Tightly-Secure Key-Encapsulation Mechanism in the Quantum Random Oracle Model

Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa

NTT Secure Platform Laboratories
3-9-11, Midori-cho Musashino-shi, Tokyo 180-8585 Japan
{saito.tsunekazu, xagawa.keita, yamakawa.takashi}@lab.ntt.co.jp

Abstract. We give a tighter security reduction for a conversion from a weakly-secure public-key encryption scheme to an IND-CCA-secure key-encapsulation mechanism scheme in the quantum random oracle model. To the best of our knowledge, previous reductions are non-tight as the security levels of the obtained schemes are degraded to at most *half or quater* of the original security level (Boneh, Dagdelen, Fischlin, Lehmann, Schafner, and Zhandry (CRYPTO 2012), Targhi and Unruh (TCC 2016-B), and Hofheinz, Hövelmanns, and Kiltz (TCC 2017)).

keywords: Tight security, chosen-ciphertext security, post-quantum cryptography, KEM.

1 Introduction

1.1 Background

Indistinguishability against chosen ciphertext attack (IND-CCA-security) is considered to be a *de facto* standard security notion of public key encryption (PKE) and key encapsulation mechanism (KEM). For constructing efficient IND-CCA-secure PKE/KEM, an idealized model called the random oracle model (ROM) [BR93] is often used. In the ROM, a hash function is idealized to be a publicly accessible oracle that simulates a truly random function. There are many known generic constructions of efficient IND-CCA-secure PKE/KEM in the ROM; Bellare-Rogaway (BR) [BR93], OAEP [BR95,FOPS04], REACT [OP01], GEM [CHJ+02], Fujisaki-Okamoto (FO) [FO99,FO13], etc. KEM variants of these constructions were studied by Dent [Den03], which is summarized in Figure 9 in Appendix A.

Quantum Random Oracle Model. Though the ROM has been widely used to heuristically analyze securities of cryptographic primitives, Boneh et al. [BDF+11] pointed out that the ROM is rather problematic when considering a *quantum* adversary. The problem is that in the ROM, an adversary is only given a classical access to a random oracle. Since a random oracle is an idealization of a real hash function, a quantum adversary should be able to quantumly compute it. Based on this observation, they proposed a new model called the quantum random oracle model (QROM) where an adversary can quantumly access to a random oracle. Since many technique used in the ROM including adaptive programmability or extractability cannot be directly translated into the ones in the QROM, proving securities in the QROM often requires different techniques from proofs in the ROM (see [BDF+11] for more details). Nonetheless, some of the above mentioned IND-CCA-secure PKE/KEM in the ROM (and their variants) can be shown also secure in the QROM: Boneh et al. [BDF+11] proved that a variant of Bellare-Rogaway is IND-CCA-secure in the QROM. Targhi and Unruh [TU16] proposed variants of the Fujisaki-Okamoto and OAEP, and proved that they are IND-CCA-secure in the QROM.

Tight Security. When proving a security of a primitive P under a hardness of a problem S, we usually construct a reduction algorithm $\mathcal{R}$ that uses an adversary $\mathcal{A}$ against the security of P as a subroutine, and solves the problem S. Let (T, ϵ) and (T', ϵ') denote running times and success probabilities of $\mathcal{A}$ and $\mathcal{R}$, respectively. We say that a reduction is tight if we have $T' \approx T$ and $\epsilon' \approx \epsilon$. Tight security is desirable since it ensures that breaking a security of P is as hard as solving an underlying hard problem S. Conversely, if a security reduction is non-tight, we cannot immediately conclude that breaking the security of a primitive P is hard even if an underlying problem S is hard. For example, Menezes [Men12] shows an example of a provably secure primitive with non-tight security that is insecure with a realistic parameter setting. Therefore tight security is important to ensure the real security of a primitive.

From that perspective, the above mentioned IND-CCA-secure PKE/KEM in the QROM does not serve as a satisfactory solution for constructing post-quantum IND-CCA-secure PKE/KEM because they are non-tight. To clarify this, we give more details on these results below, where (T, ϵ) and (T', ϵ') denote running times and success probabilities of an adversary and a reduction algorithm, respectively, q_{H} denotes the number of random oracle queries, and t_{RO} denotes a time needed to simulate one evaluation of a random oracle (for further explanation of t_{RO}, see subsection 2.2).

- Boneh et al. [BDF⁺11] proved that a KEM variant of Bellare-Rogaway is IND-CCA-secure in the QROM based on a one-way trapdoor function. [1] According to their security proof, we have $T' \approx T + q_{\mathsf{H}} \cdot t_F + (q_{\mathsf{H}} + q_{\mathsf{Dec}}) \cdot t_{RO}$ and $\epsilon' \approx \epsilon^2$ where t_F denotes a time needed for evaluating an underlying one-way trapdoor function and q_{Dec} denotes the number of decryption queries.
- Targhi and Unruh [TU16] proposed a variant of Fujisaki-Okamoto, and proved that their construction is secure in the QROM assuming OW-CPA security of an underlying PKE scheme. According to their security proof, we have $T' \geq T + O(q_{\mathsf{H}}^2)$ and $\epsilon' \approx \epsilon^4$. We note that Hofheinz et al. [HHK17] subsequently gave a modular analysis for the conversion, but they did not improve the tightness.
- Targhi and Unruh [TU16] proposed a variant of OAEP, and proved that their construction is secure in the QROM assuming a partial domain one-way function. According to their security proof, we have $T' \geq T + O(q_{\mathsf{H}}^2)$ and $\epsilon' \approx \epsilon^8$.

As seen above, known constructions of IND-CCA-secure PKE/KEM in the QROM incur at least quadratic security loss. This means that even if we have a 128-bit security for an underlying primitive, a resulting primitive may only have 64-bit security, which is clearly not enough. If one wants to ensure 128-bit security for these construction, one has to base the primitive on at least 256-bit secure underlying primitive, which incurs significant blowup of parameters. Therefore, to obtain an efficient construction of post-quantum IND-CCA-secure PKE/KEM, we need a construction with tighter security reduction, which especially does not incur a quadratic security loss.

1.2 Our Contributions

In this paper, we give a construction of an IND-CCA-secure KEM based on a deterministic PKE (DPKE) scheme that satisfies a newly introduced security notion which we call the disjoint simulatability. Our security reduction is much tighter than those of existing constructions of an IND-CCA-secure PKE schemes. and especially it does not incur quadratic security loss. By using the same notation as in the previous subsection, we have $T' \approx T + q_{\mathsf{H}} \cdot \mathsf{Time}(\mathsf{Enc}) + (q_{\mathsf{H}} + q_{\mathsf{Dec}}) \cdot t_{RO}$ and $\epsilon' \approx \epsilon$ where $\mathsf{Time}(\mathsf{Enc})$ denotes a time needed for encryption of an underlying DPKE scheme. We note that $\mathsf{Time}(\mathsf{Enc})$ is a fixed polynomial of the security parameter, and thus we believe that this blowup is much less significant than quadratic (or 4-th/8-th power) blowup for ϵ as in the previous constructions.

Moreover, we construct some DPKE schemes whose disjoint simulatabilities are tightly reduced to some post-quantum assumptions like learning with errors (LWE) and some other assumptions related to NTRU, the McEliece PKE, and the Niederreiter PKE. As a result, we obtain the first IND-CCA-secure KEMs that does not incur a quadratic security loss in the QROM based on these assumptions. We also construct a disjoint simulatable DPKE scheme from any IND-CPA-secure PKE scheme on exponentially large message space with quadratic security loss. This gives a construction of an IND-CCA-secure KEM based on an IND-CPA-secure PKE scheme on exponentially large message space with quadratic (rather than 4-th power as in previous works) security loss. Our results are summarized in Figure 1.

We implement an instantiation based on NTRU-HRSS [HRSS17] over a desktop PC and a RasPi. Assuming that NTRU-HRSS is disjoint simulatable, the obtained KEM is CCA secure in the QROM. See section 5.

1.3 Technical Overview

Here, we give a technical overview of our result.

[1] Correctly speaking, they proved that a hybrid encryption variant of the Bellare-Rogaway PKE scheme is IND-CCA-secure in the QROM based on a one-way trapdoor function plus a CCA-secure symmetric-key encryption scheme. Their proof is easily turned into the proof for the KEM variant of Bellare-Rogaway conversion.

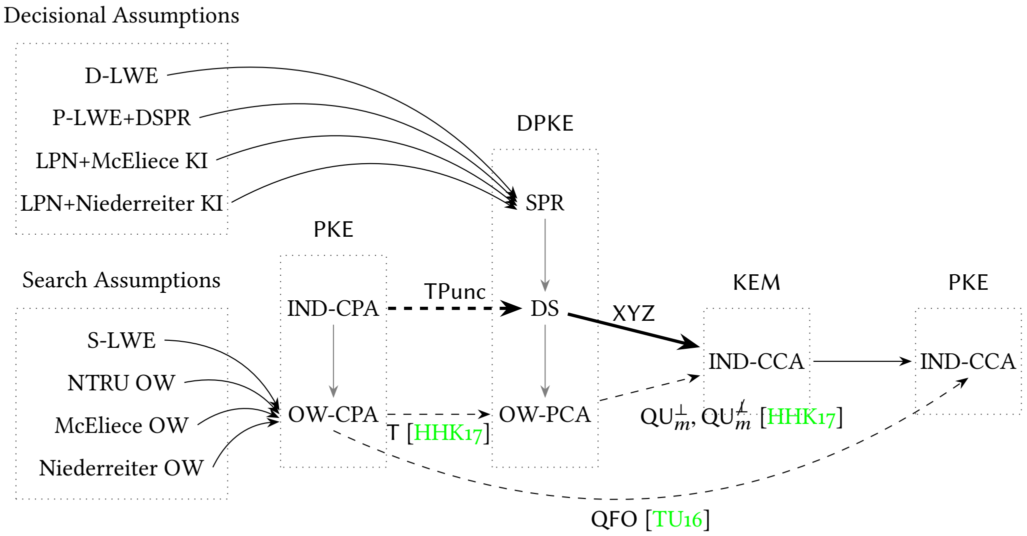

Fig. 1: Transformations among PKE, DPKE and KEM in the QROM: D-LWE and S-LWE denote the decisional and search learning-with-errors assumptions, P-LWE denotes the Poly-LWE assumption, DSPR denotes the decisional small polynomial ratio assumption, LPN denotes the learning-parity-with-noise assumption, McEliece KI and Niederreiter KI denote the McEliece-key-indistinguishability and Niederreiter-key-indistinguishability assumptions, respectively, NTRU OW, McEliece OW, and Niederreiter OW denote one-wayness of the NTRU, McEliece encryption, and Niederreiter encryption, respectively; OW-CPA, OW-PCA, IND-CPA, and IND-CCA denotes onewayness under chosen-plaintext attacks, onewayness under plaintext-checking attacks, indistinguishability under chosen-plaintext attacks, and indistinguishability under chosen-ciphertext attacks, respectively; SPR denotes the sparse pseudorandomness, DS denotes the disjoint simulatability; Solid arrows indicate quantum tight reductions, dashed arrows indicate quantum non-tight reductions, thin arrows indicate existing reductions, thick arrows indicates our new reductions, and gray arrows indicate trivial implications.

Disjoint Simulatability and Sparse Pseudorandomness. Let $\mathcal{D}_{\mathcal{M}}$ be a distribution over a message space $\mathcal{M}$. We say that a DPKE scheme is $\mathcal{D}_{\mathcal{M}}$-disjoint simulatable if a ciphertext of a message that is distributed according to $\mathcal{D}_{\mathcal{M}}$ can be simulated by a simulator that does not know a message, and simulated ciphertext is invalid (i.e., out of a range of an encryption algorithm) with overwhelming probability. As an intermediate step to construct a disjoint simulatable DPKE scheme, we consider another security notion which we call sparse pseudorandomness, and show that this is a sufficient condition for disjoint simulatability. We say that a DPKE scheme is $\mathcal{D}_{\mathcal{M}}$-sparse pseudorandom if a ciphertext of a message that is distributed according to $\mathcal{D}_{\mathcal{M}}$ is pseudorandom and a range of encryption algorithm is sparse in a ciphertext space. The $\mathcal{D}_{\mathcal{M}}$-sparse pseudorandomness implies the $\mathcal{D}_{\mathcal{M}}$-disjoint simulatability because if the sparse pseudorandomness is satisfied, then a simulator that simply outputs a random element of a ciphertext space suffices for the disjoint simulatability [2].

Instantiations of Disjoint Simulatable DPKE. We construct DPKE schemes based on the concepts of the GPV trapdoor function for LWE [GPV08], NTRU [HPS98], the McEliece PKE [McE78], and the Niederreiter PKE [Nie86], and prove that they are sparse pseudorandom (and thus disjoint simulatable) w.r.t. a certain message distribution under the LWE assumption, or other related assumptions to an underlying PKE scheme. Moreover, the reductions are tight. See subsection 3.3 and section C for details of instantiations from concrete assumptions

We also construct a disjoint simulatable DPKE scheme based on any IND-CPA-secure PKE scheme with exponentially large message space in the QROM. Unfortunately, this reduction is not tight, and incur a square security loss. See subsection 3.4 for details.

Previous Construction: BR-KEM. Before describing our construction, we review the construction and security proof of BR-KEM, which was proven IND-CCA-secure in the QROM by Boneh et al. [BDF+11] because our construction is based on their idea. BR-KEM is a construction of an IND-CCA-secure KEM from a one-way trapdoor function with an efficiently recognizable range [3]. For compatibility with ours, we treat a one-way trapdoor function as a perfectly correct OW-CPA-secure DPKE scheme by considering a function and an inversion to be an encryption and a decryption respectively. Let (Gen, Enc, Dec) denote algorithms of an underlying DPKE scheme. Then BR-KEM = $(\mathsf{Gen}_{\mathsf{BR}}, \mathsf{Enc}_{\mathsf{BR}}, \mathsf{Dec}_{\mathsf{BR}})$ is described as follows:

- $\mathsf{Gen}_{\mathsf{BR}}$ is exactly the same as Gen.
- $\mathsf{Enc}_{\mathsf{BR}}$, given a public key ek as an input, chooses a randomness m uniformly from a message space, and computes a ciphertext $C := \mathsf{Enc}(ek, m)$ and a key $K := \mathsf{H}(m)$ where H is a hash function modeled as a random oracle, and outputs (C, K).
- $\mathsf{Dec}_{\mathsf{BR}}$, given a ciphertext C and a decryption key dk as an input, checks if C is in the valid ciphertext space and returns $\perp$ if not. Otherwise it computes $K := \mathsf{H}(\mathsf{Dec}(C))$ and returns K.

In the security proof in the QROM, we first replace a random oracle H with $\mathsf{H}_q \circ \mathsf{Enc}(ek,)$ where H_q is another random oracle that is not given to an adversary. Since $\mathsf{Enc}(ek, \cdot)$ is injective due to its perfect correctness, $\mathsf{H}_q \circ \mathsf{Enc}(ek, \cdot)$ still works as a random oracle from the view of an adversary. After this replacement, we notice that decryption oracle can be simulated by using H_q without the help of a decryption key, because we have $\mathsf{H}(\mathsf{Dec}(c)) = \mathsf{H}_q \circ \mathsf{Enc}(ek, \mathsf{Dec}(c)) = \mathsf{H}_q(c)$. For proving IND-CCA security, we have to prove that $\mathsf{H}_q(c^*)$ is pseudorandom from the view of an adversary. If we were in a classical world, then this can be proven quite easily: the only way for an adversary to obtain any information of $\mathsf{H}_q(c^*)$ is to query m^* such that $c^* = \mathsf{Enc}(ek, m^*)$, in which case the adversary breaks the OW-CPA security of an underlying DPKE scheme. In a quantum world, things do not go so easily because even if an adversary queries a quantum state whose magnitude on m^* is large, a reduction algorithm cannot notice that immediately. Nonetheless, by using the One-way to hiding (OW2H) lemma proven by Unruh [Unr15] (Lemma 2.1), we can show that the advantage for an adversary to distinguish $\mathsf{H}_q(c^*)$ from a truly random string is at most a square root of the probability that measurement of a randomly chosen adversary's query to H is equal to m^*. Hence, we can reduce the IND-CCA security of BR-KEM to the OW-CPA security of the underlying DPKE scheme with a quadratic security loss. On the other hand, to avoid the quadratic security loss, it seems that we have to avoid the usage of the OW2H lemma because the lemma inherently incurs a quadratic security loss.

[2] Actually, we have to additionally assume that a ciphertext space is efficiently sampleable.

[3] The efficient recognizability of a range was not explicitly assumed in [BDF+11], but that is actually needed for their proof.

Our Conversion, XYZ. In the above proof, we used the fact that the only way for an adversary to obtain any information of $\mathsf{H}_q(c^*)$ is to query m^* to H such that $c^* = \mathsf{Enc}(ek, m^*)$. Our key idea is based on the observation that if such m^* does not exist, i.e., c^* is out of the range of $\mathsf{Enc}(ek, \cdot)$, then it is information theoretically impossible for an adversary to obtain any information of $\mathsf{H}_q(c^*)$. Indeed, though c^* is in the range of $\mathsf{Enc}(ek, \cdot)$ in the real game, if we choose an encryption randomness m according to a distribution $\mathcal{D}_{\mathcal{M}}$, then we can replace c^* with a simulated ciphertext that is out of the range of $\mathsf{Enc}(ek, \cdot)$ by using the $\mathcal{D}_{\mathcal{M}}$-disjoint simulatability. After replacing c^* with a simulated one, we can information theoretically bound an adversary's advantage and need not use the OW2H lemma. It seems that this simply resolves the problem, and we obtain an IND-CCA-secure KEM without a quadratic security loss. However, another problem arises here: a valid ciphertext space of a disjoint simulatable DPKE scheme is inherently not efficiently recognizable (otherwise it is easy to distinguish real and simulated ciphertexts) whereas the simulation of decryption algorithm has to first verify if a given ciphertext is valid or not. To resolve the problem, we modify the decryption algorithm so that if a ciphertext is invalid, then it returns a random value rather than $\perp$. In the security proof of BR-KEM, a decryption oracle is simulated just by evaluating a random oracle H_q for a ciphertext, and this enables a reduction algorithm to simulate a decryption oracle for both valid and invalid ciphertexts even though it cannot determine if a given ciphertext is valid. Hence we can reduce the IND-CCA-security of the resulting KEM without using the OW2H lemma, and thus without a quadratic security loss.

Curiously, this conversion is essentially same as $\mathsf{U}_m^{\not\perp}$ in [HHK17]. This means that we can remove an "additional" hash from $\mathsf{QU}_m^{\not\perp}$ assuming stronger underlying DPKE.

1.4 Related Work

In a concurrent and independent work, Jiang et al. [JZC+17] propose a new construction of an IND-CCA-secure KEM based on a OW-CPA-secure PKE scheme with quadratic security loss. On the other hand, they does not give any construction of an IND-CCA-secure KEM without quadratic security loss.

1.5 Version Notes

We have revised our paper throughly so that some presentations in the current version are different from the previous versions. We summarize differences below.

- In the previous versions, we defined a security notion called PR-CPA for DPKE, and our conversion XYZ was presented as a conversion from a PR-CPA-secure DPKE scheme to an IND-CCA-secure KEM. In the current version, instead of defining the PR-CPA-security, we define the disjoint simulatability because this notion is simpler and captures an essential property needed for our conversion. We note that the disjoint simulatability implies the PR-CPA-security (see section D), and all instantiations of a PR-CPA-secure DPKE scheme presented in the previous versions are actually also disjoint simulatable under the same assumption.
- In the previous versions, a reduction algorithm was not given a random oracle, and it simulated a random oracle by using a PRF, which made our proofs somehow involved. In the current version, we assume that a reduction is given a random oracle access. We remark that this is not a modification of the model since a reduction can simulate a random oracle in several ways. (See subsection 2.2 for more details.)
- In the previous versions, we gave the conversion THalf that converts an IND-CPA-secure PKE scheme to a PR-CPA-secure DPKE scheme. In the conversion THalf, the message space of the resulting scheme is a half of a massage space of an underlying scheme. We notice that actually we need not make a message space half, and puncturing by one message (say, 0) suffices. Based on this idea, we give another conversion TPunc instead of THalf, and prove that the resulting scheme is disjoint simulatable (which also implies the PR-CPA-security).

2 Preliminaries

2.1 Notation:

A security parameter is denoted by κ. We use the standard O-notations, O, Θ, Ω, and ω. The abbreviations DPT and PPT stand for deterministic polynomial time and probabilistic polynomial time. A function $f(\kappa)$ is said to be *negligible* if $f(\kappa) = \kappa^{-\omega(1)}$. We denote a set of negligible functions by $\mathsf{negl}(\kappa)$. For two finite sets $\mathcal{X}$ and $\mathcal{Y}$, $\mathrm{Map}(\mathcal{X}, \mathcal{Y})$ denotes a set of all functions whose domain is $\mathcal{X}$ and codomain is $\mathcal{Y}$.

For a distribution χ, we often write "$x \leftarrow \chi$", which indicates that we take a sample x from χ. For a finite set S, $U(S)$ denotes the uniform distribution over S. We often write "$x \leftarrow S$" instead of $x \leftarrow U(S)$. For a set S and a deterministic algorithm A, $\mathsf{A}(S)$ denotes the set $\{\mathsf{A}(x) \mid x \in S\}$.

If inp is a string, then "$\mathrm{out} \leftarrow \mathsf{A}(\mathsf{inp})$" denotes the output of algorithm A when run on input inp. If A is deterministic, then out is a fixed value and we write "$\mathrm{out} := \mathsf{A}(\mathsf{inp})$"; We also use the notation "$\mathrm{out} := \mathsf{A}(\mathsf{inp}; r)$" to make the randomness r explicit.

For the Boolean statement P, $\mathrm{boole}(P)$ denotes the bit that is 1 if P is true, and otherwise 0. For example, $\mathrm{boole}(b' \overset{?}{=} b)$ is 1 if and only if $b' = b$.

2.2 Quantum Computation

We refer to [NC00] for basic of quantum computation.

Quantum Random Oracle Model. Roughly speaking, the quantum random oracle model (QROM) is an idealized model where a hash function is modeled as a publicly and quantumly accessible random oracle. See [BDF+11] for more detailed description of the model.

Lemmas. We review some useful lemmas regarding to the quantum random oracles. The first one is called the oneway-to-hiding (OW2H) lemma, which is proven by Unruh [Unr15, Lemma 6.2]. Roughly speaking, the lemma states that if any quantum adversary issuing at most q queries to a quantum random oracle H can distinguish $(x, \mathsf{H}(x))$ from (x, y), where y is chosen uniformly at random, then we can find x by measuring one of the adversary's query though it occurs a quadratic security loss. The lemma of the following form is taken from [HHK17].

Lemma 2.1 (Algorithmic Oneway to Hiding [Unr15,HHK17]). *Let $\mathsf{H} : \mathcal{X} \to \mathcal{Y}$ be a quantum random oracle, let $\mathcal{A}$ be an adversary issuing at most q queries to H that on input $(x, y) \in \mathcal{X} \times \mathcal{Y}$ outputs either $0/1$. For all (probabilistic) algorithms F whose input space is $\mathcal{X} \times \mathcal{Y}$ and which do not make any hash queries to H, we have*

$$\begin{aligned} &\left| \begin{array}{l} \Pr[\mathcal{A}^{\mathsf{H}}(\mathrm{inp}) \to 1 \mid x \leftarrow \mathcal{X}; \mathrm{inp} \leftarrow \mathsf{F}(x, \mathsf{H}(x))] \\ \quad - \Pr[\mathcal{A}^{\mathsf{H}}(\mathrm{inp}) \to 1 \mid (x, y) \leftarrow \mathcal{X} \times \mathcal{Y}; \mathrm{inp} \leftarrow \mathsf{F}(x, y)] \end{array} \right| \\ &\le 2q \cdot \sqrt{\Pr[\mathsf{EXT}^{\mathcal{A},\mathsf{H}}(\mathrm{inp}) \to x \mid (x, y) \leftarrow \mathcal{X} \times \mathcal{Y}; \mathrm{inp} \leftarrow \mathsf{F}(x, y)]}, \end{aligned}$$

where EXT picks $i \leftarrow \{1, \ldots, q\}$, runs $\mathcal{A}^{\mathsf{H}}(\mathrm{inp})$ until i-th query $|\hat{x}\rangle$ to H, and returns $x' := \mathsf{Measure}(|\hat{x}\rangle)$ (when $\mathcal{A}$ makes less than i queries, EXT outputs $\perp \notin \mathcal{X}$).

(Unruh's original statement is recovered by letting F as identity function.)

The second one claims that a random oracle can be used as a pseudorandom function even in the quantum setting.

Lemma 2.2. *Let ℓ be an integer. Let $\mathsf{H}\colon \{0, 1\}^{\ell} \times \mathcal{X} \to \mathcal{Y}$ and $\mathsf{H}'\colon \mathcal{X} \to \mathcal{Y}$ be two independent random oracles. If an unbounded time quantum adversary $\mathcal{A}$ makes a query to H at most q_{H} times, then we have*

$$\left| \Pr[\mathcal{A}^{\mathsf{H},\mathsf{H}(s,\cdot)}() \to 1 \mid s \leftarrow \{0, 1\}^{\ell}] - \Pr[\mathcal{A}^{\mathsf{H},\mathsf{H}'}() \to 1] \right| \le q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}}$$

where all oracle accesses of $\mathcal{A}$ can be quantum.

Though this seems to be a folklore, we give a proof of this lemma in section B for completeness. [4]

Simulation of Random Oracle. In the original quantum random oracle model introduced by Boneh et al. [BDF+11], they do not allow a reduction algorithm to access to a random oracle, and it has to simulate a random oracle by itself. On the other hand, in this paper, we give a random oracle access to a reduction algorithm. We remark that this is just a convention and not a modification of the model since we can simulate a random oracle against quantum adversaries in several ways.

[4] Jiang et al. [JZC+17] also gave a proof of an essentially same lemma.

1. The first way is a simulation by a $2q$-wise independent hash function, where q denotes the number of random oracle queries by an adversary, as introduced by Zhandry [Zha12b]. The simulation is perfect, that is, any adversary cannot distinguish the real QRO with the simulated one. A drawback of this simulation is a $O(q^2)$ blowup for a running time of a reduction algorithm, since it has to compute a $2q$-wise independent hash function for each random oracle query.
2. The second way is a simulation by a quantumly secure PRF as used in [BDF+11]. If we use this simulation, then the blowup of a running time of a reduction algorithm is $O(q \cdot t_{\mathsf{PRF}})$ where t_{PRF} is a time needed for evaluating a PRF, which is usually much smaller than $O(q^2)$. On the other hand, we have to additionally assume the existence of a quantumly secure PRF, which is known to exist if a quantumly secure one-way function exists [Zha12a].
3. The third way is a simulation by a real hash function like SHA-2 and think that this is a "random oracle." Since we adopt the QROM, we idealize a real hash function as a random oracle in the construction of primitives. Thus, it may be natural to assume the same thing even in *a reduction*, that is, the reduction algorithm implement the random oracle by a concrete hash function. If we use this simulation, then the blowup of a running time of a reduction algorithm is $O(q \cdot t_{\mathsf{hash}})$ where t_{hash} denotes a time to evaluate a hash function. This gives a tightest reduction at the expense of additional idealization of a hash function. We note that a similar convention is also used by Kiltz et al. [KLS17].

We use t_{RO} to denote a time needed to simulate a random oracle. We have $t_{\mathsf{RO}} = O(q)$, t_{PRF}, or t_{hash}, if we use the first, second, or third way, respectively. We note that in the proof of quantum variants of Fujisaki-Okamoto and OAEP [TU16,HHK17], we have to simulate a random oracle in the 1st way, because a simulator has to "invert" a random oracle in a simulation.

2.3 Public-Key Encryption

The model for PKE schemes is summarized as follows:

Definition 2.1. *A PKE scheme* PKE *consists of the following triple of polynomial-time algorithms* (Gen, Enc, Dec)*.*

- $\mathsf{Gen}(1^\kappa; r_g) \to (ek, dk)$*: a key-generation algorithm which on input 1^κ, where κ is the security parameter, outputs a pair of keys (ek, dk). ek and dk are called encryption key and decryption key, respectively.*
- $\mathsf{Enc}(ek, m; r_e) \to c$*: an encryption algorithm which takes as input encryption key ek and message $m \in \mathcal{M}$, outputs ciphertext $c \in C$.*
- $\mathsf{Dec}(dk, c) \to m/\bot$*: a decryption algorithm which takes as input decryption key dk and ciphertext c, outputs message $m \in \mathcal{M}$ or a rejection symbol $\bot \notin \mathcal{M}$.*

Definition 2.2. *We say a PKE scheme* PKE *is deterministic if* Enc *is deterministic. We denote DPKE to stand for a deterministic public key encryption.*

Definition 2.3 (Correctness). *We say* PKE = (Gen, Enc, Dec) *has* perfect correctness *if for any (ek, dk) generated by* Gen *and for any $m \in \mathcal{M}$ we have that*

$$\Pr[\mathsf{Dec}(dk, c) = m \mid c \leftarrow \mathsf{Enc}(ek, m)] = 1.$$

Security: Here, we define one-wayness against chosen-plaintext attack (OW-CPA), indistinguishability against chosen-plaintext attack (IND-CPA), and indistinguishability against chosen-ciphertext attack (IND-CCA) for PKE.

Definition 2.4 (Security notions for PKE). *For any adversary $\mathcal{A}$, we define its OW-CPA, IND-CPA, and IND-CCA advantages against a PKE scheme* PKE = (Gen, Enc, Dec) *as follows:*

$$\begin{aligned}\mathsf{Adv}^{\text{ow-cpa}}_{\mathcal{A},\mathsf{PKE}}(\kappa) &:= \Pr[\mathsf{Expt}^{\text{ow-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa) = 1],\\ \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa) &:= \left|2\Pr[\mathsf{Expt}^{\text{ind-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa) = 1] - 1\right|,\\ \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},\mathcal{A}}(\kappa) &:= \left|2\Pr[\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{PKE},\mathcal{A}}(\kappa) = 1] - 1\right|,\end{aligned}$$

where $\mathsf{Expt}^{\text{ow-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa)$, $\mathsf{Expt}^{\text{ind-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa)$, *and* $\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{PKE},\mathcal{A}}(\kappa)$ *are experiments described in Figure 2. For ATK $\in$ {OW-CPA, IND-CPA, IND-CCA} we say that* PKE *is ATK-secure if* $\mathsf{Adv}^{\text{atk}}_{\mathcal{A},\mathsf{PKE}}(\kappa)$ *is negligible for any PPT adversary $\mathcal{A}$.*

```
$\mathsf{Expt}^{\text{ow-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa)$
    $(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $m^* \leftarrow \mathcal{M}$
    $c^* \leftarrow \mathsf{Enc}(ek, m^*)$
    $m' \leftarrow \mathcal{A}(ek, c^*)$
    return $\mathsf{boole}(m' \stackrel{?}{=} \mathsf{Dec}(dk, c^*))$
```

```
$\mathsf{Expt}^{\text{ind-cpa}}_{\mathsf{PKE},\mathcal{A}}(\kappa)$
    $b \leftarrow \{0, 1\}$
    $(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $(m_0, m_1, st) \leftarrow \mathcal{A}_1(ek)$
    $c^* \leftarrow \mathsf{Enc}(ek, m_b)$
    $b' \leftarrow \mathcal{A}_2(c^*, st)$
    return $\mathsf{boole}(b' \stackrel{?}{=} b)$
```

```
$\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{PKE},\mathcal{A}}(\kappa)$
    $b \leftarrow \{0, 1\}$
    $(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $(m_0, m_1, st) \leftarrow \mathcal{A}_1^{\textsc{Dec}_\perp(\cdot)}(ek)$
    $c^* \leftarrow \mathsf{Enc}(ek, m_b)$
    $b' \leftarrow \mathcal{A}_2^{\textsc{Dec}_{c^*}(\cdot)}(c^*, st)$
    return $\mathsf{boole}(b' \stackrel{?}{=} b)$
```

```
$\textsc{Dec}_a(c)$
    if $c = a$, return $\perp$
    $m := \mathsf{Dec}(dk, c)$
    return $m$
```

Fig. 2: Games for PKE schemes

2.4 Key Encapsulation

The model for KEM schemes is summarized as follows:

Definition 2.5. *A KEM scheme* KEM *consists of the following triple of polynomial-time algorithms* (Gen, Encaps, Decaps)*:*

- $\mathsf{Gen}(1^\kappa; r_g) \to (ek, dk)$*: a key-generation algorithm which on input* 1^κ*, where* κ *is the security parameter, outputs a pair of keys* (ek, dk)*.* ek *and* dk *are called encapsulation key and decapsulation key, respectively.*
- $\mathsf{Encaps}(ek; r_e) \to (c, K)$*: an encapsulation algorithm which takes as input encapsulation key* ek*, outputs ciphertext* $c \in \mathcal{C}$ *and key* $K \in \mathcal{K}$*.*
- $\mathsf{Decaps}(dk, c) \to K/\perp$*: a decapsulation algorithm which takes as input decapsulation key* dk *and ciphertext* c*, outputs key* K *or a rejection symbol* $\perp \notin \mathcal{K}$*.*

Definition 2.6 (Correctness). *We say* $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ *has* perfect correctness *if for any* (ek, dk) *generated by* Gen*, we have that*

$$\Pr[\mathsf{Decaps}(dk, c) = K : (c, K) \leftarrow \mathsf{Encaps}(ek)] = 1.$$

Security: We define indistinguishability against chosen-plaintext and chosen-ciphertext attacks (denoted by IND-CPA and IND-CCA) for KEM, respectively.

Definition 2.7. *For any adversary* $\mathcal{A}$*, we define its IND-CPA and IND-CCA advantages against a KEM scheme* $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ *as follows:*

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{KEM},\mathcal{A}}(\kappa) := \left| 2\Pr[\mathsf{Expt}^{\text{ind-cpa}}_{\mathsf{KEM},\mathcal{A}}(\kappa) = 1] - 1 \right|,$$
$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa) := \left| 2\Pr[\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa) = 1] - 1 \right|,$$

where $\mathsf{Expt}^{\text{ind-cpa}}_{\mathsf{KEM},\mathcal{A}}(\kappa)$ *and* $\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa)$ *are experiments described in Figure 3.*

For ATK $\in$ *{IND-CPA, IND-CCA}, we say that* KEM *is ATK-secure if* $\mathsf{Adv}^{\text{atk}}_{\mathcal{A},\mathsf{PKE}}(\kappa)$ *is negligible for any PPT adversary* $\mathcal{A}$*.*

2.5 eXtendable-Output Functions

An eXtendable-Output Function (XOF) is a function on input bit strings in which the output can be extended to arbitrary desired length. An XOF is denoted by $\mathsf{XOF}(X, L)$, where X is the input bit string and L is the desired output length. We modeled the XOF as a quantumly-accessible random oracle. We will employ SHAKE256, standardized as an XOF by NIST (See FIPS 202).

```
$\mathsf{Expt}^{\text{ind-cpa}}_{\mathsf{KEM},\mathcal{A}}(\kappa)$
$b \leftarrow \{0,1\}$
$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
$(c^*, K_0^*) \leftarrow \mathsf{Encaps}(ek);$
$K_1^* \leftarrow \mathcal{K}$
$b' \leftarrow \mathcal{A}(ek, c^*, K_b^*)$
return $\mathsf{boole}(b' \overset{?}{=} b)$
```

```
$\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa)$
$b \leftarrow \{0,1\}$
$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
$(c^*, K_0^*) \leftarrow \mathsf{Encaps}(ek);$
$K_1^* \leftarrow \mathcal{K}$
$b' \leftarrow \mathcal{A}^{\textsc{Dec}_{c^*}(\cdot)}(ek, c^*, K_b^*)$
return $\mathsf{boole}(b' \overset{?}{=} b)$
```

```
$\textsc{Dec}_{c^*}(c)$
if $c = c^*$, return $\perp$
$K := \mathsf{Decaps}(dk, c)$
return $K$
```

Fig. 3: Games for KEM schemes

2.6 Assumptions

Preliminaries: Let $\rho_s(x) = \exp(-\pi\|x\|^2/s^2)$ for $x \in \mathbb{R}^n$ be a Gaussian function scaled by a factor s. For any real $s > 0$ and lattice Λ, we define the discrete Gaussian distribution $D_{\Lambda,s}$ over Λ with parameter s by

$$D_{\Lambda,s}(x) = \rho_s(x)/\rho_s(\Lambda) \text{ for } x \in \Lambda,$$

where $\rho_s(\Lambda) = \sum_{x\in\Lambda} \rho_s(x)$. The following norm bound is useful.

Lemma 2.3 (Adapted version of [MR07, Lemma 4.4]). *For $\sigma = \omega(\sqrt{\log(n)})$, it holds that*

$$\Pr_{e\leftarrow D_{\mathbb{Z}^n,\sigma}}[\|e\| > \sigma\sqrt{n}] \leq 2^{-n+1}.$$

Preliminaries: We review the assumptions for lattice-based PKEs. The most basic one is the learning-with-errors (LWE) assumption [Reg09], which is a generalized version of the learning-parity-with-noise assumption [BFKL93,KSS10].

Definition 2.8 (LWE assumption in matrix form). *For all κ, let $n = n(\kappa)$ and $q = q(\kappa)$ be integers and let χ be a distribution over $\mathbb{Z}$.*

The decisional learning-with-errors (LWE) assumption $\mathsf{LWE}_{n,q}$ states that for any $m = \mathsf{poly}(\kappa)$ it is computationally hard to distinguish the following two distributions:

- *$A, sA + e$, where $A \leftarrow \mathbb{Z}_q^{n\times m}$, $s \leftarrow \mathbb{Z}_q^n$, and $e \leftarrow \chi^m$*
- *A, u, where $A \leftarrow \mathbb{Z}_q^{n\times m}$ and $u \leftarrow \mathbb{Z}_q^m$.*

We also review its polynomial version [LPR10,BV11]. We here use the Hermite-normal form of the assumption [ACPS09,LPR10,BV11], where secret s is chosen form the noise distribution.

Definition 2.9 (Poly-LWE assumption – Hermite normal form). *For all κ, let $\Phi(x) = \Phi_\kappa(x) \in \mathbb{Z}[x]$ be a polynomial of degree $n = n(\kappa)$, let $q = q(\kappa)$ be an integer, let $R := \mathbb{Z}[x]/(\Phi(x))$ and $R_q := \mathbb{Z}_q[x]/(\Phi(x))$, and let χ denote a distribution over the ring R.*

The decisional polynomial learning-with-errors (Poly-LWE) assumption $\mathsf{PolyLWE}_{\Phi,q,\chi}$ states that for any $\ell =$ $\mathsf{poly}(\kappa)$ it is hard to distinguish the following two distributions:

- *$\{(a_i, a_i s + e_i)\}_{i=1,\dots,\ell}$, where $a_i \leftarrow R_q$, $s, e_i \leftarrow \chi$*
- *$\{(a_i, u_i)\}_{i=1,\dots,\ell}$, where $a_i, u_i \leftarrow R_q$.*

Next, we recall the decisional small polynomial ratio (DSPR) assumption defined by López-Alt, Tromer, and Vaikuntanathan [LTV12]. We here employ an adapted version of the DSPR assumption.

Definition 2.10 (DSPR assumption). *For all κ, let $\Phi(x) = \Phi_\kappa(x) \in \mathbb{Z}[x]$ be a polynomial of degree $n = n(\kappa)$, let $q = q(\kappa)$ be a positive integer, let $R := \mathbb{Z}[x]/(\Phi(x))$ and $R_q := \mathbb{Z}_q[x]/(\Phi(x))$, and let χ denote a distribution over the ring R.*

The decisional small polynomial ratio (DSPR) assumption $\mathsf{DSPR}_{\Phi,q,\chi_g,\chi_f}$ says that it is hard to distinguish the following two distributions:

- *a polynomial $h := g \cdot f^{-1} \in R_q$, where $g \leftarrow \chi_g$ and $f \leftarrow \chi_f$.*
- *a polynomial $u \leftarrow R_q$.*

Remark 2.1. Stehlé and Steinfeld [SS11] showed that $\mathsf{DSPR}_{\Phi,q,\chi}$ is statistically hard if n is a power of two, $\Phi(x) = x^n + 1$, and $\chi_g = \chi_f = D_{\mathbb{Z}^n,r}$ for $r > \sqrt{q} \cdot \mathsf{poly}(\kappa)$.

3 Disjoint Simulatability of Deterministic PKE

Here, we define a new security notion, *disjoint simulatability*, for DPKE. We also define another security notion called *sparse pseudorandomness* and prove that it implies the disjoint simulatability. Then we give some instantiations of sparse pseudorandom (and thus disjoint simulatable) deterministic PKE schemes based on the LWE assumption or various assumptions related to NTRU, the McEliece PKE, and the Niederreiter PKE with tight reductions. We also construct a disjoint simulatable DPKE schemes from any IND-CPA-secure PKE scheme with a sufficiently large message space in the QROM, though the reduction is non-tight.

3.1 Definition

We define a new security notion, *disjoint simulatability*, for DPKE. Intuitively, a deterministic PKE scheme is disjoint simulatable if there exists a simulator that is only given a public key and generates a "fake ciphertext" that is indistinguishable from a real ciphertext of a random message. Moreover, we require that a fake ciphertext falls in a valid ciphertext space with negligible probability. The formal definition is as follows.

Definition 3.1 (Disjoint simulatability). *Let $\mathcal{D}_{\mathcal{M}}$ denote an efficiently sampleable distribution on a set $\mathcal{M}$. A deterministic PKE scheme* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with plaintext and ciphertext spaces $\mathcal{M}$ and C is $\mathcal{D}_{\mathcal{M}}$-*disjoint simulatable *if there exists a PPT algorithm $\mathcal{S}$ that satisfies the following.*

- *(Statistical disjointness:)*

$$\mathsf{Disj}_{\mathsf{PKE},\mathcal{S}}(\kappa) := \max_{(ek,dk)\in\mathsf{Gen}(1^\kappa;\mathcal{R})} \Pr[c \in \mathsf{Enc}(ek,\mathcal{M}) \mid c \leftarrow \mathcal{S}(ek)]$$

  is negligible, where $\mathcal{R}$ denotes a randomness space for Gen.
- *(Ciphertext-indistinguishability:) For any PPT adversary $\mathcal{A}$,*

$$\mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A},\mathcal{S}}(\kappa) := \left| \begin{array}{r} \Pr\left[\mathcal{A}(ek,c^*) \to 1 \mid (ek,dk) \leftarrow \mathsf{Gen}(1^\kappa); m^* \leftarrow \mathcal{D}_{\mathcal{M}}; c^* := \mathsf{Enc}(ek,m^*)\right] \\ -\Pr\left[\mathcal{A}(ek,c^*) \to 1 \mid (ek,dk) \leftarrow \mathsf{Gen}(1^\kappa); c^* \leftarrow \mathcal{S}(ek)\right] \end{array} \right|$$

  is negligible.

3.2 Sufficient Condition: Sparse Pseudorandomness

Here, we define another security notion for DPKE called *sparse pseudorandomness*, which is a sufficient condition to be disjoint simulatable. Intuitively, a deterministic PKE scheme is sparse pseudorandom if valid ciphertexts are sparse in a ciphertext sparse and pseudorandom when a message is randomly chosen. In other words, an encryption algorithm can be seen as a pseudorandom generator (PRG). The formal definition is as follows.

Definition 3.2 (Sparse pseudorandomness). *Let $\mathcal{D}_{\mathcal{M}}$ denote an efficiently sampleable distribution on a set $\mathcal{M}$. A deterministic PKE scheme* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with plaintext and ciphertext spaces $\mathcal{M}$ and C is $\mathcal{D}_{\mathcal{M}}$-*sparse pseudorandom *if the following two properties are satisfied.*

- *(Sparseness:)*

$$\mathsf{Sparse}_{\mathsf{PKE}}(\kappa) := \max_{(ek,dk)\in\mathsf{Gen}(1^\kappa;\mathcal{R})} \frac{|\mathsf{Enc}(ek,\mathcal{M})|}{|C|}$$

  is negligible where $\mathcal{R}$ denotes a randomness space for Gen.
- *(Pseudorandomness:) For any PPT adversary $\mathcal{A}$,*

$$\mathsf{Adv}^{\text{pr}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A}}(\kappa) := \left| \begin{array}{r} \Pr\left[\mathcal{A}(ek,c^*) \to 1 \mid (ek,dk) \leftarrow \mathsf{Gen}(1^\kappa); m^* \leftarrow \mathcal{D}_{\mathcal{M}}; c^* := \mathsf{Enc}(ek,m^*)\right] \\ -\Pr\left[\mathcal{A}(ek,c^*) \to 1 \mid c^* \leftarrow C\right] \end{array} \right|$$

  is negligible.

Then we prove that the sparse pseudorandomness implies the disjoint simulatability if a ciphertext space is efficiently sampleable.

Lemma 3.1. *If a deterministic PKE scheme* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with plaintext and ciphertext spaces $\mathcal{M}$ and C is $\mathcal{D}_{\mathcal{M}}$-sparse pseudorandom and C is efficiently sampleable, then* PKE *is also $\mathcal{D}_{\mathcal{M}}$-disjoint simulatable. In particular, there exists a PPT simulator $\mathcal{S}$ such that* $\mathsf{Disj}_{\mathsf{PKE},\mathcal{S}}(\kappa) = \mathsf{Sparse}_{\mathsf{PKE}}(\kappa)$ *and* $\mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A},\mathcal{S}}(\kappa) = \mathsf{Adv}^{\text{pr}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A}}(\kappa)$.

Proof. Let $\mathcal{S}$ be an algorithm that outputs a random element of C. Then it is clear that we have $\mathsf{Disj}_{\mathsf{PKE},\mathcal{S}}(\kappa) = \mathsf{Sparse}_{\mathsf{PKE}}(\kappa)$ and $\mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A},\mathcal{S}}(\kappa) = \mathsf{Adv}^{\text{pr}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A}}(\kappa)$. □

3.3 Instantiations

Here, we give examples of a DPKE scheme that is disjoint simulatable. In particular, we construct DPKE schemes that has the sparse pseudorandomness based on the LWE assumption or some other assumptions related to NTRU. (We further consruct them based on the McEliece PKE and the Niederreiter PKE in appendix.) We remark that the reductions are tight. By combining those with Lemma 3.1, we obtain disjoint simulatable DPKE schemes based on any of these assumptions with tight securities.

LWE-based DPKE: We review the GPV trapdoor function for LWE [GPV08,Pei09,MP12]. The LWE assumption (in matrix form) states that $(A, sA + e)$ and (A, u) are computationally indistinguishable, where $A \leftarrow \mathbb{Z}_q^{n\times m}$, $s \leftarrow \mathbb{Z}_q^n$, $e \leftarrow \chi^m$, and $u \leftarrow \mathbb{Z}_q^m$. The GPV trapdoor function for LWE exploited that if we have a "short" matrix T satisfying $AT \equiv O \bmod q$, we can retrieve s and e from $c = sA + e$. The trapdoor T for A is generated by an algorithm TrapGen:

Theorem 3.1 ([Ajt99,AP11]). *For any positive integers n and $q \geq 3$, any $\delta > 0$ and $m \geq (2 + \delta)n \lg q$, there is a probabilistic polynomial-time algorithm* TrapGen *that outputs a pair $T \in \mathbb{Z}^{m\times m}$ and $A \in \mathbb{Z}_q^{n\times m}$ such that: the distribution of A is within negligible statistical distance of uniform over $\mathbb{Z}_q^{n\times m}$, T is non-singular (over the rationals), $\|t_i\| \leq L = O(m \lg m)$ for every column vector t_i of T, and $AT \equiv O \pmod q$.*

Let us construct a DPKE scheme PKE = (Gen, Enc, Dec) as follows:

Parameters: We require several parameters; the dimension $n = n(\kappa)$, the modulus $q = q(\kappa)$, and $m = m(\kappa)$. We also employ $L = O(m \lg m)$, $\sigma = \omega(\sqrt{\lg n})$, $\beta = \sigma\sqrt{n}$. We require that $\beta L < q/2$ and $q^m \gg q^n \cdot (2\beta + 1)^m$.
- The plaintext space $\mathcal{M} := \mathbb{Z}_q^n \times B_m(\beta)$, where $B_m(\beta) := \{e \in \mathbb{Z}^m \mid \|e\| \leq \beta\}$.
- The sampler $\mathcal{D}_\mathcal{M}$ samples $s \leftarrow \mathbb{Z}_q^n$ and $e \leftarrow D_{\mathbb{Z}^m,\sigma}$ conditioned on that $\|e\| \leq \beta$.
- The ciphertext space $C := \mathbb{Z}_q^m$

Key Generation: $\mathsf{Gen}(1^\kappa)$ invokes $\mathsf{TrapGen}(1^n, 1^m, q)$ and obtains $A \in \mathbb{Z}_q^{n\times m}$ and $T \in \mathbb{Z}^{m\times m}$. It outputs $ek = A$ and $dk = (A, T)$.

Encryption: $\mathsf{Enc}(ek, (s, e))$ outputs $c = sA + e \bmod q$.

Decryption: $\mathsf{Dec}(dk, c)$ computes $e = (c \cdot T \bmod q) \cdot T^{-1}$ and $s = (c - e) \cdot A^+ \bmod q$, where $A^+ := A^\top \cdot (A \cdot A^\top) \in \mathbb{Z}_q^{m\times n}$, the left inverse of A.

The properties of PKE are summarized as follows:

Perfect Correctness: We know $c \cdot T \equiv sAT + eT \equiv eT \pmod q$. If $\|eT\|_\infty < q/2$, then $c \cdot T \bmod q = eT \in \mathbb{Z}^m$ holds and e is recovered by $e = (c\cdot T \bmod q)\cdot T^{-1}$. Once correct e is obtained, s is recovered by $(c-e)\cdot A^+ \in \mathbb{Z}_q^n$. The condition $\|eT\|_\infty < q/2$ is satisfied because $\|eT\|_\infty \leq \max_i \|e\| \cdot \|t_i\| \leq \beta L < q/2$, where t_i is the column vectors of T.

Sparseness: $|C| = q^m$ and $|\mathsf{Enc}(ek, \mathcal{M})| \leq \mathcal{M} = |\mathbb{Z}_q^n \times B_m(\beta)| \leq q^n \cdot (2\beta + 1)^m$. Sparseness follows from the fact $q^m \gg q^n \cdot (2\beta + 1)^m$.

Pseudorandomness: We consider the following hybrid games:
- (The original game 1:) The adversary is given (A, c^*), where $(A, T) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $(s, e) \leftarrow \mathcal{D}_\mathcal{M}$, and $c^* := sA + e \bmod q$.
- (The hybrid game 1:) Let us replace the public key A. We consider (A, c^*), where $A \leftarrow \mathbb{Z}_q^{n\times m}$, $(s, e) \leftarrow \mathcal{D}_\mathcal{M}$, and $c^* := sA + e \bmod q$. This change is justified by Theorem 3.1.
- (The hybrid game 2:) Let us replace the sampler $\mathcal{D}_\mathcal{M}$. We consider (A, c^*), where $A \leftarrow \mathbb{Z}_q^{n\times m}$, $(s, e) \leftarrow U(\mathbb{Z}_q^n) \times D_{\mathbb{Z}^m,\sigma}$, and $c^* := sA + e \bmod q$. This replacement is justified by Lemma 2.3.
- (The hybrid game 3:) We next replace the ciphertext c^*. We consider (A, c^*), where $A \leftarrow \mathbb{Z}_q^{n\times m}$ and $c^* \leftarrow \mathbb{Z}_q^m$. This game is computationally indistinguishable from the previous game under the LWE assumption $\mathsf{LWE}_{n,q,D_{\mathbb{Z},\sigma}}$.
- (The original game 2:) We replace the public key A. We consider (A, c^*), where $(A, T) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ and $c^* := sA + e \bmod q$. This change is justified by Theorem 3.1.

Remark 3.1. For simplicity, we employ the simple version of the GPV trapdoor function for LWE. Further improvements are available, e.g., [MP12, Section 5].

NTRU-based DPKE: We next review the original version of NTRUEncrypt [HPS98]. Let $\Phi(x) = x^n - 1 \in \mathbb{Z}[x]$, let $p < q$ be positive integers with $\gcd(p, q) = 1$, and let $R := \mathbb{Z}[x]/(\Phi(x))$ and $R_q := \mathbb{Z}_q[x]/(\Phi(x))$. We often set $p = 3$ and $q = 2^k$ for some k. Let $\mathcal{T}$ be a set of ternary-coefficient polynomials in R, that is, $\mathcal{T} := \{t = \sum_{i=0}^{n-1} t_i x^i \in R \mid t_i \in \{-1, 0, +1\}\}$. Let $\mathcal{L}_f, \mathcal{L}_g, \mathcal{L}_r, \mathcal{L}_m \subseteq \mathcal{T}$. The public key is $h = g/f$, where $f \leftarrow \mathcal{L}_f, g \leftarrow \mathcal{L}_g$ with f has inverses in R_p and R_q. The the ciphertext of $m \in \mathcal{L}_m$ with randomness $r \in \mathcal{L}_r$ is $c = prh + m$. Roughly speaking, we can retrieve m if we know f; $cf = prg + mf \in R_q$ and it holds in R.

Parameters: We require that $\|prg + mf \bmod q\|_\infty < q/2$ for any g, f, m, r in their domains, where, for $t = \sum_{i=0}^{n-1} t_i x^i \in R$, we define $\|t\|_\infty := \max_i |t_i|$. For simplicity, we assume that $\mathcal{L}_m = \mathcal{L}_r$.

- The plaintext space is $\mathcal{M} := \mathcal{L}_m \times \mathcal{L}_r$.
- The sampler $\mathcal{D}_\mathcal{M}$ samples $(m, r) \leftarrow \mathcal{L}_m \times \mathcal{L}_r$.
- The ciphertext space is $C := R_q$.

Key Generation: Gen() chooses $g \leftarrow \mathcal{L}_g$ and $f \leftarrow \mathcal{L}_f$ until f is invertible in R_q and R_p. It outputs $ek = h = g/f \in R_q$ and $dk = (h, f)$.

Encryption: $\mathsf{Enc}(ek, (m, r))$ outputs $c = prh + m \in R_q$.

Decryption: $\mathsf{Dec}(sk, c)$ computes $m := (fc \bmod q) \cdot f^{-1} \bmod p$ and $r := (c - m) \cdot (ph)^{-1} \bmod q$.

The properties of this DPKE are summarized as follows:

Perfect correctness: We have $fc \equiv prg + mf \pmod{q}$. Since $\|prg + mf \bmod q\|_\infty < q/2$ from our requirement, we have $(fc \bmod q) = prg + mf \in R$. Hence, we have $(fc \bmod q) \cdot f^{-1} \equiv (prg + mf) \cdot f^{-1} \equiv m \pmod{p}$ as we wanted. r is also recovered because $(c - m) \cdot (ph)^{-1} \equiv prh \cdot (ph)^{-1} \equiv r \pmod{q}$.

Sparseness: Sparseness follows from $|C| = q^n \gg 3^{2n} = |\mathcal{T}^2| \geq |\mathcal{L}_m \times \mathcal{L}_r| = |\mathsf{Enc}(ek, \mathcal{M})|$.

Pseudorandomness: What we want to show is

$$(h, c = prh + m) \approx_c (h, u),$$

where $h = g/f$ is a public key with $f \leftarrow \mathcal{L}_f, g \leftarrow \mathcal{L}_g$ with condition f has inverses R_p and R_q, $(m, r) \leftarrow \mathcal{L}_m \times \mathcal{L}_r$, and $u \leftarrow R_q$. Let $\chi_g := U(\mathcal{L}_g)$ and $\chi_f := U(\mathcal{L}_f \cap R_p^* \cap R_q^*)$, where R_k^* for $k \in \{p, q\}$ denotes $\{f \in R \mid f \text{ has an inverse in } R_k\}$. Let $\chi := U(\mathcal{L}_m) = U(\mathcal{L}_r)$.

- We first replace $h = g/f$ with random h', which is justified by the DSPR assumption $\mathsf{DSPR}_{\Phi,q,\chi_f,\chi_g}$.
- We next replace $c = rh' + e$ with random c', which is justified by the Poly-LWE assumption $\mathsf{PolyLWE}_{\Phi,q,\chi}$.
- We then go backward by replacing random h' with $h = g/f$, which is justified by the DSPR assumption $\mathsf{DSPR}_{\Phi,q,\chi_f,\chi_g}$ again.

3.4 Generic conversion from IND-CPA-secure PKE

Here, we show that any perfectly-correct IND-CPA-secure PKE whose plaintext space is sufficiently large can be converted into a disjoint-simulatable DPKE scheme in the quantum random oracle model. We note that the conversion is *non-tight.*

Intuitively, we replace randomness of an underlying IND-CPA-secure PKE scheme with a hash value of a message similarly to the conversion T given in [HHK17] (which is in turn based on the Fujisaki-Okamoto conversion). The difference from the conversion T is that we "puncture" a message space by 0 [5]. That is, if a message space of an underlying IND-CPA-secure PKE scheme is $\mathcal{M}$, then a message space of the resulting scheme is $\mathcal{M}' := \mathcal{M} \setminus \{0\}$. In this meaning, we call our conversion TPunc. We give the concrete description of the conversion TPunc below.

Let $\mathcal{M}$ and $\mathcal{R}$ be the message and randomness spaces of PKE, respectively, and let $\mathcal{M}' := \mathcal{M} \setminus \{0\}$. Then the resulting DPKE scheme $\mathsf{PKE}_1 = \mathsf{TPunc}[\mathsf{PKE}, \mathsf{G}]$ is described in Figure 4 where $\mathsf{G} \colon \mathcal{M} \to \mathcal{R}$ denotes a random oracle. Here, we remark that the message space of PKE_1 is restricted to $\mathcal{M}' := \mathcal{M} \setminus \{0\}$. The security of PKE_1 is stated as follows.

Theorem 3.2 (Security of TPunc). *Let $\mathcal{S}$ be the algorithm described in Figure 4. If PKE is perfectly-correct, then we have $\mathsf{Disj}_{\mathsf{PKE}_1,\mathcal{S}}(\kappa) = 0$. Moreover, for any quantum adversary $\mathcal{A}$ against PKE_1 issuing at most q_G quantum queries to G, there exist quantum adversaries $\mathcal{B}$ and C against IND-CPA security of PKE such that*

$$\mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE}_1,U_{\mathcal{M}'},\mathcal{A},\mathcal{S}}(\kappa) \leq 2q_\mathsf{G}\sqrt{\mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{PKE},\mathcal{B}}(\kappa) + \frac{2}{|\mathcal{M}|}} + \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{PKE},C}(\kappa)$$

where $U_{\mathcal{M}'}$ denotes the uniform distribution on $\mathcal{M}'$, and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(C) \approx \mathsf{Time}(\mathcal{A}) + q_\mathsf{G} \cdot t_{\mathsf{RO}}$.

Proof. It is obvious that we have $\mathsf{Disj}_{\mathsf{PKE}_1,\mathcal{S}}(\kappa) = 0$ since PKE is perfectly correct.

[5] We assume that $0 \in \mathcal{M}$. Actually, we can replace 0 with an arbitrary message in $\mathcal{M}$. We assume that $0 \in \mathcal{M}$ for notational simplicity.

```
$\mathsf{Gen}_1(1^\kappa)$
$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
return $(ek, dk)$
```

```
$\mathsf{Enc}_1(ek, m)$, where $m \in \mathcal{M}'$
$r := \mathsf{G}(m)$
$c := \mathsf{Enc}(ek, m; r)$
return $c$
```

```
$\mathsf{Dec}_1(dk, c)$
$m := \mathsf{Dec}(dk, c)$
if $m \notin \mathcal{M}'$ return $\perp$
else return $m$
```

```
$\mathcal{S}(ek)$
$r \leftarrow \mathcal{R}$
$c := \mathsf{Enc}(ek, 0; r)$
return $c$
```

Fig. 4: $\mathsf{PKE}_1 = (\mathsf{Gen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1) = \mathsf{TPunc}[\mathsf{PKE}, \mathsf{G}]$ with simulator $\mathcal{S}$.

Table 1: Summary of Games for the Security Proof of Theorem 3.2

Game	m^*	r^*	c^*	justification
Game_0	$\mathcal{M}'$	$\mathsf{G}(m^*)$	$\mathsf{Enc}(ek, m^*; r^*) = \mathsf{Enc}_1(ek, m^*)$	
Game_1	$\mathcal{M}'$	r^*	$\mathsf{Enc}(ek, m^*; r^*)$	OW-CPA security of PKE and the OW2H lemma
Game_2	0	r^*	$\mathsf{Enc}(ek, 0; r^*) = \mathcal{S}(ek)$	IND-CPA security of PKE

To prove the rest of the theorem, we consider the following sequence of games.

Game_0*:* This game is defined as follows:

$$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa); m^* \leftarrow \mathcal{M}'; r^* \leftarrow \mathsf{G}(m^*); c^* := \mathsf{Enc}(ek, m^*; r^*); b' \leftarrow \mathcal{A}^{\mathsf{G}(\cdot)}(ek, c^*); \textbf{return } b'.$$

Game_1*:* This game is the same as Game_0 except that a randomness to generate a challenge ciphertext is freshly generated:

$$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa); m^* \leftarrow \mathcal{M}'; r^* \leftarrow \mathcal{R}; c^* := \mathsf{Enc}(ek, m^*; r^*); b' \leftarrow \mathcal{A}^{\mathsf{G}(\cdot)}(ek, c^*); \textbf{return } b'.$$

Game_2*:* This game is the same as Game_1 except that a challenge ciphertext is generated by $\mathsf{Enc}(ek, m^*; r^*)$, where $m^* := 0$ rather than $m^* \leftarrow \mathcal{M}'$:

$$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa); r^* \leftarrow \mathcal{R}; c^* := \mathsf{Enc}(ek, 0; r^*); b' \leftarrow \mathcal{A}^{\mathsf{G}(\cdot)}(ek, c^*); \textbf{return } b'.$$

This completes the descriptions of games. It is easy to see that we have $\mathrm{Adv}^{\text{ds-ind}}_{\mathsf{PKE}_1, U_{\mathcal{M}'}, \mathcal{A}, \mathcal{S}}(\kappa) = |\Pr[\mathsf{Game}_0 = 1] - \Pr[\mathsf{Game}_2 = 1]|$. We give an upperbound for this by the following lemmas.

Lemma 3.2. *There exists an adversary $\mathcal{B}$ such that*

$$|\Pr[\mathsf{Game}_0 = 1] - \Pr[\mathsf{Game}_1 = 1]| \leq 2q_{\mathsf{G}}\sqrt{\mathrm{Adv}^{\text{ind-cpa}}_{\mathsf{PKE}, \mathcal{B}}(\kappa) + \frac{2}{|\mathcal{M}|}}$$

and $\mathrm{Time}(\mathcal{B}) \approx \mathrm{Time}(\mathcal{A}) + q_{\mathsf{G}} \cdot t_{\mathsf{RO}}$.

Proof. Let F be an algorithm described in Figure 5. It is easy to see that Game_0 can be restated as

$$m^* \leftarrow \mathcal{M}'; r^* \leftarrow \mathsf{G}(m^*); \mathsf{inp} := \mathsf{F}(ek, m^*; r^*); b' \leftarrow \mathcal{A}^{\mathsf{G}(\cdot)}(\mathsf{inp}); \textbf{return } b'.$$

and Game_1 can be restated as

$$m^* \leftarrow \mathcal{M}'; r^* \leftarrow \mathcal{R}; \mathsf{inp} := \mathsf{F}(ek, m^*; r^*); b' \leftarrow \mathcal{A}^{\mathsf{G}(\cdot)}(\mathsf{inp}); \textbf{return } b'.$$

Then applying the Algorithmic-OW2H lemma (Lemma 2.1) with $\mathcal{X} = \mathcal{M}'$, $\mathcal{Y} = \mathcal{R}$, $x = m^*$, $y = r^*$, and algorithms $\mathcal{A}$ and F, we have

$$|\Pr[\mathsf{Game}_0 = 1] - \Pr[\mathsf{Game}_1 = 1]| \leq 2q_{\mathsf{G}}\sqrt{\Pr[m^* \leftarrow \mathcal{B}^{\mathsf{G}}(ek, c^*)]}.$$

```
$B^{\mathsf{G}}(ek, c^*)$ :
    $\mathsf{inp} := (ek, c^*)$
    $i \leftarrow [q_{\mathsf{H}}]$
    Run $\mathcal{A}^{\mathsf{G}}(\mathsf{inp})$ until $i$-th query $|\hat{x}\rangle$ to $\mathsf{G}$
    if $i >$ number of queries to $\mathsf{G}$, return $\perp$
    else return $x' := \mathsf{Measure}(|\hat{x}\rangle)$
```

```
$\mathsf{F}(m^*, r^*)$
    $(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $c^* := \mathsf{Enc}(ek, m^*; r^*)$
    $\mathsf{inp} := (ek, c^*)$
    return $\mathsf{inp}$
```

Fig. 5: Adversary $\mathcal{B}$ and Algorithm F

```
$\overline{\mathsf{Gen}}(1^\kappa)$
    $(ek', dk') \leftarrow \mathsf{Gen}_1(1^\kappa)$
    $s \leftarrow \{0,1\}^\ell$
    $dk \leftarrow (dk', ek', s)$
    return $(ek', dk)$
```

```
$\overline{\mathsf{Enc}}(ek')$
    $m \leftarrow \mathcal{D}_{\mathcal{M}}$
    $c := \mathsf{Enc}_1(ek', m)$
    $K := \mathsf{H}(m)$
    return $(K, c)$
```

```
$\overline{\mathsf{Dec}}(dk, c)$, where $dk = (dk', ek', s)$
    $m := \mathsf{Dec}_1(dk', c)$
    if $m = \perp$, return $K := \mathsf{H}'(s, c)$
    if $c \neq \mathsf{Enc}_1(ek', m)$, return $K := \mathsf{H}'(s, c)$
    else return $K := \mathsf{H}(m)$
```

Fig. 6: $\mathsf{KEM} := \mathsf{XYZ}[\mathsf{PKE}_1, \mathsf{H}, \mathsf{H}']$.

where $\mathcal{B}^{\mathsf{G}}$ is an algorithm described in Figure 5, $(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa), m^* \leftarrow \mathcal{M}', r^* \leftarrow \mathcal{R}$ and $c^* := \mathsf{Enc}(ek, m^*, r^*)$. Since the statistical distance between uniform distributions on $\mathcal{M}$ and $\mathcal{M}'$ is $\frac{1}{|\mathcal{M}|}$, we have $\Pr[m^* \leftarrow \mathcal{B}^{\mathsf{G}}(ek, c^*)] \leq \mathsf{Adv}^{\text{ow-cpa}}_{\mathsf{PKE},\mathcal{B}}(\kappa) + \frac{1}{|\mathcal{M}|}$ where the probability in the left-hand side is taken as in the above. (Remark that additional $\frac{1}{|\mathcal{M}|}$ appears because m^* is taken from $\mathcal{M}' = \mathcal{M} \setminus \{0\}$ in the left-hand side probability.) Moreover, we have $\mathsf{Adv}^{\text{ow-cpa}}_{\mathsf{PKE},\mathcal{B}}(\kappa) \leq \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{PKE},\mathcal{B}}(\kappa) + \frac{1}{|\mathcal{M}|}$ in general. By combining these inequalities, the lemma is proven,

Lemma 3.3. *There exists an adversary C such that* $|\Pr[\mathsf{Game}_1 = 1] - \Pr[\mathsf{Game}_2 = 1]| \leq \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{PKE},C}(\kappa)$ *and* $\mathsf{Time}(C) \approx \mathsf{Time}(\mathcal{A}) + q_{\mathsf{G}} \cdot t_{\mathsf{RO}}$.

Proof. We construct an adversary C against the IND-CPA security of PKE as follows.

$C^{\mathsf{G}}(ek)$: It chooses $m_0 \leftarrow \mathcal{M}'$ and sets $m_1 := 0$. Then it queries (m_0, m_1) to its challenge oracle and obtains $c^* \leftarrow \mathsf{Enc}(ek, m^*; r^*)$, where m^* is m_b for a random bit b chosen by the challenger. It invokes $b' \leftarrow \mathcal{A}^{\mathsf{G}}(ek, c^*)$, and outputs b'.

This completes the description of C. It is obvious that C perfectly simulates Game_{b+1} depending on the challenge bit $b \in \{0,1\}$. Therefore, we have

$$\begin{aligned}
\mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{PKE},C}(\kappa) &= |2\Pr[b' = b] - 1| \\
&= |(1 - \Pr[b' = 1 \mid b = 0]) + \Pr[b' = 1 \mid b = 1] - 1| \\
&= |1 - \Pr[\mathsf{Game}_1 = 1] + \Pr[\mathsf{Game}_2 = 1] - 1| \\
&= |\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_1 = 1]|
\end{aligned}$$

as we wanted.

By combining the above lemmas, Theorem 3.2 is proven.

4 Conversion from Disjoint Simulatability to IND-CCA

In this section, we give a conversion from a disjoint simulatable DPKE scheme to an IND-CCA-secure KEM. Let $\mathsf{PKE}_1 = (\mathsf{Gen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ be a deterministic PKE scheme and let $\mathsf{H}\colon \mathcal{M} \to \mathcal{K}$ and $\mathsf{H}'\colon \{0,1\}^\ell \times C \to \mathcal{K}$ be random oracles. Our conversion XYZ is described in Figure 6. The security of our conversion can be stated as follows.

Table 2: Summary of Games for the Proof of Theorem 4.1

					Decryption of		
Game	H	c^*	K_0^*	K_1^*	valid c	invalid c	justification
Game_0	$\mathsf{H}(\cdot)$	$\mathsf{Enc}_1(ek', m^*)$	$\mathsf{H}(m^*)$	random	$\mathsf{H}(m)$	$\mathsf{H}'(s, c)$	
Game_1	$\mathsf{H}(\cdot)$	$\mathsf{Enc}_1(ek', m^*)$	$\mathsf{H}(m^*)$	random	$\mathsf{H}(m)$	$\mathsf{H}_q(c)$	Lemma 2.2
$\mathsf{Game}_{1.5}$	$\mathsf{H}'_q(\mathsf{Enc}_1(ek', \cdot))$	$\mathsf{Enc}_1(ek', m^*)$	$\mathsf{H}(m^*)$	random	$\mathsf{H}(m)$	$\mathsf{H}_q(c)$	Perfect correctness
Game_2	$\mathsf{H}_q(\mathsf{Enc}_1(ek', \cdot))$	$\mathsf{Enc}_1(ek', m^*)$	$\mathsf{H}(m^*)$	random	$\mathsf{H}(m)$	$\mathsf{H}_q(c)$	Conceptual
Game_3	$\mathsf{H}_q(\mathsf{Enc}_1(ek', \cdot))$	$\mathsf{Enc}_1(ek', m^*)$	$\mathsf{H}_q(c^*)$	random	$\mathsf{H}_q(c)$	$\mathsf{H}_q(c)$	Perfect correctness
Game_4	$\mathsf{H}_q(\mathsf{Enc}_1(ek', \cdot))$	$\mathcal{S}(ek')$	$\mathsf{H}_q(c^*)$	random	$\mathsf{H}_q(c)$	$\mathsf{H}_q(c)$	DS-IND

Theorem 4.1 (Security of XYZ). *Let* PKE_1 *be a DPKE scheme that satisfies the* $\mathcal{D}_{\mathcal{M}}$*-disjoint simulatability with a simulator* $\mathcal{S}$*. For any IND-CCA quantum adversary* $\mathcal{A}$ *against* KEM *issuing* q_{H} *and* $q_{\mathsf{H}'}$ *random oracle queries to* H *and* H' *and* $q_{\overline{\mathsf{Dec}}}$ *decryption queries, there exist an adversary* $\mathcal{B}$ *against the disjoint simulatability of* PKE_1 *such that*

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa) \leq \mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE}_1,\mathcal{D}_{\mathcal{M}},\mathcal{S},\mathcal{B}}(\kappa) + \mathsf{Disj}_{\mathsf{PKE}_1,\mathcal{S}}(\kappa) + q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}}$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + q_{\mathsf{H}} \cdot \mathsf{Time}(\mathsf{Enc}_1) + (q_{\mathsf{H}} + q_{\mathsf{H}'} + q_{\overline{\mathsf{Dec}}}) \cdot t_{\mathsf{RO}}$.

The proof of Theorem 4.1 follows.

4.1 Security Proof

We use game-hopping proof. The overview of all games is given in Table 2.

Game_0*:* This is the original game, $\mathsf{Expt}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa)$.

Game_1*:* This game is the same as Game_0 except that $\mathsf{H}'(s, c)$ in the decryption oracle is replaced with $\mathsf{H}_q(c)$ where $\mathsf{H}_q : \mathcal{C} \longrightarrow \mathcal{K}$ is another random oracle. We remark that $\mathcal{A}$ is not given a direct access to H_q.

$\mathsf{Game}_{1.5}$*:* This game is the same as Game_1 except that the random oracle $\mathsf{H}(\cdot)$ is simulated by $\mathsf{H}'_q(\mathsf{Enc}_1(ek, \cdot))$ where H'_q is yet another random oracle. We remark that a decryption oracle and generation of K_0^* also use $\mathsf{H}'_q(\mathsf{Enc}_1(ek, \cdot))$ as $\mathsf{H}(\cdot)$ and that $\mathcal{A}$ is not given a direct access to H'_q.

Game_2*:* This game is the same as $\mathsf{Game}_{1.5}$ except that the random oracle $\mathsf{H}(\cdot)$ is simulated by $\mathsf{H}_q(\mathsf{Enc}_1(ek, \cdot))$ instead of $\mathsf{H}'_q(\mathsf{Enc}_1(ek, \cdot))$. We remark that a decryption oracle and generation of K_0^* also use $\mathsf{H}_q(\mathsf{Enc}_1(ek, \cdot))$ as $\mathsf{H}(\cdot)$.

Game_3*:* This game is the same as Game_2 except that K_0^* is set as $H_q(c^*)$ and the decryption oracle always returns $H_q(c)$ as long as $c \neq c^*$. We denote the modified decryption oracle by $\overline{\mathsf{Dec}}'$.

Game_4*:* This game is the same as Game_3 except that c^* is set as $\mathcal{S}(ek')$.

The above completes the descriptions of games. It is clear that we have

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{KEM},\mathcal{A}}(\kappa) = |2\Pr[\mathsf{Game}_0 = 1] - 1|$$

by the definition. We upperbound this by the following lemmas.

Lemma 4.1. *We have*

$$|\Pr[\mathsf{Game}_0 = 1] - \Pr[\mathsf{Game}_1 = 1]| \leq q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}}.$$

Proof. This is obvious from Lemma 2.2. □

Lemma 4.2. *We have*

$$\Pr[\mathsf{Game}_1 = 1] = \Pr[\mathsf{Game}_{1.5} = 1].$$

Proof. Since we assume that PKE_1 has a perfect correctness, $\mathsf{Enc}_1(ek', \cdot)$ is injective. Therefore if $\mathsf{H}'_q(\cdot)$ is a random function, then $\mathsf{H}'_q(\mathsf{Enc}_1(ek, \cdot))$ is also a random function. Remarking that an access to H'_q is not given to $\mathcal{A}$, it causes no difference from the view of $\mathcal{A}$ if we replace $\mathsf{H}(\cdot)$ with $\mathsf{H}'_q(\mathsf{Enc}_1(ek, \cdot))$. □

Lemma 4.3. *We have*

$$\Pr[\mathsf{Game}_{1.5} = 1] = \Pr[\mathsf{Game}_2 = 1].$$

Proof. We call a ciphertext c is valid if we have $\mathsf{Enc}_1(ek', \mathsf{Dec}_1(dk', c)) = c$ and invalid else. We remark that H_q is used only for decrypting an invalid ciphertext c as $\mathsf{H}_q(c)$ in $\mathsf{Game}_{1.5}$. This means that a value of $\mathsf{H}_q(c)$ for a valid c is not used at all in $\mathsf{Game}_{1.5}$. On the other hand, any output of $\mathsf{Enc}_1(ek', \cdot)$ is valid due to the perfect correctness of PKE_1. Since H'_q is only used for evaluating an output of $\mathsf{Enc}(ek', \cdot)$, a value of $\mathsf{H}_q(c)$ for a valid c is not used at all in $\mathsf{Game}_{1.5}$. Hence it causes no difference from the view of $\mathcal{A}$ if we use the same random oracle H_q instead of two independent random oracles H_q and H'_q. □

Lemma 4.4. *We have*

$$\Pr[\mathsf{Game}_2 = 1] = \Pr[\mathsf{Game}_3 = 1].$$

Proof. Since we set $\mathsf{H}(\cdot) := \mathsf{H}_q(\mathsf{Enc}_1(ek', \cdot))$, for any valid c and $m := \mathsf{Dec}_1(dk', c)$, we have $\mathsf{H}(m) = \mathsf{H}_q(\mathsf{Enc}_1(ek', m)) = \mathsf{H}_q(c)$. Therefore responses of the decryption oracle is unchanged. We also have $\mathsf{H}(m^*) = \mathsf{H}_q(c^*)$ due to the similar reason. □

Lemma 4.5. *There exists an adversary $\mathcal{B}$ such that*

$$|\Pr[\mathsf{Game}_3 = 1] - \Pr[\mathsf{Game}_4 = 1]| \le \mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE}_1, \mathcal{D}_{\mathcal{M}}, \mathcal{S}, \mathcal{B}}(\kappa).$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + q_{\mathsf{H}} \cdot \mathsf{Time}(\mathsf{Enc}_1) + (q_{\mathsf{H}} + q_{\mathsf{H}'} + q_{\overline{\mathsf{Dec}}}) \cdot t_{\mathsf{RO}}$.

Proof. We construct an adversary $\mathcal{B}$, which is allowed to access to two random oracles H_q and H', against the disjoint simulatability as follows [6].

$\mathcal{B}^{\mathsf{H}_q, \mathsf{H}'}(ek', c^*)$: It picks $b \leftarrow \{0, 1\}$, sets $K_0^* := \mathsf{H}_q(c^*)$ and $K_1^* \leftarrow \mathcal{K}$, and invokes $b' \leftarrow \mathcal{A}^{\mathsf{H}, \mathsf{H}', \overline{\mathsf{Dec}}'}(ek', c^*, K_b^*)$ where $\mathcal{A}'s$ oracles are simulated as follows.

- $\mathsf{H}(\cdot)$ is simulated by $\mathsf{H}_q(\mathsf{Enc}_1(ek', \cdot))$.
- H' can be simulated because $\mathcal{B}$ has an access to an oracle H'.
- $\overline{\mathsf{Dec}}'(\cdot)$ is simulated by forwarding to $\mathsf{H}_q(\cdot)$.

Then $\mathcal{B}$ returns $\mathsf{bool}(b \overset{?}{=} b')$.

This completes the description of $\mathcal{B}$. It is easy to see that $\mathcal{B}$ perfectly simulates Game_3 if $c^* = \mathsf{Enc}_1(ek, m^*)$, and perfectly simulates Game_4 if $c^* = \mathcal{S}(ek')$. Therefore we have

$$|\Pr[\mathsf{Game}_3 = 1] - \Pr[\mathsf{Game}_4 = 1]| \le \mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE}_1, \mathcal{D}_{\mathcal{M}}, \mathcal{S}, \mathcal{B}}(\kappa)$$

as wanted. Since $\mathcal{B}$ invokes $\mathcal{A}$ once, H is simulated by one evaluation of Enc_1 plus one evaluation of a random oracle, and H' and $\overline{\mathsf{Dec}}'$ are simulated by one evaluation of random oracles, we have $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + q_{\mathsf{H}} \cdot \mathsf{Time}(\mathsf{Enc}_1) + (q_{\mathsf{H}} + q_{\mathsf{H}'} + q_{\overline{\mathsf{Dec}}}) \cdot t_{\mathsf{RO}}$. □

Lemma 4.6. *We have*

$$|2\Pr[\mathsf{Game}_4 = 1] - 1| \le \mathsf{Disj}_{\mathsf{PKE}_1, \mathcal{S}}(\kappa).$$

Proof. Let Bad denotes an event that $c^* \in \mathsf{Enc}_1(ek', \mathcal{M})$ in Game_4. It is easy to see that we have

$$\Pr[\mathsf{Bad}] \le \mathsf{Disj}_{\mathsf{PKE}_1, \mathcal{S}}(\kappa).$$

When Bad does not occurs, i.e., $c^* \notin \mathsf{Enc}_1(ek', \mathcal{M})$, $\mathcal{A}$ obtains no information about $K_0^* = \mathsf{H}_q(c^*)$. This is because queries to H only reveals $\mathsf{H}_q(c)$ for $c \in \mathsf{Enc}_1(ek', \mathcal{M})$, and $\overline{\mathsf{Dec}}'(c)$ returns $\perp$ if $c = c^*$. Therefore we have

$$\Pr[\mathsf{Game}_4 = 1 \mid \overline{\mathsf{Bad}}] = 1/2.$$

[6] We allow a reduction algorithm to access to random oracles. See subsection 2.2 for details.

```
$\mathsf{Gen}(1^\kappa)$
    $g, f \leftarrow \mathcal{T}_+$
    $f_q := [1/f]_{(q,\Phi_n)}$
    $h := [\Phi_1 g f_q]_{\mathfrak{q}}$
    return $dk = f, ek = h$
```

```
$\mathsf{Enc}(h, m), m \in \mathcal{T}$
    $r \leftarrow \mathcal{T}$
    $c := [prh + \mathrm{Lift}_p(m)]_{\mathfrak{q}}$
    return $c$
```

```
$\mathsf{Dec}(f, c)$
    $m' := \left[[cf]_{\mathfrak{q}} f^{-1}\right]_{\mathfrak{p}}$
    return $m'$
```

Fig. 7: $\mathsf{NTRU}_{\mathsf{HRSS17}}$

Combining the above we have

$$\begin{aligned}
&|2\Pr[\mathsf{Game}_4 = 1] - 1| \\
&= \left|\Pr[\mathrm{Bad}](2\Pr[\mathsf{Game}_4 = 1|\mathrm{Bad}] - 1) + \Pr[\overline{\mathrm{Bad}}](2\Pr[\mathsf{Game}_4 = 1|\overline{\mathrm{Bad}}] - 1)\right| \\
&\leq \Pr[\mathrm{Bad}] + \left|2\Pr[\mathsf{Game}_4 = 1|\overline{\mathrm{Bad}}] - 1\right| \\
&\leq \mathsf{Disj}_{\mathsf{PKE}_1,\mathcal{S}}(\kappa)
\end{aligned}$$

as we wanted. □

Combining the above lemmas, Theorem 4.1 is proven.

5 Implementation

We report the implementation results on a desktop PC and on a RasPi, based on the previous implementation of a variant of NTRU [HRSS17].

5.1 NTRU-HRSS

We review a variant of NTRU, which we call $\mathsf{NTRU}_{\mathsf{HRSS17}}$, in [HRSS17].

Let $\Phi_m(x) \in \mathbb{Z}[x]$ be the m-th cyclotomic polynomial. We have $\Phi_1 = x - 1$. If m is prime, then we have $\Phi_m = 1 + x + \cdots + x^{m-1}$. Define $S_n := \mathbb{Z}[x]/(\Phi_n)$ and $R_n := \mathbb{Z}[x]/(x^n - 1)$. For prime n, we have $x^n - 1 = \Phi_1\Phi_n$ and $R_n \simeq S_1 \times S_n$. We define $\mathrm{Lift}_p \colon S_n/(p) \to R_n$ as

$$\mathrm{Lift}_p(v) := \left[\Phi_1[v/\Phi_1]_{(p,\Phi_n)}\right]_{(x^n-1)}.$$

By definition, we have $\mathrm{Lift}_p(v) \equiv 0 \pmod{\Phi_1}$ and $\mathrm{Lift}_p(v) \equiv v \pmod{(p, \Phi_n)}$. Let $\mathfrak{p} = (p, \Phi_n)$ and $\mathfrak{q} = (q, x^n - 1)$. Let

$$\begin{aligned}
\mathcal{T} &:= \{a \in \mathbb{Z}[x] : a = [a]_{\mathfrak{p}}\} = \{a \in \mathbb{Z}[x] : a_i \in (p) \text{ and } \deg(a) < \deg(\Phi_n)\} \\
\mathcal{T}_+ &:= \{a \in \mathcal{T} : \langle xa, a\rangle \geq 0\}.
\end{aligned}$$

The definition of $\mathsf{NTRU}_{\mathsf{HRSS17}}$ is in Figure 7. Notice that all ciphertexts are equivalent to 0 modulo (q, Φ_1), which prevents a trivial distinguishing attack.

Hülsing et al. choose $(n, p, q) = (701, 3, 8192)$: The scheme is perfectly correct and they claimed 128-bit post-quantum security of this parameter set. The implementation of $\mathsf{NTRU}_{\mathsf{HRSS17}}$ and $\mathsf{QFO}^{\perp}[\mathsf{NTRU}_{\mathsf{HRSS17}}, \mathsf{G}, \mathsf{H}, \mathsf{H}']$ is reported in [HRSS17].

Our Modification: We want PKE_1 to be *deterministic*. Hence, we consider a pair of (m, r) as a plaintext and make the decryption algorithm output (m, r) rather than m. The modification $\mathsf{NTRU}_{\mathsf{HRSS17}}{}'$ is summarized in Figure 8.

The properties of this DPKE are summarized as follows:

Perfect Correctness: This follows from the perfect correctness of the original PKE.
Sparseness: This follows from the parameter setting of the original PKE.

```
$\mathsf{Gen}'(1^\kappa) = \mathsf{Gen}$
    $g, f \leftarrow \mathcal{T}_+$
    $f_q := [1/f]_{(q,\Phi_n)}$
    $h := [\Phi_1 g f_q]_{\mathfrak{q}}$
    return $dk = f, ek = h$
```

```
$\mathsf{Enc}'(h, (m, r)), (m, r) \in \mathcal{T}^2$
    $c := [prh + \mathsf{Lift}_p(m)]_{\mathfrak{q}}$
    return $c$
```

```
$\mathsf{Dec}'(f, c)$
    $m' := \left[[cf]_{\mathfrak{q}} f^{-1}\right]_{\mathfrak{p}}$
    $r' := \left[\left[(c - \mathsf{Lift}_p(m')) \cdot (ph)^{-1}\right]_{\mathfrak{q}}\right]_{\mathfrak{p}}$
    return $(m', r')$
```

Fig. 8: Our Modification $\mathsf{NTRU}_{\mathsf{HRSS17}}{}'$

Table 3: Experimental Results: We have $|ek| = 1140$ bytes, $|dk| = 2557$ bytes, and $|c| = 1140$ bytes.

(a) Our Experiments on a PC

	min	med.	avg.	max
Gen_1	1 767	1 778	1 815	2 592
Enc_1	327	329	328	331
Dec_1	958	959	959	1 021
	min	med.	avg.	max
$\overline{\mathsf{Gen}}$	2 565	2 580	2 579	2 601
$\overline{\mathsf{Enc}}$	332	334	333	336
$\overline{\mathsf{Dec}}$	1 280	1 282	1 282	1 286

(b) Our Experiments on a RasPi

	min	med.	avg.	max
Gen_1	33 675	33 685	33 687	45 460
Enc_1	3 085	3 089	3 091	3 121
Dec_1	8 839	8 851	8 850	8 880
	min	med.	avg.	max
$\overline{\mathsf{Gen}}$	49 151	49 169	49 174	49 263
$\overline{\mathsf{Enc}}$	3 200	3 205	3 207	3 232
$\overline{\mathsf{Dec}}$	11 837	11 841	11 843	11 888

Pseudorandomness: We assume that the modified PKE $\mathsf{NTRU}_{\mathsf{HRSS17}}{}'$ satisfies pseudorandomness.

We also implement $\mathsf{XYZ}[\mathsf{NTRU}_{\mathsf{HRSS17}}{}', \mathsf{H}, \mathsf{H}']$, where H and H' are implemented by SHAKE256. We define

$$\mathsf{H}(m, r) := \mathsf{XOF}\big((r, m, 0), 256, 0\big) \text{ and } \mathsf{H}'(s, c) := \mathsf{XOF}\big((c, (s \| 00 \cdots 00), 1), 256\big),$$

where we treat $r \in R_n/(q)$ and the last bit is the context string.

In order to avoid the inversion of polynomials in decapsulation, we add f^{-1} modulo $\mathfrak{p}$ to dk as [HRSS17] did. This requires extra 139 bytes. In addition, we put $(ph)^{-1}$ modulo $\mathfrak{q}$ in dk, which requires extra 1140 bytes. Thus, our decapsulation key is of length 2557 bytes.

5.2 Experimental Results

We preform the experiment with

- one core of an Intel Core i7-6700 at 3.40GHz on a desktop machine with 8GB memory and Ubuntu16.04 and
- a RasPi3 with 32-bit Rasbian.

We use gcc to compile the programs with option -O3. We generates 200 keys and ciphertexts to estimate the running time of key generation, encryption, and decryption. The experimental results are summarized in Table 3a. The Basic and CCA KEM implies $\mathsf{NTRU}_{\mathsf{HRSS17}}{}'$ and $\mathsf{XYZ}[\mathsf{NTRU}_{\mathsf{HRSS17}}{}']$. The results reflect that HRSS17's constant-time implementation and ours. Our conversion adds only small extra amount of costs for hashing in encryption and adds about T_{Enc_1} for re-encrypting in decryption.

We note that our implementations are for reference and we did not optimize them. Further optimizations will speed up the algorithms as [HRSS17] did.

The source code is available at https://info.isl.ntt.co.jp/crypt/eng/archive/contents.html#sxy.

References

ACPS09. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Heidelberg, 2009. 9

Ajt99. Miklós Ajtai. Generating hard instances of the short basis problem. In Jirí Wiedermann, Peter van Emde Boas, and Mogens Nielsen, editors, *ICALP '99*, volume 1644 of *LNCS*, pages 1–9, 1999. 11

AP11. Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48(3):535–553, April 2011. A preliminary versions appeared in *STACS 2009*, 2009. 11

BDF^{+}11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, 2011. 1, 2, 4, 6, 7

BFKL93. Avrim. Blum, Merrick L. Furst, Michale J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In Douglas R. Stinson, editor, *CRYPTO '93*, volume 773 of *LNCS*, pages 278–291. Springer, Heidelberg, 1993. 9, 22

BR93. Mihir Bellare and Phillip Rogaway. Random oracle are practical: A paradigm for designing efficient protocols. In *CCS '93*, pages 62–73. ACM, 1993. 1

BR95. Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT '94*, volume 950 of *LNCS*, pages 92–111. Springer, Heidelberg, 1995. 1

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer, Heidelberg, 2011. 9

CFS01. Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 157–174. Springer, Heidelberg, 2001. 22

CHJ^{+}02. Jean-Sébastien Coron, Helena Handschuh, Marc Joye, Pascal Paillier, David Pointcheval, and Christophe Tymen. GEM: A Generic chosen-ciphertext secure Encryption Method. In Bart Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 175–184. Springer, Heidelberg, 2002. 1

DDMQN12. Nico Döttling, Rafael Dowsley, Jörn Müller-Quade, and Anderson C. A. Nascimento. A CCA2 secure variant of the McEliece cryptosystem. *IEEE Transactions on Information Theory*, 58(10):6672–6680, 2012. A preliminary version appeared in *CT-RSA 2008*, 2008. 22

Den03. Alexander W. Dent. A designer's guide to KEMs. In Kenneth G. Paterson, editor, *IMA 2003*, volume 2898 of *LNCS*, pages 133–151. Springer, Heidelberg, 2003. 1, 21

FGK^{+}13. David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. *Journal of Cryptology*, 26(1):39–74, 2013. Preliminary versions appeared in *PKC 2010*, 2010. 22

FO99. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO '99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, 1999. 1, 21

FO00. Eiichiro Fujisaki and Tatsuaki Okamoto. How to enhance the security of public-key encryption at minimum cost. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 83(1):24–32, 2000. A preliminary version appeared in *PKC '99*, 1999. 21

FO13. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *J. Cryptology*, 26(1):80–101, 2013. 1

FOPS04. Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. *Journal of Cryptology*, 17(2):81–104, 2004. 1

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Cynthia Dwork, editor, *STOC 2008*, pages 197–206. ACM, 2008. see also https://eprint.iacr.org/2007/432. 4, 11

HHK17. Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Heidelberg, 2017. version, 20170808:094949. See also https://eprint.iacr.org/2017/604. 2, 3, 5, 6, 7, 12, 21

HPS98. Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In Joe Buhler, editor, *ANTS-III*, volume 1423 of *LNCS*, pages 267–288. Springer, Heidelberg, 1998. 4, 12

HRSS17. Andreas Hülsing, Joost Rijneveld, John Schanck, and Peter Schwabe. High-speed key encapsulation from NTRU. In Wieland Fischer and Naofumi Homma, editors, *CHES 2018*, volume 10529 of *LNCS*, pages 232–252. Springer, Heidelberg, 2017. See also https://eprint.iacr.org/2017/667. 2, 17, 18

JZC^{+}17. Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. Post-quantum IND-CCA-secure KEM without additional hash. *IACR Cryptology ePrint Archive*, 2017:1096, 2017. 5, 6

KLS17. Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. *IACR Cryptology ePrint Archive*, 2017:916, 2017. 7

KSS10. Jonathan Katz, Ji Sun Shin, and Adam Smith. Parallel and concurrent security of the HB and HB^{+} protocols. *Journal of Cryptology*, 23(3):402–421, 2010. 9, 22

LDW94. Yuanxing Li, Robert H. Deng, and Xinmei Wang. On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. *IEEE Trans. Information Theory*, 40(1):271–273, 1994. 23, 24

LPR10. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, 2010. See also https://eprint.iacr.org/2012/230. 9

LTV12. Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *STOC 2012*, pages 1219–1234. ACM, 2012. 9

McE78. Robert J. McEliece. A public key cryptosystem based on algebraic coding theory. Technical report, DSN progress report, 1978. 4, 23

Men12. Alfred Menezes. Another look at provable security. Invited Talk at EUROCRYPT 2012, 2012. 1

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, 2012. See also https://eprint.iacr.org/2011/501. 11

MR07. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007. A preliminary version appeared in *FOCS 2004*, 2004. 9

NC00. Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000. 6

Nie86. Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15:159–166, 1986. 4

OP01. Tatsuaki Okamoto and David Pointcheval. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 159–175. Springer, Heidelberg, 2001. 1

Pei09. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *STOC 2009*, pages 333–342. ACM, 2009. 11

Reg09. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):Article 34, 2009. A preliminary version appeared in *STOC 2005*, 2005. 9

SS11. Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Heidelberg, 2011. 9

SY17. Fang Song and Aaram Yun. Quantum security of NMAC and related constructions - PRF domain extension against quantum attacks. In *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, pages 283–309, 2017. 20

TU16. Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 192–216. Springer, Heidelberg, 2016. See also https://eprint.iacr.org/2015/1210. 1, 2, 3, 7

Unr15. Dominique Unruh. Revocable quantum timed-release encryption. *Journal of the ACM*, 62(6):No.49, 2015. The preliminary version appeared in *EUROCRYPT 2014*. See also https://eprint.iacr.org/2013/606. 4, 6

Zha12a. Mark Zhandry. How to construct quantum random functions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 679–687, 2012. 7

Zha12b. Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, Heidelberg, 2012. 7

A Transformations in the Random Oracle Model

We summarize transformations among PKE, DPKE and KEM in the ROM in Figure 9.

B Omitted Proofs

B.1 Proof of Lemma 2.2

Here, we prove Lemma 2.2. Before proving the lemma, we introduce another lemma, which gives a lower bound for a decisional variant of Grover's search problem.

Lemma B.1. *([SY17, Lemma C.1]) Let $g_s : \{0,1\}^\ell \to \{0,1\}$ denotes a function defined as $g_s(s) := 1$ and $g_s(s') := 0$ for all $s' \neq s$, and $g_\perp : \{0,1\}^\ell \to \{0,1\}$ denotes a function that returns 0 for all inputs. Then for any unbounded time adversary $\mathcal{A}$ that issues at most q quantum queries to its oracle, we have*

$$\Pr[1 \leftarrow \mathcal{A}^{g_s}() \mid s \leftarrow \{0,1\}^\ell] - \Pr[1 \leftarrow \mathcal{A}^{g_\perp}()] \leq q \cdot 2^{\frac{\ell-1}{2}}.$$

Then we prove Lemma 2.2 relying on the above lemma.

Proof. (of Lemma 2.2) In order to prove the theorem, we consider the following sequance of games for an algorithm $\mathcal{A}$.

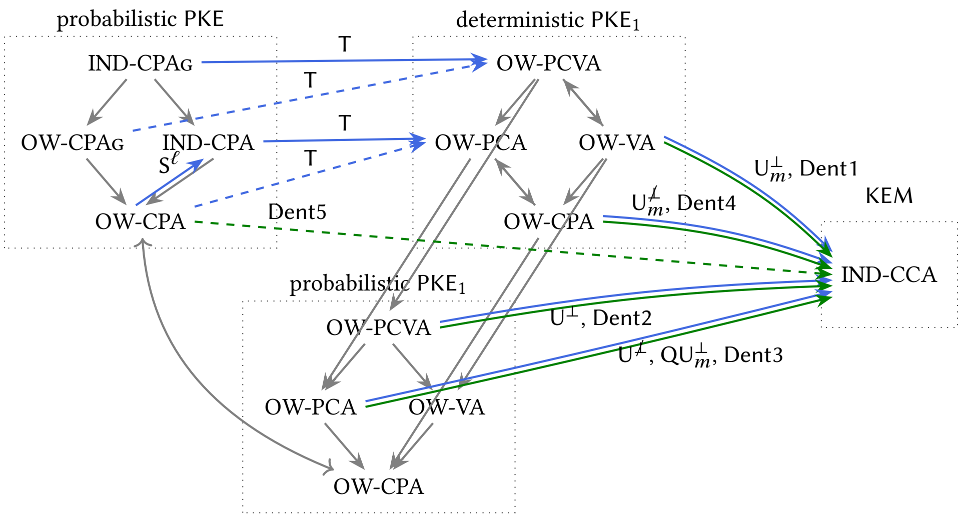

Fig. 9: Transformations in the ROM. GOAL-ATTACKG indicate that the class of PKEs which is GOAL-ATTACK-secure and $\omega(1)$-spreading [FO00,FO99]. Solid arrows indicate tight reductions, dashed arrows indicate non-tight reductions, thin arrows indicates trivial reductions, thick NGreen arrows indicates reduction in [Den03], and thick RoyalBlue arrows indicates reductions in [HHK17].

Game 0: This game returns as $\mathcal{A}^{\mathsf{H},\mathsf{H}(s,\cdot)}()$ outputs, where $s \leftarrow \{0,1\}^{\ell}$ and $\mathsf{H} : \{0,1\}^{\ell} \times \mathcal{X} \to \mathcal{Y}$ be a random function.

Game 1: This game returns as $\mathcal{A}^{O[s,\mathsf{H}_0,\mathsf{H}_1],\mathsf{H}_1(\cdot)}()$ outputs, where $s \leftarrow \{0,1\}^{\ell}$, $\mathsf{H}_0 \colon \{0,1\}^{\ell} \times \mathcal{X} \to \mathcal{Y}$ and $\mathsf{H}_1 \colon \mathcal{X} \to \mathcal{Y}$ be independent random functons, and $O[s,\mathsf{H}_0,\mathsf{H}_1]$ be a function defined as

$$O[s,\mathsf{H}_0,\mathsf{H}_1](s',x) := \begin{cases} \mathsf{H}_0(s',x) & \text{if } s' \neq s, \\ \mathsf{H}_1(x) & \text{if } s' = s. \end{cases} \tag{1}$$

Game 2: This game returns as $\mathcal{A}^{\mathsf{H}_0,\mathsf{H}_1}()$ outputs, where $\mathsf{H}_0 \colon \{0,1\}^{\ell} \times \mathcal{X} \to \mathcal{Y}$ and $\mathsf{H}_1 \colon \mathcal{X} \to \mathcal{Y}$ be independent random functions.

This completes the descriptions of games. What we want to prove is that $|\Pr[\mathrm{Game}_2 = 1] - \Pr[\mathrm{Game}_0 = 1]| \leq q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}}$. It is easy to see that we have $\Pr[\mathrm{Game0} = 1] = \Pr[\mathrm{Game1} = 1]$. What is left is to ptove that $|\Pr[\mathrm{Game}_2 = 1] - \Pr[\mathrm{Game}_1 = 1]| \leq q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}}$. We prove this by a reduction to Lemma B.1. We consider the following algorithm $\mathcal{B}$ that has an access to g which is g_s for randomly chosen $s \leftarrow \{0,1\}^{\ell}$ or $g_{\perp}$ where g_s and $g_{\perp}$ are as defined in Lemma B.1.

$\mathcal{B}^{g}$: It picks two random functions $\mathsf{H}_0 \colon \{0,1\}^{\ell} \times \mathcal{X} \to \mathcal{Y}$ and $\mathsf{H}_1 \colon \mathcal{X} \to \mathcal{Y}$, and runs $\mathcal{A}^{O,\mathsf{H}_1}$ where $\mathcal{B}$ simulates O as follows: When $\mathcal{A}$ queries (s',x) to O, $\mathcal{B}$ queries s' to its own oracle g to obtain a bit b. If $b = 0$, then $\mathcal{B}$ returns $\mathsf{H}_0(s',x)$ to $\mathcal{A}$ and if $b = 1$, then $\mathcal{B}$ returns $\mathsf{H}_1(x')$ to $\mathcal{A}$.

This completes a description of $\mathcal{B}$. It is easy to see that if $g = g_s$ for randomly chosen $s \leftarrow \{0,1\}^{\ell}$, then $\mathcal{B}$ perfectly simulates Game_1, and if $g = g_{\perp}$, then $\mathcal{B}$ perfectly simulates Game_2. Therefore we have

$$|\Pr[\mathrm{Game}_1 = 1] - \Pr[\mathrm{Game}_2 = 1]| = \left|\Pr[1 \leftarrow \mathcal{B}^{g_s}() \mid s \leftarrow \{0,1\}^{\ell}] - \Pr[1 \leftarrow \mathcal{B}^{g_{\perp}}()]\right|.$$

On the other hand, by Lemma B.1, we have

$$\left|\Pr[1 \leftarrow \mathcal{B}^{g_s}() \mid s \leftarrow \{0,1\}^{\ell}] - \Pr[1 \leftarrow \mathcal{B}^{g_{\perp}}()]\right| \leq q_{\mathsf{H}} \cdot 2^{\frac{\ell-1}{2}},$$

since the number of $\mathcal{B}$'s query to its own oracle is exactly the same as the number of $\mathcal{A}$'s query to O, which is equal to q_{H}. This completes the proof of Lemma 2.2. □

C Instantiations of DPKE from Codes

C.1 Preliminaries

$\mathbb{F}$ denotes GF(2). For a vector $e \in \mathbb{F}^n$, $\mathrm{wt}(e)$ denotes the Hamming weight of e, that is, the number of 1s in e. Let $S(n,t)$ be the set of n-dimensional vectors of Hamming weight at most t, that is, $S(n,t) := \{e \in \mathbb{F}^n \mid \mathrm{wt}(e) \leq t\}$. Let $\mathrm{GL}(n,\mathbb{F})$ and $\mathrm{Perm}(n,\mathbb{F})$ denotes the general-linear group of degree n over $\mathbb{F}$ and the group of permutation matrices of degree n over $\mathbb{F}$.

We assume that, for appropriately chosen integers $n = n(\kappa)$, $k = k(\kappa)$, and $t = t(\kappa)$, there exist PPT algorithms CodeGen and Decode satisfying the followings:

- $\mathsf{CodeGen}(1^\kappa, n, k, t)$ outputs $G \in \mathbb{F}^{k \times n}$ and Γ, where G is a generator matrix of a $[n,k]_\mathbb{F}$ linear code.
- $\mathsf{Decode}(\Gamma, mG + e)$ outputs e if $e \in S(n,t)$.

For a $[n,k]_\mathbb{F}$ linear code with a generator matrix $G \in \mathbb{F}^{k\times n}$ of rank k, its parity-check matrix $H \in \mathbb{F}^{(n-k)\times n}$ of rank $n-k$ satisfies $G \cdot H^\top = O$. We assume that there exist a deterministic algorithm G2H that, on input G, outputs its parity-check matrix H and a deterministic algorithm H2G that, on input H, outputs its generator matrix G. For example, the algorithm G2H computes a systematic form $G' = [I_k \mid A] \cdot P$ of G, where $A \in \mathbb{F}^{k \times n}$ and $P \in \mathrm{Perm}(n,\mathbb{F})$, and outputs $H = [-A^\top \mid I_{n-k}]P^{-\top}$. [7]

Let B_τ be the Bernoulli distribution with parameter $\tau \in (0, 1/2)$, that is, $\Pr_{x \leftarrow B_\tau}[x = 1] = \tau$ and $\Pr_{x \leftarrow B_\tau}[x = 0] = 1 - \tau$. Let $\tau = t/n - \epsilon$ and $\alpha = \epsilon n$ for $\epsilon > 0$. Applying the Hoeffding bound, we obtain

$$\Pr_{e \leftarrow B_\tau^n}\left[\mathrm{wt}(e) - E[\mathrm{wt}(e)] \geq \alpha\right] \leq \exp(-2n\alpha^2).$$

Since $E[\mathrm{wt}(e)] = \tau n = t - \alpha$, the statement $\mathrm{wt}(e) - E[\mathrm{wt}(e)] \geq \alpha$ implies $\mathrm{wt}(e) \geq E[\mathrm{wt}(e)] + \alpha = t - \alpha + \alpha = t$. The RHS $\exp(-2n(\epsilon n)^2)$ is negligible. Thus, we obtain the following bound:

Lemma C.1. *For $\tau = t/n - \epsilon$ with $\epsilon > 0$, it holds that*

$$\Pr_{e \leftarrow B_\tau^n}[\mathrm{wt}(e) \geq t] \leq \exp(-2\epsilon^2 n^3).$$

Assumptions: Blum et al. [BFKL93] introduced the learning-parity-with-noise (LPN) problem. Its decisional version is formalized by Katz, Shin and Smith [KSS10].

Definition C.1 (LPN assumption in matrix form). *For all κ, let $k = k(\kappa)$ and $q = q(\kappa)$ be integers and let $\tau = \tau(\kappa)$ be a real in $(0, 1/2)$. The decisional learning-parity-with-noise (LPN) assumption $\mathsf{LPN}_{k,\tau}$ states that for any $n = \mathrm{poly}(\kappa)$, it is computationally hard to distinguish the following two distributions:*

- *$A, sA + e$, where $A \leftarrow \mathbb{F}^{k\times n}$, $s \leftarrow \mathbb{F}^k$, and $e \leftarrow B_\tau^n$*
- *A, u, where $A \leftarrow \mathbb{F}^{k \times n}$ and $u \leftarrow \mathbb{F}^n$.*

The McEliece-key-indistinguishability assumption is introduced in [CFS01] for signature context. The statements states the public key of the McEliece encryption scheme is pseudorandom. See e.g., [DDMQN12].

Definition C.2 (McEliece-key-indistinguishability assumption w.r.t. CodeGen). *For all κ, let $k = k(\kappa)$, $n = n(\kappa)$, and $t = t(\kappa)$ be positive integers. The McEliece-key-indistinguishability assumption w.r.t.* CodeGen*, denoted by* $\mathsf{McE}_{k,n,t,\mathsf{CodeGen}}$*, states that it is computationally hard to distinguish the following two distributions:*

- *$\tilde{G} := SGP$, where $(G,\Gamma) \leftarrow \mathsf{CodeGen}(k,n)$, $S \leftarrow \mathrm{GL}(k,\mathbb{F})$, and $P \leftarrow \mathrm{Perm}(n,\mathbb{F})$.*
- *$\tilde{G} \leftarrow \mathbb{F}^{k\times n}$*

We additionally introduce the Niederreiter-key-indistinguishability assumption w.r.t. CodeGen, in which we employ parity-check matrices instead of generator matrices. See e.g., [FGK+13].

Definition C.3 (Niederreiter-key-indistinguishability assumption w.r.t. CodeGen). *For all κ, let $k = k(\kappa)$, $n = n(\kappa)$, and $t = t(\kappa)$ be positive integers. The Niederreiter-key-indistinguishability assumption w.r.t.* CodeGen*, denoted by* $\mathsf{Nie}_{k,n,t,\mathsf{CodeGen}}$*, states that it is computationally hard to distinguish the following two distributions:*

- *$\tilde{H} := MHP$, where $(G,\Gamma) \leftarrow \mathsf{CodeGen}(k,n)$, $H := \mathsf{G2H}(G)$, $M \leftarrow \mathrm{GL}(n-k,\mathbb{F})$, and $P \leftarrow \mathrm{Perm}(n,\mathbb{F})$.*
- *$\tilde{H} \leftarrow \mathbb{F}^{(n-k)\times n}$*

[7] Letting $G = [G_{\mathsf{left}} \mid G_{\mathsf{right}}] \cdot P$ with $G_{\mathsf{left}} \in \mathrm{GL}(k,\mathbb{F})$ and $P \in \mathrm{Perm}(n,\mathbb{F})$, we have $G' = G_{\mathsf{left}}^{-1} G P = [I_k \mid G_{\mathsf{left}}^{-1} \cdot G_{\mathsf{right}}]P$ and set $A = G_{\mathsf{left}}^{-1} \cdot G_{\mathsf{right}}$. We obtain $G \cdot H^\top = G_{\mathsf{left}} G' P \cdot ([-A^\top \mid I_{n-k}]P^{-\top})^\top = G_{\mathsf{left}}[I_k \mid A] \cdot P \cdot P^{-1}\binom{-A}{I_{n-k}} = G_{\mathsf{left}}(-A + A) = O$.

C.2 Code-based DPKEs

MeEliece-based DPKE: We review the McEliece PKE [McE78]. Let n, k, and t be positive integers with $n > k$. We consider $[n,k]_{\mathbb{F}}$-linear code with an efficient decoder that can correct any patter of up to t errors.

The public key is $\tilde{G} = SGP$, where S is a random non-singular $k \times k$ matrix, G is a generator matrix in $\mathbb{F}^{k \times n}$ of $[n,k]_{\mathbb{F}}$-linear code, and P is a random $n \times n$ permutation matrix. The ciphertext of $m \in \mathbb{F}^k$ with randomness $e \in \mathcal{S}_t$ is $c = m\tilde{G} + e \in \mathbb{F}^m$. We can retrieve m using a secret key because we compute $yP^{-1} = mSG + eP^{-1}$, decode it into mS, and obtain m. Observe that we can retrieve e also by computing $e := c - m\tilde{G}$. Thus, we interpret (m, e) as plaintext and obtain DPKE.

Correctly speaking, $\mathsf{Decode}(\Gamma, yP^{-1})$ in our definition outputs eP^{-1}. Thus, we obtain $e := eP^{-1} \cdot P$ and $mSG = yP^{-1} - eP^{-1}$, and so on. Now, we describe the McEliece-based DPKE.

Parameters: Let n, k, t, τ be parameters with $\tau = t/n - \epsilon$ for $\epsilon > 0$.
- The plaintext space is $\mathcal{M} := \mathbb{F}^k \times \mathcal{S}_t$.
- The sampler $\mathcal{D}_{\mathcal{M}}$ samples $m \leftarrow \mathbb{F}^k$ and $r \leftarrow B_\tau^n$ until $\mathrm{wt}(r) \leq t$.
- The ciphertext space is $C := \mathbb{F}^n$.
- We require $2^n \gg 2^k \cdot \sum_{i=0}^{t} \binom{n}{t}$. E.g., $2^{n-k} \gg t \cdot n^t \geq \sum_{i=0}^{t} \binom{n}{t}$.

Key Generation: $\mathsf{Gen}(1^\kappa)$ generates $(G, \Gamma) \leftarrow \mathsf{CodeGen}(1^\kappa, n, k, t)$, a random non-singular $k \times k$ matrix S, and a random $n \times n$ permutation matrix P. It outputs $ek = \tilde{G} = SGP \in \mathbb{F}^{k \times n}$ and $dk = (S, G, P, \Gamma)$.

Encryption: $\mathsf{Enc}(ek, (m, e))$ outputs $c = m\tilde{G} + e \in \mathbb{F}^n$.

Decryption: $\mathsf{Dec}(sk, c)$ computes $y := cP^{-1}$, our decoder outputs $e' := \mathsf{Decode}(\Gamma, y)$, computes $d' := y - e'$, computes m' such that $m' \cdot G = d'$, and computes $m := m'S^{-1}$.

The properties of this DPKE are summarized as follows:

Perfect correctness: If $c = mSGP + e$, then $y = cP^{-1} = mSG + eP^{-1}$, which is a codeword of mS plus error vector eP^{-1} of weight at most t. Thus, Decode on input Γ and y outputs $e' = eP^{-1}$. Now, we have $d' = y - e' = mSG$ and $m' = mS$. Hence, we get $m = m'S^{-1}$ as we wanted.

Sparseness: Sparseness follows from $|C| = 2^n \gg 2^k \cdot \sum_{i=0}^{t} \binom{n}{t} = |\mathcal{M}| = |\mathsf{Enc}(ek, \mathcal{M})|$.

Pseudorandomness: What we want to show is

$$(\tilde{G}, c = m\tilde{G} + e) \approx_c (\tilde{G}, u),$$

where $(\tilde{G}, dk) \leftarrow \mathsf{Gen}(1^\kappa)$, $(m, e) \leftarrow \mathcal{D}_{\mathcal{M}}$, and $u \leftarrow \mathbb{F}^n$.
- We first replace $\tilde{G}$ with random $\bar{G}$. This is justified by the McEliece assumption w.r.t. CodeGen.
- We next replace e with random $e' \leftarrow B_\tau^n$. This is justified by Lemma C.1 with our parameter setting.
- We next replace $c = m\bar{G} + e'$ with random u. This is justified by the LPN assumption $\mathsf{LPN}_{k,\tau}$.
- We then go backward by replacing random $\bar{G}$ with $\tilde{G}$. This is justified by the McEliece assumption w.r.t. CodeGen again.

Niederreiter-based DPKE: It is well-known that the Niederreiter PKE is the dual of the McEliece PKE [LDW94].[8] Let us consider $(n,k)_q$-code C with error-decoder up to t errors. The public key is $\tilde{H} = MHP$, where M is a random non-singular $(n-k) \times (n-k)$ matrix, H is an $(n-k) \times n$ parity-check matrix of code C, and P is a random $n \times n$ permutation matrix. The ciphertext of $e \in \mathcal{S}_t$ is $c = e\tilde{H}^\top \in \mathbb{F}^{n-k}$. We can retrieve e using a secret key because we compute $cM^{-\top}$ ($= e\tilde{H}^\top M^{-\top} = eP^\top H^\top$), decode it into $eP^\top$, and compute $eP^{-\top} = e$.

Parameters: Let n, k, t, τ be parameters with $\tau = t/n - \epsilon$ for $\epsilon > 0$.
- The plaintext space is $\mathcal{M} := \mathcal{S}_t$.
- The sampler $\mathcal{D}_{\mathcal{M}}$ samples $e \leftarrow B_\tau^n$ until $\mathrm{wt}(e) \leq t$.
- The ciphertext space is $C := \mathbb{F}^n$.
- We require $2^n \gg 2^k \cdot \sum_{i=0}^{t} \binom{n}{t}$. E.g., $2^{n-k} \gg t \cdot n^t \geq \sum_{i=0}^{t} \binom{n}{t}$.

Key Generation: $\mathsf{Gen}(1^\kappa)$ generates $(G, \Gamma) \leftarrow \mathsf{CodeGen}(1^\kappa, n, k, t)$, $H := G^*$, a random non-singular $(n-k) \times (n-k)$ matrix M, and a random $n \times n$ permutation matrix P. It outputs $ek = \tilde{H} = MHP \in \mathbb{F}^{(n-k) \times n}$ and $dk = (M, H, P, \Gamma)$.

Encryption: $\mathsf{Enc}(ek, e)$ outputs $c = e\tilde{H}^\top \in \mathbb{F}^{n-k}$.

[8] Li, Deng, and Wang showed that the onewayness of the Niederreiter PKE is equivalent to that of the McEliece PKE [LDW94].

Decryption: $\mathsf{Dec}(sk, c)$ computes $c' := cM^{-\top}$, decodes it into $e' := \mathsf{Decode}(\Gamma, c')$, and computes $e := e'P^{-\top}$.

The properties of this DPKE are summarized as follows:

Perfect correctness: This is obvious.
Sparseness: Sparseness follows from $|C| = 2^{n-k} \gg \sum_{i=0}^{t} \binom{n}{t} = |\mathcal{M}| = |\mathsf{Enc}(ek, \mathcal{M})|$.
Pseudorandomness: What we want to show is

$$(\tilde{H}, c = e \cdot \tilde{H}^{\top}) \approx_c (\tilde{H}, u),$$

where $(\tilde{H}, dk) \leftarrow \mathsf{Gen}(1^{\kappa})$, $e \leftarrow \mathcal{D}_{\mathcal{M}}$, and $u \leftarrow \mathbb{F}^{n-k}$.

- We first replace $\tilde{H}$ with random $\bar{H}$. This is justified by the Niederreiter assumption w.r.t. CodeGen.
- We next replace e with random $e' \leftarrow B_{\tau}^{n}$. This is justified by Lemma C.1 with our parameter setting.
- We next replace $c = e \cdot \bar{H}^{\top}$ with random u. This is justified by the LPN assumption $\mathsf{LPN}_{k,\tau}$. (See [LDW94].)
- We then go backward by replacing random $\bar{H}$ with $\tilde{H}$. This is justified by the Niederreiter assumption w.r.t. CodeGen.

D PR-CPA security

Here, we recall the security notion of DPKE called PR-CPA defined in previous versions of this paper. Then we prove that PR-CPA-security is implied by the disjoint simulatability. For PR-CPAsecurity, we require a DPKE scheme to have two additional PPT algorithms $\widetilde{\mathsf{Gen}}$ and $\widetilde{\mathsf{Enc}}$: $\widetilde{\mathsf{Gen}}$ is a PPT algorithm that takes the security parameter as input and outputs a fake encryption key $\widetilde{ek}$, which is indistinguishable from a real encryption key. This means that the original encryption algorithm Enc should be able to encrypt a message even with a fake encryption key. $\widetilde{\mathsf{Enc}}$ is a PPT algorithm that takes a fake encryption key as input and outputs a random fake ciphertext, which is indistinguishable from a random real ciphertext with a fake encryption key. We further require that the probability that a random fake ciphertext with a fake encryption key falls in the range of a real ciphertext with a fake encryption keyis negligible. For example, this condition is satisfied if a set of real ciphertexts is sufficiently sparser than a set of fake ciphertext or a set of real ciphertexts is disjoint with a set of fake ciphertext. The formal definition follows:

Definition D.1. *Let $\mathcal{D}_{\mathcal{M}}$ be a distribution on $\mathcal{M}$. A deterministic PKE scheme* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with plaintext and ciphertext spaces $\mathcal{M}$ and C is $\mathcal{D}_{\mathcal{M}}$-*PR-CPA secure *if the following properties hold; There exist two PPT algorithms $\widetilde{\mathsf{Gen}}$ and $\widetilde{\mathsf{Enc}}$ that satisfy the followings:*

- *(Statistical Disjointness:) for any $\widetilde{ek}$ generated by $\widetilde{\mathsf{Gen}}(1^{\kappa})$, the probability that a fake ciphertext is in the range of a real ciphertext generated by $\mathsf{Enc}(\widetilde{ek}, \cdot)$ is negligible, that is,*

$$\Pr[c \in \mathsf{Enc}(\widetilde{ek}, \mathcal{M}) \mid c \leftarrow \widetilde{\mathsf{Enc}}(\widetilde{ek})]$$

  is negligible.
- *(PR-Key Security:) for any PPT adversary $\mathcal{A}$, its advantage to distinguish a real key from a fake key, denoted by $\mathsf{Adv}^{\text{pr-key}}_{\mathcal{A},\mathsf{PKE}}(\kappa)$, is negligible;*

$$\mathsf{Adv}^{\text{pr-key}}_{\mathcal{A},\mathsf{PKE}}(\kappa) := \left| \begin{array}{c} \Pr\left[1 \leftarrow \mathcal{A}(ek) \mid (ek, dk) \leftarrow \mathsf{Gen}(1^{\kappa});\right] \\ -\Pr\left[1 \leftarrow \mathcal{A}(\widetilde{ek}) \mid \widetilde{ek} \leftarrow \widetilde{\mathsf{Gen}}(1^{\kappa})\right] \end{array} \right|$$

  is negligible.
- *(PR-Ciphertexts Security:) for any PPT adversary $\mathcal{A}$, its advantage to distinguish a real ciphertext from a fake ciphertext with a fake key, denoted by $\mathsf{Adv}^{\text{pr-cipher}}_{\mathcal{A},\mathcal{D}_{\mathcal{M}},\mathsf{PKE}}(\kappa)$, is negligible;*

$$\mathsf{Adv}^{\text{pr-cipher}}_{\mathcal{A},\mathcal{D}_{\mathcal{M}},\mathsf{PKE}}(\kappa) := \left| \begin{array}{c} \Pr\left[1 \leftarrow \mathcal{A}(\widetilde{ek}, c^*) \mid \widetilde{ek} \leftarrow \widetilde{\mathsf{Gen}}(1^{\kappa}); m^* \leftarrow \mathcal{D}_{\mathcal{M}}; c^* := \mathsf{Enc}(\widetilde{ek}, m^*)\right] \\ -\Pr\left[1 \leftarrow \mathcal{A}(\widetilde{ek}, c^*) \mid \widetilde{ek} \leftarrow \widetilde{\mathsf{Gen}}(1^{\kappa}); c^* \leftarrow \widetilde{\mathsf{Enc}}(\widetilde{ek})\right] \end{array} \right|$$

  is negligible.

Remark D.1. Though the above definition is essentially the same as the one in previous versions, there are some notational differences described below.

- In previous versions, we implicitly assumed that message space $\mathcal{M}$ is associated with a certain distribution, and used $m \leftarrow \mathcal{M}$ to mean m is sampled according to this distribution. To clarify this, we explicitly denote a distribution $\mathcal{D}_{\mathcal{M}}$, and define the PR-CPA-security respect to the distribution.
- In previous versions, we used a (T, ϵ)-type definition. For compatibility to the other part of the current version, we quit it.

We prove that the disjoint simulatability implies the PR-CPA-security

Lemma D.1. *If a DPKE scheme* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* $\mathcal{D}_{\mathcal{M}}$*-disjoint simulatable, then* PKE *is* $\mathcal{D}_{\mathcal{M}}$*-PR-CPA-secure.*

Proof. Let $\mathcal{S}$ be a simulator that satisfies the properties in Definition 3.1. Then we construct $\widetilde{\mathsf{Gen}}$ and $\widetilde{\mathsf{Enc}}$ as follows.

$\widetilde{\mathsf{Gen}}(1^\kappa)$: This algorithm runs $(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$ and outputs $\widetilde{ek} := ek$.
$\widetilde{\mathsf{Enc}}(\widetilde{ek})$: This algorithm runs $c \leftarrow \mathcal{S}(\widetilde{ek})$ and outputs c.

It is easy to see that we have $\Pr[c \in \mathsf{Enc}(\widetilde{ek}, \mathcal{M}) | c \leftarrow \widetilde{\mathsf{Enc}}(\widetilde{ek})] = \mathsf{Disj}_{\mathsf{PKE},\mathcal{S}}(\kappa)$, $\mathsf{Adv}^{\text{pr-key}}_{\mathcal{A},\mathsf{PKE}}(\kappa) = 0$, and $\mathsf{Adv}^{\text{pr-cipher}}_{\mathcal{A},\mathcal{D}_{\mathcal{M}}\mathsf{PKE}}(\kappa) = \mathsf{Adv}^{\text{ds-ind}}_{\mathsf{PKE},\mathcal{D}_{\mathcal{M}},\mathcal{A},\mathcal{S}}(\kappa)$. Therefore the lemma follows.