On Post-Quantum Signature with Message Recovery from Hash-and-Sign in QROM

Bohang Chen, Shuai Han(✉), and Shengli Liu

School of Computer Science,
Shanghai Jiao Tong University, Shanghai 200240, China
{bohang0918,dalen17,slliu}@sjtu.edu.cn

Abstract. Post-quantum signatures have been suffering from their inefficiency compared to traditional signatures. To reduce space consumption, a promising approach is to design signature schemes with message recovery, which can embed part of the message into the signature without increasing its length. A notable example is the PSS-R signature scheme (probabilistic signature scheme with recovery) proposed by Bellare and Rogaway (EUROCRYPT 1996) in the classical ROM (random oracle model). However, there were few works considering post-quantum signature schemes with message recovery.

In this work, we study the design of post-quantum signature scheme with message recovery in the QROM (quantum ROM). Specifically, we adapt the classic PSS-R signature scheme to the post-quantum setting and prove its security in the QROM. Our security proof is conceptually similar to the original proof of PSS-R, but faces challenges in QROM when we need to reprogram two random oracles with correlated inputs/outputs. To address these issues, we extend the tight adaptive reprogramming theorem (Grilo et al., ASIACRYPT 2021) and the measure-and-reprogram theorem (Don et al., CRYPTO 2020) to our setting, to support reprogramming two oracles successively, where the input to one oracle depends on the other oracle's output. With these extended proof techniques, we provide the first security proof of a PSS-R-like signature scheme with message recovery in the QROM.

Keywords: Post-quantum security, digital signature, message recovery, QROM

1 Introduction

Post-quantum signatures with message recovery. We revisit a type of signature with message recovery adapted from hash-and-sign schemes in the post-quantum setting. Signature with message recovery is an enhancement of signature schemes that packs some part of the message into the signature, saving space for applications that require a large number of signatures stored (e.g. blockchain applications), or run in a memory-sensitive environment (e.g. IoT applications). A notable example of such signature schemes is the PSS-R scheme

(probabilistic signature scheme with recovery) proposed by Bellare and Rogaway [3].

Up to now, there have been few works considering post-quantum signature schemes with message recovery. Among them, del Pino et al. [15] proposed a post-quantum signature scheme with message recovery for a preliminary version of Falcon [9], by modifying a previous hash-and-sign signature scheme in [7] and achieving message recovery with a similar construction to PSS-R. With message recovery, the total size of a message and its signature in (the preliminary version of) Falcon can be reduced by up to 250 bytes. However, they only provided a security proof under the classical random oracle model (ROM), leaving a gap when taking the oracle's quantum accessibility into account.

In this work, we consider the post-quantum adaptation of the PSS-R scheme [3], which was originally instantiated under number-theoretic assumptions such as RSA, and was proved in the classical ROM.

Security of signature in QROM. The gap between security in the classical ROM and quantum ROM (QROM) has been studied for years, with many works trying to bridge the gap in a generic way [4,2,16,17,11,6].

In particular, Yamakawa and Zhandry [16] provided a lifting theorem, which can translate an adversary in some specific kinds of games that have access to only the classical random oracle, to another one that has access to the quantum random oracle. For digital signatures, based on such lifting theorem, they proved that with certain properties, the EUF-CMA security (existential unforgeability under chosen message attacks) of a digital signature scheme in the ROM can be lifted to the EUF-CMA security in the QROM directly.

However, it is hard to lift the security of the PSS-R signature scheme [3] to QROM via the lifting theorem [16], since the PSS-R scheme does not meet the requirements of the lifting theorem. More precisely, the main problem is that the PSS-R scheme [3] makes use of two random oracles successively, and the query for one oracle depends on the output from the other oracle. This makes it hard to develop a method to simulate the signing oracle by only non-adaptively reprogramming the random oracle, as required by the lifting theorem [16]. Besides, their theorem does not state the concrete security loss of lifting, making it hard to quantify the tightness of such reduction.

Therefore, to provide a security proof for the PSS-R signature scheme [3] under QROM with concrete security loss, one may need to resort to more fundamental techniques such as the tight adaptive reprogramming technique introduced by Grilo et al. [11] and/or the measure-and-reprogram technique introduced by Don et al. [6], and build the security proof from scratch.

However, the techniques in [11,6] remain unsuitable for proving the security of the PSS-R scheme [3] under QROM. Specifically, the tight adaptive reprogramming technique in [11] cannot be used to reprogram two random oracles where the input to one oracle depends on the output from the other oracle, as is the case with the PSS-R scheme. Moreover, the measure-and-reprogram technique in [6] focuses on reprogramming a single random oracle, and can hardly be used to reprogram two random oracles simultaneously.

This makes the post-quantum adaptation of PSS-R even more challenging. It is interesting to ask the following question:

How to adapt the PSS-R signature scheme under QROM based on post-quantum assumptions?

Our contributions. In this work, we answer the above question in the affirmative.

- *Generalized PSS-R from preimage sampleable function.* To adapt the PSS-R scheme [3] under post-quantum assumptions, we generalize it by proposing a generic construction of signature schemes with message recovery based on a preimage sampleable function (PSF) [10], which was known to have many post-quantum instantiations.
- *Security proof under QROM.* As our main contribution, we prove the EUF-CMA security of our generalized PSS-R signature scheme under QROM, and show its concrete security loss. To this end, we extend the tight adaptive reprogramming technique in [11] and the measure-and-reprogram technique in [6] to support the reprogramming of two random oracles where the input for one oracle depends on the output from the other oracle, as is the case of the PSS-R and our schemes.

  With our extended techniques, our security proof under QROM goes with two main steps. First, we reduce the EUF-NMA security (existential unforgeability under no message attacks, i.e., without signing oracle) of our scheme to the EUF-CMA security, by using the extended tight adaptive reprogramming technique. Then, we prove the EUF-NMA security based on the properties of PSF, by using the extended measure-and-reprogram technique. We refer to Sect. 4 for more details.

  To the best of our knowledge, this yields the *first* security proof under QROM for (variants of) the PSS-R scheme.

From a technical perspective, our extensions of the techniques in [11,6] are non-trivial, as explained below.

- *Extended tight adaptive reprogramming technique.* To support the reprogramming of two random oracles, we first extend the tight adaptive reprogramming technique [11] in a more quantum-friendly way by establishing a superposition-controlled reprogramming lemma (i.e., Lemma 2), so that we can control the position of the random oracle to be reprogrammed with quantum superposition, rather than with a classical variable from a distribution as was done in [11]. Then, we apply our Lemma 2 to the quantum representation of the reprogramming game involving two random oracles to erase the reprogramming of the two oracles one by one, even if the input for one oracle depends on the output from the other oracle. This shows that the reprogramming of the two oracles is unnoticeable to a quantum adversary (i.e., Theorem 5). We refer to Sect. 3 for more details.

  We note that our extension is non-trivial, as their technique [11] doesn't imply our theorem even in a white-box way (see Subsect. 3.1 for more explanations).

- *Extended measure-and-reprogram technique.* The second extension is a reformulation of the measure-and-reprogram technique [6] to handle the case where two random oracles are involved. While this may appear analogous to the situation considered by the measure-and-reprogram theorem for multiple queries to a single random oracle with enforced extraction order (i.e., [6, Theorem 7]), it is actually quite different since we deal with two random oracles. Therefore, we adapt their theorem to our setting and give a concrete bound. We refer to Theorem 3 in Subsect. 2.4 for more details.

Instantiations and comparison with Falcon. By instantiating the PSF following the Falcon signature's approach [9], we can obtain a concrete signature scheme with message recovery via our generic construction. Specifically, we use the NTRU-Lattice [12] for a trapdoor, and use the fast Fourier sampling [8] to sample the preimage. This results in the *first* post-quantum signature scheme supporting message recovery in QROM.

Below we compare our scheme with Falcon-512 [9] to show the advantage of message recovery.

- When instantiating our scheme with the PSF and the parameters from Falcon-512 [9], our signature scheme has 1252 bytes signature size, where we can embed 787 bytes message into the signature.[1]
- The Falcon-512 signature scheme has a signature size of 666 bytes, which does not support message embedding.

So with our signature scheme, for a message of size $\ell_m \geq 787$ bytes, we can embed 787 bytes into the signature, so that the total size of the message and its signature is $\ell_m - 787 + 1252 = \ell_m + 465$ bytes; while for messages of size $\ell_m < 787$ bytes, we can embed the whole message into the signature, so that the total size of the message and its signature is 1252 bytes.

In comparison, for Falcon-512, the total size of the message and signature is $\ell_m + 666$ bytes regardless of the message length ℓ_m.

This shows that our signature scheme with message recovery has an advantage over Falcon-512 for messages of size $\ell_m \geq 586$ bytes (so that both $\ell_m + 465$ and $1252 \leq \ell_m + 666$). Compared to Falcon-512 (whose signature is 666 bytes), our scheme can save up to 201 bytes of space, cutting the total space overhead of the signature by approximately 30% .

We note that our advantage holds even if the messages are hashed to short digests, as is done in many concrete deployment scenarios. The point is that even if the signatures are generated w.r.t. short digests (say $H(m)$), one still needs to store the entire message (say m). With our scheme, one can parse $m = (m_1, m_2)$, and generate a signature σ over $m_1||H(m)$, in which case one only needs to store

[1] Jumping ahead, our generic construction is shown in Fig. 5 in Sect. 4, and here we instantiate the set $\mathcal{Y} = \mathcal{Y}_1 \times \mathcal{Y}_2 \times \mathcal{Y}_3$ with $\mathcal{Y}_1 = \{0,1\}^{320}$, $\mathcal{Y}_2 = \mathcal{M}_1 = \{0,1\}^{787\times 8}$ and $\mathcal{Y}_3 = \{0,1\}^{320}$, instantiate the PSF $\mathsf{PSF} = (\mathsf{TrapGen}, \mathsf{SampleDom}, \mathsf{SamplePre})$ with the one involved in Falcon-512 [9], and instantiate the hash functions H and G similar to the ones used in Falcon-512 [9].

(m_2, σ), thus saving the space for storing m_1. This approach achieves our claimed advantage.

More related works. Among recent progress on QROM proof techniques, two lines of work are particularly relevant to our technical approach.

Specifically, Pan and Zeng [14] discussed the computational version of the adaptive reprogramming technique, but its direct application to statistical settings like ours would incur a substantial security loss.

Concurrently, Cojocaru et al. [5] developed the coherent measure-and-reprogram technique. Their improvement replaces a standard measurement with a coherent operation that writes the result into a separate quantum register. Although this method does not directly apply to our use case, its core insight—delaying the irreversible collapse of a quantum register to harness superposition—is conceptually similar to our extension of the adaptive reprogramming lemma.

More discussions. Finally, we note that our established superposition-controlled reprogramming lemma (i.e., Lemma 2) might have broader applications beyond serving merely as a tool for proving the adaptive reprogramming theorem for two random oracles. Our Lemma 2 allows reprogramming controlled by quantum superposition, instead of a classical distribution. This provides better compatibility with quantum operation, and convinces us of its plausible application to security proofs in a fully quantum setting, where both the cryptographic system and the adversary are assumed to have quantum power. We leave it as future work to find such applications.

2 Preliminaries

Let $\lambda \in \mathbb{N}$ denote the security parameter throughout the paper. If x is defined by y or the value of y is assigned to x, we write $x := y$. For $n \in \mathbb{N}$, define $[n] := \{1, 2, ..., n\}$. For a set $\mathcal{X}$, denote by $x \leftarrow \mathcal{X}$ the procedure of sampling x from $\mathcal{X}$ uniformly at random. If $\mathcal{D}$ is a distribution, $x \leftarrow \mathcal{D}$ means that x is sampled according to $\mathcal{D}$. All our algorithms are probabilistic unless stated otherwise. We use $y \leftarrow \mathcal{A}(x)$ to define the random variable y obtained by executing algorithm $\mathcal{A}$ on input x. We also use $y \leftarrow \mathcal{A}(x; r)$ to make explicit the random coins r used in the probabilistic computation. "PPT" abbreviates probabilistic polynomial-time. Denote by negl some negligible function. For two sets $\mathcal{X}$ and $\mathcal{Y}$, we denote the collection of all functions from $\mathcal{X}$ to $\mathcal{Y}$ as $\mathcal{Y}^{\mathcal{X}}$. In our security proofs, we use $\Pr_i$ to denote the probability of the event that game G_i returns 1.

2.1 Digital Signature

Definition 1 (Digital Signature). *A digital signature scheme over message space $\mathcal{M}$ consists of three PPT algorithms $\mathsf{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy})$:*

- $(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$*: This algorithm takes as input the security parameter 1^λ, outputting a verification key vk and a signing key sk.*

– $\sigma \leftarrow \mathsf{Sign}(sk, m)$: *This algorithm takes as input a signing key sk and a message $m \in \mathcal{M}$, outputting a signature σ for the message.*
– $0/1 \leftarrow \mathsf{Vrfy}(vk, m, \sigma)$: *This is a deterministic algorithm that takes as input a verification key vk, a message m and a signature σ, outputting a bit indicating whether the signature is valid.*

In some memory sensitive applications, we may want to use a digital signature scheme with message recovery, which supports embedding some part of the message (say m_1) into the signature, so that only the remaining part of the message (say m_2) is needed in the verification of the signature.

Definition 2 (Digital Signature with Message Recovery). *A digital signature scheme with message recovery over message space $\mathcal{M} = \mathcal{M}_1 \times \mathcal{M}_2$ consists of three PPT algorithms $\mathsf{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy})$:*

– $(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $\sigma \leftarrow \mathsf{Sign}(sk, m)$: *These algorithms work as per the ordinary digital signature scheme in Definition 1.*
– $(0, \perp)/(1, m_1') \leftarrow \mathsf{Vrfy}(vk, m_2, \sigma)$: *This is a deterministic algorithm that takes as input a verification key vk, the second part $m_2 \in \mathcal{M}_2$ of a message and a signature σ, outputting $(0, \perp)$ if the signature is invalid, and outputting $(1, m_1' \in \mathcal{M}_1)$ if the signature is valid.*

The **correctness** property requires that for any $(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and any $m = (m_1, m_2) \in \mathcal{M}$, it holds that

$$\Pr\left[\, \mathsf{Vrfy}(vk, m_2, \sigma) = (1, m_1) : \sigma \leftarrow \mathsf{Sign}(sk, m) \,\right] = 1 - \mathsf{negl}(\lambda).$$

Below we define the EUF-CMA security (existential unforgeability under chosen message attacks) and EUF-NMA security (existential unforgeability under no message attacks) for digital signature with message recovery.

Definition 3 (EUF-CMA Security). *A digital signature scheme with message recovery SIG over message space $\mathcal{M} = \mathcal{M}_1 \times \mathcal{M}_2$ is* EUF-CMA *secure, if for any PPT adversary $\mathcal{A}$, it holds that $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}} := \Pr[\,\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}} = 1\,] \leq \mathsf{negl}(\lambda)$, where the experiment $\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}}$ is shown in Fig. 1.*

Definition 4 (EUF-NMA Security). *A digital signature scheme with message recovery SIG over message space $\mathcal{M} = \mathcal{M}_1 \times \mathcal{M}_2$ is* EUF-NMA *secure, if for any PPT adversary $\mathcal{A}$, it holds that $\mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}} := \Pr[\,\mathsf{Exp}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}} = 1\,] \leq \mathsf{negl}(\lambda)$, where the experiment $\mathsf{Exp}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}}$ is shown in Fig. 1.*

2.2 Preimage Sampleable Function

Below we recall the definition of preimage sampleable function (PSF) from [10].

Definition 5 (Preimage Sampleable Function [10]). *A collection of preimage sampleable functions consists of three PPT algorithms $\mathsf{PSF} = (\mathsf{TrapGen}, \mathsf{SampleDom}, \mathsf{SamplePre})$:*

$\underline{\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}}}$:	$\underline{\mathsf{Exp}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}}}$:	$\underline{\textsc{Sign}(m = (m_1, m_2))}$:
$\mathcal{S} := \emptyset$	$(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$	$\sigma \leftarrow \mathsf{Sign}(sk, m)$
$(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$	$(m_2^*, \sigma^*) \leftarrow \mathcal{A}(vk)$	$\mathcal{S} := \mathcal{S} \cup \{(m_1, m_2)\}$
$(m_2^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{Sign}(\cdot)}(vk)$	$(b, m_1^*) \leftarrow \mathsf{Vrfy}(vk, m_2^*, \sigma^*)$	Return σ
$(b, m_1^*) \leftarrow \mathsf{Vrfy}(vk, m_2^*, \sigma^*)$	Return $b = 1$	
Return $(m_1^*, m_2^*) \notin \mathcal{S} \wedge b = 1$		

Fig. 1. The EUF-CMA experiment $\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}}$ (left) and the EUF-NMA experiment $\mathsf{Exp}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}}$ (middle) for digital signature with message recovery, where the oracle SIGN (right) is used in the former experiment.

- $(a, t) \leftarrow \mathsf{TrapGen}(1^\lambda)$*: This algorithm takes as input the security parameter* 1^λ*, outputting 'a' as the description of a trapdoor function* $f_a\colon \mathcal{X} \to \mathcal{Y}$ *(for some sets* $\mathcal{X}$ *and* $\mathcal{Y}$ *depending on* λ*) and 't' as a trapdoor for* f_a*.*
- $x \leftarrow \mathsf{SampleDom}(1^\lambda)$*: This algorithm takes as input the security parameter* 1^λ*, sampling an element* x *from some distribution* $\mathcal{D}_x$ *over* $\mathcal{X}$*.*
- $x \leftarrow \mathsf{SamplePre}(y, t)$*: This algorithm takes as input* $y \in \mathcal{Y}$ *and the trapdoor* t *for* f_a*, outputting an element* $x \in \mathcal{X}$*.*

Fixing $(a, t) \leftarrow \mathsf{TrapGen}(1^\lambda)$, we require a preimage sampleable function PSF to satisfy the following properties:

- **Image Distribution**: For $x \leftarrow \mathsf{SampleDom}(1^\lambda)$, $f_a(x)$ follows the uniform distribution over $\mathcal{Y}$.
- **Preimage Sampling Correctness**: For any $y \in \mathcal{Y}$, $x \leftarrow \mathsf{SamplePre}(y, t)$ satisfies $f_a(x) = y$.
- **Preimage Sampling Distribution**: For any $y \in \mathcal{Y}$, $x \leftarrow \mathsf{SamplePre}(y, t)$ follows the distribution $\mathcal{D}_x$ conditioned on $f_a(x) = y$.
- **Preimage Min-entropy**: For every $y \in \mathcal{Y}$, the conditional probability,

$$p_{\max} := \max_{x' \in \mathcal{X}} \Pr_x [x = x' \mid f_a(x) = y] = 2^{-\omega(\log \lambda)},$$

  where the probability is over $x \leftarrow \mathsf{SampleDom}(1^\lambda)$.
- **Collision Resistance**: For any PPT adversary $\mathcal{A}$, it holds that

$$\mathsf{Adv}^{\mathsf{CR}}_{\mathsf{PSF},\mathcal{A}} := \Pr [\, x \neq x' \wedge f_a(x) = f_a(x') \,] \leq \mathsf{negl}(\lambda),$$

  where the probability is over $(a, t) \leftarrow \mathsf{TrapGen}(1^\lambda)$ and $(x, x') \leftarrow \mathcal{A}(a)$.

2.3 Quantum Computation

We rely on the common notation as in [13] for quantum computation. For completeness, we provide an introduction to the notations that we use.

Let $\mathcal{H}$ be a finite-dimensional complex Hilbert space, representing the state space of a quantum system. We represent the vectors in $\mathcal{H}$ and its dual space with the standard bra-ket notation.

Let X be a quantum register with its state space being $\mathcal{H}_X$. We use the register's name as subscripts for vectors and operators. Specifically, we denote $|\cdot\rangle_X$ and $\langle\cdot|_X$ for vectors in $\mathcal{H}_X$ and its dual space, respectively. We also denote U_X for operators acting on $\mathcal{H}_X$. If the operator is a density operator, it will be describing the state of register X. If there are multiple registers' names written in the subscripts, it represents vectors or operators of a multi-register system.

For multi-register systems, the state space is the tensor product of the single-register spaces with an implicitly fixed register order. To simplify notation, we use subscripts to permute registers as needed, and the displayed tensor-product order is immaterial. Furthermore, for an operator U, we may omit some registers' subscripts when it acts as identity on those registers within the multi-register system.

For a Hilbert space $\mathcal{H}$ of dimension d, we mainly work on the computational basis $|0\rangle$, $|1\rangle$, $\cdots$, $|d-1\rangle$. We also utilize the Fourier basis $|\phi_0\rangle$, $|\phi_1\rangle$, $\cdots$, $|\phi_{d-1}\rangle$ given by $|\phi_i\rangle \stackrel{\text{def}}{=} \mathcal{F}|i\rangle$, where $\mathcal{F}$ is the operator of Quantum Fourier Transform (QFT) [13, Chapter 5]. Nevertheless, among the vectors in the Fourier basis, we only need to note the form of $|\phi_0\rangle$, which is the "uniform state" $|\phi_0\rangle \stackrel{\text{def}}{=} \frac{1}{\sqrt{d}}\sum_{i=0}^{d-1}|i\rangle$.

Our quantum systems always employ binary encoding. When we say a register X contains a superposition over a finite non-empty set $\mathcal{X}$, we mean the register X contains $\lceil \log_2 |\mathcal{X}| \rceil$ qubits, and each element $x \in \mathcal{X}$ is encoded as a binary string of the same length within X.

Quantum Gates. Let us introduce some well-known quantum gates. One is the CNOT gate and its extension. The ordinary $\mathsf{CNOT}_{X,Y}$ gate is a unitary acting on two qubits and does $\mathsf{CNOT}_{X,Y}|b_1\rangle_X|b_2\rangle_Y = |b_1\rangle_X|b_2 \oplus b_1\rangle_Y$, for all $b_1, b_2 \in \{0,1\}$ where $\oplus$ denotes the XOR operation and the former qubit is called the control bit. We may also use $\mathsf{CNOT}_{X,Y}$ to denote its multi-qubit version, acting on two registers X and Y, each of n qubits. The multi-qubit $\mathsf{CNOT}_{X,Y}$ gate applies single-qubit $\mathsf{CNOT}_{X,Y}$ gates for n times, the i-th time on the i-th qubit of X and Y with X's qubit always being the control bit.

Another is the SWAP gate, acting on two registers X and Y that share the same state space $\mathcal{H}_X$. The $\mathsf{SWAP}_{X,Y}$ gate does $\mathsf{SWAP}_{X,Y}|x_1\rangle_X|x_2\rangle_Y = |x_2\rangle_X|x_1\rangle_Y$ for all $x_1, x_2 \in \mathcal{X}$. Intuitively, the SWAP gate's action is equivalent to swapping the two registers physically.

Useful Metrics. We now introduce some useful metrics that help us analyze the quantum system. We introduce the fidelity and the trace distance, both to describe the similarity between two quantum states.

The *fidelity* of two quantum states (in density operator form) ρ and σ is defined as $F(\rho,\sigma) = \mathrm{tr}\left(\sqrt{\rho^{1/2}\sigma\rho^{1/2}}\right)$, where tr denotes the trace of an operator. For pure states $|\psi\rangle$ and $|\phi\rangle$, the fidelity can be written in a simpler form as $F(|\psi\rangle\langle\psi|, |\phi\rangle\langle\phi|) = |\langle\psi|\phi\rangle|$.

The *trace distance* of two quantum states ρ and σ is defined as $D(\rho,\sigma) = \frac{1}{2}\mathrm{tr}(|\rho-\sigma|)$, where $|\rho-\sigma| \stackrel{\text{def}}{=} \sqrt{(\rho-\sigma)^\dagger(\rho-\sigma)}$.

The trace distance is closely related to distinguishing two quantum states. It is shown that for all possible measurement D, the probability of distinguishing two quantum states ρ and σ is upper bounded by trace distance [13, Chapter 9], i.e.,

$$|\Pr[\mathsf{D}(\rho) = 1] - \Pr[\mathsf{D}(\sigma) = 1]| \leq D(\rho, \sigma).$$

In our analysis, we will use fidelity to bound the trace distance, using the inequality $D(\rho, \sigma) \leq \sqrt{1 - F(\rho, \sigma)^2}$ in [13, Chapter 9].

2.4 Quantum Random Oracle

A quantum random oracle extends the classical random oracle by allowing quantum access to the oracle. Formally, for finite non-empty sets $\mathcal{X}$ and $\mathcal{Y}$, a quantum random oracle first samples a function $H \leftarrow \mathcal{Y}^{\mathcal{X}}$ uniformly at random. Then, it answers each query by applying a unitary on quantum registers X and Y, where X contains a superposition over $\mathcal{X}$ and Y contains a superposition over $\mathcal{Y}$. This unitary maps the transformation $|x\rangle_X |y\rangle_Y \to |x\rangle_X |y \oplus H(x)\rangle_Y$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

Following Grilo et al. [11], our formulation of the adaptive reprogramming relies on the superposition oracle [17].

The superposition oracle is a way to simulate the quantum random oracle with a quantum circuit (inefficiently), with an internal register F serving as a truth table. The internal register F can be split into slots of registers denoted by F_x for $x \in \mathcal{X}$, corresponding to the value of $H(x)$. It works as follows:

- At the beginning of the simulation, the simulator prepares its internal register F, such that the state of F_x is $|\phi_0\rangle_{F_x}$ for all $x \in \{0,1\}^n$.
- On a quantum random oracle query $|x\rangle_X |y\rangle_Y$, the simulator applies a unitary,

$$\mathcal{O}^{(H)}_{XYF} \stackrel{\text{def}}{=} \sum_{x \in \{0,1\}^n} |x\rangle\langle x|_X \otimes \mathsf{CNOT}_{F_x, Y}$$

To understand how this simulation works, we can treat the superposition oracle as storing the uniform superposition of all possible truth tables of the random oracle in its internal register F, and looking up the table to answer random oracle queries.

At first glance, adding a measurement of F right after the preparation of the uniform state seems to be necessary, so that the simulation lines up with the ordinary description of a quantum random oracle, where H is sampled uniformly from $\mathcal{X} \to \mathcal{Y}$ at the beginning of the simulation.

However, this is unnecessary, as in the quantum gate $\mathcal{O}^{(H)}_{XYF}$, the register F always acts as a control register, thus $\mathcal{O}^{(H)}_{XYF}$ commutes with a measurement acting on F. Therefore, we can always postpone the measurement of F to the end of the simulation, or even ignore it equivalently, and the simulation is still perfect.

The superposition oracle is useful especially when we need to reprogram the random oracle on certain positions, as it allows us to formulate the reprogramming process simply by replacing the original register within F with a fresh uniform random state, as was done in [11].

When we need an efficient simulation of the random oracle, we use the compressed oracle technique, which is also introduced in [17]. This allows us to simulate the quantum random oracle with polynomial-sized registers and within polynomial time.

To analyze such replacements properly, we need the following lemma:

Lemma 1 (Decomposition of superposition oracle's state [11,1]). *Let $|\gamma\rangle_{AF}$ be the joint adversary-oracle state after an adversary has made q queries to a superposition oracle with register F. Then, the state can be decomposed as,*

$$|\gamma\rangle_{AF} = \sum_{\substack{S\subset\mathcal{X}\\|S|\leq q}} \beta_S \left|\gamma^{(S)}\right\rangle_{AF_S} \otimes \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}},$$

where F_S for $S\subset\mathcal{X}$ denotes the combination of registers F_x for all $x\in S$, and $\left|\gamma^{(S)}\right\rangle_{AF_S}$ are vectors such that $\langle\phi_0|_{F_x}\left|\gamma^{(S)}\right\rangle_{AF_S}=0$ for all $x\in S$.

Another important result is the optimality of Grover Search theorem in the language of quantum random oracle.

Theorem 1 (Optimality of Grover Search [17], adapted). *For any adversary $\mathcal{A}$ making q quantum queries to a random oracle $H:\mathcal{X}\to\mathcal{Y}$, the probability of finding the preimage of an arbitrary $y\in\mathcal{Y}$ is given by*

$$\Pr\left[H(x)=y : x\leftarrow\mathcal{A}^H(y)\right] \leq \left(\sqrt{\tfrac{(q+1)^2}{|\mathcal{Y}|}}+\sqrt{\tfrac{1}{|\mathcal{Y}|}}\right)^2 = \tfrac{(q+2)^2}{|\mathcal{Y}|}.$$

A crucial tool for our analysis is the measure-and-reprogram theorem as follows.

Theorem 2 (Measure and Reprogram, single input [6]). *Let $\mathcal{X}_0,\mathcal{X}_1$ be finite non-empty sets. There exists a black-box polynomial-time 2-stage quantum algorithm $\mathcal{S}$ such that, for an arbitrary oracle quantum algorithm $\mathcal{A}$ that makes q queries to a uniform random $H:\mathcal{X}_0\to\mathcal{X}_1$ and that outputs some $x\in\mathcal{X}_0$ and a (possibly quantum) output z, the algorithm $\mathcal{S}^{\mathcal{A}}$ works as follows:*

- *In the first stage, it takes the initial input, and outputs some $x\in\mathcal{X}_0$.*
- *In the second stage, it takes some $\Theta\in\mathcal{X}_1$ as an additional input, and outputs the final output (x,z).*

Then, for uniform random $\Theta\leftarrow\mathcal{X}_1$, for any $x^\circ\in\mathcal{X}_0$ and any (possibly quantum) predicate V:

$$\Pr_{\Theta}\left[x=x^\circ\wedge V(x,\Theta,z) : (x,z)\leftarrow\left\langle\mathcal{S}^{\mathcal{A}},\Theta\right\rangle\right] \geq \tfrac{1}{(2q+1)^2}\cdot\Pr_H\left[x=x^\circ\wedge V(x,H(x),z):(x,z)\leftarrow\mathcal{A}^H\right].$$

We consider the case of applying measure-and-reprogram to two successive oracle queries $(h^{(x)} = H(x), g^{(x)} = G(h^{(x)}))$ with two functions $H : \mathcal{X}_0 \to \mathcal{X}_1$ and $G : \mathcal{X}_1 \to \mathcal{X}_2$. In this case, we must reprogram two different oracles in an enforced order (i.e., reprogram oracle G after reprogramming H). The following theorem shows an adaptation of the theorem in [6] to our setting.

Theorem 3 (Measure and Reprogram, two oracles). *Let $\mathcal{X}_0, \mathcal{X}_1$ and $\mathcal{X}_2$ be finite non-empty sets. There exists a black-box polynomial-time 3-stage quantum algorithm $\mathcal{S}$ such that, for an arbitrary oracle quantum algorithm $\mathcal{A}$ that makes q_H queries to a uniform random $H : \mathcal{X}_0 \to \mathcal{X}_1$ and q_G queries to a uniform random $G : \mathcal{X}_1 \to \mathcal{X}_2$, and that outputs $x_H \in \mathcal{X}_0$ and a (possibly quantum) output z, the algorithm $\mathcal{S}^{\mathcal{A}}$ works as follows:*

- *In the first stage, it takes the initial input, and outputs some $x_H \in \mathcal{X}_0$.*
- *In the second stage, it takes some $\Theta_H \in \mathcal{X}_1$ as an additional input, and outputs some $x_G \in \mathcal{X}_1$.*
- *In the third stage, it takes some $\Theta_G \in \mathcal{X}_2$ as an additional input, and outputs the final output (x_H, z).*

Moreover, it satisfies the following properties. Let $\mathcal{A}$ be an arbitrary oracle quantum algorithm that outputs $x_H \in \mathcal{X}_0$ and a (possibly quantum) auxiliary output z. Then, for uniform random $\Theta_H \leftarrow \mathcal{X}_1$ and $\Theta_G \leftarrow \mathcal{X}_2$, for any $x^\circ \in \mathcal{X}_0$ and for any predicate V:

$$\begin{aligned}&\Pr_{\Theta_H,\Theta_G}\left[x_H = x^\circ \wedge V(x_H, \Theta_H, \Theta_G, z) : (x_H, z) \leftarrow \langle \mathcal{S}^{\mathcal{A}}, \Theta_H, \Theta_G\rangle\right] \\ &\geq \tfrac{1}{(2\max\{q_H,q_G\}+1)^4} \cdot \Pr_{H,G}\left[x = x^\circ \wedge V\left(x_H, h^{(x)}, g^{(x)}, z\right) : (x_H, z) \leftarrow \mathcal{A}^{H,G}\right] - \varepsilon_{x^\circ},\end{aligned}$$

where $h^{(x)} \stackrel{def}{=} H(x)$, $g^{(x)} \stackrel{def}{=} G(h^{(x)})$, and $\sum_{x^\circ \in \mathcal{X}_0} \varepsilon_{x^\circ} = \frac{1}{|\mathcal{X}_1|}$.

Although adapted from the original theorem for multiple input measure-and-reprogram with enforced extraction order (Theorem 7 in [6]), our result differs in the sense that we are dealing with two random oracles at once. Therefore, we provide a proof of the theorem here.

Intuitively, the proof first applies Theorem 2 twice on both oracles respectively. At this point, it is not guaranteed that the two oracles are reprogrammed in our desired order. Therefore, we then carefully analyze the probability of the event that G is reprogrammed before H is reprogrammed, and exclude the occurrence of this event to complete the proof.

Proof (Theorem 3). Consider an intermediate adversary $\mathcal{A}_+^{H,G}$ that calls $\mathcal{A}$ to obtain $(x_H, z) \leftarrow \mathcal{A}^{H,G}$, and then queries $x_G := y'_H := H(x_H)$, $y'_G := G(x_G)$ before outputting $(x_H, x_G, y'_H, y'_G, z)$.

We first reprogram the oracle H. To do this, we consider two new predicates

$$V'(x_H, x_G, y_H, y_G, z) = [V(x_H, y_H, y_G, z) \wedge x_G = y_H],$$

and,

$$V''(x_H, x_G, y_H, y_G, z) = [V'(x_H, x_G, y_H, y_G, z) \wedge y_G = G(y_H)].$$

Note that computing $G(y_H)$ with the predicate is possible in this case, as we can w.l.o.g use the compressed oracle technique [17] to simulate G, and output the oracle G's internal states (which is of polynomial size) within z when we are reprogramming the oracle H.

Clearly, for any $x_H^\circ \in \mathcal{X}_0$,

$$\begin{aligned}
&\Pr\left[x_H = x_H^\circ \wedge V(x_H, h^{(x_H)}, g^{(x_H)}, z) = 1 : (x_H, z) \leftarrow \mathcal{A}^{H,G}\right] \\
=&\Pr\left[x_H = x_H^\circ \wedge V''(x_H, x_G, h^{(x_H)}, g^{(x_H)}, z) = 1 : (x_H, x_G, y'_G, z) \leftarrow \mathcal{A}_+^{H,G}\right] \\
=&\Pr\left[x_H = x_H^\circ \wedge V''(x_H, x_G, h^{(x_H)}, y'_G, z) = 1 : (x_H, x_G, y'_G, z) \leftarrow \mathcal{A}_+^{H,G}\right].
\end{aligned}$$

By Theorem 2, there exists a two-stage simulator $\mathcal{S}_1$ such that,

$$\begin{aligned}
&\Pr\left[x_H = x_H^\circ \wedge V''(x_H, x_G, \Theta_H, y'_G, z) = 1 : (x_H, x_G, y'_G, z) \leftarrow \left\langle \mathcal{S}_1^{\mathcal{A}_+^G}, \Theta_H \right\rangle\right] \\
\geq&\tfrac{1}{(2q_H+1)^2} \Pr\left[x_H = x_H^\circ \wedge V'(x_H, x_G, h^{(x_H)}, y'_G, z) = 1 : (x_H, x_G, y'_G, z) \leftarrow \mathcal{A}_+^{H,G}\right]
\end{aligned}$$

As $V''(x_H, x_G, \Theta_H, y'_G, z) = 1$ indicates $y'_G = G(\Theta_H) = G(x_G)$, we also have,

$$\begin{aligned}
&\Pr\left[x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, G(x_G), z) = 1 : (x_H, x_G, y'_G, z) \leftarrow \left\langle \mathcal{S}_1^{\mathcal{A}_+^G}, \Theta_H \right\rangle\right] \\
\geq&\Pr\left[x_H = x_H^\circ \wedge V''(x_H, x_G, \Theta_H, y'_G, z) = 1 : (x_H, x_G, y'_G, z) \leftarrow \left\langle \mathcal{S}_1^{\mathcal{A}_+^G}, \Theta_H \right\rangle\right].
\end{aligned}$$

Let $\mathcal{B}^G$ be an adversary that samples Θ_H and calls $\left\langle \mathcal{S}_1^{\mathcal{A}_+^G}, \Theta_H \right\rangle$ to obtain (x_H, x_G, y'_G, z). We make $(x_H, x_G, y'_G, z, \Theta_H)$ to be $\mathcal{B}$'s output. Now, we can apply Theorem 2 to reprogram the oracle G, treating $\mathcal{B}$ as the algorithm $\mathcal{A}$ in Theorem 2. Therefore, for any $x_G^\circ \in \mathcal{X}_1$,

$$\begin{aligned}
&\Pr\left[(x_H, x_G) = (x_H^\circ, x_G^\circ) \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 : (x_H, x_G, y'_G, z, \Theta_H) \leftarrow \left\langle \mathcal{S}_2^{\mathcal{B}^G}, \Theta_G \right\rangle\right] \\
\geq&\tfrac{1}{(2q_G+1)^2} \Pr\left[(x_H, x_G) = (x_H^\circ, x_G^\circ) \wedge V'(x_H, x_G, \Theta_H, G(x_G), z) = 1 : (x_H, x_G, y'_G, z, \Theta_H) \leftarrow \mathcal{B}^G\right].
\end{aligned}$$

With all above, we found that,

$$\begin{aligned}
&\Pr\left[(x_H, x_G) = (x_H^\circ, x_G^\circ) \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 : (x_H, x_G, y'_G, z, \Theta_H) \leftarrow \left\langle \mathcal{S}_2^{\mathcal{B}^G}, \Theta_G \right\rangle\right] \\
\geq&\tfrac{1}{(2q_G+1)^2(2q_H+1)^2} \Pr\left[(x_H, x_G) = (x_H^\circ, x_G^\circ) \wedge V(x_H, h^{(x_H)}, g^{(x_H)}, z) = 1 : (x_H, z) \leftarrow \mathcal{A}^{H,G}\right].
\end{aligned}$$

Summing over all $x_G^\circ \in \mathcal{X}_1$, we obtain,

$$\begin{aligned}
&\Pr\left[x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 : (x_H, x_G, y'_G, z, \Theta_H) \leftarrow \left\langle \mathcal{S}_2^{\mathcal{B}^G}, \Theta_G \right\rangle\right] \\
\geq&\tfrac{1}{(2q_G+1)^2(2q_H+1)^2} \Pr\left[x_H = x_H^\circ \wedge V(x_H, h^{(x_H)}, g^{(x_H)}, z) = 1 : (x_H, z) \leftarrow \mathcal{A}^{H,G}\right].
\end{aligned}$$

At this point, we are just one step away from completing the proof. Considering the running procedure of $\mathcal{S}_2^{\mathcal{B}}$ as a whole, we can interpret it as a 3-stage simulator. Θ_H and Θ_G are given as input in the second and third stage, but the order is not determined. We reformulate this simulator to make the order explicit with an indicator bit b.

Specifically, we view the entire procedure of $\mathcal{S}_2^{\mathcal{B}}$ as a three-stage simulator $\mathcal{S} = (\mathcal{S}^1, \mathcal{S}^2, \mathcal{S}^3)$.

Stage $\mathcal{S}_1$ chooses a bit $b \in \{0,1\}$ that fixes the output order and then outputs

$$x_0 \stackrel{\text{def}}{=} \begin{cases} x_H & \text{if } b = 0, \\ x_G & \text{if } b = 1. \end{cases}$$

Stage $\mathcal{S}_2$ is given the matching reprogrammed value of x_0 (i.e., Θ_H if $b = 0$, and Θ_G if $b = 1$) as input and outputs the other element:

$$x_1 \stackrel{\text{def}}{=} \begin{cases} x_G & \text{if } b = 0, \\ x_H & \text{if } b = 1. \end{cases}$$

Stage $\mathcal{S}_3$ receives the remaining reprogrammed value (i.e., Θ_G if $b = 0$, and Θ_H if $b = 1$) and returns the final output (x_H, x_G, y'_G, z).

In our theorem, we require that the simulator reprograms the oracle H before reprogramming the oracle G. This is equivalent to requiring $b = 0$ in the above reformulation.

By the total probability theorem, for $(x_H, x_G, y'_G, z) \leftarrow \langle \mathcal{S}, \Theta_H, \Theta_G \rangle$,

$$\begin{aligned} &\Pr[\, x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1] \\ =&\Pr[\, V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 \mid b = 0] \Pr[\, b = 0] \\ &\qquad + \Pr[\, V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 \mid b = 1] \Pr[\, b = 1] \end{aligned} \tag{1}$$

Note when $b = 1$, Θ_H is given to $\mathcal{S}$ after x_G is returned. In this case, to make $V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1$, we need to make $\Theta_H = x_G$. As Θ_H is sampled uniformly, independent of the already fixed x_G, we have that for $(x_H, x_G, y'_G, z) \leftarrow \langle \mathcal{S}, \Theta_H, \Theta_G \rangle$,

$$\begin{aligned} &\Pr[\, x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 \mid b = 1] \\ &\qquad \leq \Pr[\, x_H = x_H^\circ \wedge x_G = \Theta_H \mid b = 1] = \frac{\Pr[\, x_H = x_H^\circ \mid b = 1]}{|\mathcal{X}_1|} \end{aligned} \tag{2}$$

Finally, with (1) and (2), we conclude that for $(x_H, x_G, y'_G, z) \leftarrow \langle \mathcal{S}, \Theta_H, \Theta_G \rangle$,

$$\begin{aligned} &\Pr[\, x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 \wedge b = 0] \\ \geq&\Pr[\, x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1] \\ &\qquad - \Pr[\, V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 \mid b = 1] \Pr[\, b = 1] \\ \geq&\Pr[\, x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1] \\ &\qquad - |\mathcal{X}_1|^{-1} \Pr[\, x_H = x^\circ \mid b = 1] \end{aligned}$$

Finally, by the definition of V', we have that

$$\begin{aligned}
&\Pr\left[x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1 \wedge b = 0\right] \\
=&\Pr\left[x_H = x_H^\circ \wedge x_G = \Theta_H \wedge V(x_H, \Theta_H, \Theta_G, z) = 1 \wedge b = 0\right] \\
\leq&\Pr\left[x_H = x_H^\circ \wedge V(x_H, \Theta_H, \Theta_G, z) = 1 \wedge b = 0\right]
\end{aligned}$$

Now, we define a new simulator $\mathcal{S}'$ that follows the same procedure as $\mathcal{S}$ but aborts immediately when $b = 1$ and packs $x_G,\ y'_G$ into z. With $\mathcal{S}'$, we finally obtain the conclusion,

$$\begin{aligned}
&\Pr\left[x_H = x_H^\circ \wedge V(x_H, \Theta_H, \Theta_G, z) = 1 : (x_H, z) \leftarrow \langle \mathcal{S}', \Theta_H, \Theta_G\rangle\right] \\
=&\Pr\left[x_H = x_H^\circ \wedge V(x_H, \Theta_H, \Theta_G, z) = 1 \wedge b = 0 : (x_H, z) \leftarrow \langle \mathcal{S}', \Theta_H, \Theta_G\rangle\right] \\
\geq&\Pr\left[x_H = x_H^\circ \wedge V'(x_H, x_G, \Theta_H, \Theta_G, z) = 1\right] - |\mathcal{X}_1|^{-1} \Pr\left[x_H = x^\circ \mid b = 1\right] \\
\geq&\tfrac{1}{(2q_G+1)^2(2q_H+1)^2} \Pr\left[x_H = x_H^\circ \wedge V(x_H, h^{(x_H)}, g^{(x_H)}, z) = 1 : (x_H, z) \leftarrow \mathcal{A}^{H,G}\right] \\
&\qquad - |\mathcal{X}_1|^{-1} \Pr\left[x_H = x_H^\circ \mid b = 1\right]
\end{aligned}$$

This completes the proof of Theorem 3. □

3 Extended Adaptive Reprogramming

In this section, we extend the tight adaptive reprogramming theorem in [11] to handle two random oracles.

We first recall the tight adaptive reprogramming theorem in [11].

Theorem 4 (Adaptive Reprogramming [11]). *Let $\mathcal{X}, \mathcal{Y}$ be finite sets, and D be a distinguisher in the game defined in Fig. 2. Let D query the reprogram oracle R times and let q_r denote the number of queries to the random oracle $|\mathsf{O}_b\rangle$ between the $(r-1)$-th and r-th query to the reprogram oracle. By $p^{(r)}$ we denote the distribution p in the input of the r-th query to the reprogram oracle. Furthermore, by $p^{(r)}_{\mathcal{X}}$ we denote the marginal distribution of x, according to $p^{(r)}$. We also define,*

$$p^r_{\max} := \mathrm{E} \max_x p^{(r)}_{\mathcal{X}}(x),$$

where the expectation is taken over the behavior of D until its r-th query to the reprogram oracle. Then,

$$\left|\Pr\left[\mathsf{REPRO}^{\mathsf{D}}_1 = 1\right] - \Pr\left[\mathsf{REPRO}^{\mathsf{D}}_0 = 1\right]\right| \leq \sum_{r=1}^{R} \left(\sqrt{\hat{q_r} p^{(r)}_{\max}} + \frac{1}{2}\hat{q_r} p^{(r)}_{\max}\right),$$

where $\hat{q_r} = \sum_{i=1}^{r} q_i$.

Our goal is to extend Theorem 4 to support reprogramming two oracles successively, in which case the query for one oracle depends on the output of the other oracle. The theorem is stated as follows:

$\underline{\text{GAME } \mathsf{REPRO}_b^{\mathsf{D}}}$	$\underline{\mathsf{REPROGRAM}(p)}$
01: $\mathsf{O}_0 \leftarrow \mathcal{Y}^{\mathcal{X}}$	05: $(x, x') \leftarrow p$
02: $\mathsf{O}_1 := \mathsf{O}_0$	06: $y \leftarrow \mathcal{Y}$
03: $b' \leftarrow \mathsf{D}_b^{\lvert\mathsf{O}_b\rangle,\mathsf{REPROGRAM}}$	07: $\mathsf{O}_1 := \mathsf{O}_1^{x \to y}$
04: Return b'	08: Return (x, x')

Fig. 2. The adaptive reprogramming game.

Theorem 5 (Adaptive Reprogramming, extended). *Let $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ be finite sets with $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2$, and D be a distinguisher in the game defined in Fig. 3. Let D query the reprogram oracle R times and let q_{H_r} and q_{G_r} denote the number of queries to the random oracles $\lvert\mathsf{O}_{H_b}\rangle$ and $\lvert\mathsf{O}_{G_b}\rangle$, respectively, between the $(r-1)$-th and r-th query to the reprogram oracle.*

$$\left|\Pr\left[\mathsf{REPRO}_1^{\mathsf{D}} = 1\right] - \Pr\left[\mathsf{REPRO}_0^{\mathsf{D}} = 1\right]\right| \leq \sum_{r=1}^{R} \left(\sqrt{2\hat{q}_{H_r}|\mathcal{X}_2|^{-1}} + \sqrt{2\hat{q}_{G_r}|\mathcal{Y}|^{-1}}\right)$$

where $\hat{q}_{H_r} = \sum_{i=1}^{r} q_{H_i}$ and $\hat{q}_{G_r} = \sum_{i=1}^{r} q_{G_i}$.

$\underline{\text{GAME } \mathsf{REPRO}_b^{\mathsf{D}}}$	$\underline{\mathsf{REPROGRAM}(x_1)}$
01: $\mathsf{O}_{H_0} \leftarrow \mathcal{Y}^{\mathcal{X}}$	07: $x_2 \leftarrow \mathcal{X}_2$
02: $\mathsf{O}_{G_0} \leftarrow \mathcal{Z}^{\mathcal{Y}}$	08: $y \leftarrow \mathcal{Y}$
03: $\mathsf{O}_{H_1} := \mathsf{O}_{H_0}$	09: $z \leftarrow \mathcal{Z}$
04: $\mathsf{O}_{G_1} := \mathsf{O}_{G_0}$	10: $\mathsf{O}_{H_1} := \mathsf{O}_{H_1}^{(x_1,x_2) \to y}$
05: $b' \leftarrow \mathsf{D}_b^{\lvert\mathsf{O}_{H_b}\rangle,\lvert\mathsf{O}_{G_b}\rangle,\mathsf{REPROGRAM}}$	11: $\mathsf{O}_{G_1} := \mathsf{O}_{G_1}^{y \to z}$
06: Return b'	12: Return x_2

Fig. 3. The adaptive reprogramming game for reprogramming two oracles successively.

Remark 1 (Reprogramming the random oracle as a man-in-the-middle). Although we write the theorem as if the challenger has full control over the random oracle, it is worth noting that it doesn't have to own the random oracle to do the reprogramming. In fact, the challenger as a man in the middle between the adversary and the oracle is already enough to reprogram the random oracle.

Specifically, let H be the original random function. When reprogramming H to $H^{x \to y}$, the challenger can query the original random oracle H on x, and keep the classical $\Delta y_x \stackrel{\text{def}}{=} y \oplus H(x)$ locally. Then the challenger constructs a quantum gate, denoted by U_{Δ_x}. When it acts on a state $|x'\rangle|y'\rangle$ for $x' \in \mathcal{X} \setminus \{x\}$, it does nothing. When it acts on the state $|x\rangle|y'\rangle$, it turns $|x\rangle|y'\rangle$ to $|x\rangle|y' \oplus \Delta y_x\rangle$.

Upon an RO query from the adversary after the reprogramming, whose input is $|x'\rangle_X|y'\rangle_Y$, the challenger first applies the x-controlled gate U_{Δ_x}, then, it sends $\mathsf{U}_{\Delta_x}|x'\rangle_X|y'\rangle_Y$ to the original random oracle H. It is easy to verify that the resulting transform is identical to an ordinary reprogrammed oracle.

In this section, we will provide a proof for Theorem 5 with a new lemma, the superposition-controlled reprogramming lemma (cf. Lemma 2). In some sense, this lemma is an adaptation of the tight adaptive reprogramming theorem in [11] to allow the reprogramming controlled by a quantum superposition state, instead of a classical variable from a distribution.

Before giving the proof, we will first show why Theorem 4 (i.e., the original tight adaptive reprogramming theorem) cannot be directly extended to meet our demand of adaptively reprogramming two oracles successively, even if we use it in a white-box way.

3.1 Limitations of the tight adaptive reprogramming theorem [11]

At first glance, the formulation of Theorem 5 is very similar to the statement of Theorem 4, except that we have to reprogram two random oracles successively. So we initially expected that applying their theorem twice to reprogram the two oracles one by one would be enough to prove our theorem. Specifically, we may first use their theorem to reprogram H oracle, with the required min-entropy from $x_2 \leftarrow \mathcal{X}_2$. Then, we may use the theorem again to reprogram G oracle, with the required min-entropy from $y := H(x_1, x_2)$.

However, such an approach is not correct. When using their theorem statement literally to reprogram oracle G, we have to treat the description of oracle H as a part of the distinguisher's view to match the theorem's description. In this case, we cannot conclude that the value of $H(m\|r)$ (which is y sampled in the previous reprogram procedure of H in this case) is statistically uniform, as the distinguisher can access the oracle H at will.

We then try to open the black-box and use the superposition oracle to describe the random oracle's behavior, attempting to follow their proof strategy to improve the theorem. Their proof includes a crucial step, which is to take the expectation of some metrics of the entire system (including the internal states of the random oracle to be reprogrammed) over the input distribution p. During their proof, the variable x from $(x, x') \leftarrow p$ is treated as a fresh random variable outside of the system.

However, in our use of the theorem, our desired entropy source $y := H(x_1, x_2)$ when reprogramming the oracle G is in fact a part of the system. If we follow their proof strategy, we will still have to show how this coupling between y and the system affects the theorem's analysis. If the reprogramming process becomes more complicated, such coupling will become more difficult to analyze.

At this point, we realized that the original tight adaptive reprogramming theorem formulation is hard to extend to a case with a more complicated reprogramming process. Therefore, we decided to re-prove their theorem with a more general formulation.

Our approach to dealing with the issue is rooted in the observation that the theorem requires unconditional min-entropy (or, min-entropy from the view of the entire system) rather than min-entropy in some party's view. This requirement is not so desirable in the presence of multiple oracles. Therefore, we take

advantage of quantum computation to keep the random variables in their quantum superposition, preserving their entropy, instead of using classical random values as in [11]. To do this, we develop a more fundamental lemma, Lemma 2, which can reprogram the oracle given an appropriate superposition state which controls the position to be reprogrammed, instead of a classical distribution.

3.2 Superposition-controlled reprogramming

Let $\mathcal{A}$ be a quantum algorithm with access to a superposition random oracle $|H\rangle_F$, where $H : \mathcal{X} \to \mathcal{Y}$. Let X and Z denote two working registers of $\mathcal{A}$, where X contains a superposition over $\mathcal{X}$ and Z contains the remaining workspace state. Let q denote the number of $\mathcal{A}$'s queries to the random oracle. Let the system state after $\mathcal{A}$ terminates be $|\gamma_0\rangle_{XZF}$.

We consider the following quantum gate RE acting on $|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W$:

$$\mathsf{RE} = \sum_{x\in\mathcal{X}} |x\rangle\langle x|_X \otimes \mathsf{SWAP}_{F_x,W}.$$

Intuitively, RE reprograms F_x with a fresh uniformly random state by swapping F_x and the new register W, where the reprogrammed position x is controlled by the register X.

Then we have the following lemma:

Lemma 2 (Superposition-controlled reprogramming). *For any $\mathcal{A}$ that makes q queries to the superposition oracle and outputs $|\gamma_0\rangle_{XZF}$ such that X is separable from ZF, i.e., $|\gamma_0\rangle_{XZF} = |x\rangle_X \otimes |\gamma\rangle_{ZF}$, and for any (unbounded) quantum distinguisher D,*

$$\left|\Pr\left[\,\mathsf{D}(|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W) = 1\,\right] - \Pr\left[\,\mathsf{D}\big(\mathsf{RE}\,(|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W)\,\big) = 1\,\right]\right| \le \sqrt{2q\alpha_{\max}^2},$$

$\alpha_{\max} = \max_{x'\in\mathcal{X}} |\langle x'|x\rangle_X|$ and $|\phi_0\rangle_W$ is the uniform superposition as defined in Subsect. 2.3.

Proof. We prove the lemma by computing the fidelity between the states before and after the reprogramming. We first write the state in a form that makes the action of RE explicit.

We can represent the state $|\gamma\rangle_{XZF}$ as a sum of tensor-product states with amplitudes $\{\alpha_x\}_{x\in\mathcal{X}}$:

$$\begin{aligned}|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W &= \sum_{x\in\mathcal{X}} \alpha_x |x\rangle_X \otimes |\gamma\rangle_{ZF} \otimes |\phi_0\rangle_W \\ &= \sum_{x\in\mathcal{X}} \alpha_x |x\rangle_X \otimes \sum_{\substack{S\subset\mathcal{X}\\ |S|\le q}} \beta_S \left|\gamma^{(S)}\right\rangle_{ZF_S} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} \otimes |\phi_0\rangle_W .\end{aligned}$$

The decomposition over registers Z and F in the second equation is given by Lemma 1, where we recall that $\left|\gamma^{(S)}\right\rangle_{ZF_S}$ satisfies $\langle\phi_0|_{F_x} \left|\gamma^{(S)}\right\rangle_{ZF_S} = 0$ for all

$x \in S$. To make the effect of swapping explicit, we further expand the state in the Fourier basis $\left\{\phi_0, \phi_1, \cdots, \phi_{|\mathcal{Y}|-1}\right\}$ of register F_x.

$$|\psi\rangle \stackrel{\text{def}}{=} |\gamma\rangle_{XZF} \otimes |\phi_0\rangle_W = \sum_{x\in\mathcal{X}} \alpha_x |x\rangle_X \otimes$$
$$\Bigg(\underbrace{\sum_{\substack{S\subset\mathcal{X} \\ |S|\leq q,\, x\in S}} \beta_S \sum_{i=1}^{|\mathcal{Y}|-1} |\phi_i\rangle\langle\phi_i|_{F_x} \left|\gamma^{(S)}\right\rangle_{ZF_S} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} \otimes |\phi_0\rangle_W}_{\stackrel{\text{def}}{=}|\psi_{x,0}\rangle_{ZFW}} +$$
$$\underbrace{\sum_{\substack{S\subset\mathcal{X} \\ |S|\leq q,\, x\notin S}} \beta_S \left|\gamma^{(S)}\right\rangle_{ZF_S} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} \otimes |\phi_0\rangle_W}_{\stackrel{\text{def}}{=}|\psi_{x,1}\rangle_{ZFW}} \Bigg).$$

We note that the summation over i starts from 1 instead of 0 because the $i = 0$ term vanishes due to the property $\langle\phi_0|_{F_x} \left|\gamma^{(S)}\right\rangle_{ZF_S} = 0$ for all $x \in S$.

Now we apply RE to the state and get

$$|\psi'\rangle \stackrel{\text{def}}{=} \mathsf{RE}\left(|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W\right) = \sum_{x\in\mathcal{X}} \alpha_x |x\rangle_X \otimes$$
$$\Bigg(\underbrace{\sum_{\substack{S\subset\mathcal{X} \\ |S|\leq q, x\in S}} \beta_S \sum_{i=1}^{|\mathcal{Y}|-1} |\phi_0\rangle\langle\phi_i|_{F_x} \left|\gamma^{(S)}\right\rangle_{ZF_S} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} \otimes |\phi_i\rangle_W}_{\stackrel{\text{def}}{=}\left|\psi'_{x,0}\right\rangle_{ZFW}} +$$
$$\underbrace{\sum_{\substack{S\subset\mathcal{X} \\ |S|\leq q, x\notin S}} \beta_S \left|\gamma^{(S)}\right\rangle_{ZF_S} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} \otimes |\phi_0\rangle_W}_{\stackrel{\text{def}}{=}\left|\psi'_{x,1}\right\rangle_{ZFW}} \Bigg).$$

Here we define some new notations $|\psi\rangle, |\psi'\rangle$ and $|\psi_{x,0}\rangle, |\psi_{x,1}\rangle, \left|\psi'_{x,0}\right\rangle, \left|\psi'_{x,1}\right\rangle$ in the equation above to make our following presentation more compact. We further denote $\langle\phi_i|_{F_x} \left|\gamma^{(S)}\right\rangle_{ZF_S}$ as $\left|\gamma^{(S)}_{x,i}\right\rangle_{ZF_{S\setminus x}}$, so that we can write $|\psi_{x,0}\rangle$ and $\left|\psi'_{x,0}\right\rangle$ as

$$|\psi_{x,0}\rangle = \sum_{\substack{S\subset\mathcal{Y} \\ |S|\leq q, x\in S}} \beta_S \sum_{i=1}^{|\mathcal{Y}|-1} |\phi_i\rangle_{F_x} \left|\gamma^{(S)}_{x,i}\right\rangle_{ZF_{S\setminus x}} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} |\phi_0\rangle_W$$

$$\left|\psi'_{x,0}\right\rangle = \sum_{\substack{S\subset\mathcal{X} \\ |S|\leq q, x\in S}} \beta_S \sum_{i=1}^{|\mathcal{Y}|-1} |\phi_0\rangle_{F_x} \left|\gamma^{(S)}_{x,i}\right\rangle_{ZF_{S\setminus x}} \left(|\phi_0\rangle^{\otimes|\mathcal{X}\setminus S|}\right)_{F_{\mathcal{X}\setminus S}} |\phi_i\rangle_W$$

With these notations and the explicit representation of the state vectors, we are now getting into computing the fidelity of the two states.

$$F(|\psi\rangle\langle\psi|, |\psi'\rangle\langle\psi'|) = |\langle\psi|\psi'\rangle| = \left| \sum_{x\in\mathcal{X}} |\alpha_x|^2 \langle x|x\rangle_X \cdot (\langle\psi_{x,0}| + \langle\psi_{x,1}|) \left(\left|\psi'_{x,0}\right\rangle + \left|\psi'_{x,1}\right\rangle\right) \right|$$
$$= \left| \sum_{x\in\mathcal{X}} |\alpha_x|^2 \left\langle\psi_{x,1}\middle|\psi'_{x,1}\right\rangle \right|. \tag{3}$$

We omit the vanishing terms $\left\langle\psi_{x,0}\middle|\psi'_{x,1}\right\rangle$, $\left\langle\psi_{x,1}\middle|\psi'_{x,0}\right\rangle$ and $\left\langle\psi_{x,0}\middle|\psi'_{x,0}\right\rangle$ in (3). We show that the inner product in one of the registers is zero for the vanishing terms. For $\left\langle\psi_{x,0}\middle|\psi'_{x,1}\right\rangle$, we have $\langle\psi_{x,0}|_{ZFW} |\phi_0\rangle_{F_x} = 0$, in which the projection of $\psi'_{x,1}$ onto the register F_x is $|\phi_0\rangle_{F_x}$. Similarly, we have $\langle\phi_0|_W \left|\psi'_{x,0}\right\rangle_{ZFW} = 0$, which indicates $\left\langle\psi_{x,1}\middle|\psi'_{x,0}\right\rangle = 0$ and $\left\langle\psi_{x,0}\middle|\psi'_{x,0}\right\rangle = 0$.

For the leftover term , we have

$$F(|\psi\rangle\langle\psi|, |\psi'\rangle\langle\psi'|) = \left| \sum_{x\in\mathcal{X}} |\alpha_x|^2 \left\langle\psi_{x,1}\middle|\psi'_{x,1}\right\rangle \right| = \sum_{x\in\mathcal{X}} |\alpha_x|^2 \sum_{\substack{S\subset\mathcal{X} \\ |S|\le q, x\in S}} |\beta_S|^2$$
$$= \sum_{\substack{S\subset\mathcal{X} \\ |S|\le q}} |\beta_S|^2 \sum_{x\in\mathcal{X}\setminus S} |\alpha_x|^2 \ge \sum_{\substack{S\subset\mathcal{X} \\ |S|\le q}} |\beta_S|^2 \cdot (1 - q\alpha_{\max}^2) = 1 - q\alpha_{\max}^2.$$

Finally, we can bound the probability with fidelity by

$$\left|\Pr\left[\,\mathsf{D}(|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W) = 1\,\right] - \Pr\left[\,\mathsf{D}\big(\mathsf{RE}\,(|\gamma_0\rangle_{XZF} \otimes |\phi_0\rangle_W)\,\big) = 1\,\right]\right|$$
$$= D(|\psi\rangle\langle\psi|, |\psi'\rangle\langle\psi'|) \le \sqrt{1 - F(|\psi\rangle\langle\psi|, |\psi'\rangle\langle\psi'|)^2} \le \sqrt{1 - (1 - q\alpha_{\max}^2)^2}$$
$$= \sqrt{2q\alpha_{\max}^2 - q^2\alpha_{\max}^4} \le \sqrt{2q\alpha_{\max}^2},$$

where the (in)equalities in the second line are from the properties of the two metrics mentioned in Subsect. 2.3.

This completes the proof of Lemma 2. □

With Lemma 2, we can restate Theorem 4 as follows:

Theorem 6 (Adaptive Reprogramming [11, Theorem 1], restated). *Let the symbols be the same as defined in Theorem 4. Then, for the game defined in Fig. 2,*

$$\left|\Pr\left[\,\mathsf{REPRO}_1^{\mathsf{D}} = 1\,\right] - \Pr\left[\,\mathsf{REPRO}_0^{\mathsf{D}} = 1\,\right]\right| \le \sum_{r=1}^{R} \sqrt{2\hat{q}_r p_{\max}^{(r)}}.$$

Comparing Theorem 6 with Theorem 4 (i.e. the original theorem in [11]), our bound does not include the linear term $\hat{q_r} p_{\max}^{(r)}$ but has a larger factor up to $\sqrt{2}$ times in the leading square root term $\sqrt{\hat{q_r} p_{\max}^{(r)}}$. This might be because our lemma (Lemma 2) allows the distinguisher to be given the quantum superposition $|x\rangle_X$, whereas [11] requires a measured classical x. Nevertheless, using our theorem does not increase the security loss bound in the main application results (i.e., Proposition 1 and Proposition 2 of [11]), and our result still provides a tight bound up to constant factors.

As the proof shares the same idea with that of Theorem 5 and the conclusion is similar to the original theorem in [11], we omit the proof here.

3.3 Proof of Extended Adaptive Reprogramming Theorem

Now we present the proof of Theorem 5. The idea of the proof is to simulate all the classical random sampling in the game with quantum circuits, and then apply Lemma 2 to each step.

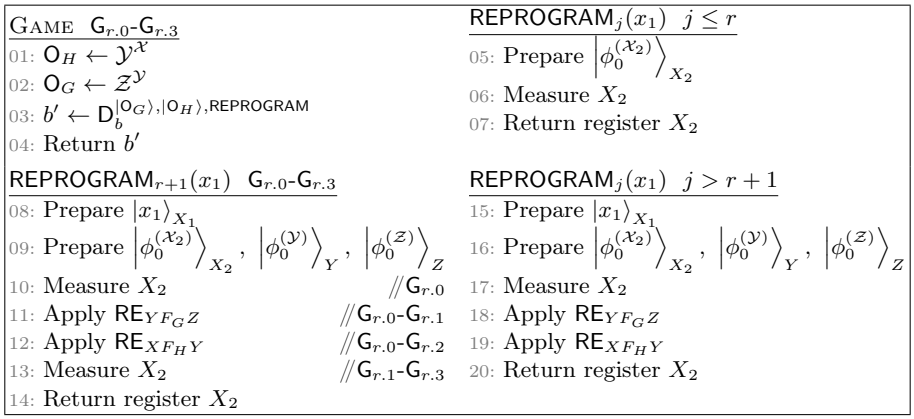


Fig. 4. The games $\mathsf{G}_{r.0}$ to $\mathsf{G}_{r.3}$ for the proof of Theorem 5, where $r = 0, 1, 2, \cdots, R-1$, and $\mathsf{REPROGRAM}_j$ denotes the j-th query to the $\mathsf{REPROGRAM}$ oracle.

Proof (Theorem 5). Following the proof of the original theorem in [11], we start with the superposition oracle version of the game in Fig. 3. We let G_0 be the superposition oracle version of the original game $\mathsf{REPRO}_1^{\mathsf{D}}$. For those familiar with superposition oracle, it is easy to see that the game is identical to the game $\mathsf{REPRO}_1^{\mathsf{D}}$ in Fig. 3, so we omit the details of game transition to G_0 here. For completeness, we provide the details of game transition to G_0 in Appendix A.

Now we use game transitions to prove the theorem. For $r = 1, 2, \cdots, R$, we let game G_r proceed the same as G_0, except that for the first r queries to the $\mathsf{REPROGRAM}$ oracle, it only returns a fresh uniform random $x_r \in \mathcal{X}_1$ and does nothing else (i.e., does not reprogram the oracles), as shown in Fig. 4.

We see that for $r = 0, 1, \cdots, R-1$, the difference between G_r and G_{r+1} lies only in the $(r+1)$-th query to the $\mathsf{REPROGRAM}$ oracle. We now use another sequence of games, from $\mathsf{G}_{r.0}$ to $\mathsf{G}_{r.3}$, to bound the distinguishability between the game G_r and G_{r+1}.

Game $\mathsf{G}_{r.0}$. This game is the same as G_r. We only define this game for consistency of the notation.

Game $\mathsf{G}_{r.1}$. In this game, the measurement of X_2 in the $(r+1)$-th query to the $\mathsf{REPROGRAM}$ oracle is moved to the end of the reprogramming function. We note that in the operator RE_{XF_HY}, register X is only used as control bits, thus RE_{XF_HY} commutes with the measurement of X_2. This gives us $\Pr_{r.1} = \Pr_{r.0}$.

Game $\mathsf{G}_{r.2}$. In this game, we do not apply RE_{YF_GZ} anymore. We note that before RE_{YF_GZ} is applied in Game G_3, the register Y is separable from the other part of the system. To apply Lemma 2, we need to ensure that the internal register F_G is accessed only through the oracle queries.

Luckily this is the case, as the $\mathsf{REPROGRAM}$ oracle queries prior to the $(r+1)$-th query now do nothing to the oracle G due to the previous game transitions. Moreover, prior to the $(r+1)$-th oracle query to $\mathsf{REPROGRAM}$ we see that there are only $\hat{q}_{G_{r+1}}$ queries to the oracle G. Finally, recalling that register Y is in the state $\left|\phi_0^{(\mathcal{Y})}\right\rangle_Y = \sum_{y\in\mathcal{Y}} \sqrt{|\mathcal{Y}|^{-1}}\,|y\rangle_Y$, we conclude from Lemma 2 that, $|\Pr_{r.2} - \Pr_{r.1}| \leq \sqrt{2\hat{q}_{G_{r+1}}|\mathcal{Y}|^{-1}}$.

Game $\mathsf{G}_{r.3}$. In this game, we do not apply RE_{XF_HY} anymore. Similarly, before RE_{XF_HY} is applied in Game G_4, the register X is separable from the other part of the system and X is in the state $|x_1\rangle_{X_1}\left|\phi_0^{(\mathcal{X}_2)}\right\rangle_{X_2} = |x_1\rangle_{X_1} \otimes \sum_{x_2\in\mathcal{X}_2} \sqrt{|\mathcal{X}_2|^{-1}}\,|x_2\rangle_{X_2}$. Moreover, prior to the r-th oracle query to $\mathsf{REPROGRAM}$, the oracle H is accessed only through oracle queries (instead of accessing its internal states directly), and there are only $\hat{q}_{H_{r+1}}$ such queries to H. We conclude from Lemma 2 that, $|\Pr_{r.3} - \Pr_{r.2}| \leq \sqrt{2\hat{q}_{H_{r+1}}|\mathcal{X}_2|^{-1}}$.

At the end of this game sequence, we delete the unused registers in the game $\mathsf{G}_{r.3}$, and find that the game is identical to the game G_{r+1}. This shows that,

$$|\Pr_{r+1} - \Pr_r| = |\Pr_{r.3} - \Pr_{r.0}| \leq \sqrt{2\hat{q}_{H_{r+1}}|\mathcal{X}_2|^{-1}} + \sqrt{2\hat{q}_{G_{r+1}}|\mathcal{Y}|^{-1}}.$$

Now, consider the final game G_R, in which none of the $\mathsf{REPROGRAM}$ calls reprogram the random oracles H or G anymore. That is to say that G_R is identical to the game REPRO_0, where the distinguisher D is given random oracles that are never reprogrammed. Therefore, we sum up the previous results and conclude that,

$$\begin{aligned}\left|\Pr\left[\mathsf{REPRO}_1^{\mathsf{D}} = 1\right] - \Pr\left[\mathsf{REPRO}_0^{\mathsf{D}} = 1\right]\right| &\leq \sum_{r=0}^{R-1} |\Pr_{r+1} - \Pr_r| \\ &\leq \sum_{r=1}^{R} \left(\sqrt{2\hat{q}_{H_r}|\mathcal{X}_2|^{-1}} + \sqrt{2\hat{q}_{G_r}|\mathcal{Y}|^{-1}}\right).\end{aligned}$$

This completes the proof of Theorem 5. □

4 Generic Construction of Signature Scheme with Message Recovery

In this section, we present a generic construction of signature schemes with message recovery from preimage sampleable function, and prove its EUF-CMA security in the QROM. This scheme is a post-quantum adaptation of the PSS-R scheme proposed by Bellare and Rogaway [3].

Let $\mathcal{M} = \mathcal{M}_1 \times \mathcal{M}_2$ be the message space. We require the following building blocks to construct the scheme:

- A preimage sampleable function $\mathsf{PSF} = (\mathsf{TrapGen}, \mathsf{SampleDom}, \mathsf{SamplePre})$ with a preimage space $\mathcal{X}$ and an image space $\mathcal{Y} = \mathcal{Y}_1 \times \mathcal{Y}_2 \times \mathcal{Y}_3$, where $\mathcal{Y}_2 = \{0,1\}^{l_2} \supseteq \mathcal{M}_1$ and $\mathcal{Y}_3 = \{0,1\}^{l_3}$ for some positive integers $l_2, l_3 \in \mathbb{N}$.
- Two hash functions $H : \mathcal{M}_1 \times \mathcal{M}_2 \times \mathcal{Y}_3 \to \mathcal{Y}_1$ and $G : \mathcal{Y}_1 \to \mathcal{Y}_2 \times \mathcal{Y}_3$, where G can be further partitioned into two hash functions $G_1 : \mathcal{Y}_1 \to \mathcal{Y}_2$ and $G_2 : \mathcal{Y}_1 \to \mathcal{Y}_3$ such that $G(\omega) = G_1(\omega) \| G_2(\omega)$ holds for all $\omega \in \mathcal{Y}_1$.

With the above building blocks, we present the signature scheme with message recovery in Fig. 5. It is straightforward to check the correctness of the signature scheme based on the preimage sampling correctness of PSF.

$\underline{\mathsf{KeyGen}(1^\lambda)}$:	$\underline{\mathsf{Sign}(sk = t, m = (m_1, m_2))}$:	$\underline{\mathsf{Vrfy}(vk = a, m_2, \sigma)}$:
$(a, t) \leftarrow \mathsf{TrapGen}(1^\lambda)$	$r \leftarrow \mathcal{Y}_3$	$\omega' \| M' \| R' := f_a(\sigma)$
$vk := a$	$\omega := H(m_1 \| m_2 \| r)$	$m_1' := M' \oplus G_1(\omega')$
$sk := t$	$M := m_1 \oplus G_1(\omega)$	$r' := R' \oplus G_2(\omega')$
Return (vk, sk)	$R := r \oplus G_2(\omega)$	If $H(m_1' \| m_2 \| r') = \omega'$ then
	$\sigma \leftarrow \mathsf{SamplePre}(\omega \| M \| R, t)$	return $(1, m_1')$
	Return σ	else return $(0, \perp)$

Fig. 5. Generic construction of signature scheme with message recovery from preimage sampleable function.

Theorem 7 (EUF-CMA security). *If* PSF *is a collision-resistant preimage sampleable function and H, G are modeled as quantum accessible random oracles, then the signature scheme in Fig. 5 is* EUF-CMA *secure. Specifically, for any PPT adversary $\mathcal{A}$ against the* EUF-CMA *security of the scheme that makes q_H and q_G queries to the oracles H and G, respectively, and makes q_S queries to the signing oracle* Sign*, there exists a PPT adversary $\mathcal{B}$ against the collision-resistance of* PSF*, such that*

$$\mathsf{Adv}^{\mathsf{CR}}_{\mathsf{PSF},\mathcal{B}} \geq \frac{1-p_{max}}{(2\max\{q_H, q_G+1\}+1)^4} \cdot \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}} - \frac{(1-p_{max})q_S(q_H+q_S+2)^2}{(2\max\{q_H, q_G+1\}+1)^4|\mathcal{Y}_1|} - \frac{(1-p_{max})q_S}{(2\max\{q_H, q_G+1\}+1)^4}\left(\sqrt{\frac{2q_H}{|\mathcal{Y}_3|}} + \sqrt{\frac{2q_G}{|\mathcal{Y}_1|}}\right) - \frac{1-p_{max}}{|\mathcal{Y}_1|},$$

where p_{max} is the preimage min-entropy parameter of PSF *defined in Subsect. 2.2.*

To prove this theorem, we break it down into two lemmas. First, in Lemma 3, we show the reduction from EUF-NMA to EUF-CMA by using our extended adaptive reprogramming theorem (i.e., Theorem 5). Then in Lemma 4, we prove the EUF-NMA security of the scheme based on the collision-resistance of PSF by using our extended measure-and-reprogram theorem (i.e., Theorem 3).

Lemma 3 (EUF-NMA **to** EUF-CMA)**.** *For any PPT adversary $\mathcal{A}$ against the* EUF-CMA *security of the scheme in Fig. 5, there exists a PPT adversary $\mathcal{B}$ against the* EUF-NMA *security, such that*

$$\left|\mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{B}} - \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}}\right| \leq q_S\left(\sqrt{2q_H|\mathcal{Y}_3|^{-1}} + \sqrt{2q_G|\mathcal{Y}_1|^{-1}}\right) + \frac{q_S(q_H+q_S+2)^2}{|\mathcal{Y}_1|}.$$

Proof. We use game transitions to prove the lemma. See Fig. 6 for the description of the games.

		$\underline{\textsc{Sign}(m=(m_1,m_2))\quad \mathsf{G}_0\text{-}\mathsf{G}_5}$	
$\underline{\textsc{Game}\ \mathsf{G}_0\text{-}\mathsf{G}_5}$		11: $r \leftarrow \mathcal{Y}_3$	
01: $\mathcal{S} := \emptyset$		12: $\sigma^* \leftarrow \mathsf{SampleDom}(1^\lambda)$	$/\!/\mathsf{G}_4\text{-}\mathsf{G}_5$
02: $\mathcal{W} := \emptyset$	$/\!/\mathsf{G}_3\text{-}\mathsf{G}_5$	13: $\omega^*\|M^*\|R^* := f_a(\sigma^*)$	$/\!/\mathsf{G}_4\text{-}\mathsf{G}_5$
03: $(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$		14: $\omega^* \leftarrow \mathcal{Y}_1$	$/\!/\mathsf{G}_1\text{-}\mathsf{G}_3$
04: $(m_2', \sigma') \leftarrow \mathcal{A}^{\textsc{Sign}(\cdot),H,G}(vk)$		15: $M^* \leftarrow \mathcal{Y}_2,\ R^* \leftarrow \mathcal{Y}_3$	$/\!/\mathsf{G}_1\text{-}\mathsf{G}_3$
05: $\omega'\|M'\|R' := f_a(\sigma^*)$		16: $H := H^{m_1\|m_2\|r\to\omega^*}$	$/\!/\mathsf{G}_1\text{-}\mathsf{G}_5$
06: $m_1' := M' \oplus G_1(\omega')$		17: $G := G^{\omega^*\to(M^*\oplus m_1)\|(R^*\oplus r)}$	$/\!/\mathsf{G}_1\text{-}\mathsf{G}_5$
07: $r' := R' \oplus G_2(\omega')$		18: $\omega := H(m_1\|m_2\|r)$	$/\!/\mathsf{G}_0\text{-}\mathsf{G}_1$
08: If $(m_1', m_2') \in \mathcal{S}$ then Return 0		19: $M := m_1 \oplus G_1(\omega)$	$/\!/\mathsf{G}_0\text{-}\mathsf{G}_1$
09: If $\omega' \in \mathcal{W}$ then Return 0	$/\!/\mathsf{G}_3\text{-}\mathsf{G}_5$	20: $R := r \oplus G_2(\omega)$	$/\!/\mathsf{G}_0\text{-}\mathsf{G}_1$
10: Return $\omega' = H(m_1'\|m_2'\|r')$		21: $\sigma \leftarrow \mathsf{SamplePre}(\omega\|M\|R, t)$	$/\!/\mathsf{G}_0\text{-}\mathsf{G}_1$
		22: $\sigma \leftarrow \mathsf{SamplePre}(\omega^*\|M^*\|R^*, t)$	$/\!/\mathsf{G}_2\text{-}\mathsf{G}_4$
		23: $\mathcal{S} := \mathcal{S} \cup \{(m_1, m_2)\}$	
		24: $\mathcal{W} := \mathcal{W} \cup \{\omega^*\}$	$/\!/\mathsf{G}_3\text{-}\mathsf{G}_5$
		25: Return σ	$/\!/\mathsf{G}_0\text{-}\mathsf{G}_4$
		26: Return σ^*	$/\!/\mathsf{G}_5$

Fig. 6. The game transition for the reduction from EUF-NMA to EUF-CMA (Lemma 3).

Game G_0. This is the original EUF-CMA game for signature scheme with message recovery (cf. Fig. 1).

Game G_1. This game reprograms the two random oracles H and G. To show the indistinguishability, we construct a simulator with Theorem 5 such that O_{H_b} and O_{G_b} are set to the oracles H and G respectively. Specifically, our simulator simulates the game G_0 except that we sample r using the REPROGRAM oracle, i.e., $r \leftarrow \mathsf{REPROGRAM}(m_1\|m_2)$. Clearly, if we are given O_{H_0} and O_{G_0}, this simulation is identical to the game G_0. If we are given O_{H_1} and O_{G_1}, this simulation is identical to the game G_1. Therefore, the indistinguishability follows from Theorem 5 that $|\Pr_1 - \Pr_0| \leq q_S\left(\sqrt{2q_H|\mathcal{Y}_3|^{-1}} + \sqrt{2q_G|\mathcal{Y}_1|^{-1}}\right)$.

Game G_2. In this game, we switch the first input to the SamplePre algorithm from $\omega\|M\|R$ to $\omega^*\|M^*\|R^*$, where the starred variables are the intermediate values sampled for reprogramming in G_1. In fact, as the oracles H and G are already reprogrammed, we can see that,

$$\begin{aligned}\omega &= H(m_1\|m_2\|r) = \omega^*,\\ M &= m_1 \oplus G_1(\omega) = m_1 \oplus M^* \oplus m_1 = M^*,\\ R &= r \oplus G_2(\omega) = r \oplus R^* \oplus r = R^*.\end{aligned}$$

This indicates that G_2 is identical to G_1, i.e., $\Pr_2 = \Pr_1$.

Game G_3. In this game, we add an aborting condition to the game, i.e., if ω' is equal to any of the ω^* sampled in the game, we abort the game and return 0. This game is identical to G_2 as long as the Bad event $\omega' \in \mathcal{W} \wedge (m_1', m_2') \notin \mathcal{S} \wedge \omega' = H(m_1'\|m_2'\|r')$ does not happen.

Let H° be the un-reprogrammed version of H, we will bound the probability of the Bad event with the optimality of Grover Search with random oracle H° as shown in Theorem 1. Consider an adversary $\mathcal{B}_s$ who obtains some $y \in \mathcal{Y}_1$, and is searching for a preimage x such that $H^\circ(x) = y$. Let $\mathcal{B}_s$ simulates the game G_3 with the adversary $\mathcal{A}$ in G_3, except that,

1. When $\mathcal{B}_s$ needs to reprogram the random oracle H, it does the reprogramming as a man-in-the-middle between H° and $\mathcal{A}$ following the method described in Remark 1.
2. $\mathcal{B}_s$ samples $t \leftarrow [q_S]$ and replaces ω^* with y in the t-th query to the Sign oracle.
3. $\mathcal{B}_s$ returns $m_1'\|m_2'\|r'$ directly before checking whether $(m_1', m_2') \in \mathcal{S}$.

As y is chosen uniformly, the view to $\mathcal{A}$ provided by $\mathcal{B}_s$ is identical to that of $\mathcal{A}$ in the real game of G_3. We then have that,

$$\begin{aligned}\Pr[\mathsf{Bad}] &= \Pr[\omega' \in \mathcal{W} \wedge (m_1', m_2') \notin \mathcal{S} \wedge \omega' = H(m_1'\|m_2'\|r')]\\ &\le \Pr[\omega' \in \mathcal{W} \wedge (m_1', m_2') \notin \mathcal{S} \wedge \omega' = H^\circ(m_1'\|m_2'\|r')] \qquad (4)\\ &= q_S \Pr[\omega' = y \wedge (m_1', m_2') \notin \mathcal{S} \wedge \omega' = H^\circ(m_1'\|m_2'\|r')] \qquad (5)\\ &\le q_S \Pr[\omega' = y \wedge \omega' = H^\circ(m_1'\|m_2'\|r')].\end{aligned}$$

The inequality (4) in the second line follows from that $m_1'\|m_2'\|r'$ is not a reprogrammed position of oracle H as $(m_1', m_2') \notin \mathcal{S}$.

For the equation (5) in the third line, we recall that one of the $\omega^* \in \mathcal{W}$ is replaced by an identically distributed element y, and the probability of each element to be replaced is equal. Therefore, there is $\frac{1}{q_S}$ probability that the resulting ω' happens to be y.

As $\mathcal{B}_s$ queries H° for $q_H + q_S$ times (q_H times from $\mathcal{A}$ and q_S times from the reprogramming process within Sign), by Theorem 1, we can bound its probability of finding such x' by,

$$\Pr[\omega' = y \wedge \omega' = H^\circ(x') : x' \leftarrow \mathcal{B}_s(y)] \le \frac{(q_H + q_S + 2)^2}{|\mathcal{Y}_1|}.$$

We finally conclude that,

$$|\Pr_3 - \Pr_2| \leq \Pr[\mathsf{Bad}] \leq \frac{q_S(q_H + q_S + 2)^2}{|\mathcal{Y}_1|}.$$

Game G_4. In this game, we no longer sample ω^*, M^*, and R^* directly. Instead, we sample σ^* from $\mathsf{SampleDom}(1^\lambda)$ and then parse ω^*, M^*, and R^* from $f_a(\sigma^*)$. By the image distribution property of PSF introduced in Subsect. 2.2, the distribution of $f_a(\sigma^*)$ in G_4 is uniform. Therefore, ω^*, M^*, and R^* in G_4 are perfectly indistinguishable from those sampled from a uniform distribution in G_3. This then concludes that $\Pr_4 = \Pr_3$.

Game G_5. In this game, we change the output of SIGN oracle from σ to σ^*. Due to the preimage sampling distribution property of PSF, the distribution of σ follows D_x conditioned on $f_a(\sigma) = \omega^*\|M^*\|R^*$, which is exactly the same distribution as the posterior distribution of σ^* when fixing $f_a(\sigma^*) = \omega^*\|M^*\|R^*$. This indicates that $(\sigma^*, \omega^*\|M^*\|R^*)$ in this game is perfectly indistinguishable from $(\sigma, \omega^*\|M^*\|R^*)$. This proves that $\Pr_5 = \Pr_4$.

We can now translate an adversary $\mathcal{A}$ in G_5 to an adversary $\mathcal{B}$ against the $\mathsf{EUF\text{-}NMA}$ security such that $\mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{B}} \geq \Pr_5$.

Specifically, denote the un-reprogrammed random oracles by H° and G°, let $\mathcal{B}$ work as follows:

- On input the verification key vk from the challenger $\mathcal{C}$ of the $\mathsf{EUF\text{-}NMA}$ game, $\mathcal{B}$ passes vk to $\mathcal{A}$.
- On $\mathcal{A}$'s query to the SIGN oracle, $\mathcal{B}$ simulates the SIGN oracle as in G_5, and passes the output to $\mathcal{A}$. This simulation contains the step of reprogramming the random oracles H and G. We note that, although $\mathcal{B}$ does not control the random oracles, it can still complete the reprogramming process as a man-in-the-middle between $\mathcal{C}$ and $\mathcal{A}$, as shown in Remark 1.
- On $\mathcal{A}$'s query to the random oracles H or G, $\mathcal{B}$ does as shown in Remark 1 to answer the query.
- On $\mathcal{A}$'s outputting the message-signature pair (m'_2, σ'), it outputs (m'_2, σ') to the challenger $\mathcal{C}$ of the $\mathsf{EUF\text{-}NMA}$ game.

We see that $\mathcal{B}$'s simulation provides a view to $\mathcal{A}$ that is identical to $\mathcal{A}$'s view in G_5, and we will show that the output of $\mathcal{B}$ is a valid signature as long as $\mathcal{A}$'s output wins the simulated G_5.

Consider the case where $\mathcal{A}$'s output wins the simulated G_5. As $(m'_1, m'_2) \notin \mathcal{S}$, we know immediately that for any $r \in \mathcal{Y}_3$, $m'_1\|m'_2\|r$ is not a reprogrammed position of oracle H. Moreover, as $\omega' \notin \mathcal{W}$, we also know ω' is not a reprogrammed position of oracle G. Therefore, $H(m'_1\|m'_2\|r') = H^\circ(m'_1\|m'_2\|r')$ and $G(\omega') = G^\circ(\omega')$. As (m'_2, σ') passes the signature verification with the queried values $H(m'_1\|m'_2\|r')$ and $G(\omega')$, it also passes the signature verification with the original random oracles H° and G°.

This shows that $\mathcal{B}$ forges a valid signature in the $\mathsf{EUF\text{-}NMA}$ game with probability at least $\mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{B}} \geq \Pr_5$.

Finally, with all the game transitions above, we conclude that,

$$\left|\mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{B}} - \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{SIG},\mathcal{A}}\right| \leq q_S\left(\sqrt{2q_H|\mathcal{Y}_3|^{-1}} + \sqrt{2q_G|\mathcal{Y}_1|^{-1}}\right) + \frac{q_S(q_H+q_S+2)^2}{|\mathcal{Y}_1|}.$$

This completes the proof of Lemma 3. □

Lemma 4 (EUF-NMA security). *For any PPT adversary $\mathcal{A}$ against the* EUF-NMA *security of the scheme that makes q_H and q_G queries to the oracles H and G, respectively, there exists a PPT adversary $\mathcal{B}$ against the collision resistance of* PSF*, such that*

$$\mathsf{Adv}^{\mathsf{CR}}_{\mathsf{PSF},\mathcal{B}} \geq \frac{1-p_{max}}{(2\max\{q_H,q_G+1\}+1)^4} \cdot \mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}} - \frac{1-p_{max}}{|\mathcal{Y}_1|}.$$

Proof. Let $\mathcal{A}$ be an adversary in the EUF-NMA game of the scheme in Fig. 5. To construct $\mathcal{B}$ against the collision resistance of PSF, we first construct an intermediate adversary $\mathcal{A}_+$ as shown in Fig. 7. On input the description of an instance a of PSF, $\mathcal{A}_+$ interprets a as the signature's vk, and outputs the signature σ^* with corresponding $x' := m'_1\|m'_2\|r'$ under the help of $\mathcal{A}$.

$\underline{\mathcal{A}_+(a)}$
01: $vk := a$
02: $(m'_2, \sigma') \leftarrow \mathcal{A}^{H,G}(vk)$
03: $\omega'\|M'\|R' := f_a(\sigma')$
04: $m'_1 := M' \oplus G_1(\omega')$
05: $r' := R' \oplus G_2(\omega')$
06: $x' := m'_1\|m'_2\|r'$
07: Return (x', σ')

Fig. 7. The intermediate adversary $\mathcal{A}_+$ constructed from the EUF-NMA adversary $\mathcal{A}$.

We consider a predicate $V(x', (y_H, y_G), \sigma')$ which outputs 1 *iff* $\omega' = y_H$ and $(m'_1\|r') \oplus G(\omega') = (M'\|R')$, where $\omega'\|M'\|R' := f_a(\sigma^*)$. This is the predicate version of the signature verification. Specifically, denote $h^{(x')} = H(x')$ and $g^{(x')} = G(h^{(x')})$, if (m'_2, σ') returned by $\mathcal{A}$ is a valid signature, then $V(x', (h^{(x')}, g^{(x')}), \sigma') = 1$, and vice versa.

Therefore, by the definition of $\mathcal{A}$ and $\mathcal{A}_+$, for some fixed $(a, t) \leftarrow \mathsf{TrapGen}(1^\lambda)$,

$$\Pr\left[V(x', (h^{(x')}, g^{(x')}), \sigma') = 1 : (x', \sigma') \leftarrow \mathcal{A}_+(a)\right] = \mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}}.$$

According to Theorem 3 (by treating $\mathcal{A}_+$ as the algorithm $\mathcal{A}$ in Theorem 3) and summing over all $x^\circ \in \mathcal{X}_0$, we know that there exists a simulator $\mathcal{S}$ such that,

$$\begin{aligned}&\Pr\left[V(x', (\Theta_H, \Theta_G), \sigma') = 1 : (x', \sigma') \leftarrow \left\langle \mathcal{S}^{\mathcal{A}_+}, \Theta_H, \Theta_G\right\rangle\right]\\ &\geq \frac{1}{(2\max\{q_H,q_G+1\}+1)^4} \cdot \Pr\left[V(x', (h^{x'}, g^{x'}), \sigma') = 1 : (x', \sigma') \leftarrow \mathcal{A}_+(a)\right] - \frac{1}{|\mathcal{Y}_1|}.\end{aligned}$$

Now, we use the statistical property of the preimage sampleable function to replace the Θ_H and Θ_G. Specifically, we first sample $\sigma^* \leftarrow \mathsf{SampleDom}(1^\lambda)$ and then write $\omega^* \| M^* \| R^* := f_a(\sigma^*)$. Upon the simulator's outputting $x' = m_1' \| m_2' \| r'$ in the first stage, we compute $\Theta'_H := \omega^*$ and $\Theta'_G := (M^* \oplus m_1') \| (R^* \oplus r')$, and replace Θ_H and Θ_G with Θ'_H and Θ'_G respectively. With the preimage sampling distribution property of PSF, we have that,

$$\begin{aligned}&\Pr\left[\, V(x', (\Theta_H, \Theta_G), \sigma') = 1 : (x', \sigma') \leftarrow \left\langle \mathcal{S}^{\mathcal{A}+}, \Theta_H, \Theta_G \right\rangle \right] \\ &\qquad = \Pr\left[\, V(x', (\Theta'_H, \Theta'_G), \sigma') = 1 : (x', \sigma') \leftarrow \left\langle \mathcal{S}^{\mathcal{A}+}, \Theta'_H, \Theta'_G \right\rangle \right].\end{aligned}$$

We will show that σ' and σ^* are very likely to be a pair of collision for the PSF if the simulator's output passed the predicate V.

In fact, it's easy to see $f_a(\sigma^*) = f_a(\sigma')$ from the fact that $V(x', \Theta'_H, \Theta'_G, \sigma') = 1$. Furthermore, as replacing Θ_H and Θ_G with Θ'_H and Θ'_G does not give away any information about the signature σ^* other than $f_a(\sigma^*)$, σ^* still satisfies the distribution $\mathcal{D}_x$ conditioned on $f_a(\sigma^*) = f_a(\sigma')$ under the simulator $\mathcal{S}$'s view.

This implies that, for $(x', \sigma') \leftarrow \left\langle \mathcal{S}^{\mathcal{A}+}, \Theta'_H, \Theta'_G \right\rangle$,

$$\Pr\left[\, \sigma' = \sigma^* \mid f_a(\sigma') = f_a(\sigma^*) \,\right] \le \max_{\sigma''} \Pr\left[\, \sigma^* = \sigma'' \mid f_a(\sigma') = f_a(\sigma^*) \,\right] \le p_{\max}.$$

Therefore, for $(x', \sigma') \leftarrow \left\langle \mathcal{S}^{\mathcal{A}+}, \Theta'_H, \Theta'_G \right\rangle$, we conclude that,

$$\begin{aligned}&\Pr\left[\, \sigma' \neq \sigma^* \wedge f_a(\sigma') = f_a(\sigma^*) \,\right] \\ =&\Pr\left[\, \sigma' \neq \sigma^* \mid f_a(\sigma') = f_a(\sigma^*) \,\right] \Pr\left[\, f_a(\sigma') = f_a(\sigma^*) \,\right] \\ \ge&\Pr\left[\, \sigma' \neq \sigma^* \mid f_a(\sigma') = f_a(\sigma^*) \,\right] \Pr\left[\, V(x', \Theta'_H, \Theta'_G, \sigma') = 1 \,\right] \\ \ge&(1 - p_{\max}) \cdot \Pr\left[\, V(x', \Theta'_H, \Theta'_G, \sigma') = 1 \,\right] \\ \ge&\tfrac{1-p_{\max}}{(2\max\{q_H, q_G+1\}+1)^4} \cdot \mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}} - \tfrac{1-p_{\max}}{|\mathcal{Y}_1|}.\end{aligned}$$

Finally, we are ready to show our constructed adversary $\mathcal{B}$ against the collision resistance of PSF. $\mathcal{B}$ runs the simulator $\mathcal{S}^{\mathcal{A}+}$ and prepares the inputs Θ'_H and Θ'_G with the sampled $\sigma^* \leftarrow \mathsf{SampleDom}(1^\lambda)$ in the way mentioned above. Upon the simulator's outputting x', σ', $\mathcal{B}$ outputs (σ^*, σ') as a collision of the PSF. With the above analysis, we conclude that,

$$\mathsf{Adv}^{\mathsf{CR}}_{\mathsf{PSF},\mathcal{B}} \ge \tfrac{1-p_{\max}}{(2\max\{q_H, q_G+1\}+1)^4} \cdot \mathsf{Adv}^{\mathsf{EUF\text{-}NMA}}_{\mathsf{SIG},\mathcal{A}} - \tfrac{1-p_{\max}}{|\mathcal{Y}_1|}.$$

This completes the proof of Lemma 4. □

Finally, we conclude that Theorem 7 follows directly from combining Lemma 3 and Lemma 4, and this shows the EUF-CMA security of our scheme in the QROM.

Acknowledgments. The authors sincerely appreciate the reviewers' valuable comments and helpful suggestions. The authors also thank You Lyu for insightful discussions during the early stages of this research. The authors were partially supported by National Natural Science Foundation of China (62522212, 62572295, 62372292), National Key R&D Program of China (2025YFA1017100), and National Cryptologic Science Fund of China (2025NCSF01009).

References

1. Alagic, G., Majenz, C., Russell, A., Song, F.: Quantum-access-secure message authentication via blind-unforgeability. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part III. Lecture Notes in Computer Science, vol. 12107, pp. 788–817. Springer, Cham, Switzerland, Zagreb, Croatia (May 10–14, 2020). https://doi.org/10.1007/978-3-030-45727-3_27
2. Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semi-classical oracles. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part II. Lecture Notes in Computer Science, vol. 11693, pp. 269–295. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2019). https://doi.org/10.1007/978-3-030-26951-7_10
3. Bellare, M., Rogaway, P.: The exact security of digital signatures: How to sign with RSA and Rabin. In: Maurer, U.M. (ed.) Advances in Cryptology – EUROCRYPT'96. Lecture Notes in Computer Science, vol. 1070, pp. 399–416. Springer Berlin Heidelberg, Germany, Saragossa, Spain (May 12–16, 1996). https://doi.org/10.1007/3-540-68339-9_34
4. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. Lecture Notes in Computer Science, vol. 7073, pp. 41–69. Springer Berlin Heidelberg, Germany, Seoul, South Korea (Dec 4–8, 2011). https://doi.org/10.1007/978-3-642-25385-0_3
5. Cojocaru, A., Garay, J.A., Liu, Q., Song, F.: Improved quantum lifting by coherent measure-and-reprogram. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024, Part IX. Lecture Notes in Computer Science, vol. 15492, pp. 158–184. Springer, Singapore, Singapore, Kolkata, India (Dec 9–13, 2024). https://doi.org/10.1007/978-981-96-0947-5_6
6. Don, J., Fehr, S., Majenz, C.: The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020, Part III. Lecture Notes in Computer Science, vol. 12172, pp. 602–631. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2020). https://doi.org/10.1007/978-3-030-56877-1_21
7. Ducas, L., Lyubashevsky, V., Prest, T.: Efficient identity-based encryption over NTRU lattices. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology – ASIACRYPT 2014, Part II. Lecture Notes in Computer Science, vol. 8874, pp. 22–41. Springer Berlin Heidelberg, Germany, Kaoshiung, Taiwan, R.O.C. (Dec 7–11, 2014). https://doi.org/10.1007/978-3-662-45608-8_2
8. Ducas, L., Prest, T.: Fast fourier orthogonalization. In: Proceedings of the 2016 ACM International Symposium on Symbolic and Algebraic Computation. p. 191–198 (2016)
9. Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z., et al.: Falcon: Fast-fourier lattice-based compact signatures over ntru. Submission to the NIST's post-quantum cryptography standardization process **36**(5), 1–75 (2018)
10. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th Annual ACM Symposium on Theory of Computing. pp. 197–206. ACM Press, Victoria, BC, Canada (May 17–20, 2008). https://doi.org/10.1145/1374376.1374407
11. Grilo, A.B., Hövelmanns, K., Hülsing, A., Majenz, C.: Tight adaptive reprogramming in the QROM. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2021, Part I. Lecture Notes in Computer Science, vol.

13090, pp. 637–667. Springer, Cham, Switzerland, Singapore (Dec 6–10, 2021). https://doi.org/10.1007/978-3-030-92062-3_22
12. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J.H., Whyte, W.: NTRUSIGN: Digital signatures using the NTRU lattice. In: Joye, M. (ed.) Topics in Cryptology – CT-RSA 2003. Lecture Notes in Computer Science, vol. 2612, pp. 122–140. Springer Berlin Heidelberg, Germany, San Francisco, CA, USA (Apr 13–17, 2003). https://doi.org/10.1007/3-540-36563-X_9
13. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press (2010)
14. Pan, J., Zeng, R.: Selective opening security in the quantum random oracle model, revisited. In: Tang, Q., Teague, V. (eds.) PKC 2024: 27th International Conference on Theory and Practice of Public Key Cryptography, Part III. Lecture Notes in Computer Science, vol. 14603, pp. 92–122. Springer, Cham, Switzerland, Sydney, NSW, Australia (Apr 15–17, 2024). https://doi.org/10.1007/978-3-031-57725-3_4
15. del Pino, R., Lyubashevsky, V., Pointcheval, D.: The whole is less than the sum of its parts: Constructing more efficient lattice-based AKEs. In: Zikas, V., De Prisco, R. (eds.) SCN 16: 10th International Conference on Security in Communication Networks. Lecture Notes in Computer Science, vol. 9841, pp. 273–291. Springer, Cham, Switzerland, Amalfi, Italy (Aug 31 – Sep 2, 2016). https://doi.org/10.1007/978-3-319-44618-9_15
16. Yamakawa, T., Zhandry, M.: Classical vs quantum random oracles. In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021, Part II. Lecture Notes in Computer Science, vol. 12697, pp. 568–597. Springer, Cham, Switzerland, Zagreb, Croatia (Oct 17–21, 2021). https://doi.org/10.1007/978-3-030-77886-6_20
17. Zhandry, M.: How to record quantum queries, and applications to quantum indifferentiability. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part II. Lecture Notes in Computer Science, vol. 11693, pp. 239–268. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2019). https://doi.org/10.1007/978-3-030-26951-7_9

A Quantum Representation of Game $\mathsf{REPRO}_1^{\mathsf{D}}$ in Theorem 5

In this appendix, we provide the missing transition from the ordinary formulation of Theorem 5 to its superposition oracle form, which is the one we use as the starting point in the proof. We will see the equivalence between the two formulations, as the following claim shows.

Claim. The game G_0 in Fig. 4 is perfectly indistinguishable from the game $\mathsf{REPRO}_1^{\mathsf{D}}$ in Fig. 3.

Proof. It suffices to show the claim in the case where $R = 1$, which means the challenger only queries the REPROGRAM oracle once. For $R > 1$, a similar reasoning of hybrid argument as was done in the proof of Theorem 5 will work.

We consider the game transition shown in Fig. 8.

Game H_0. It is the same game as the original game in Fig. 3, but represents all the classical procedures in a quantumly simulated way, and samples random

$\underline{\text{GAME } \mathsf{REPRO}_b^{\mathsf{D}}}$	$\underline{\mathsf{REPROGRAM}(x_1) \quad \mathsf{H}_0\text{-}\mathsf{H}_5}$	
	07: Prepare $\lvert x_1\rangle_{X_1}$	
	08: Prepare $\left\lvert\phi_0^{(\mathcal{X}_2)}\right\rangle_{X_2}$, $\left\lvert\phi_0^{(\mathcal{Y})}\right\rangle_{Y}$, $\left\lvert\phi_0^{(\mathcal{Z})}\right\rangle_{Z}$	
01: $\mathsf{O}_{H0} \leftarrow \mathcal{Y}^{\mathcal{X}}$	09: Measure X_2, Y, Z	$/\!/\mathsf{H}_0$
02: $\mathsf{O}_{G0} \leftarrow \mathcal{Z}^{\mathcal{X}}$	10: Prepare $\lvert 0\rangle_{Y'}, \lvert 0\rangle_{Z'}$	$/\!/\mathsf{H}_1$-H_5
03: $\mathsf{O}_{H1} := \mathsf{O}_{H0}$	11: Measure X_2	$/\!/\mathsf{H}_1$-H_5
04: $\mathsf{O}_{G1} := \mathsf{O}_{G0}$	12: Apply $\mathsf{CNOT}_{YY'}$, $\mathsf{CNOT}_{ZZ'}$	$/\!/\mathsf{H}_1$
05: $b' \leftarrow \mathsf{D}_b^{\lvert\mathsf{O}_{Gb}\rangle,\lvert\mathsf{O}_{Hb}\rangle,\mathsf{REPROGRAM}}$	13: Apply RE_{YF_GZ}	
06: Return b'	14: Apply RE_{XF_HY}	
	15: Apply $\sum_{x\in\mathcal{X}} \lvert x\rangle\langle x\rvert_X \otimes \mathsf{CNOT}_{F_{Hx}Y'}$	$/\!/\mathsf{H}_2$
	16: Apply $\sum_{y\in\mathcal{Y}} \lvert y\rangle\langle y\rvert_{Y'} \otimes \mathsf{CNOT}_{F_{Gy}Z'}$	$/\!/\mathsf{H}_2$
	17: Apply O_H on input registers XY'	$/\!/\mathsf{H}_3$-H_4
	18: Apply O_G on input registers $Y'Z'$	$/\!/\mathsf{H}_3$
	19: Return register X_2	

Fig. 8. Game transitions from the game REPRO_1^D in Fig. 3 to the game G_0 in Fig. 4

variables by preparing and measuring corresponding quantum states. The game is identical to the game $\mathsf{REPRO}_1^{\mathsf{D}}$ in Fig. 3.

Game H_1. This game uses the purified measurement technique, copying the registers X_2, Y, Z to X_2', Y', Z' using CNOT gates. This is perfectly indistinguishable from measuring the registers X_2, Y, Z with respect to their computational basis if the extra registers X_2', Y', Z' are all hidden from the other parts of the quantum circuit, as the view of the distinguisher would, when represented by density operators, be the same.

Game H_2. We try to postpone the purified measurement to after the RE gates. It is trivial that $\mathsf{CNOT}_{X_2X_2'}$ commutes with both RE_{YF_GZ} and RE_{XF_HY} and that $\mathsf{CNOT}_{YY'}$ is commutable with RE_{YF_GZ}.

For the following calculations, we heavily use the fact that two quantum gates commute if, 1) both gates act on disjoint sets of registers or, 2) all registers accessed by both gates are used as control registers in both gates.

To move $\mathsf{CNOT}_{ZZ'}$ after RE_{YF_GZ}, we have,

$$
\begin{aligned}
&\mathsf{RE}_{XF_HY} \circ \mathsf{RE}_{YF_GZ} \circ \mathsf{CNOT}_{ZZ'} \\
=&\mathsf{RE}_{XF_HY} \circ \sum_{y\in\mathcal{Y}} \lvert y\rangle\langle y\rvert_Y \otimes (\mathsf{SWAP}_{F_{Gy},Z} \circ \mathsf{CNOT}_{F_{Gy}Z'}) \\
=&\mathsf{RE}_{XF_HY} \circ \sum_{y\in\mathcal{Y}} \lvert y\rangle\langle y\rvert_Y \otimes (\mathsf{CNOT}_{F_{Gy}Z'} \circ \mathsf{SWAP}_{F_{Gy},Z}) \\
=&\mathsf{RE}_{XF_HY} \circ \left(\sum_{y\in\mathcal{Y}} \lvert y\rangle\langle y\rvert_Y \otimes \mathsf{CNOT}_{F_{Gy}Z'}\right) \circ \mathsf{RE}_{YF_GZ}.
\end{aligned}
$$

As the purified measurement of Y register has already been applied, and the register Y acts as a control register in the RE_{YF_GZ} gate, the system state can

be decomposed as

$$\sum_{y\in\mathcal{Y}} c_y \left|yy\right\rangle_{YY'} \otimes \left|\psi_y\right\rangle_{XZX'Z'F_H},$$

for some coefficient $c_y \in \mathbb{C}$ and vectors $\left|\psi_y\right\rangle_{ZX'Z'F_H}$. In other words, the registers Y and Y' are still entangled, and contain the same element when measuring to the computational basis.

This means we can rewrite $\sum_{y\in\mathcal{Y}} \left|y\right\rangle\!\left\langle y\right|_Y \otimes \mathsf{CNOT}_{F_{G_y}Z'}$ to $\sum_{y\in\mathcal{Y}} \left|y\right\rangle\!\left\langle y\right|_{Y'} \otimes \mathsf{CNOT}_{F_{G_y}Z'}$. With this representation, we have that,

$$\begin{aligned}
&\mathsf{RE}_{XF_HY} \circ \mathsf{RE}_{YF_GZ} \circ \mathsf{CNOT}_{ZZ'} \\
=&\mathsf{RE}_{XF_HY} \circ \left(\sum_{y\in\mathcal{Y}} \left|y\right\rangle\!\left\langle y\right|_{Y'} \otimes \mathsf{CNOT}_{F_{G_y}Z'}\right) \circ \mathsf{RE}_{YF_GZ} \\
=&\left(\sum_{y\in\mathcal{Y}} \left|y\right\rangle\!\left\langle y\right|_{Y'} \otimes \mathsf{CNOT}_{F_{G_y}Z'}\right) \circ \mathsf{RE}_{XF_HY} \circ \mathsf{RE}_{YF_GZ}.
\end{aligned}$$

To move $\mathsf{CNOT}_{YY'}$ after RE_{XF_HY}, we have,

$$\begin{aligned}
\mathsf{RE}_{XF_HY} \circ \mathsf{CNOT}_{YY'} &= \sum_{x\in\mathcal{X}} \left|x\right\rangle\!\left\langle x\right|_X \otimes (\mathsf{SWAP}_{F_{Hx},Y} \circ \mathsf{CNOT}_{YY'}) \\
&= \sum_{x\in\mathcal{X}} \left|x\right\rangle\!\left\langle x\right|_X \otimes (\mathsf{CNOT}_{F_{Hx}Y'} \circ \mathsf{SWAP}_{F_{Hx},Y}) \\
&= \left(\sum_{x\in\mathcal{X}} \left|x\right\rangle\!\left\langle x\right|_X \otimes \mathsf{CNOT}_{F_{Hx}Y'}\right) \circ \mathsf{RE}_{XF_HY}.
\end{aligned}$$

With the above calculations, we show that the game H_2 is perfectly indistinguishable from the game H_1, i.e., $\Pr_2 = \Pr_1$.

Game H_3. We find that the controlled CNOT gates share the same expression as QRO queries, thus we use the QRO query to represent them instead. This is a literal translation, so this game is identical to the previous game, i.e., $\Pr_3 = \Pr_2$.

Game H_4. In this game, we do not query the random oracle G on the registers Y' and Z'. As Y' and Z' are internal registers hidden from the distinguisher, removing this does not affect the distinguisher's view. Therefore, $\Pr_4 = \Pr_3$.

Game H_5. In this game, we do not query the random oracle H on the registers X and Y'. As X is measured, and Y' is an internal register hidden from the distinguisher, this also doesn't affect the distinguisher's view. Therefore $\Pr_5 = \Pr_4$

Now we can see that H_5 in Fig. 8 is identical to G_0 in Fig. 4, and this completes the proof of our claim. □