{aligned}\delta(L_1, L_2, pk_2, m) :=& \frac{|\mathcal{R}_{\mathsf{bad}}(L_1, L_2; pk_2, m)|}{|\mathcal{R}|} \leq \sum_{i\in[1,N],j\in[1,2Nl]} \frac{|\mathcal{R}_{\mathsf{bad}}(i, j; pk_2, m)|}{|\mathcal{R}|} \\ &= \frac{|\mathcal{R}_{\mathsf{bad}}(i^*, j^*; pk_2, m)|}{|\mathcal{R}|} = \delta(i^*, j^*; pk_2, m).\end{aligned}$$

Let $\delta(L_1, L_2) := \max^{pk_2\in L_2}_{m\in\{0,1\}^n} \delta(L_1, L_2; pk_2, m)$. By taking the expectation on $\delta(L_1, L_2)$ over the generation of L_1 and L_2, $\exists j^*$, such that

$$\mathrm{E}(\delta(L_1, L_2)) = \mathrm{E}\left(\max^{pk_2\in L_2}_{m\in\{0,1\}^n} (\delta(L_1, L_2; pk_2, m))\right) \qquad \leq \mathrm{E}(\delta(i^*, j^*)) = \delta_2,$$

where the expectation in line 2 is taken over $(pk_{1,i^*}, sk_{1,i^*}) \leftarrow \mathsf{KGen1}$ and $(pk^{j^*}_{2,i^*}, sk^{j^*}_{2,i^*}) \leftarrow \mathsf{KGen2}$.

To give the upper bound of $\mathsf{Adv}_1 - \mathsf{Adv}_{1-1}$, we utilize the distinguisher between Game 1 and Game 2 for $b = 1$ and $b = 0$ together with Lemma 3 to construct an unbounded quantum adversary $\mathcal{B}^{|F\rangle}$ to solve the generic distinguishing problem.

$\mathcal{B}$, on input randomly chosen L_1, L_2, simulates the game as in Game 1. Let $\lambda(pk_2, m) = \delta(L_1, L_2; pk_2, m)$. Let $F(pk_2, m)$ be bernoulli-distributed $B_{\lambda(pk_2,m)}$ or always 0 with respect to the generic distinguishing problem. Define G as

$$G(pk_2, m) = \begin{cases} \mathsf{Sample}(\mathcal{R}_{\mathsf{good}}(L_1, L_2; pk_2, m); g_q(m)) & \text{if } F(pk_2, m) = 0 \\ \mathsf{Sample}(\mathcal{R}_{\mathsf{bad}}(L_1, L_2; pk_2, m); g_q(m)) & \text{o.w.} \end{cases}$$

where $\mathsf{Sample}(S; r)$ outputs an element from a set S with randomness r.

When $F(pk_2, m)$ is bernoulli-distributed according to $B_{\lambda(pk_2,m)}$, G is as in Game 1; when it is always 0, G is the same as in Game 1-1. At last, $\mathcal{B}$ just returns what $\mathcal{A}$ guesses.

For both $b = 1$ and 0, $\mathcal{B}^{|F\rangle}(L_1, L_2)$ perfectly simulates Game 1-1 or Game 1 for $\mathcal{A}$ corresponding to F is always 0 or bernoulli-distributed. By further applying Lemma 3 with $\lambda = \delta(L_1, L_2)$, we have

$$\begin{aligned}
&|\Pr[b' = 1|b = 1 \text{ (resp.0) in Game 1}] - \Pr[b' = 1|b = 1 \text{ (resp.0) in Game 1-1}]| \\
&\quad = \Pr[\mathcal{B}^{|F\rangle}(L_1, L_2) \to 1|F \leftarrow B_{\lambda(pk_2,m)}] - \Pr[\mathcal{B}^{|F\rangle}(L_1, L_2) \to 1|F \equiv 0] \\
&\quad \leq 8 \cdot (q_G + 2q)^2 \delta(L_1, L_2).
\end{aligned}$$

By taking the expectation over L_1 and L_2, we have $\mathsf{Adv}_1 - \mathsf{Adv}_{1-1} \leq 16 \cdot (q_G + 2q)^2 \delta_2$.

Now, we consider the replacement of G_1. Define $\delta(pk_1, m_1) = \frac{|\mathcal{R}_{1\mathsf{bad}}(pk_1,m_1)|}{|\mathcal{R}_1|}$. Then, $\delta_1 = \mathrm{E}(\max_{pk_1,m}(\delta(pk_1, m_1)))$.

By constructing a similar unbounded quantum adversary $\mathcal{B}^{|F\rangle}$, where $F(pk_1, m_1)$ is bernoulli-distrubution $B_{\delta(pk_1,m_1)}$ or always 0, with the similar analysis for the switch of G_1, we have $\mathsf{Adv}_{1-1} - \mathsf{Adv}_2 \leq 16(q_{G_1} + 2q)^2 \delta_1$.

4.1.2 Proof of Lemma 5: $\mathsf{Adv}_2 \leq N \cdot \mathsf{Adv}_3 + 2N(2Nl + q_{h'} + q_{h'_1})2^{\frac{-n+1}{2}}$.

Let Game 2-1 be an internal game in which we guess whom is the owner of test session, i.e., U_A. If the guess is wrong, abort. Obviously, $\mathsf{Adv}_2 = N \cdot \mathsf{Adv}_{2-1}$.

To argue the difference between Adv_{2-1} and Adv_3, there are two cases that should be handled, namely, either U_A is an initiator or a responder. Here, we prove the case when U_A is an initiator. Note that s_A is totally random for $\mathcal{A}$. By Lemma 2, we construct an algorithm $\mathcal{T}$ to distinguish which oracle it is given access to, $h'_q(\cdot, \cdot, \cdot)$ or $h'(\cdot, \cdot, s_A, \cdot)$. To this end, $\mathcal{T}$ simulates the AKE game. It first guesses the owner of test session, generates the static public-secret keys for every user except U_A. For user U_A, $\mathcal{T}$ only honestly generates (pk_A, sk_A) but without knowing s_A. For an invalid ciphertext C_A^j sent to initiator U_A who uses $pk_{2,A}^j$ as ephemeral public key, $\mathcal{T}$ makes query to the challenge random oracle $h'(\cdot, \cdot, s_A, \cdot)$ or $h'_q(\cdot, \cdot, \cdot)$ with tuple $pk_A, pk_{2,A}^j, C_A^j$ depending on $\sigma = 1$ or 0, to set the encapsulated key K. After receiving b' from $\mathcal{A}$, $\mathcal{T}$ returns b' as the guess of σ.

If the challenge oracle $\mathcal{T}$ queried is $h'(\cdot, \cdot, s_A, \cdot)$, it is Game 2-1. If the oracle $\mathcal{T}$ queried is $h'_q(\cdot, \cdot, \cdot)$, it is Game 3. The number of $\mathcal{T}$'s queries to h'_q is less than $Nl + q_{h'}$. By Lemma 2, for both $b = 1$ and 0, $\Pr[b' = 1|\sigma = 1] - \Pr[b' = 1|\sigma = 0] \leq (Nl + q_{h'})2^{\frac{-n+1}{2}}$. To further consider the similar replacement of h'_1 when U_A is a responder, we have $\mathsf{Adv}_{2-1} - \mathsf{Adv}_3 \leq 2(Nl + q_{h'})2^{\frac{-n+1}{2}} + 2(Nl + q_{h'_1})2^{\frac{-n+1}{2}}$. Thus, $\mathsf{Adv}_2 \leq N \cdot \mathsf{Adv}_3 + 2N(2Nl + q_{h'} + q_{h'_1})2^{\frac{-n+1}{2}}$. □

4.1.3 Proof of Lemma 6:

$$\mathsf{Adv}_6 - Nl \cdot \mathsf{Adv}_7 \leq 2Nl \cdot (q + q_f)2^{\frac{-n+1}{2}} + 2Nl \cdot \varepsilon_{\mathsf{pk2}} + Nl^2 2^{-n+1}.$$

Define an intermediate Game 6-1, in which it aborts when there exists a public key in $L_{2\mathsf{after}}$ equal to the ephemeral public key used by adversary in test session. Obviously $\mathsf{Adv}_6 - \mathsf{Adv}_{6-1} \leq Nl \cdot \varepsilon_{\mathsf{pk2}}$.

Define an intermediate Game 6-2, in which the simulator guesses who is the responder (here assume it is U_B) of test session and which session is the test session. If the guess succeeds, go on as Game 6-1, otherwise abort. The probability of successful guess is exactly $1/Nl$. Thus, $\mathsf{Adv}_{6-1} = Nl \cdot \mathsf{Adv}_{6-2}$.

Define an intermediate Game 6-3, in which the message used by U_B is randomly chosen. Note that s'_B is totally random for $\mathcal{A}$. By Lemma 2, we construct an algorithm $\mathcal{T}$ to distinguish oracle f_r from oracle $f(s'_B, \cdot)$. Given quantum accessible random oracle

f, $\mathcal{T}$ is given access to $f(s'_B, \cdot)$ if $\sigma = 1$; otherwise $\mathcal{T}$ is given access to f_r. $\mathcal{T}$ finally outputs a guess σ' for σ. To this end, $\mathcal{T}$ simulates the AKE game. $\mathcal{T}$ honestly sets the static public/secret keys of every user except U_B. For user U_B, $\mathcal{T}$ honestly sets the static secret key but leaves s'_B as empty. For any session that involves U_B, $\mathcal{T}$ queries oracle $f(s'_B, \cdot)$ or $f_r(\cdot)$ with r_B. After receiving b' from $\mathcal{A}$ as the guess of b, $\mathcal{T}$ returns b' as the guess of σ. If $\mathcal{T}$ queried $f(s'_B, \cdot)$, the AKE game is Game 6-2. If $\mathcal{T}$ queried $f_r(\cdot)$, the AKE game is Game 6-3. By Lemma 2, for both $b = 1$ and 0, $\Pr[b' = 1|\sigma = 1] - \Pr[b' = 1|\sigma = 0] \leq (q + q_f)2^{\frac{-n+1}{2}}$. Thus $\mathsf{Adv}_{6-2} - \mathsf{Adv}_{6-3} \leq 2(q + q_f)2^{\frac{-n+1}{2}}$.

Furthermore, since now m^*_B is randomly chosen, the probability that it is equal to any message used by $\mathcal{A}$ (to interacts with U_A) before the test session, is less than $l \times 2^{-n}$.

To sum up, $\mathsf{Adv}_6 - Nl \cdot \mathsf{Adv}_7 \leq 2Nl(q + q_f)2^{\frac{-n+1}{2}} + Nl \cdot \varepsilon_{\mathsf{pk2}} + Nl^2 \times 2^{-n+1}$.

4.1.4 Proof of Lemma 7: $\mathsf{Adv}_7 - \mathsf{Adv}_8 \leq 2(q_G + q_h + 2q)\sqrt{\mathrm{Adv}^{[\mathsf{OW\text{-}CPA},\cdot]}_{\mathsf{2PKE}}(\mathcal{D})}$.

Let $\tilde{h}^1_q$ be the function, which aborts on set $S = \{pk_A\} \times \mathcal{D}_2 \setminus L_{\mathsf{2after}} \times \{m^*_B\}$, and is equal to h^1_q everywhere else. Since any $\mathsf{SessionKeyReveal}$ query on non-test session does not need h^1_q on S, $\tilde{h}^1_q$ and h^3_q could be used to answer the $\mathsf{SessionKeyReveal}$ query that involves U_A in both Game 7 and Game 8.

Define $\mathcal{S}^{G\times h}$ as the following: on input $z = (L_1, L_2, pk_A, \mathsf{Enc}(pk_A, \cdot, m^*_B; G(\cdot, m^*_B)), h(pk_A, \cdot, m^*_B), \tilde{h}^1_q, h^3_q)$, where $\mathsf{Enc}(pk_A, \cdot, m^*_B; G(\cdot, m^*_B)), h(pk_A, \cdot, m^*_B)$ can be seen as two one-time oracles, generate the static public-secret key pairs for U_P as in Game 7. Set pk_A as the static public key of U_A. For any $\mathsf{SessionKeyReveal}$ query that involves U_A, utilize $\tilde{h}^1_q$ and h^3_q to compute the encapsulated key and session key. When $\mathcal{A}$ sends pk^*_{2A} in test session, $\mathcal{S}$ computes $C^*_B := \mathsf{Enc}(pk_A, pk^*_{2A}, m^*_B; G(pk^*_{2A}, m^*_B))$, and $K^* := h(pk_A, pk^*_{2A}, m^*_B)$ with pk^*_{2A} and z. $\mathcal{S}$ chooses $b \leftarrow \{0, 1\}$. If $b = 0$, set $SK^* = H(K^*, K_1, C_A, C^*_B, U_A, U_B)$, else $SK^* = H(K, K_1, C_A, C^*_B, U_A, U_B)$, where K_1 is extracted from C_A using sk_B and $K \leftarrow \{0, 1\}^n$. $\mathcal{S}$ then returns what $\mathcal{A}$ outputs.

Thus, on input z, $\mathcal{S}^{G\times h}$ could simulate Game 7 perfectly. By replacing $G \times h$ with $\ddot{G} \times \ddot{h}$, on the same input, $\mathcal{S}^{\ddot{G}\times\ddot{h}}$ could also simulate Game 8 perfectly. Thus,

$$\mathsf{Adv}_7 = |\Pr[\mathcal{S}^{G\times h} \Rightarrow 1|b = 1] - \Pr[\mathcal{S}^{G\times h} \Rightarrow 1|b = 0]|;$$
$$\mathsf{Adv}_8 = |\Pr[\mathcal{S}^{\ddot{G}\times\ddot{h}} \Rightarrow 1|b = 1] - \Pr[\mathcal{S}^{\ddot{G}\times\ddot{h}} \Rightarrow 1|b = 0]|.$$

Let $B^{\ddot{G}\times\ddot{h}}$ be an oracle algorithm that: with the input z does as following: randomly choose $k \leftarrow \{1, \cdots, q_G + q_h\}$, run $\mathcal{S}^{\ddot{G}\times\ddot{h}}(z)$ until (exactly before the starting of) the k-th query, measure all queried input registers in the computational basis, and output the measurement as outcomes.

For $b = 0$ and 1, applying the OW2H (Lemma 1) by setting $X = \mathcal{D}_1 \times \mathcal{D}_2 \times \mathcal{M}$, $Y = \{0, 1\}^{r+n}$, $S = \{pk_A\} \times \mathcal{D}_2 \setminus L_{\mathsf{2after}} \times \{m^*_B\}$, $\mathcal{O}_1 = \ddot{G} \times \ddot{h}$, $\mathcal{O}_2 = G \times h$, and $z = (L_1, L_2, pk_A, \mathsf{Enc}(pk_A, \cdot, m^*_B; G(\cdot, m^*_B)), h(pk_A, \cdot, m^*_B), \tilde{h}^1_q, h^3_q)$, we have

$$|\Pr[\mathcal{S}^{\ddot{G}\times\ddot{h}}(z) \Rightarrow 1] - \Pr[\mathcal{S}^{G\times h}(z) \Rightarrow 1]| \leq 2(q_G + q_h + 2q)\sqrt{\Pr[s \in S|s \leftarrow B^{\ddot{G}\times\ddot{h}}(z)]}.$$

To bound the probability $\Pr[s \in S|s \leftarrow B^{\ddot{G}\times\ddot{h}}(z)]$, we construct an adversary $\mathcal{D}$ against the $[\mathsf{OW\text{-}CPA}, \cdot]$-security.

- Select $k \leftarrow \{1, \cdots, q_G + q_h\}$.
- Given the pk_A from the $[\mathsf{OW\text{-}CPA}, \cdot]$ challenger, generate a list of static keys and include an additional pair $(pk_A, -)$ to get L_1, and generate a list of ephemeral keys, L_2.
- $\mathcal{D}$ randomly guesses which users are the initiator U_A and responder U_B in the test session, and which session is the test session.

– Pick $2q_G$ ($2q_{G_1}$, $2q_h$, $2q_{h_1}$, $2q_{h'}$, $2q_{h'_1}$, $2q_f$, $2q_{f_1}$,$2q_H$, $2q_{h_q^1}$, $2q_{h_q^2}$, $2q_{h_q^3}$, $2q_{h_q^4}$)-wise independent function uniformly to simulate the random oracle $\ddot{G}$ (G_1, $\ddot{h}$, h_1, h', h'_1, f, f_1, H, h_q^1, h_q^2, h_q^3, h_q^4).
– For any $\mathsf{SessionKeyReveal}$ query that does not involve U_A, the challenger uses static secret key to extract encapsulated and session key. For any $\mathsf{SessionKeyReveal}$ query that involves U_A, it answers using h_q^i, for $1 \leq i \leq 4$.
– On receiving pk_{2A}^* and C_A in test session, forward pk_{2A}^* to $[\mathsf{OW\text{-}CPA}, \cdot]$ game. On receiving challenge ciphertext C^* under pk_A and pk_{2A}^*, choose $K \leftarrow \mathcal{K}$ and return $SK^* = H(K, K_1 = h_1(pk_B, \mathsf{Dec}_1(sk_B, C_A)), C_A, C^*, U_A, U_B)$.
– Measure the argument of the k-th query to $\ddot{G} \times \ddot{h}$ and output m.

Since $\mathcal{D}$ simulates perfectly, $\Pr[s \in \mathcal{S} | s \leftarrow B^{\ddot{G} \times \ddot{h}}(z)] = \mathrm{Adv}_{\mathsf{2PKE}}^{[\mathsf{OW\text{-}CPA}, \cdot]}(\mathcal{D})$.

By summing the equations, $\mathsf{Adv}_7 - \mathsf{Adv}_8 \leq 4(q_G + q_h + 2q)\sqrt{\mathrm{Adv}_{\mathsf{2PKE}}^{[\mathsf{OW\text{-}CPA}, \cdot]}(\mathcal{D})}$. □

5 Prior AKEs and New Construction

In this section, we first give a new instantiation, CSIAKE, based on commutative supersingular isogeny. Then, we integrate several prior works to $\mathsf{AKE}_{\mathsf{QRO}}$, thus answering open problems on their securities in QROM.

5.1 CSIAKE from Commutative Supersingular Isogenies

Castryck et al. [15] proposed a commutative supersingular isogeny Diffie-Hellman (CSIDH) key exchange. Although the concrete choice of security parameters for CSIDH is heavily debated [9,33,4,11], CSIDH is considered as a candidate of quantum-resistant primitive. We propose 2-key and 1-key PKEs based on CSIDH in Fig.5. By integrating them to $\mathsf{AKE}_{\mathsf{QRO}}$, we get a CSIDH-based AKE in QROM, CSIAKE.

Let $p = 4 \times l_1 \cdots l_n - 1$ be a large prime, where each l_i is a small distinct odd prime. p and the supersingular elliptic curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$ with endomorphism ring $\mathcal{O} = \mathbb{Z}[\pi]$ are public parameters. The CSIDH key exchange works as following: Alice randomly chooses $(e_{A1}, \cdots, e_{An})$ from a range $[-m, m]$. These integers represent the ideal class $[\mathfrak{a}] = [\mathfrak{l}_1^{e_{A1}} \cdots \mathfrak{l}_n^{e_{An}}] \in \mathrm{cl}(\mathcal{O})$. Alice computes $[\mathfrak{a}]E_0$. Bob chooses his own secret $[\mathfrak{b}]$ and computes $[\mathfrak{b}]E_0$. They both could compute the common curve $[\mathfrak{a}][\mathfrak{b}]E_0 = [\mathfrak{b}][\mathfrak{a}]E_0$ in the form $y^2 = x^3 + sx^2 + x$. The share secret is the Montgomery coefficient of common curve, i.e., s.

$\mathsf{KGen1}$	$\mathsf{Enc}(pk_1, pk_2, m_1 \| \| m_2)$;	$\mathsf{Dec}(sk_1, sk_2, C)$
$(e_{11}, \cdots, e_{1n}) \leftarrow [-m, m]^n$ $sk_1 = [\mathfrak{a}_1] = [\mathfrak{l}_1^{e_{11}} \cdots \mathfrak{l}_n^{e_{1n}}]$ $pk_1 = [\mathfrak{a}_1]E_0$	$(f_1, \cdots, f_n) \leftarrow [-m, m]^n$ $c_1 = [\mathfrak{b}]E_0 = [\mathfrak{l}_1^{f_1} \cdots \mathfrak{l}_n^{f_n}]E_0$ $c_2 = h(\mathsf{Coef}([\mathfrak{b}]pk_1)) \oplus m_1$ $c_3 = h(\mathsf{Coef}([\mathfrak{b}]pk_2)) \oplus m_2$	$(c_1, c_2, c_3) \leftarrow C$ $m_1 = c_2 \oplus h(\mathsf{Coef}([\mathfrak{a}_1]c_1))$ $m_2 = c_3 \oplus h(\mathsf{Coef}([\mathfrak{a}_2]c_1))$
$\mathsf{KGen2}$	$\mathsf{Enc}_1(pk_1, m_1)$	$\mathsf{Dec}_1(sk_1, C)$
$(e_{21}, \cdots, e_{2n}) \leftarrow [-m, m]^n$ $sk_2 = [\mathfrak{a}_2] = [\mathfrak{l}_1^{e_{21}} \cdots \mathfrak{l}_n^{e_{2n}}]$ $pk_2 = [\mathfrak{a}_2]E_0$	$(f_1, \cdots, f_n) \leftarrow [-m, m]^n$ $c_1 = [\mathfrak{b}]E_0 = [\mathfrak{l}_1^{f_1} \cdots \mathfrak{l}_n^{f_n}]E_0$ $c_2 = h(\mathsf{Coef}([\mathfrak{b}]pk_1)) \oplus m_1$	$(c_1, c_2) \leftarrow C$ $m_1 = c_2 \oplus h(\mathsf{Coef}([\mathfrak{a}_1]c_1))$

Fig. 5. $\mathsf{2PKE} = (\mathsf{KGen1}, \mathsf{KGen2}, \mathsf{Enc}, \mathsf{Dec})$ and $\mathsf{PKE} = (\mathsf{KGen1}, \mathsf{Enc}_1, \mathsf{Dec}_1)$ based on CSIDH. $\mathsf{Coef}(\cdot)$ is Montgomery coefficient of the input curve. $h : \{0,1\}^* \to \{0,1\}^n$ is a random pairwise independent hash function.

Lemma 8. *Based on the Commutative Supersingular Decisional Diffie-Hellman (CSI-DDH) assumption, 2PKE in Fig. 5 is* [OW-CPA, OW-CPA] *secure and PKE is OW-CPA secure.*

The CSI-DDH problem is, given $(E_0, [\mathfrak{a}]E_0, [\mathfrak{b}]E_0, E')$, to decide $E' = [\mathfrak{a}][\mathfrak{b}]E_0$ or $E' = [\mathfrak{c}]E_0$ for some random secret $[\mathfrak{c}]$. Please refer to Appendix C for the definition of CSI-DDH and a formal proof of Lemma 8. We note that, similar to 2-key PKE in SIAKE (in next subsection), we can not reduce its IND-like security to a standard assumption, since c_3 contains information about randomness.

A recent work [11] points out that the parameters recommended in [15] do not achieve the concrete quantum security as claimed, and also suggest CSIDH-5280 to have the same security as AES-128. We utilize CSIDH-5280 to make a comparison. With CSIDH-5280 [11], the total communication of CSIAKE is 2028 bytes. On the contrary, when integrating CSIDH-5280 to HKSU-AKE [21], the QROM version of FSXY, the communication is 2688 bytes. Since there is no decryption error, the re-encryption only needs to check the correctness of the first cihpertext. Thus, the initiator and responder in HKSU need 6 and 6 isogeny computation, while those in CSIAKE need 6 and 5, respectively.

5.2 Previous Instantiations of X3LH-AKE

5.2.1 SIAKE from Supersingular Isogenies. Inspired by X3LH-AKE, Xu et al. [38] proposed SIAKE, a two round supersingular isogeny based protocol. By augmenting SIAKE to $\mathsf{AKE}_{\mathsf{QRO}}$, we conclude that SIAKE is secure in QROM, which answers their open problem [38].

Here is a brief recall of some notations. Let $p = \ell_1^{e_1}\ell_2^{e_2} \cdot f \pm 1$ be a large prime. Assume E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$ with order $|E_0(\mathbb{F}_{p^2})| = (\ell_1^{e_1}\ell_2^{e_2} \cdot f)^2$. Let $E_0[m]$ be a group of m-torsion points for $m \in \{\ell_1^{e_1}, \ell_2^{e_2}\}$. Assume $E_0[\ell_1^{e_1}] =< P_1, Q_1 >$ and $E_0[\ell_2^{e_2}] =< P_2, Q_2 >$. Define $\{t, s\} = \{1, 2\}$, $\mathfrak{g} = (E_0; P_1, Q_1, P_2, Q_2)$ and $\mathfrak{e} = (\ell_1, \ell_2, e_1, e_2)$. Let $\mathfrak{a} \in \mathbb{Z}_{\ell_1^{e_1}}$ and $\mathfrak{b} \in \mathbb{Z}_{\ell_2^{e_2}}$. Then, with computations for $\mathfrak{g}^{\mathfrak{a}}$, $\mathfrak{g}^{\mathfrak{b}}$, $(\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}}$, $(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$, as defined in [18] and Appendix D, SIDH key exchange is: Alice computes $\mathfrak{g}^{\mathfrak{a}}$ with $\mathfrak{a}$, while Bob computes $\mathfrak{g}^{\mathfrak{b}}$ with $\mathfrak{b}$. The shared key is $j = (\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}} = (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$.

From the above notations, Fig. 6 shows the [OW-CPA, OW-CPA] secure 2-key PKE and OW-CPA secure PKE in the supersingular-isogeny setting. Similar with a comparison in [38], the initiator and responder in HKSU need 6 and 6 isogeny computation while those in SIAKE need 6 and 5, respectively.

$\mathsf{KGen1}$	$\mathsf{Enc}(pk_1, pk_2, m_1 \Vert m_2)$;	$\mathsf{Dec}(sk_1, sk_2, C)$
$\mathfrak{a}_1 \leftarrow \mathbb{Z}_{\ell_s^{e_s}}$	$\mathfrak{b} \leftarrow \mathbb{Z}_{\ell_t^{e_t}}$, $c_1 = \mathfrak{g}^{\mathfrak{b}}$	$(c_1, c_2, c_3) \leftarrow C$
$pk_1 = \mathfrak{g}^{\mathfrak{a}_1}$	$c_2 = h((\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}) \oplus m_1$	$m_1 = c_2 \oplus h\,(c_1^{\mathfrak{a}_1})$
$sk_1 = \mathfrak{a}_1$	$c_3 = h((\mathfrak{g}^{\mathfrak{a}_2})^{\mathfrak{b}}) \oplus m_2$	$m_2 = c_3 \oplus h\,(c_1^{\mathfrak{a}_2})$
$\mathsf{KGen2}$	$\mathsf{Enc}_1(pk_1, m_1)$	$\mathsf{Dec}_1(sk_1, C)$
$\mathfrak{a}_2 \leftarrow \mathbb{Z}_{\ell_s^{e_s}}$	$\mathfrak{b} \leftarrow \mathbb{Z}_{\ell_t^{e_t}}$, $c_1 = \mathfrak{g}^{\mathfrak{b}}$	$(c_1, c_2) \leftarrow C$
$pk_2 = \mathfrak{g}^{\mathfrak{a}_2}, sk_2 = \mathfrak{a}_2$	$c_2 = h((\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}) \oplus m_1$,	$m_1 = c_2 \oplus h\,(c_1^{\mathfrak{a}_1})$

Fig. 6. $\mathsf{2PKE} = (\mathsf{KGen1}, \mathsf{KGen2}, \mathsf{Enc}, \mathsf{Dec})$ and $\mathsf{PKE} = (\mathsf{KGen1}, \mathsf{Enc}_1, \mathsf{Dec}_1)$. $h : \{0,1\}^* \to \{0,1\}^n$ is a random pair-wise independent hash function.

5.2.2 FSXY-AKE As pointed out by [37], FSXY [17] can be regarded as a non-compact instantiation of 2-key PKE with the parallel execution of two PKEs in Fig. 7. The IND/OW-CPA security of PKE implies the [OW-CPA, OW-CPA] security of resulting 2-key PKE [37].

By integrating such a result to $\mathsf{AKE}_{\mathsf{QRO}}$, we re-answer the open problem on (modified) FSXY-AKE's security in QROM behind [21]. In our answer, the underlying security requirement of PKE is the same with FSXY, say OW-CPA, while [21] requires IND-CPA. Our reduction is relatively tighter for the factors of N and l where N is the number of users, l is the number of sessions between two users, in terms of decryption error (N vs. N^4l^2 in [21]), underlying scheme (N^2l vs. N^4l^2) and the entropy of public keys (N^2l vs. N^6l^3). In [21], they define "S" as the number of sessions the adversary could established, which is bounded by N^2l. We note that, when applying to FSXY, our framework has a drawback that it needs one more re-encryption than HKSU.

Since the parallel execution of two IND/OW-CPA secure PKEs is a [OW-CPA, OW-CPA] secure 2-key PKE, several NIST post-quantum proposals [31] in the third round, such as Kyber [6], SIKE [22] etc. could be applied modularly.

KGen1	KGen2	$\mathsf{Enc}(pk_1, pk_2, m_1 \Vert m_2)$	$\mathsf{Dec}(sk_2, sk_1, c_1 \Vert c_2)$
$(pk_1, sk_1) \leftarrow \mathsf{KGen}_1$;	$(pk_2, sk_2) \leftarrow \mathsf{KGen}_1$	$c_1 = \mathsf{Enc}_1(pk_1, m_1; r_1)$	$m_1 = \mathsf{Dec}_1(sk_1, c_1)$
		$c_2 = \mathsf{Enc}_1(pk_2, m_2; r_2)$	$m_2 = \mathsf{Dec}_1(sk_2, c_2)$

Fig. 7. 2-key PKE from 1-key scheme. Let $\mathsf{PKE} := (\mathsf{KGen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ be a 1-key PKE

5.2.3 2Kyber-AKE Xue et al. [37] proposed a 2-key PKE based on Module-LWE, namely 2Kyber. By combining 2Kyber with Kyber [6], and updating them to $\mathsf{AKE}_{\mathsf{QRO}}$, we will get a compact Module-LWE-based AKE in QROM.

6 Conclusion

In this work, we propose a generic construction of two-pass AKE in the quantum random oracle model. Our work not only answers several open problems on the QROM security of prior works, but also introduces new construction.

References

1. Ambainis, A., Rosmanis, A., Unruh, D.: Quantum attacks on classical proof systems. In FOCS 2014, pp. 474-483.
2. Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semiclassical oracles. Cryptology ePrint Archive, Report 2018/904
3. Bernstein D. J.: Multi-user Schnorr security, revisited. Cryptology ePrint Archive, Report 2015/996
4. Bernstein, D. J. Re: [pqc-forum] https://groups.google.com/a/list.nist.gov/forum/#!original/pqc-forum/svm1kDy6c54/0gFOLitbAgAJ
5. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In ASIACRYPT 2011, pp. 41-69.
6. Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J. M., Schwabe, P., Stehlé D.: CRYSTALS - Kyber: a CCA-secure Module-lattice-based KEM. In 2018 S&P, pp. 353-367.
7. Bindel, N., Hamburg, M., Hülsing, A., Persichetti, E.: Tighter proofs of CCA security in the quantum random oracle model, In TCC 2019, pp. 61-90.
8. Bader, C., Hofheinz, D., Jager, T., Kiltz, E., Li, Y.: Tightly-secure authenticated key exchange. In TCC 2015, pp. 629-658.
9. Bernstein, D. J., Lange, T., Martindale, C., Panny, L.: Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies. In EUROCRYPT 2019, pp. 409-441.
10. Bernstein, D. J., Hamburg M., Re: [pqc-forum] NIST pqc-forum mailing list, 2018, https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/SrFO_vK3xbI/m/utjUZ9hJDwAJ

11. Bonnetain, X., Schrottenloher, A.: Quantum Security Analysis of CSIDH. In EUROCRYPT 2020, pp. 493-522.
12. Cohn-Gordon, K., Cremers, C., Gjøsteen, K., Jacobsen, H., Jager, T.: Highly Efficient Key Exchange Protocols with Optimal Tightness. In, CRYPTO 2019, pp. 767-797
13. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In EUROCRYPT 2001, pp. 453-474.
14. Cremers, C.J.F.: Formally and Practically Relating the CK, CK-HMQV, and eCK Security Models for Authenticated Key Exchange. Cryptology ePrint Archive, Report 2009/253
15. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: An efficient post-quantum commutative group action. In ASIACRYPT 2018, pp. 395C427.
16. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly Secure Authenticated Key Exchange from Factoring Codes and Lattices. In PKC 2012, pp. 467-484.
17. Fujioka A., Suzuki K., Xagawa K., Yoneyama K.: Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In AsiaCCS 2013, pp. 83-94.
18. Fujioka, A., Takashima, K., Terada, S., Yoneyama, K.: Supersingular Isogeny Diffie-Hellman Authenticated Key Exchange. In ICISC 2018, pp. 177-195.
19. Gjøsteen, K., Jager, T.: Practical and tightly-secure digital signatures and authenticated key exchange. In CRYPTO 2018, pp. 95C125.
20. Hofheinz, D., Hövelmanns, K., and Kiltz, E.: A Modular Analysis of the Fujisaki-Okamoto Transformation. In TCC 2017, pp. 341-371.
21. Hövelmanns, K., Kiltz, E., Schäge, S. and Unruh, D.: Generic Authenticated Key Exchange in the Quantum Random Oracle Model. In PKC 2020, pp. 389C422.
22. Jao, D., Azarderakhsh, R., Campagna, M., et al.: SIKE candidate for NIST.
23. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: IND-CCA-Secure Key Encapsulation Mechanism in the Quantum Random Oracle Model, Revisited. In CRYPTO 2018, pp. 96-125. ePrint 2017/1096 (cited version 201907)
24. Jiang, H., Zhang, Z., Ma, Z.: Key Encapsulation Mechanism with Explicit Rejection in the Quantum Random Oracle Model, In PKC 2019, pp. 618-645.
25. Jiang, H., Zhang, Z., Ma, Z.: On the non-tightness of measurement-based reductions for key encapsulation mechanism in the quantum random oracle model. Cryptology ePrint Archive, Report 2019/494
26. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In CRYPTO 2005, pp. 546-566.
27. Kirkwood, D., Lackey, B.C., McVey, J., Motley, M., Solinas, J.A., Tuller, D.: Failure is not an option: Standardization issues for post-quantum key agreement, 2015.
28. Kuchta, V., Sakzad, A., Stehl, D, Steinfeld, R., Sun, S.: Measure-Rewind-Measure: Tighter Quantum Random Oracle Model Proofs for One-Way to Hiding and CCA Security. In EUROCRYPT 2020, pp. 703-728.
29. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In ProvSec 2007, pp. 1-16.
30. Longa, P.: A Note on Post-Quantum Authenticated Key Exchange from Supersingular Isogenies. Cryptology ePrint Archive, Report 2018/267.
31. NIST Post-Quantum Cryptography Standardization, https://csrc.nist.gov/Projects/Post-Quantum-Cryptography
32. Peikert, C.: Lattice Cryptography for the Internet. In PQCrypto 2014, pp. 197-219.
33. Peikert, C.: He gives C-sieves on the CSIDH, In EUROCRYPT 2020, pp. 463-492.
34. Saito, T., Xagawa, K., Yamakawa, T.: Tightly-Secure Key-Encapsulation Mechanism in the Quantum Random Oracle Model. In EUROCRYPT (3) 2018, pp. 520-551
35. Targhi, E.E., Unruh, D.: Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In TCC 2016-B, pp. 192-216
36. Unruh, D.: Revocable quantum timed-release encryption. Journal of the ACM, 62(6):No.49, 2015. A preliminary version appeared in EUROCRYPT 2014.
37. Xue, H., Li, B., Lu, X., Liang, B., He, J.: Understanding and Constructing AKE via Double-Key Key Encapsulation Mechanism. In ASIACRYPT 2018, pp. 158-189.
38. Xu, X., Xue, H., Wang, K., Au, M., Tian, S.: Strongly Secure Authenticated Key Exchange from Supersingular Isogenies. In ASIACRYPT 2019, pp. 278-308.
39. Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In CRYPTO 2012, pp. 758-775
40. Zhandry, M.: How to record quantum queries, and applications to quantum indifferentiability. In CRYPTO 2019, pp. 239-268.

Appendix A: Existing proofs for FO transform in QROM

Properties and requirements of existing transformations from probabilistic PKE to CCA secure KEM are summarized in Table 4.

In QROM (and also classical ROM), the main challenges include how to simulate the decapsulation oracle without secret key, and how to argue the randomness of encapsulated key in the challenge ciphertext.

To simulate the decapsulation oracle without secret key in QROM, the additional hash [1,20] has high overhead and reduction loss, while injective map and private RO have low overhead and tight reduction. Boneh et al. [5] firstly introduced the technique of modeling $h(m)$ with $h_q \circ f(m)$ to provide decapsulation oracle, where h_q is a private RO and f is an *injective* function. Inspired by [5], Saito et al. [34] and Jiang et al. [23] independently assumed $\mathsf{Enc}(pk, \cdot; G(\cdot))$ is injective and modeled $h(m)$ as $h_q \circ \mathsf{Enc}(pk, m; G(m))$. Thus, $K = h(m)$ encapsulated in $C = \mathsf{Enc}(pk, m; G(m))$ can be computed as $h_q(C)$. We call this technique as injective mapping with encryption under fixed public key and illustrate it in Fig. 2.

With this decapsulation oracle using injective mapping with encryption under fixed public key, several techniques are used to argue the randomness of encapsulated key in challenge ciphertext. Saito et al. [34] introduced the "puncture" technique which relies on the IND-CPA security. Jiang et al. [23] applied their extended OW2H lemma to $G \times h$ as an unified oracle, thereby reducing the security to a weaker primitive, OW-CPA secure PKE. Additional result include explicit rejection [24]. Recently, Hövelmanns et al. [21] extended the "puncture" analysis of [34] by considering decryption error.

Schemes	Inj. map.	prob. PKE	Additional Hash	Security Bound	DecError
[35,20]	-	IND/OW-CPA	len.-pre	$q^{3/2}\sqrt[4]{\varepsilon}$	✓
[34]	✓	IND-CPA	×	$q\sqrt{\varepsilon}$	×
[21]	✓	IND-CPA	×	$q\sqrt{\varepsilon}$	✓
[23]	✓	OW-CPA	×	$q\sqrt{\varepsilon}$	✓
[2]	✓	IND-CPA	×	$\sqrt{q\varepsilon}$	✓
[7]	✓	IND-CPA	×	$\sqrt{q\varepsilon}$	✓
[28]	✓	IND-CPA	×	$q^2\varepsilon$	✓

Table 4. Comparison of proof for FO type probabilistic PKE-to-KEM transform in the QROM. Inj. map. indicates the injective mapping with encryption under fixed public key. len.-pre means additional hash should be length preserving. q is the number of random oracle queries. ε is the advantage against OW/IND-CPA security of PKE.

Ambainis et al. [2] proposed an improved OW2H lemma, namely, the semi-classical OW2H, which implies the extended OW2H in [23] and gives better security bounds in several PKEs. Bindel, et al. [7] proposed a double-sided OW2H lemma and reduced q factor from reduction loss. Recently, Kuchta et al. [28] by pass the square-root advantage loss using an updated double-sided OW2H lemma with the rewinding technique.

After the propose of OW2H lemma, as mentioned above, several variants are proposed. We summarize them in Table 5. S is the set that two oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ differ. The 'Must know' column shows the oracles available to the one-wayness attacker. 1_S refers to the indicator function of S. The one-wayness attacker outputs an element in S with probability ϵ. The lemma shows an upper bound of the difference between $\mathcal{A}^{\mathcal{O}_1}$ and $\mathcal{A}^{\mathcal{O}_1}$ as function of ϵ or q, the number of queries to oracle.

OW2H Variants	$\|S\|$	Must know	Bound
Original [36,2]	Arbitrary	$\mathcal{O}_1$ or $\mathcal{O}_2$	$q\sqrt{\epsilon}$
Semi-classical [2]	Arbitrary	($\mathcal{O}_1$ or $\mathcal{O}_2$) and 1_S	$\sqrt{q\epsilon}$
Double-side [7]	1	$\mathcal{O}_1$ and $\mathcal{O}_2$	$\sqrt{\epsilon}$
Double-side-Revisited [28]	Arbitrary	$\mathcal{O}_1$ and $\mathcal{O}_2$	$q\epsilon$

Table 5. Comparison of OW2H lemmas.

Appendix B: Discussions on Security Models: CK, CK_{HMQV}, eCK and CK^+

To achieve a stronger security after the CK model [13] was introduced, CK_{HMQV} [26], eCK [29], and CK^+ [16] models are being proposed. Due to subtle but crucial differences between them, these models are incomparable. Cremers [14] formally analyzed the relation of CK [13], eCK [29], and CK_{HMQV} [26]. We give a brief discussion here. For formal details, please refer to [14] and [16].

Matching session: A session s could be defined with s_A (the owner of s), s_B (intended peer), s_R (the role performed by s_A), s_{send}/s_{recv} (the message send/received by s_A), and/or s_{sid} (the session identifier, only used in CK model). In CK model, two sessions s and s' match if $s_A = s'_B$, $s_B = s'_A$ and $s_{sid} = s'_{sid}$. In CK_{HMQV}, if $s_A = s'_B$, $s_B = s'_A$, $s_{send} = s'_{recv}$, and $s_{recv} = s'_{send}$. s and s' match. For eCK, s and s' are matching sessions if they match in CK_{HMQV} and $s_R \neq s'_R$. CK^+ claims to reformulate CK_{HMQV}, but uses the definition of matching session in eCK. The subtle differences are crucial since the role-symmetric AKE that is secure in one model may be insecure in other model [14].

Session State Reveal vs Ephemeral Key Reveal: The CK model [13] allows the session state reveal by adversary, but leaves the AKE protocol to specify the contents of the session state. Depending on the content of the session state reveal, a weaker AKE may be proved secure [29]. Thus, eCK replaces session state reveal with the ephemeral key reveal, which is equivalent to specify the content of session state as ephemeral secret key. However, if more session state is allowed to be revealed, the eCK secure scheme may be insecure [14].

How and when the ephemeral/static secret key related to test session is given to adversary. In CK model, the compromise of static secret key of the test session's owner is not allowed before the test session expire, thus not detecting key compromise impersonation (KCI) attack. To capture KCI, CK_{HMQV}, eCK and CK^+ allow the compromise of the static secret key of the owner before test session ends. CK_{HMQV}, eCK and CK^+ also consider the weak version of perfect forward security (wPFS), i.e., the corruption of executor or peer is allowed after the end of test session only if the matching session exists. The exposure of ephemeral/static secret key related to test session in eCK is modeled by adaptive queries to long-term key reveal/ephemeral key reveal. In CK^+ (and CK_{HMQV}), the exposure is not modeled by such queries. The ephemeral secret key is given to adversary directly after it is generated (thus they actually allow adaptive queries to ephemeral key reveal); the exposure of static secret key is modeled as giving to the adversary directly when it is allowed.

Our Choice: We use CK^+ model here with more strict definition. We require the session state reveal at least includes ephemeral secret key. The freshness still forbid the session state reveal on the test session and its matching session, while the exposure of static or ephemeral secret key related to test session is allowed to capture KCI, wPFS and maximal exposure attack (MEX). To capture KCI, static secret key of the owner of test session is given to the adversary directly after it is determined; for wPFS, the static secret keys of the owner and its peer in test session is given to adversary at the end of test session; for maximal exposure attack (MEX), the ephemeral secret key is given to adversary after it is determined.

Appendix C: CSI-DDH and the proof of Lemma 8

Definition 2 (CSI-DDH Problem[9]). *Let* $E_A = [\mathfrak{a}]E_0$, $E_B = [\mathfrak{b}]E_0$, $E_C = [\mathfrak{c}]E_0$ *be randomly chosen.* $b \leftarrow \{0,1\}$. *If* $b = 0$, $E' = E_C$, *otherwise* $E' = [\mathfrak{a}][\mathfrak{b}]E_0$ *Given* (E_0, E_A, E_B, E'), *the problem is to output* b' *as a guess of* b.

Let $\mathrm{Adv}_{\mathcal{B}}^{csiddh} = \Pr[b = b'] - 1/2$ be the advantage of solving CSI-DDH problem. The CSI-DDH assumption claims, for any PPT adversary $\mathcal{B}$, $\mathrm{Adv}_{\mathcal{B}}^{csiddh}$ is negligible.

We reduce the $[\mathsf{OW\text{-}CPA}, \cdot]$ security to CSI-DDH assumption. It is analogous for the $[\cdot, \mathsf{OW\text{-}CPA}]$ security and the $\mathsf{OW\text{-}CPA}$ security of 1-key PKE. In Game i ($i \geq 1$), we denote success guess as Succ_i

Game 0: This is the original $[\mathsf{OW\text{-}CPA}, \cdot]$ challenge game in Fig. 3.

Game 1: In this game we modify $[\mathsf{OW\text{-}CPA}, \cdot]$ challenge game by requiring that the adversary wins the game if $m_1' = m_1$. We denote this event as Succ_1. Note that in Game 0, the adversary wins only if both $m_1' = m_1$ and $m_2' = m_2$. Thus, we have $\Pr[\mathsf{Succ}_0] \leq \Pr[\mathsf{Succ}_1]$.

Game 2: In this game, we modify the computation of challenge ciphertext. Specifically, $[\mathfrak{b}]pk_1$ is replaced by a random $[\mathfrak{c}]E_0$. We construct an algorithm $\mathcal{B}$ to solve the CSI-DDH problem given an instance $(E_0, [\mathfrak{a}]E_0, [\mathfrak{a}]E_0, E')$, if there exists an algorithm $\mathcal{A}$ to distinguish Game 1 and Game 2.

$\mathcal{B}(E_0, E_A, E_B, E')$
01 $pk_1 \leftarrow E_A$
02 $pk_2^*, \mathrm{state} \leftarrow \mathcal{A}(pk_1)$
03 $m_1 \leftarrow \{0,1\}^n$
04 $c_1^* = E_B$, $c_2^* = h(\mathsf{Coef}(E')) \oplus m_1$, $c_3^* \leftarrow \{0,1\}^n$
05 $m_1' || m_2' \leftarrow \mathcal{A}(\mathrm{state}, (c_1^*, c_2^*, c_3^*))$
06 If $m_1' = m_1$, $b' = 1$, else $b' \leftarrow \{0,1\}$.

If (E_0, E_A, E_B, E') is an CSI-DDH tuple, $\mathcal{B}$ perfectly simulates Game 1, else $\mathcal{B}$ perfectly simulates Game 2. In the CSI-DDH challenge, we have

$$\begin{aligned}
\mathrm{Adv}_{\mathcal{B}}^{csiddh} &= \Pr[b = b'] - 1/2 \\
&= 1/2(\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]) \\
&= 1/2(\Pr[b' = 1 | \text{Game 1}] - \Pr[b' = 1 | \text{Game 2}]) \\
&= 1/2(\Pr[\mathsf{Succ}_1] - \Pr[\mathsf{Succ}_2]).
\end{aligned}$$

Game 3: In this game, we modify the computation of the challenge ciphertext. Specifically, $h(\mathsf{Coef}([\mathfrak{c}]E_0))$ is replaced by a random string h^*. Now c_2^* is a completely random string in $\{0,1\}^n$. Thus, the advantage to compute m_1 is $\Pr[\mathsf{Succ}_3] = 1/2^n$. Note that, since h is a pairwise independent hash function, by the leftover hash lemma, $|\Pr[\mathsf{Succ}_2] - \Pr[\mathsf{Succ}_3]|$ is negligible.

To sum them up, we have that $\Pr[\mathsf{Succ}_0] \leq 2\mathrm{Adv}_{\mathcal{B}}^{csiddh} + 1/2^n + negl$.

Appendix D: SIDH and Crypto-friendly Description.

We recall briefly the SIDH protocol using the same notation as [22]. Let p be a large prime with a form $p = \ell_1^{e_1}\ell_2^{e_2} \cdot f \pm 1$, where ℓ_1 and ℓ_2 are two small primes, and f is an integer cofactor. Then we can construct a supersingular elliptic curve E_0 defined over $\mathbb{F}_{p^2}$ with order $|E_0(\mathbb{F}_{p^2})| = (\ell_1^{e_1}\ell_2^{e_2} \cdot f)^2$. Let $\mathbb{Z}_m$ be the ring of residue classes modulo m.

[9] It is defined as a generalized form for n-way by using cryptographic invariant maps in: Dan Boneh, Darren Glass, Daniel Krashen, Kristin Lauter, Shahed Sharif, Alice Silverberg, Mehdi Tibouchi, Mark Zhandry: Multiparty Non-Interactive Key Exchange and More From Isogenies on Elliptic Curves. In MATHCRYPT 2018,

The subgroup $E_0[m]$ of m-torsion points is isomorphic to $\mathbb{Z}_m \times \mathbb{Z}_m$ for $m \in \{\ell_1^{e_1}, \ell_2^{e_2}\}$. Let P_1, Q_1 be two points that generate $E_0[\ell_1^{e_1}]$ and P_2, Q_2 be two points that generate $E_0[\ell_2^{e_2}]$. The public parameters are $(E_0; P_1, Q_1; P_2, Q_2; \ell_1, \ell_2, e_1, e_2)$.

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi_A} & E_A = E_0/\langle R_A \rangle \\ \downarrow{\scriptstyle \phi_B} & & \downarrow{\scriptstyle \phi_{AB}} \\ E_B = E_0/\langle R_B \rangle & \xrightarrow{\phi_{BA}} & E_{AB} = E_0/\langle R_A, R_B \rangle \end{array}$$

Fig. 8. SIDH

The SIDH, as depicted in Figure 8, works as follows. Alice chooses her secret key k_a from $\mathbb{Z}_{\ell_1^{e_1}}$ and computes the isogeny $\phi_A : E_0 \to E_A$ whose kernel is the subgroup $\langle R_A \rangle = \langle P_1 + [k_a]Q_1 \rangle$. She then sends to Bob her public key which is E_A together with the two points $\phi_A(P_2), \phi_A(Q_2)$. Similarly, Bob chooses his secret key k_b from $\mathbb{Z}_{\ell_2^{e_2}}$ and computes the isogeny $\phi_B : E_0 \to E_B$ with kernel subgroup $\langle R_B \rangle = \langle P_2 + [k_b]Q_2 \rangle$. He sends to Alice his public key which is E_B together with the two points $\phi_B(P_1), \phi_B(Q_1)$. To get the shared secret, Alice computes the isogeny $\phi_{BA} : E_B \to E_{BA}$ with kernel subgroup generated by $\phi_B(P_1) + [k_a]\phi_B(Q_1)$. Similarly, Bob computes the isogeny $\phi_{AB} : E_A \to E_{AB}$ with kernel subgroup generated by $\phi_A(P_2) + [k_b]\phi_A(Q_2)$. Since the composed isogeny $\phi_{AB} \circ \phi_A$ has the same kernel $\langle R_A, R_B \rangle$ as $\phi_{BA} \circ \phi_B$, Alice and Bob can share the same j-invariant $j(E_{AB}) = j(E_{BA})$.

It will be helpful to have a crypto-friendly description of SIDH for the presentation of our AKEs. We follow the treatment of Fujioka et al. [18]. In what follows we assume $\{t, s\} = \{1, 2\}$, and denote the public parameters by $\mathfrak{g} = (E_0; P_1, Q_1, P_2, Q_2)$ and $\mathfrak{e} = (\ell_1, \ell_2, e_1, e_2)$. We define the sets of supersingular curves and those with an auxiliary basis as

$$\begin{aligned} \mathrm{SSEC}_p &= \{\text{supersingular elliptic curves } E \text{ over } \mathbb{F}_{p^2} \text{ with } E(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}_{\ell_1^{e_1}\ell_2^{e_2} f})^2\}; \\ \mathrm{SSEC}_A &= \{(E; P'_t, Q'_t) | E \in \mathrm{SSEC}_p, (P'_t, Q'_t) \text{ is basis of } E[\ell_t^{e_t}]\}; \\ \mathrm{SSEC}_B &= \{(E; P'_s, Q'_s) | E \in \mathrm{SSEC}_p, (P'_s, Q'_s) \text{ is basis of } E[\ell_s^{e_s}]\}. \end{aligned}$$

Let $\mathfrak{a} = k_a$ and $\mathfrak{b} = k_b$, then we define,

$$\begin{aligned} \mathfrak{g}^{\mathfrak{a}} &= (E_A; \phi_A(P_t), \phi_A(Q_t)) \in \mathrm{SSEC}_A, \\ &\qquad \text{where } R_A = P_s + [k_a]Q_s, \phi_A : E_0 \to E_A = E_0/\langle R_A \rangle; \\ \mathfrak{g}^{\mathfrak{b}} &= (E_B; \phi_B(P_s), \phi_B(Q_s)) \in \mathrm{SSEC}_B, \\ &\qquad \text{where } R_B = P_t + [k_b]Q_t, \phi_B : E_0 \to E_B = E_0/\langle R_B \rangle; \\ (\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}} &= j(E_{BA}), \text{ where } R_{BA} = \phi_B(P_s) + [k_a]\phi_B(Q_s), \phi_{BA} : E_B \to E_{BA} = E_B/\langle R_{BA} \rangle; \\ (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}} &= j(E_{AB}), \text{ where } R_{AB} = \phi_A(P_t) + [k_b]\phi_A(Q_t), \phi_{AB} : E_A \to E_{AB} = E_A/\langle R_{AB} \rangle. \end{aligned}$$

Using this notation, the SIDH looks almost exactly like the classical Diffie-Hellman. That is, the public parameters are $\mathfrak{g}$ and $\mathfrak{e}$. Alice chooses a secret key $\mathfrak{a}$ and sends $\mathfrak{g}^{\mathfrak{a}}$ to Bob, while Bob chooses a secret key $\mathfrak{b}$ and sends $\mathfrak{g}^{\mathfrak{b}}$ to Alice. The shared key is, as we expect, $j = (\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}} = (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$.

The SI-DDH problem is, given $\mathfrak{e}$ and $(\mathfrak{g}, \mathfrak{g}^{\mathfrak{b}}, j')$, to decide j is a random j-invariant or $(\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}}$. The SI-DDH assumption claims that for any PPT adversary the SI-DDH prblem is still hard.