Practical and Post-Quantum Authenticated Key Exchange from One-Way Secure Key Encapsulation Mechanism

Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, Kazuki Yoneyama
NTT Secure Platform Laboratories
3-9-11, Midori-cho Musashino-shi
Tokyo 180-8585 Japan

ABSTRACT

This paper discusses how to realize practical *post-quantum* authenticated key exchange (AKE) with strong security, i.e., CK$^+$ security (Krawczyk, CRYPTO 2005). It is known that strongly secure post-quantum AKE protocols exist on a generic construction from IND-CCA secure key encapsulation mechanisms (KEMs) in the standard model. However, when it is instantiated with existing IND-CCA secure post-quantum KEMs, resultant AKE protocols are far from practical in communication complexity. We propose a generic construction of AKE protocols from OW-CCA secure KEMs and prove CK$^+$ security of the protocols in the random oracle model. We exploit the random oracle and instantiate AKE protocols from various assumptions; DDH, gap DH, CDH, factoring, RSA, DCR, (ring-)LWE, McEliece one-way, NTRU one-way, subset sum, multi-variate quadratic systems, and more. For example, communication costs of our lattice-based scheme is approximately 14 times lower than the previous instantiation (for 128-bit security). Also, in the case of code-based scheme, it is approximately 25 times lower.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; E.3 [**Data**]: Data Encryption—*Public key cryptosystems*

General Terms

Security

Keywords

authenticated key exchange, CK$^+$ model, key encapsulation mechanism, random oracle model, post-quantum cryptography, ring-LWE assumption.

ASIA CCS'13, May 8–10, 2013, Hangzhou, China.
Copyright 2013 ACM 978-1-4503-1767-2/13/05 ...$15.00.

1. INTRODUCTION

1.1 Background

Following the celebrated Diffie-Hellman key exchange [17], researchers proposed several cryptographic schemes based on various problems. The problems are classified into two classes. The one is number-theoretical problems such as factoring [50], RSA [53], and the decisional Diffie-Hellman (DDH) problem [21]. The other is combinatorial problems such as subset-sum problem [42] and code problem [41]. After Shor [55] proposed *quantum* polynomial-time algorithm for factorization and the discrete logarithm problem, cryptographic researchers have proposed a great number of cryptographic primitives based on combinatorial problems, since combinatorial problems seem hard even for quantum algorithms. As examples of *post-quantum cryptography*, we have *public-key encryption* (PKE) and *key-encapsulation mechanism* (KEM) with provable security based on subset-sum problem [39], lattice problems [3, 27, 52], code problem [41], and problems on multi-variate quadratic (MQ) systems [30].

However, we only know a few results on *authenticated key exchange* (AKE) based on the combinatorial problems [10, 23], which is often employed to construct a secure channel between two parties in a public channel for secrecy and authenticity. When the parties share secret information (called *a session key*), they can construct a secure channel by symmetric-key encryptions and message authentication codes with the shared session key. AKE provides a solution to share a session key even if parts of secret information are exposed to an adversary. Formally speaking, in an AKE protocol, each party has public information, called a *static public key*, which is authorized by a trusted third party, e.g., certification authority (CA), and the corresponding secret information, called a *static secret key*. A user who wants to share information with an entity exchanges *ephemeral public keys*, generated from the corresponding *ephemeral secret keys*, and computes some *session state* from their static public keys, the corresponding static secret keys, the exchanged ephemeral public keys, and the corresponding ephemeral secret keys. Both parties then derive a *session key* from these values including session state with a function called the *key derivation function*. The desirable security notion of AKE is formulated as CK$^+$ security [34, 23]. Here, CK$^+$ security guarantees the Canetti-Krawczyk (CK) security [12], weak perfect forward secrecy (wPFS), resilience to key compromise impersonation (KCI), and resilience to maximal exposure attacks (MEX) which may exploit exposed secret information of the target session.

Recently, generic constructions of AKE from KEM have been investigated [10, 23]. Boyd et al. [10] and Fujioka et al. [23] gave constructions of CK and CK$^+$ secure AKE protocols from IND-CCA secure KEM schemes with pseudo-random functions (PRFs), respectively. Plugging the existing IND-CCA secure KEM schemes based on lattice problems and code problems in the standard model (StdM) into the generic constructions, we obtain AKE protocols CK and CK$^+$ secure against quantum adversaries.

Unfortunately, KEM schemes secure in the StdM are often inefficient on time and memory. In the case of the KEM schemes based number-theoretic assumptions, we have less problem on efficiency. However, when we consider post-quantum cryptography, we have unsatisfactory inefficiency. For example, on the IND-CCA secure KEM scheme based on the ring-LWE problem [40], our rough estimation says that the sizes of encapsulation key and ciphertext are about 540 kbits. Consequently, the AKE protocol from the KEM scheme also are inefficient on the communication costs, 1.12 Mbits (see 7-th rows in Table 1).

1.2 Our Contribution

In this paper, we investigate how efficiency of AKE is improved by relaxing the security model, i.e., adapting the random oracle model (ROM) [5]. We answer the following natural questions:

1. Which level of security in KEM is required to construct CK$^+$ secure AKE?
2. Which (post-quantum) assumptions can be used to construct CK$^+$ secure AKE?
3. How is efficiency of AKE improved?

As the answer of Question 1, we show that CK$^+$ secure AKE protocols can be constructed from OW-CCA secure KEM schemes with hash functions (regarded as the random oracles (RO)).

Regarding to Question 2, we show the followings:

- Rabin-KEM [16] and RSAP-H [15] can be used since they are OW-CCA secure. Rabin-KEM and RSAP-H are secure under the factoring and RSA assumptions in the ROM, respectively.
- Regarding to lattice problems, which seem to be hard even for quantum machines, we have efficient KEM schemes [40, 56]. They are secure under the ring-learning with errors (LWE) assumption, however, they achieve only IND-CPA security. Thus, we apply the Fujisaki-Okamoto (FO) conversion [24] to enhance their security to IND-CCA one, and then, construct CK$^+$ secure AKE protocols using the converted schemes. It may be worth noting that although applying the FO conversion requires the ROM in their security proofs, it is no problem since the security proof of the proposed construction needs the ROM, also.
- For a code-based KEM scheme, we can use an IND-CCA secure KEM scheme under the McEliece one-way assumption [33] as it is OW-CCA secure in the ROM, also.
- NTRU encryption scheme [27, 31] is also IND-CCA secure under the NTRU one-way assumption in the ROM. We can construct an efficient AKE from it.
- Similarly, we have a KEM scheme IND-CCA secure in the ROM under the subset-sum assumption [39]. Thus, the proposed construction generates an AKE protocol CK$^+$ secure under the subset-sum assumption with help of the FO conversion.
- Under the new assumption on multi-variate quadratic systems [30], we have an IND-CCA secure KEM scheme in the ROM. Thus, we also obtain a CK$^+$ secure AKE protocol from the assumption in the ROM.

As the answer of Question 3, we show that adapting the ROM in the security proofs, when we set the security parameter as $\kappa = 128$, each protocol becomes efficient on the communication costs as approximately half under the factoring assumption (from 22.74 kbits to 13.00 kbits), approximately $1/14$ under the ring-LWE assumption (from 1.12 Mbits to 80.65 kbits), and approximately $1/25$ under the code assumption (from 1.31 Mbits to 52.32 kbits). Thus, adapting ROM brings much benefit in efficiency. The details are in Section 5.4 and Section 6.

Our contributions are summarized as follows:

- We propose a generic CK$^+$ secure AKE construction from a OW-CCA secure KEM in the ROM.
- We achieve efficient CK$^+$ secure AKE protocols based on the hardness of integer factoring, the ring-LWE problem, and the code problem in the ROM.
- We achieve new CK$^+$ secure AKE protocols based on the hardness of the NTRU problem, subset-sum problem, and MQ problem.
- Especially, we have *quasi-linear* time AKE protocols based on the hardness of the ring-LWE problem in the ROM.

For the summary, see Table 1.

Table 1: Comparison between the previous schemes and instantiations of our scheme.

			Comm.
	Assumption	StdM?	Costs (bits)
[23]	DDH	StdM	2,048
[23]	Factoring	StdM	22,736
HMQV [34]	GapDH+KEA	ROM	512
Ours	GapDH/CDH	ROM	2,048
Ours	RSA	ROM	16,240
Ours	Factoring	ROM	12,992
[23]	Ring-LWE	StdM	≈ 1,117,000
[23]	Codes	StdM	≥ 1,312,672
Ours	Ring-LWE	ROM	≈ 80,600
Ours	Codes	ROM	52,320
Ours	NTRU	ROM	19,756
Ours	Subset Sum	ROM	1,295,960
Ours	MQ [30]	ROM	≈ 11,440,000
Ours	MQ [30]	ROM	≈ 7,672,000

Open problem.

We left an open problem to show our construction is secure in "quantum accessible" random oracle model. In this paper, we successfully showed that our construction is secure in ROM if the underlying KEM is secure. In addition, our proof does not require rewinding. Hence, one would consider our construction also quantumly secure. However, what we show is that if underlying problem is hard then there are no quantum adversary violating CK$^+$ security with *classical access to the random oracles*. In the real world, the random oracle is instantiated by a hash function H and a quantum adversary can evaluate on a quantum superposition of input. A quantum-accessible random oracle model (QAROM) [9, 60] captures this ability. Therefore, it would be interesting to show our construction is secure in the QAROM.

2. CK+ SECURITY MODEL

In this section, we quote the CK$^+$ model that was introduced by [34, 23]. We show the model for two-pass protocols. It can be trivially extended to n-pass protocols for any $n > 2$.

U_i denotes a party indexed by i. Parties are modeled as probabilistic polynomial-time (PPT) interactive Turing machines w.r.t. security parameter $\kappa \in \mathbb{N}$. For party U_i, we denote static secret (public) key by s_i (S_i) and ephemeral secret (public) key by x_i (X_i), respectively. Party U_i generates its own keys, s_i and S_i, and the static public key S_i is linked with U_i's identity in some systems like PKI.[1]

Session.

We call an invocation of a protocol *session*. A session is activated by an incoming message of the forms $(\Pi, \mathcal{I}, U_A, U_B)$ or $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, where Π is a protocol identifier, $\mathcal{I}$ and $\mathcal{R}$ are role identifiers, and U_A and U_B are user identifiers. If U_A is activated with $(\Pi, \mathcal{I}, U_A, U_B)$, then U_A is called the session *initiator*. If U_B is activated with $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, then U_B is called the session *responder*. The initiator U_A outputs X_A on activation, 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 computes the session key SK if U_A received the message. On the contrary, the responder U_B outputs X_B, and computes the session key SK.

If U_A is the initiator of a session, the session is identified by $\mathsf{sid} = (\Pi, \mathcal{I}, U_A, U_B, X_A)$ or $\mathsf{sid} = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$. If U_B is the responder of a session, the session is identified by $\mathsf{sid} = (\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$. We say that U_A is the *owner* of session sid, if the third coordinate of session sid is U_A. We say that U_A is the *peer* of session sid, if the fourth coordinate of session sid is U_A. We say that a session is *completed* if its owner computes the session key. The *matching session* of $(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ is session $(\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$ and vice versa.

Adversary.

The adversary $\mathcal{A}$, which is modeled as a PPT interactive Turing machine, controls all communications between parties including session activation by performing the following adversary query.

- Send(message): The message has one of the following forms: $(\Pi, \mathcal{I}, U_A, U_B)$, $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, or $(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$. The adversary $\mathcal{A}$ obtains the response from the party.

To capture exposure of secret information, the adversary $\mathcal{A}$ is allowed to issue the following queries.

- $\mathsf{SessionKeyReveal}(\mathsf{sid})$: The adversary $\mathcal{A}$ obtains the session key SK for the session sid if the session is completed.

- $\mathsf{SessionStateReveal}(\mathsf{sid})$: The adversary $\mathcal{A}$ obtains the session state of the owner of session sid if the session is not completed (the session key is not established yet). The session state includes all ephemeral secret keys and intermediate computation results except for immediately erased information but does not include the static secret key.

- $\mathsf{Corrupt}(U_i)$: This query allows the adversary $\mathcal{A}$ to obtain all information of the party U_i. If a party is corrupted by a $\mathsf{Corrupt}(U_i)$ query issued by the adversary $\mathcal{A}$, then we call the party U_i *dishonest*. If not, we call the party *honest*.

Freshness.

For the security definition, we need the notion of freshness.

Definition 1. Let $\mathsf{sid}^* = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ or $(\Pi, \mathcal{R}, U_A, U_B, X_B, X_A)$ be a completed session between honest users U_A and U_B. If the matching session exists, then let $\overline{\mathsf{sid}}^*$ be the matching session of sid^*. We say session sid^* is *fresh* if none of the following conditions hold:

1. $\overline{\mathsf{sid}}^*$ exists and the adversary $\mathcal{A}$ makes either of the following queries;
   - $\mathsf{SessionKeyReveal}(\mathsf{sid}^*)$ or
   - $\mathsf{SessionKeyReveal}(\overline{\mathsf{sid}}^*)$:
2. $\overline{\mathsf{sid}}^*$ exists and the adversary $\mathcal{A}$ makes either of the following queries;
   - $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$ or
   - $\mathsf{SessionStateReveal}(\overline{\mathsf{sid}}^*)$:
3. $\overline{\mathsf{sid}}^*$ does not exist and the adversary $\mathcal{A}$ makes the following query; $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$.

Security experiment.

For the security definition, we consider the following security experiment. Initially, the adversary $\mathcal{A}$ is given a set of honest users and makes any sequence of the queries described above. During the experiment, the adversary $\mathcal{A}$ makes the following query.

- $\mathsf{Test}(\mathsf{sid}^*)$: sid^* must be a fresh session. Select random bit $b \in_U \{0, 1\}$, and return the session key held by sid^* if $b = 0$, and return a random key if $b = 1$.

The experiment continues until the adversary $\mathcal{A}$ makes a guess b'. The adversary $\mathcal{A}$ *wins* the game if the test session sid^* is still fresh and if the guess of the adversary $\mathcal{A}$ is correct, i.e., $b' = b$. The advantage of the adversary $\mathcal{A}$ in the AKE experiment with the PKI-based AKE protocol Π is defined as

$$\mathsf{Adv}^{\mathsf{ake\text{-}ck^+}}_{\Pi,\mathcal{A}}(\kappa) = \Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}.$$

We define the security as follows.

Definition 2. We say that a PKI-based AKE protocol Π is *secure in the* CK$^+$ *model* if the following conditions hold:

1. (Completeness:) if two honest parties complete matching sessions, then, except with negligible probability, they both compute the same session key.
2. (Soundness:) for any PPT bounded adversary $\mathcal{A}$, its advantage $\mathsf{Adv}^{\mathsf{ake\text{-}ck^+}}_{\Pi,\mathcal{A}}(\mathcal{A})$ is negligible in security parameter κ for the test session sid^*,
   (a) if $\overline{\mathsf{sid}}^*$ does not exist, and the static secret key of the owner of sid^* is given to $\mathcal{A}$.
   (b) if $\overline{\mathsf{sid}}^*$ does not exist, and the ephemeral secret key of sid^* is given to $\mathcal{A}$.
   (c) if $\overline{\mathsf{sid}}^*$ exists, and the static secret key of the owner of sid^* and the ephemeral secret key of $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$.
   (d) if $\overline{\mathsf{sid}}^*$ exists, and the ephemeral secret key of sid^* and the ephemeral secret key of $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$.

[1] Static public keys must be known to both parties in advance. They can be obtained by exchanging them before starting the protocol or by receiving them from a certificate authority. This situation is common for *all* PKI-based AKE schemes.

(e) if $\overline{\mathsf{sid}}^*$ exists, and the static secret key of the owner of sid^* and the static secret key of the peer of sid^* are given to $\mathcal{A}$.

(f) if $\overline{\mathsf{sid}}^*$ exists, and the ephemeral secret key of sid^* and the static secret key of the peer of sid^* are given to $\mathcal{A}$.

Note that the item 2.a corresponds to resistance to KCI, item 2.e corresponds to wPFS, and items 2.b, 2.c, 2.d and 2.f correspond to resistance to MEX.

3. GENERIC AKE CONSTRUCTION FROM KEM

In this section, we propose a generic construction (GCwR) of CK$^+$ secure AKE from one-way KEMs in the random oracle model. Before describing our GCwR, we recall the notion of KEM.

3.1 Security Notions of KEM Schemes

Let us recall the model and security notions of KEM.

Definition 3. A KEM scheme KEM consists of the following 3-tuple (KeyGen, EnCap, DeCap):

$(ek, dk) \leftarrow \mathsf{KeyGen}(1^\kappa; r_g)$: a key-generation algorithm which on inputs 1^κ and $r_g \in \mathcal{RS}_G$, where κ is the security parameter and $\mathcal{RS}_G$ is a randomness space, outputs a pair of an encapsulation key and a decapsulation key (ek, dk).

$(K, C) \leftarrow \mathsf{EnCap}_{ek}(r_e)$: an encapsulation algorithm which takes as inputs encapsulation key ek and $r_e \in \mathcal{RS}_E$, outputs session key $K \in \mathcal{KS}$ and ciphertext $C \in \mathcal{CS}$, where $\mathcal{RS}_E$ is a randomness space, $\mathcal{KS}$ is a session key space, and $\mathcal{CS}$ is a ciphertext space.

$K/\perp \leftarrow \mathsf{DeCap}_{dk}(C)$: a decapsulation algorithm which takes as inputs decapsulation key dk and ciphertext $C \in \mathcal{CS}$, outputs session key $K \in \mathcal{KS}$ or a rejection symbol $\perp$.

We next recall the definitions of one-wayness under chosen-ciphertext and chosen-plaintext attacks (denoted by OW-CCA and OW-CPA) for key encapsulation, respectively.

Definition 4. A KEM scheme KEM is (t, ϵ)-OW-CCA secure for KEM if the following property holds for security parameter κ; For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with a time complexity at most t,

$$\mathsf{Adv}^{\mathsf{ow\text{-}cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa) = \Pr\left[\begin{array}{l} (ek, dk) \leftarrow \mathsf{KeyGen}(1^\kappa; r_g); \\ state \leftarrow \mathcal{A}_1^{\mathcal{DO}_\perp(dk,\cdot)}(ek); \\ (K^*, C^*) \leftarrow \mathsf{EnCap}_{ek}(r); \\ K'^* \leftarrow \mathcal{A}_2^{\mathcal{DO}_{C^*}(dk,\cdot)}(ek, C^*, state); \\ K'^* = K^* \end{array}\right] \leq \epsilon$$

where $\mathcal{DO}_a(dk, C)$ is the decapsulation oracle that returns $K = \mathsf{DeCap}_{dk}(C)$ if $C \neq a$ and returns a reject symbol otherwise, $state$ is state information which $\mathcal{A}$ wants to preserve from $\mathcal{A}_1$ to $\mathcal{A}_2$, and $\mathcal{A}$ runs in at most t steps.

We say a KEM scheme is OW-CPA secure for key encapsulation if $\mathcal{A}$ does not access $\mathcal{DO}$.

3.2 Construction from One-way KEM

We propose a generic construction (GCwR), which is based on a OW-CCA secure KEM and the ROM. (GCwR stands for generic construction with random oracles.) At the price of using ROs, we can weaken the assumption on the KEM from IND-CCA to OW-CCA. Since a OW-CCA secure KEM is efficiently obtained in the ROM (e.g., RSAP-H, RSA-KEM, and ECIES-KEM) and we do not need PRFs and a strong randomness extractor, we can efficiently instantiate GCwR. The details of instantiations are given in Section 5 and Section 6. To achieve ephemeral secret key exposure resistance, the ordinary NAXOS technique [36] is known and used in some protocols [36, 43, 23]. The NAXOS technique means that the DH-exponent is set as an output of a RO on inputs static and ephemeral secret keys. Since adversaries are not allowed to expose both static and ephemeral secret keys, the DH-exponent is not exposed if one of secret keys are exposed. In GCwR, we also apply the NAXOS technique to generating randomness of EnCap instead of generating the DH-exponent.

Generic construction GCwR.

Let KEM = (KeyGen, EnCap, DeCap) be a OW-CCA secure KEM and wKEM = (wKeyGen, wEnCap, wDeCap) be a OW-CPA secure KEM.

The protocol of GCwR is as follows.

Public Parameters: Let κ be the security parameter and let $H_1 : \{0,1\}^* \rightarrow \mathcal{RS}_E$ and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^\kappa$ be hash functions modeled as ROs. These are provided as a part of the public parameters.

Secret and Public Keys: User U_I randomly selects $r_I \in \mathcal{RS}_G$, and runs the generation algorithm $(dk_{I,1}, ek_{I,1}) \leftarrow \mathsf{KeyGen}(r_I)$. User U_I's static secret and public keys are $(dk_{I,1}, ek_{I,1})$.

Key Exchange: User U_A with secret and public keys $(dk_{A,1}, ek_{A,1})$, who is the initiator, and user U_B with secret and public keys $(dk_{B,1}, ek_{B,1})$, who is the responder, perform the following two-pass key exchange protocol.

1. To initialize the protocol, user U_A selects ephemeral secret keys $r_{A,1} \in \{0,1\}^\kappa$ and $r_{A,2} \in \mathcal{RS}_G$ randomly. User U_A computes $(C_{A,1}, K_{A,1}) \leftarrow \mathsf{EnCap}_{ek_{B,1}}(H_1(r_{A,1}, dk_{A,1}))$ and $(dk_{A,2}, ek_{A,2}) \leftarrow \mathsf{wKeyGen}(r_{A,2})$, and sends $(U_A, U_B, C_{A,1}, ek_{A,2})$ to user U_B.

2. Upon receiving $(U_A, U_B, C_{A,1}, ek_{A,2})$, user U_B chooses ephemeral secret keys $r_{B,1} \in \{0,1\}^\kappa$ and $r_{B,2} \in \mathcal{RS}_E$ uniformly at random, computes $(C_{B,1}, K_{B,1}) \leftarrow \mathsf{EnCap}_{ek_{A,1}}(H_1(r_{B,1}, dk_{B,1}))$ and $(C_{B,2}, K_{B,2}) \leftarrow \mathsf{wEnCap}_{ek_{A,2}}(r_{B,2})$, and sends $(U_A, U_B, C_{B,1}, C_{B,2})$ to user U_A.

   User U_B computes $K_{A,1} \leftarrow \mathsf{DeCap}_{dk_{B,1}}(C_{A,1})$, sets the session identity $\mathsf{sid} = (U_A, U_B, ek_{A,1}, ek_{B,1}, C_{A,1}, ek_{A,2}, C_{B,1}, C_{B,2})$ and the session key $SK = H_2(K_{A,1}, K_{B,1}, K_{B,2}, \mathsf{sid})$, and completes the session.

3. Upon receiving $(U_A, U_B, C_{B,1}, C_{B,2})$, user U_A computes $K_{B,1} \leftarrow \mathsf{DeCap}_{dk_{A,1}}(C_{B,1})$, $K_{B,2} \leftarrow \mathsf{wDeCap}_{dk_{A,2}}(C_{B,2})$, sets the session identity $\mathsf{sid} = (U_A, U_B, ek_{A,1}, ek_{B,1}, C_{A,1}, ek_{A,2}, C_{B,1}, C_{B,2})$ and the session key $SK = H_2(K_{A,1}, K_{B,1}, K_{B,2}, \mathsf{sid})$, and completes the session.

The session state of a session owned by U_A contains ephemeral secret keys $(r_{A,1}, r_{A,2})$ and a KEM key K_{A_1}. Similarly, the session state of a session owned by U_B contains ephemeral secret keys $(r_{B,1}, r_{B,2})$ and KEM keys $(K_{B,1}, K_{B,2})$.

Security.

We show the following theorem.

Theorem 1. *If a KEM scheme* KEM = (KeyGen, EnCap, DeCap) *is* OW-CCA *secure, and a KEM scheme* wKEM = (wKeyGen, wEnCap, wDeCap) *is* OW-CPA *secure, then the AKE scheme with* GCwR *is* CK$^+$ *secure where H_1 and H_2 are modeled as ROs.*

The precise proof of Theorem 1 is in Appendix A.

Here, we give an overview of the security proof for the case that the test session has a matching session.

We have to consider the following four exposure patterns in the CK$^+$ security model:

2-(c) the static secret key of the initiator and the ephemeral secret key of the responder

2-(d) both ephemeral secret keys

2-(e) both static secret keys

2-(f) the ephemeral secret key of the initiator and the static secret key of the responder

Intuitively speaking, in case 2-(c), $K_{A,1}$ is protected by the security of $CT_{A,1}$ because $H_1(r_{A,1}, dk_{A,1})$ is hidden and $dk_{B,1}$ is not exposed. In case 2-(d), $K_{A,1}$ and $K_{B,1}$ are protected by the security of $C_{A,1}$ and $C_{B,1}$ because $H_1(r_{A,1}, dk_{A,1})$ and $H_1(r_{B,1}, dk_{B,1})$ are hidden, and $dk_{A,1}$ and $dk_{B,1}$ are not exposed. In case 2-(e), $K_{B,2}$ is protected by the security of $C_{B,2}$ because $dk_{A,2}$ and $r_{B,2}$ are not exposed. In case 2-(f), $K_{B,1}$ is protected by the security of $C_{B,1}$ because $H_1(r_{B,1}, dk_{B,1})$ is hidden and $dk_{A,1}$ is not exposed.

Then, we construct a reduction from the OW-CCA game (for cases 2-(c), 2-(d), and 2-(f)) or the OW-CPA game (for 2-(e)) to the CK$^+$ security game. The simulator embeds the challenge ciphertext in the OW-CCA (OW-CPA) game into the hash list of H_2 corresponding to the test session. If an adversary in the CK$^+$ security game succeeds with non-negligible probability, then the simulator in the OW-CCA (OW-CPA) game also succeeds with non-negligible probability. The simulation of SessionStateReveal queries is done by the power of the decryption oracle in the OW-CCA game. For case 2-(e), the simulator does not need the decryption oracle because he knows all static secret keys. We can show a similar proof in non-matching cases.

4. FUJISAKI–OKAMOTO CONVERSION, RECONSIDERED

In this section, we briefly review the Fujisaki–Okamoto (FO) conversion [24] and discuss that we can slightly weaken the conditions.

Let us review the model and security notions of public key encryption (PKE) scheme. The model for PKE schemes is summarized as follows:

Definition 5. A PKE scheme PKE consists of the following 3-tuple (Gen, Enc, Dec):

$(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa; r_g)$: a key-generation algorithm which on input 1^κ, where κ is the security parameter, outputs a pair of keys (pk, sk). pk and sk are called public key and secret key, respectively.

$C \leftarrow \mathsf{Enc}_{pk}(M; r_e)$: an encryption algorithm which takes as input public key pk and plaintext M, outputs ciphertext C.

$M/\bot \leftarrow \mathsf{Dec}_{sk}(C)$: a decryption algorithm which takes as input secret key sk and ciphertext C, outputs plaintext M or a rejection symbol $\bot$.

We say PKE has perfect correctness if for any (pk, sk) generated by Gen, $M \in \mathcal{MS}$, and $r_e \in \mathcal{RS}_E$, we have that $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(M; r_e)) = M$.

The security of PKE schemes is defined by several notions like one-wayness and indistinguishability. Here, we recall the definition of indistinguishability under chosen-ciphertext and chosen-plaintext attacks (denoted by IND-CCA and IND-CPA) for PKE, respectively.

Definition 6. A PKE scheme is (t, ϵ)-IND-CCA secure if the following property holds for security parameter κ; For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathsf{Adv}_{\mathcal{A}}^{\text{ind-cca}}(\kappa) = \left| \Pr\left[\begin{array}{l} b \leftarrow \{0,1\}; \\ (pk, sk) \leftarrow \mathsf{Gen}(1^\kappa); \\ (M_0, M_1, state) \leftarrow \mathcal{A}_1^{\mathcal{DO}_\bot(sk,\cdot)}(pk); \\ C^* \leftarrow \mathsf{Enc}_{pk}(M_b); \\ b' \leftarrow \mathcal{A}_2^{\mathcal{DO}_{C^*}(sk,\cdot)}(pk, C^*, state); \\ b' = b \end{array} \right] - \frac{1}{2} \right| \leq \epsilon,$$

where $\mathcal{DO}_a(sk, C)$ is the decryption oracle that returns a message $M = \mathsf{Dec}_{sk}(C) \in \mathcal{MS} \cup \{\bot\}$ if $C \neq a$ and returns a reject symbol otherwise, *state* is state information (possibly including pk, M_0 and M_1) which $\mathcal{A}$ wants to preserve, and $\mathcal{A}$ runs in at most t steps. We say a PKE scheme is IND-CPA secure, if $\mathcal{A}$ does not access $\mathcal{DO}$.

Review of the FO conversion.

The conversion transforms an IND-CPA secure PKE scheme, which has perfect correctness and γ-uniformity (defined later) into an IND-CCA secure PKE scheme in the ROM.

Let $\mathsf{wPKE} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wDec})$ be a PKE scheme with a message space $w\mathcal{MS}$, randomness spaces $w\mathcal{RS}_G$ and $w\mathcal{RS}_E$, and a ciphertext space $w\mathcal{CS}$. We define γ-uniformity, which essentially says that ciphertext has a min-entropy at least $-\lg \gamma$.

Definition 7. We say a PKE scheme $\mathsf{wPKE} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wDec})$ is *γ-uniform* if for any $(pk, sk) \leftarrow \mathsf{Gen}(1^n; r_g)$, $M \in w\mathcal{MS}$, and $C \in w\mathcal{CS}$, the inequality $\Pr_{r_e}\left[\mathsf{Enc}_{pk}(M; r_e) = C\right] \leq \gamma$ holds.

We decompose $w\mathcal{MS}$ into two finite sets $\mathcal{MS}$ and $\mathcal{RS}_E$, that is, $w\mathcal{MS} = \mathcal{MS} \times \mathcal{RS}_E$. Let $H : w\mathcal{MS} \to w\mathcal{RS}_E$ be a hash function modeled as the random oracle. The FO conversion converts wPKE into an encryption scheme $\mathsf{PKE} = \mathrm{FO}(\mathsf{wPKE}) = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with a message space $\mathcal{MS}$, a randomness space $\mathcal{RS}_E$, and a ciphertext space $\mathcal{CS} = w\mathcal{CS}$ defined as follows:

$\mathsf{Gen}(1^n; r_g)$**:** Output $(pk, sk) \leftarrow \mathsf{wGen}(1^n; r_g)$.

$\mathsf{Enc}_{pk}(M; r_e)$**:** Output $C \leftarrow \mathsf{wEnc}_{pk}((M, r_e); H(M, r_e))$.

$\mathsf{Dec}_{sk}(C)$**:** Compute $M' \leftarrow \mathsf{wDec}_{sk}(C)$. If $M' = \bot$ then output $\bot$. Otherwise parse $(M, r_e) \leftarrow M'$ and verify C by checking $C = \mathsf{wEnc}_{pk}(M'; H(M'))$. If the verification is not passed then output $\bot$. Otherwise output M.

They showed that PKE is IND-CCA secure if wPKE is IND-CPA secure, γ-uniform, and perfectly correct.

Relaxation.

We define semi-uniformity in order to relax the requirements on a PKE scheme. We only require that the min-entropy of the ciphertext is at least $-\lg \gamma$ for all but δ fraction of key pairs rather than any key pairs.

Definition 8. We say a PKE scheme $\mathsf{wPKE} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wDec})$ is *(δ, γ)-semi-uniform* if for all $M \in w\mathcal{MS}$ and $C \in w\mathcal{CS}$,

$$\Pr_{r_g}\left[\Pr_{r_e}[\mathsf{wEnc}_{pk}(M; r_e) = C] \leq \gamma : (pk, sk) \leftarrow \mathsf{wGen}(1^n; r_g)\right] \geq 1 - \delta.$$

It is obvious that $(0, \gamma)$-semi-uniformity implies γ-uniformity.

This relaxation does not harm the security of the converted scheme. Without modifications, we can adapt the proof of the security by introducing δ into the bound below.

Theorem 2 (Theorem 5.4 in [24], adapted). *Suppose that* wPKE *is a (δ, γ)-semi-uniform PKE scheme. Let* PKE = FO(wPKE) *be the converted scheme from* wPKE. *If* wPKE *is* $(t_{\mathsf{cpa}}, \epsilon_{\mathsf{cpa}})$-IND-CPA *secure then* PKE *is* $(t_{\mathsf{cca}}, \epsilon_{\mathsf{cca}})$-IND-CCA *secure, where*

$$t_{\mathsf{cca}} \leq t_{\mathsf{cpa}} - Q_H \cdot (T_{\mathsf{wEnc}} + O(\kappa)),$$

$$\epsilon_{\mathsf{cca}} \leq \frac{1}{(1-\gamma)^{Q_D}} \cdot \epsilon_{\mathsf{cpa}} + \frac{2Q_H}{|w\mathcal{RS}_E|} + \delta,$$

Q_D and Q_H denote the numbers of the queries to the decryption oracle and the random oracle H, and T_{wEnc} denotes the computational running time of wEnc.

5. INSTANTIATIONS FROM RING-LWE

We have new concrete AKE protocols based on the worst-case hardness of the (ring-)LWE problems derived from our generic construction, GCwR. We can employ IND-CCA secure PKE schemes in the standard model [47, 45, 13, 1, 2, 57, 40] as OW-CCA secure KEM schemes, as Fujioka et al. did [23]. Unfortunately, the obtained AKE protocols are inefficient since these PKE schemes require huge keys, say, the quadratic or cubic order of the security parameter.

In order to instantiate practical AKE protocols with CK$^+$ security, we construct efficient OW-CCA KEM schemes in the ROM from IND-CPA secure PKE scheme under the ring-LWE assumption. Formally speaking, we construct IND-CCA secure PKE schemes by applying the Fujisaki–Okamoto conversion [24] to IND-CPA secure PKE schemes [40, 56]. Since IND-CCA secure PKE schemes with sufficiently large plaintext space are OW-CCA secure, we have instantiated OW-CCA secure KEM schemes from them. An obstacle to applying the Fujisaki–Okamoto conversion is that the (ring-)LWE-based PKE schemes do not have perfect correctness but negligible decryption errors. To eliminate this obstacle, i.e., eliminate the decryption errors, we carefully choose parameters and slightly strengthen the assumptions. We should note that one can also eliminate errors by using the technique proposed by Dwork, Naor, and Reingold [20]. We do not adopt it since it uses pseudorandom generators.

Remark 1. We note that our CCA-secure schemes are based on the worst-case hardness of the ring-LWE problem and *not* based on the worst-case hardness of the approximating shortest vector problem. This is because we employ the Fujisaki-Okamoto conversion and the proof of the conversion in quantum setting is still open [9, 60].

5.1 Ring-LWE Assumption

We next review the ring-LWE assumption [40]. For simplicity, practical issues, and limitation of space, we adapt a simplified notion of the ring-LWE problem by Ducas and Durms [19] with parameter $n = 2^z$.

Hereafter, an element $c \in \mathbb{Z}_q$ is represented by a corresponding integer in $[-(q-1)/2, (q-1)/2]$. For an element $c \in \mathbb{Z}_q$, $|c|$ denotes the Lee value, which is defined by $|c| = c$ if $0 \leq c < q/2$ and $-c$ if $-q/2 < c < 0$. For a real $x \in \mathbb{R}$, $\lfloor x \rceil$ denotes the nearest integer, that is, $\lceil x - 1/2 \rceil$. In what follows, $\lg(x)$ denotes $\log_2(x)$.

Let $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$ and $\mathcal{R}_q^* = \{\mathbf{x} \in \mathcal{R}_q : \exists \mathbf{y}, \mathbf{xy} = 1\}$. We identify a polynomial $\mathbf{f} = f_0 + f_1 X + \cdots + f_{n-1}X^{n-1} \in \mathcal{R}_q$ with a vector $\vec{f} = (f_0, \ldots, f_{n-1})^T \in \mathbb{Z}_q^n$. The ℓ_p and maximum norm of a polynomial $\mathbf{f}$ is denoted by $\|\mathbf{f}\|_p$ and $\|\mathbf{f}\|_\infty$, which is $(\sum_i |f_i|^p)^{1/p}$ and $\max_i |f_i|$, respectively.

The Gaussian distribution with mean 0 and variance σ^2 is denoted by $N(0, \sigma)$. For a real s and $q \in \mathbb{N}$, $\bar{\Psi}_s$ denotes the folded and discretized Gaussian distribution over $\mathbb{Z}_q$, that is, $\lfloor N(0, s^2/2\pi) \rceil$ mod q. For $s \in \mathbb{R}$, we define n-dimensional Gaussian function ρ_s as $\rho_s(\vec{x}) = \exp(-\pi \cdot \|\vec{x}\|_2^2 / s^2)$. For countable set $L \subseteq \mathbb{R}^n$, the discrete Gaussian distribution over L with parameter s is $D_{L,s}(\vec{x}) = \rho_s(\vec{x}) / (\sum_{\vec{y} \in L} \rho_s(\vec{y}))$ for $\vec{x} \in L$. Identifying $\mathcal{R}_q$ and $\mathbb{Z}_q^n$, we may choose a polynomial in $\mathcal{R}_q^n$ from the distribution $\bar{\Psi}_s^n$ or $D_{\mathbb{Z}^n, \sigma}$ over $\mathbb{Z}^n$.

For a polynomial $\mathbf{s} \in \mathcal{R}_q$ and a distribution χ over $\mathcal{R}_q$, we define the oracle $A_{\mathbf{s},\chi}$ as follows: (1) take samples $\mathbf{a} \leftarrow \mathcal{R}_q$ and $\mathbf{e} \leftarrow \chi$ and (2) output $(\mathbf{a}, \mathbf{as} + \mathbf{e})$. Let $U(\mathcal{R}_q^2)$ be a oracle which returns two random elements in $\mathcal{R}_q$.

Let us define the RLWE$_{q,\chi}$ problem and an assumption related to the problem.

Definition 9. Let $n = 2^z$, let q be a prime congruent to 1 modulo $2n$, and let χ be a distribution over $\mathcal{R}_q$. The RLWE$_{q,\chi}$ *problem* is distinguishing $A_{\mathbf{s},\chi}$ from $U(\mathcal{R}_q^2)$, where $\mathbf{s}$ is chosen at random from $\mathcal{R}_q$. The advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{RLWE}}(n) = \left| \Pr[\mathcal{A}^{A_{\mathbf{s},\chi}}(1^n) = 1] - \Pr[\mathcal{A}^{U(\mathcal{R}_q^2)}(1^n) = 1] \right|,$$

where the probability is defined by $\mathcal{A}$'s random coins, choices of $\mathbf{s} \in \mathcal{R}_q$, and randomness of the oracles. We say that the RLWE$_{q,\chi}$ *assumption* holds if for any PPT adversary $\mathcal{A}$ its advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{RLWE}}(n)$ is negligible in n.

Lemma 1 ([4, Lemma 2], adapted, and [56]). *Let $n = 2^z$ and q be a prime with $q \equiv 1 \bmod 2n$. Then, for any $k \geq 1$, there is a randomized reduction from distinguishing $A_{\mathbf{s}',\chi}$ from $U(\mathcal{R}_q^2)$ with $k+1$ samples where $\mathbf{s}' \leftarrow \chi$ to distinguishing $A_{\mathbf{s},\chi}$ from $U(\mathcal{R}_q^2)$ with k samples where $\mathbf{s} \leftarrow U(\mathcal{R}_q)$.*

We finally note that the RLWE$_{q,\chi}$ problem is reduced to the approximating shortest vector problem in any ideal of the polynomial ring. The details are in [40, 19].

5.2 The Lyubashevsky–Peikert–Regev Scheme

We here describe the modified version of the PKE scheme which appeared in the presentation of [40]. The differences are error checking steps to wGen and wEnc, which make the scheme perfectly correct.

Let us define the parameters as the follows:

$$\begin{array}{lll} q = \mathrm{poly}(n) : \text{prime}, & p = 2, & t = \omega(\sqrt{\lg n}), \\ s \leq \sqrt{\dfrac{q-p}{2(2n+1)pt^2}}, & \chi = \bar{\Psi}_s^n, & \lambda = n \lg(s), \\ \gamma = 2^{-\lambda/2}, & \delta = 2^{-\lambda/2}, & \end{array}$$

where t is a threshold parameter.

The following is a description of the PKE scheme wPKE$_{\mathrm{LPR}}$ = (wGen, wEnc, wDec) with $w\mathcal{MS} = \mathbb{Z}_p^n$, $w\mathcal{RS}_E = \{0,1\}^*$, $w\mathcal{CS} = \mathcal{R}_q^2$. The public parameter contains the security parameter and a polynomial $\mathbf{a}$ chosen from $\mathcal{R}_q^*$ uniformly at random.

wGen$(1^n; r_g)$: (1) Generate two polynomials $\mathbf{s}, \mathbf{e} \leftarrow \chi$. (2: error checking) If $\|\mathbf{s}\|_\infty, \|\mathbf{e}\|_\infty \leq st$, then go to next step; otherwise go to step (2). (3) Compute $\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e} \in \mathcal{R}_q$. (4) The encryption key is $pk = (\mathbf{a}, \mathbf{b})$ and the decryption key is $sk = \mathbf{s}$.

wEnc$_{pk}(\mathbf{k}; r_e)$: (1) Generate three polynomials $\mathbf{t}, \mathbf{e}_1, \mathbf{e}_2 \leftarrow \chi$. (2: error checking) If $\|\mathbf{t}\|_\infty, \|\mathbf{e}_1\|_\infty, \|\mathbf{e}_2\|_\infty \leq st$, then go to next step; otherwise go to step (1). (3) Compute $\mathbf{u} \leftarrow \mathbf{at} + \mathbf{e}_1$ and $\mathbf{v} \leftarrow \mathbf{bt} + \mathbf{e}_2$. (4) Compute $\mathbf{c} \leftarrow \mathbf{v} + \lfloor (q/p)\mathbf{k} \rceil$. (5) The ciphertext is $(\mathbf{u}, \mathbf{c})$.

wDec$_{sk}(\mathbf{u}, \mathbf{c})$: (1) Compute $\mathbf{d} \leftarrow \mathbf{c} - \mathbf{us} \in \mathcal{R}_q$. (2) Output $\mathbf{k}' \leftarrow \lfloor (p/q)\mathbf{d} \rceil$ mod p.

We can show the following lemmas from our parameter settings.

Lemma 2 (Perfect Correctness). *There are no decryption errors in* $\mathsf{wPKE}_{\mathrm{LPR}}$.

Lemma 3 (IND-CPA Security). *Under the* $\mathrm{RLWE}_{q,\chi}$ *assumption, the scheme* $\mathsf{wPKE}_{\mathrm{LPR}}$ *is* IND-CPA *secure.*

Lemma 4 (Semi-Uniformity). $\mathsf{wPKE}_{\mathrm{LPR}}$ *is* (δ, γ)*-semi-uniform.*

The proofs of the above lemmas will appear in the full version.

Applying the FO conversion, we obtain an IND-CCA secure PKE scheme, $\mathsf{PKE}_{\mathrm{LPR}}$, based on the $\mathrm{RLWE}_{q,\chi}$ problem.

Theorem 3. *Let* $\mathsf{PKE}_{\mathrm{LPR}} = \mathsf{FO}(\mathsf{wPKE}_{\mathrm{LPR}})$ *where we employ the random oracle* $H : \mathbb{Z}_p^n \to w\mathcal{RS}_E$ *with* $\mathcal{MS} = \mathcal{RS}_E = \mathbb{Z}_p^{n/2}$*. Then* $\mathsf{PKE}_{\mathrm{LPR}}$ *is* IND-CCA *secure if the* $\mathrm{RLWE}_{q,\chi}$ *assumption holds.*

5.3 The Stehlé and Steinfeld Scheme

Moreover, we can employ a secure variant of the NTRU encryption scheme [27] by Stehlé and Steinfeld [56] with the simplified variant of Ring-LWE by Ducas and Durms [19].

Comparing with the LPR scheme, there are additional parameters. A parameter σ and a constant ϵ defines the uniformity of the public key. We define the parameters as follows:

$$\begin{aligned} &q = \mathrm{poly}(n) : \text{prime}, && p = 2 \in \mathcal{R}_q^*, && t = \omega(\sqrt{\lg n}), \\ &s \leq \frac{q}{4n\sigma t p(4p+1)}, && \epsilon \in (0, 1/4), && \sigma \geq 2n\sqrt{\ln(8nq)} \cdot q^{\frac{1}{2}+2\epsilon}, \\ &\chi = \bar{\Psi}_s^n, && \lambda = n \lg(s), && \\ &\gamma = 2^{-\lambda/2}, && \delta = 2^{-\lambda/2}. && \end{aligned}$$

We modify the original scheme by adding error checking steps to wGen and wEnc. The following is a description of the PKE scheme $\mathsf{wPKE}_{\mathrm{SS}} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wDec})$ with $w\mathcal{MS} = \mathcal{R}/p\mathcal{R}$, $w\mathcal{RS}_E = \{0,1\}^*$, $w\mathcal{CS} = \mathcal{R}_q$.

$\mathsf{wGen}(1^n; r_g)$**:** (1) Generate a random polynomial $\mathbf{f}' \leftarrow D_{\mathbb{Z}^n,\sigma}$ and compute $\mathbf{f} = p\mathbf{f}' + 1$; if $\mathbf{f} \notin \mathcal{R}_q^*$, re-sample; if $\|\mathbf{f}\|_2 \geq 4p\sqrt{n}\sigma$, re-sample. (2) Generate a random polynomial $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$; if $\mathbf{g} \notin \mathcal{R}_q^*$, re-sample; if $\|\mathbf{g}\|_2 \geq \sqrt{n}\sigma$, re-sample. (3) Compute $\mathbf{h} \leftarrow p\mathbf{g}/\mathbf{f} \in \mathcal{R}_q^*$. (4) The encryption key is $pk = \mathbf{h}$ and the decryption key is $sk = \mathbf{f}$.

$\mathsf{wEnc}_{pk}(\mathbf{k}; r_e)$**:** (1) Generate three polynomials $\mathbf{s}, \mathbf{e} \leftarrow \chi$. (2: error checking) If $\|\mathbf{s}\|_\infty, \|\mathbf{e}\|_\infty \leq st$ then go to next step; otherwise, go to step (1). (3) Compute $\mathbf{c} \leftarrow \mathbf{hs} + p\mathbf{e} + \mathbf{k} \in \mathcal{R}_q$. (4) The ciphertext is $\mathbf{c}$.

$\mathsf{wDec}_{sk}(\mathbf{c})$**:** (1) Compute $\mathbf{d} \leftarrow \mathbf{cf} \in \mathcal{R}_q$. (2) Output $\mathbf{k}' \leftarrow \mathbf{d} \bmod p$.

The properties of the scheme are summarized as follows:

Lemma 5 (Perfect Correctness). *There are no decryption errors in* $\mathsf{wPKE}_{\mathrm{SS}}$.

Lemma 6 (IND-CPA Security). *Under the* $\mathrm{RLWE}_{q,\chi}$ *assumption, the scheme* $\mathsf{wPKE}_{\mathrm{SS}}$ *is* IND-CPA *secure*

Lemma 7 (Semi-Uniformity). $\mathsf{wPKE}_{\mathrm{SS}}$ *is* (δ, γ)*-semi-uniform.*

The proofs will appear in the full version.

Applying the FO conversion, we again obtain an IND-CCA secure PKE scheme based on the $\mathrm{RLWE}_{q,\chi}$ assumption.

Theorem 4. *Let* $\mathsf{PKE}_{\mathrm{SS}} = \mathsf{FO}(\mathsf{wPKE}_{\mathrm{SS}})$ *employing the random oracle* $H : \mathbb{Z}_p^n \to w\mathcal{RS}_E$ *with* $\mathcal{MS} = \mathcal{RS}_E = \mathbb{Z}_p^{n/2}$*. Then* $\mathsf{PKE}_{\mathrm{SS}}$ *is* IND-CCA *secure if the* $\mathrm{RLWE}_{q,\chi}$ *assumption holds.*

The proof is obtained by combining the previous lemmas.

Remark 2. We note that Steinfeld et al. [58, Sect. 4] proposed an IND-CCA secure PKE scheme based on $\mathsf{wPKE}_{\mathrm{SS}}$. From their parameter settings [58, Sect. 4.2], a public key and a ciphertext includes at least 8 polynomials, while ours consists of one polynomial.

5.4 Efficiency of the Ring-LWE-based Construction

As a concrete example, we consider the AKE protocol instantiated from $\mathsf{PKE}_{\mathrm{LPR}}$. In the protocol, the initiator sends $C \in \mathcal{R}_q^2$ and $ek \in \mathcal{R}_q$ and the responder sends $C \in \mathcal{R}_q^2$ twice. Hence, the communication costs are seven polynomials, which are $7n \lg q$ bits.

Next, we take parameters chosen by Rückert and Schneider [54, Table 12], which are more conservative than these chosen by Lindner and Peikert [38, Section 1.2]. We let $n = 512$ and $q = 5941249$ by following them. We additionally set $s = 8.0$ and $t = 4.5$ in order to make scheme perfectly correct. Such parameters yield $7n \lg q \approx 80.65$ kbits as the concrete communication costs.

On the above parameter $(n, q) = (512, 5941249)$, Pöppelmann and Güneysu [49] reported FPGA implementation of the computations in $\mathbb{Z}_q[X]/(X^{2^z} + 1)$. Their implementation requires 53 μs per polynomial multiplication on a Spartan-6 LX100.

We compare the existing AKE protocol with our instantiated protocol under the Ring-LWE assumption. Using the generic construction by Fujioka et al. [23] with IND-CCA secure KEM scheme under the ring-LWE assumption, we obtain an AKE protocol based on the ring-LWE problem. There are CCA secure PKE schemes based on the ring-LWE assumption in the StdM: the ring-LWE version of the Peikert PKE scheme [45] over polynomial ring, and the scheme [37] obtained by applying the CHK conversion [8] to the ring-LWE version of the ABB IBE scheme [1]. Those schemes require larger keys and ciphertext, $\kappa \cdot O(\lg q)$ polynomials in the former scheme and $O(\lg q)$ polynomials in the latter scheme, and q should be larger than ours from technical reasons. As a concrete example, we can set $2\lceil \lg q \rceil + 1$ as the number of polynomials in keys and ciphertexts. If we adopted $(n, q) = (512, 5941249)$ as the parameters, then a ciphertext consists from 47 polynomials and the length of it results in approximately 541.50 kbits. In the protocol, the initiator sends a ciphertext of IND-CCA KEM and an encapsulation key of IND-CPA KEM, and the responder sends a ciphertext of IND-CCA KEM and that of IND-CPA KEM. Adopting $\mathsf{wPKE}_{\mathrm{LPR}}$ as IND-CPA KEM, we obtain 97 polynomials ≈ 1.12 Mbits as the communication costs. For comparison, see Table 1.

6. OTHER INSTANTIATIONS

Instantiations from Diffie-Hellman problems.

We can realize various AKE protocols as concrete instantiations of our generic construction GCwR in Section 3 from OW-CCA KEM schemes based on the hardness of the DH problem and its variants. For instance, we can adopt the following efficient KEM schemes based on computational DH (CDH) assumption and gap DH (GDH) assumption.

From the CDH assumption, we have PSEC-KEM [32] that is IND-CCA secure in the ROM under the CDH assumption [24].

From the GDH assumption, we have ECIES-KEM [32] that is IND-CCA secure in the ROM under the GDH assumption [16].

In addition, we have a KEM scheme ($ek = g^x$, $K = H(g^r)$, and $C = (g^x)^r$) that is already IND-CCA secure in the ROM under the GDH assumption, where y is a public key, as Boyen [11] pointed out.

Instantiations from factoring or RSA.

We have new AKE protocols as concrete instantiations of our generic construction GCwR in Section 3 based on the hardness of integer factoring, which are slightly more efficient than those in the standard model in [23], say, from 2.27kbits to 1.30kbits.

From the factoring assumption, we have Rabin-SAEP$^+$ [7] or Rabin-KEM [16] that are IND-CCA secure in the ROM under the factoring assumption.

For the RSA assumption, we have the REACT PKE scheme [44] that was shown to be OW-CCA secure PKE scheme under the RSA assumption.

If we employ the Rabin and RSA PKE schemes as wKEM, we may face an efficiency issue because these schemes require prime generation in the key-generation algorithm, which is executed in the key-generation algorithms of wKEM in our generic construction GCwR. In order to address this issue, we introduce the following simple OW-CPA secure KEM based on Wee's IND-CPA PKE scheme [59], which requires no prime generation in the key-generation algorithm and has ciphertext of length $2|N|$. Let us briefly recall the properties of the signed quadratic residues [28, 29] Fix a Blum integer $N = pq$ for safe primes $p, q \equiv 3 \pmod 4$. The signed quotient group $\mathbb{QR}_N^t = \mathbb{QR}_N/(\pm 1)$ is a cyclic group of order $(p-1)(q-1)/4$ and efficiently recognizable by computing Jacobi symbol. Let g be a random generator of $\mathbb{QR}_N^t$. The public parameter is (N, g) and the key pair is (ek, dk), where $dk \leftarrow [1, (N-1)/4]$ and $ek = g^{2dk}$. The encapsulation algorithm outputs $K = g^r$ and $C = (u, \tau) = (g^{2r}, (ek \cdot g)^r)$, where $r \leftarrow [1, (N-1)/4]$. The decapsulation algorithm outputs $K = \tau \cdot u^{-dk}$. The one-wayness follows from the hardness of factoring the Blum integer with safe primes.

Instantiations from codes.

We can instantiate new AKE protocols as concrete instantiations of our generic construction GCwR in Section 3 from code-based problems, though the protocols are not so efficient.

We have Dowsley et al.'s PKE [18] scheme that is IND-CCA secure in the StdM under the McEliece and LPN assumptions. (See Ref. [18] for definitions of these assumptions.)

We also have a padding for the McEliece PKE scheme [33] that is IND-CCA secure in the ROM under the McEliece one-way assumption. Cayrel, Hoffmann, and Persichetti [14] provided a variant of McEliece PKE scheme, converted it into IND-CCA secure schemes by the FO conversion, and reported implementation result.

On the key size, Bernstein et al. [6] estimated the size of a public key of the McEliece using binary Goppa codes at about 1.8 Mbits for 128-bit security. Since the key is huge, there are several attempts to reduce the size of keys. Cayrel et al. [14] reported an implementation result based on very structured code in [48].

If we adapt the scheme in [14] with parameter set in [48, Table 3][2], then we obtain an AKE protocol with communication costs about $|ek| + 3|C| = 52.32$ kbits. If we adopt the IND-CCA secure PKE scheme in [18] with parameter set in [48, Table 3] and a one-time signature scheme whose verification key and signature are of length 128 bits, then the communication costs results in approximately $|ek| + |C| + 2 \cdot 128 \cdot |C| \approx 1.31$ Mbits.

Instantiation from NTRU.

We have already a padding scheme for NTRU encryption [27, 31]. Hence, we can have an AKE protocol based on the NTRU encryption scheme. The scheme enjoys fast computation and light communication costs, as the ring-LWE based AKE does. For example, the communication costs is only $|ek| + 3|C| \approx 19.76$ kbits if we employ the parameter set [31, ees449ep1] for 128-bit security and optimized size. [3]

Instantiation from subset sum.

As a bonus of the ROM, we also have a new AKE protocol from the subset-sum problem. Lyubashevsky, Palacio, and Segev [39] proposed an IND-CPA secure PKE scheme based on the subset-sum problem. To obtain IND-CCA PKE scheme by applying the FO conversion [24], we have to modify the parameters of the scheme slightly, and the details will appear in the full version of this paper. For example, if we set the security parameter n as 256, then we have 1.30 Mbits for the communication cost. [4]

Instantiation from multi-variate quadratic systems.

It is known that solving multi-variate quadratic equations over $\mathbb{F}_q$ is hard in the worst case. There were several PKE schemes exploiting its hardness and several attacks against them.

Huang, Liu, and Yang [30] proposed a new PKE scheme based on the hardness of solving a special case of multi-variate quadratic equations over $\mathbb{F}_q$. A slight modification on the parameters makes their scheme perfectly correct as in the case of subset-sum based construction (the details appear in the full version). When we adapt their estimation [30, Case 1], the communication cost is 11.44 Mbits. [5] Meanwhile, we can hide the quadratic part from public parameters. Employing the idea in the case of LWE-based PKE [46], we have $pp \in \mathbb{Z}_q^{m \times n}$, $ek \in \mathbb{Z}_q^{m \times \ell}$, and $C \in \mathbb{Z}_q^{n+\ell}$. In this case, we have $|ek| = m\ell \lg q = 7.57$ Mbits and $|C| = (n+\ell)\lg q = 33.71$ kbits. They result in 7.67 Mbits as the communication cost.

7. REFERENCES

[1] Agrawal, S., Boneh, D., and Boyen, X. Efficient lattice (H)IBE in the standard model. In Gilbert [25], pp. 553–572.

[2] Agrawal, S., Boneh, D., and Boyen, X. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Rabin [51], pp. 98–115.

[3] Ajtai, M., and Dwork, C. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC '97* (1997), ACM, pp. 284–293. See also ECCC TR96-065.

[4] Applebaum, B., Cash, D., Peikert, C., and Sahai, A. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Halevi [26], pp. 595–618.

[2] Persichetti [48] proposed the parameters $(q, m, n, k, s, t) = (2^5, 2, 992, 415, 2^5, 9)$ for 128-bit security. We have $|ek| = kmt \lg q = 37440$ and $|C| = n \lg q = 4960$ bits.

[3] In the parameter set [31, ees449ep1], the parameters are $(N, q, p) = (449, 2048, 3)$. Since $ek, C \in \mathbb{Z}_q[X]/(X^N - 1)$, we have $|ek| = |C| = N \lg q = 4939$ bits.

[4] We set parameters as $(n, \ell, q) = (256, 256, 10n^2)$ instead of $q = 10n \log^2 n$ to make a scheme perfectly correct. We have $pp \in \mathbb{Z}_q^{n \times n}$, $ek \in \mathbb{Z}_q^{n \times \ell}$, and $C \in \mathbb{Z}_q^{n+\ell}$. Hence, we have $|ek| = n\ell \lg q = 1266.28$ kbits and $|C| = (n+\ell)\lg q = 9.89$ kbits.

[5] Huang et al. proposed the parameters $(n, m) = (200, 400)$ and $q \approx 2^{73.93292}$ [30, Case 1]. We additionally set $\ell = 256$, which defines the length of the plaintext. We have $pp \in \mathbb{Z}_q^{m \times n} \times (\mathbb{Z}_q^{n \times n})^m$, $ek \in \mathbb{Z}_q^m$, and $C \in (\mathbb{Z}_q^{n+1})^\ell$. Hence, we have $|ek| = m \lg q = 29.57$ kbits and $|C| = \ell(n+1)\lg q = 3.80$ Mbits. Consequently, we have 11.44 Mbits as the communication cost.

[5] Bellare, M., and Rogaway, P. Random oracle are practical: A paradigm for designing efficient protocols. In *CCS '93* (1993), ACM, pp. 62–73.

[6] Bernstein, D. J., Lange, T., and Peters, C. Smaller decoding exponents: Ball-collision decoding. In *CRYPTO 2011* (2011), P. Rogaway, Ed., vol. 6841 of *LNCS*, Springer, Heidelberg, pp. 743–760.

[7] Boneh, D. Simplified OAEP for the RSA and Rabin functions. In *CRYPTO 2001* (2001), J. Kilian, Ed., vol. 2139 of *LNCS*, Springer, Heidelberg, pp. 275–291.

[8] Boneh, D., Canetti, R., Halevi, S., and Katz, J. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing 36*, 5 (12 2006), 1301–1328.

[9] Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., and Zhandry, M. Random oracles in a quantum world. In *ASIACRYPT 2011* (2011), D. H. Lee and X. Wang, Eds., vol. 7073 of *LNCS*, Springer, Heidelberg, pp. 41–69.

[10] Boyd, C., Cliff, Y., González Nieto, J. M., and Paterson, K. G. One-round key exchange in the standard model. *International Journal of Applied Cryptography (IJACT) 1*, 3 (2009), 181–199. A preliminary version appeared in *ACISP 2008*, 2008.

[11] Boyen, X. Miniature CCA2 PK encryption: Tight security without redundancy. In Kurosawa [35], pp. 485–501.

[12] Canetti, R., and Krawczyk, H. Analysis of key-exchange protocols and their use for building secure channels. In *EUROCRYPT 2001* (2001), B. Pfitzmann, Ed., vol. 2045 of *LNCS*, Springer, Heidelberg, pp. 453–474.

[13] Cash, D., Hofheinz, D., Kiltz, E., and Peikert, C. Bonsai trees, or how to delegate a lattice basis. In Gilbert [25], pp. 523–552.

[14] Cayrel, P.-L., Hoffmann, G., and Persichetti, E. Efficient implementation of a CCA2-secure variant of McEliece using generalized Srivastava codes. In Fischlin et al. [22], pp. 138–155.

[15] Coron, J.-S., Gouget, A., Paillier, P., and Villegas, K. SPAKE: A single-party public-key authenticated key exchange protocol for contact-less applications. In *FC 2010 Workshops* (2010), R. Sion, R. Curtmola, S. Dietrich, A. Kiayias, J. M. Miret, K. Sako, and F. Sebé, Eds., vol. 6054 of *LNCS*, Springer, Heidelberg, pp. 107–122.

[16] Dent, A. W. A designer's guide to KEMs. In *IMA 2003* (2003), K. G. Paterson, Ed., vol. 2898 of *LNCS*, Springer, Heidelberg, pp. 133–151.

[17] Diffie, W., and Hellman, M. E. New directions in cryptography. *IEEE Transactions on Information Theory 22* (November 1976), 644–654.

[18] Dowsley, R., Müller-Quade, J., and Nascimento, A. C. A. A CCA2 secure public key encryption scheme based on the McEliece assumptions in the standard model. In *CT-RSA 2009* (2009), M. Fischlin, Ed., vol. 5473 of *LNCS*, Springer, Heidelberg, pp. 240–251.

[19] Ducas, L., and Durmus, A. Ring-LWE in polynomial rings. In Fischlin et al. [22], pp. 34–51.

[20] Dwork, C., Naor, M., and Reingold, O. Immunizing encryption schemes from decryption errors. In *EUROCRYPT 2004* (2004), C. Cachin and J. Camenisch, Eds., vol. 3027 of *LNCS*, Springer, Heidelberg, pp. 342–360.

[21] ElGamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transaction on Information Theory 31*, 4 (1985), 469–472.

[22] Fischlin, M., Buchmann, J., and Manulis, M., Eds. *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012, Proceedings* (2012), vol. 7293 of *LNCS*, Springer, Heidelberg.

[23] Fujioka, A., Suzuki, K., Xagawa, K., and Yoneyama, K. Strongly secure authenticated key exchange from factoring, codes, and lattices. In Fischlin et al. [22], pp. 467–484.

[24] Fujisaki, E., and Okamoto, T. How to enhance the security of public-key encryption at minimum cost. *IEICE transactions on fundamentals of electronics, communications and computer sciences 83*, 1 (2000), 24–32. A preliminary version appeared in *PKC '99*, 1999.

[25] Gilbert, H., Ed. *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30-June 3, 2010. Proceedings* (2010), vol. 6110 of *LNCS*, Springer, Heidelberg.

[26] Halevi, S., Ed. *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings* (2009), vol. 5677 of *LNCS*, Springer, Heidelberg.

[27] Hoffstein, J., Pipher, J., and Silverman, J. H. NTRU: A ring-based public key cryptosystem. In *ANTS-III* (1998), J. Buhler, Ed., vol. 1423 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 267–288.

[28] Hofheinz, D., and Kiltz, E. The group of signed quadratic residues and applications. In Halevi [26], pp. 637–653.

[29] Hofheinz, D., and Kiltz, E. Practical chosen ciphertext secure encryption from factoring. In *EUROCRYPT 2009* (2009), A. Joux, Ed., vol. 5479 of *LNCS*, Springer, Heidelberg, pp. 313–332.

[30] Huang, Y.-J., Liu, F.-H., and Yang, B.-Y. Public-key cryptography from new multivariate quadratic assumptions. In Fischlin et al. [22], pp. 190–205.

[31] IEEE. *IEEE P1363.1/D12 Draft Standard for Public-Key Cryptographic Techniques Based on Hard Problems over Lattices*, October 2008. Available at `http://grouper.ieee.org/groups/1363/lattPK/`.

[32] ISO/IEC. *ISO/IEC 18033-2 Information technology—Security techniques—Encryption algorithms—Part 2: Asymmetric ciphers*. Geneva, 2006.

[33] Kobara, K., and Imai, H. Semantically secure McEliece public-key cryptosystems –conversions for McEliece PKC–. In *PKC 2001* (2001), K. Kim, Ed., vol. 1992 of *LNCS*, Springer, Heidelberg, pp. 19–35.

[34] Krawczyk, H. HMQV: A high-performance secure Diffie-Hellman protocol. In *CRYPTO 2005* (2005), V. Shoup, Ed., vol. 3621 of *LNCS*, Springer, Heidelberg, pp. 546–566.

[35] Kurosawa, K., Ed. *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, December 2-6, 2007, Proceedings* (2007), vol. 4833 of *LNCS*, Springer, Heidelberg.

[36] LaMacchia, B. A., Lauter, K., and Mityagin, A. Stronger security of authenticated key exchange. In *ProvSec 2007* (2007), W. Susilo, J. K. Liu, and Y. Mu, Eds., vol. 4784 of *LNCS*, Springer, Heidelberg, pp. 1–16.

[37] Langlois, A., and Stehlé, D. Hardness of decision (R)LWE for any modulus. Cryptology ePrint Archive, Report 2012/091, 2012. Available at `http://eprint.iacr.org/2012/091`.

[38] Lindner, R., and Peikert, C. Better key sizes (and attacks) for LWE-based encryption. In *CT-RSA 2011* (2011), A. Kiayias, Ed., vol. 6558 of *LNCS*, Springer, Heidelberg, pp. 319–330.

[39] Lyubashevsky, V., Palacio, A., and Segev, G. Public-key cryptographic primitives provably as secure as subset sum. In *TCC 2010* (2010), D. Micciancio, Ed., vol. 5978 of *LNCS*, Springer, Heidelberg, pp. 382–400.

[40] Lyubashevsky, V., Peikert, C., and Regev, O. On ideal lattices and learning with errors over rings. In Gilbert [25], pp. 1–23.

[41] McEliece, R. J. A public key cryptosystem based on algebraic coding theory. Tech. rep., DSN progress report, 1978.

[42] Merkle, R. C., and Hellman, M. E. Hiding information and signatures in trap door knapsacks. *IEEE Transactions on Information Theory 24*, 5 (September 1978), 525–530.

[43] Okamoto, T. Authenticated key exchange and key encapsulation in the standard model. In Kurosawa [35], pp. 474–484.

[44] Okamoto, T., and Pointcheval, D. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In *CT-RSA 2001* (2001), D. Naccache, Ed., vol. 2020 of *LNCS*, Springer, Heidelberg, pp. 159–175.

[45] Peikert, C. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC 2009* (2009), M. Mitzenmacher, Ed., ACM, pp. 333–342.

[46] Peikert, C., Vaikuntanathan, V., and Waters, B. A framework for efficient and composable oblivious transfer. In *CRYPTO 2008* (2008), D. Wagner, Ed., vol. 5157 of *LNCS*, Springer, Heidelberg, pp. 554–571.

[47] Peikert, C., and Waters, B. Lossy trapdoor functions and their applications. In *STOC 2008* (2008), C. Dwork, Ed., ACM, pp. 187–196.

[48] Persichetti, E. Compact McEliece keys based on quasi-dyadic Srivastava codes. *Journal of Mathematical Cryptology 6*, 2 (2012), 149–169.

[49] Pöppelman, T., and Güneysu, T. Towards efficient arithmetic for lattice-based cryptography on reconfigurable hardware. In *LATINCRYPT 2012* (2012), A. Hevia and G. Neven, Eds., vol. 7533 of *LNCS*, Springer, Heidelberg, pp. 139–158.

[50] Rabin, M. O. Digitalized signatures and public-key functions as intractable as factorization. Tech. rep., MIT, January 1979.

[51] Rabin, T., Ed. *Advances in Cryptology - CRYPTO 2010, 30th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings* (2010), vol. 6223 of *LNCS*, Springer, Heidelberg.

[52] Regev, O. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM 56*, 6 (2009), Article 34. A preliminary version appeared *STOC 2005*, 2005.

[53] Rivest, R. L., Shamir, A., and Adleman, L. M. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM 21*, 2 (Febrary 1978), 120–126.

[54] Rückert, M., and Schneider, M. Estimating the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2010/137, 2010. Available at `http://eprint.iacr.org/2010/137`.

[55] Shor, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing 26*, 5 (1997), 1484–1509.

[56] Stehlé, D., and Steinfeld, R. Making NTRU as secure as worst-case problems over ideal lattices. In *EUROCRYPT 2011* (2011), K. G. Paterson, Ed., vol. 6632 of *LNCS*, Springer, Heidelberg, pp. 27–47.

[57] Stehlé, D., Steinfeld, R., Tanaka, K., and Xagawa, K. Efficient public key encryption based on ideal lattices. In *ASIACRYPT 2009* (2009), M. Matsui, Ed., vol. 5912 of *LNCS*, Springer, Heidelberg, pp. 617–635.

[58] Steinfeld, R., Ling, S., Pieprzyk, J., Tartary, C., and Wang, H. NTRUCCA: How to strengthen NTRUEncrypt to chosen-ciphertext security in the standard model. In Fischlin et al. [22], pp. 353–371.

[59] Wee, H. Efficient chosen-ciphertext security via extractable hash proofs. In Rabin [51], pp. 314–332.

[60] Zhandry, M. Secure identity-based encryption in the quantum random oracle model. In *CRYPTO 2012* (2012), R. Safavi-Naini and R. Canetti, Eds., vol. 7417 of *LNCS*, Springer, Heidelberg, pp. 758–775.

APPENDIX

A. PROOF OF THEOREM 1

In the experiment of CK$^+$ security, we suppose that sid^* is the session identity for the test session. We will show that if a PPT bounded adversary $\mathcal{A}$ can distinguish the session key of a fresh session from a randomly chosen session key, we can construct a OW-CCA or OW-CPA adversary $\mathcal{S}$. Let κ be the security parameter, and let $\mathcal{A}$ be a polynomially (in κ) bounded adversary. We use adversary $\mathcal{A}$ to construct $\mathcal{S}$ that succeeds with non-negligible probability. Suc denotes the event that $\mathcal{A}$ wins. Let AskH be the event that adversary $\mathcal{A}$ poses $(K_{A,1}, K_{B,1}, K_{B,2}, U_A, U_B, ek_{A,1}, ek_{B,1}, C_{A,1}, ek_{A,2}, C_{B,1}, C_{B,2})$ to H_2. Let $\overline{\text{AskH}}$ be the complement of event AskH. Let sid be any completed session owned by an honest party such that $\mathsf{sid} \neq \mathsf{sid}^*$ and sid is non-matching to sid^*. Since sid and sid^* are distinct and non-matching, the inputs to the key derivation function H_2 are different for sid and sid^*. Since H_2 is a random oracle, $\mathcal{A}$ cannot obtain any information about the test session key from the session keys of non-matching sessions. Hence, $\Pr[\text{Suc} \wedge \overline{\text{AskH}}] \leq \frac{1}{2}$ and $\Pr[\text{Suc}] = \Pr[\text{Suc} \wedge \text{AskH}] + \Pr[\text{Suc} \wedge \overline{\text{AskH}}] \leq \Pr[\text{Suc} \wedge \text{AskH}] + \frac{1}{2}$. Henceforth, the event Suc ∧ AskH is denoted by Suc*.

For party P, we denote the static secret key by $dk_{P,1}$, the ephemeral secret key for an initiator by $r_{P,1} \in \{0,1\}^\kappa$ and $r_{P,2} \in \mathcal{RS}_G$ and the ephemeral secret key for a responder by $r_{P,1} \in \{0,1\}^\kappa$ and $r_{P,2} \in \mathcal{RS}_E$. Assume that $\mathcal{A}$ succeeds in an environment with N users, activates at most ℓ sessions within a party.

We consider the following events.

- Let AskS be the event that $\mathcal{A}$ poses the static secret key $dk_{A,1}$ to H_1 when U_A is the owner of sid^*.
- Let $\overline{\text{AskS}}$ be the complement of event AskS.

More specifically, we consider the following events that cover all cases of the behavior of $\mathcal{A}$.

- Let E_1 be the event that the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the owner of sid^* is the initiator and the static secret key of the initiator is given to $\mathcal{A}$.
- Let E_2 be the event that the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the owner of sid^* is the initiator and the ephemeral secret key of sid^* is given to $\mathcal{A}$.

- Let E_3 be the event that the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the owner of sid^* is the responder and the static secret key of the responder is given to $\mathcal{A}$.
- Let E_4 be the event that the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the owner of sid^* is the responder and the ephemeral secret key of sid^* is given to $\mathcal{A}$.
- Let E_5 be the event that the test session sid^* has matching session $\overline{\mathsf{sid}}^*$, and both static secret keys of the initiator and the responder are given to $\mathcal{A}$.
- Let E_6 be the event that the test session sid^* has matching session $\overline{\mathsf{sid}}^*$, and both ephemeral secret keys of sid^* and $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$.
- Let E_7 be the event that the test session sid^* has matching session $\overline{\mathsf{sid}}^*$, and the static secret key of the owner of sid^* and the ephemeral secret key of $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$.
- Let E_8 be the event that the test session sid^* has matching session $\overline{\mathsf{sid}}^*$, and the ephemeral secret key of sid^* and the static secret key of the owner of $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$.

To finish the proof, we investigate events $\textsc{AskS} \wedge \textsc{Suc}^*$ and $E_i \wedge \overline{\textsc{AskS}} \wedge \textsc{Suc}^*$ ($i = 1, \ldots, 8$) that cover all cases of event $\textsc{Suc}^*$.

A.1 Event $\textsc{AskS} \wedge \textsc{Suc}^*$

$\mathcal{S}$ receives the challenge ciphertext C^* for the key K^*. In the event $\textsc{AskS}$, $\mathcal{A}$ poses the static secret key $dk_{A,1}$ to H_1. $\mathcal{S}$ embeds the instance as $C^*_{B,1} = C^*$ and decrypts C^* with $dk_{A,1}$. Then, $\mathcal{S}$ obtains K^*.

A.2 Event $E_1 \wedge \overline{\textsc{AskS}} \wedge \textsc{Suc}^*$

In the event E_1, the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the static secret key of U_A is given to $\mathcal{A}$. In the case of event $E_1 \wedge \overline{\textsc{AskS}} \wedge \textsc{Suc}^*$, OW-CCA adversary $\mathcal{S}$ performs the following steps.

Initialization.

$\mathcal{S}$ receives the public key ek^* as a challenge. Also, $\mathcal{S}$ receives the challenge ciphertext C^* for the key K^*.

Setup.

$\mathcal{S}$ randomly selects two parties U_A, U_B and $i \in [1, \ell]$ that becomes a guess of the test session with probability $1/N^2\ell$. $\mathcal{S}$ sets all N users' static secret and public keys except U_B. $\mathcal{S}$ selects $r_I \in \mathcal{RS}_G$ randomly, runs the key generation algorithm $(dk_{I,1}, ek_{I,1}) \leftarrow \mathsf{KeyGen}(r_I)$, and sets U_I's static secret and public key as $(dk_{I,1}, ek_{I,1})$. $\mathcal{S}$ sets ek^* as the static public key of U_B.

Also, $\mathcal{S}$ sets the ephemeral public key of i-th session of U_A as follows: $\mathcal{S}$ generates $ek_{A,2}$ obeying the protocol and sets $(C^*, ek_{A,2})$ as the ephemeral public key.

Simulation.

$\mathcal{S}$ simulates oracle queries by $\mathcal{A}$ as follows. $\mathcal{S}$ maintains the lists $\mathcal{L}_{H_1}$ and $\mathcal{L}_{H_2}$ that contains queries and answers of the H_1 and H_2 oracles respectively, and the list $\mathcal{L}_{SK}$ that contains queries and answers of $\mathsf{SessionKeyReveal}$.

1. $H_1(r_i, dk_i)$: If there exists a tuple $(r_i, dk_i, *) \in \mathcal{L}_{H_1}$, $\mathcal{S}$ returns the registered value [6]; otherwise, $\mathcal{S}$ chooses $h_i \in \mathcal{RS}_E$ randomly, returns h_i and records it to $\mathcal{L}_{H_1}$.
2. $H_2(K_{P,1}, K_{Q,1}, K_{Q,2}, U_P, U_Q, ek_{P,1}, ek_{Q,1}, C_{P,1}, ek_{P,2}, C_{Q,1}, C_{Q,2})$:
   (a) If $P = A$, $Q = B$, $C_{A,1} = C^*$, and $(\Pi, \mathcal{I}, U_A, U_B, ek_{A,1}, ek^*, (C^*, ek_{A,2}), (C_{B,1}, C_{B,2}))$ is i-th session of U_A, then $\mathcal{S}$ outputs $K_{A,1}$ as the answer of OW-CCA game (i.e., K^*);
   (b) else if there exists a tuple $(K_{P,1}, K_{Q,1}, K_{Q,2}, U_P, U_Q, ek_{P,1}, ek_{Q,1}, C_{P,1}, ek_{P,2}, C_{Q,1}, C_{Q,2}, h) \in \mathcal{L}_{H_2}$, $\mathcal{S}$ returns registered value h;
   (c) else if $P = B$ and there exists a tuple $(U_B, U_B, ek^*, ek_{Q,1}, (C_{B,1}, ek_{B,2}), (C_{Q,1}, CT_{Q,1}), h) \in \mathcal{L}_{SK}$, and $K_{B,1} = K'_{B,1}$, $K_{Q,1} = K'_{Q,1}$ and $K_{Q,2} = K'_{Q,2}$, where $\mathcal{S}$ computes $K'_{B,1} = \mathsf{DeCap}_{dk_{Q,1}}(C_{B,1})$ and $K'_{Q,2} = \mathsf{wDeCap}_{dk_{P,2}}(C_{Q,2})$, poses $C_{Q,1}$ to the decryption oracle, and the oracle outputs $K'_{Q,1}$, then $\mathcal{S}$ returns recorded value h, and records $(K_{B,1}, K_{Q,1}, K_{Q,2}, U_B, U_Q, ek^*, ek_{Q,1}, (C_{B,1}, ek_{B,2}), (C_{Q,1}, C_{Q,2}), h)$ in the list $\mathcal{L}_H$;
   (d) else if $Q = B$ and there exists a tuple $(U_P, U_B, ek_{P,1}, ek^*, (C_{P,1}, ek_{P,2}), (C_{B,1}, C_{B,2}), h) \in \mathcal{L}_{SK}$, and $K_{P,1} = K'_{P,1}$, $K_{B,1} = K'_{B,1}$ and $K_{B,2} = K'_{B,2}$, where $\mathcal{S}$ computes $K'_{B,1} = \mathsf{DeCap}_{dk_{P,1}}(C_{B,1})$ and $K'_{B,2} = \mathsf{wDeCap}_{dk_{P,2}}(C_{B,2})$, poses $C_{P,1}$ to the decryption oracle, and the oracle outputs $K'_{P,1}$, then $\mathcal{S}$ returns recorded value h, and records $(K_{P,1}, K_{B,1}, K_{B,2}, U_P, U_B, ek_{P,1}, ek^*, (C_{P,1}, ek_{P,2}), (C_{B,1}, C_{B,2}), h)$ in the list $\mathcal{L}_H$;
   (e) else if there exists a tuple $(U_P, U_Q, ek_{P,1}, ek_{Q,1}, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}), h) \in \mathcal{L}_{SK}$, and $K_{P,1} = K'_{P,1}$, $K_{Q,1} = K'_{Q,1}$ and $K_{Q,2} = K'_{Q,2}$, where $\mathcal{S}$ computes $K'_{Q,1} = \mathsf{DeCap}_{dk_{P,1}}(C_{Q,1})$, $K'_{P,1} = \mathsf{DeCap}_{dk_{Q,1}}(C_{P,1})$ and $K'_{Q,2} = \mathsf{DeCap}_{dk_{P,2}}(C_{Q,2})$, than $\mathcal{S}$ returns recorded value h and record $(K_{P,1}, K_{Q,1}, K_{Q,2}, U_P, U_Q, ek_{P,1}, ek_{Q,1}, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}), h)$ in the list $\mathcal{L}_{H_2}$;
   (f) otherwise, $\mathcal{S}$ returns a random value $SK \in \{0, 1\}^\kappa$ and records it in the list $\mathcal{L}_{H_2}$.
3. $\mathsf{Send}(\Pi, \mathcal{I}, U_P, U_Q)$: If $P = A$ and the session is i-th session of U_A, $\mathcal{S}$ returns the ephemeral public key $(C^*, ek_{A,2})$ computed in the setup. Otherwise, $\mathcal{S}$ computes the ephemeral public key $(C_{P,1}, ek_{P,2})$ obeying the protocol, returns it and records $(\Pi, U_P, U_Q, (C_{P,1}, ek_{P,2}))$.
4. $\mathsf{Send}(\Pi, \mathcal{R}, U_Q, U_P, (C_{P,1}, ek_{P,2}))$: $\mathcal{S}$ computes the ephemeral public key $(C_{Q,1}, C_{Q,2})$ obeying the protocol, and the session key SK as follows:
   (a) If $Q = B$ and there exists a tuple $(K_{P,1}, K_{B,1}, K_{B,2}, U_P, U_B, ek_P, ek^*, (C_{P,1}, ek_{P,2}), (C_{B,1}, C_{B,2}), h) \in \mathcal{L}_H$ for $(K_{P,1}, K_{B,1}, K_{B,2})$, where $K_{B,1}$ and $K_{B,2}$ are known for $\mathcal{S}$, and $\mathcal{S}$ poses $C_{P,1}$ to the decryption oracle and the oracle outputs $K_{P,1}$, then $\mathcal{S}$ sets $SK = h$;
   (b) else if there exists a tuple $(K_{P,1}, K_{Q,1}, K_{Q,2}, U_P, U_Q, ek_{P,1}, ek_{Q,1}, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}), h) \in \mathcal{L}_{H_2}$ for $(K_{P,1}, K_{Q,1}, K_{Q,2})$, where $K_{P,1} = \mathsf{DeCap}_{dk_{Q,1}}(C_{P,1})$, and $K_{Q,1}$ and $K_{Q,2}$ are known for $\mathcal{S}$, then $\mathcal{S}$ sets $SK = h$;
   (c) otherwise, $\mathcal{S}$ chooses $SK \in \{0, 1\}^\kappa$ randomly.

   Finally, $\mathcal{S}$ records $(\Pi, U_P, U_Q, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}))$ as the completed session and SK in the list $\mathcal{L}_{SK}$.
5. $\mathsf{Send}(\Pi, \mathcal{I}, U_P, U_Q, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}))$:
   (a) If $P = B$ and there exists a tuple $(K_{B,1}, K_{Q,1}, K_{Q,2}, U_B, U_Q, ek^*, ek_Q, (C_{B,1}, ek_{B,2}), (C_{Q,1}, C_{Q,2}), h) \in \mathcal{L}_H$ for $(K_{B,1}, K_{Q,1}, K_{Q,2})$, where $K_{B,1} = \mathsf{DeCap}_{dk_{Q,1}}(C_{B,1})$ and $K_{Q,2} = \mathsf{wDeCap}_{dk_{B,2}}(C_{Q,2})$, and $\mathcal{S}$ poses $C_{Q,1}$ to the decryption oracle and the oracle outputs $K_{Q,1}$, then $\mathcal{S}$ sets $SK = h$;
   (b) else if $Q = B$ and there exists a tuple $(K_{P,1}, K_{B,1}, K_{B,2}, U_P, U_B, ek_P, ek^*, (C_{P,1}, ek_{P,2}), (C_{B,1}, C_{B,2}), h) \in \mathcal{L}_H$ for $(K_{P,1},$

[6] $H_1(r_{A,1}, dk_{A,1})$ is not registered in $\mathcal{L}_{H_1}$. However, $\mathcal{A}$ does not pose $\mathsf{SessionStateReveal}(\mathsf{sid}^*)$ by the security definition; thus, $\mathcal{A}$ cannot know information about $r_{A,1}$. Thus, $\mathcal{A}$ cannot distinguish the real experiment from the simulation by such queries.

$K_{B,1}, K_{B,2})$, where $K_{B,1} = \mathsf{DeCap}_{dk_{P,1}}(C_{B,1})$ and $K_{B,2} = \mathsf{wDeCap}_{dk_{P,2}}(C_{B,2})$, and $\mathcal{S}$ poses $C_{P,1}$ to the decryption oracle and the oracle outputs $K_{P,1}$, then $\mathcal{S}$ sets $SK = h$;

(c) else if there exists a tuple $(K_{P,1}, K_{Q,1}, K_{Q,2}, U_P, U_Q, ek_{P,1}, ek_{Q,1}, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}), h) \in \mathcal{L}_{H_2}$ for $(K_{P,1}, K_{Q,1}, K_{Q,2})$, $K_{P,1} = \mathsf{DeCap}_{dk_{Q,1}}(C_{P,1})$, $K_{Q,1} = \mathsf{DeCap}_{dk_{P,1}}(C_{Q,1})$ and $K_{Q,2} = \mathsf{DeCap}_{dk_{P,2}}(C_{Q,2})$, then $\mathcal{S}$ sets $SK = h$;

(d) otherwise, $\mathcal{S}$ chooses $SK \in \{0,1\}^\kappa$ randomly.

Finally, $\mathcal{S}$ records $(\Pi, U_P, U_Q, (C_{P,1}, ek_{P,2}), (C_{Q,1}, C_{Q,2}))$ as the completed session and SK in the list $\mathcal{L}_{SK}$.

6. $\mathsf{SessionKeyReveal}(\mathsf{sid})$:

   (a) If the session sid is not completed, $\mathcal{S}$ returns an error message;

   (b) else if sid is recorded in the list $\mathcal{L}_{SK}$, then $\mathcal{S}$ returns the recorded value SK;

   (c) otherwise, $\mathcal{S}$ returns a random value $SK \in \{0,1\}^\kappa$ and records it in the list $\mathcal{L}_{SK}$.

7. $\mathsf{SessionStateReveal}(\mathsf{sid})$: $\mathcal{S}$ responds the ephemeral secret key and intermediate computation results of sid as the definition. If the owner of sid is U_B, $\mathcal{S}$ poses ciphertexts received by U_B to the decryption oracle and can simulate all intermediate computation results. Note that the $\mathsf{SessionStateReveal}$ query is not posed to the test session from the freshness definition.

8. $\mathsf{Corrupt}(U_P)$: $\mathcal{S}$ responds the static secret key of U_P as the definition.

9. $\mathsf{Test}(\mathsf{sid})$: If sid is not i-th session of U_A, then $\mathcal{S}$ aborts with failure. Otherwise, $\mathcal{S}$ responds to the query as the definition.

10. If $\mathcal{A}$ outputs a guess b', $\mathcal{S}$ aborts with failure.

A.2.1 Analysis.

The simulation for $\mathcal{S}$ is perfect except with negligible probability. The probability that $\mathcal{A}$ selects the session as the test session sid^* is at least $\frac{1}{N^2 \ell}$.

Under the event Suc^*, $\mathcal{A}$ poses correctly formed $K_{P,1}, K_{Q,1}, K_{Q,2}$ to H_2. Therefore, $\mathcal{S}$ is successful and does not abort.

Thus, $\mathcal{S}$ is successful with non-negligible probability.

A.3 Other Events

A.3.1 Event $E_2 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_2, the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the ephemeral secret key of sid^* is given to $\mathcal{A}$. Thus, $\mathcal{A}$ cannot obtain any information about $dk_{A,1}$ except negligible guessing probability, since H_1 is the random oracle. Hence, $\mathcal{S}$ performs the reduction same as in the case of event $E_1 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

A.3.2 Event $E_3 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_3, the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the static secret key of U_B is given to $\mathcal{A}$. Thus, $\mathcal{A}$ cannot obtain any information about $dk_{B,1}$ except negligible guessing probability, since H_1 is the random oracle. Hence, $\mathcal{S}$ performs the reduction same as in the case of event $E_1 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

A.3.3 Event $E_4 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_4, the test session sid^* has no matching session $\overline{\mathsf{sid}}^*$, the ephemeral secret key of sid^* is given to $\mathcal{A}$. Thus, $\mathcal{A}$ cannot obtain any information about $dk_{B,1}$ except negligible guessing probability, since H_1 is the random oracle. Hence, $\mathcal{S}$ performs the reduction same as in the case of event $E_2 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

A.3.4 Event $E_5 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_5, the test session sid^* has the matching session $\overline{\mathsf{sid}}^*$, both static secret keys of U_A and U_B are given to $\mathcal{A}$. OW-CPA adversary $\mathcal{S}$ embeds ek^* to $ek_{A,2}$ and C^* to $C_{B,2}$. Then, $\mathcal{S}$ obtains K^* by the hash list $\mathcal{L}_{H_2}$.

A.3.5 Event $E_6 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_6, the test session sid^* has the matching session $\overline{\mathsf{sid}}^*$, both the ephemeral secret keys of sid^* and $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$. Then, $\mathcal{A}$ cannot obtain any information about $dk_{A,1}$ except negligible guessing probability because H_1 is the random oracle. Hence, $\mathcal{S}$ performs the reduction same as in the case of event $E_2 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

A.3.6 Event $E_7 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_7, the test session sid^* has the matching session $\overline{\mathsf{sid}}^*$, the static secret keys of U_A and the ephemeral secret key of $\overline{\mathsf{sid}}^*$ are given to $\mathcal{A}$. Then, $\mathcal{A}$ cannot obtain any information about $r_{A,1}$ except negligible guessing probability because H_1 is the random oracle. Hence, $\mathcal{S}$ performs the reduction same as in the case of event $E_1 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

A.3.7 Event $E_8 \wedge \overline{\mathrm{AskS}} \wedge \mathrm{Suc}^*$.

In the event E_8, the test session sid^* has the matching session $\overline{\mathsf{sid}}^*$, the static secret keys of U_B and the ephemeral secret key of sid^* are given to $\mathcal{A}$. $\mathcal{S}$ embeds ek^* to $ek_{A,1}$ and C^* to $C_{B,1}$. Then, $\mathcal{S}$ obtains K^* by the hash list $\mathcal{L}_{H_2}$.

□