SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

October 2024, Vol. 67, Iss. 10, 202101:1–202101:21
https://doi.org/10.1007/s11432-022-3893-3

Committed-programming reductions: formalizations, implications and relations

Jiang ZHANG[1*], Yu YU[2], Dengguo FENG[1], Shuqin FAN[1] & Zhenfeng ZHANG[3]

[1]*State Key Laboratory of Cryptology, Beijing* 100878*, China;*
[2]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai* 200240*, China;*
[3]*Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing* 100190*, China*

Received 22 August 2022/Revised 16 May 2023/Accepted 13 October 2023/Published online 26 September 2024

Abstract In this work, we introduce a class of black-box (BB) reductions called committed-programming reduction (CPRed) in the random oracle model (ROM) and obtain the following interesting results: (1) we demonstrate that some well-known schemes, including the full-domain hash (FDH) signature (Eurocrypt 1996) and the Boneh-Franklin identity-based encryption (IBE) scheme (Crypto 2001), are provably secure under CPReds; (2) we prove that a CPRed associated with an instance-extraction algorithm implies a reduction in the quantum ROM (QROM). This unifies several recent results, including the security of the Gentry-Peikert-Vaikuntanathan IBE scheme by Zhandry (Crypto 2012) and the key encapsulation mechanism (KEM) variants using the Fujisaki-Okamoto transform by Jiang et al. (Crypto 2018) in the QROM. Finally, we show that CPReds are incomparable to non-programming reductions (NPReds) and randomly-programming reductions (RPReds) formalized by Fischlin et al. (Asiacrypt 2010).

Keywords provable security, random oracle model, quantum random oracle model, black-box reduction/separation, programmability

1 Introduction

In the random oracle model (ROM), all parties, including the adversary, are given access to an "idealized" random function, i.e., a random oracle (RO). Since its introduction [1], the ROM has been widely used to design and analyze many well-known schemes, such as the optimal asymmetric encryption padding (OAEP) encryption [2] and the full-domain hash (FDH) signature [3]. Informally, ROM provides black-box (BB) reduction (i.e., the reductions using the adversary as a "BB" oracle) with two main abilities: observability (i.e., seeing all RO queries and answers [4]) and programmability (i.e., setting the answers to the adversary's RO queries [5]). According to the proof techniques enabled by the ROM, researchers [4–6] have tried classifying BB reductions into different classes. At Asiacrypt 2010, Fischlin et al. [5] formalized three notions of ROM BB reductions with different levels of programmability: fully-programming reduction (FPRed), which has full programmability (and contains all ROM BB-reductions), randomly-programming reduction (RPRed), which has limited programmability and is only allowed to program the RO in a restricted way, and non-programming reduction (NPRed), which has no programmability. Notably, all three classes of reductions have full observability. In addition, Fischlin et al. [5] demonstrated that NPRed is strictly weaker than RPRed, which in turn is strictly weaker than FPRed.

Most "honestly-designed" schemes in the ROM seem to keep the security in practice, but the soundness of the ROM has been questioned by researchers [6–8]. Canetti, Goldreich, and Halevi [7] provided the first counterexample, revealing that signature and encryption schemes are secure in the ROM but become insecure when implemented the RO in practice. As we enter the era of post-quantum cryptography (where the cryptosystems are typically classical, but the adversary is modeled with access to quantum computers), the ROM may even be problematic for quantum adversaries. Boneh et al. [9] first introduced the quantum ROM (QROM), where honest parties, such as the post-quantum cryptosystems, still classically access

* Corresponding author (email: jiangzhang09@gmail.com)

the RO but the adversary is explicitly permitted to make quantum queries to the RO. They justified the necessity of the QROM by presenting an artificial identification protocol that is secure in the ROM but is insecure in the QROM. In addition, they noticed that common ROM proof techniques such as programmability and observability (or extractability/preimage awareness in [9]) are not known to carry over to the quantum settings [9].

Since the introduction of QROM [9] at Asiacrypt 2011, we have witnessed considerable progress in studying the security of cryptosystems in the QROM. Some existing ROM schemes such as the FDH signature [3] and the Gentry-Peikert-Vaikuntanathan identity-based encryption (GPV-IBE) [10] are known to be provably secure in the QROM [11–17]. However, most of these security reductions in the QROM are usually specific to the concrete design of the schemes. Moreover, many other ROM schemes are still not known to be provably secure or insecure in the QROM. For example, as shown in [18], some sigma protocols are quantum insecure under assumptions sufficient for classical security. Thus, a question arises: Is there a class of ROM reductions using particular proof techniques that can be lifted to the quantum setting?[1)] If yes, what is the relation between this class and those known, such as NPRed in the literature?

1.1 Our results

In this research, we first introduce committed-programming reduction (CPRed) to abstract a class of ROM BB-reductions that do not use the RO queries of adversaries or adaptively program their answers to simulate the crypto-oracle (such as the signing oracle) queries (and thus the RO simulation can somehow be isolated from other behaviors). Informally, CPReds use two subalgorithms (with some common inputs) that do not interact with each other to simulate the RO queries and the crypto-oracle queries. We show that some well-known schemes [3, 10, 14, 19] such as the FDH signature from trapdoor permutation (TDP) [3] and the Boneh-Franklin IBE (BF-IBE) [20] are provable under CPReds.

Subsequently, we demonstrate that a CPRed associated with an instance-extraction algorithm can be lifted to the QROM. This not only unifies several recent results [9, 11, 16] such as the security of the FDH signature from TDP [3] and the GPV-IBE scheme from learning with error (LWE) [10] by Zhandry [11] and the security of the "implicit-rejection" key encapsulation mechanism (KEM) variants using the Fujisaki-Okamoto transform by Jiang et al. [16], but also provides new security reductions for the "implicit-rejection" KEM variants from TDP [19] in the QROM.

Finally, we compare CPReds with the notions of BB reductions formalized by Fischlin et al. [5] and show that CPReds are incomparable to NPReds and RPReds. This is accomplished by demonstrating that the OAEP encryption from TDP [2], which was proven secure under NPReds by Fujisaki et al. [21], is not provable under CPReds, and that the FDH signature [3], which was shown to be unprovable under RPReds by Fischlin et al. [5], is provably secure under CPReds.

1.2 Technical overview

Formalizations of CPReds. Many classical security proofs in the ROM relies on adaptively programming the RO responses, usually requiring security reductions to copy and store the RO queries from the adversary. However, it is impossible to copy and store the quantum RO (QRO) queries of the adversary due to the quantum no-cloning theorem. As observed by Zhandry [17], this difficulty led researchers to use reductions which basically fix the simulation strategy of the RO at the very beginning. To capture this common feature, we motivate and introduce the notion of CPRed to abstract a class of BB reductions in the ROM for which the RO simulation strategy is fixed before the interaction with the adversary and can somehow be isolated from other behaviors of the reduction such that we can focus on the RO simulation strategy and check the influence on the reduction if the adversary is allowed to make QRO queries.

A common reason for using ROs is that the reduction cannot normally answer some crypto-oracle queries, such as the FDH signing queries, because of the embedding of the target problem (e.g., inverting a TDP instance). Existing reductions in the literature usually apply the following two RO simulation strategies to successfully answer the crypto-oracle queries: (1) The reduction simulates the RO in a particular way such that the adversary may distinguish the simulated RO from a real one; however, the crypto-oracle queries can be successfully answered in regard to the simulated RO if certain conditions on the queries of the adversary are satisfied; or (2) the reduction perfectly simulates the RO for the

1) The history-free reduction given in [9] can be lifted to the QROM, but this notion is restricted to ROM signatures.

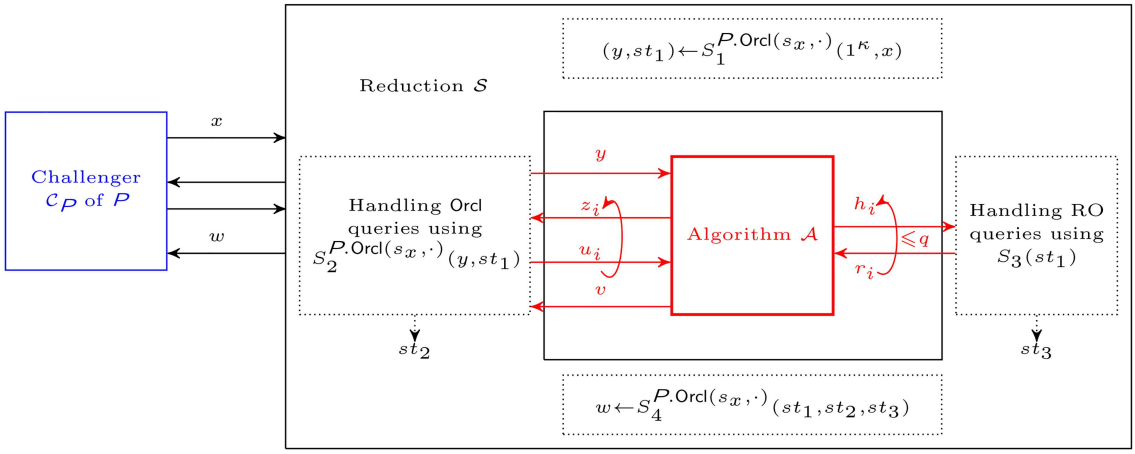

Figure 1 (Color online) A (F, I)-splittable reduction $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ from P to Q, where $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_4$ can have access to $P.\mathsf{Orcl}(s_x, \cdot)$, but $\mathcal{S}_3$ does not access $P.\mathsf{Orcl}(s_x, \cdot)$.

adversary but deviates from the simulated RO in answering some crypto-oracle queries by (implicitly) replacing the responses of the RO at some unknown points to which it cannot directly compute the RO responses with random values, implying that the adversary can detect this inconsistency by making an RO query with any one of those points. Furthermore, in this case, the reduction usually expects to obtain one of the unknown points by interacting with the adversary to solve its own problem.

We now restrict our attention to the reductions that only apply the above two strategies to simulate ROs. Because several ROs may exist, we allow the reduction to apply different strategies to different ROs but only apply one of the two strategies to each RO. We distinguish the ROs for a given reduction $\mathcal{S}$ into two types and consider that an RO is a Type-I RO if $\mathcal{S}$ applies the first simulation strategy to it and a Type-II RO if $\mathcal{S}$ applies the second one to it. Subsequently, we define a splittable property and consider that a reduction $\mathcal{S}$ in the ROM is a splittable reduction if it can be split into four subalgorithms $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ (Figure 1): an initialization subalgorithm $\mathcal{S}_1$ that takes a challenge instance of the underlying hard problem as input and produces some inputs to $\mathcal{S}_2$ and $\mathcal{S}_3$; a subalgorithm $\mathcal{S}_2$ that takes inputs from $\mathcal{S}_1$ and handles all the crypto-oracle queries (e.g., the signing queries) from the adversary as well as abort if certain conditions on the queries of adversary are not satisfied for the simulation of Type-I RO); a subalgorithm $\mathcal{S}_3$ that takes some inputs from $\mathcal{S}_1$ and handles the RO queries from the adversary and may randomly output one of the Type-II RO to enable the reduction to obtain some unknown points; and a finalization algorithm $\mathcal{S}_4$ that produces a solution to the challenge instance using the outputs from $\mathcal{S}_2$ and $\mathcal{S}_3$. Intuitively, $\mathcal{S}_1$ and $\mathcal{S}_4$ are two "internal" subalgorithms for the initial and final work of $\mathcal{S}$, while $\mathcal{S}_2$ and $\mathcal{S}_3$ are two "external" subalgorithms for directly interacting with adversaries and handling the queries from the adversaries. Herein, we allow $\mathcal{S}_2$ and $\mathcal{S}_3$ to abort or terminate before $\mathcal{A}$ terminates. Evidently, $\mathcal{S}_3$ represents all the RO simulation strategies of $\mathcal{S}$, and $\mathcal{S}$ can only use the ability of the adversary via the outputs of $\mathcal{S}_2$ and $\mathcal{S}_3$ to solve its own challenge. The isolation of the RO simulation from other behaviors is achieved by not allowing $\mathcal{S}_2$ and $\mathcal{S}_3$ to interact with each other.

To formally capture the feature that the RO simulation strategies are fixed before interacting with the adversary, we associate a reduction with a deterministic function $F(\cdot, \cdot)$ for Type-I RO and a fixed index function $I(\cdot)$ for Type-II RO. In particular, we consider that $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ is (F, I)-splittable reduction (Definition 2) if: (1) $\mathcal{S}_3$ will only use the deterministic function $F(\tau, \cdot)$ with some input τ from $\mathcal{S}_1$ to answer Type-I RO queries and a real random function $f(\cdot)$ from $\mathcal{S}_1$ for Type-II RO queries, i.e., $\mathcal{S}_3$ will either use $F(\tau, \cdot)$ or $f(\cdot)$ to answer an RO query and (2) given a crypto-oracle query z, $\mathcal{S}_2$ will only deviate at some Type-II RO queries whose positions are specified by $I(z)$ from the Type-II RO simulated by $\mathcal{S}_3$ (i.e., $I(z)$ specifies the positions of all the Type-II RO queries that deviate from the simulated Type-II RO in answering a crypto-oracle query z) by replacing the responses to those RO queries with random values.

A CPRed is a parameterized (F_λ, I)-splittable reduction for $\lambda \in (0, 1)$, satisfying some necessary conditions (Definition 3) such as the probability that $\mathcal{S}_2$ will not abort is noticeable, ensuring the nontriviality of the reduction, and the view of any adversary given access to the simulated Type-I RO using $F_\lambda(\tau, \cdot)$ is not far from the case where it is given a real RO; otherwise, the adversary can distinguish the simulated

game from the real one by simply making Type I RO queries. The term "committed-programming" comes from the facts that $F_\lambda(\cdot,\cdot)$ and $I(\cdot)$ are deterministic and fixed before interacting with the adversary, and that the reduction does not adaptively program the RO for $\mathcal{S}_2$ to handle the crypto-oracle queries.

To demonstrate the nontriviality and usefulness of CPReds, we will show that some well-known schemes [3, 10, 14, 19, 20] such as the FDH signature [3] and the BF-IBE [20] are provably secure under CPReds.

Implications of CPReds. Considering a CPRed $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ from problem P to problem Q, namely, $\mathcal{S}$ can solve problem P by using a BB adversary solving problem Q. The intuition behind that the CPRed $\mathcal{S}$ can be lifted to a reduction from P to Q in the QROM is simple. Because the RO simulation (i.e., $\mathcal{S}_3$) of $\mathcal{S}$ is independent of other behaviors and the simulation of crypto-oracles (i.e., $\mathcal{S}_2$) is oblivious to the RO queries from the adversary $\mathcal{A}$, it seems harmless when $\mathcal{A}$ is given quantum access to the (simulated) RO. Moreover, because $\mathcal{S}_3$ is essentially implemented using functions $F(\tau,\cdot)$ and $f(\cdot)$, we can directly quantize functions F and f to handle QRO queries from $\mathcal{A}$. Our aim is to prove that the reduction $\mathcal{S}' = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}'_3, \mathcal{S}_4)$ is a valid quantum reduction from P to Q, where $\mathcal{S}'_3$ is a quantized version of $\mathcal{S}_3$.

To achieve the above goal, we require the CPRed $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ to have two extra properties (the two properties are merged into a single one in Theorem 1 for simplicity): (1) the view of the quantum algorithm given access to the simulated Type-I RO using $F(\tau,\cdot)$ is not far from the case where it is given a real QRO (note that the Type II RO simulated using a random function $f(\cdot)$ is identical to a real RO) and (2) $\mathcal{S}$ possesses an associated instance-extraction algorithm, which can extract an instance of P (together with some auxiliary information that is needed by $\mathcal{S}_2$ and $\mathcal{S}'_3$) from an instance of Q. The first property is natural and intuitive, as otherwise the adversary can distinguish the security game of Q simulated by $\mathcal{S}'$ from the real one by simply making QRO queries. The second property is somewhat artificial for proving that $\mathcal{S}' = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}'_3, \mathcal{S}_4)$ is a valid reduction from P to Q in the QROM, which requires the establishment of a direct connection between the simulated game of Q by $\mathcal{S}'$ and real one. However, in the real security game of Q there is no conception of P instance, while in the simulated game of Q by $\mathcal{S}'$ a P instance is explicitly required to run $\mathcal{S}_1$ and generate the inputs (e.g., τ) for $\mathcal{S}_2$ and $\mathcal{S}'_3$ (to compute $F(\tau,\cdot)$). To connect the above two games via game sequences in a formal proof, we need to embed a P instance into the security game of Q and generate the necessary information that is required by $\mathcal{S}_2$ and $\mathcal{S}'_3$ in some intermediate game. The instance-extraction algorithm provides a natural way to achieve this, and it exists for some well-known existing schemes. Considering the FDH signature from TDP [3] as an example, given a challenge verification key vk^* of the FDH signature, which is basically a TDP function index, one can naturally extract a problem instance of inverting the underlying TDP by outputting vk together with a uniformly random point y^* selected from the permutation range. In addition, other information required for $\mathcal{S}_2$ and $\mathcal{S}_3$ can be generated by running $\mathcal{S}_1$ with inputs (vk, y^*). Notably, the history-free reduction [9] directly contains a similar instance-extraction algorithm (restricted to the signatures) in the definition.

With the above two properties, the proof that $\mathcal{S}' = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}'_3, \mathcal{S}_4)$ is a valid reduction from P to Q in the QROM is intuitive and direct except that the one-way to hiding (O2H) lemma [14, 22, 23] is carefully applied to handle the case in which $\mathcal{S}_2$ may deviate from the Type-II RO simulation at the RO queries with positions specified by $I(z)$ in answering a crypto-oracle query z. Briefly, the O2H lemma says that if the adversary can make some QRO queries to detect the inconsistency between $\mathcal{S}_2$ and $\mathcal{S}_3$ at some Type-II RO query h with non-negligible probability, then one can recover h from the QRO queries of the adversary with non-negligible probability, which is sufficient for $\mathcal{S}'$ to solve its own problem. Notably, our reduction $\mathcal{S}'$ not only unifies the reductions for the FDH signature [3] and the GPV-IBE scheme in [11] for the preimage sampleable trapdoor function (PSF)-FDH signature [10] in [9] and the "implicit rejections" KEM variants from the Fujisaki-Okamoto (FO) transform [14, 24] given in [16] but also provides new security reductions for the "implicit-rejection" KEM variants from TDP [19] in the QROM.

Incomparability of CPReds. We investigate the relation between CPReds and the notions of ROM BB reductions formalized by Fischlin et al. [5]. We demonstrate that (1) an NPRed does not imply a CPRed (i.e., NPRed $\not\Rightarrow$ CPRed) and (2) a CPRed does not imply an RPRed (i.e., CPRed $\not\Rightarrow$ RPRed). Because an NPRed always implies an RPRed (i.e., NPRed $\Rightarrow$ RPRed) [5], we immediately obtain that CPRed $\not\Rightarrow$ NPRed and RPRed $\not\Rightarrow$ CPRed. This implies that CPReds are incomparable to NPReds and RPReds. Technically, our above results are achieved by investigating the security reductions of two well-known schemes: the OAEP encryption from TDP [2] and the FDH signature [3].

First, we show that the OAEP encryption from TDP [2], which was provably secure under NPReds

by Fujisaki et al. [21], is not provable under CPReds. The key point is that the reduction [21] for the chosen-ciphertext attack (CCA) security of the OAEP encryption relies on the observability of the RO queries to answer the decryption queries. This is not achievable for CPReds because the subalgorithm for simulating the decryption oracle in a CPRed does not interact with the subalgorithm for simulating the RO, and it cannot use the RO queries from the adversary to answer the decryption queries. We use the two-oracle separation technique of Hsiao and Reyzin [25] to rule out the existence of CPReds for the CCA security of the OAEP encryption.

Second, we show that the FDH signature [3], which was shown not provable under RPReds by Fischlin et al. [5], is provably secure under CPReds. Our main observation is that existing reductions for the unforgeability of the FDH signature need to program the RO response such that the corresponding preimage under some permutation is known in any signing query. This is not achievable for an RPRed because it cannot directly set the RO responses; however, it is achievable for a CPRed because the simulation of the RO can be performed using a subalgorithm equipped with two random functions: one for guessing if an RO query from the adversary will be used in a signing query and the other for picking a random preimage to program the RO response if the guess is "yes".

1.3 Open problems and future work

One problem is to find new CPReds applications and obtain security proofs for more schemes in the QROM. Another problem is to generalize our notion of CPReds to obtain a larger class of BB reductions that unify the results in [12, 17, 26–30] in which no CPReds exist in the ROM but are already known to be provably secure in the QROM and establish more connections between the proof techniques in the classical ROM and those in the QROM.

2 Preliminaries

Let κ be the security parameter; the standard notations O and ω are used to classify the growth of functions. Furthermore, log is denoted as the logarithm with base 2. A function $f(\kappa)$ is negligible in κ if for every positive c, we have $f(\kappa) < \kappa^{-c}$ for sufficiently large κ. We denote an arbitrary negligible function using $\mathsf{negl}(\kappa)$. The notation $\xleftarrow{\$}$ denotes randomly selecting elements from a distribution or the uniform distribution over a finite set. We denote ϵ (resp., $\emptyset$) as an empty string (resp., set).

Let $\mathbb{C}$ be the set of complex numbers and $\mathbb{C}^N$ be the complex vector space of N dimension, where $N \geqslant 1$ is an integer. The bra-ket notations of $\langle\cdot|$ and $|\cdot\rangle$ denote row and column vectors in $\mathbb{C}^N$, respectively. For any vector $v \in \mathbb{C}^N$, v^{T} denotes the transpose of v, and v^* denotes the conjugate transpose of v. For any vectors $w = (w_0, \ldots, w_{N-1})^{\mathrm{T}}, v = (v_0, \ldots, v_{N-1})^{\mathrm{T}} \in \mathbb{C}^N$, the inner product between w and v is defined as $\langle w|v\rangle = \sum_{i=0}^{N-1} w_i^* v_i \in \mathbb{C}$.

We briefly recall some background for quantum computation and refer to [31] for more information. Formally, a quantum system $\mathcal{Q}$ with N configurations labeled by $\{0, \ldots, N-1\}$ is associated to the Hilbert space $\mathcal{H}_N = \mathbb{C}^N$ with the inner product $\langle w|v\rangle = \sum_{i=0}^{N-1} w_i^* v_i \in \mathbb{C}$. A pure state of $\mathcal{Q}$ is specified using a column vector $|\phi\rangle \in \mathcal{H}_N$ of norm 1 (i.e., $\langle\phi|\phi\rangle = 1$), which assigns a (complex) weight to each configuration in $\{0, \ldots, N-1\}$. The "computational basis" for $\mathcal{Q}$ is $\{|0\rangle, |1\rangle, \ldots, |N-1\rangle\}$, where $|i\rangle$ assigns weight 1 to configuration i and weight 0 to any other configuration $j \neq i$. By definition, $\{|0\rangle, |1\rangle, \ldots, |N-1\rangle\}$ forms an orthonormal basis for $\mathcal{H}_N$ and any pure state $|\phi\rangle$ can be written as $|\phi\rangle = \sum_i \alpha_i |i\rangle$, where $\sum_i |\alpha_i|^2 = 1$. A qubit is a quantum system with $N = 2$ configurations labeled by $\{0, 1\}$. An n-qubit system is the joint quantum system of n qubits. The standard computational basis $\{|x\rangle\}_{x\in\{0,1\}^n}$ for an n-qubit system is given by $|x_1\rangle \otimes \cdots \otimes |x_n\rangle$, where $x = x_1 \ldots x_n \in \{0, 1\}^n$.

Quantum measurement. Let $B = \{|x\rangle\}_{x\in\mathcal{X}}$ be any orthonormal basis of $\mathcal{H}_N$. Given a pure state $|\phi\rangle \in \mathcal{H}_N$, we can measure it in the basis B, obtaining a value x with probability $|\langle x|\phi\rangle|^2$. This operation induces a probability distribution $D_\phi(x) = |\langle x|\phi\rangle|^2$ over $\mathcal{X}$. After measurement, the state $|\phi\rangle$ collapses to $|x\rangle$ and does not change under subsequent measurements on the same basis B; however, it may still change under measurements on other bases. In addition, we can perform partial measurements on a pure state in a joint quantum system.

Quantum algorithms. A quantum algorithm $\mathcal{A}$ over a Hilbert space $\mathcal{H}_N$ with an orthonormal basis $\{|x\rangle\}_{x\in\mathcal{X}}$ is specified by a unitary transformation U, which takes an initial state $|\phi\rangle$ as input and outputs a result obtained by performing a measurement on the final state $|\psi\rangle = U|\phi\rangle$. A quantum algorithm $\mathcal{A}$

is considered efficient if U comprises a polynomial number of universal basis gates (e.g., the Hadamard, CNOT, and $\pi/8$ gates). Let $\mathcal{X}, \mathcal{Y}$, and $\mathcal{Z}$ be any sets such that $\mathcal{Y}$ is the additive group of $\mathbb{Z}_2^\ell$ for some $\ell \in \mathbb{N}$. Let $B = \{|x, y, z\rangle\}_{x\in\mathcal{X}, y\in\mathcal{Y}, z\in\mathcal{Z}}$ be the corresponding orthonormal basis. For any function $f : \mathcal{X} \to \mathcal{Y}$, define U_f as the unitary transformation that maps $|x, y, z\rangle$ into $|x, y \oplus f(x), z\rangle$. By definition, the inverse transformation of U_f is U_f.

Let O_f be an oracle that computes the unitary transformation U_f. An oracle quantum algorithm A^{O_f} with at most q queries to O_f is specified by a sequence of $q+1$ unitary transformation $U_0, \ldots, U_q$. Specifically, given an initial state $|\phi\rangle$, the algorithm A^{O_f} outputs a result obtained through performing a measurement on the final state $|\psi\rangle = U_q U_f U_{q-1} \cdots U_f U_0 |\phi\rangle$. Similarly, one can define oracle algorithm $A^{O_{f_1},\ldots,O_{f_k}}$ with access to a collection of oracles $\{O_{f_1}, \ldots, O_{f_k}\}$, which is polynomially equivalent to an oracle algorithm B^O with access to a single oracle $O(k, x) = O_{f_k}(x)$. Lemma 1 is implicit in [22].

Lemma 1 (Algorithmic O2H [14, 22])**.** Let $\mathcal{F}$ be the family of functions from $\mathcal{X}$ to $\mathcal{Y}$ and $\mathcal{O} : \mathcal{X} \to \mathcal{Y}$ be a random oracle uniformly selected from $\mathcal{F}$. Let D be an arbitrary probability distribution over $\mathcal{X}$. Let E be any arbitrary (probabilistic) algorithm that takes a pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ as inputs and outputs a bit string $\mathrm{inp} \in \{0, 1\}^*$. Consider an oracle algorithm $\mathcal{A}$ that makes at most q quantum queries to $\mathcal{O}$. Let $\mathcal{B}$ be an oracle algorithm that on input $\mathrm{inp} \in \{0, 1\}^*$ performs the following: pick $k \xleftarrow{\$} \{1, \ldots, q\}$, run $\mathcal{A}^{\mathcal{O}}(\mathrm{inp})$ until (just receiving) the k-th quantum query, measure the argument of the query in the computational basis, and output the measurement outcome (if $\mathcal{A}$ makes less than k queries, $\mathcal{B}$ outputs $\perp$). Let

$$P^1_{\mathcal{A}} := \Pr[b' = 1 : \mathcal{O} \xleftarrow{\$} \mathcal{F}, x \xleftarrow{\$} D, y = \mathcal{O}(x), \mathrm{inp} \leftarrow E(x, y), b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}}(\mathrm{inp})],$$
$$P^2_{\mathcal{A}} := \Pr[b' = 1 : \mathcal{O} \xleftarrow{\$} \mathcal{F}, x \xleftarrow{\$} D,\ y \xleftarrow{\$} \mathcal{Y},\ \mathrm{inp} \leftarrow E(x, y), b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}}(\mathrm{inp})],$$
$$P_{\mathcal{B}} := \Pr[x = x' : \mathcal{O} \xleftarrow{\$} \mathcal{F}, x \xleftarrow{\$} D,\ y \xleftarrow{\$} \mathcal{Y},\ \mathrm{inp} \leftarrow E(x, y), x' \xleftarrow{\$} \mathcal{B}^{\mathcal{O}}(\mathrm{inp})].$$

Then $|P^1_{\mathcal{A}} - P^2_{\mathcal{A}}| \leqslant 2q\sqrt{P_{\mathcal{B}}}$.

We clarify that the original O2H lemma in [14, 22] considers the uniform distribution, i.e., $x \xleftarrow{\$} \mathcal{X}$. However, the proof given in [22] basically applies to any distribution D, which is confirmed in an improved version of the O2H lemma given in [23].

3 CPReds

Let $\mathcal{O}(\cdot) \xleftarrow{\$} \mathcal{F}_{n,m}$ be an RO. A cryptographic problem equipped with the RO $\mathcal{O}(\cdot)$ consisting of three algorithms $P = (\mathsf{IGen}^{\mathcal{O}(\cdot)}, \mathsf{Orcl}^{\mathcal{O}(\cdot)}, \mathsf{Vrfy}^{\mathcal{O}(\cdot)})$. Specifically, the instance generator $\mathsf{IGen}^{\mathcal{O}(\cdot)}$ is a polynomial time algorithm which takes as input the security parameter κ, outputs an instance x and a secret value s_x, i.e., $(x, s_x) \leftarrow \mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa)$. The stateful oracle algorithm $\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_x, \cdot)$ takes as input a query $q \in \{0, 1\}^*$, returns a response r to q. The deterministic verification algorithm $\mathsf{Vrfy}^{\mathcal{O}(\cdot)}$ takes as inputs an instance x, a secret value s_x and a candidate solution w, returns 1 if w is a correct solution of x, and 0 otherwise. Finally, we note that the oracle $\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_x, \cdot)$ may accept many different types of queries (i.e., it is not necessarily restricted to a single functionality), and any polynomial number of ROs can also be easily simulated by using a single one, e.g., "a query (i, h) to a RO" can be treated as "a query h to the i-th RO".

Definition 1 (Hard problem)**.** Let $\mathcal{O}(\cdot) \xleftarrow{\$} \mathcal{F}_{n,m}$ be an RO. A cryptographic problem $P = (\mathsf{IGen}^{\mathcal{O}(\cdot)}, \mathsf{Orcl}^{\mathcal{O}(\cdot)}, \mathsf{Vrfy}^{\mathcal{O}(\cdot)})$ is said to be hard with respect to a threshold function $t(\cdot)$, if for all polynomial time algorithm $\mathcal{A}$, the advantage of $\mathcal{A}$ in the game with a challenger $\mathcal{C}$ who provides inputs to $\mathcal{A}$ and answers $\mathcal{A}$'s queries to the $\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ and the RO $\mathcal{O}(\cdot)$ is negligible in the security parameter κ:

$$\mathrm{Adv}_{P,\mathcal{A}}(1^\kappa) = \Pr\left[\begin{array}{c} \mathcal{O}(\cdot) \xleftarrow{\$} \mathcal{F}_{n,m} \\ (x, s_x) \leftarrow \mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa) \\ w \leftarrow \mathcal{A}^{\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_x,\cdot),\mathcal{O}(\cdot)}(1^\kappa, x) \end{array} : \mathsf{Vrfy}^{\mathcal{O}(\cdot)}(x, s_x, w) = 1\right] - t(\kappa).$$

Typically, we have $t(\kappa) = 0$ for computational problems, and $t(\kappa) = 1/2$ for decisional problems. In this paper, we only consider falsifiable problems, which require that $\mathsf{IGen}, \mathsf{Orcl}$ and Vrfy are polynomial time algorithms. Since a standard cryptographic problem can be seen as a problem $P = (\mathsf{IGen}^{\mathcal{O}(\cdot)}, \mathsf{Orcl}^{\mathcal{O}(\cdot)}, \mathsf{Vrfy}^{\mathcal{O}(\cdot)})$ where the three algorithms do not use the RO $\mathcal{O}(\cdot)$, Definition 1 captures

many cryptographic problems such as small integer solution (SIS) and CCA/chosen-plaintext attack (CPA)-secure public-key encryption (PKE) (in the ROM).

A BB reduction $\mathcal{R}$ from a problem P_1 to another problem P_2 (in the ROM) is a polynomial time oracle algorithm such that $\mathcal{R}^{\mathcal{A}}$ solves P_1 whenever $\mathcal{A}$ solves P_2. A BB-reduction $\mathcal{R}$ in the ROM can exploit various properties (e.g., observability and programmability) of the RO to answer the queries to the "crypto-oracle" $\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ (e.g., the signing oracle) from $\mathcal{A}$.

3.1 Definitions

In the following, we focus on the class of BB reductions in the ROM that only apply the following two strategies to simulate ROs for answering the crypto-oracle queries: (1) the reduction simulates the RO in a particular way such that the adversary may distinguish the simulated RO from a real one, but the crypto-oracle queries can be successfully answered w.r.t. the simulated RO if certain conditions on the adversary's queries are satisfied; (2) the reduction perfectly simulates the RO for the adversary, but it will deviate from the simulated ROs in answering some crypto-oracle queries by (implicitly) replacing the responses of the RO at some unknown points (to which it cannot directly compute the responses) with random values, which means that the adversary can detect this inconsistency by making an RO query with any one of those points. As a problem may involve many ROs, we allow the reduction to apply different simulation strategies for different ROs (but only apply one of the above two strategies to each RO). We distinguish the ROs for a given reduction $\mathcal{S}$ into two types, and say that an RO is a Type-I RO if $\mathcal{S}$ will apply the first RO simulation strategy to it, otherwise a Type-II RO if $\mathcal{S}$ will apply the one to it. Note that for a fixed $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ and query z, when and how $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ uses the RO $\mathcal{O}(\cdot)$ to compute a response to z is also fixed, one can count the RO queries required by $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ in producing the response, and use an index function (taking z as input) to specify the positions of the "bad" Type-II RO queries that deviate from the simulation of the Type-II ROs in this computation.

In Definition 2, we define a splittable property for a BB-reduction in the ROM, which will be used to formalize the notion of CPRed in Definition 3. In both definitions, we will omit the superscript $\mathcal{O}(\cdot)$ and denote problem $P = (\mathsf{IGen}, \mathsf{Orcl}, \mathsf{Vrfy})$ for simplicity of exposition. Intuitively, the splittable property says that the simulation of $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ and the RO $\mathcal{O}(\cdot)$ can be done by using two sub-algorithms that do not interact with each other. We associate a splittable reduction with a pair of deterministic functions (F, I), where $F(\cdot,\cdot)$ is a function for answering Type-I RO queries, and $I(\cdot)$ is an index function specifying the position of the "bad" Type-II RO queries in computing the response to a crypto-oracle query. As a reduction may only have Type-I (resp., Type-II) RO, we allow dummy $I(\cdot) = \emptyset$ (resp., $F = \perp$).

Definition 2 (Splittability)**.** Using the notations above, a BB-reduction $\mathcal{S}$ from problem $P = (\mathsf{IGen}, \mathsf{Orcl}, \mathsf{Vrfy})$ to $Q = (\mathsf{IGen}^{\mathcal{O}(\cdot)}, \mathsf{Orcl}^{\mathcal{O}(\cdot)}, \mathsf{Vrfy}^{\mathcal{O}(\cdot)})$ in the ROM is (F, I)-splittable if for any $\mathcal{A}$ solving Q with at most q RO queries, $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ can be split into four sub-algorithms (as depicted in Figure 1):

(1) Given a security parameter κ and an instance x of P as inputs, $\mathcal{S}$ first runs sub-algorithm $\mathcal{S}_1^{P.\mathsf{Orcl}(s_x,\cdot)}(1^\kappa, x)$ to generate an instance y of Q and a state $\mathrm{st}_1 = (\tau, f)$ consisting of a string τ for Type-I RO and a real random function $f(\cdot)$ for Type-II RO, i.e., $(y, \mathrm{st}_1) \leftarrow \mathcal{S}_1^{P.\mathsf{Orcl}(s_x,\cdot)}(1^\kappa, x)$, which are then used as inputs to run $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ and $\mathcal{S}_3(\mathrm{st}_1)$ to interact with $\mathcal{A}$.

(2) The sub-algorithm $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ takes an instance y of Q and a state $\mathrm{st}_1 = (\tau, f)$ as inputs, invokes a single instance of $\mathcal{A}$ with input y:

(i) Whenever receiving a $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z_i from $\mathcal{A}$, $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ may abort if certain condition on z_i is not satisfied for the simulation of Type-I RO. Otherwise, let $\{h_{i,j}\}_{j\in\{1,\ldots,q_i\}}$ be all possible RO queries (indexed by the order of the query) required for computing a response u_i to z_i. Then, a set $\{r_{i,j}\}_{j\in\{1,\ldots,q_i\}}$ is used as the RO responses to $\{h_{i,j}\}_{j\in\{1,\ldots,q_i\}}$ for generating u_i such that

- if $h_{i,j}$ is a Type-I RO query, $r_{i,j} = F(\tau, h_{i,j})$;
- else if $h_{i,j}$ is a Type-II RO query and $j \in I(z_i)$, randomly choose $r_{i,j}$ with consistency (i.e., the same $r_{i,j}$ is chosen for the same $h_{i,j}$ during the whole simulation)[2)]; otherwise $r_j = f(h_{i,j})$;

(ii) Whenever $\mathcal{A}$ outputs a solution v of y at some time, $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}$ may abort if certain condition on v is not satisfied for the simulation of Type-I RO, otherwise outputs a state st_2 and terminates.

2) In the case that $j \in I(z_i)$, the algorithm $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}$ may not really pick r_j as both values $(h_{i,j}, r_j)$ might come from its oracle $P.\mathsf{Orcl}(s_x,\cdot)$, and are unknown to him.

(3) The sub-algorithm $\mathcal{S}_3(\mathrm{st}_1)$ takes a state $\mathrm{st}_1 = (\tau, f)$ as input, randomly chooses an integer $k^* \xleftarrow{\$} \{1, \ldots, q_2\}$, where q_2 is the maximum number of Type-II RO queries from $\mathcal{A}$. Whenever receiving an RO query h from $\mathcal{A}$, $\mathcal{S}_3$ outputs $\mathrm{st}_3 = h$ and terminates if h is the k^*-th Type-II RO query. Otherwise, answer $\mathcal{A}$ with $r = F(\tau, h)$ if h is a Type-I RO query, or $r = f(h)$ if h is a Type-II RO query. If $\mathcal{A}$ terminates, $\mathcal{S}_3$ outputs $\mathrm{st}_3 = \perp$ and terminates.

(4) Finally, $\mathcal{S}$ computes $w \leftarrow \mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}(\mathrm{st}_1, \mathrm{st}_2, \mathrm{st}_3)$ as his own solution of x.

Note that $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ and $\mathcal{S}_3(\mathrm{st}_1)$ are the exact parts of $\mathcal{S}$ directly interacting with the algorithm $\mathcal{A}$ (as depicted in Figure 1), and solely try to answer the queries from (a single instance of) $\mathcal{A}$. In particular, a splittable reduction $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ will not rewind $\mathcal{A}$. Moreover, $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ and $\mathcal{S}_3(\mathrm{st}_1)$ only share the common input st_1, and will not interact with each other after being invoked. We finally note that $\mathcal{S}_3$ does not have access to $P.\mathsf{Orcl}(s_x, \cdot)$, which is necessary for isolating the behaviors of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}$ and $\mathcal{S}_3$.

For a successful reduction $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$, we need to consider: (1) the difference between the simulated Type-I RO $\mathcal{O}_{F_\lambda}(\cdot) = F(\mathrm{st}_1, \cdot)$ and a real RO $\mathcal{O}(\cdot)$ in the adversary's view. For this, we will use a notation $\gamma(q, \mathcal{O}_{F_\lambda})$:

$$\gamma(q, \mathcal{O}_{F_\lambda}) = \max_{\mathcal{A}} \left| \Pr[\mathcal{A}^{\mathcal{O}_{F_\lambda}(\cdot)}(1^\kappa) = 1] - \Pr[\mathcal{A}^{\mathcal{O}(\cdot)}(1^\kappa) = 1] \right|,$$

where q is a positive integer, and the maximum is taken over all algorithms $\mathcal{A}$ making q classical queries to its oracle; (2) the event E_1 that $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts because certain condition is not satisfied for the simulation of Type-I RO. A non-trivial reduction has to make sure that the probability $\Pr[\neg E_1] = 1 - \Pr[E_1]$ is noticeable; (3) the event E_2 that $\mathcal{A}$ makes a "bad" Type-II RO query $h_{i,j}$ to $\mathcal{S}_3(\mathrm{st}_1)$ such that $h_{i,j}$ is the j-th RO query required for computing a response to some $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z_i and $j \in I(z_i) \neq \emptyset$ (and thus may detect the inconsistence between the behavior of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ and the simulated ROs using $\mathcal{S}_3$). Usually, a non-trivial adversary is ensured to make such a "bad" Type-II query with $h_{i,j}$ such that the reduction can obtain the unknown $h_{i,j}$ for solving its own problem. By definition, we have $\Pr[E_1] = 0$ (or $\Pr[E_2] = 0$, respectively) if there is no Type-I (or Type-II, respectively) RO.

Definition 3 (CPRed)**.** Using the notations of Definition 2 and above, we say that problem $P = (\mathsf{IGen}, \mathsf{Orcl}, \mathsf{Vrfy})$ can be reduced to $Q = (\mathsf{IGen}^{\mathcal{O}(\cdot)}, \mathsf{Orcl}^{\mathcal{O}(\cdot)}, \mathsf{Vrfy}^{\mathcal{O}(\cdot)})$ under $(\{F_\lambda\}_{\lambda\in(0,1)}, I)$-CPRed, or $(\{F_\lambda\}_{\lambda\in(0,1)}, I)$-CPRed in brief, if for any $\lambda \in (0, 1)$, and for any (even unbounded) algorithm $\mathcal{A}$ solving Q with non-negligible advantage ϑ_A by making q_1 $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries and q_2 RO queries, there is a (F_λ, I)-splittable reduction $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ such that

(1) there exists a fixed polynomial $\mathsf{poly}_1(\cdot)$ such that $\Pr[\neg E_1] > \lambda - \mathsf{poly}_1(q_1)\lambda^2$;

(2) there exists a fixed polynomial $\mathsf{poly}_2(\cdot)$ such that $\gamma(q_3, \mathcal{O}_{F_\lambda}) \leqslant \mathsf{poly}_2(q_3)\lambda^2$ for any positive integer q_3 and $\mathcal{O}_{F_\lambda}(\cdot) = F_\lambda(\tau, \cdot)$;

(3) given a $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z, except at the Type-II RO queries with positions specified by $I(z)$, the behavior of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ in computing a response to z, conditioned on E_1 not happening, is statistically close to that of the real $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ with the RO $\mathcal{O}(\cdot)$ being changed to the simulated one by $\mathcal{S}_3$;

(4) if $I(\cdot) = \emptyset$, the advantage that $\mathcal{A}$ outputs a solution v of y to $\mathcal{S}_2^{P\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$, conditioned on E_1 not happening, is at least $\vartheta_A - \gamma(q_3, \mathcal{O}_{F_\lambda})$ for some integer q_3. Moreover, given a solution v of y, $\mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}$ can find a solution w of x;

(5) if $I(\cdot) \neq \emptyset$, the advantage that $\mathcal{A}$ outputs a solution v of y to $\mathcal{S}_2^{P\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$, conditioned on E_1 not happening, is negligible. Moreover, given a "bad" Type-II RO query $h_{i,j}$ for some $j \in I(z_i)$, $\mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}$ can find a solution w of x.

Informally, the first two conditions say that we can set the parameter λ such that (1) $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ does not abort with noticeable probability; (2) the simulated Type-I RO $\mathcal{O}_{F_\lambda}(\cdot)$ is not far from a real RO (note that the simulation of the Type-II RO using a random function $f(\cdot)$ is identical to a real RO). The third condition implies that if the RO queries from $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ are all answered with the simulated RO by $\mathcal{S}_3$, then the behavior of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$, conditioned on $E_1 \vee E_2$ not happening, are statistically close to that of the real $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ in the adversary's view. The last two are the successful condition of $\mathcal{S}$. In particular, the last one says that no adversary can win the game simulated by $\mathcal{S}$, which implies that

any non-trivial adversary will always distinguish the simulated game from a real one by making a "bad" Type-II RO query (i.e., E_2 will always happen for any non-trivial adversary), and any "bad" Type-II RO query is sufficient for $\mathcal{S}$ to solve its own problem P. To better elaborate the notions of CPReds, we will give some concrete examples of CPReds for some well-known existing schemes in Subsection 3.2.

3.2 Examples of CPReds

We now take the FDH signature [3] and the BF-IBE scheme [20] as concrete examples to explain the notion of CPReds.

3.2.1 *The FDH signature*

Let $\mathsf{Perm}(\cdot,\cdot):\mathcal{K}\times\{0,1\}^\ell\to\{0,1\}^\ell$ be some family of TDPs: given a permutation index $s\in\mathcal{K}$, $\mathsf{Perm}(s,\cdot)$ is a permutation. Let $H:\{0,1\}^*\to\{0,1\}^\ell$ be an FDH function modeled as an RO. We first briefly recall the FDH signature from TDP, which consists of three algorithms $\Pi_{\mathrm{FDH}}=(\mathsf{KeyGen},\mathsf{Sign},\mathsf{Verify})$.

• $\mathsf{KeyGen}(1^\kappa)$: the key generation algorithm takes the security parameter κ as input, randomly chooses a permutation index s together with a trapdoor td, sets $\mathrm{vk}=s$ as the verification key and $\mathrm{sk}=\mathrm{td}$ as the signing key, i.e., $(\mathrm{vk},\mathrm{sk})\leftarrow\mathsf{KeyGen}(1^\kappa)$.

• $\mathsf{Sign}(\mathrm{td},\mu)$: given the signing key sk and a message $\mu\in\{0,1\}^*$, the signing algorithm computes $H(\mu)\in\{0,1\}^\ell$ by using the hash function H (modeled as an RO) and $x\in\{0,1\}^\ell$ such that $\mathsf{Perm}(s,x)=H(\mu)$ by using the trapdoor td, return $\sigma=x$ as the signature on message μ, i.e., $\sigma\leftarrow\mathsf{Sign}(\mathrm{td},\mu)$.

• $\mathsf{Verify}(\mathrm{vk},\mu,\sigma)$: the verification algorithm takes as inputs $\mathrm{vk}=s$, a message μ and a signature $\sigma=x$, returns 1 if $\mathsf{Perm}(s,x)=H(\mu)$, and 0 otherwise, i.e., $1/0\leftarrow\mathsf{Verify}(\mathrm{vk},\mu,\sigma)$.

The goal of a reduction for the FDH signature is to break the one-wayness of TDP: given a permutation index s^* and a uniformly random $y^*\xleftarrow{\$}\{0,1\}^\ell$, output a preimage $x^*\in\{0,1\}^\ell$ such that $y^*=\mathsf{Perm}(s^*,x^*)$. We now show that for any adversary $\mathcal{A}$ breaking the existential unforgeability against chosen message attacks (i.e., EUF-CMA, see Appendix A) of the FDH signature with non-negligible advantage ϑ_A by making q_1 signing queries and q_2 RO queries, there is a $(\{F_\lambda\}_{\lambda\in(0,1)},\emptyset)$-CPRed $\mathcal{S}=(\mathcal{S}_1,\mathcal{S}_2,\mathcal{S}_3,\mathcal{S}_4)$ breaking the one-wayness of TDP, which is implicit in [3,9,11]. Specifically, for any constant $\lambda\in(0,1)$, pick positive integers p_1,p such that $\bar{\lambda}=p_1/p$ and $\lambda-\bar{\lambda}\leqslant\lambda^2$. Given a TDP challenge instance (s^*,y^*) as inputs, $\mathcal{S}=(\mathcal{S}_1,\mathcal{S}_2,\mathcal{S}_3,\mathcal{S}_4)$ works as follows:

• $(s^*,\mathrm{st}_1=(f_1,f_2,s^*,y^*))\leftarrow\mathcal{S}_1(1^\kappa,(s^*,y^*))$, where $f_1:\{0,1\}^*\to\{1,\ldots,p\}$ and $f_2:\{0,1\}^*\to\{0,1\}^\ell$ are two random functions (which can be instantiated with k-wise independent functions [11]. We directly use "random functions" to simplify the analysis);

• $\mathcal{S}_2(s^*,\mathrm{st}_1)$ uses $\mathrm{vk}=s^*$ as the challenge verification key to invoke $\mathcal{A}$. For a signing query $z_i\in\{0,1\}^\ell$ from $\mathcal{A}$, $\mathcal{S}_2$ outputs $\mathrm{st}_2=\perp$ and aborts if $f_1(z_i)\leqslant p_1$, and otherwise returns $u_i=f_2(z_i)\in\{0,1\}^\ell$ to $\mathcal{A}$. Whenever $\mathcal{A}$ outputs a forgery $v=(z^*,u^*)$ and terminates, $\mathcal{S}_2$ outputs $\mathrm{st}_2=\perp$ and aborts if $f_1(z^*)>p_1$, otherwise outputs $\mathrm{st}_2=(z^*,u^*)$ and terminates;

• $\mathcal{S}_3(\mathrm{st}_1)$ uses $F_\lambda(\mathrm{st}_1,\cdot)$ to answer an RO query z from $\mathcal{A}$:

$$F_\lambda(\mathrm{st}_1,z)=\begin{cases}y^*, & \text{if } f_1(z)\leqslant p_1;\\ \mathsf{Perm}(s^*,f_2(z)), & \text{otherwise.}\end{cases}$$

Whenever $\mathcal{A}$ terminates, $\mathcal{S}_3$ outputs $\mathrm{st}_3=\perp$ and terminates;

• $\mathcal{S}_4(\mathrm{st}_1,\mathrm{st}_2,\mathrm{st}_3)$ outputs $w^*=u^*$ if $\mathrm{st}_2=(z^*,u^*)\wedge\mathsf{Perm}(s^*,u^*)=y^*$, otherwise outputs a random $x\xleftarrow{\$}\{0,1\}^\ell$.

We first note that for the above reduction $\mathcal{S}$, there is only a Type-I RO H. Let E_1 be the event that $\mathcal{S}_2$ aborts at some time. Then we have the following facts:

(1) As f_1 is a random function, we have $\Pr[\neg E_1]\geqslant\bar{\lambda}(1-\bar{\lambda})^{q_1}$. Using the fact that $(1-\bar{\lambda})^{q_1}\geqslant 1-q_1\bar{\lambda}$ and $\lambda<1$, we have $\Pr[\neg E_1]\geqslant\bar{\lambda}(1-q_1\bar{\lambda})\geqslant\lambda-(2q_1+1)\lambda^2$.

(2) As f_2 is a random function and $y^*\in\{0,1\}^\ell$ is uniformly random, any algorithm making q_3 queries can only distinguish the simulated RO $\mathcal{O}_{F_\lambda}(\cdot)=F_\lambda(\mathrm{st}_1,\cdot)$ from a real RO $\mathcal{O}(\cdot)$ by making two queries z_1,z_2, s.t., $F_\lambda(\mathrm{st}_1,z_1)=F(\mathrm{st}_1,z_2)=y^*$ (i.e., $f_1(z_1)\leqslant p_1\wedge f_1(z_2)\leqslant p_1$). As f_1 is a random function, for any q_3 RO queries, the probability that there is at most one query hitting y^* is $(1-\bar{\lambda})^{q_3}+q_3\bar{\lambda}(1-\bar{\lambda})^{q_3-1}=(1+(q_3-1)\bar{\lambda})(1-\bar{\lambda})^{q_3-1}$. Using the fact that $(1-\bar{\lambda})^{q_3-1}\geqslant 1-(q_3-1)\bar{\lambda}$ and $\lambda-\bar{\lambda}\leqslant\lambda^2$, we have

$$\gamma(q_3,\mathcal{O}_{F_\lambda})\leqslant 1-(1-\bar{\lambda})^{q_3}-q_3\bar{\lambda}(1-\bar{\lambda})^{q_3-1}\leqslant(q_3-1)^2\bar{\lambda}^2\leqslant 2(q_3-1)^2\lambda^2.$$

(3) The behavior of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y,\mathrm{st}_1)$ in answering a signing query, conditioned on E_1 not happening, is identical to that of the real signing oracle with the RO $H(\cdot)$ being changed to the simulated one by $\mathcal{S}_3(\mathrm{st}_1)$.

(4) The advantage that $\mathcal{A}$ outputs a valid forgery (z^*,u^*) (i.e., $\mathsf{Perm}(s^*,u^*)=y^*$), conditioned on E_1 not happening, is at least $\vartheta_A-2(q_3-1)^2\lambda^2$ for $q_3=q_1+q_2$, as the adversary will trigger at most q_3 RO queries. Moreover, given a forgery (z^*,u^*), $\mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}$ will output u^*, s.t., $\mathsf{Perm}(s^*,u^*)=y^*$.

Clearly, the advantage of $\mathcal{S}$ in outputting u^* satisfying $\mathsf{Perm}(s^*,u^*)=y^*$ is at least $(\vartheta_A-2(q_3-1)^2\lambda^2)\cdot\Pr[\neg E_1]\geqslant\lambda\vartheta_A-(2(q_3-1)^2+2q_1+1)\lambda^2$. By setting $\mathsf{poly}_3(q_1,q_2)=2(q_1+q_2-1)^2+2q_1+1$ and $\lambda=\vartheta_A/(2\mathsf{poly}_3(q_1,q_2))$, the advantage of $\mathcal{S}$ is at least $\vartheta_A^2/(4\mathsf{poly}_3(q_1,q_2))$, which is non-negligible if ϑ_A is non-negligible[3)]. This shows that $\mathcal{S}$ is a $(\{F_\lambda\}_{\lambda\in(0,1)},\emptyset)$-CPRed for the EUF-CMA security of the FDH signature from TDP [3]. Besides, the variant PSF-FDH signature where the TDP is replaced with PSF, and the GPV-IBE from LWE [10] are also provably secure under $(\{F_\lambda\}_{\lambda\in(0,1)},\emptyset)$-CPReds, which can be confirmed by inspection of the reductions given in [9–11].

3.2.2 *The BF-IBE*

We now take the IND-CPA security of the BF-IBE scheme [20] as an example to elaborate a bit more on the notion of CPRed (especially on the role of $I(\cdot)$). We first briefly recall the BF-IBE scheme, which consists of four algorithms $\Pi_{\text{BF-IBE}}=(\mathsf{KeyGen},\mathsf{Extract},\mathsf{Enc},\mathsf{Dec})$. Let $\mathbb{G}_1,\mathbb{G}_2$ be two groups of prime order p. Let $e:\mathbb{G}_1\times\mathbb{G}_1\to\mathbb{G}_2$ be a bilinear map and let g be a generator of $\mathbb{G}_1$. Let $H_1:\{0,1\}^*\to\mathbb{G}_1$ and $H_2:\mathbb{G}_2\to\{0,1\}^n$ be two hash functions (both modeled as ROs).

• $\mathsf{KeyGen}(1^\kappa)$. The key generation algorithm takes the security parameter κ as input, randomly chooses $s\xleftarrow{\$}\mathbb{Z}_p^*$, outputs the master public key $\mathrm{mpk}=(g,h=g^s)$ and master secret key $\mathrm{msk}=s$, i.e., $(\mathrm{mpk},\mathrm{msk})\leftarrow\mathsf{KeyGen}(1^\kappa)$.

• $\mathsf{Extract}(\mathrm{msk},\mathrm{id})$. The user key extraction algorithm takes $\mathrm{msk}=s$ and an identity $\mathrm{id}\in\{0,1\}^*$ as inputs, computes $u_{\mathrm{id}}=H_1(\mathrm{id})\in\mathbb{G}_1$ and outputs the user secret key $\mathrm{usk}_{\mathrm{id}}=u_{\mathrm{id}}^s$, i.e., $\mathrm{usk}_{\mathrm{id}}\leftarrow\mathsf{Extract}(\mathrm{msk},\mathrm{id})$.

• $\mathsf{Enc}(\mathrm{mpk},\mathrm{id},\mu)$. The encryption algorithm takes $\mathrm{mpk}=(g,h)$, an identity $\mathrm{id}\in\{0,1\}^*$ and message $\mu\in\{0,1\}^\ell$ as inputs, first compute $u_{\mathrm{id}}=H_1(\mathrm{id})$. Then, it randomly chooses $r\xleftarrow{\$}\mathbb{Z}_p^*$, computes $c_1=g^r,R=e(u_{\mathrm{id}},h)^r=e(u_{\mathrm{id}},g)^{rs}$ and $c_2=\mu\oplus H_2(R)$. Finally, outputs the ciphertext $C_{\mathrm{id}}=(c_1,c_2)$, i.e., $C_{\mathrm{id}}\leftarrow\mathsf{Enc}(\mathrm{mpk},\mathrm{id},\mu)$.

• $\mathsf{Dec}(\mathrm{usk}_{\mathrm{id}},C_{\mathrm{id}})$. The decryption algorithm takes a user secret key $\mathrm{usk}_{\mathrm{id}}=u_{\mathrm{id}}^s$ for some identity $\mathrm{id}\in\{0,1\}^*$ and a ciphertext $C_{\mathrm{id}}=(c_1=g^r,c_2)$ encrypted using identity id as inputs, computes and returns $\mu=c_2\oplus H_2(R)$ where $R=e(u_{\mathrm{id}}^s,g^r)=e(u_{\mathrm{id}},g)^{rs}$, i.e., $\mu\leftarrow\mathsf{Dec}(\mathrm{usk}_{\mathrm{id}},C_{\mathrm{id}})$.

The goal of a reduction for the semantic security (i.e., IND-ID-CPA security, see Appendix B) of the BF-IBE scheme is to solve the bilinear Diffie-Hellman (BDH) problem: given (g,g^a,g^b,g^c) with uniformly random $a,b,c\xleftarrow{\$}\mathbb{Z}_q^*$ as inputs, output $e(g,g)^{abc}$. In order to use the ability of the adversary attacking the IND-ID-CPA security of the BF-IBE scheme to solve the BDH problem, the reduction has to answer two types of crypto-oracles queries (i.e., user key extraction query and challenge ciphertext query), and two ROs (i.e., H_1 and H_2) queries from the adversary. We now show that for any adversary $\mathcal{A}$ breaking the semantic security of the BF-IBE scheme with advantage ϑ_A by making q_1 user key extraction queries, a single challenge ciphertext query, $q_{2,1}$ queries to H_1 and $q_{2,2}$ queries to H_2 (i.e., the total RO queries is $q_2=q_{2,1}+q_{2,2}$), there is a (F,I)-CPRed $\mathcal{S}=(\mathcal{S}_1,\mathcal{S}_2,\mathcal{S}_3,\mathcal{S}_4)$ solving the BDH problem, which is implicit in [20]. As before, for any constant $\lambda\in(0,1)$, pick positive integers p_1,p such that $\bar{\lambda}=p_1/p$ and $\lambda-\bar{\lambda}\leqslant\lambda^2$. Given a BDH challenge instance $\mathsf{inst}=(g,g^a,g^b,g^c)$ as inputs, $\mathcal{S}=(\mathcal{S}_1,\mathcal{S}_2,\mathcal{S}_3,\mathcal{S}_4)$ works as follows:

(i) $(\mathrm{mpk}=(g,g^a),\mathrm{st}_1=(\tau,f_2))\leftarrow\mathcal{S}_1(1^\kappa,\mathsf{inst})$, where $\tau=(f_{1,1},f_{1,2},\mathsf{inst})$, $f_{1,1}:\{0,1\}^*\to\{1,\ldots,p\}$, $f_{1,2}:\{0,1\}^*\to\mathbb{Z}_p^*$ and $f_2:\mathbb{G}_2\to\{0,1\}^\ell$ are three random functions (this implicitly sets the master secret key $\mathrm{msk}=a$);

(ii) $\mathcal{S}_2((g,g^a),\mathrm{st}_1)$ uses $\mathrm{mpk}=(g,h=g^a)$ as the master public key to invoke $\mathcal{A}$, and answers $\mathcal{A}$'s queries as follows:

• Whenever receiving a user key extraction query $\mathrm{id}\in\{0,1\}^*$ from $\mathcal{A}$, if $f_{1,1}(\mathrm{id})\leqslant p_1$, output $\perp$ and abort, else compute $u_{\mathrm{id}}=H_1(\mathrm{id})=g^{f_{1,2}(\mathrm{id})}$ (by simulating an H_1 query id), and return the user secret

3) Note that one can use a more precise argument to obtain a tighter argument. Here, we only focus on showing that $\mathcal{S}$ is a CPRed.

key $\mathrm{usk}_{\mathrm{id}} = u_{\mathrm{id}}^{a} = h^{f_{1,2}(\mathrm{id})}$ to $\mathcal{A}$;

• When receiving a challenge ciphertext query $(\mathrm{id}^*, \mu_0, \mu_1)$ for $\mathrm{id}^* \in \{0,1\}^*$ and $\mu_0, \mu_1 \in \{0,1\}^n$, if $f_{1,1}(\mathrm{id}^*) > p_1$, output $\perp$ and abort. Otherwise, compute $u_{\mathrm{id}^*} = H_1(\mathrm{id}^*) = g^b$ (by simulating an H_1 query id^*), and return the challenge ciphertext $C_{\mathrm{id}^*} = (g^c, \mu_\delta \oplus \rho)$ by randomly choosing $\rho \xleftarrow{\$} \{0,1\}^n$ and $\delta \xleftarrow{\$} \{0,1\}$ (i.e., implicitly setting $H_2(R) = \rho$ for $R = e(u_{\mathrm{id}^*}, h)^c = e(g,g)^{abc}$).

Whenever $\mathcal{A}$ outputs a guess $\delta' \in \{0,1\}$ for δ and terminates, $\mathcal{S}_2$ outputs $\mathrm{st}_2 = \perp$ and terminates;

(iii) $\mathcal{S}_3(\mathrm{st}_1)$ parses $\mathrm{st}_1 = (\tau, f_2)$ and uniformly chooses an integer $k^* \xleftarrow{\$} \{1, \ldots, q_{2,2}\}$ at random. Then, it answers $\mathcal{A}$'s RO queries as follows:

• Whenever receiving an H_1 query $\mathrm{id} \in \{0,1\}^*$ from $\mathcal{A}$, compute and return $F_\lambda(\tau, \mathrm{id})$ using the following function:

$$F_\lambda(\tau, \mathrm{id}) = \begin{cases} g^b, & \text{if } f_{1,1}(\mathrm{id}) \leqslant p_1; \\ g^{f_{1,2}(\mathrm{id})}, & \text{if } f_{1,1}(\mathrm{id}) > p_1. \end{cases}$$

• Whenever receiving an H_2 query $R \in \mathbb{G}_2$ from $\mathcal{A}$, if it is the k^*-th query to H_2 (i.e., $t = 2$), outputs $\mathrm{st}_3 = R$ and terminates. Otherwise, return $f_2(R)$ to $\mathcal{A}$.

Whenever $\mathcal{A}$ terminates, $\mathcal{S}_3$ outputs $\mathrm{st}_3 = \perp$ and terminates;

(iv) $\mathcal{S}_4(\mathrm{st}_1, \mathrm{st}_2, \mathrm{st}_3)$ returns $w^* = R$ if $\mathrm{st}_3 = R$, else return a random $R \xleftarrow{\$} \mathbb{G}_2$.

We first note that for the above reduction $\mathcal{S}$, there is a Type-I RO H_1 and a Type-II RO H_2. Let E_1 be the event that $\mathcal{S}_2$ aborts at some time. Let E_2 be the event that the adversary makes a "bad" H_2 query $R = e(g,g)^{abc} \in \mathbb{G}_2$. Then, we have the following facts:

(1) $\Pr[\neg E_1] \geqslant \lambda - (2q_1 + 1)\lambda^2$ by the same analysis as for the FDH signature;

(2) $\gamma(q_3, \mathcal{O}_{F_\lambda}) \leqslant 2(q_3 - 1)^2\lambda^2$ for any positive integer q_3 and $\mathcal{O}_{F_\lambda}(\cdot) = F_\lambda(\mathrm{st}_1, \cdot)$ by the same analysis as for the FDH signature;

(3) The behavior of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$, conditioned on E_1 not happening, only deviates at the RO queries specified by $I(z)$ in computing the response to the challenger ciphertext query z from that of the real one with the ROs being changed to the simulated RO by $\mathcal{S}_3(\mathrm{st}_1)$, where

$$I(\mathrm{query}) = \begin{cases} \emptyset, & \text{if "query" is a user key extraction query;} \\ 2, & \text{if "query" is a challenge ciphertext query.} \end{cases}$$

In other words, the reduction will only replace the second RO query in generating the challenge ciphertext with a random one (as it does not know $R = e(g,g)^{abc}$, and cannot normally compute $H_2(R)$).

(4) The advantage that $\mathcal{A}$ outputs $\delta' = \delta$, conditioned on E_1 not happening, is 0 (as the random ρ perfectly hides δ from $\mathcal{A}$). Moreover, given a "bad" H_2 query $R = e(g,g)^{abc}$, $\mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}$ will output a valid solution R.

Using the above facts, we have $\Pr[E_2] \geqslant (\vartheta_A - 2(q_3-1)^2\lambda^2) \cdot \Pr[\neg E_1]$ by a simple analysis. Moreover, if E_2 happens, the probability that $\mathcal{S}$ can output $R = e(g,g)^{abc}$ is at least $1/q_{2,2}$. Thus, the advantage of $\mathcal{S}$ outputting $R = e(g,g)^{abc}$ is at least $(\vartheta_A \cdot \Pr[\neg E_1] - 2(q_3-1)^2\lambda^2)/q_{2,2} \geqslant (\lambda\vartheta_A - 2\lambda^2 - (2q_1+1)\lambda^2\vartheta_A)/q_{2,2}$, where $q_3 = q_1 + q_{2,1} + 1$. By setting $\mathsf{poly}(q_1, q_{2,1}) = 2q_1 + 1 + 2(q_1 + q_{2,1})^2$ and $\lambda = \vartheta_A/(2\mathsf{poly}(q_1, q_{2,1}))$, the advantage of $\mathcal{S}$ is at least $\vartheta_A^2/(2q_{2,2}\mathsf{poly}(q_1, q_{2,1}))$, which is non-negligible if ϑ_A is non-negligible. This shows that $\mathcal{S}$ is a $(\{F_\lambda\}_{\lambda \in (0,1)}, I)$-CPRed for the BF-IBE scheme [20]. Besides, the "implicit rejection" variant of the KEM from TDP due to Shoup [19], and the "implicit rejection" [14] variants of the KEM from the FO transform [24] are also provably secure under $(\emptyset, I)$-CRPeds for some index function I.

4 Lifting CPReds to reductions in the QROM

In this section, we show that if a problem P can be reduced to Q under CPReds associated with an instance-extraction algorithm in the ROM, then there is a reduction from P to Q in the QROM. Informally, the instance-extraction algorithm can extract an instance of problem P (and produce an auxiliary information for the RO simulation) from an instance of Q, which is needed to ensure that the sub-algorithm (of the CPRed) for simulating the RO can be "safely" quantized to simulate a QRO, and that one can smoothly replace a real QRO with a simulated one in the security proof. We note that the history reduction given in [9] also requires a similar (and possibly stronger) algorithm.

Theorem 1. Using the notations of Definition 3, if problem $P = (\mathsf{IGen}, \mathsf{Orcl}, \mathsf{Vrfy})$ can be reduced to $Q = (\mathsf{IGen}^{\mathcal{O}(\cdot)}, \mathsf{Orcl}^{\mathcal{O}(\cdot)}, \mathsf{Vrfy}^{\mathcal{O}(\cdot)})$ under $(\{F_\lambda\}_{\lambda\in(0,1)}, I)$-CPRed, and for any constant λ, the (F_λ, I)-splittable reduction $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ is associated with an instance-extraction algorithm $Q.\mathsf{Extract}(\cdot,\cdot)$ for Q such that for any $(y, s_y) \leftarrow Q.\mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa)$ and $(x, s_x, \mathrm{st}_1 = (\tau, f)) \leftarrow Q.\mathsf{Extract}(y, s_y)$:

• There exists a fixed polynomial $\mathsf{poly}_2'(\cdot)$ such that given y and s_y, the output distribution of any (even unbounded) algorithm $\mathcal{A}$ making q_3 quantum queries to the oracle $\mathcal{O}_{F_\lambda}(\cdot) : |h_1, h_2\rangle \to |h_1, F_\lambda(\tau, h_1) \oplus h_2\rangle$ is at most $\mathsf{poly}_2'(q_3)\lambda^2$ far from the case when $\mathcal{A}$ is given access to $\mathcal{O}_{f_1}(\cdot) : |h_1, h_2\rangle \to |h_1, f_1(h_1) \oplus h_2\rangle$ with a uniform $f_1 \xleftarrow{\$} \mathcal{F}_{n,m}$[4].

• The distribution of (x, s_x, y, τ) is identical to that of $(x', s_x', y', \mathrm{st}_1')$ generated by $(x', s_x') \leftarrow P.\mathsf{IGen}(1^\kappa)$ and $(y', \mathrm{st}_1') \leftarrow \mathcal{S}_1^{P.\mathsf{Orcl}(s_x', \cdot)}(1^\kappa, x)$, over the randomness used by $Q.\mathsf{IGen}^{\mathcal{O}(\cdot)}, Q.\mathsf{Extract}$, $P.\mathsf{IGen}$ and $\mathcal{S}_1^{P.\mathsf{Orcl}(s_x', \cdot)}$, then for any quantum polynomial time algorithm $\mathcal{A}$ against Q using at most q_1 queries to $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ and q_2 QRO queries such that $dq_1 + q_2 \leqslant q_3$, there is a reduction $\mathcal{S}'$ from P to Q in the QROM, where d is the maximum number of RO queries that is required for computing a response to any $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query.

One can check that the schemes discussed in Subsection 3.2 have CPReds associated with natural instance-extraction algorithms. Taking the FDH signature [3] as an example, let $((\mathrm{vk}, y^*), \mathrm{sk}, \mathrm{st}_1 = (f_1, f_2, \mathrm{vk}, y^*)) \leftarrow \mathsf{Extract}(\mathrm{vk}, \mathrm{sk})$ be an algorithm which takes a verification key vk and a signing key sk as input, outputs (vk, y^*), sk and $\mathrm{st}_1 = (f_1, f_2, \mathrm{vk}, y^*)$ by randomly choosing $y^* \xleftarrow{\$} \{0,1\}^\ell$, $f_1 : \{0,1\}^* \to \{1, \ldots, p\}$ and $f_2 : \{0,1\}^* \to \{0,1\}^\ell$. First, the above algorithm $\mathsf{Extract}(\mathrm{vk}, \mathrm{sk})$ satisfies the second condition for the CPRed reduction given in Subsection 3.2. Second, by Zhandry's RO simulation technique (i.e., replacing the random functions (f_1, f_2) with k-wise independent functions) and [11, Corollary 4.3], one can check that the above $\mathsf{Extract}(\mathrm{vk}, \mathrm{sk})$ also satisfies the first condition.

By Theorem 1, we have that there are reductions from the security of the FDH signature [3] and its variant the PSF-FDH signature, the GPV-IBE scheme [10], and the "implicit-rejection" KEM variants from TDP [19] and the FO transform [14,24], to their underlying hard problems in the QROM. We clarify that Theorem 1 provides guarantee on the security of a scheme only when its underlying problems are hard for quantum polynomial time adversaries. In particular, Theorem 1 implies that there is a reduction from the IND-ID-CPA security of the BF-IBE scheme [20] to the hardness of the BDH problem in the QROM, but it provides no guarantee on the security of BF-IBE scheme against quantum adversaries since the BDH problem can be efficiently solved by using the Shor algorithm [32]. We finally note that Theorem 1 not only unifies the security reductions for the FDH signature [3], the PSF-FDH signature and the GPV-IBE scheme [10] given in [9, 11], and for the "implicit rejections" KEMs from the FO transform [14, 24] given in [16], but also gives new security reductions for the "implicit-rejection" KEM variants from TDP [19] in the QROM.

We now give a formal proof of Theorem 1, which are mainly based on the facts that (1) for a $(\{F_\lambda\}_{\lambda\in(0,1)}, I)$-CPRed, the simulation of the RO and that of the $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ are relatively independent; and that (2) the simulation of the RO can be "safely" quantized by the assumption in Theorem 1.

Proof. Let $\mathcal{A}$ be any quantum algorithm that solves problem Q with non-negligible advantage ϑ_A, by making at most q_1 queries to $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ and q_2 QRO queries. Let $\lambda \in (0,1)$ be a parameter that will be determined later. Let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4)$ be the (F_λ, I)-splittable reduction in Definition 3. We first give the description of the reduction $\mathcal{S}'$.

Formally, given the security parameter κ and an instance x of P as inputs, $\mathcal{S}'$ first computes $(y, \mathrm{st}_1 = (\tau, f)) \leftarrow \mathcal{S}_1^{P.\mathsf{Orcl}(s_x, \cdot)}(1^\kappa, x)$ and sets $\mathrm{st}_2 = \mathrm{st}_3 = \epsilon$. Then, it uniformly chooses $k^* \xleftarrow{\$} \{1, \ldots, q_2\}$. Define the quantum oracle $\mathcal{O}_{F_\lambda}(\cdot) : |h_1, h_2\rangle \to |h_1, F_\lambda(\tau, h_1) \oplus h_2\rangle$ and $\mathcal{O}_f(\cdot) : |h_1, h_2\rangle \to |h_1, f(h_1) \oplus h_2\rangle$. Then, $\mathcal{S}'$ invokes $\mathcal{A}$ with input y, answers the $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries from $\mathcal{A}$ by running $\mathrm{st}_2 \leftarrow \mathcal{S}_2^{P.\mathsf{Orcl}(s_x, \cdot)}(y, \mathrm{st}_1)$ and handles a QRO query $|\phi\rangle = |h_1, h_2\rangle$ from $\mathcal{A}$ as follows: if $|\phi\rangle$ is a Type-I RO query, return $|\psi\rangle = \mathcal{O}_{F_\lambda}(|\phi\rangle)$ to $\mathcal{A}$. Otherwise, if $|\phi\rangle$ is the k^*-th Type-II RO query, measure the argument of the query and set st_3 to be the measurement outcome, else return $\mathcal{O}_f(|\phi\rangle)$ to $\mathcal{A}$. If $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x, \cdot)}(y, \mathrm{st}_1)$ aborts, then $\mathcal{S}'$ runs a trivial algorithm for problem P and outputs whatever it obtains. Otherwise, $\mathcal{S}'$ computes and outputs $w \leftarrow \mathcal{S}_4^{P.\mathsf{Orcl}(s_x, \cdot)}(\mathrm{st}_1, \mathrm{st}_2, \mathrm{st}_3)$ as the solution to its own challenge x of problem P.

It suffices to show that $\mathcal{S}'$ is a valid reduction in the QROM. We will finish the proof by using a sequence

4) The fact that $\mathcal{A}$ has the knowledge of (y, s_y) is very crucial for the proof of Theorem 1 because we need to consider an adversary that can make $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, \cdot)$ queries, and in particular it may trigger an RO query depending on the information of s_y by making some $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, \cdot)$ query.

Table 1 Outline of the changes between two consecutive games

Game	Changes w.r.t. previous game	Note
G_0	Instance: pick a $f_1, f_2 \xleftarrow{\$} \mathcal{F}_{n,m}$, and set $\mathcal{O}_{f_1}(\cdot): \lvert h_1, h_2\rangle \to \lvert h_1, f_1(h_1) \oplus h_2\rangle$ $\mathcal{O}_{f_2}(\cdot): \lvert h_1, h_2\rangle \to \lvert h_1, f_2(h_1) \oplus h_2\rangle$ compute $(y, s_y) \leftarrow Q.\mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa)$ $Q.\mathsf{Orcl}$ query z: compute $u = Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, z)$ QRO query $\lvert\phi\rangle$: compute $\mathcal{O}_{\text{quant}}(\phi)$	Real game
G_1	Compute $(x, s_x, \mathrm{st}_1 = (\tau, f)) \leftarrow Q.\mathsf{Extract}(y, s_y)$. After $\mathcal{A}$ outputs a solution and terminates, simulate the behavior of $\mathcal{A}$ to $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$, and omit the output of $\mathcal{A}$ if $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts during the simulation	A conceptual change of G_0 in the adversary's view
G_2	Answer the RO queries using $\mathcal{O}_{F_\lambda}(\cdot): \lvert h_1, h_2\rangle \to \lvert h_1, F_\lambda(\tau, h_1) \oplus h_2\rangle$ and $\mathcal{O}_f(\cdot): \lvert h_1, h_2\rangle \to \lvert h_1, f(h_1) \oplus h_2\rangle$ for Type I and Type II RO queries, respectively (i.e., replace $\mathcal{O}_{f_1}(\cdot)$ and $\mathcal{O}_{f_2}(\cdot)$ in game G_1 with $\mathcal{O}_{F_\lambda}(\cdot)$ and $\mathcal{O}_f(\cdot)$, respectively)	The difference on the outputs between G_2 and G_1 is bounded by $\mathsf{poly}_2'(q_3)\lambda^2$
G_3	For each $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z, immediately feed z to $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$, and abort if $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts	A conceptual change of G_2
$G_{3,j}$	Let $k_j = \lfloor \frac{j-1}{d} \rfloor + 1$ and $\ell_j = j \mod d$. For the ℓ_j-th RO query h_{k_j,ℓ_j} in computing the response to the k_j-th $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z_{k_j} and $\ell_j \in I(z_{k_j})$, namely, if h_{k_j,ℓ_j} is a "bad" Type-II RO query, replace $f(h_{k_j,\ell_j})$ with a random and consistent r_{k_j,ℓ_j}	Let $G_{3,0} = G_3$, and then games $G_{3,j-1}$ and $G_{3,j}$ are connected by the O2H Lemma 1
G_4	For each $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z, replace $f(h_\ell)$ with a random and consistent r_ℓ if h_ℓ is ℓ-th RO query in computing the response to z and $\ell \in I(z)$, namely, h_ℓ is a "bad" Type-II RO query	A conceptual change of G_{3,dq_1}
G_5	Compute $(x, s_x) \leftarrow P.\mathsf{IGen}(1^\kappa), (y, \mathrm{st}_1) \leftarrow \mathcal{S}_1^{P.\mathsf{Orcl}(s_x,\cdot)}(x)$, and use $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ to answer all $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries	The responses to $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries generated by $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ are statistically close to those generated in G_4

of six main games G_i for $i \in \{0, \ldots, 5\}$, where game G_0 is the real game and game G_5 is essentially the game simulated by $\mathcal{S}'$. For technical reason, we also define a sequence of sub-games $G_{3,j}$ between the two main games G_3 and G_4 for $j = \{0, \ldots, dq_1\}$ such that $G_{3,0} = G_3$, $G_{3,dq_1} = G_4$, and $G_{3,j}$ is modified from $G_{3,j-1}$ by replacing at most one Type-II RO response with a uniformly random one (to apply Lemma 1). We outline the changes between two consecutive games in Table 1. The game sequences are formally given below.

Game G_0**.** This game is the real security game of Q in the QROM. Concretely, the challenger $\mathcal{C}_Q$ picks two functions $f_1 \xleftarrow{\$} \mathcal{F}_{n,m}$ and $f_2 \xleftarrow{\$} \mathcal{F}_{n,m}$ for Type-I and Type-II RO queries, respectively. Set $\mathcal{O}_{f_1}(\cdot): \lvert h_1, h_2\rangle \to \lvert h_1, f_1(h_1) \oplus h_2\rangle$ and $\mathcal{O}_{f_2}(\cdot): \lvert h_1, h_2\rangle \to \lvert h_1, f_2(h_1) \oplus h_2\rangle$. Then, computes $(y, s_y) \leftarrow Q.\mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa)$, and invokes $\mathcal{A}$ with input y:

• When receiving a QRO query $\lvert\phi\rangle = \lvert h_1, h_2\rangle$ from $\mathcal{A}$, return $\mathcal{O}_{f_1}(\lvert\phi\rangle)$ to $\mathcal{A}$ if this is a Type-I RO query, else return $\mathcal{O}_{f_2}(\lvert\phi\rangle)$ if this is a Type-II RO query.

• When receiving a $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z from $\mathcal{A}$, compute $u = Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, z)$ by using s_y. Whenever the computation stops to make an RO query h, use $f_1(h)$ if h is a Type-I RO query or $f_2(h)$ if h is a Type-II RO query to continue the computation. Finally, return u to $\mathcal{A}$.

• When $\mathcal{A}$ outputs a solution v of y, compute and output $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v)$.

Let Succ_i be the event that the the challenger outputs 1, i.e., the algorithm $\mathcal{A}(y)$ succeeds to output a solution v of y such that $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v) = 1$, where $i \in \{0, \ldots, 5\}$. By definition, the advantage of $\mathcal{A}$ in game G_0 is equal to $\Pr[\mathsf{Succ}_0] - t(\kappa)$, i.e., $\Pr[\mathsf{Succ}_0] = \vartheta_A + t(\kappa)$, where $t(\cdot)$ is the threshold function for problem Q in Definition 1.

Game G_1**.** This game is almost identical to game G_0 except the following: the challenger $\mathcal{C}_Q$ records all the $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries from $\mathcal{A}$. After $\mathcal{A}$ finishes and outputs v, the challenger $\mathcal{C}_Q$ computes $(x, s_x, \mathrm{st}_1) \leftarrow Q.\mathsf{Extract}(y, s_y)$, and simulate the behavior of $\mathcal{A}$ to $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$. If $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts during the simulation, $\mathcal{C}_Q$ runs a trivial solving algorithm to find a solution v' of y and outputs $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v')$ (in this case, we have $\Pr[Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v') = 1] = t(\kappa) + \mathsf{negl}(\kappa)$. Otherwise, $\mathcal{C}_Q$ outputs $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v)$.

Let E_1 be the event that $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}$ aborts in game G_1.

Lemma 2. $\Pr[\mathsf{Succ}_1] \geqslant t(\kappa) + \lambda\vartheta_A - \mathsf{poly}_1(q_1)\lambda^2 + \mathsf{negl}(\kappa)$, where $\mathsf{poly}_1(\cdot)$ is the fixed polynomial in Definition 3.

Proof. Note that the view of $\mathcal{A}$ in both games G_0 and G_1 are identical. Thus, if E_1 does not happen, the probability that $\mathcal{C}_Q$ outputs 1 in game G_1 is equal to that in game G_0 (i.e., $\Pr[\mathsf{Succ}_1|\neg E_1] = \Pr[\mathsf{Succ}_0]$), else the probability that $\mathcal{C}_Q$ outputs 1 is $t(\kappa) + \mathsf{negl}(\kappa)$ (i.e., $\Pr[\mathsf{Succ}_1|E_1] = t(\kappa) + \mathsf{negl}(\kappa)$). By the law of total probability, we have that $\Pr[\mathsf{Succ}_1] = \vartheta_A \cdot \Pr[\neg E_1] + t(\kappa) + \mathsf{negl}(\kappa)$. In addition, by Definition 3 the probability $\Pr[\neg E_1] \geqslant \lambda - \mathsf{poly}_1(q_1)\lambda^2$ for any even unbounded algorithm $\mathcal{A}$. Since $0 \leqslant \vartheta_A \leqslant 1$, we have that $\Pr[\mathsf{Succ}_1] \geqslant t(\kappa) + \lambda\vartheta_A - \mathsf{poly}_1(q_1)\lambda^2 + \mathsf{negl}(\kappa)$. This completes the proof.

Game G_2. This game is almost identical to game G_1 except the following: after obtaining (y, s_y), $\mathcal{C}_Q$ computes $(x, s_x, \mathrm{st}_1 = (\tau, f)) \leftarrow Q.\mathsf{Extract}(y, s_y)$. Then, it answers all the Type-I and Type-II RO queries using $F_\lambda(\mathrm{st}_1, \cdot)$ and $f(\cdot)$ for classical queries from $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ and $\mathcal{O}_{F_\lambda}(\cdot) : |h_1, h_2\rangle \to |h_1, F_\lambda(\mathrm{st}_1, h_1) \oplus h_2\rangle$ and $\mathcal{O}_f(\cdot) : |h_1, h_2\rangle \to |h_1, f(h_1) \oplus h_2\rangle$ for QRO queries from $\mathcal{A}$.

Lemma 3. There exist is a fixed polynomial $\mathsf{poly}_3(\cdot,\cdot)$ such that for $\lambda = 2\vartheta_A/\mathsf{poly}_3(q_1, q_3)$, we have $\Pr[\mathsf{Succ}_2] \geqslant t(\kappa) + \vartheta_A^2/\mathsf{poly}_3(q_1, q_3) + \mathsf{negl}(\kappa)$.

Proof. As f is real random function, the oracle $\mathcal{O}_f(\cdot)$ in game G_2 is identical to $\mathcal{O}_{f_2}(\cdot)$ in game G_1. Thus, the only difference between games G_2 and G_1 is that $\mathcal{C}_Q$ replaces the real RO $\mathcal{O}_{f_1}(\cdot)$ in game G_1 with $\mathcal{O}_{F_\lambda}(\cdot)$ in game G_2. Since $\mathcal{A}$ will make at most q_1 $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries (each $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query requires at most d RO queries) and q_2 QRO queries, the output distribution of $\mathcal{A}$ in game G_2 is at most $\mathsf{poly}_2'(q_3)\lambda^2$ to that in game G_1 by assumption, where $q_3 \geqslant dq_1 + q_2$. Thus, $\Pr[\mathsf{Succ}_2] \geqslant \Pr[\mathsf{Succ}_1] - \mathsf{poly}_2'(q_3)\lambda^2$. As $\Pr[\mathsf{Succ}_1] \geqslant t(\kappa) + \lambda\vartheta_A - \mathsf{poly}_1(q_1)\lambda^2 + \mathsf{negl}(\kappa)$, by setting $\mathsf{poly}_3(q_1, q_3) = 4(\mathsf{poly}_1(q_1) + \mathsf{poly}_2'(q_3))$ and $\lambda = 2\vartheta_A/\mathsf{poly}_3(q_1, q_3)$, we have $\Pr[\mathsf{Succ}_2] \geqslant t(\kappa) + \lambda\vartheta_A - (\mathsf{poly}_1(q_1) + \mathsf{poly}_2'(q_3))\lambda^2 + \mathsf{negl}(\kappa) \geqslant t(\kappa) + \vartheta_A^2/\mathsf{poly}_3(q_1, q_3) + \mathsf{negl}(\kappa)$, which completes the proof.

Game G_3. This game is almost identical to game G_2 except that whenever receiving a $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z from $\mathcal{A}$, $\mathcal{C}_Q$ first simulates the response to z by using $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$. If $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts, $\mathcal{C}_Q$ runs a trivial solving algorithm to find a solution v' of y, outputs $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v')$ and terminates. Otherwise, $\mathcal{C}_Q$ performs as in game G_2.

Lemma 4. $\Pr[\mathsf{Succ}_3] = \Pr[\mathsf{Succ}_2]$.

Proof. This lemma follows because game G_3 is a conceptual change of G_2.

Let $G_{3,0} = G_3$ and $\mathsf{Succ}_{3,0} = \mathsf{Succ}_3$. For $1 \leqslant j \leqslant dq_1$, define $k_j = \lfloor \frac{j-1}{d} \rfloor + 1$ and $\ell_j = j \mod d$. Now, we define a sequence of sub-games $G_{3,j}$ for $1 \leqslant j \leqslant dq_1$. Informally, game $G_{3,j}$ is identical to game $G_{3,j-1}$ except that when computing the response to the k_j-th $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z_{k_j} from $\mathcal{A}$, the challenger $\mathcal{C}_Q$ replaces $f(h_{k_j,\ell_j})$ in game $G_{3,j-1}$ with a random and consistent r_{k_j,ℓ_j} (chosen from an appropriate domain) in game $G_{3,j}$ if h_{k_j,ℓ_j} is the ℓ_j-th RO query required by $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ in computing the response to z_{k_j} and $\ell_j \in I(z_{k_j}) \subseteq \{1, \ldots, d\}$. Clearly, $\mathcal{A}$ can only distinguish game $G_{3,j}$ from $G_{3,j-1}$ by making a QRO query containing h_{k_j,ℓ_j}. Let $\mathsf{Succ}_{3,j}$ be the event that $\mathcal{O}$ outputs 1 in game $G_{3,j}$, where $j \in \{1, \ldots, dq_1\}$. Let L be an initially empty list in the following games, which is used to keep the consistency of the "bad" Type-II RO responses.

Game $G_{3,j}$. Let $k_j = \lfloor \frac{j-1}{d} \rfloor + 1$ and $\ell_j = j \mod d$. This game is almost identical to game $G_{3,j-1}$ except that the challenger $\mathcal{C}_Q$ handles the ℓ_j-th RO query h_{k_j,ℓ_j} required in computing the response u_{k_j} to the k_j-th $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z_{k_j} from $\mathcal{A}$ as follows: if $\ell_j \notin I(z_{k_j})$, set the response to the RO query h_{k_j,ℓ_j} as in game $G_{3,j-1}$. Else, if there does not exist a pair $(h_{k_j,\ell_j}, r) \in L$ for some r, choose a uniformly random r as the response to the RO query h_{k_j,ℓ_j}, and append (h_{k_j,ℓ_j}, r) to the list L. Let $\mathcal{B}_{3,j}$ be an algorithm which interacts with $\mathcal{A}$ as the challenger in game $G_{3,j}$ except that it chooses an integer $k^* \xleftarrow{\$} \{1, \ldots, q_2\}$, runs $\mathcal{A}$ until just after receiving the k^*-th QRO query $|\phi\rangle = |h_1, h_2\rangle$ from $\mathcal{A}$, measures the argument of the query in the computational basis, and outputs the measurement result $\hat{h}$ ($\mathcal{B}_{3,j}$ outputs $\perp$ if $\mathcal{A}$ makes less than k^* RO queries). Let $\delta_{3,j}$ be the probability that $\hat{h} = h_{k_j,\ell_j} \wedge \ell_j \in I(z_{k_j})$, where z_{k_j} is the k_j-th $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query from $\mathcal{A}$, and h_{k_j,ℓ_j} is the ℓ_j-th RO query required for computing $u = Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, z_{k_j})$. Clearly, $\delta_{1,j}$ is only useful for a (F_λ, I)-CPRed with $I(\cdot) \neq \emptyset$.

Lemma 5. If $I(\cdot) = \emptyset$, then $\Pr[\mathsf{Succ}_{3,j-1}] = \Pr[\mathsf{Succ}_{3,j}]$, else we have that $|\Pr[\mathsf{Succ}_{3,j-1}] - \Pr[\mathsf{Succ}_{3,j}]| \leqslant 2q_2\sqrt{\delta_{3,j}}$ holds.

Proof. Note that if $I(\cdot) = \emptyset$, the change from game $G_{3,j-1}$ to game $G_{3,j}$ is conceptual, i.e., we always

have that $\Pr[\mathsf{Succ}_{3,j-1}] = \Pr[\mathsf{Succ}_{3,j}]$. Otherwise, the only difference between $G_{3,j-1}$ and $G_{3,j}$ is that the challenger $\mathcal{C}_Q$ might replace the value $f(h_{k_j,\ell_j})$ in game $G_{3,j-1}$ with a uniformly random r_{k_j,ℓ_j} in game $G_{3,j}$ if z_{k_j} is the k_j-th $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query from $\mathcal{A}$, h_{k_j,ℓ_j} is the ℓ_j-th RO query required for computing $u_{k_j} = Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, z_{k_j})$ and $\ell_j \in I(z_{k_j})$. By using the fact that f is a random function and Lemma 1, we have that $|\Pr[\mathsf{Succ}_{3,j-1}] - \Pr[\mathsf{Succ}_{3,j}]| \leqslant 2q_2\sqrt{\delta_{3,j}}$ holds. This completes the proof.

Game G_4. The challenger $\mathcal{C}_Q$ first computes $(y, s_y) \leftarrow Q.\mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa)$, and $(x, s_x, \mathrm{st}_1 = (\tau, f)) \leftarrow Q.\mathsf{Extract}(y, s_y)$. Let $\mathcal{O}_f(\cdot) : |h_1, h_2\rangle \to |h_1, f(h_1) \oplus h_2\rangle$ and $\mathcal{O}_{F_\lambda}(\cdot) : |h_1, h_2\rangle \to |h_1, F_\lambda(\mathrm{st}_1, h_1) \oplus h_2\rangle$. Then, $\mathcal{C}_Q$ invokes $\mathcal{A}$ with y:

(i) When receiving a QRO query $|\phi\rangle = |h_1, h_2\rangle$ from $\mathcal{A}$, return $\mathcal{O}_{F_\lambda}(|\phi\rangle)$ to $\mathcal{A}$ if $|\phi\rangle$ is a Type-I RO query, other return $\mathcal{O}_f(|\phi\rangle)$ to $\mathcal{A}$ if it is a Type-II RO query;

(ii) When receiving a $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z_i from $\mathcal{A}$, feed z to $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$. If $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts, run a trivial solving algorithm to find a solution v' of y, output $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v')$ and terminate. Else, compute $u = Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, z)$ by using s_y and count the RO queries required by the computation. Whenever the computation stops to make the ℓ-th RO query h for some $\ell \in \{1, \ldots, d\}$, distinguish the following cases:

- If h is a Type-I RO query, use $F_\lambda(\mathrm{st}_1, h)$ to continue the computation.
- Else if h is a Type-II RO query and $\ell \notin I(z)$, use $f(h)$ to continue the computation. Otherwise, if there does not exist a pair $(h, r) \in L$, choose a random r to continue the computation and append (h, r) to the list L.

Finally, return u to $\mathcal{A}$;

(iii) Whenever $\mathcal{A}$ outputs a solution v of y, $\mathcal{C}_Q$ feed v to $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$. If $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts, run a trivial solving algorithm to find a solution v' of y and output $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v')$. Otherwise, output $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v)$.

Let $\mathcal{B}_i$ be an algorithm which interacts with $\mathcal{A}$ as the challenger in game G_i for $i \in \{4, 5, 6\}$ except that it chooses an integer $k^* \xleftarrow{\$} \{1, \ldots, q_2\}$, runs $\mathcal{A}$ until just after receiving the k^*-th QRO query $|\phi\rangle = |h_1, h_2\rangle$ from $\mathcal{A}$, measures the argument of the query and outputs the measurement outcome $\hat{h}$ ($\mathcal{B}_i$ outputs $\perp$ if $\mathcal{A}$ makes less than k^* RO queries). Let δ_i be the probability that $\hat{h} = h_{k_j,\ell_j} \wedge \ell_j \in I(z_{k_j})$ for any $1 \leqslant j \leqslant dq_1$, where z_{k_j} is the k_j-th $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query from $\mathcal{A}$, and h_{k_j,ℓ_j} is the ℓ_j-th RO query required for computing $u_{k_j} = Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}(s_y, z_{k_j})$.

Lemma 6. $\Pr[\mathsf{Succ}_4] = \Pr[\mathsf{Succ}_{3,dq_1}]$. Moreover, for $I(\cdot) \neq \emptyset$, we have $\delta_4 \geqslant \delta_{3,j}$ for any $1 \leqslant j \leqslant dq_1$.

Proof. This first claim follows from the fact that games G_4 and G_{3,dq_1} are identical. Note that if $\mathcal{A}$ does not make a QRO query containing any element in $\{\{h_{k_j,\ell_j}\}_{\ell_j \in I(z_{k_j})}\}_{1 \leqslant j \leqslant dq_1}$, the view of $\mathcal{A}$ in games $G_{3,j}$ for $0 \leqslant j \leqslant dq_1$ is identically distributed. Moreover, before $\mathcal{A}$ making a QRO query containing one of the elements in $\{\{h_{k_j,\ell_j}\}_{\ell_j \in I(z_{k_j})}\}_{1 \leqslant j \leqslant dq_1}$, we always have that the view of $\mathcal{A}$ in games $G_{3,j}$ for $0 \leqslant j \leqslant dq_1$ is identically distributed, which implies that $\delta_4 \geqslant \delta_{3,j}$ for any $1 \leqslant j \leqslant dq_1$. This completes the proof.

Game G_5. This game is almost identical to game G_4 except that $\mathcal{C}_Q$ generates $(x, s_x, y, \mathrm{st}_1)$ by running $(x, s_x) \leftarrow P.\mathsf{IGen}(1^\kappa), (y, \mathrm{st}_1) \leftarrow \mathcal{S}_1^{P.\mathsf{Orcl}(s_x,\cdot)}(x)$, and handles the $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query by directly using $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$. Whenever $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ aborts, $\mathcal{C}_Q$ first runs a trivial solving algorithm to find a solution v' of y, then outputs $Q.\mathsf{Vrfy}^{\mathcal{O}(\cdot)}(y, s_y, v')$ and terminates.

Lemma 7. $|\Pr[\mathsf{Succ}_5] - \Pr[\mathsf{Succ}_4]| \leqslant \mathsf{negl}(\kappa)$. Moreover, for $I(\cdot) \neq \emptyset$, we have $|\delta_5 - \delta_4| \leqslant \mathsf{negl}(\kappa)$.

Proof. There are only two differences between game G_5 and game G_6: the way of generating $(x, s_x, y, \mathrm{st}_1)$ and the way of answering the $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ queries. First, by the assumption on $Q.\mathsf{Extract}$, we have that the distribution of $(x, s_x, y, \mathrm{st}_1)$ generated by using $(x, s_x) \leftarrow P.\mathsf{IGen}(1^\kappa)$ and $(y, \mathrm{st}_1) \leftarrow \mathcal{S}_1^{P.\mathsf{Orcl}(s_x,\cdot)}(1^\kappa, x)$ in game G_5 is identical to that generated by using $(y, s_y) \leftarrow Q.\mathsf{IGen}^{\mathcal{O}(\cdot)}(1^\kappa)$ and $(x, s_x, \mathrm{st}_1) \leftarrow Q.\mathsf{Extract}(y, s_y)$ in game G_4. Second, for each $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z, by the definition of game G_4, except at the RO queries with positions specified by $I(z)$, the behavior of generating the response to z in game G_4 is identical to that of the real $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ with the RO $\mathcal{O}(\cdot)$ being changed to the simulated RO by $\mathcal{S}_3$. Moreover, by the third condition in Definition 3 that for each $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ query z, except at the RO queries with positions specified by $I(z)$, the behavior of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y, \mathrm{st}_1)$ in game G_5 is statistically close to that of the real $Q.\mathsf{Orcl}^{\mathcal{O}(\cdot)}$ with the RO $\mathcal{O}(\cdot)$ being changed to the simulated RO by $\mathcal{S}_3$. This means that except at the

RO queries with positions specified by $I(z)$, the behavior of generating the response to z in game G_4 is statistically close to that of $\mathcal{S}_2^{P.\mathsf{Orcl}(s_x,\cdot)}(y,\mathrm{st}_1)$ in game G_5. As the behaviors of both games G_4 and G_5 are identical at the RO queries with positions specified by $I(z)$, we have that game G_5 and G_4 are statistically indistinguishable. This completes the proof.

It is easy to check that the view of $\mathcal{A}$ in game G_5 is identical to the one simulated by $\mathcal{S}'$. Thus, it suffices to analyze the behavior of $\mathcal{A}$ in game G_5. First, if $I(\cdot)=\emptyset$, by Lemmas 2–7 we have $\Pr[\mathsf{Succ}_5] \geqslant t(\kappa)+\vartheta_A^2/\mathsf{poly}_3(q_1,q_3)+\mathsf{negl}(\kappa)$. This means that $\mathcal{S}'$ can find a solution v of y with advantage at least $\vartheta_A^2/\mathsf{poly}_3(q_1,q_3)$ from $\mathcal{A}$. By computing $w \leftarrow \mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}(\mathrm{st}_1,v)$, we have that $\mathcal{S}'$ can output a solution w of x with advantage $\vartheta_A^2/\mathsf{poly}_3(q_1,q_3)$ by Definition 3.

Second, if $I(\cdot)\neq\emptyset$, by Definition 3 the advantage that $\mathcal{A}$ outputs a valid solution v of y in game G_5 is negligible (i.e., $\Pr[\mathsf{Succ}_5]=\mathsf{negl}(\kappa)$). By Lemmas 2–7 we have $t(\kappa)+\vartheta_A^2/\mathsf{poly}_3(q_1,q_3)+\mathsf{negl}(\kappa) \leqslant \Pr[\mathsf{Succ}_2] \leqslant \Pr[\mathsf{Succ}_5]+2q_2\sum_{j=1}^{dq_1}\sqrt{\delta_{3,j}}+\mathsf{negl}(\kappa) \leqslant t(\kappa)+2dq_1q_2\sqrt{\delta_5}+\mathsf{negl}(\kappa)$. Thus, we have that

$$\vartheta_A^2/\mathsf{poly}_3(q_1,q_3) \leqslant 2dq_1q_2\sqrt{\delta_5}+\mathsf{negl}(\kappa).$$

Thus, we have that δ_5 is non-negligible if ϑ_A is non-negligible. Note that $\mathcal{S}'$ will compute $w \leftarrow \mathcal{S}_4^{P.\mathsf{Orcl}(s_x,\cdot)}(\mathrm{st}_1,\hat{h})$ by first randomly measuring one of the QRO queries from $\mathcal{A}$ to obtain $\hat{h}$. The probability that $\hat{h}$ is a "bad" RO query is equal to δ_5 by definition, which is non-negligible. Thus, we have that $\mathcal{S}'$ can output a solution w of x with a non-negligible advantage by Definition 3. This completes the proof.

5 Incomparability of CPReds with NPReds and RPReds

By restricting the programmability, Fischlin et al. [5] formalized three notions of BB reductions in the ROM: FPRed, RPRed, and NPRed. Briefly, the FPRed formalizes the standard conception of BB reductions in the ROM, which have full control over the RO, while the NPRed considers the setting where all parties are given access to an external RO (i.e., non-programmable RO [6]) that is not controlled by any party; however, the reduction is allowed to learn all the RO queries made by $\mathcal{A}$. The RPRed can only program the RO via an interface not fully controlled by itself and is formalized via the notion of randomly programmable RO. Formally, a RPRO $\mathcal{O}=(R_{\mathrm{eval}},R_{\mathrm{rand}},R_{\mathrm{prog}})$ is an idealized object consisting of a public interface R_{eval}, and two private interfaces R_{rand} and R_{prog}, where R_{eval} behaves as a conventional RO mapping $\mathrm{Dom}\rightarrow\mathrm{Rng}$; R_{rand} maps a string in $\{0,1\}^*$ to a random value in Rng; and $R_{\mathrm{prog}}(X,Y)$ takes an $X\in\mathrm{Dom}$ and a string $Y\in\{0,1\}^*$ as inputs, sets $R_{\mathrm{eval}}(X)=R_{\mathrm{rand}}(Y)$. As NPReds, an RPRed (i.e., a BB reduction in the model equipped with an RPRO) is allowed to learn all the RO queries made by the adversary $\mathcal{A}$ via the public interface R_{eval}, but unlike NPReds, an RPRed is also allowed to program the answer of a given RO query X made by the adversary $\mathcal{A}$ via the two private interfaces R_{rand} and R_{prog}. Specifically, after receiving an RO query X from $\mathcal{A}$, the reduction can make several queries to R_{rand} and R_{prog} before returning the answer $R_{\mathrm{eval}}(X)$ (which can be programmed using the interface R_{prog}) to $\mathcal{A}$. Note that an RPRed cannot directly set the RO responses with any values of its own choices.

In the following, we investigate the relation of CPReds to the notions of BB reductions formalized by Fischlin et al. [5], i.e., FPRed, RPRed, and NPRed. As shown in [5], an NPRed implies an RPRed (i.e., NPRed $\Rightarrow$ RPRed), which in turn implies an FPRed by definition (i.e., RPRed $\Rightarrow$ FPRed). Since FPRed essentially formalizes the standard concept of all BB reductions in the ROM, we immediately have that a CPRed implies an FPRed (i.e., CPRed $\Rightarrow$ FPRed). By carefully examining the OAEP encryption [2] and the FDH signature [3], we show that NPRed $\not\Rightarrow$ CPRed in Subsection 5.1 and CPRed $\not\Rightarrow$ RPRed in Subsection 5.2. Combining our results with the fact that NPRed $\Rightarrow$ RPRed, we immediately have that CPRed $\not\Rightarrow$ NPRed and RPRed $\not\Rightarrow$ CPRed (otherwise we have CPRed $\Rightarrow$ NPRed $\Rightarrow$ RPRed), implying that CPReds are incomparable to both NPReds and RPReds.

We first recall the definition of TDPs. A TDP family $\Pi_{\mathrm{TDP}}=(\mathsf{Gen},\mathsf{Eval},\mathsf{Inv})$ with respect to length function $\ell(\cdot)$ consists of three algorithms. The parameter generator $\mathsf{Gen}(1^\kappa)$ takes a security parameter κ as input, outputs an index-trapdoor pair (s,td), i.e., $(s,\mathrm{td})\leftarrow\mathsf{Gen}(1^\kappa)$. The evaluation algorithm $\mathsf{Eval}(s,x)$ takes an index s and a string $x\in\{0,1\}^{\ell(\kappa)}$ as inputs, outputs a string $y\in\{0,1\}^{\ell(\kappa)}$, i.e., $y\leftarrow\mathsf{Eval}(s,x)$. The inversion algorithm $\mathsf{Inv}(\mathrm{td},y)$ takes a trapdoor td and a string $y\in\{0,1\}^{\ell(\kappa)}$ as inputs, outputs a string $x\in\{0,1\}^{\ell(\kappa)}$, i.e., $x\leftarrow\mathsf{Inv}(\mathrm{td},y)$.

For correctness, we require that for all $(s,\mathrm{td}) \leftarrow \mathsf{Gen}(1^\kappa)$ and all $x \in \{0,1\}^{\ell(\kappa)}$, the equation $\mathsf{Inv}(\mathrm{td}, \mathsf{Eval}(s,x)) = x$ always holds. Moreover, we say that $\Pi_{\mathrm{TDP}} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$ is one-way if for all probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$, we have that

$$\Pr[(s,\mathrm{td}) \leftarrow \mathsf{Gen}(1^\kappa), x \xleftarrow{\$} \{0,1\}^{\ell(\kappa)}, x' \leftarrow \mathcal{A}(s, \mathsf{Eval}(s,x)) : x' = x] = \mathsf{negl}(\kappa).$$

A TDP $\Pi_{\mathrm{TDP}} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$ is said to be partial-domain one-way with respect to $(\ell(\cdot), \ell_0(\cdot))$ if for all PPT algorithm $\mathcal{A}$, we have

$$\Pr\left[\begin{array}{r}(s,\mathrm{td}) \leftarrow \mathsf{Gen}(1^\kappa),\ x_0 \xleftarrow{\$} \{0,1\}^{\ell_0(\kappa)}, x_1 \xleftarrow{\$} \{0,1\}^{\ell(\kappa)-\ell_0(\kappa)} \\ x_0' \leftarrow \mathcal{A}(s, \mathsf{Eval}(s, x_0\|x_1)) : x_0' = x_0\end{array}\right] = \mathsf{negl}(\kappa).$$

5.1 NPRed $\not\Rightarrow$ CPRed

In this subsection, we show that the OAEP encryption in [2], which was provably secure under NPReds, is not provable under CPReds. Formally, let κ be the security parameter. Let $n_1, n_2, n_3 \geqslant \kappa$ be any positive integers. Let $\ell(\cdot)$ and $\ell_0(\cdot)$ be two arbitrary integer functions such that $\ell(\kappa) = \ell_0(\kappa) + n_3 = n_1 + n_2 + n_3$. Let $\Pi_{\mathrm{TDP}} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$ be a trapdoor partial-domain one-way permutation family with respect to functions $(\ell(\cdot), \ell_0(\cdot))$. Let $G : \{0,1\}^{n_3} \to \{0,1\}^{n_1+n_2}$ and $H : \{0,1\}^{n_1+n_2} \to \{0,1\}^{n_3}$ be two hash functions. The PKE scheme $\Pi_{\mathrm{OAEP}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ given in [2] works as follows:

- $\mathsf{KeyGen}(1^\kappa)$. Take a security parameter κ as input and compute and output the public and secret key pair $(\mathrm{pk}, \mathrm{sk}) = (s, \mathrm{td}) \leftarrow \Pi_{\mathrm{TDP}}.\mathsf{Gen}(1^\kappa)$.
- $\mathsf{Enc}(\mathrm{pk}, \mu)$. Take the public key $\mathrm{pk} = s$ and a message $\mu \in \{0,1\}^{n_1}$ as inputs and first append n_2 zeros to μ and obtain $\hat{\mu} = \mu\|0_{n_2} \in \{0,1\}^{n_1+n_2}$. Then uniformly choose $r \xleftarrow{\$} \{0,1\}^{n_3}$, compute $x_0 = \hat{\mu} \oplus G(r)$, $x_1 = r \oplus H(x_0)$ and $y = \Pi_{\mathrm{TDP}}.\mathsf{Eval}(s, x_0\|x_1)$. Finally, output the ciphertext $c = y$.
- $\mathsf{Dec}(\mathrm{sk}, c)$. Take the secret key $\mathrm{sk} = \mathrm{td}$ and a ciphertext $c = y \in \{0,1\}^{\ell(\kappa)}$ as inputs, first compute $x_0\|x_1 = \Pi_{\mathrm{TDP}}.\mathsf{Inv}(\mathrm{td}, y)$, $r = x_1 \oplus H(x_0)$ and $\hat{\mu} = x_0 \oplus G(r)$, where $x_0, \hat{\mu} \in \{0,1\}^{n_1+n_2}$ and $x_1 \in \{0,1\}^{n_3}$. Then, parse $\hat{\mu} = \mu\|\mu_0$ into $\mu \in \{0,1\}^{n_1}$ and $\mu_0 \in \{0,1\}^{n_2}$. If $\mu_0 \neq 0_{n_2}$, return $\perp$. Else return μ.

We have Lemma 8 implicit in [5, 21].

Lemma 8. Let $n_1, n_2, n_3 \geqslant \kappa$ be any positive integers. Let $(\ell(\cdot), \ell_0(\cdot))$ be two arbitrary integer functions such that $\ell(\kappa) = \ell_0(\kappa) + n_3 = n_1 + n_2 + n_3$. If the TDP family Π_{TDP} is partial-domain one-way with respect to $(\ell(\cdot), \ell_0(\cdot))$, the PKE scheme Π_{OAEP} is provably CCA-secure under NPReds.

Notably, in [5], the reduction for the PKE scheme Π_{OAEP} given in [21] does not need to program the RO, and thus is an NPRed. Specifically, assuming that the TDP family $\Pi_{\mathrm{TDP}} = (\mathsf{Gen}, , \mathsf{Eval}, \mathsf{Inv})$ is partial-domain one-way, Π_{PKE} is provably CCA-secure under NPReds (Appendix C for the CCA-security of PKE). We note that the reduction in [21] crucially relies on the knowledge of all RO query/response pairs made by the adversary to answer the decryption queries, which is not allowed for a CPRed (since the simulation of the decryption oracle is oblivious to the RO queries made by the adversary). We can show that the scheme Π_{OAEP} is not provable under CPReds.

Theorem 2. Let $n_1, n_2, n_3 \geqslant \kappa$ be any positive integers. Let $(\ell(\cdot), \ell_0(\cdot))$ be any two integer functions such that $\ell(\kappa) = \ell_0(\kappa) + n_3 = n_1 + n_2 + n_3$. There is no CPRed from the CCA-security of Π_{OAEP} to the partial-domain one-wayness of the TDP family Π_{TDP} with respect to $(\ell(\cdot), \ell_0(\cdot))$.

We first give a sketch of the proof. Technically, we use the two-oracle separation technique of Hsiao and Reyzin [25] by giving two oracles Λ and Ω such that (1) Λ is an ideal TDP and Ω is a "breaking" oracle; (2) there is a PPT adversary $\mathcal{A}^{\Lambda,\Omega}$ which could break the CCA-security of the instantiation $\Pi_{\mathrm{OAEP}}^{\Lambda}$ of Π_{OAEP} using Λ; (3) no PPT CPRed $\mathcal{S}$, given access to Λ and $\mathcal{A}^{\Lambda,\Omega}$ (i.e., $\mathcal{S}^{\Lambda,\mathcal{A}^{\Lambda,\Omega}}$), could break the partial-domain one-wayness of Λ. The basic idea is that $\mathcal{S}$ can only access the "breaking" oracle Ω by interacting with $\mathcal{A}$ which will only use Ω after being convinced that Ω is "useless" to $\mathcal{S}$ (which requires $\mathcal{S}$ to invert the permutation and correctly responds to a random decryption query without "seeing" the RO queries). In other words, the ability of $\mathcal{A}$ to access the breaking oracle Ω cannot be utilized by $\mathcal{S}$ to break the partial-domain one-wayness of Λ.

Proof. Let $\mathcal{E} = (\mathcal{E}_{\mathsf{Perm}}, \mathcal{E}_E, \mathcal{E}_{E^{-1}})$ be an oracle, which is initialized with a random permutation $\mathsf{Perm} : \{0,1\}^\kappa \to \{0,1\}^\kappa$ and a keyed family of random permutations $E : \{0,1\}^\kappa \times \{0,1\}^{\ell(\kappa)} \to \{0,1\}^{\ell(\kappa)}$ (i.e., given a key $s \in \{0,1\}^\kappa$, $E(s,\cdot) : \{0,1\}^{\ell(\kappa)} \to \{0,1\}^{\ell(\kappa)}$ is a random permutation). The two interfaces $\mathcal{E}_{\mathsf{Perm}}$ and $\mathcal{E}_E$ allow to evaluate Perm and E, respectively, whereas the interface $\mathcal{E}_{E^{-1}}$ allows to evaluate

the inversion E^{-1} of E, i.e., $E^{-1}(s, E(s,x)) = x$. Now, we define the first oracle $\Lambda = (\mathsf{Gen}^{\mathcal{E}}, \mathsf{Eval}^{\mathcal{E}}, \mathsf{Inv}^{\mathcal{E}})$ consisting of three interfaces:

- $\mathsf{Gen}^{\mathcal{E}}(1^\kappa)$. First randomly choose $\mathrm{td} \xleftarrow{\$} \{0,1\}^\kappa$, then send td to $\mathcal{E}_{\mathsf{Perm}}$ and obtain a response $s = \mathsf{Perm}(\mathrm{td})$. Finally, return (s, td), i.e., $(s, \mathrm{td}) \leftarrow \mathsf{Gen}^{\mathcal{E}}(1^\kappa)$.
- $\mathsf{Eval}^{\mathcal{E}}(s,x)$. First send $(s,x) \in \{0,1\}^\kappa \times \{0,1\}^{\ell(\kappa)}$ to $\mathcal{E}_E$ and obtain a response $y = E(s,x)$. Then, return $y \in \{0,1\}^{\ell(\kappa)}$, i.e., $y \leftarrow \mathsf{Eval}^{\mathcal{E}}(s,x)$.
- $\mathsf{Inv}^{\mathcal{E}}(\mathrm{td}, y)$. First send $\mathrm{td} \in \{0,1\}^\kappa$ to $\mathcal{E}_{\mathsf{Perm}}$ and obtain a response $s = \mathsf{Perm}(\mathrm{td})$. Then, send $(s,y) \in \{0,1\}^\kappa \times \{0,1\}^{\ell(\kappa)}$ to $\mathcal{E}_{E^{-1}}$ and obtain a response $x = E^{-1}(s,y)$. Finally, return $x \in \{0,1\}^{\ell(\kappa)}$, i.e., $x \leftarrow \mathsf{Inv}^{\mathcal{E}}(\mathrm{td}, y)$.

We first show that Λ is a family of ideal TDPs, i.e., no PPT algorithm $\mathcal{S}^\Lambda$ given oracle access to Λ can break the partial-domain one-wayness of Λ with respect to functions $\ell(\kappa) \geqslant \ell_0(\kappa) \geqslant \kappa$. Notably, given a challenge pair $s^* = \mathsf{Perm}(\mathrm{td}^*)$ and $y^* = E(s^*, x^*)$ as inputs, $\mathcal{S}^\Lambda$ is asked to output the first $\ell_0(\kappa)$-bit of x^* by making at most a polynomial number of queries to Λ. First, since Perm is a random permutation and $\mathrm{td}^* \in \{0,1\}^\kappa$ is uniformly chosen at random, the probability that $\mathcal{S}^\Lambda$ can find td^* is negligible, implying that the inversion oracle $\Lambda.\mathsf{Inv}^{\mathcal{E}}$ cannot help $\mathcal{S}^\Lambda$ to invert $E(s^*, \cdot)$. Second, since $E(s^*, \cdot)$ is a random permutation, conditioned on that $\mathcal{S}^\Lambda$ cannot find td^*, the probability that $\mathcal{S}^\Lambda$ can output the first $\ell_0(\kappa)$-bit of x^* is also negligible (because it can only determine the first $\ell_0(\kappa)$-bit of x^* by making a query (s^*, x^*) to $\Lambda.\mathsf{Eval}^{\mathcal{E}}(\cdot,\cdot)$). This shows that Λ is partial-domain one-way.

The second oracle $\Omega = (\mathcal{R}_1, \mathcal{R}_2, \mathsf{Inv}^{\mathcal{E}})$ also consists of three interfaces, where $\mathcal{R}_1(\cdot)$ evaluates a random function from $\{0,1\}^{2^\kappa}$ to $\{0,1\}^{n_1}$, $\mathcal{R}_2(\cdot)$ evaluates a random function from $\{0,1\}^{2^\kappa}$ to $\{0,1\}^{n_3}$, and $\mathsf{Inv}^{\mathcal{E}}(\cdot,\cdot)$ simply relays its query to the oracle $\mathcal{E}$ via the interface $\mathcal{E}_{E^{-1}}$ and returns whatever it obtains from the oracle. Now, we give an adversary $\mathcal{A}^{\Lambda,\Omega}$ breaking the CCA-security of the Π^Λ_{PKE}, but no CPRed $\mathcal{S}^{\Lambda, \mathcal{A}^{\Lambda,\Omega}}$ can break the partial-domain one-wayness of Λ.

Description of $\mathcal{A}^{\Lambda,\Omega}$. Given a security parameter κ and a public key $\mathrm{pk} = s$ of $\Pi^\Lambda_{\mathrm{OAEP}}$ as inputs, the goal of $\mathcal{A}^{\Lambda,\Omega}$ is to break the CCA-security of $\Pi^\Lambda_{\mathrm{OAEP}}$. The adversary $\mathcal{A}^{\Lambda,\Omega}$ which is also given access to a G-oracle and H-oracle used by the scheme $\Pi^\Lambda_{\mathrm{OAEP}}$ works as follows:

(1) Let $V = \kappa \| s$ (i.e., the initial view of $\mathcal{A}^{\Lambda,\Omega}$);

(2) Send $r = \Omega.\mathcal{R}_2(V)$ to the G-oracle and obtain a response z_1;

(3) Update $V := V \| z_1$, and compute $\mu = \Omega.\mathcal{R}_1(V) \in \{0,1\}^{n_1}$;

(4) Send $x_0 = (\mu \| 0_{n_2}) \oplus z_1$ to the H-oracle and obtain a response z_2;

(5) Send $y = \Lambda.\mathsf{Eval}^{\mathcal{E}}(s, x_0 \| x_1))$ to the decryption oracle and obtain a result μ', where $x_1 = r \oplus z_2$;

(6) If $\mu \neq \mu'$, output $\perp$ and abort;

(7) Update $V := V \| z_2 \| \mu'$, compute $\mu^*_0 = \Omega.\mathcal{R}_1(V\|0), \mu^*_1 = \Omega.\mathcal{R}_1(V\|1)$, and send (μ^*_0, μ^*_1) to obtain a challenge ciphertext y^*;

(8) Compute $x^* = x^*_0 \| x^*_1 = \Omega.\mathsf{Inv}^{\mathcal{E}}(s, y^*)$, send $x^*_0 \in \{0,1\}^{n_1+n_2}$ to the H-oracle and obtain a response z^*_2;

(9) Send $r^* = x^*_1 \oplus z^*_2 \in \{0,1\}^{n_3}$ to the G-oracle and obtain a response z^*_1;

(10) If $z^*_1 \oplus x^*_0 = \mu^*_0 \| 0_{n_2}$, output 0, else output 1.

Notably, no matter how the oracles G and H are instantiated, $\mathcal{A}^{\Lambda,\Omega}$ can always break the CCA-security of $\Pi^\Lambda_{\mathrm{OAEP}}$. Let (s^*, y^*) be the challenge pair of the partial-domain one-wayness of Λ. We now show that no CPRed $\mathcal{S}(s^*, y^*)$ given oracle access to Λ and $\mathcal{A}^{\Lambda,\Omega}$ can find the first $\ell_0(\kappa)$-bit of $x^* = E^{-1}(s^*, y^*)$ with non-negligible probability. The basic idea is that $\mathcal{A}^{\Lambda,\Omega}$ will not use the inversion oracle $E^{-1}(s^*, \cdot)$ until it has been convinced that $\mathcal{S}^\Lambda$ can invert the function $E(s^*, \cdot)$, which is infeasible for $\mathcal{S}^\Lambda$ given only oracle access to Λ.

By Definition 3, a CPRed $\mathcal{S}^\Lambda = (\mathcal{S}^\Lambda_1, \mathcal{S}^\Lambda_2, \mathcal{S}^\Lambda_3, \mathcal{S}^\Lambda_4)$ is a single-instance reduction and will not rewind the adversary $\mathcal{A}^{\Lambda,\Omega}$. Moreover, given a security parameter κ and a challenge pair (s^*, y^*) as inputs, $\mathcal{S}^\Lambda = (\mathcal{S}^\Lambda_1, \mathcal{S}^\Lambda_2, \mathcal{S}^\Lambda_3, \mathcal{S}^\Lambda_4)$ works as follows[5]: First, it runs $(\mathrm{pk}, \mathrm{st}_1) \leftarrow \mathcal{S}^\Lambda_1(1^\kappa, (s^*, y^*))$ to obtain a public key pk and a state st_1. Second, it runs $\mathrm{st}_2 \leftarrow \mathcal{S}^\Lambda_2(\mathrm{pk}, \mathrm{st}_1)$ to obtain a state st_2 by invoking the adversary $\mathcal{A}^{\Lambda,\Omega}$ and answering the decryption queries from $\mathcal{A}^{\Lambda,\Omega}$. Third, it runs $\mathrm{st}_3 \leftarrow \mathcal{S}^\Lambda_3(\mathrm{st}_1)$ to simulate the ROs for G, H and obtain a state st_3 which is either an empty string ϵ or one of the oracle queries to G and H. Finally, it runs $\mathcal{S}^\Lambda_4(\mathrm{st}_1, \mathrm{st}_2, \mathrm{st}_3)$ to obtain a candidate solution x'.

5) Here, we allow the subalgorithm $\mathcal{S}_3$ to access the oracle Λ, which does not conflict with the restriction that "$\mathcal{S}_3$ does not have access to $P.\mathsf{Oral}(s_x, \cdot)$" in Definition 2 since Λ is an oracle which implements the TDPs and is assumed to be publicly known to all parties, while $P.\mathsf{Oral}(s_x, \cdot)$ is assumed to be a private oracle that can only be accessed by the party who owns the secret value s_x. The problem we considered here is to invert the TDPs (i.e., the partial-domain one-wayness of Λ), which is a noninteractive problem (Section 5 for the definition), i.e., the corresponding oracle "$P.\mathsf{Oral}(s_x, \cdot)$" is a dummy one $P.\mathsf{Oral}(s_x, \cdot) = \perp$.

If $\mathcal{A}^{\Lambda,\Omega}$ does not use s^* as the first input in a query to $\Omega.\mathsf{Inv}^{\mathcal{E}}(\cdot,\cdot)$ (and thus does not use s^* as the first input to the inversion oracle $E^{-1}(\cdot,\cdot)$), then $\mathcal{S}^{\Lambda}$ cannot obtain any advantage from $\mathcal{A}^{\Lambda,\Omega}$ to invert $y^* = E(s^*, x^*)$. This is because for any $s \neq s^*$, $E(s^*,\cdot)$ is a random permutation independent of $E(s,\cdot)$. Since $\mathcal{A}^{\Lambda,\Omega}$ will only make a single query using $\mathrm{pk} = s$ as the first input to $\Omega.\mathsf{Inv}^{\mathcal{E}}(\cdot,\cdot)$ after $\mathcal{S}_2^{\Lambda}$ correctly answers a random decryption query, it suffices to show that $\mathcal{S}_2^{\Lambda}$ cannot correctly answer the decryption query from $\mathcal{A}^{\Lambda,\Omega}$ with non-negligible probability when $s = s^*$.

Recalling that given a security parameter κ and a public key $\mathrm{pk} = s^*$ as inputs, $\mathcal{A}^{\Lambda,\Omega}$ works as follows: (1) set $V = \kappa \| s^*$ and compute $r = \Omega.\mathcal{R}_2(V)$; (2) send r as a G-oracle query to $\mathcal{S}_3^{\Lambda}(\mathrm{st}_1)$ and obtain a response z_1; (3) update $V := V \| z_1$, and compute $\mu = \Omega.\mathcal{R}_1(V) \in \{0,1\}^{n_1}$; (4) send $x_0 = (\mu \| 0_{n_2}) \oplus z_1$ as an H-oracle query to $\mathcal{S}_3^{\Lambda}(\mathrm{st}_1)$ and obtain a response z_2; (5) send $y = \Lambda.\mathsf{Eval}^{\mathcal{E}}(s, x_0 \| x_1))$ as a decryption query to $\mathcal{S}_2^{\Lambda}(\mathrm{pk}, \mathrm{st}_1)$ and obtain a result μ', where $x_1 = r \oplus z_2$. We now show that the probability that $\mu' = \mu$ is negligible. First, by the fact that $\Omega.\mathcal{R}_1$ and $\Omega.\mathcal{R}_2$ evaluates random functions, we have that r and μ are both uniformly random and, in particular, independent from the inputs (st_1, y) of $\mathcal{S}_2^{\Lambda}$. Second, since $\mathcal{S}_2^{\Lambda}$ is unaware of the RO queries made to G and H when answering the decryption query $y = \Lambda.\mathsf{Eval}^{\mathcal{E}}(s, x_0 \| x_1)$, and

$$x_0 \| x_1 = ((\mu \| 0_{n_2}) \oplus z_1) \| (r \oplus z_2) = ((\mu \| 0_{n_2}) \oplus G(r)) \| (r \oplus H(x_0))$$

has min-entropy at least $2^{n_1} \geqslant 2^{\kappa}$ in the view of $\mathcal{S}_2^{\Lambda}$ (since μ is uniformly distributed over $\{0,1\}^{n_1}$), the probability that $\mathcal{S}_2^{\Lambda}$ can output $\mu' = \mu$ by making a polynomial number of queries to Λ is negligible (because $\mathcal{S}_2^{\Lambda}$ can only determine μ by making a query $(s^*, x_0 \| x_1)$ to $\Lambda.\mathsf{Eval}^{\mathcal{E}}(\cdot,\cdot)$). In other words, $\mathcal{A}^{\Lambda,\Omega}$ will abort at step 6 with probability negligibly close to 1. Thus, $\mathcal{S}^{\Lambda}$ can only obtain negligible advantage from $\mathcal{A}^{\Lambda,\Omega}$ in inverting $y^* = E(s^*, x^*)$, which completes the proof of Theorem 2.

Combining the results in Lemma 8 and Theorem 2, we have that NPReds does not imply CPReds, namely, NPReds $\not\Rightarrow$CPReds.

5.2 CPRed $\not\Rightarrow$ RPRed

Let κ be the security parameter, and ℓ_μ be a positive integer. Let $\ell(\cdot)$ be an integer function. Let $\Pi_{\mathrm{TDP}} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$ be a trapdoor one-way permutation family with respect to $\ell(\cdot)$. Let $H : \{0,1\}^{\ell_\mu} \to \{0,1\}^{\ell(\kappa)}$ be a hash function. The signature scheme $\Pi_{\mathrm{FDH}} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ given in [3] works as follows:

- $\mathsf{KeyGen}(1^\kappa)$. Take a security parameter κ as input and compute and return the verification and signing key pair $(\mathrm{vk}, \mathrm{sk}) = (s, \mathrm{td}) \leftarrow \Pi_{\mathrm{TDP}}.\mathsf{Gen}(1^\kappa)$.
- $\mathsf{Sign}(\mathrm{sk}, \mu)$. Take the signing key $\mathrm{sk} = \mathrm{td}$ and a message $\mu \in \{0,1\}^{\ell_\mu}$ as inputs and compute $y = H(\mu)$ and return the signature $\sigma = x = \Pi_{\mathrm{TDP}}.\mathsf{Inv}(\mathrm{td}, y)$.
- $\mathsf{Verify}(\mathrm{vk}, \mu, \sigma)$. Take the verification key $\mathrm{vk} = s$, a message $\mu \in \{0,1\}^{\ell_\mu}$ and a signature $\sigma = x \in \{0,1\}^{\ell(\kappa)}$ as inputs, first compute $y = H(\mu) \in \{0,1\}^{\ell(\kappa)}$. Then, return 1 if $y = \Pi_{\mathrm{TDP}}.\mathsf{Eval}(s, x)$, else return 0.

As shown in Subsection 3.2, the FDH signature from TDP is provable under CPReds. Formally, we have Theorem 3.

Theorem 3. If the TDP family Π_{TDP} is one-way, the signature scheme Π_{FDH} is provably EUF-CMA secure under CPReds.

Besides, we also have Lemma 9 implicit in [5].

Lemma 9. There is no RPRed from the EUF-CMA security of the signature scheme Π_{FDH} to the one-wayness of the TDP family Π_{TDP}.

This lemma directly follows from two facts in [5]: (1) An RPRed implies a reduction in the weakly programming ROM (WPROM); and (2) Π_{FDH} is not provable even against key-only attacks in the WPROM. We omit the details.

Combining the results in Theorem 3 and Lemma 9, we have that CPReds does not imply RPReds, namely, CPReds $\not\Rightarrow$RPReds.

6 Conclusion

We introduce the notion of CPRed, aiming to capture a class of BB reductions in the ROM. We first demonstrate the nontriviality of CPRed by showing that some well-known schemes, such as the FDH

signature and the BF-IBE, are provably secure under CPReds. Then, we showed that a CPRed associated with an instance-extraction algorithm implies a reduction in the QROM, which explicitly connects reductions in the ROM and QROM. Finally, we show that CPRed is incomparable to NPRed and RPRed known in the literature. One interesting problem is to give a more powerful notion of reductions that applies to all schemes already known to be provably secure in both ROM and QROM.

Acknowledgements Jiang ZHANG was supported by National Natural Science Foundation of China (Grant Nos. 62022018, 61932019) and National Key Research and Development Program of China (Grant No. 2022YFB2702000). Yu YU was supported by National Natural Science Foundation of China (Grant Nos. 62125204, 92270201), National Key Research and Development Program of China (Grant No. 2018YFA0704701), and Major Program of Guangdong Basic and Applied Research (Grant No. 2019B030302008). Yu YU also acknowledges the support from the XPLORER PRIZE.

References

1 Bellare M, Rogaway P. Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security, 1993. 62–73
2 Bellare M, Rogaway P. Optimal asymmetric encryption. In: Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, 1995. 92–111
3 Bellare M, Rogaway P. The exact security of digital signatures — how to sign with RSA and Rabin. In: Proceedings of International Conference on the Theory and Application of Cryptographic Techniques Saragossa, 1996. 399–416
4 Ananth P, Bhaskar R. Non observability in the random oracle model. In: Proceedings of the 7th International Conference on Provable Security, 2013. 86–103
5 Fischlin M, Lehmann A, Ristenpart T, et al. Random oracles with(out) programmability. In: Proceedings of the 16th International Conference on the Theory and Application of Cryptology and Information Security, 2010. 303–320
6 Nielsen J B. Separating random oracle proofs from complexity theoretic proofs: the non-committing encryption case. In: Proceedings of the 22nd Annual International Cryptology Conference, 2002. 111–126
7 Canetti R, Goldreich O, Halevi S. The random oracle methodology, revisited. J ACM, 2004, 51: 557–594
8 Maurer U, Renner R, Holenstein C. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Proceedings of the 1st Theory of Cryptography Conference, 2004. 21–39
9 Boneh D, Dagdelen Ö, Fischlin M, et al. Random oracles in a quantum world. In: Proceedings of the 17th International Conference on the Theory and Application of Cryptology and Information Security, 2011
10 Gentry C, Peikert C, Vaikuntanathan V. Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, 2008. 197–206
11 Zhandry M. Secure identity-based encryption in the quantum random oracle model. In: Proceedings of Annual Cryptology Conference, 2012. 758–775
12 Dagdelen Ö, Fischlin M, Gagliardoni T. The Fiat-Shamir transformation in a quantum world. In: Proceedings of International Conference on the Theory and Application of Cryptology and Information Security, 2013. 62–81
13 Targhi E E, Unruh D. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In: Proceedings of Theory of Cryptography Conference, 2016. 192–216
14 Hofheinz D, Hövelmanns K, Kiltz E. A modular analysis of the Fujisaki-Okamoto transformation. In: Proceedings of Theory of Cryptography Conference, 2017. 341–371
15 Kiltz E, Lyubashevsky V, Schaffner C. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In: Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2018. 552–586
16 Jiang H, Zhang Z, Chen L, et al. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisted. In: Proceedings of Annual International Cryptology Conference, 2018. 96–125
17 Zhandry M. How to record quantum queries, and applications to quantum indifferentiability. In: Proceedings of Annual International Cryptology Conference, 2019. 239–268
18 Ambainis A, Rosmanis A, Unruh D. Quantum attacks on classical proof systems: the hardness of quantum rewinding. In: Proceedings of the 55th Annual Symposium on Foundations of Computer Science, 2014. 474–483
19 Shoup V. A proposal for an ISO standard for public key encryption. 2001. https://www.shoup.net/papers/iso-2_1.pdf#:~:text=This%20document%20is%20an%20initial%20proposal%20for%20a
20 Boneh D. Franklin M. Identity-based encryption from the Weil pairing. In: Proceedings of Annual International Cryptology Conference, 2001. 213–229
21 Fujisaki E, Okamoto T, Pointcheval D, et al. RSA-OAEP is secure under the RSA assumption. J Cryptol, 2004, 17: 81–104
22 Unruh D. Revocable quantum timed-release encryption. J ACM, 2015, 62: 1–76
23 Ambainis A, Hamburg M, Unruh D. Quantum security proofs using semi-classical oracles. In: Proceedings of the 39th Annual International Cryptology Conference, 2019. 269–295
24 Fujisaki E, Okamoto T. Secure integration of asymmetric and symmetric encryption schemes. J Cryptol, 2013, 26: 80–101
25 Hsiao C Y, Reyzin L. Finding collisions on a public road, or do secure hash functions need secret coins? In: Proceedings of Annual International Cryptology Conference, 2004. 92–105
26 Unruh D. Non-interactive zero-knowledge proofs in the quantum random oracle model. In: Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2015. 755–784
27 Chiesa A, Manohar P, Spooner N. Succinct arguments in the quantum random oracle model. In: Proceedings of Theory of Cryptography Conference, 2019. 1–29
28 Liu Q, Zhandry M. On finding quantum multi-collisions. In: Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2019. 189–218
29 Don J, Fehr S, Majenz C, et al. Security of the fiat-shamir transformation in the quantum random-oracle model. In: Proceedings of Annual International Cryptology Conference, 2019. 356–383
30 Hosoyamada A, Iwata T. 4-round luby-rackoff construction is a qPRP: tight quantum security bound. In: Proceedings of International Conference on the Theory and Application of Cryptology and Information Security, 2019
31 Nielsen M A, Chuang I. Quantum Computation and Quantum Information. Cambridge: Cambridge University Press, 2000
32 Shor P W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J Comput, 1997, 26: 1484–1509

Appendix A Definition and security of signatures

A digital signature scheme $\Pi_{\rm sig} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ consists of three PPT algorithms. Taking the security parameter κ as input, the key generation algorithm outputs a verification key vk and a secret signing key sk, i.e., $(\mathrm{vk}, \mathrm{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$. The signing algorithm takes vk, sk and a message $M \in \{0,1\}^*$ as inputs, outputs a signature σ on M, briefly denoted as $\sigma \leftarrow \mathsf{Sign}(\mathrm{sk}, M)$. The verification algorithm takes vk, message $M \in \{0,1\}^*$ and a string $\sigma \in \{0,1\}^*$ as inputs, outputs 1 if σ is a valid signature on M, else outputs 0, denoted as $1/0 \leftarrow \mathsf{Verify}(\mathrm{vk}, M, \sigma)$. For correctness, we require that for any $(\mathrm{vk}, \mathrm{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, any message $M \in \{0,1\}^*$, and any $\sigma \leftarrow \mathsf{Sign}(\mathrm{sk}, M)$, the equation $\mathsf{Verify}(\mathrm{vk}, M, \sigma) = 1$ holds with overwhelming probability, where the probability is taken over the choices of the random coins used in KeyGen, Sign and Verify. The standard security notion for the digital signature scheme is the EUF-CMA, which (informally) says that any PPT forger, after seeing valid signatures on a polynomial number of adaptively chosen messages, cannot create a valid signature on a new message. Formally, consider the following game between a challenger $\mathcal{C}$ and a forger $\mathcal{F}$:

KeyGen. The challenger $\mathcal{C}$ first runs $(\mathrm{vk}, \mathrm{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$ with the security parameter κ. Then, it gives the verification key vk to the forger $\mathcal{F}$, and keeps the signing secret key sk to itself.

Signing. The forger $\mathcal{F}$ is allowed to ask the signature on any fresh message M. The challenger $\mathcal{C}$ computes and sends $\sigma \leftarrow \mathsf{Sign}(\mathrm{sk}, M)$ to the forger $\mathcal{F}$. The forger can repeat this any polynomial times.

Forge. $\mathcal{F}$ outputs a message-signature pair (M^*, σ^*). Let Q be the set of all messages queried by $\mathcal{F}$ in the signing phase. If $M^* \notin Q$ and $\mathsf{Verify}(\mathrm{vk}, M^*, \sigma^*) = 1$, the challenger $\mathcal{C}$ outputs 1, else outputs 0.

We say that $\mathcal{F}$ wins the game if the challenger $\mathcal{C}$ outputs 1. The advantage of $\mathcal{F}$ in the above security game is defined as $\mathrm{Adv}^{\text{euf-cma}}_{\Pi_{\rm sig},\mathcal{F}}(1^\kappa) = \Pr[\mathcal{C} \text{ outputs } 1]$.

Definition 4 (EUF-CMA security)**.** Let κ be the security parameter. A signature scheme $\Pi_{\rm sig}$ is said to be EUF-CMA if the advantage $\mathrm{Adv}^{\text{euf-cma}}_{\Pi_{\rm sig},\mathcal{F}}(1^\kappa)$ is negligible in κ for any PPT forger $\mathcal{F}$.

Appendix B Definition and security of IBE

An IBE scheme consists of four PPT algorithms $\Pi_{\rm ibe} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$. Taking the security parameter κ as input, the randomized key generation algorithm Setup outputs a master public key mpk and a master secret key msk, denoted as $(\mathrm{mpk}, \mathrm{msk}) \leftarrow \mathsf{Setup}(1^\kappa)$. The (randomized) extract algorithm takes mpk, msk and an identity id as inputs, outputs a user private key $\mathrm{sk}_{\rm id}$ for id, briefly denoted as $\mathrm{sk}_{\rm id} \leftarrow \mathsf{Extract}(\mathrm{msk}, \mathrm{id})$. The randomized encryption algorithm Enc takes mpk, id and a plaintext M as inputs, outputs a ciphertext C, denoted as $C \leftarrow \mathsf{Enc}(\mathrm{mpk}, \mathrm{id}, M)$. The deterministic algorithm Dec takes $\mathrm{sk}_{\rm id}$ and C as inputs, outputs a plaintext M, or a special symbol $\perp$, which is denoted as $M/\perp \leftarrow \mathsf{Dec}(\mathrm{sk}_{\rm id}, C)$. In addition, for all $(\mathrm{mpk}, \mathrm{msk}) \leftarrow \mathsf{Setup}(1^\kappa), \mathrm{sk}_{\rm id} \leftarrow \mathsf{Extract}(\mathrm{msk}, \mathrm{id})$ and any plaintext M, we require that $\mathsf{Dec}(\mathrm{sk}_{\rm id}, C) = M$ holds for any $C \leftarrow \mathsf{Enc}(\mathrm{mpk}, \mathrm{id}, M)$. The standard semantic security of IBE was first introduced in [20]. Formally, consider the following game played by an adversary $\mathcal{A}$.

Setup. The challenger $\mathcal{C}$ first runs $\mathsf{Setup}(1^\kappa)$ with the security parameter κ. Then, it gives the adversary $\mathcal{A}$ the master public key mpk, and keeps the master secret key msk to itself.

Phase 1. The adversary is allowed to query the user private key for any identity id. The challenger $\mathcal{C}$ runs $\mathrm{sk}_{\rm id} \leftarrow \mathsf{Extract}(\mathrm{msk}, \mathrm{id})$ and sends $\mathrm{sk}_{\rm id}$ to the adversary $\mathcal{A}$. The adversary can repeat the user private key query any polynomial times for different identities.

Challenge. The adversary $\mathcal{A}$ outputs challenge plaintexts M_0^*, M_1^* and a challenge identity id^* with a restriction that id^* is not used in the user private key query in phase 1. The challenger $\mathcal{C}$ chooses a bit $b \in \{0,1\}$. Then, it computes $C^* \leftarrow \mathsf{Enc}(\mathrm{mpk}, \mathrm{id}^*, M_b^*)$. Finally, it sends C^* to $\mathcal{A}$.

Phase 2. The adversary can adaptively make more user private key queries with any identity $\mathrm{id} \neq \mathrm{id}^*$. The challenger $\mathcal{C}$ responds as in Phase 1.

Guess. Finally, $\mathcal{A}$ outputs a guess $b \in \{0,1\}$. If $b = b^*$, the challenger $\mathcal{C}$ outputs 1, else outputs 0.

The advantage of $\mathcal{A}$ in the above security game is defined as $\mathrm{Adv}^{\text{ind-id-cpa}}_{\Pi_{\rm ibe},\mathcal{A}}(\kappa) = |\Pr[b = b^*] - \frac{1}{2}|$.

Definition 5 (IND-ID-CPA security)**.** We say an IBE scheme $\Pi_{\rm ibe}$ is IND-ID-CPA secure if for any PPT adversary $\mathcal{A}$, its advantage $\mathrm{Adv}^{\text{ind-id-cpa}}_{\Pi_{\rm ibe},\mathcal{A}}(\kappa)$ is negligible in κ.

Appendix C Definition and security of PKE

A PKE scheme $\Pi_{\rm PKE}$ consists of three algorithms $\Pi_{\rm PKE} = (\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$. $\mathtt{Gen}$ is a randomized algorithm which takes a security parameter k as input, outputs a key pair $(\mathrm{pk}, \mathrm{sk})$. The probabilistic algorithm $\mathtt{Enc}$ takes as input a public key pk and a message m, returns a ciphertext c of m. $\mathtt{Dec}$ is a deterministic algorithm that takes as input a ciphertext c and a secret sk, outputs a message m or a special symbol $\perp$. We now recall the IND-CCA security of the PKE scheme. Consider the following game between a challenger and an adversary $\mathcal{A}$.

Setup. Given the security parameter k, the challenger generates $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{Gen}(1^k)$ and gives the public key pk to $\mathcal{A}$.

Phase 1. The adversary $\mathcal{A}$ may adaptively make a number of decryption queries on ciphertext c, the challenger responds with $\mathtt{Dec}(\mathrm{sk}, c)$.

Challenge. At some point, $\mathcal{A}$ outputs two equal-length messages m_0, m_1. The challenger chooses a random bit $b \in \{0,1\}$ and returns $c^* \leftarrow \mathtt{Enc}(\mathrm{pk}, m_b)$.

Phase 2. The adversary $\mathcal{A}$ makes more decryption queries as in phase 1, but with a constraint that decryption queries on c^* are not allowed.

Guess. Eventually, The adversary $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$.

The adversary wins the game if $b = b'$. The advantage of $\mathcal{A}$ in the above game is defined as $\mathrm{Adv}^{\text{ind-cca}}_{\Pi_{\rm PKE},\mathcal{A}}(k) = |\Pr[b = b'] - 1/2|$.

Definition 6 (IND-CCA)**.** We say that a PKE scheme $\Pi_{\rm PKE}$ is secure under chosen-ciphertext attacks (IND-CCA), if for any polynomial time adversary $\mathcal{A}$, its advantage in the above game is negligible.