QCCA-Secure Generic Transformations in the Quantum Random Oracle Model

Tianshu Shan[1, 2], Jiangxia Ge[1, 2], and Rui Xue[1, 2]

[1]State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China

{shantianshu, gejiangxia, xuerui}@iie.ac.cn

Abstract

The post-quantum security of cryptographic systems assumes that the quantum adversary only receives the classical result of computations with the secret key. Furthermore, if the adversary is able to obtain a superposition state of the result, it is unknown whether the post-quantum secure schemes still remain secure.

In this paper, we formalize one class of public-key encryption schemes, named oracle-masked schemes, relative to random oracles. For each oracle-masked scheme, we design a preimage extraction procedure and prove that it simulates the quantum decryption oracle with a certain loss. We also observe that the implementation of the preimage extraction procedure for some oracle-masked schemes does not need to take the secret key as input. This contributes to the IND-qCCA security proof of these schemes in the quantum random oracle model (QROM). As an application, we prove the IND-qCCA security of schemes obtained by the Fujisaki-Okamoto (FO) transformation and REACT transformation in the QROM, respectively.

Notably, our security reduction for FO transformation is tighter than the reduction given by Zhandry (Crypto 2019).

Keywords: public-key encryption, FO transformation, REACT transformation, quantum random oracle model, quantum chosen ciphertext security

1 Introduction

Cryptographic schemes often have efficient constructions in the random oracle model (ROM) [BR93], in which schemes are proven to be secure assuming the existence of the publicly accessible random oracle. When to build a concrete scheme, the random oracle is instantiated with a cryptographic hash function. Thus in the real world attack, any quantum attacker is able to evaluate the hash function in superposition. To capture this issue, Boneh et al. [BDF+11] proposed the quantum random oracle model (QROM) where the quantum adversary can query the random oracle with superposition states.

Boneh and Zhandry [BZ13] then introduced the indistinguishability under quantum chosen ciphertext attacks (IND-qCCA) security notion for encryption schemes, where the adversary can make quantum queries to the decryption oracle. In this paper, we consider the IND-qCCA security for public-key encryption (PKE) schemes in the QROM.

Generic transformations are widely used to enhance the security of PKE [BR93, FO99, OP01, JSHJ+02]. The Fujisaki-Okamoto (FO) transformation [FO99] turns an arbitrary PKE that is one-wayness under the chosen plaintext attacks (OW-CPA) into a PKE that is indistinguishable under the chosen ciphertext attacks (IND-CCA) in the ROM. The REACT transformation [OP01] turns an arbitrary PKE that is one-wayness under the plaintext check attacks (OW-PCA) into an IND-CCA secure PKE in the ROM.

Boneh et al. [BDF+11] summarized several proof techniques that are commonly used in the ROM but not appropriate to the quantum setting straight forwardly. "Extractability/Preimage Awareness", as one of them, is that the simulator learns the preimages the adversary takes interest in when simulating the random oracle for the adversary. And this technique is the core to simulate answers to decryption queries in the security proof for both FO and REACT in the ROM. This had been an obstacle to the security proof of FO in QROM. To circumvent it, Targhi and Unruh [TU16] and the follow-up result by Ambainis et al. [AHU19] modified FO transformation by appending an extra hash function to the ciphertext and gave the security proof for the modified ones in the QROM.

Zhandry [Zha19] proposed the compressed oracle technique, with which the simulator can "record" quantum queries to the random oracle while simulating it efficiently. This makes it feasible to use Preimage Awareness technique in the quantum setting and thus makes it possible to prove the security of the unmodified FO in QROM. Indeed in the full version of [Zha19], Zhandry gave a proof that FO transforms any OW-CPA secure PKE into an IND-qCCA secure PKE in the QROM.

However, in his proof, as was pointed out by Don et al. [DFMS22], the answers to decryption queries in Hybrids 2 to 4 are simulated by applying (purified) measurements on the internal state of the compressed oracle, yet these measurements are hard to be determined explicitly from their respective descriptions. Until now, this is considered as the gap that prevents the analysis of the disturbance caused by those measurements.

1.1 Our Result

In this paper, we review Zhandry's proof for FO transform and specify the procedures for simulating the decryption oracle, based on which we prove the IND-qCCA security for FO and REACT in the QROM, respectively. The concrete security bounds for FO and REACT are shown in Table 1.

Transformation	Underlying security	Achieved security	Security bound($\approx$)
FO	OW-CPA	IND-qCCA	$d/\sqrt{2^{\gamma}} + (q+d)\cdot\sqrt{\epsilon^{asy}} + \epsilon^{sy}$
REACT	OW-qPCA	IND-qCCA	$d/\sqrt{2^{n}} + q\cdot d\cdot\sqrt{\epsilon^{asy}} + \epsilon^{sy}$

Table 1: Concrete security bounds for FO and REACT in the QROM. The "Underlying security" column omits the one-time security of the underlying secret-key encryption(SKE) scheme for both FO and REACT. ϵ^{asy} is the advantage of the reduced adversary against the security of the underlying PKE. ϵ^{sy} is the advantage against the security of the underlying SKE scheme. d is the number of decryption queries. q is the total number of random oracle queries. γ is the min-entropy of the ciphertext of the underlying PKE. n is the length of the hash value being one part of the ciphertext of the achieved PKE.

Firstly, we define a class of PKEs called oracle-masked. This class contains the hybrid schemes obtained by the FO transform and that by REACT transform.

One property of oracle-masked schemes is that the Preimage Awareness technique is applied in their classical IND-CCA security proof in the ROM. More specifically, in the classical ROM proofs, the reduction algorithm record queries to the random oracle while simulating it for the adversary. When to reply to adversary's decryption queries, the reduction algorithm learns the message the adversary has interest in from the recorded random oracle queries.

Then we design the preimage extraction procedure $\mathrm{U_{Ext}}$. This is a tool of applying Preimage Awareness technique to the IND-qCCA security proof for oracle-masked schemes in the QROM. Fix a public/secret key-pair, the original decryption oracle corresponds to a unitary operator CCA , and we have $\|(\mathrm{CCA}-\mathrm{U_{Ext}})|\phi\rangle\| \leq O(1)\cdot\sqrt{\eta}$ for one type of state $|\phi\rangle$, where η depends on the concrete scheme. As applications of this tool, we provide a tighter security reduction for unmodified FO transform than the security reduction given by Zhandry, and give the IND-qCCA security proof for REACT transform with a concrete bound.

1.2 Related work

Abstract frameworks were proposed to simplify the use of the compressed oracle technique in different situations[CMS19, CFHL21, DFMS22]. They formalize properties that are satisfied in the presence of random oracle, and lift them to the quantum setting.

Particularly, Don et al. [DFMS22] have considered Preimage Awareness in a more general form. Specifically, they define a simulator that simulates the random oracle and also allows the extraction query, that is replied with the guess of the preimage of the query. They then prove that this simulation of the random oracle is statistically indistinguishable from the real ones if some properties are satisfied. In their security proof, the extraction query is restricted to be classical in the simulation. Therefore, their results seem to be tailored for post-quantum security proofs, yet are not sufficient to prove the IND-qCCA security.

2 Preliminaries

2.1 Notation

Denote $\mathcal{M}$, $\mathcal{C}$ and $\mathcal{R}$ as key space, message space and ciphertext space, respectively. For a finite set $\mathcal{X}$, denote $|\mathcal{X}|$ as the number of elements $\mathcal{X}$ contains, and denote $x \overset{\$}{\leftarrow} \mathcal{X}$ as uniformly choose a random element x from $\mathcal{X}$. $[b = b']$ is an integer, that is 1 if $b = b'$ and 0 otherwise. $\Pr[P : Q]$ is the probability that predicate P keeps true where all the variables in P are assigned according to the program in Q.

Algorithms take a security parameter λ as input, and we omit it for convenience. A non-negative function $f(\lambda)$ is a negligible function if it is smaller than the inverse of any non-negative polynomial $p(\lambda)$ for sufficient large λ. Time(f) is denoted as the time complexity of an algorithm computing function f.

2.2 Quantum Background

Here we only give some background on quantum computation and quantum information, and we refer to [NC02] for more discussion.

A quantum system Q is a complex Hilbert space $\mathcal{H}_Q$ with an inner product $\langle \cdot | \cdot \rangle$, the notation '$|\cdot\rangle$' or '$\langle \cdot |$' is called the Dirac notation. The state $|\psi\rangle$ of quantum system Q is a unit vector of $\mathcal{H}_Q$, it can be a superposition of computational basis state $\{|x\rangle\}_x$ like $\sum_x \alpha_x |x\rangle$. The norm of $|\psi\rangle$ is defined as $\||\psi\|| = \sqrt{\langle \psi | \psi \rangle}$. The tensor product $Q_1 \otimes Q_2$ of quantum systems Q_1 and Q_2 is a composite quantum system and the product state is $|\psi_1\rangle \otimes |\psi_2\rangle \in Q_1 \otimes Q_2$ where $|\psi_1\rangle \in Q_1$, $|\psi_2\rangle \in Q_2$. A transformation U is called a unitary operation over $\mathcal{H}_Q$ if $\mathrm{U}\mathrm{U}^\dagger = \mathrm{U}^\dagger \mathrm{U} = \mathbf{I}$, where $\mathrm{U}^\dagger$ is the Hermitian conjugate of U and $\mathbf{I}$ is the identity operator over $\mathcal{H}_Q$. A measurement on a quantum system Q are described by a collection of measurement operators $\{M_m\}_m$, that satisfy the completeness equation $\sum_m M_m^\dagger M_m = \mathbf{I}$. One of the most frequently used measurement is the computational basis measurement in which measurement operator $M_m = |m\rangle\langle m|$, where m is a computational basis vector. Moreover, unitary operations and measurements can be generally described by completely positive trace-preserving operations.

A quantum system with an exactly known state $|\psi\rangle$ is said to be in a pure state. If a quantum system whose state is a pure state $|\psi_i\rangle$ with probability p_i, then this quantum system is called in a mixed state, that can be denote as $\{p_i, |\psi_i\rangle\}_i$. In the quantum information theory, the statistical behavior of the above mixed state can be described by a density operator $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$. For two density operator ρ and ρ', the trace distance of them is denoted as $D(\rho, \rho')$, which measures the statistical distance of the quantum information stored in state ρ and ρ'.

Quantum algorithms. A quantum algorithm A is a family of generalized quantum circuits $\{A_n\}_n$ over a finite universal gate set. If A is an oracle algorithm, i.e., it has access to several different oracles, then each of these oracles is modeled with one separate oracle gate. The size of a quantum circuit is the number of gates it contains plus the number of input and output qubits. A is polynomial-time if there is a polynomial $p(\cdot)$ such that the size of A_n is at most $p(n)$ for every n. And we call A is an efficient algorithm if A is polynomial-time.

For a quantum oracle algorithm A have quantum access to oracle $\mathcal{O}$, suppose A starts with an initial state $|\psi\rangle$. We call A a unitary quantum oracle algorithm if it alternately queries oracle $\mathcal{O}$ and applies a fixed unitary operation on the registers of A. And the final state of A can be represented by $\mathrm{U}_q\mathrm{O}\ldots\mathrm{U}_1\mathrm{O}\mathrm{U}_0|\psi\rangle$, where q is the number of queries made by A and $\mathrm{U}_0, \ldots, \mathrm{U}_q$ are unitary operations between queries. There is a well-known fact that, with simple preprocessing, any quantum oracle algorithm A can be transformed into a unitary quantum oracle algorithm without increase the query times.

2.3 Quantum Random Oracle Model

In the ROM, we assume the existence of the random oracle $\mathcal{O} : \mathcal{X} \to \mathcal{Y}$, and $\mathcal{O}$ is publicly accessible to all parties. For conceteness, let $\mathcal{Y} = \{0,1\}^n$. $\mathcal{O}$ is initialized by choosing $H \xleftarrow{\$} \Omega_H$, where Ω_H is the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$. In the QROM, quantum algorithms can query with superposition states, and the oracle performs the unitary mapping $|x,y\rangle \mapsto |x, y \oplus \mathcal{O}(x)\rangle$ on the query state. Oracle $\mathcal{O}$ also allows making classical queries. To query x, set the input and output state to be $|x,0\rangle$ and measure it after querying $\mathcal{O}$ to obtain $\mathcal{O}(x)$.

Below, we introduce several tools for QROM, that are used in this paper. We begin with two ways for the simulation of the random oracle in the QROM.

Theorem 1 ([Zha15]). *Let H be a function chosen from the set of $2q$-wise independent functions uniformly at random. Then for any quantum algorithm A with at most q queries,*

$$\Pr[b = 1 : b \leftarrow A^H()] = \Pr[b = 1 : b \leftarrow A^{\mathcal{O}}()].$$

The Compressed Oracle. Here we briefly introduce the compressed oracle technique, and we only consider the Compressed Standard Oracles(CStO), one version of the compressed oracle, with query number at most q. We refer to the full version of [Zha19] for more details of the compressed oracle.

The core idea of the compressed oracle technique is the purification of the quantum-accessible random oracle, and the purified oracle imperfectly records quantum queries to the random oracle. In the QROM, the random oracle is initialized by uniformly sampling a function H from Ω_H. If random oracle is queried with a quantum state $|x,y\rangle$, then the replied state is a mixed state and can be represented as $\{p_i, |x, y \oplus H_i(x)\rangle\}$, where $p_i = 1/|\Omega_H|$, $i = 1, \ldots, |\Omega_H|$. This mixed state can be purified to state $1/|\Omega_H| \sum_H |x, y \oplus H(x), H\rangle$, where $|H\rangle$ is the internal state of oracle $\mathcal{O}$ and H of $|H\rangle$ is a truth table of function H.

Instead of a superposition state of H, CStO takes a superposition of database as its internal state and simulates random oracle $\mathcal{O}$. We denote this compressed oracle by $\mathsf{CStO}_{\mathcal{O}}$, and database by D. D is represented by an element of set $(\mathcal{X} \times \bar{\mathcal{Y}})^l$ where $\bar{\mathcal{Y}} = \mathcal{Y} \cup \{\perp\}$, l is the length of D. For any $x \in \mathcal{X}$, if (x,y) exists as an entry of D, then $(x,y) \in D$ and $D(x) = y$. Otherwise, $D(x) = \perp$. Denote $|D|$ as the total number of $x \in \mathcal{X}$ such that $D(x) \neq \perp$. Then for any $y \in \mathcal{Y}$ and D that $D(x) = \perp$, $|D| < l$, define $D \cup (x,y)$ to be the database that $D \cup (x,y)(x') = D(x')$ for any $x' \neq x$ and $D \cup (x,y)(x) = y$. Moreover, any D is written in the form of $((x_1,y_1), \ldots, (x_s,y_s), (0,\perp), \ldots, (0,\perp))$ such that $|D| = s \leq l$, $x_1 < x_2 < \cdots < x_s$.

For any $x \in \mathcal{X}$, define unitary $\mathsf{StdDecomp}_x$ applied on the database state as below:

- For D that $D(x) = \perp$ and $|D| = l$, $\mathsf{StdDecomp}_x|D\rangle = |D\rangle$.
- For D that $D(x) = \perp$ and $|D| < l$, $\mathsf{StdDecomp}_x|D \cup (x,\beta_r)\rangle = |D \cup (x,\beta_r)\rangle$ for any $r \neq 0$, $\mathsf{StdDecomp}_x|D \cup (x,\beta_0)\rangle = |D\rangle$, $\mathsf{StdDecomp}_x|D\rangle = |D \cup (x,\beta_0)\rangle$,
  where state $|D \cup (x,\beta_r)\rangle = 1/\sqrt{2^n} \sum_{y \in \mathcal{Y}} (-1)^{y \cdot r} |D \cup (x,y)\rangle$ for any $r \in \mathcal{Y}$.

$\mathrm{CStO}_{\mathcal{O}}$ initializes a database state $|(0,\perp)^q\rangle$ with length q.For any query to random oracle $\mathcal{O}$, $\mathrm{CStO}_{\mathcal{O}}$ does three steps: First, perform the unitary $|x,y,D\rangle \mapsto |x,y\rangle \mathsf{StdDecomp}_x|D\rangle$ in superposition. Next, apply the map $|x,y,D\rangle \mapsto |x, y \oplus D(x), D\rangle$. Finally, repeat the first step.

Theorem 2 ([Zha19]). *$\mathsf{CStO}_{\mathcal{O}}$ and random oracle $\mathcal{O}$ are indistinguishable for any quantum algorithm A, i.e.,*

$$\Pr[b = 1 : b \leftarrow A^{\mathsf{CStO}_{\mathcal{O}}}()] = \Pr[b = 1 : b \leftarrow A^{\mathcal{O}}()].$$

It is also observed that any quantum state on the database register is orthogonal to state $|D \cup (x,\beta_0)\rangle$ in the simulation of $\mathsf{CStO}_{\mathcal{O}}$. Therefore, the database state should be the superposition state of $|D \cup (x,\beta_r)\rangle$ for $r \neq 0$. This fact will be used later.

Semi-classical Oracle. For set $\mathcal{X}$ and $\mathcal{S}$, define $f_{\mathcal{S}} : \mathcal{X} \to \{0,1\}$ to be an indicator function such that $f_{\mathcal{S}}(x) = 1$ if $x \in \mathcal{S}$ and 0 otherwise. Then we define the semi-classical oracle $\mathcal{O}_{\mathcal{S}}^{SC} : \mathcal{X} \to \{0,1\}$. For any quantum query, $\mathcal{O}_{\mathcal{S}}^{SC}$ does the following steps. First, initialize a qubit T to be $|0\rangle$. Then evaluate the mapping $|x,0\rangle \mapsto |x, f_{\mathcal{S}}(x)\rangle$ in superposition. Finally, measure T in the computational basis and obtain a bit $b \in \{0,1\}$ as its output.

Theorem 3 (Semi-classical O2H [AHU19]). *Let $\mathcal{S}$ be a random subset of $\mathcal{X}$, $H : \mathcal{X} \to \mathcal{Y}$ be a random function, z be a random bitstring. And H,$\mathcal{S}$,z may have arbitrary joint distribution. Let $H \setminus \mathcal{S}$ be an oracle that on input $x \in \mathcal{X}$, first queries $H(x)$ and then $\mathcal{O}_{\mathcal{S}}^{SC}(x)$. Let A be a quantum oracle algorithm with query depth d. In the execution of $A^{H\setminus\mathcal{S}}(z)$, let Find be the event that $\mathcal{O}_{\mathcal{S}}^{SC}$ ever outputs 1. Then*

$$\left|\Pr[b = 1 : b \leftarrow A^{H}(z)] - \Pr[b = 1 : b \leftarrow A^{H\setminus\mathcal{S}}(z)]\right| \leq 2\sqrt{(d+1)\cdot \Pr[\text{Find} : A^{H\setminus\mathcal{S}}(z)]}.$$

The following theorem gives an upper bound for the probability that Find occurs.

Theorem 4 ([AHU19]). *Let $\mathcal{S} \subseteq X$ and $z \in \{0,1\}^*$. And $\mathcal{S}, z$ may have arbitrary joint distribution. Let A be a quantum oracle algorithm making at most d queries to $O_{\mathcal{S}}^{SC}$ with domain $\mathcal{X}$. Let B be an algorithm that on input z chooses $i \overset{\$}{\leftarrow} \{1,\dots,d\}$, runs $A^{\mathcal{O}_{\varnothing}^{SC}}(z)$ until (just before) the i-th query, and then measures all query input registers in the computational basis. Denote by $\mathcal{T}$ the set of measurement outcomes. Then*

$$\Pr\left[\text{Find} : A^{\mathcal{O}_{\mathcal{S}}^{SC}}(z)\right] \leq 4d \cdot \Pr[\mathcal{S} \cap \mathcal{T} \neq \varnothing : \mathcal{T} \leftarrow B(z)].$$

3 Preimage Extraction of the Oracle-Masked Scheme

In this section, we start by the formalization of a class of PKE scheme Π named the oracle-masked scheme. Then we will introduce preimage extraction game $\text{Game}_{A,\Pi}^{\text{Ext}}$ for adversary A, and end this section with a theorem that bounds the difference between $\Pr[\text{Game}_{A,\Pi}^{\text{IND-qCCA}} \to 1]$ and $\Pr[\text{Game}_{A,\Pi}^{\text{Ext}} \to 1]$, where the definition of IND-qCCA security game $\text{Game}_{A,\Pi}^{\text{IND-qCCA}}$ is shown in the Appendix A.

Definition 5. *Let $\mathcal{O}$ be the random oracle with codomain $\mathcal{Y}$, let $\Pi = (\text{Gen}, \text{Enc}^{\mathcal{O}}, \text{Dec}^{\mathcal{O}})$ be a PKE scheme relative to $\mathcal{O}$. We say that Π is an oracle-masked scheme if there exist deterministic polynomial time algorithm A_1, A_2, A_3, A_4 such that for any $(pk, sk) \leftarrow \text{Gen}$, $\text{Enc}^{\mathcal{O}}$ and $\text{Dec}^{\mathcal{O}}$ are written in Fig 1a and Fig 1b, respectively. And tuple $(\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4)$ is called the decomposition of Π.*

```
$\text{Enc}^{\mathcal{O}}(pk, m; r)$
1: $x \leftarrow \mathrm{A}_1(pk, m, r)$
2: $y := \mathcal{O}(x)$
3: $c \leftarrow \mathrm{A}_2(pk, x, y)$
4: return $c$
```

(a) Algorithm $\text{Enc}^{\mathcal{O}}$ using A_1 and A_2

```
$\text{Dec}^{\mathcal{O}}(sk, c)$
1: $x \leftarrow \mathrm{A}_3(sk, c)$
2: if $x = \perp$, return $\perp$
3: else $y := \mathcal{O}(x)$
4: $c' \leftarrow \mathrm{A}_2(pk, x, y)$
5: if $c \neq c'$, return $\perp$
6: else $m \leftarrow \mathrm{A}_4(x)$
7: return $m$
```

(b) Algorithm $\text{Dec}^{\mathcal{O}}$ using A_2, A_3 and A_4

Figure 1: Algorithm $\text{Enc}^{\mathcal{O}}$ and $\text{Dec}^{\mathcal{O}}$ of an oracle-masked scheme Π

For oralce-masked scheme Π, parameter η of Π is defined to be

$$\eta := \max_{(pk,sk),\, c} \left|\{y \in \mathcal{Y} : c = \mathrm{A}_2\left(pk, \mathrm{A}_3(sk, c), y\right)\}\right| / |\mathcal{Y}|$$

where (pk, sk) is generated by Gen *and $c \in \mathcal{C}$ satisfies $\mathrm{A}_3(sk, c) \neq \perp$.*

Let Π be an oracle-masked scheme. For quantum adversary A in the security game $\text{Game}_{A,\Pi}^{\text{IND-qCCA}}$ in the QROM, it can query random oracle $\mathcal{O}$ and decryption oracle $\text{Dec}^{\mathcal{O}}$ both in superposition. Write C and Z to denote the input and output register of the decryption query of A, respectively. Then we introduce a new game $\text{Game}_{A,\Pi}^{\text{Sim}}$, that is identical with $\text{Game}_{A,\Pi}^{\text{IND-qCCA}}$ except that random oracle $\mathcal{O}$ is simulated by CStO. In game $\text{Game}_{A,\Pi}^{\text{Sim}}$, quantum queries to oracle $\mathcal{O}$ are recorded in the database register D.

The decryption oracle $\mathrm{Dec}_{sk}^{\mathcal{O}}(\cdot)$ in $\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}$ can be simulated by a unitary operator $\mathrm{U}_{\mathrm{Dec}}$ applied on register C and Z, i.e., for any computational basis state $|c,z\rangle$, $\mathrm{U}_{\mathrm{Dec}}$ acts as follows:

$$\mathrm{U}_{\mathrm{Dec}}|c,z\rangle = \begin{cases} |c, z\oplus\perp\rangle & \text{if } c^* \text{ is defined and } c=c^*,\\ |c, z\oplus \mathrm{Dec}_{sk}^{\mathcal{O}}(c)\rangle & \text{else.}\end{cases}$$

where c^* is the challenge ciphertext in $\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}$. In $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$, the decryption oracle answers queries in the same process as shown in Fig 1b. Since oracle $\mathcal{O}$ is simulated by CStO, the decryption process in this game can be simulated by a unitary operator applied on register C, Z, D. We denote this operator by $\mathrm{U}_{\mathrm{Sim}}$. Then by Theorem 2, $\mathrm{U}_{\mathrm{Dec}}$ and $\mathrm{U}_{\mathrm{Sim}}$, these two simulations of the decryption oracle are perfectly indistinguishable for any quantum adversary.

Notice that in the execution of algorithm Dec, A_3 is computed first to obtain x and then A_2 is applied to check if $c = \mathrm{A}_2(pk, x, \mathcal{O}(x))$. Therefore, the query x to oracle $\mathcal{O}$ must be recorded in the database D when $\mathrm{U}_{\mathrm{Sim}}$ is used to simulate the decryption oracle. Now we design a new unitary to reply decryption queries, and it is defined as follows.

Definition 6 (Preimage Extraction Procedure)**.** *Let Π be an oracle-masked scheme, and $(\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4)$ be its decomposition. For any (pk, sk) of Π, define unitary operation $\mathrm{U}_{\mathrm{Ext}}$, as the preimage extraction procedure of Π, applied on register C, Z, D as follows.*

$\mathrm{U}_{\mathrm{Ext}}|c,z,D\rangle$:

1. *If the challenge ciphertext c^* is defined and $c = c^*$, return $|c, z\oplus\perp, D\rangle$.*
2. *Else if database D contains no pair $(x, D(x))$ such that $\mathrm{A}_2(pk, x, D(x)) = c$, return $|c, z\oplus\perp, D\rangle$.*
3. *Else, for each tuple $(x, D(x))$ satisfying $\mathrm{A}_2(pk, x, D(x)) = c$, check if $\mathrm{A}_3(sk, c) = x$ and do the following procedure:*
   - *(a) If a tuple $(x, D(x))$ passes this check[1], compute $m = \mathrm{A}_4(x)$ and return $|c, z\oplus m, D\rangle$.*
   - *(b) Otherwise, return $|c, z\oplus\perp, D\rangle$.*

The implementation of $\mathrm{U}_{\mathrm{Ext}}$ is shown in Appendix B.

Compared with $\mathrm{U}_{\mathrm{Sim}}$, $\mathrm{U}_{\mathrm{Ext}}$ extracts the preimage x of the function, that is determined by A_2 and pk, from the database D and does not need to compute the inverse function A_3 at first. And thus we call $\mathrm{U}_{\mathrm{Ext}}$ a preimage extraction procedure. Then, for any oracle-masked scheme, a preimage extraction procedure $\mathrm{U}_{\mathrm{Ext}}$ exists, and it can be used to reply to quantum decryption queries.

By the definition of $\mathrm{U}_{\mathrm{Ext}}$, for any computational basis state $|c,z,D\rangle$, $\mathrm{U}_{\mathrm{Ext}}$ has no effect on $|D\rangle$, and does not need to query to oracle $\mathcal{O}$. Here we define a new game $\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}}$ named preimage extraction game that differs from $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$ in the way to answer decryption queries: In game $\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}}$, the decryption procedure is implemented by preimage extraction procedure unitary $\mathrm{U}_{\mathrm{Ext}}$ while that in game $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$ is implemented by unitary $\mathrm{U}_{\mathrm{Sim}}$.

Now we introduce two properties of $\mathrm{U}_{\mathrm{Ext}}$ by the following lemmas, and their detailed proofs are shown in Appendix C.

Lemma 7. *Fix an oracle-masked scheme and a public/secret key pair. Let $|\psi\rangle$ be a quantum state on register C, Z, D, and $|\psi\rangle$ is orthogonal to state $\sum_{c,z,D,x}\alpha_{c,z,D,x}|c,z,D\cup(x,\beta_0)\rangle$. Then*

$$\|(\mathrm{U}_{\mathrm{Sim}} - \mathrm{U}_{\mathrm{Ext}})|\psi\rangle\| \le 5\sqrt{\eta}.$$

Lemma 8. *Given any $x\in\{0,1\}^*$, unitary $\mathrm{StdDecomp}_x$ is performed on register D. For any quantum state $|\psi\rangle$ on register C, Z and D, we have*

$$\left\|(\mathrm{U}_{\mathrm{Ext}}\circ \mathsf{StdDecomp}_x - \mathsf{StdDecomp}_x\circ \mathrm{U}_{\mathrm{Ext}})|\psi\rangle\right\| \le 7\sqrt{\eta}.$$

Then we present Theorem 9 to bound the difference of the output distribution of game $\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}$ and $\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}}$.

[1] Such a tuple is unique, since c determines the value of $P_3(sk,c)$.

Theorem 9. *Let Π be an oracle-masked scheme. For any quantum adversary A against the IND-qCCA security of Π in the QROM, if A makes at most q decryption queries, then*

$$\left|\Pr[\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}} \to 1] - \Pr[\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}} \to 1]\right| \le 5q \cdot \sqrt{\eta}.$$

Proof. Given Π and A, recall that game $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$ is identical with game $\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}$ except that the random oracle is simulated by CStO. By Theorem 2,

$$\Pr[\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}} \to 1] = \Pr[\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}} \to 1].$$

In the following, we prove that

$$\left|\Pr[\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}} \to 1] - \Pr[\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}} \to 1]\right| \le 5q \cdot \sqrt{\eta}.$$

For any fixed (pk, sk), the decryption procedure in game $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$ and that in game $\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}}$ are implemented by unitary $\mathrm{U}_{\mathsf{Sim}}$ and $\mathrm{U}_{\mathrm{Ext}}$, respectively.

For any $i = 1, \ldots, q$, define G_i to be a game that is the same as $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$ until just before the i-th decryption query of A, then replaces decryption procedure $\mathrm{U}_{\mathsf{Sim}}$ with $\mathrm{U}_{\mathrm{Ext}}$. Then game G_1 is game $\mathrm{Game}_{A,\Pi}^{\mathrm{Ext}}$ for A. We also denote game $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$ by G_{q+1}. For $i = 1, \ldots, q+1$, denote by σ_i the final joint state on the register of A and the database register in G_i. By the triangle inequality of the trace distance, $D(\sigma_1, \sigma_{q+1}) \le \sum_{i=1}^{q} D(\sigma_i, \sigma_{i+1})$.

Fix $1 \le i \le q$. Denote by ρ the joint state of A and the database register just before the i-th decryption query. All the operations after i-th decryption query can be represented by a trace-preserving operation, that is denoted by $\mathcal{E}$. Since game G_i and G_{i+1} only differ in the i-th decryption procedure, σ_i and σ_{i+1}, can be represented by $\sigma_i = \mathcal{E}(\mathrm{U}_{\mathsf{Sim}}\, \rho \mathrm{U}_{\mathsf{Sim}}^{\dagger})$ and $\sigma_{i+1} = \mathcal{E}(\mathrm{U}_{\mathrm{Ext}}\, \rho \mathrm{U}_{\mathrm{Ext}}^{\dagger})$, respectively. And we have $D(\sigma, \sigma_{i+1}) \le D(\mathrm{U}_{\mathsf{Sim}}\, \rho \mathrm{U}_{\mathsf{Sim}}^{\dagger}, \mathrm{U}_{\mathrm{Ext}}\, \rho \mathrm{U}_{\mathrm{Ext}}^{\dagger})$.

Here we give an upper bound for $D(\mathrm{U}_{\mathsf{Sim}}\, \rho \mathrm{U}_{\mathsf{Sim}}^{\dagger}, \mathrm{U}_{\mathrm{Ext}}\, \rho \mathrm{U}_{\mathrm{Ext}}^{\dagger})$. Let $\sum_j p_j |\psi_j\rangle\langle\psi_j|$ be a spectral decomposition of ρ. Then by the convexity of the trace distance,

$$\begin{aligned} D(\mathrm{U}_{\mathsf{Sim}}\, \rho \mathrm{U}_{\mathsf{Sim}}^{\dagger}, \mathrm{U}_{\mathrm{Ext}}\, \rho \mathrm{U}_{\mathrm{Ext}}^{\dagger}) &= D\Big(\sum_j p_j \mathrm{U}_{\mathsf{Sim}} |\psi_j\rangle\langle\psi_j| \mathrm{U}_{\mathsf{Sim}}^{\dagger}, \sum_j p_j \mathrm{U}_{\mathrm{Ext}} |\psi_j\rangle\langle\psi_j| \mathrm{U}_{\mathrm{Ext}}^{\dagger}\Big) \\ &\le \sum_j p_j D(\mathrm{U}_{\mathsf{Sim}} |\psi_j\rangle\langle\psi_j| \mathrm{U}_{\mathsf{Sim}}^{\dagger}, \mathrm{U}_{\mathrm{Ext}} |\psi_j\rangle\langle\psi_j| \mathrm{U}_{\mathrm{Ext}}^{\dagger}) \le \sum_j p_j \|(\mathrm{U}_{\mathsf{Sim}} - \mathrm{U}_{\mathrm{Ext}})|\psi_j\rangle\|. \end{aligned}$$

Note that before the i-th decryption query, the decryption procedure is $\mathrm{U}_{\mathsf{Sim}}$ and A is simulated to be in game $\mathrm{Game}_{A,\Pi}^{\mathrm{Sim}}$. Thus any pure state $|\psi_j\rangle$ in the spectral decomposition of ρ is in the form of the superposition state in Lemma 7. Then by Lemma 7, $\|(\mathrm{U}_{\mathsf{Sim}} - \mathrm{U}_{\mathrm{Ext}})|\psi_j\rangle\| \le 5\sqrt{\eta}$. Therefore, we have

$$D(\sigma_i, \sigma_{i+1}) \le \sum_j p_j \cdot \|(\mathrm{U}_{\mathsf{Sim}} - \mathrm{U}_{\mathrm{Ext}})|\psi_j\rangle\| \le \sum_j p_j \cdot 5\sqrt{\eta} = 5\sqrt{\eta}$$

and $D(\sigma_1, \sigma_{q+1}) \le \sum_{i=1}^{q} D(\sigma_i, \sigma_{i+1}) \le 5q \cdot \sqrt{\eta}$. This completes the proof. □

4 Application in the Quantum Security Proof

In this section, we apply Theorem 9 of the oracle-masked scheme to provide the IND-qCCA security proof for FO and REACT transformation in the QROM, respectively.

4.1 FO: from OW-CPA to IND-qCCA in the QROM

The FO transformation combines a one-time (OT) secure secret-key encryption(SKE) scheme with a well-spread OW-CPA secure PKE scheme to obtain an IND-CCA secure PKE in the random oracle model. The definitions of the OW-CPA security for PKE, the well-spread property of PKE and the OT security for SKE are given in Appendix A.

Let $\Pi^{asy} = (\mathrm{Gen}^{asy}, \mathrm{Enc}^{asy}, \mathrm{Dec}^{asy})$ be a PKE with message space $\mathcal{M}^{asy}$, randomness space $\mathcal{R}^{asy}$ and ciphertext space $\mathcal{C}^{asy}$. Let $\Pi^{sy} = (\mathrm{Enc}^{sy}, \mathrm{Dec}^{sy})$ be a symmetric encryption scheme with key space $\mathcal{K}^{sy}$, message space $\mathcal{M}^{sy}$ and ciphertext space $\mathcal{C}^{sy} = \{0,1\}^n$. Let H and G be hash functions such that

$$H : \{0,1\}^* \to \mathcal{R}^{asy}, \quad G : \{0,1\}^* \to \mathcal{K}^{sy}.$$

These hash functions are modeled as random oracles in the following security proof.

Definition 10. *The hybrid scheme* $\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]=(\mathrm{Gen},\mathrm{Enc},\mathrm{Dec})$ *obtained from the FO transformation is constructed as follows.*

1. Gen*: The Key Generation algorithm runs* Gen^{asy} *and takes its output* (pk,sk) *as a public key/secret key-pair.*
2. Enc*: The Encryption algorithm on input* pk *and message* m *picks* $\delta\in\mathcal{M}^{asy}$ *uniformly, computes*
$$d=\mathrm{Enc}^{sy}_{G(\delta)}(m),\quad c=\mathrm{Enc}^{asy}_{pk}(\delta;H(\delta,d))$$
and outputs (c,d) *as a ciphertext.*
3. Dec*: The Decryption algorithm on input* sk *and ciphertext* (c,d) *computes*
$$\delta=\mathrm{Dec}^{asy}_{sk}(c),\quad c'=\mathrm{Enc}^{asy}_{pk}(\delta;H(\delta,d)).$$
If $c'=c$*, compute* $m=\mathrm{Dec}^{sy}_{G(\delta)}(d)$ *and output* m*. Otherwise, output* $\perp$*.*

Lemma 11. *Assume that* H *is the random oracle and* Π^{asy} *is* γ*-spread, then* $\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]=(\mathrm{Gen},\mathrm{Enc}^H,\mathrm{Dec}^H)$ *is an oracle-masked scheme with* $\eta\le 1/2^\gamma$*.*

Proof. We define deterministic polynomial-time algorithm A_1, A_2, A_3 and A_4 as follows.

- A_1 takes δ and m as input, evaluates $k=G(\delta)$ and $d=\mathrm{Enc}^{sy}_k(m)$, then outputs (δ,d).
- A_2 takes pk, (δ,d) and $y\in\mathcal{R}^{asy}$ as input, computes $c=\mathrm{Enc}^{asy}_{pk}(\delta;y)$, then outputs (c,d).
- A_3 takes sk and (c,d) as input, evaluates $\delta=\mathrm{Dec}^{asy}_{sk}(c)$. If $\delta\ne\perp$, output (δ,d). Otherwise, output $\perp$.
- A_4 takes (δ,d) as input, computes $k=G(\delta)$ and $m=\mathrm{Dec}^{sy}_k(d)$, then outputs m.

It can be verified that with the above four algorithms, Enc^H and Dec^H are written as in Definition 5. Then $\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]$ is an oracle-masked scheme.

By the definition of η, we obtain η of $\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]$ as below.
$$\eta=\max_{(pk,sk),\,c}\left|\{r\in\mathcal{R}^{asy}:\ c=\mathrm{Enc}^{asy}_{pk}(\mathrm{Dec}^{asy}_{sk}(c);r)\}\right|/|\mathcal{R}^{asy}|$$
where (pk,sk) is generated by Gen and $c\in\mathcal{C}^{asy}$ satisfies $\mathrm{Dec}^{asy}_{sk}(c)\notin\perp$, i.e., $\mathrm{Dec}^{asy}_{sk}(c)\in\mathcal{M}^{asy}$.

On the other hand, for any (pk,sk) generated by Gen and $m\in\mathcal{M}^{asy}$, we have
$$\max_{c\in\mathcal{C}^{asy}}\left|\{r\in\mathcal{R}^{asy}:\ c=\mathrm{Enc}^{asy}_{pk}(m;r)\}\right|/|\mathcal{R}^{asy}|\le 1/2^\gamma$$
due to the γ-spread property of Π^{asy}. Therefore, $\eta\le 1/2^\gamma$.

□

Note that hash function G can be replaced with any function, which is accessed by an oracle. Then algorithm A_1 and A_4 become oracle algorithms denoted by A_1^G and A_4^G, respectively. In this case, the notions in Definition 5 still work, and Theorem 9 holds.

As shown in Lemma 11, the hybrid scheme $\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]$ obtained from the FO transformation is an oracle-masked scheme. Then we use Theorem 9 of the oracle-masked scheme to prove the IND-qCCA security of $\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]$.

Theorem 12. *Assume that* Π^{asy} *is* γ*-spread, for any adversary against the* IND-qCCA *security of scheme* $\Pi=\mathrm{FO}[\Pi^{asy},\Pi^{sy},H,G]$*, making at most* q_D *quantum queries to the decryption oracle, at most* q_H *quantum queries to random oracle* H *and at most* q_G *quantum queries to random oracle* G*, there exists an adversary* A_{asy} *against the* OW-CPA *security of* Π^{asy} *and an adversary* A_{sy} *against the* OT *security of* Π^{sy} *such that*
$$\mathrm{Adv}^{\text{IND-qCCA}}_{A,\Pi}\le q_D\cdot\frac{12}{\sqrt{2^\gamma}}+2(d+1)\cdot\sqrt{\mathrm{Adv}^{\text{OW-CPA}}_{A_{asy},\Pi^{asy}}}+4d\cdot\mathrm{Adv}^{\text{OW-CPA}}_{A_{asy},\Pi^{asy}}+\mathrm{Adv}^{\text{OT}}_{A_{sy},\Pi^{sy}}$$
where $d=q_D+q_H+2q_G$*,* $\mathrm{Time}(A_{sy})=\mathrm{Time}(A_{asy})\approx\mathrm{Time}(A)+O\left(d^2+q_H\cdot q_D\cdot\mathrm{Time}(\mathrm{Enc}^{asy})\right)$*.*

Proof. Denote by Ω_H and Ω_G sets of all functions $H : \{0,1\}^* \to \mathcal{R}^{asy}$ and $G : \{0,1\}^* \to \mathcal{K}^{sy}$, respectively. The input and output register of the decryption query of A are denoted by C and Z, respectively. Define **Game 0** to be game $\text{Game}^{\text{IND-qCCA}}_{A,\Pi}$. In the following, we will introduce a sequence of games to bound the difference between $1/2$ and the probability that **Game 0** outputs 1. By the definition of **Game 0**, we obtain

$$\left|\Pr[\textbf{Game 0} \to 1] - \frac{1}{2}\right| = \text{Adv}^{\text{IND-qCCA}}_{A,\Pi}. \tag{1}$$

```
Game 0:
1: $G \xleftarrow{\$} \Omega_G$, $H \xleftarrow{\$} \Omega_H$, $\delta^* \xleftarrow{\$} \mathcal{M}^{asy}$, $(pk, sk) \leftarrow \text{Gen}$
2: $(m_0, m_1) \leftarrow A^{H,G,\text{Dec}_{sk}}(pk)$
3: $b \xleftarrow{\$} \{0,1\}$, $d^* = \text{Enc}^{sy}_{G(\delta^*)}(m_b)$, $c^* = \text{Enc}^{asy}_{pk}(\delta^*; H(\delta^*, d^*))$
4: $b' \leftarrow A^{H,G,\text{Dec}^*_{sk}}(pk, c^*)$
5: Return $[b = b']$
```

From **Game 1** begin, random oracle H is simulated with CStO and its database register is denoted as D. This change is undetectable for A by Theorem 2.

Game 1: In this game, we replace the decryption oracle in **Game 0** with oracle CCA_1. When to reply to decryption queries, CCA_1 performs the preimage extraction procedure of Π. The construction of CCA_1 is shown in Appendix B.1.

It is obvious that **Game 1** is the preimage extraction game $\text{Game}^{\text{Ext}}_{A,\Pi}$. Then by Theorem 7, we obtain $\big|\Pr[\textbf{Game 0} \to 1] - \Pr[\textbf{Game 1} \to 1]\big| \le 5q_D \cdot \sqrt{\eta}$ for any fixed $G \in \Omega_G$. Therefore,

$$\big|\Pr[\textbf{Game 0} \to 1] - \Pr[\textbf{Game 1} \to 1]\big| \le 5q_D \cdot \sqrt{\eta} \tag{2}$$

where variable G both in **Game 0** and **Game 1** are sampled from Ω_G, uniformly.

Note that as presented in Appendix B.1, the preimage extraction procedure of Π and CCA_1 is constructed without sk. Starting from **Game 1**, sk is no longer used to reply to decryption queries.

Game 2: We change oracle CCA_1 by CCA_2. CCA_2 is defined as follows. Before the challenge query, CCA_2 acts the same as CCA_1. After that, CCA_2 answers the decryption query in three steps:

1. Perform unitary $\mathsf{StdDecomp}_{(\delta^*,d^*)}$ to register D.
2. Apply CCA_1 on register C, Z and D.
3. Apply $\mathsf{StdDecomp}_{(\delta^*,d^*)}$ to register D a second time.

By the definition of CCA_2, we only analyze the difference between CCA_1 and CCA_2 in the case that the challenge ciphertext has been determined. Observe that if we flip the order of the last two steps of CCA_2, then $\mathsf{StdDecomp}_{(\delta^*,d^*)} \circ \mathsf{StdDecomp}_{(\delta^*,d^*)}$ is an identity operator and in this way, CCA_2 performs identically as CCA_1.

Then by Lemma 8, for any joint state ρ of A and the database, $D(\text{CCA}_1\rho\text{CCA}_1^\dagger, \text{CCA}_2\rho\text{CCA}_2^\dagger) \le 7\sqrt{\eta} \le 7/\sqrt{2^\gamma}$. At most q_D decryption queries are made after the challenge query, and then by the hybrid argument,

$$|\Pr[\textbf{Game 1} \to 1] - \Pr[\textbf{Game 2} \to 1]| \le q_D \cdot \frac{7}{\sqrt{2^\gamma}}. \tag{3}$$

Game 3: In this game, we change the way to answer random oracle queries in some cases. Let E be a constant zero function with quantum access. When random oracle H or G is queried by A or G is applied in the process of CCA_2, we perform E and then perform the random oracle.

Since E is a constant zero function, the random oracle query does not change after performing E, and we have

$$\Pr[\textbf{Game 2} \to 1] = \Pr[\textbf{Game 3} \to 1]. \tag{4}$$

Game 4: In this game, the only change is that semi-classical oracle $O^{SC}_{\mathcal{S}}$ is applied before each query to E, and set $\mathcal{S} := \{\delta^*, \delta^*\|\cdot\}$.

Let $z = \delta^*$, and $B^E(\delta^*)$ be the algorithm that simulates **Game 3**. Then we have

$$\Pr[\textbf{Game 3} \to 1] = \Pr[b = 1 : b \leftarrow B^E(\delta^*), \delta^* \xleftarrow{\$} \mathcal{M}^{asy}],$$
$$\Pr[\textbf{Game 4} \to 1] = \Pr[b = 1 : b \leftarrow B^{E\setminus\mathcal{S}}(\delta^*), \delta^* \xleftarrow{\$} \mathcal{M}^{asy}],$$
$$\Pr[\text{Find} : \textbf{Game 4}] = \Pr[\text{Find} : B^{E\setminus\mathcal{S}}(\delta^*), \delta^* \xleftarrow{\$} \mathcal{M}^{asy}].$$

B makes at most $d = q_H + q_G + 2q_D$ queries to E.

- E is queried each time H or G is queried.
- A makes at most q_H queries to H and at most q_G queries to G.
- A makes at most q_D decryption queries, and CCA$_3$ requires 2 queries to G for each decryption query (c, d): If $((\delta_0, d_0), y) \in D$ satisfies $d_0 = d$ and $\text{Enc}^{asy}_{pk}(\delta_0; y) = c$, one query to G is needed when computing $\text{Dec}^{sy}_{G(\delta_0)}(d)$, and another one query is to uncompute $G(\delta_0)$.

Let $d = q_H + q_G + 2q_D$, then we apply Theorem 3 to obtain

$$|\Pr[\textbf{Game 3} \to 1] - \Pr[\textbf{Game 4} \to 1]| \leq 2\sqrt{(d+1)\Pr[\text{Find} : \textbf{Game 4}]}. \tag{5}$$

Furthermore, the database state on (δ^*, d^*) is not disturbed by the decryption process of CCA$_2$ until Find occurs, which is analyzed as follows. Notice that if Find does not occur, then A has never queried (δ^*, d^*) to H, which implies that the database of H does not record (δ^*, d^*) until Find occurs or H is queried by (δ^*, d^*) when producing the challenge ciphertext. And we only care about the latter situation: It is apparent that the database state on (δ^*, d^*) is not disturbed by CCA$_2$ before the challenge query. Then we argue that the database state on (δ^*, d^*) is not disturbed by CCA$_2$ after the challenge query as follows.

For decryption query (c, d) after the challenge query, CCA$_2$ outputs $\perp$ if $d \neq d^*$. Then we assume $d = d^*$. In this case, CCA$_2$ computes the value of $H(\delta^*, d^*)$ through unitary $\text{StdDecomp}_{(\delta^*,d^*)}$ and tests if $\text{Enc}^{asy}_{pk}(\delta^*; H(\delta^*, d^*)) = c$. However, by the correctness of Π^{asy}, $\text{Enc}^{asy}_{pk}(\delta^*; H(\delta^*, d^*)) = c$ only when $c = c^*$, in which case CCA$_2$ outputs $\perp$.

Game 5: In this game, oracle CCA$_2$ is replaced with oracle CCA$_3$. CCA$_3$ is defined as follows. Before the challenge query, CCA$_3$ acts the same as CCA$_1$. After that, CCA$_3$ performs the preimage extraction procedure of Π but the database state on (δ^*, d^*) is not involved in this procedure.

Before the challenge query, oracle CCA$_2$ and CCA$_3$ act the same by their definitions. In the following, we consider the case when the challenge query has happened.

After the challenge query, (δ^*, d^*) is recorded in the database. In the process of CCA$_2$, $H(\delta^*, d^*)$ is computed to test if $\text{Enc}^{asy}_{pk}(\delta^*; H(\delta^*, d^*)) = c$. But as analyzed in **Game 4**, $H(\delta^*, d^*)$ fails that test in each process of CCA$_2$. Therefore, unitary $\mathsf{StdDecomp}_{(\delta^*,d^*)}$ in the first and last step of CCA$_2$ and that test can be removed from the process of CCA$_2$ and the modified CCA$_2$ has the same effect as CCA$_2$. By the definition of CCA$_3$, the modified CCA$_2$ is exactly oracle CCA$_3$ and thus we obtain

$$\Pr[\text{Find} : \textbf{Game 4}] = \Pr[\text{Find} : \textbf{Game 5}], \tag{6}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 4} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 5} \to 1]. \tag{7}$$

Moreover, it is clear that the database state on (δ^*, d^*) will not be disturbed by the decryption procedure of CCA$_3$.

Game 6: In this game, we pick $k^* \in \mathcal{K}^{sy}$ and $r^* \in \mathcal{R}^{asy}$ uniformly and use them to compute $d^* = \text{Enc}^{sy}_{k^*}(m_b)$ and $c^* = \text{Enc}^{asy}_{pk}(\delta^*; r^*)$ when producing the challenge ciphertext (c^*, d^*).

In **Game 5**, $H(\delta^*, d^*)$ and $G(\delta^*)$ is used to produce (c^*, d^*), and they are replaced by k^* and r^* in **Game 6**. If Find does not occur, then random oracle G has never been queried by δ^* except for producing the challenge ciphertext and thus $G(\delta^*)$ is uniformly random in A's view. $H(\delta^*, d^*)$ is uniformly random in A's view for two reasons: Firstly, the fact that Find does not occur means that A has never queried (δ^*, d^*) to H. Secondly, the database state on (δ^*, d^*) is not disturbed by the decryption process of CCA$_3$. Therefore, it is undetectable for adversary A to produce the challenge ciphertext with uniformly chosen $r_1 \in \mathcal{K}^{sy}$ and $r_2 \in \mathcal{R}^{asy}$. Then the view of A in **Game 5** and that in **Game 6** are identical until Find occurs. Thus, we have

$$\Pr[\text{Find} : \textbf{Game 5}] = \Pr[\text{Find} : \textbf{Game 6}], \tag{8}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 5} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 6} \to 1]. \tag{9}$$

Game 7: In this game, we change oracle CCA_3 back to CCA_1.

If Find does not occur, (δ^*, d^*) is not recorded in the database of H. In this case, oracle CCA_1 and CCA_3 act the same by their definitions. Therefore, **Game 6** and **Game 7** are perfectly indistinguishable for A until Find happens. Thus,

$$\Pr[\text{Find} : \textbf{Game 6}] = \Pr[\text{Find} : \textbf{Game 7}], \tag{10}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 6} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 7} \to 1]. \tag{11}$$

Then by equation (6), (7), (8), (9), (10) and (11),

$$\Pr[\text{Find} : \textbf{Game 4}] = \Pr[\text{Find} : \textbf{Game 7}], \tag{12}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 4} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 7} \to 1]. \tag{13}$$

By equation (13),

$$|\Pr[\textbf{Game 4} \to 1] - \Pr[\textbf{Game 7} \to 1]| = |\Pr[\text{Find} \wedge \textbf{Game 4} \to 1] - \Pr[\text{Find} \wedge \textbf{Game 7} \to 1]| .$$

Combines this equation with equation (12), and we obtain

$$|\Pr[\textbf{Game 4} \to 1] - \Pr[\textbf{Game 7} \to 1]| \leq \Pr[\text{Find} : \textbf{Game 7}]. \tag{14}$$

Lemma 13. *There exists a quantum adversary A_{sy} invoking A such that*

$$\left|\Pr[\textbf{Game 7} \to 1] - \frac{1}{2}\right| = \text{Adv}^{\text{OT}}_{A_{sy},\Pi^{sy}} \tag{15}$$

and $\text{Time}(A_{sy}) \approx \text{Time}(A) + O((q_H + q_G + 2q_D)^2 + q_H \cdot q_D \cdot \text{Time}(\text{Enc}^{asy}))$.

Proof. A quantum algorithm A_{sy} that runs A and breaks the one-time security of Π^{sy} is constructed as follows:

A_{sy} generates $(pk, sk) \leftarrow \text{Gen}$, picks $\delta^* \xleftarrow{\$} \mathcal{M}^{asy}$ and simulates **Game 7** for A. Random oracle G is simulated by a $2(q_G + 2q_D)$-wise independent function, and other oracles used in **Game 7** can be implemented efficiently by A_{sy}. For A's challenge query (m_0, m_1), A_{sy} sends it to the challenger in $\text{Game}^{\text{OT}}_{A_{sy},\Pi^{sy}}$. After receiving d^*, A_{sy} picks $r \in \mathcal{R}^{asy}$ uniformly, then computes $c^* = \text{Enc}^{asy}_{pk}(\delta^*; r)$ and sends (c^*, d^*) back to A. After receiving b' from A, A_{sy} output b'.

From the construction of A_{sy}, the view of A invoked by A_{sy} is identical with that in **Game 7**, and the output of A_{sy} is correct if and only if A guesses correctly. Thus we have $\Pr[\text{Game}^{\text{OT}}_{A_{sy},\Pi^{sy}} \to 1] = \Pr[\textbf{Game 7} \to 1]$ and

$$\left|\Pr[\textbf{Game 7} \to 1] - \frac{1}{2}\right| = \left|\Pr[\text{Game}^{\text{OT}}_{A_{sy},\Pi^{sy}} \to 1] - \frac{1}{2}\right| = \text{Adv}^{\text{OT}}_{A_{sy},\Pi^{sy}}.$$

Denote by $\text{T}_{\mathcal{O}}$ the time needed to simulate oracle $\mathcal{O}$, then the running time of B is given by $\text{Time}(B) = \text{Time}(A) + \text{T}_G + \text{T}_H + \text{T}_{\text{CCA}_1}$, where $\text{T}_G = O\left((q_G + 2q_D)^2\right)$, $\text{T}_H = O(q_H^2)$, $\text{T}_{\text{CCA}_1} = O(q_D \cdot q_H \cdot \text{Time}(\text{Enc}^{asy}))$ by Appendix B.1.

□

Lemma 14. *There exists a quantum adversary A_{asy} invoking A such that*

$$\Pr[\text{Find} : \textbf{Game 7}] \leq 4d \cdot \text{Adv}^{\text{OW-CPA}}_{A_{asy},\Pi^{asy}} \tag{16}$$

and $\text{Time}(A_{asy}) \approx \text{Time}(A) + O((q_H + q_G + 2q_D)^2 + q_H \cdot q_D \cdot \text{Time}(\text{Enc}^{asy}))$.

Proof. Define $B^{\mathcal{O}_{\mathcal{S}}^{SC}}$ as a quantum oracle algorithm that on input pk, c^*, runs A and simulates **Game 7** for it. Then we have $\Pr[\text{Find} : \textbf{Game 7}] = \Pr[\text{Find} : B^{\mathcal{O}_{\mathcal{S}}^{SC}}(pk, c^*)]$, where $c^* = \text{Enc}_{pk}^{asy}(\delta^*)$, δ^* is sampled uniformly from $\mathcal{M}^{asy}$. As analyzed in **Game 5**, B makes at most $d = q_H + q_G + 2q_D$ queries, then by Theorem 4,

$$\Pr[\text{Find} : B^{\mathcal{O}_{\mathcal{S}}^{SC}}(pk, c^*)] \leq 4d \cdot \Pr[(\delta, d) \in \mathcal{S} : (\delta, d) \leftarrow D(pk, c^*)].$$

Here D is a quantum algorithm invoking B. On input (pk, c^*), D chooses $i \xleftarrow{\$} \{1, \ldots, d\}$, runs $B^{\mathcal{O}_{\varnothing}^{SC}}(pk, c^*)$ until (just before) i-th query of B, and then measures the state on the input register of $\mathcal{O}_{\varnothing}^{SC}$ and obtains (δ, d). Note that the running time of D and that of B are almost the same.

Because $\mathcal{S} = \{\delta^*, \delta^* \| \cdot\}$, $(\delta, d) \in \mathcal{S}$ is equivalent to $\delta = \delta^*$. Then D can also be considered as a quantum algorithm A_{asy} that breaks the OW-CPA security of Π^{asy}. Therefore,

$$\Pr[(\delta, d) \in \mathcal{S} : (\delta, d) \leftarrow D(pk, c^*)] = \Pr[\delta = \delta^* : (\delta, d) \leftarrow D(pk, c^*)] = \text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-CPA}}.$$

The running time of B is $\text{Time}(B) = \text{Time}(A) + \text{T}_G + \text{T}_H + \text{T}_{\text{CCA}_1}$, where $\text{T}_G = O\left((q_G + 2q_D)^2\right)$, $\text{T}_H = O(q_H^2)$, $\text{T}_{\text{CCA}_1} = O(q_D \cdot q_H \cdot \text{Time}(\text{Enc}^{asy}))$. □

Combining equation (5), (12), (14), (15) and (16), we obtain

$$\left|\Pr[\textbf{Game 3} \rightarrow 1] - \frac{1}{2}\right| \leq 2d \cdot \sqrt{\text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-CPA}}} + 4d \cdot \text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-CPA}} + \text{Adv}_{A_{sy}, \Pi^{sy}}^{\text{OT}}. \tag{17}$$

Summarizing equation (1), (2), (3), (4), and (17), we have

$$\text{Adv}_{A, \Pi}^{\text{IND-qCCA}} \leq q_D \cdot \frac{12}{\sqrt{2^\gamma}} + 2d \cdot \sqrt{\text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-CPA}}} + 4d \cdot \text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-CPA}} + \text{Adv}_{A_{sy}, \Pi^{sy}}^{\text{OT}}.$$

□

Furthermore, compared with Zhandry's proof for FO transform, we notice that oracle CCA_1 in this proof acts the same as the decryption procedure defined in Hybrid 4 in his proof for ciphertext (c, d) such that $c \neq c^*$. And by Theorem 9, we can prove that any polynomial time quantum adversary distinguishes Hybrid 1 from Hybrid 4 with a negligible probability. On the other hand, by equation (2), it seems unnecessary to restrict that the decryption oracle outputs $\perp$ directly for ciphertext (c, d) such that $c = c^*$. Thus Hybrid 1 can be removed from his reduction, which contributes a tighter security reduction.

4.2 REACT: from OW-qPCA to IND-qCCA in the QROM

Let $\Pi^{asy} = (\text{Gen}^{asy}, \text{Enc}^{asy}, \text{Dec}^{asy})$ be a PKE with key space $\mathcal{K}^{asy}$, message space $\mathcal{M}^{asy}$, randomness space $\mathcal{R}^{asy}$ and ciphertext space $\mathcal{C}^{asy}$. Let $\Pi^{sy} = (\text{Enc}^{sy}, \text{Dec}^{sy})$ be a symmetric encryption scheme with message space $\mathcal{M}^{sy}$, ciphertext space $\mathcal{C}^{sy}$, key space $\mathcal{K}^{sy}$. Let H and G be hash functions such that

$$H : \{0,1\}^* \rightarrow \{0,1\}^n, \quad G : \{0,1\}^* \rightarrow \mathcal{R}^{sy}.$$

Definition 15. *The hybrid scheme* $\text{REACT}[\Pi^{asy}, \Pi^{sy}, H, G] = (\text{Gen}, \text{Enc}, \text{Dec})$ *obtained from the REACT transformation is constructed as follows.*

1. Gen*: The Key Generation algorithm runs* Gen^{asy} *and takes its output* (pk, sk) *as a public/secret key pair.*

2. Enc*: The Encryption algorithm on input* pk *and* $m \in \mathcal{M}^{sy}$ *picks* $R \xleftarrow{\$} \mathcal{M}^{asy}$, $r \xleftarrow{\$} \mathcal{R}^{asy}$, *computes*

$$c_1 = \text{Enc}_{pk}^{asy}(R; r), \quad c_2 = \text{Enc}_{G(R)}^{sy}(m), \quad c_3 = H(R, m, c_1, c_2)$$

and outputs (c_1, c_2, c_3) *as a ciphertext.*

3. Dec*: The Decryption algorithm on input sk and* (c_1, c_2, c_3)*, computes*

$$R = \mathrm{Dec}^{asy}_{sk}(c_1), \quad m = \mathrm{Dec}^{sy}_{G(R)}(c_2), \quad c_3' = H(R, m, c_1, c_2).$$

If $R = \perp$ *or* $m = \perp$*, output* $\perp$*. Else if* $c_3' = c_3$*, output* m*. Else, output* $\perp$*.*

Lemma 16. *Let* H *be the random oracle, then* $\mathrm{REACT}[\Pi^{asy}, \Pi^{sy}, H, G] = (\mathrm{Gen}, \mathrm{Enc}^H, \mathrm{Dec}^H)$ *is an oracle-masked scheme with* $\eta = 1/2^n$*.*

Proof. We define deterministic polynomial time algorithm A_1, A_2, A_3 and A_4 as follows.

- A_1 takes pk, R, r and m as input, evaluates $c_1 = \mathrm{Enc}^{asy}_{pk}(R; r)$, $k = G(R)$, $c_2 = \mathrm{Enc}^{sy}_k(m)$, and then outputs (R, m, c_1, c_2).
- A_2 takes (R, m, c_1, c_2) and $y \in \{0,1\}^n$ as input, defines $c_3 := y$, then outputs (c_1, c_2, c_3).
- A_3 takes sk and (c_1, c_2, c_3) as input, computes $R = \mathrm{Dec}^{asy}_{sk}(c_1)$. If $R = \perp$, output $\perp$. Else, compute $k = G(R)$ and $m = \mathrm{Dec}^{sy}_k(c_2)$. If $m = \perp$, output $\perp$. Otherwise, output (R, m, c_1, c_2).
- A_4 takes (R, m, c_1, c_2) as input and outputs m.

We can verify that with four algorithms defined as above, Enc^H and Dec^H are written as in Definition 5. And thus Π is an oracle-masked scheme and tuple $(\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4)$ is Π's decomposition.

By the definition of η,

$$\begin{aligned}\eta &= \max_{(pk,sk),\,(c_1,c_2,c_3)} 1/2^n \cdot \left|\{y \in \{0,1\}^n : (c_1, c_2, c_3) = \mathrm{A}_2(pk, \mathrm{A}_3(c_1, c_2, c_3), y)\}\right| \\ &= \max_{(pk,sk),\,(c_1,c_2,c_3)} 1/2^n \cdot \left|\{y \in \{0,1\}^n : c_3 = y\}\right| = 1/2^n\end{aligned}$$

where (pk, sk) is generated by Gen and $(c_1, c_2, c_3) \in \mathcal{C}^{asy} \times \mathcal{C}^{sy} \times \{0,1\}^n$ satisfies $\mathrm{A}_3(sk, c_1, c_2, c_3) \neq \perp$. □

Theorem 17. *For any adversary* A *against the* IND-qCCA *security of* $\Pi = \mathrm{REACT}[\Pi^{asy}, \Pi^{sy}, H, G]$ *in the QROM, making at most* q_D *queries to the decryption oracle, at most* q_G *queries to random oracle* G *and at most* q_H *queries to random oracle* H*, there exists an adversary* A_{asy} *against the* OW-qPCA *security of* Π^{asy} *and an adversary* A_{sy} *against the* OT *security of* Π^{sy} *such that*

$$\mathrm{Adv}^{\text{IND-qCCA}}_{A,\Pi} \le q_D \cdot \frac{12}{\sqrt{2^n}} + 2(d+1) \cdot \sqrt{\mathrm{Adv}^{\text{OW-qPCA}}_{A_{asy},\Pi^{asy}}} + 4d \cdot \mathrm{Adv}^{\text{OW-qPCA}}_{A_{asy},\Pi^{asy}} + \mathrm{Adv}^{\text{OT}}_{A_{sy},\Pi^{sy}}$$

where $d = q_H + q_G + 2q_H \cdot q_D$*,* $\mathrm{Time}(A_{sy}) = \mathrm{Time}(A_{asy}) \approx \mathrm{Time}(A) + O(d^2)$*.*

Proof. Denote Ω_H and Ω_G as sets of all functions $H : \{0,1\}^* \to \{0,1\}^n$ and $G : \{0,1\}^* \to \mathcal{R}^{sy}$, respectively. The input and output register of the decryption query of A are denoted by C and Z, respectively. The proof of the IND-qCCA security of REACT transformation follows a proof outline for FO transformation as in the proof of Theorem 12, and thus we only give a brief analysis here.

Let **Game 0** be game $\mathrm{Game}^{\text{IND-qCCA}}_{A,\Pi}$. We then introduce a sequence of games to bound the difference between 1/2 and the probability that **Game 0** outputs 1.

$$\left|\Pr[\textbf{Game 0} \to 1] - \frac{1}{2}\right| = \mathrm{Adv}^{\text{IND-qCCA}}_{A,\Pi}. \tag{18}$$

```
Game 1:
1: $H \xleftarrow{\$} \Omega_H$, $G \xleftarrow{\$} \Omega_G$, $(pk, sk) \leftarrow \mathrm{Gen}^{asy}$, $R^* \xleftarrow{\$} \mathcal{M}^{asy}$, $r^* \xleftarrow{\$} \mathcal{R}^{asy}$
2: $(m_0, m_1) \leftarrow A^{H,G,\mathrm{Dec}_{sk}}(pk)$
3: $b \xleftarrow{\$} \{0,1\}$, $c_1^* = \mathrm{Enc}^{asy}_{pk}(R^*; r^*)$, $c_2^* = \mathrm{Enc}^{sy}_{G(R^*)}(m_b)$, $c_3^* = H(R^*, m_b^*, c_1^*, c_2^*)$
4: $b' \leftarrow A^{H,G,\mathrm{Dec}^*_{sk}}(pk, (c_1^*, c_2^*, c_3^*))$
5: return$[b = b']$
```

Starting from **Game 1**, random oracle H is simulated with CStO and its database register is denoted as D.

Game 1: In this game, we replace the decryption oracle in **Game 0** with oracle CCA_1. When to answer decryption queries, CCA_1 applies the preimage extraction procedure of Π. The construction of CCA_1 is shown in Appendix B.2. Note that as presented in Appendix B.2, the preimage extraction procedure of Π and CCA_1 is constructed by invoking oracle $\mathrm{PCA}_{sk}(\cdot)$, instead of using sk directly.

Since **Game 1** is the preimage extraction game $\mathrm{Game}^{\mathrm{Ext}}_{A,\Pi}$, we apply Theorem 9 to obtain

$$|\Pr[\text{Game } 0 \to 1] - \Pr[\text{Game } 1 \to 1]| \leq q_D \cdot \frac{5}{\sqrt{2^n}} \tag{19}$$

by similar analysis performed in the proof of Theorem 12.

Game 2: In this game, we replace oracle CCA_1 with CCA_2. Define CCA_2 as follows. Before the challenge query, CCA_2 acts the same as CCA_1. After that, CCA_2 replies to the decryption query in three steps:

1. Perform unitary $\mathsf{StdDecomp}_{(R^*,m_b,c_1^*,c_2^*)}$ to register D.
2. Apply CCA_1 on register C, Z and D.
3. Perform $\mathsf{StdDecomp}_{(R^*,m_b,c_1^*,c_2^*)}$ to register D again.

Oracle CCA_1 and CCA_2 differs only after the challenge query happens. In this case, CCA_1 and CCA_2 has the same effect on register C, Z and D if $\mathsf{StdDecomp}_{(R^*,m_b,c_1^*,c_2^*)}$ and CCA_1 are commutative. Lemma 8 implies that unitary $\mathsf{StdDecomp}_{(R^*,m_b,c_1^*,c_2^*)}$ commutes with CCA_1 with a loss, and thus $D(\mathrm{CCA}_1\rho\mathrm{CCA}_1^\dagger, \mathrm{CCA}_2\rho\mathrm{CCA}_2^\dagger) \leq 7\sqrt{\eta} = 7/\sqrt{2^n}$ holds for any joint state ρ of A and database. Further, adversary A issues at most q_D decryption queries after the challenge query, and then by the hybrid argument, we obtain

$$|\Pr[\textbf{Game 1} \to 1] - \Pr[\textbf{Game 2} \to 1]| \leq q_D \cdot \frac{7}{\sqrt{2^n}}. \tag{20}$$

Game 3: We change the process of replying random oracle queries: When random oracles are queried by A or oracle G is applied in the process of CCA_2, we perform E and then perform the random oracle, where E is a constant zero function with quantum access. Then we have

$$\Pr[\textbf{Game 2} \to 1] = \Pr[\textbf{Game 3} \to 1]. \tag{21}$$

Game 4: In this game, the only change is that semi-classical oracle $\mathcal{O}_{\mathcal{S}}^{SC}$ is applied before performing E, where set $\mathcal{S} := \{R^*, R^*\|\cdot\|\cdot\|\cdot\}$.

E is queried at most $q_H + q_G + 2q_H \cdot q_D$ times in **Game 5**:

- E is queried each time H or G is queried.
- A makes at most q_H queries to H and at most q_G queries to G.
- A makes at most q_D decryption queries, and CCA_2 requires at most $2q_H$ queries to G for each decryption query (c_1, c_2, c_3): At most q_H queries is needed to check if any $((R, m, c_1', c_2'), y) \in D$ matches with (c_1, c_2, c_3), i.e., $y = c_3$, $c_1' = c_1$, $c_2' = c_2$, $\mathrm{Dec}^{asy}_{sk}(c_1') = R$ and $\mathrm{Dec}^{sy}_{G(R)}(c_2') = m$. Another q_H queries is to uncompute these checks.

Let $d = q_H + q_G + 2q_H \cdot q_D$. By applying Theorem 3, we obtain

$$|\Pr[\textbf{Game 3} \to 1] - \Pr[\textbf{Game 4} \to 1]| \leq 2\sqrt{(d+1)\Pr[\text{Find} : \textbf{Game 4}]}. \tag{22}$$

If Find does not occur, CCA_2 will not disturb the database state on (R^*, m_b, c_1^*, c_2^*) by similar arguments in **Game 4** in the proof of Theorem 12.

Game 5: In this game, oracle CCA_2 is replaced with oracle CCA_3. Oracle CCA_2 and CCA_3 differs only after the challenge query happens. In this case, CCA_3 applies the preimage extraction procedure of Π but the database state on (δ^*, d^*) is not involved in this procedure.

If Find does not occur, oracle CCA_2 and CCA_3 acts identically. Thus,

$$\Pr[\text{Find} : \textbf{Game 4}] = \Pr[\text{Find} : \textbf{Game 5}], \tag{23}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 4} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 5} \to 1]. \tag{24}$$

Game 6: In this game, we pick $k^* \in \mathcal{K}^{sy}$ and $c_3^* \in \{0,1\}^n$ uniformly. We use them to produce the challenge ciphertext (c_1^*, c_2^*, c_3^*) instead of using $G(R^*)$ and $H(R^*, m_b^*, c_1^*, c_2^*)$.

A has never queried (R^*, m_b, c_1^*, c_2^*) to H until Find occurs. Furthermore, CCA_3 will not disturb the database state on (R^*, m_b, c_1^*, c_2^*) and thus $H(R^*, m_b, c_1^*, c_2^*)$ is uniformly random in A's view. On the other hand, if Find does not occur, then G has never been queried by δ^* except for producing the challenge ciphertext, which means that $G(R^*)$ is uniformly random in A's view. Thus, it is undetectable for adversary A to produce the challenge ciphertext with uniformly chosen $k^* \in \mathcal{K}^{sy}$ and $c_3^* \in \{0,1\}^n$. Hence,

$$\Pr[\text{Find} : \textbf{Game 5}] = \Pr[\text{Find} : \textbf{Game 6}], \tag{25}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 5} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 6} \to 1]. \tag{26}$$

Game 7: In this game, we turn oracle CCA_3 back to CCA_1.

If Find does not occur, (δ^*, d^*) is not recorded in the database of H both in these two games. In this case, oracle CCA_1 and CCA_3 act the same by their definitions. Therefore, we have

$$\Pr[\text{Find} : \textbf{Game 6}] = \Pr[\text{Find} : \textbf{Game 7}], \tag{27}$$
$$\Pr[\neg\text{Find} \wedge \textbf{Game 6} \to 1] = \Pr[\neg\text{Find} \wedge \textbf{Game 7} \to 1]. \tag{28}$$

Then by equation (23), (24), (25), (26), (27) and (28), we obtain

$$\Pr[\text{Find} : \textbf{Game 4}] = \Pr[\text{Find} : \textbf{Game 7}] \tag{29}$$

and

$$|\Pr[\textbf{Game 4} \to 1] - \Pr[\textbf{Game 7} \to 1]| \leq \Pr[\text{Find} : \textbf{Game 7}]. \tag{30}$$

Lemma 18. *There exists a quantum adversary A_{sy} invoking A such that*

$$\left|\Pr[\textbf{Game 7} \to 1] - \frac{1}{2}\right| = \mathrm{Adv}^{\mathrm{OT}}_{A_{sy},\Pi^{sy}} \tag{31}$$

and $\mathrm{Time}(A_{sy}) \approx \mathrm{Time}(A) + O\left((q_H + q_G + 2q_H \cdot q_D)^2\right)$.

Proof. A quantum algorithm A_{sy} that runs A and breaks the one-time security of Π^{sy} is constructed as follows:

A_{sy} generates $(pk, sk) \leftarrow \mathrm{Gen}$, picks $R^* \xleftarrow{\$} \mathcal{M}^{asy}$ and simulates **Game 7** for A. Random oracle G is simulated by CStO, and other oracles used in **Game 7** can be implemented efficiently by A_{sy}. When A makes challenge query (m_0, m_1), A_{sy} sends it to the challenger in $\mathrm{Game}^{\mathrm{OT}}_{A_{sy},\Pi^{sy}}$. After receiving c^*, A_{sy} picks $r^* \in \mathcal{R}^{asy}$ and $c_3^* \in \{0,1\}^n$ uniformly, then computes $c_1^* = \mathrm{Enc}^{asy}_{pk}(R^*; r^*)$ and sends (c_1^*, c^*, c_3^*) back to A. After receiving b' from A, A_{sy} output b'.

By the construction of A_{sy}, the view of A invoked by A_{sy} and that in **Game 7** are identical, and the output of A_{sy} is correct if and only if A guesses correctly . Thus we have $\Pr[\mathrm{Game}^{\mathrm{OT}}_{A_{sy},\Pi^{sy}} \to 1] = \Pr[\textbf{Game 7} \to 1]$ and

$$\left|\Pr[\textbf{Game 7} \to 1] - \frac{1}{2}\right| = \left|\Pr[\mathrm{Game}^{\mathrm{OT}}_{A_{sy},\Pi^{sy}} \to 1] - \frac{1}{2}\right| = \mathrm{Adv}^{\mathrm{OT}}_{A_{sy},\Pi^{sy}}.$$

Moreover, the running time of B is $\mathrm{Time}(B) = \mathrm{Time}(A) + \mathrm{T}_G + \mathrm{T}_H + \mathrm{T}_{\mathrm{CCA}_1}$, where $\mathrm{T}_G = O\left((q_G + 2q_H \cdot q_D)^2\right)$, $\mathrm{T}_H = O(q_H^2)$, $\mathrm{T}_{\mathrm{CCA}_1} = O(q_D \cdot q_H)$ by Appendix B.2.

□

Lemma 19. *There exists a quantum adversary A_{asy} invoking A such that*

$$\Pr[\text{Find} : \textbf{Game 7}] \leq 4d \cdot \mathrm{Adv}^{\text{OW-qPCA}}_{A_{asy},\Pi^{asy}} \tag{32}$$

and $\mathrm{Time}(A_{asy}) \approx \mathrm{Time}(A) + O\left((q_H + q_G + 2q_H \cdot q_D)^2\right)$.

Proof. Define $B^{\mathcal{O}_{\mathcal{S}}^{SC}}$ as a quantum oracle algorithm that on input pk, c^*, runs A and simulates **Game 7** for it. Then we have $\Pr[\text{Find} : \textbf{Game 7}] = \Pr[\text{Find} : B^{\mathcal{O}_{\mathcal{S}}^{SC}}(pk, c^*)]$, where $c^* = \text{Enc}_{pk}^{asy}(R^*)$, R^* is sampled uniformly from $\mathcal{M}^{asy}$. As analysis in **Game 5**, B makes at most $d = q_H + q_G + 2q_H \cdot q_D$ queries, then by Theorem 4,

$$\Pr[\text{Find} : B^{\mathcal{O}_{\mathcal{S}}^{SC}}(pk, c^*)] \leq 4d \cdot \Pr[(R, m, c_1, c_2) \in \mathcal{S} : (R, m, c_1, c_2) \leftarrow D(pk, c^*)].$$

Here D is a quantum algorithm invoking B. On input (pk, c^*), D chooses $i \xleftarrow{\$} \{1, \ldots, d\}$, runs $B^{\mathcal{O}_{\varnothing}^{SC}}(pk, c^*)$ until (just before) i-th query of B, and then measures the state on the input register of $\mathcal{O}_{\varnothing}^{SC}$ and obtains (R, m, c_1, c_2). Note that the running time of D and that of B are almost the same.

By the definition of $\mathcal{S}$, $(R, m, c_1, c_2) \in \mathcal{S}$ is equivalent to $R = R^*$. Then D can also be considered as a quantum algorithm A_{asy} that breaks the OW-qPCA security of Π^{asy}. Therefore,

$$\Pr[(R, m, c_1, c_2) \in \mathcal{S} : (\delta, d) \leftarrow D(pk, c^*)] = \Pr[R = R^* : (R, m, c_1, c_2) \leftarrow D(pk, c^*)] = \text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-qPCA}}.$$

The running time of B is $\text{Time}(B) = \text{Time}(A) + \text{T}_G + \text{T}_H + \text{T}_{\text{CCA}_1}$, where $\text{T}_G = O\left((q_G + 2q_H \cdot q_D)^2\right)$, $\text{T}_H = O(q_H^2)$, $\text{T}_{\text{CCA}_1} = O(q_D \cdot q_H)$.

□

Combining equation (22), (29), (30) (31), and (32),

$$\left|\Pr[\textbf{Game 3} \to 1] - \frac{1}{2}\right| \leq 2(d+1) \cdot \sqrt{\text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-qPCA}}} + 4d \cdot \text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-qPCA}} + \text{Adv}_{A_{sy}, \Pi^{sy}}^{\text{OT}} \tag{33}$$

Finally, we summarize equation (18), (19), (20), (21), and (33), and obtain

$$\text{Adv}_{A, \Pi}^{\text{IND-qCCA}} \leq q_D \cdot \frac{12}{\sqrt{2^n}} + 2(d+1) \cdot \sqrt{\text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-qPCA}}} + 4d \cdot \text{Adv}_{A_{asy}, \Pi^{asy}}^{\text{OW-qPCA}} + \text{Adv}_{A_{sy}, \Pi^{sy}}^{\text{OT}}.$$

□

References

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In *Annual International Cryptology Conference*, pages 269–295. Springer, 2019.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *International conference on the theory and application of cryptology and information security*, pages 41–69. Springer, 2011.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[BZ13] Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In *Annual cryptology conference*, pages 361–379. Springer, 2013.

[CFHL21] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 598–629. Springer, 2021.

[CMS19] Alessandro Chiesa, Peter Manohar, and Nicholas Spooner. Succinct arguments in the quantum random oracle model. In *Theory of Cryptography Conference*, pages 1–29. Springer, 2019.

[DFMS22] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Online-extractability in the quantum random-oracle model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 677–706. Springer, 2022.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Annual international cryptology conference*, pages 537–554. Springer, 1999.

[JSHJ$^+$02] Coron Jean-Sébastien, Helena Handschuh, Marc Joye, Pascal Paillier, David Pointcheval, and Christophe Tymen. Gem: A generic chosen-ciphertext secure encryption method. In *Cryptographers Track at the RSA Conference*, pages 263–276. Springer, 2002.

[NC02] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

[OP01] Tatsuaki Okamoto and David Pointcheval. React: Rapid enhanced-security asymmetric cryptosystem transform. In *Cryptographers Track at the RSA Conference*, pages 159–174. Springer, 2001.

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the fujisaki-okamoto and oaep transforms. In *Theory of Cryptography Conference*, pages 192–216. Springer, 2016.

[Zha15] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. *International Journal of Quantum Information*, 13(04):1550014, 2015.

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In *Annual International Cryptology Conference*, pages 239–268. Springer, 2019.

A Cryptographic Primitives and Security Definitions

Definition 20 (Secret-key encryption scheme)**.** *A secret-key encryption scheme* Π *consists of a pair of polynomial-time (in the security parameter* λ*) algorithms* (E, D).

1. E*, the encryption algorithm, on input a message* m *and a key* k *outputs a ciphertext* c*.*
2. D*, the decryption algorithm, on input a ciphertext* c *and a key* k *outputs either a message* m *or a special symbol* $\perp$ *if* c *is invalid.*

Definition 21 (One-time security)**.** *Given secret-key encryption scheme* $\Pi = (\mathrm{E}, \mathrm{D})$ *and adversary* A*, we define* $\mathrm{Game}^{\mathrm{OT}}_{A,\Pi}(1^\lambda)$ *in the following:*

1. *Query: Adversary* A *chooses two messages* m_0, m_1 *with the same length, and sends them to the challenger. The challenger chooses* $b \overset{\$}{\leftarrow} \{0,1\}$*, computes* $c = \mathrm{E}(k, m_b)$ *and returns it to the adversary.*
2. *Guess: Adversary* A *outputs a guess* b'*, if* $b' = b$*, output* 1*.*

$$\mathrm{Adv}^{\mathrm{OT}}_{A,\Pi}(\lambda) := \left| \Pr[\mathrm{Game}^{\mathrm{OT}}_{A,\Pi}(1^\lambda) \to 1] - \frac{1}{2} \right|$$

$\Pi = (\mathrm{E}, \mathrm{D})$ *is one-time secure if for any quantum polynomial time adversary* A*, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that* $\mathrm{Adv}^{\mathrm{OT}}_{A,\Pi}(\lambda) \leq \mathrm{negl}(\lambda)$.

Definition 22 (Public-key encryption scheme)**.** *A public-key encryption scheme* Π *consists of a triple of polynomial-time (in the security parameter* λ*) algorithms* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$.

1. Gen*, the key generation algorithm, on input* 1^λ *outputs a public/secret key-pair* (pk, sk)*.*
2. Enc*, the encryption algorithm, on input a public key* pk *and a message* m *outputs a ciphertext* c*.*
3. Dec*, the decryption algorithm, on input a secret key* sk *and a ciphertext* c *outputs either a message* m *or a special symbol* $\perp$ *if* c *is invalid.*

Definition 23 (γ-spread)**.** *A public-key encryption scheme* Π *is* γ*-spread (for* λ*) if for any* pk *produced by* $\mathrm{Gen}(1^\lambda)$ *and any message* m*,*

$$\max_{c\in\{0,1\}^*} \Pr[c \leftarrow \mathrm{Enc}(pk,m)] \leq \frac{1}{2^\gamma}.$$

Especially, the encryption scheme Π *is well-spread if* $\gamma = \omega(\log(\lambda))$*.*

Definition 24. *We say a function* $H : \{0,1\}^m \to \{0,1\}^n$ *has min-entropy* k *if*

$$-\log \max_{y\in\{0,1\}^n} \Pr[y = H(x) : x \xleftarrow{\$} \{0,1\}^m] = k.$$

Definition 25 (OW-CPA security)**.** *Given a public-key encryption scheme* $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *and adversary* A*, we define* $\mathrm{Game}_{A,\Pi}^{\text{OW-CPA}}(1^\lambda)$ *in the following:*

1. *KeyGen: The challenger runs* $\mathrm{Gen}(1^\lambda)$ *to obtain* (pk, sk)*, then sends* pk *to adversary* A*.*
2. *Challenge query: The challenger chooses* $m^* \xleftarrow{\$} \mathcal{M}$*, computes* $c^* \leftarrow \mathrm{Enc}(pk, m^*)$*, and then sends* c^* *to adversary* A*.*
3. *Guess: Adversary* A *produces a guess* m*. The challenger outputs* $[m^* = m]$*.*

Define the advantage of adversary A *in* $\mathrm{Game}_{A,\Pi}^{\text{OW-CPA}}(1^\lambda)$ *as*

$$\mathrm{Adv}_{A,\Pi}^{\text{OW-CPA}}(\lambda) := \Pr[\mathrm{Game}_{A,\Pi}^{\text{OW-CPA}}(1^\lambda) \to 1].$$

$\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *is* OW-CPA *secure if for any quantum polynomial time adversary* A*, a negligible function* $\mathrm{negl}(\cdot)$ *exists such that* $\mathrm{Adv}_{A,\Pi}^{\text{OW-CPA}}(\lambda) \leq \mathrm{negl}(\lambda)$*.*

Definition 26 (OW-qPCA security)**.** *Given a public-key encryption scheme* $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *and adversary* A*, we define* $\mathrm{Game}_{A,\Pi}^{\text{OW-qPCA}}(1^\lambda)$ *in the following:*

1. *KeyGen: The challenger runs* $\mathrm{Gen}(1^\lambda)$ *to obtain* (pk, sk)*, then sends* pk *to adversary* A*.*
2. *Challenge query: The challenger chooses* $m^* \xleftarrow{\$} \mathcal{M}$*, computes* $c^* \leftarrow \mathrm{Enc}(pk, m^*)$*, and then sends* c^* *to adversary* A*.*
3. *Guess: Adversary* A *queries oracle* $\mathrm{PCA}_{sk}(\cdot,\cdot)$ *with superposition state, where* $\mathrm{PCA}_{sk}(\cdot,\cdot)$ *is an oracle that takes input* (m, c) *and outputs* $[m = \mathrm{Dec}(sk, c)]$*. Finally, adversary* A *produces a guess* m*. The challenger outputs* $[m^* = m]$*.*

Define the advantage of adversary A *in* $\mathrm{Game}_{A,\Pi}^{\text{OW-qPCA}}(1^\lambda)$ *as*

$$\mathrm{Adv}_{A,\Pi}^{\text{OW-qPCA}}(\lambda) := \Pr[\mathrm{Game}_{A,\Pi}^{\text{OW-qPCA}}(1^\lambda) \to 1].$$

$\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *is* OW-qPCA *secure if for any quantum polynomial time adversary* A*, a negligible function* $\mathrm{negl}(\cdot)$ *exists such that* $\mathrm{Adv}_{A,\Pi}^{\text{OW-qPCA}}(\lambda) \leq \mathrm{negl}(\lambda)$*.*

Definition 27 (IND-qCCA security)**.** *Given a public-key encryption scheme* $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *and adversary* A*, we define* $\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}(1^\lambda)$ *as follows:*

1. *KeyGen: The challenger runs* $\mathrm{Gen}(1^\lambda)$ *to obtain* (pk, sk) *and sends* pk *to* A*.*
2. *Query: The adversary* A *is allowed to make the following two types of queries:*
   (a) *Challenge query: The adversary* A *chooses a pair of message* (m_0, m_1) *with the same length and sends them to the challenger. The challenger computes* $c^* = \mathrm{Enc}(pk, m_b)$*, and returns it to adversary* A*.*
   (b) *Decryption queries: Suppose challenge query is* c^**, and adversary* A *makes a quantum query to the function* CCA*, where CCA is defined as follows.*

$$\mathrm{CCA}(c) = \begin{cases} \perp & \textit{if the challenge query has happened and } c = c^*, \\ \mathrm{Dec}(sk, c) & \textit{otherwise.} \end{cases}$$

3. Guess: The adversary A produces a guess b'. If $b' = b$, output 1.

Define the advantage of adversary A in $\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}(1^\lambda)$ *as*

$$\mathrm{Adv}_{A,\Pi}^{\text{IND-qCCA}}(\lambda) := \left|\Pr[\mathrm{Game}_{A,\Pi}^{\text{IND-qCCA}}(1^\lambda) \to 1] - \frac{1}{2}\right|.$$

$\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *is* IND-qCCA *secure if for any quantum polynomial time adversary A, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that* $\mathrm{Adv}_{A,\Pi}^{\text{IND-qCCA}}(\lambda) \leq \mathrm{negl}(\lambda)$.

B The implementation of $\mathrm{U}_{\mathbf{Ext}}$

Let $\mathcal{O}$ be the random oracle with codomain $\mathcal{Y}$. Let Π be an oracle-masked scheme relative to $\mathcal{O}$, and tuple $(\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4)$ be the decomposition of Π. And (pk, sk) determines unitary $\mathrm{U}_{\mathrm{Ext}}$ by definition 6.

To implement $\mathrm{U}_{\mathrm{Ext}}$, we first give some notations, then introduce algorithm **Extract**, as a primitive of $\mathrm{U}_{\mathrm{Ext}}$, and finally give the implementation of $\mathrm{U}_{\mathrm{Ext}}$.

As is shown in definition 6, $\mathcal{O}$ is simulated by CStO and we introduce two definitions related to database D: For any $c \in \mathcal{C}$, a completion in D is a pair $(x, y) \in D$ such that $\mathrm{A}_2(pk, x, y) = c$ and $\mathrm{A}_3(sk, c) = x$. Define D_c to be the subset of D such that $\mathrm{A}_2(pk, x, y) = c$ for any (x, y) in D_c. Then any completion of c in set D is necessarily in set D_c. Note that D contains at most one completion of c, since c determines $\mathrm{A}_3(sk, c)$.

Define relation $\mathcal{R}_1(pk, sk)$ and $\mathcal{R}_2(pk, sk)$ for any (pk, sk) of Π as below.

$$\mathcal{R}_1(pk, sk) := \{(x, c) \in \mathcal{X} \times \mathcal{C} : \exists\, y \in \mathcal{Y} \text{ s.t. } \mathrm{A}_2(pk, x, y) = c\},$$

$$\mathcal{R}_2(pk, sk) := \{(x, c) \in \mathcal{X} \times \mathcal{C} : \mathrm{A}_3(sk, c) = x\}$$

where $\mathcal{X}$ is the output space of algorithm A_1. And we give the definition of the verification oracle $\mathbf{V}(pk, sk, \cdot, \cdot)$ of Π. $\mathbf{V}(pk, sk, \cdot, \cdot)$ takes input $(x, c) \in \mathcal{X} \times \mathcal{C}$ and outputs a bit $b \in \{0, 1\}$. For any $(x, c) \in \mathcal{R}_1(pk, sk)$, $\mathbf{V}(pk, sk, x, c) = 1$ if and only if $(x, c) \in \mathcal{R}_2(pk, sk)$.

Next, we define a classical algorithm **Extract**. **Extract** takes pk, sk, c and D as input. It looks for a completion of c in D. If a completion $(x, y) \in D$ is found, **Extract** outputs $(1, x)$. Otherwise, it outputs $(0, 0)$.

Then we give a construction of **Extract** relative to oracle $\mathbf{V}$. **Extract** on input c and D, finds a completion in two steps: For each pair (x, y) in D, it computes $c' = \mathrm{A}_2(pk, x, y)$ and compares c' with c for equality to check whether $(x, y) \in D_c$. Then to extract a completion from D_c, it invokes $\mathbf{V}$ and computes $\mathbf{V}(pk, sk, x, y)$ for each pair $(x, y) \in D_c$. If $(x, y) \in D$ exists such that $\mathbf{V}(pk, sk, x, y) = 1$, **Extract** outputs $(1, x)$. Otherwise, it outputs $(0, 0)$.

Then we implement $\mathrm{U}_{\mathrm{Ext}}$ by **Extract**, and we start with the case when the challenge query does not happen.

1. Evaluate $(b, x) = \mathbf{Extract}(pk, sk, c, D)$ in superposition and xor the output into a newly created register.
2. Apply the following conditional procedures in superposition:
3. Condition on $b = 0$, evaluate the map $|c, z, D, b, x\rangle \mapsto |c, z \oplus \perp, D, b, x\rangle$.
4. Condition on $b = 1$, evaluate the map $|c, z, D, b, x\rangle \mapsto |c, z \oplus \mathrm{A}_4(x), D, b, x\rangle$.
5. Uncompute (b, x) by evaluating $\mathbf{Extract}(pk, sk, c, D)$ in superposition again. Then discord the new register.

After the challenge query, the challenge ciphertext c^* is produced and $\mathrm{U}_{\mathrm{Ext}}$ is implemented below.

1. Apply the following conditional procedures in superposition:
2. Condition on $c = c^*$, evaluate the map $|c, z, D\rangle \mapsto |c, z \oplus \perp, D\rangle$.
3. Condition on $c \neq c^*$, apply the procedure in the case when c^* is undefined.

In addition, the running time of $\mathrm{U_{Ext}}$ is upper bounded as follows. Denote the length of database by l. For each database D, $|D| \leq l$ and Extract invokes A_2 and $\mathbf{V}$ at most l times during the execution. Thus $O(l \cdot \mathrm{Time}(\mathrm{A}_2) + l \cdot \mathrm{Time}(\mathbf{V}))$ is an upper bound of the running time of $\mathrm{U_{Ext}}$.

In the following B.1 and B.2, we give respective implementations of $\mathrm{U_{Ext}}$ for $\mathrm{FO}[\Pi^{asy}, \Pi^{sy}, H, G]$ and $\mathrm{REACT}[\Pi^{asy}, \Pi^{sy}, H, G]$. From the above general construction of $\mathrm{U_{Ext}}$, we observe that a specific implementation of $\mathbf{V}$ is sufficient to determine the implementation of $\mathrm{U_{Ext}}$ for an oracle-masked scheme Π. Thus, we only give respective constructions of the verification oracle $\mathbf{V}$ for scheme $\mathrm{FO}[\Pi^{asy}, \Pi^{sy}, H, G]$ and $\mathrm{REACT}[\Pi^{asy}, \Pi^{sy}, H, G]$.

B.1 The implementation of $\mathrm{U_{Ext}}$ for FO

For scheme $\Pi = \mathrm{FO}[\Pi^{asy}, \Pi^{sy}, H, G]$, we first present relation $\mathcal{R}_1(pk, sk)$ and $\mathcal{R}_2(pk, sk)$ to determine the input form of the verification oracle $\mathbf{V}$, then give an implementation of $\mathbf{V}$.

By Lemma 11, relation $\mathcal{R}_1(pk, sk)$ and $\mathcal{R}_2(pk, sk)$ are subsets of $\mathcal{M}^{asy} \times \mathcal{C}^{sy} \times \mathcal{C}^{asy} \times \mathcal{C}^{sy}$ for any (pk, sk) of Π. Any tuple $(\delta, d_1, c, d_2) \in \mathcal{R}_1(pk, sk)$ satisfies that $d_1 = d_2$ and $r \in \mathcal{R}^{asy}$ exists such that $\mathrm{Enc}^{asy}_{pk}(\delta; r) = c$. Tuple $(\delta, d_1, c, d_2) \in \mathcal{R}_2(pk, sk)$ if $d_1 = d_2$ and $\delta = \mathrm{Dec}^{asy}_{sk}(c)$.

Further, any tuple $(\delta, d_1, c, d_2) \in \mathcal{R}_1(pk, sk)$ also satisfies $\mathrm{Dec}^{asy}_{sk}(c) = \delta$ by the correctness of Π^{asy}, and thus $(\delta, d_1, c, d_2) \in \mathcal{R}_2(pk, sk)$. Then $\mathcal{R}_2(pk, sk)$ is a subset of $\mathcal{R}_1(pk, sk)$. By similar arguments, we also conclude that $(\delta, d_1, c, d_2) \notin \mathcal{R}_1(pk, sk)$ implies $(\delta, d_1, c, d_2) \notin \mathcal{R}_2(pk, sk)$ for any (pk, sk). Thus for any (pk, sk) of Π, $\mathcal{R}_1(pk, sk) = \mathcal{R}_2(pk, sk)$ and

$$\mathcal{R}_2(pk, sk) = \{(\delta, d, c, d) : c \in \mathcal{C}^{asy}, \delta = \mathrm{Dec}^{asy}_{sk}(c), d \in \mathcal{C}^{sy}\}.$$

By the definition of the verification oracle, $\mathbf{V}$ for Π can be simply simulated by an algorithm that takes tuple (δ, d_1, c, d_2) as input and trivially outputs 1. Moreover, notice that sk is not used in the implementation of $\mathrm{U_{Ext}}$ except for the verification oracle. Therefore, $\mathrm{U_{Ext}}$ for Π can be implemented without sk.

Finally, the running time of $\mathrm{U_{Ext}}$ is given by $O(l \cdot \mathrm{Time}(\mathrm{Enc}^{asy}))$.

B.2 The implementation of $\mathrm{U_{Ext}}$ for REACT

For scheme $\Pi = \mathrm{REACT}[\Pi^{asy}, \Pi^{sy}, H, G]$, we only give an implementation of oracle $\mathbf{V}$ here.

By Lemma 16, $\mathcal{R}_1(pk, sk)$ and $\mathcal{R}_2(pk, sk)$ are subsets of $\mathcal{M}^{asy} \times \mathcal{M}^{sy} \times \mathcal{C}^{asy} \times \mathcal{C}^{sy} \times \mathcal{C}^{asy} \times \mathcal{C}^{sy} \times \{0,1\}^n$ for any (pk, sk). Any tuple $(R, m, c_1, c_2, c'_1, c'_2, c'_3) \in \mathcal{R}_1(pk, sk)$ if $c_1 = c'_1$, $c_2 = c'_2$. And this tuple is an element of $\mathcal{R}_2(pk, sk)$ if $R = \mathrm{Dec}^{asy}_{sk}(c'_1)$, $m = \mathrm{Dec}^{sy}_{G(R)}(c'_2)$, $c_1 = c'_1$, $c_2 = c'_2$. Thus, we have

$$\mathcal{R}_1(pk, sk) = \{(R, m, c_1, c_2, c_1, c_2, c_3) : \mathcal{R} \in \mathcal{M}^{asy}, m \in \mathcal{M}^{sy}, c_1 \in \mathcal{C}^{asy}, c_2 \in \mathcal{C}^{sy}, c_3 \in \{0,1\}^n\},$$
$$\mathcal{R}_2(pk, sk) = \{(R, m, c_1, c_2, c_1, c_2, c_3) : c_1 \in \mathcal{C}^{asy}, R = \mathrm{Dec}^{asy}_{sk}(c_1), c_2 \in \mathcal{C}^{sy}, m = \mathrm{Dec}^{sy}_{G(R)}(c_2), c_3 \in \{0,1\}^n\}.$$

According to $\mathcal{R}_1(pk, sk)$ of Π, we assume the input form of $\mathbf{V}$ to be $(R, m, c_1, c_2, c_1, c_2, c_3)$ for convenience. Then we present an algorithm V_{Sim} relative to oracle PCA. V_{Sim} takes pk and tuple $(R, m, c_1, c_2, c_1, c_2, c_3)$ as input. It first invokes oracle $\mathrm{PCA}_{sk}(\cdot, \cdot)$ and obtain $b := \mathrm{PCA}_{sk}(R, c_1)$. If $b = 0$, V_{Sim} outputs 0. Else, it computes $m' = \mathrm{Dec}^{sy}_{G(R)}(c_2)$. If $m \neq m'$, output 0. Else, output 1. Then by the definition of PCA in Appendix A, it is easily verified that $\mathbf{V}$ can be simulated by V_{Sim}. In this way, $\mathrm{U_{Ext}}$ for Π is implemented by invoking oracle PCA_{sk} instead of using sk directly.

The running time of $\mathrm{U_{Ext}}$ is given by $O(l)$.

C The Properties of $\mathrm{U_{Ext}}$

Let $\Pi = (\mathrm{Gen}, \mathrm{Enc}^H, \mathrm{Dec}^H)$ be an oracle-masked scheme with parameter η. Let tuple $(\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4)$ be Π's decomposition. Fix any (pk, sk) generated by Gen and then we give some notations. Let $\{0,1\}^n$ be the codomain of H. Relation $\mathcal{R}_1(pk, sk)$ and $\mathcal{R}_2(pk, sk)$ of Π is as defined in Appendix B:

$$\mathcal{R}_1(pk, sk) := \{(x, c) \in \mathcal{X} \times \mathcal{C} : \exists y \in \{0,1\}^n \text{ s.t. } \mathrm{A}_2(pk, x, y) = c\},$$

$$\mathcal{R}_2(pk, sk) := \{(x, c) \in \mathcal{X} \times \mathcal{C} : \mathrm{A}_3(sk, c) = x\},$$

where $\mathcal{X}$ is the output space of algorithm A_1. For any $c \in \mathcal{C}$, define $\mathcal{S}_c := \{y \in \{0,1\}^n : x = \mathrm{A}_3(sk, c), c = \mathrm{A}_2(pk, x, y)\}$, and $|\mathcal{S}_c| \leq 2^n \cdot \eta$ from the definition of η.

C.1 Proof of Lemma 7

Proof. Let $\triangle := \mathrm{U}_{\mathrm{Sim}} - \mathrm{U}_{\mathrm{Ext}}$. Denote by c^* the challenge ciphertext. The decryption oracle $\mathrm{Dec}_{sk}^{H}(\cdot)$ corresponds to unitary operator $\mathrm{U}_{\mathrm{Sim}}$. Given ciphertext c and database D, let $x := \mathrm{A}_3(sk, c)$ and then we define several cases for x, c, D.

1. $c = c^*$, or $x = \perp$, or $(x, c) \notin \mathcal{R}_1(pk, sk)$. Then $\mathrm{Dec}_{sk}^{H}(c) = \perp$ and $\mathrm{U}_{\mathrm{Sim}}|c, z, D\rangle = |c, z \oplus \perp, D\rangle$. On the other hand, except for the $c = c^*$ case, no $(x, y) \in D$ exists such that $\mathrm{A}_2(pk, x, y) = c$ and $\mathrm{A}_3(sk, c) = x$. And therefore, $\mathrm{U}_{\mathrm{Ext}}|c, z, D\rangle = |c, z \oplus \perp, D\rangle$.
Let P_1 be the projection onto c, D that $c = c^*$ or $x = \perp$, or $(x, c) \notin \mathcal{R}_1(pk, sk)$. Then for any state $|\psi\rangle$, $\triangle \circ P_1|\psi\rangle = 0$.

2. $c \neq c^*$, $(x, c) \in \mathcal{R}_1(pk, sk)$ but $D(x) = \perp$. Then $\mathrm{U}_{\mathrm{Ext}}|c, z, D\rangle = |c, z \oplus \perp, D\rangle$. In this case, $x \neq \perp$, $\mathrm{Dec}_{sk}^{H}(\cdot)$ first queries CStO_H the value of $H(x)$, and then decrypts with $H(x)$. Thus, unitary $\mathrm{U}_{\mathrm{Sim}}$ performs as follows:

$$\begin{aligned}&\mathrm{U}_{\mathrm{Sim}}|c, z, D\rangle\\ &= \sum_{r \in \mathcal{S}_c} \frac{1}{\sqrt{2^n}} \mathsf{StdDecomp}_x |c, z \oplus \mathrm{A}_4(x), D \cup (x, r)\rangle + \sum_{r \notin \mathcal{S}_c} \frac{1}{\sqrt{2^n}} \mathsf{StdDecomp}_x |c, z \oplus \perp, D \cup (x, r)\rangle.\end{aligned}$$

Then
$$\triangle|c, z, D\rangle = \sum_{r \in \mathcal{S}_c} \frac{1}{\sqrt{2^n}} \big(|c, z \oplus \mathrm{A}_4(x)\rangle - |c, z \oplus \perp\rangle\big) \mathsf{StdDecomp}_x |D \cup (x, r)\rangle.$$

Let P_2 be the projection onto c, D such that $(x, c) \in \mathcal{R}_1(pk, sk)$ and $D(x) = \perp$.

3. $c \neq c^*$, $(x, c) \in \mathcal{R}_1(pk, sk)$ and $D(x) = \perp$, $r \neq 0$.

$$\mathrm{U}_{\mathrm{Ext}}|c, z, D \cup (x, \beta_r)\rangle = \sum_{\omega \in \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} |c, z \oplus \mathrm{A}_4(x), D \cup (x, \omega)\rangle + \sum_{\omega \notin \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} |c, z \oplus \perp, D \cup (x, \omega)\rangle.$$

By similar arguments in case 2,

$$\begin{aligned}&\mathrm{U}_{\mathrm{Sim}}|c, z, D \cup (x, \beta_r)\rangle\\ &= \sum_{\omega \in \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} \mathsf{StdDecomp}_x |c, z \oplus \mathrm{A}_4(x), D \cup (x, \omega)\rangle + \sum_{\omega \notin \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} \mathsf{StdDecomp}_x |c, z \oplus \perp, D \cup (x, \omega)\rangle.\end{aligned}$$

Thus we obtain

$$\begin{aligned}&\triangle|c, z, D \cup (x, \beta_r)\rangle\\ &= \sum_{\omega \in \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} \left(\mathsf{StdDecomp}_x - \mathrm{I}\right) |c, z \oplus \mathrm{A}_4(x), D \cup (x, \omega)\rangle\\ &\quad + \sum_{\omega \notin \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} \left(\mathsf{StdDecomp}_x - \mathrm{I}\right) |c, z \oplus \perp, D \cup (x, \omega)\rangle\\ &= \sum_{\omega \in \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} |c, z \oplus \mathrm{A}_4(x)\rangle \Big(\frac{1}{\sqrt{2^n}}|D\rangle - \frac{1}{\sqrt{2^n}}|D \cup (x, \beta_0)\rangle\Big)\\ &\quad + \sum_{\omega \notin \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} |c, z \oplus \perp\rangle \Big(\frac{1}{\sqrt{2^n}}|D\rangle - \frac{1}{\sqrt{2^n}}|D \cup (x, \beta_0)\rangle\Big).\end{aligned}$$

For any $r \neq 0$ and subset $\mathcal{S}$ of $\{0,1\}^n$, $\sum_{\omega \in \mathcal{S}} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} + \sum_{\omega \notin \mathcal{S}} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} = 0$ holds, and therefore

$$\triangle|c, z, D \cup (x, \beta_r)\rangle = \sum_{\omega \in \mathcal{S}_c} \frac{(-1)^{\omega \cdot r}}{\sqrt{2^n}} \big(|c, z \oplus \mathrm{A}_4(x)\rangle - |c, z \oplus \perp\rangle\big)\Big(\frac{1}{\sqrt{2^n}}|D\rangle - \frac{1}{\sqrt{2^n}}|D \cup (x, \beta_0)\rangle\Big).$$

Let P_3 be the projection onto state of the form $\sum_{c,z,D,r\neq 0} \alpha_{c,z,D,r}|c, z, D \cup (x, \beta_r)\rangle$, where the support is over c, D such that $(x, c) \in \mathcal{R}_1(pk, sk)$, $D(x) = \perp$.

For any state $|\psi\rangle$, $P_2|\psi\rangle = \sum_{c,z,D} \alpha_{c,z,D}|c,z,D\rangle$, where the support is over c, D defined in case 2. By the calculation in case 2,

$$\triangle \circ P_2|\psi\rangle = \sum_{c,z,D,r\in\mathcal{S}_c} \frac{1}{\sqrt{2^n}} \cdot \alpha_{c,z,D}\big(|c, z \oplus \mathrm{A}_4(x)\rangle - |c, z\oplus \perp\rangle\big)\mathsf{StdDecomp}_x|D \cup (x,r)\rangle.$$

Then we have

$$\begin{aligned}\|\triangle \circ P_2|\psi\rangle\|^2 &= \left\| \sum_{c,z,D,r\in\mathcal{S}_c} \frac{1}{\sqrt{2^n}} \cdot \alpha_{c,z,D}\left(|c, z \oplus \mathrm{A}_4(x)\rangle - |c, z\oplus \perp\rangle\right)\mathsf{StdDecomp}_x|D \cup (x,r)\rangle \right\|^2 \\ &= \sum_{c,z,D,r\in\mathcal{S}_c} \frac{1}{2^n} \cdot |\alpha_{c,z,D}|^2 \cdot \||c, z \oplus \mathrm{A}_4(x)\rangle - |c, z\oplus \perp\rangle\|^2.\end{aligned}$$

Since $\||a\rangle - |b\rangle\|^2 \le 2\left(\||a\rangle\|^2 + \||b\rangle\|^2\right)$ holds for any state $|a\rangle$ and $|b\rangle$ on the same register. Thus

$$\begin{aligned}\|\triangle \circ P_2|\psi\rangle\|^2 &\le \sum_{c,z,D,r\in\mathcal{S}_c} \frac{2}{2^n} \cdot |\alpha_{c,z,D}|^2 \cdot \left(\||c, z \oplus \mathrm{A}_4(x)\rangle\|^2 + \||c, z\oplus \perp\rangle\|^2\right) \\ &= \sum_{c,z,D} \frac{4}{2^n} \cdot |\mathcal{S}_c| \cdot |\alpha_{c,z,D}|^2 \le \sum_{c,z,D} 4\cdot\eta\cdot |\alpha_{c,z,D}|^2 = 4\cdot\eta\cdot\|P_2|\psi\rangle\|^2.\end{aligned}$$

For any state $|\psi\rangle$, $P_3|\psi\rangle = \sum_{c,z,D,r\neq 0} \alpha_{c,z,D,r}|c,z,D\cup(x,\beta_r)\rangle$ where c, D, r is defined in case 3. By the calculation in case 3,

$$\triangle\circ P_3|\psi\rangle = \sum_{c,z,D,r\neq 0} \alpha_{c,z,D,r}\cdot\left(\sum_{\omega\in\mathcal{S}_c} \frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}(|c, z \oplus \mathrm{A}_4(x)\rangle - |c, z\oplus \perp\rangle)(\frac{1}{\sqrt{2^n}}|D\rangle - \frac{1}{\sqrt{2^n}}|D\cup(x,\beta_0)\rangle)\right).$$

Then

$$\begin{aligned}&\left\|\triangle \circ P_3|\psi\rangle\right\|^2 \\ &= \left\| \sum_{c,z,D,r\neq 0} \alpha_{c,z,D,r}\cdot\left(\sum_{\omega\in\mathcal{S}_c} \frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}(|c, z \oplus \mathrm{A}_4(x)\rangle - |c, z\oplus \perp\rangle)(\frac{1}{\sqrt{2^n}}|D\rangle - \frac{1}{\sqrt{2^n}}|D\cup(x,\beta_0)\rangle)\right)\right\|^2 \\ &= \sum_{c,z,D} \frac{1}{2^n}\cdot\left|\sum_{r\neq 0}\alpha_{c,z,D,r}\left(\sum_{\omega\in\mathcal{S}_c}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\right)\right|^2 \cdot \||c, z \oplus \mathrm{A}_4(x)\rangle - |c, z\oplus \perp\rangle\|^2 \cdot \||D\rangle - |D\cup(x,\beta_0)\rangle\|^2 \\ &\le \sum_{c,z,D} \frac{2}{2^n}\cdot\left|\sum_{r\neq 0}\alpha_{c,z,D,r}\left(\sum_{\omega\in\mathcal{S}_c}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\right)\right|^2 \cdot \left(\||c, z \oplus \mathrm{A}_4(x)\rangle\|^2 + \||c, z\oplus \perp\rangle\|^2\right) \cdot \||D\rangle - |D\cup(x,\beta_0)\rangle\|^2 \\ &= \sum_{c,z,D} \frac{8}{2^n}\cdot\left|\sum_{r\neq 0}\alpha_{c,z,D,r}\left(\sum_{\omega\in\mathcal{S}_c}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\right)\right|^2.\end{aligned}$$

On the other hand, $\|P_3|\psi\rangle\|^2 = \|\sum_{c,z,D,r\neq 0}\alpha_{c,z,D,r}|c,z,D\cup(x,\beta_r)\rangle\|^2$, which can be rewritten as

$$\begin{aligned}\|P_3|\psi\rangle\|^2 &= \left\|\sum_{c,z,D,r\neq 0}\alpha_{c,z,D,r}\sum_{\omega}\frac{(-1)^{r\cdot\omega}}{\sqrt{2^n}}|c,z,D\cup(x,\omega)\rangle\right\|^2 \\ &= \left\|\sum_{c,z,D,\omega}\sum_{r\neq 0}\alpha_{c,z,D,r}\frac{(-1)^{r\cdot\omega}}{\sqrt{2^n}}|c,z,D\cup(x,\omega)\rangle\right\|^2 = \sum_{c,z,D,\omega}\left|\sum_{r\neq 0}\frac{(-1)^{r\cdot\omega}}{\sqrt{2^n}}\alpha_{c,z,D,r}\right|^2.\end{aligned}$$

Therefore,

$$\begin{aligned}\left\|\triangle \circ P_3|\psi\rangle\right\|^2 &\le \sum_{c,z,D} 8/2^n \cdot \left|\sum_{r\neq 0}\alpha_{c,z,D,r}\left(\sum_{\omega\in\mathcal{S}_c}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\right)\right|^2\\ &= \sum_{c,z,D} 8/2^n \cdot \left|\sum_{\omega\in\mathcal{S}_c}\left(\sum_{r\neq 0}\alpha_{c,z,D,r}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\right)\right|^2\\ &\le \sum_{c,z,D} 8/2^n \cdot |\mathcal{S}_c| \cdot \left(\sum_{\omega\in\mathcal{S}_c}\left|\sum_{r\neq 0}\alpha_{c,z,D,r}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\right|^2\right)\\ &\le \sum_{c,z,D} 8\cdot\eta\cdot \left(\sum_{\omega\in\mathcal{S}_c}\left|\sum_{r\neq 0}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\alpha_{c,z,D,r}\right|^2\right)\\ &\le \sum_{c,z,D} 8\cdot\eta\cdot \left(\sum_{\omega}\left|\sum_{r\neq 0}\frac{(-1)^{\omega\cdot r}}{\sqrt{2^n}}\alpha_{c,z,D,r}\right|^2\right) = 8\cdot\eta\cdot\|P_3|\psi\rangle\|^2.\end{aligned}$$

Notice that for any state $|\psi\rangle$ in Lemma 7, $(P_1+P_2+P_3)|\psi\rangle = |\psi\rangle$, and thus

$$\|\triangle|\psi\rangle\| \le \sum_{i=1}^{3}\|\triangle\circ P_i|\psi\rangle\| \le (2+2\sqrt{2})\cdot\sqrt{\eta} \le 5\sqrt{\eta}.$$

□

C.2 Proof of Lemma 8

Proof. Let $\triangle := \mathrm{U}_{\mathrm{Ext}} \circ \mathsf{StdDecomp}_{x^*} - \mathsf{StdDecomp}_{x^*} \circ \mathrm{U}_{\mathrm{Ext}}$. For any $x^* \in \{0,1\}^*$, we define several cases for x^*, ciphertext c and database D.

1. $(x^*, c) \notin \mathcal{R}_1(pk, sk)$, or $(x^*, c) \notin \mathcal{R}_2(pk, sk)$. Then the value of $D(x^*)$ does not affect the decryption of c. $\mathsf{StdDecomp}_{x^*}$ only affects the value of database D on x^*, and therefore $\triangle|c,z,D\rangle = 0$. Let P_1 be the projection onto x^*, c, D such that $c = c^*$, or $(x^*, c) \notin \mathcal{R}_1(pk, sk)$, or $(x^*, c) \notin \mathcal{R}_2(pk, sk)$. Then for any state $|\psi\rangle$, $\triangle \circ P_1|\psi\rangle = 0$.

2. $(x^*, c) \in \mathcal{R}_1(pk, sk)$, $(x^*, c) \in \mathcal{R}_2(pk, sk)$ but $D(x^*) = \perp$. We obtain

$$\mathsf{StdDecomp}_{x^*} \circ \mathrm{U}_{\mathrm{Ext}}|c,z,D\rangle = \mathsf{StdDecomp}_{x^*}|c, z\oplus\perp, D\rangle = |c, z\oplus\perp, D\cup(x^*,\beta_0)\rangle$$

and

$$\begin{aligned}\mathrm{U}_{\mathrm{Ext}} \circ \mathsf{StdDecomp}_{x^*}|c,z,D\rangle &= \mathrm{U}_{\mathrm{Ext}}\sum_{r}\frac{1}{\sqrt{2^n}}|c,z,D\cup(x^*,r)\rangle\\ &= \frac{1}{\sqrt{2^n}}\sum_{r\in\mathcal{S}_c}|c, z\oplus \mathrm{A}_4(x^*), D\cup(x^*,r)\rangle + \frac{1}{\sqrt{2^n}}\sum_{r\notin\mathcal{S}_c}|c, z\oplus\perp, D\cup(x^*,r)\rangle.\end{aligned}$$

Therefore,

$$\triangle|c,z,D\rangle = \frac{1}{\sqrt{2^n}}\sum_{r\in\mathcal{S}_c}(|c, z\oplus\mathrm{A}_4(x^*)\rangle - |c, z\oplus\perp\rangle)|D\cup(x^*,r)\rangle.$$

Let P_2 be the projection onto x^*, c, D such that $(x^*, c) \in \mathcal{R}_1(pk, sk)$, $(x^*, c) \in \mathcal{R}_2(pk, sk)$ and $D(x^*) = \perp$.

3. $(x^*, c) \in \mathcal{R}_1(pk, sk)$, $(x^*, c) \in \mathcal{R}_2(pk, sk)$ and $D(x^*) \notin \mathcal{S}_c$. Rewrite D as $D' \cup (x^*, y)$, where $D'(x^*) = \perp$, $y = D(x^*)$. Then we calculate

$$\begin{aligned}\mathsf{StdDecomp}_{x^*} \circ \mathrm{U}_{\mathrm{Ext}}|c,z,D'\cup(x^*,y)\rangle &= \mathsf{StdDecomp}_{x^*}|c, z\oplus\perp, D'\cup(x^*,y)\rangle\\ &= |c, z\oplus\perp\rangle\left(|D'\cup(x^*,y)\rangle + \frac{1}{\sqrt{2^n}}|D'\rangle - \frac{1}{\sqrt{2^n}}|D'\cup(x^*,\beta_0)\rangle\right)\end{aligned}$$

and

$$\mathrm{U}_{\mathrm{Ext}}\circ\mathsf{StdDecomp}_{x^*}|c,z,D'\cup(x^*,y)\rangle = \mathrm{U}_{\mathrm{Ext}}|c,z\rangle\left(|D'\cup(x^*,y)\rangle + \frac{1}{\sqrt{2^n}}|D'\rangle - \frac{1}{\sqrt{2^n}}|D'\cup(x^*,\beta_0)\rangle\right)$$

where $\mathrm{U}_{\mathrm{Ext}}|c,z,D'\cup(x^*,y)\rangle = |c,z\oplus\perp,D'\cup(x^*,y)\rangle$, $\mathrm{U}_{\mathrm{Ext}}|c,z,D'\rangle = |c,z\oplus\perp,D'\rangle$ and

$$\mathrm{U}_{\mathrm{Ext}}|c,z,D'\cup(x^*,\beta_0)\rangle = \frac{1}{\sqrt{2^n}}\sum_{r\in\mathcal{S}_c}|c,z\oplus\mathrm{A}_4(x^*),D'\cup(x^*,r)\rangle + \frac{1}{\sqrt{2^n}}\sum_{r\notin\mathcal{S}_c}|c,z\oplus\perp,D'\cup(x^*,r)\rangle.$$

Thus

$$\triangle|c,z,D'\cup(x^*,y)\rangle = \frac{1}{2^n}\sum_{r\in\mathcal{S}_c}(|c,z\oplus\perp\rangle - |c,z\oplus\mathrm{A}_4(x^*)\rangle)|D'\cup(x^*,r)\rangle.$$

Let P_3 be the projection onto x^*, c, D', y such that $(x^*,c)\in\mathcal{R}_1(pk,sk)$, $(x^*,c)\in\mathcal{R}_2(pk,sk)$, $D'(x^*)=\perp$, $y\notin\mathcal{S}_c$.

4. $(x^*,c)\in\mathcal{R}_1(pk,sk)$, $(x^*,c)\in\mathcal{R}_2(pk,sk)$ and $D(x^*)\in\mathcal{S}_c$. We represent D as $D'\cup(x^*,y)$, where $D'(x^*)=\perp$, $y=D(x^*)$. Then we have

$$\begin{aligned}&\mathsf{StdDecomp}_{x^*}\circ\mathrm{U}_{\mathrm{Ext}}|c,z,D'\cup(x^*,y)\rangle = \mathsf{StdDecomp}_{x^*}|c,z\oplus\mathrm{A}_4(x^*),D'\cup(x^*,y)\rangle\\ &= |c,z\oplus\mathrm{A}_4(x^*)\rangle\left(|D'\cup(x^*,y)\rangle + \frac{1}{\sqrt{2^n}}|D'\rangle - \frac{1}{\sqrt{2^n}}|D'\cup(x^*,\beta_0\rangle\right)\end{aligned}$$

and

$$\mathrm{U}_{\mathrm{Ext}}\circ\mathsf{StdDecomp}_{x^*}|c,z,D'\cup(x^*,y)\rangle = \mathrm{U}_{\mathrm{Ext}}|c,z\rangle\left(|D'\cup(x^*,y)\rangle + \frac{1}{\sqrt{2^n}}|D'\rangle - \frac{1}{\sqrt{2^n}}|D'\cup(x^*,\beta_0\rangle\right)$$

where $\mathrm{U}_{\mathrm{Ext}}|c,z,D'\cup(x^*,y)\rangle = |c,z\oplus\mathrm{A}_4(x^*),D'\cup(x^*,y)\rangle$, $\mathrm{U}_{\mathrm{Ext}}|c,z,D'\rangle = |c,z\oplus\perp,D'\rangle$ and

$$\mathrm{U}_{\mathrm{Ext}}|c,z,D'\cup(x^*,\beta_0)\rangle = \frac{1}{\sqrt{2^n}}\sum_{r\in\mathcal{S}_c}|c,z\oplus\mathrm{A}_4(x^*),D'\cup(x^*,r)\rangle + \frac{1}{\sqrt{2^n}}\sum_{r\notin\mathcal{S}_c}|c,z\oplus\perp,D'\cup(x^*,r)\rangle.$$

Then we obtain

$$\begin{aligned}&\triangle|c,z,D'\cup(x^*,y)\rangle\\ &= \frac{1}{\sqrt{2^n}}(|c,z\oplus\perp\rangle - |c,z\oplus\mathrm{A}_4(x^*)\rangle)|D'\rangle + \sum_{r\notin\mathcal{S}_c}\frac{1}{2^n}\big(|c,z\oplus\mathrm{A}_4(x^*)\rangle - |c,z\oplus\perp\rangle\big)|D'\cup(x^*,r)\rangle.\end{aligned}$$

Let P_4 be the projection onto x^*, c, D', y such that $(x^*,c)\in\mathcal{R}_1(pk,sk)$, $(x^*,c)\in\mathcal{R}_2(pk,sk)$, $D'(x^*)=\perp$ and $y\in\mathcal{S}_c$.

For any state $|\psi\rangle$, $P_2|\psi\rangle = \sum_{c,z,D}\alpha_{c,z,D}|c,z,D\rangle$, $\|P_2|\psi\rangle\|^2 = \sum_{c,z,D}|\alpha_{c,z,D}|^2$ where c, z, D is defined in case 2. By the calculation in case 2,

$$\triangle\circ P_2|\psi\rangle = \sum_{c,z,D}\alpha_{c,z,D}\left(\frac{1}{\sqrt{2^n}}\sum_{r\in\mathcal{S}_c}\big(|c,z\oplus\mathrm{A}_4(x^*)\rangle - |c,z\oplus\perp\rangle\big)|D\cup(x^*,r)\rangle\right),$$

$$\begin{aligned}\|\triangle\circ P_2|\psi\rangle\|^2 &= \sum_{c,z,D,r\in\mathcal{S}_c}\frac{1}{2^n}\cdot|\alpha_{c,z,D}|^2\cdot\left\|\big(|c,z\oplus\mathrm{A}_4(x^*)\rangle - |c,z\oplus\perp\rangle\big)|D\cup(x^*,r)\rangle\right\|^2\\ &\le \sum_{c,z,D,r\in\mathcal{S}_c}\frac{2}{2^n}\cdot|\alpha_{c,z,D}|^2\cdot\big(\||z\oplus\mathrm{A}_4(x^*)\rangle\|^2 + \||z\oplus\perp\rangle\|^2\big) \le 4\cdot\eta\cdot\|P_2|\psi\rangle\|^2.\end{aligned}$$

For any state $|\psi\rangle$, $P_3|\psi\rangle = \sum_{c,z,D',y\notin\mathcal{S}_c}\alpha_{c,z,D',y}|c,z,D'\cup(x^*,y)\rangle$, where c, z, D', y is defined in case 3. Then $\|P_3|\psi\rangle\|^2 = \sum_{c,z,D',y\notin\mathcal{S}_c}|\alpha_{c,z,D',y}|^2$. By the calculation in case 3,

$$\triangle\circ P_3|\psi\rangle = \sum_{c,z,D',y\notin\mathcal{S}_c}\alpha_{c,z,D',y}\left(\frac{1}{2^n}\sum_{r\in\mathcal{S}_c}(|c,z\oplus\perp\rangle - |c,z\oplus\mathrm{A}_4(x^*)\rangle)\,|D'\cup(x^*,r)\rangle\right),$$

$$\begin{aligned}
\|\triangle \circ P_3|\psi\rangle\|^2 &= \sum_{c,z,D',r\in\mathcal{S}_c} \frac{1}{4^n}\cdot\left|\sum_{y\notin\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2 \cdot \||c,z\oplus\perp\rangle - |c,z\oplus \mathrm{A}_4(x^*)\rangle\|^2 \cdot \||D'\cup(x^*,r)\rangle\|^2 \\
&= \sum_{c,z,D'} \frac{1}{4^n}\cdot|\mathcal{S}_c|\cdot\left|\sum_{y\notin\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2 \cdot \||c,z\oplus\perp\rangle - |c,z\oplus \mathrm{A}_4(x^*)\rangle\|^2 \\
&\le \sum_{c,z,D'} \frac{2}{4^n}\cdot|\mathcal{S}_c|\cdot\left|\sum_{y\notin\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2 \cdot \left(\||z\oplus\perp\rangle\|^2 + \||z\oplus \mathrm{A}_4(x^*)\rangle\|^2\right) \\
&= \sum_{c,z,D'} \frac{4}{4^n}\cdot|\mathcal{S}_c|\cdot\left|\sum_{y\notin\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2 \\
&\le \sum_{c,z,D'} \frac{4}{4^n}\cdot|\mathcal{S}_c|\cdot(2^n-|\mathcal{S}_c|)\cdot\left(\sum_{y\notin\mathcal{S}_c}|\alpha_{c,z,D',y}|^2\right) \\
&\le \sum_{c,z,D',y\notin\mathcal{S}_c} 4\cdot\eta\cdot|\alpha_{c,z,D',y}|^2 = 4\cdot\eta\cdot\|P_3|\psi\rangle\|^2.
\end{aligned}$$

For any state $|\psi\rangle$, $P_4|\psi\rangle = \sum_{c,z,D',y\in\mathcal{S}_c}\alpha_{c,z,D',y}|c,z,D'\cup(x^*,y)\rangle$, where c, z, D', y is defined in case 4. Then $\|P_4|\psi\rangle\|^2 = \sum_{c,z,D',y\in\mathcal{S}_c}|\alpha_{c,z,D',y}|^2$. By the calculation in case 4,

$$\begin{aligned}
\triangle\circ P_4|\psi\rangle = \sum_{c,z,D',y\in\mathcal{S}_c}\alpha_{c,z,D',y}\Bigg(&\frac{1}{\sqrt{2^n}}(|c,z\oplus\perp\rangle - |c,z\oplus\mathrm{A}_4(x^*)\rangle)|D'\rangle \\
&+\sum_{r\notin\mathcal{S}_c}\frac{1}{2^n}\big(|c,z\oplus\mathrm{A}_4(x^*)\rangle - |c,z\oplus\perp\rangle\big)|D'\cup(x^*,r)\rangle\Bigg),
\end{aligned}$$

$$\begin{aligned}
&\|\triangle\circ P_4|\psi\rangle\|^2 \\
&= \sum_{c,z,D'}\frac{1}{2^n}\cdot\left|\sum_{y\in\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2\cdot\big\||c,z\oplus\mathrm{A}_4(x^*)\rangle - |c,z\oplus\perp\rangle\|^2\cdot\||D'\rangle\big\|^2 \\
&\quad+\sum_{c,z,D',r\notin\mathcal{S}_c}\frac{1}{4^n}\cdot\left|\sum_{y\in\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2\cdot\||c,z\oplus\perp\rangle - |c,z\oplus\mathrm{A}_4(x^*)\rangle\|^2\cdot\||D'\cup(x^*,r)\rangle\|^2 \\
&\le \sum_{c,z,D'}\frac{2}{2^n}\cdot\left|\sum_{y\in\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2\cdot\left(\||z\oplus\mathrm{A}_4(x^*)\rangle\|^2 + \||z\oplus\perp\rangle\|^2\right) \\
&\quad+\sum_{c,z,D',r\notin\mathcal{S}_c}\frac{2}{4^n}\cdot\left|\sum_{y\in\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2\cdot\left(\||z\oplus\perp\rangle\|^2 + \||z\oplus\mathrm{A}_4(x^*)\rangle\|^2\right) \\
&= \sum_{c,z,D'}\frac{4}{2^n}\cdot\left|\sum_{y\in\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2 + \sum_{c,z,D'}\frac{4}{4^n}\cdot(2^n-|\mathcal{S}_c|)\cdot\left|\sum_{y\in\mathcal{S}_c}\alpha_{c,z,D',y}\right|^2 \\
&\le \sum_{c,z,D'}\frac{4}{2^n}\cdot|\mathcal{S}_c|\cdot\left(\sum_{y\in\mathcal{S}_c}|\alpha_{c,z,D',y}|^2\right) + \sum_{c,z,D'}\frac{4}{4^n}\cdot(2^n-|\mathcal{S}_c|)\cdot|\mathcal{S}_c|\cdot\left(\sum_{y\in\mathcal{S}_c}|\alpha_{c,z,D',y}|^2\right) \\
&\le \sum_{c,z,D',y\in\mathcal{S}_c} 8\cdot\eta\cdot|\alpha^2_{c,z,D',y}| = 8\cdot\eta\cdot\|P_4|\psi\rangle\|^2.
\end{aligned}$$

Because $P_1+P_2+P_3+P_4=\mathbf{I}$, $\|\triangle|\psi\rangle\| \le \sum_{i=1}^4\|\triangle\circ P_i|\psi\rangle\| \le (4+2\sqrt{2})\cdot\sqrt{\eta}\le 7\sqrt{\eta}$. □