White-Box Watermarking Signatures against Quantum Adversaries and Its Applications

Fuyuki Kitagawa[⋆♢] and Ryo Nishimaki[⋆♢]

[⋆]NTT Social Informatics Laboratories, Tokyo, Japan

{fuyuki.kitagawa,ryo.nishimaki}@ntt.com

[♢]NTT Research Center for Theoretical Quantum Information, Atsugi, Japan

February 18, 2025

Abstract

Software watermarking for cryptographic functionalities enables embedding an arbitrary message (a mark) into a cryptographic function. An extraction algorithm, when provided with a (potentially unauthorized) circuit, retrieves either the embedded mark or a special symbol $\mathsf{unmarked}$ indicating the absence of a mark. It is difficult to modify or remove the embedded mark without destroying the functionality of a marked function. Previous works have primarily employed black-box extraction techniques, where the extraction algorithm requires only input-output access to the circuit rather than its internal descriptions (white-box extraction). Zhandry (CRYPTO 2021) identified several challenges in watermarking public-key encryption (PKE) with black-box extraction and introduced the notion of privacy for white-box watermarking against classical adversaries. Kitagawa and Nishimaki (Journal of Cryptology 37(3)) extended watermarking techniques to pseudorandom functions (PRFs) and PKE in the presence of quantum adversaries, enabling extraction from pirate quantum circuits but failing to achieve privacy.

In this work, we investigate *white-box* watermarking for *digital signatures* secure against *quantum* adversaries. Our constructions enable the extraction of embedded marks from the description of a pirate quantum circuit that produces valid signatures while ensuring that black-box access to a marked signing function does not reveal information about the embedded mark. We define and construct white-box watermarking signatures that are secure against quantum adversaries, leveraging the leaning with errors (LWE) assumption and quantum fully homomorphic encryption. Furthermore, we highlight that privacy concerns are even more critical in the context of signatures than in PKE. We also present a compelling practical application of white-box watermarking signatures.

Additionally, we explore the concept of universal copy protection for signatures. We define universal copy protection as a mechanism that transforms any quantumly secure signature scheme into a copy-protected variant without altering the verification key or verification algorithm. This approach is preferable to developing specific copy-protected signature schemes, as it allows existing schemes to be secured without modifying their published verification keys. We demonstrate that universal copy protection for all quantum secure signatures is impossible by leveraging our white-box watermarking signatures secure against quantum adversaries.

Contents

1 Introduction

1.1 Background

Watermarking. Software watermarking [BGI+12] for cryptographic functionalities [CHN+18, GKM+19] enables embedding arbitrary messages (marks) into cryptographic functions modeled as circuits, such as decryption functions in encryption schemes and signing functions in digital signatures. A marked circuit retains the functionality of the original unmarked circuit. An extraction algorithm, when provided with a potentially marked circuit C, can retrieve the embedded mark or indicate that no mark is present (output special symbol $\mathsf{unmarked}$). Importantly, it is difficult to remove or alter the embedded mark without impairing the circuit's functionality. Applications of software watermarking include identifying ownership of objects and tracing unauthorized distributions. For example, (collusion-resistant) watermarking decryption functions can be seen as a form of traitor tracing, where unique marks are embedded in individual decryption keys to identify and track unauthorized distributions.[1]

Black-box extraction. Most cryptographic watermarking schemes (secure against arbitrary strategies) except one scheme employ black-box extraction methods, where the extraction algorithm relies only input-output behavior rather than internal circuit descriptions [CHN+18, BLW17, KW21, QWZ18, KW19, GKM+19, YAL+19, YAYX20, Nis20, GKWW21, BBL24].[2] This approach is natural in cryptographic software watermarking, as pirate software may be obfuscated, making non-black-box analysis challenging.

Public extraction. Public extractability is often preferable to private extractability, where extraction requires a secret key for extraction. In privately extractable watermarking, the authority that holds the secret extraction key must not be compromised. *Publicly extractable watermarking schemes allow anyone to extract an embedded mark*, that is, verify ownership and detect unauthorized distribution, much like watermarking in perceptual media (e.g., images or cash). Accordingly, many prior works have explored black-box public extraction schemes [CHN+18, GKM+19, YAL+19, GKWW21].

Privacy issue in black-box public extraction. Zhandry [Zha21] identified privacy risks in black-box public extraction. Although his work focused on *traitor tracing*, similar concerns arise in software watermarking as he referred to in the future direction section of his work [Zha21, Section 1.3]. A critical issue is that *public extraction allows anyone to retrieve embedded information by observing the functional behavior* of cryptographic operations. For instance, to deter unauthorized distribution and verify ownership, watermarking schemes may embed sensitive personal information such as bank account numbers into cryptographic keys [NWZ16]. Such watermarking schemes may inadvertently expose this data to unauthorized observers. To address this, Zhandry [Zha21] identifies a natural scenario where we use traitor tracing and users can observe other users' decryption function behavior and break privacy by using black-box public tracing. Zhandry introduced the concept of *white-box* traitor tracing to resolve the privacy problem above in the traitor tracing setting. White-box traitor tracing relies on non-black-box algorithms that analyze the internal structure of circuits rather their input-output behavior.

White-box watermarking signatures. In this work, we focus on white-box watermarking for *signing* functions (white-box watermarking signatures), where extraction requires access to circuit descriptions rather than input-output behavior. Digital signatures play a fundamental role in *authentication* and security protocols. Privacy risks are particularly severe in the signature setting because messages and corresponding signatures are often publicly observable. *A watermarking scheme with black-box public extraction would allow any external observer to extract embedded marks (potentially sensitive information) from publicly available message-signature pairs, posing a significant privacy threat.* Moreover, ensuring post-quantum security is increasingly important due to advancements in quantum computing. Thus, our primary research questions are:

[1] A user decryption key dk_i is a marked decryption key $\mathsf{Mark}(\mathsf{dk}, \mu_i)$ where dk is the original decryption key and μ_i is an embedded mark. Hence, adversaries could obtain many marked keys and we need to consider collusion-resistant watermarking for public-key encryption in a sense by Goyal et al. [GKM+19] to achieve traitor tracing. We do not consider the collusion-resistant setting in this work.

[2] The extraction algorithm of the watermarking PRF by Yang et al. [YYAS22] uses circuit descriptions in a non-black-box way since they use unobfuscatable PRFs as a building block. However, they did not study the privacy issue of watermarking (explained below).

What are the formal definitions of white-box watermarking signatures? And,
Can we achieve white-box watermarking signatures that are secure against quantum adversaries?

Why do we need "white-box" watermarking signatures? A compelling application of white-box watermarking signatures is as follows. Consider a service that offers discount coupons to users affiliated with a specific organization (e.g., a university). Each member of the organization receives a signing key $\mathsf{sk}_{\mathsf{sen\text{-}info}}$, which embeds the user's sensitive personal information, $\mathsf{sen\text{-}info}$.[3] The organization registers the corresponding verification key, vk. Users can claim discounts by submitting a valid signature under vk. It is important to note that white-box watermarking signatures are *not* used as e-cash but rather for authentication—specifically, for proving eligibility for certain services. A key advantage of this approach is that it discourages users from illegally sharing their signing keys outside the designated group. This deterrence is due to the unremovability property of watermarking: embedded strings $\mathsf{sen\text{-}info}$ (potentially sensitive personal information) can be publicly extracted from signing function descriptions, making any unauthorized key distribution traceable. However, *if watermarking signatures were black-box publicly extractable, anyone could extract sensitive information* $\mathsf{sen\text{-}info}$ *simply by analyzing pairs of signatures and messages (i.e., input-output behavior)*. This poses a privacy risk, necessitating the privacy-preserving properties of white-box watermarking signatures to protect users' sensitive data.

One might initially consider group signatures [Cv91] as a suitable cryptographic alternative. However, group signatures rely on a central authority (group manager) who has the ability to reveal a user's identity from their *signatures*. In our scenario, we prefer to avoid such a central authority, as it could become a single point of compromise. Unlike group signatures, white-box watermarking signatures do not allow information extraction from signatures while still enabling the embedding of arbitrary strings. In contrast, group signatures only disclose a user index $i \in [N]$, where N represents the total number of users, rather than embedding arbitrary data. For these reasons, white-box watermarking signatures are well-suited for the described application and, and in come cases, may serve as an alternative to group signatures. Additionally, it is important to recognize the distinct purposes of these two cryptographic tools. Group signatures are designed for traceability, enabling authorities to identify individual who have violated rules (e.g., committed a crime) based on the time and location of a generated signature. White-box watermarking signatures, on the other hand, primarily serve as a deterrent against unauthorized distribution of signing keys.

On the impossibility of universal copy-protection for signatures. Interestingly, white-box watermarking against quantum adversaries is closely related to the impossibility of universal copy-protection. Quantum copy-protection [Aar09] is a cryptographic primitive that transforms classical programs into quantum states, allowing computation of the same functionality as the original program while preventing duplication of the quantum state. Previous research has demonstrated that *all learnable functions* and *certain point functions* cannot be copy-protected [Aar09, AL21, AK22]. However, these results do not rule out the possibility of universal copy-protection for signature schemes. A universal copy-protection scheme for signatures would provide a single method to transform *any* quantumly secure (EUF-qCMA secure) signature scheme into one where the signing key is copy-protected, while keeping the verification key and algorithm unchanged. From a practical perspective, such a universal transformation would be highly desirable [DN21]. Although Liu, Liu, Qian, and Zhandry [LLQZ22] introduced a specific bounded collusion-resistant copy-protection scheme for signatures, their approach does not provide a universal construction. The question of whether universal copy-protection for signatures is possible remains an intriguing open question. In this work, we investigate the impossibility of universal copy-protection for signatures through the lens of white-box watermarking signatures against quantum adversaries.

1.2 Our Results

We present two main contributions in this work. First, we introduce the definitions of white-box watermarking signatures against quantum adversaries and analyze their properties. Second, we construct white-box watermarkable signature

[3] The organization can provide $\mathsf{sk}_{\mathsf{sen\text{-}info}}$ without knowing $\mathsf{sen\text{-}info}$ by using secure two-party computation. If the organization needs to check a user embeds valid personal information (e.g., bank account number), another entity (e.g., a bank) joins, and they can use secure three-party computation.

schemes that are secure against quantum adversaries under standard cryptographic assumptions. A a byproduct of our results, we establish the impossibility of universal copy-protection for signature schemes. Below, we provide a detailed overview of these contributions.

Definitions. We introduce two types of watermarking signature syntax:

1. Pre-embedded white-box watermarking signatures — The embedded mark is determined during the key generation phase.
2. Standard watermarking signatures — The embedded mark is assigned after key generation.

A watermarking signatures scheme must satisfy both unforgeability and unremovability, as defined by Goyal et al. [GKM^{+}19]. We extend these definitions to quantum adversaries by adapting the watermarking PRF framework against quantum adversaries introduced by Kitagawa and Nishimaki [KN24]. Additionally, we introduce privacy as a crucial property of white-box watermarking signatures.

Our privacy guarantee ensures that an adversary cannot infer any information about the embedded mark μ, provided they can only access a signing oracle that returns $\sigma \leftarrow \mathsf{Sign}(\widetilde{\mathsf{sk}}_{\mu}, \mathsf{m})$ in a black-box manner, where $\widetilde{\mathsf{sk}}_{\mu}$ is a marked signing key and m is the queried message. This formulation is a natural adaptation of privacy in white-box traitor tracing. In the non-pre-embedded (i.e., standard watermarking signatures) setting, we consider a stronger adversarial model in which *attackers can generate their own verification and signing key pairs* $(\mathsf{vk}, \mathsf{sk})$. In this setting, privacy remains intact even against a malicious signature authority.[4] Notably, our framework does not require a watermarking authority, as our constructions do not rely on any secret key for embedding or extracting marks.

We also define strong correctness for watermarking signatures. This property ensures that an adversary cannot find a message m^* such that a marked signing function generates an invalid signature for the input m^*. Since marked signing functions do not exhibit perfect correctness, there exist certain inputs that could potentially cause failure. Our goal is to prevent adversaries from exploiting this weakness to make a watermarked signing key fail when generating valid signatures.

Constructions. We propose a pre-embedded white-box watermarking signature scheme constructed from standard cryptographic tools. All components, except for quantum fully homomorphic encryption (QFHE)[5], can be instantiated under the learning with errors (LWE) assumption. This leads to the following result:

Theorem 1.1 (informal). *If the LWE assumption holds and QFHE exists, then a pre-embedded white-box watermarking signature scheme secure against quantum adversaries exists.*

This result represents the first construction of a white-box watermarking signature scheme designed to withstand quantum adversaries. Notably, achieving pre-embedded white-box watermarking signatures is non-trivial, even against classical adversaries. This is in contrast to watermarking signatures with black-box extraction. Goyal et al. [GKM^{+}19, Section B.1.2, in eprint ver.] observed that if the verification key depends on the embedded mark, there is a trivial watermarking signature scheme that satisfies unremovability. However, this approach fails in the white-box setting because the mark would be explicitly included in the verification key, immediately violating privacy. Furthermore, constructing pre-embedded white-box watermarking signatures *against quantum adversaries* is significantly more challenging than their classical counterparts (similar to the difficulties in watermarking PRFs against quantum adversaries [KN24]). This is due to the nature of quantum circuits, where running a circuit may irreversibly alter its quantum state, and the approximate correctness condition on pirate circuits. We also stress that pre-embedded white-box watermarking signatures are sufficient for the applications in Section 1.1 if each user generates a key pair.

To achieve white-box watermarking signatures against quantum adversaries, we introduce a fascinating non-black-box extraction technique. Specifically, we define two new cryptographic primitives.

[4] A user can receive $\widetilde{\mathsf{sk}}_{\mu}$ from an authority who has sk via secure two-party computation without revealing μ and $\widetilde{\mathsf{sk}}_{\mu}$ to the authority. Hence, this setting is meaningful.

[5] Not leveled QFHE but QFHE. We need to assume circular security of encryption to achieve QFHE [Mah18, Bra18].

1. After-the-fact leakage-resilient quantum unobfuscatable point functions — These ensure that quantum black-box unlearnability holds even if partial information about output messages (i.e., the output corresponding to the point) is leaked.
2. Functional encryption with ciphertext uniformity — This guarantees that ciphertexts appear random if the decrypted result is a uniformly random value.

These new primitives serve as essential building blocks in our construction. Beyond their use in this work, they may have independent cryptographic applications. This technique is an interesting application of leakage-resilient cryptography. See Section 1.3 for the details.

Additionally, we extend our pre-embedded white-box watermarking signature scheme to a white-box watermarking signature scheme by employing a non-black-box transformation using standard EUF-CMA secure signatures.

Theorem 1.2 (informal). *If the LWE assumption holds and QFHE exists, then a white-box watermarking signature scheme secure against quantum adversaries exists.*

Impossibility of universal copy-protection. To demonstrate the impossibility of universal copy-protection for signatures, we must consider signature schemes secure against quantum superposition attacks (EUF-qCMA) [BZ13]. Boneh and Zhandry [BZ13] showed that an EUF-CMA secure signature can become completely insecure when subjected to quantum chosen message attacks, as the classical signing key can be fully recovered. Since an adversary with a (potentially quantum) description of the signing algorithm can execute it in superposition and extract the singing key, universal copy-protection for EUF-CMA secure signatures is ruled out. However, this does not immediately preclude universal copy-protection for EUF-qCMA secure signatures.

We can construct an EUF-qCMA signature scheme whose signing key cannot be copy-protected by combining:

- Standard EUF-qCMA secure signatures
- One-way functions
- Pre-embedded white-box watermarking signatures against quantum adversaries

Since EUF-qCMA secure signatures can be instantiated under the LWE assumption [BZ13], we obtain the following result:

Theorem 1.3 (informal). *If the LWE assumption holds and QFHE exists, then universal quantum copy-protection for EUF-qCMA secure signatures is impossible.*

This result marks the first known impossibility proof for universal copy-protection of *signature schemes*.

1.3 Technical Overview

Syntax of pre-embedded white-box watermarking signature. We first introduce the syntax of pre-embedded white-box watermarking signatures against quantum adversaries. A pre-embedded white-box watermarking signature scheme consists of four algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathcal{E}xtract)$. The first three algorithms form a standard digital signature scheme, except that KeyGen takes a secret message μ as input. Also, we require that Sign be a deterministic algorithm. Finally, $\mathcal{E}xtract$ is the extraction algorithm to extract the secret message embedded into the key pair from a possibly obfuscated quantum signing program generated using the key pair. More concretely, $\mathcal{E}xtract$ takes as input a verification key vk, a quantum program $\widetilde{C}$[6], and a threshold parameter ϵ, and outputs some μ'.

[6]In this work, we treat only quantum programs with classical input and output that consist of a unitary and an initial quantum state. For the formal definition, see Definition 2.2.

Security notions. For white-box watermarking signatures against quantum adversaries, aside from unforgeability as digital signature, we consider the following three security notions, that is, unremovability, privacy, and strong correctness.

- We say that a white-box watermarking signature scheme satisfies unremovability if given a pair of verification key vk and signing key sk that has the embedded secret message μ, any adversary cannot generate a quantum program $\widetilde{C}$ such that it is an "ϵ-good program", but the extraction algorithm executed with the parameter ϵ fails to output the embedded secret message μ from it. We roughly define a quantum signing program as an "ϵ-good program" if it outputs a valid signature for a randomly chosen message with a probability greater than ϵ. More specifically, to consider the stateful nature of quantum programs, we use the notion of "ϵ-live program" defined by Zhandry [Zha20] in the context of quantum traitor tracing. Roughly speaking, "ϵ-live program" is a quantum program such that if we measure the success probability of it using the method called projective implementation introduced by Zhandry [Zha20], we obtain the measurement result greater than ϵ with overwhelming probability. As the name suggests, projective implementation is a method that measures the success probability of a quantum program in a projective manner, which means if we measure the success probability twice successively, we obtain the same result.[7] We use this simplified definition of "ϵ-live program" in this overview.

- We say that a white-box watermarking signature scheme is private if any adversary who is given vk and can get quantum access to the signing oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$ cannot obtain any information of the secret message μ that is tied to $(\mathsf{vk}, \mathsf{sk})$ (i.e., $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, \mu)$.). Quantum access means the adversary is allowed to query two registers R_1 and R_2 and the oracle applies the map $|a\rangle |b\rangle \rightarrow |a\rangle |b \oplus \mathsf{Sign}(\mathsf{sk}, a)\rangle$[8] to the registers and returns them. We consider an indistinguishability-based notion. Hence, the adversary's task is to distinguish two secret messages chosen by the adversary itself.

- We say that a white-box watermarking signature scheme satisfies strong correctness if any adversary who is given vk and can get access to the signing oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$ cannot find m^* such that $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}^*)) = 0$

Construction strategy for white-box watermarking signature. Our basic idea is to turn a quantum unobfuscatable function [ABDS21, AL21] into a signature scheme, achieving unremovability. Concretely, we use a non-interactive zero-knowledge (NIZK) argument and design our scheme so that a signature is a proof of NIZK for a statement related to the quantum unobfuscatable function. To implement this idea, we also use functional encryption (FE) that satisfies the newly introduced property ciphertext uniformity. We below explain our main building blocks in detail.

Quantum unobfuscatable point function Quantum unobfuscatable point function UOPF consists of $\mathsf{UOPF.Gen}$ and $\mathsf{UOPF}.\mathcal{Extract}$. $\mathsf{UOPF.Gen}$ is given a secret message μ as an input and outputs a uniformly generated point function $f_{\alpha,\beta} : \{0,1\}^{\ell_{\mathsf{in}}} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}}$ that outputs β if the input is α and $0^{\ell_{\mathsf{out}}}$ otherwise, together with an auxiliary information aux. $\mathsf{UOPF}.\mathcal{Extract}$ takes as input a quantum program $\widetilde{C}$ and aux, and outputs μ'.

Usually, quantum unobfuscatable point functions satisfy the following correctness and security. The correctness notion guarantees that if $\mathsf{UOPF}.\mathcal{Extract}$ is given a quantum program that maps α to β with overwhelming probability together with aux, it outputs the secret message μ used to generate the point function $f_{\alpha,\beta}$ and aux. The security notion guarantees that any adversary cannot compute μ given aux and quantum oracle access to $f_{\alpha,\beta}$.

In this work, we decompose the above security notion into the following indistinguishability of messages and indistinguishability of points.

Indistinguishability of messages It requires that for any μ_0 and μ_1, aux_0 and aux_1 are computationally indistinguishable, where $(f_{\alpha,\beta}, \mathsf{aux}_b) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu_b)$ for $b \in \{0,1\}$.

Indistinguishability of points It requires that for any μ, α is indistinguishable from a completely independent random string $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$ given aux, where $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu)$.

[7] Projective implementation is an inefficient method. Hence, we use an approximate variant in the actual technical sections. We ignore this issue in this overview.

[8] Recall that Sign is deterministic.

Indistinguishability of points intuitively ensures that quantum oracle access to $f_{\alpha,\beta}$ is useless. Then, the indistinguishability of messages is sufficient to imply the standard security notion of quantum unobfuscatable functions.

FE with ciphertext uniformity An FE scheme FE consists of four algorithms $(\mathsf{FE.Setup}, \mathsf{FE.KG}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$. $\mathsf{FE.Setup}$ takes as input a security parameter and outputs a public key $\mathsf{fe.pk}$ and a master secret key $\mathsf{fe.msk}$. $\mathsf{FE.KG}$ takes as input the master secret key $\mathsf{fe.msk}$ and a function f and outputs a functional decryption key fsk. $\mathsf{FE.Enc}$ takes as input $\mathsf{fe.pk}$ and an input x, and outputs a ciphertext ct. We can decrypt ct with fsk using $\mathsf{FE.Dec}$, and obtain $f(x)$. The ciphertext uniformity requires that $\mathsf{FE.Enc}(\mathsf{fe.pk}, x)$ be computationally indistinguishable from a uniformly random string even given fsk for a function f, if the value $f(x)$ distributes uniformly at random.[9]

In this work, we use FE with ciphertext uniformity for 1-out-of-2 oblivious transfer (OT) functionality

$$F[\beta](i, x_0, x_1) = x_{\beta[i]},$$

where $\beta[i]$ is the i-th bit of β. We show that FE with ciphertext uniformity for 1-out-of-2 OT functionality can be achieved from the LWE assumption.

Statistical NIZK argument A statistical NIZK argument $\mathsf{NIZK} = (\mathsf{NIZK.Prove}, \mathsf{NIZK.Vrfy})$ for a relation $\mathcal{R}$ satisfies three properties completeness, computational soundness, and statistical zero-knowledge. Completeness ensures that honestly generated proof $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w)$ for $(x, w) \in \mathcal{R}$ is always accepted by $\mathsf{NIZK.Vrfy}$, where crs is the common reference string generated by a trusted third party. The computational soundness guarantees that any efficient adversary cannot find a valid proof for a statement x outside of $\mathcal{R}$. Finally, statistical zero-knowledge guarantees that any computationally unbounded adversary cannot obtain any information from an honestly generated proof $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w)$ for $(x, w) \in \mathcal{R}$ except the fact that x is in $\mathcal{R}$.

In addition to the above building blocks, we use length-doubling PRG g and statistically binding commitment Commit.[10] Also, in the actual construction, we use a (quantum-accessible) pseudorandom function to make the signing algorithm deterministic. However, we omit the de-randomization for simplicity in this overview.

First attempt. We first present a simplified scheme PWMSIG' that satisfies unremovability but not privacy and even (existential) unforgeability. The relation $\mathcal{R}$ of NIZK in PWMSIG' is defined as $(x = (\mathsf{m}, \gamma, \mathsf{com}), w = (\mathsf{fsk}, r)) \in \mathcal{R}$ if and only if it holds that

$$\mathsf{com} = \mathsf{Commit}(\mathsf{fsk}; r) \wedge g(\mathsf{FE.Dec}(\mathsf{fsk}, \mathsf{m})) \neq \gamma.$$

The descriptions of $\mathsf{PWMSIG}'.\mathsf{KeyGen}$, $\mathsf{PWMSIG}'.\mathsf{Sign}$, and $\mathsf{PWMSIG}'.\mathsf{Vrfy}$ are as follows.

$\mathsf{PWMSIG}'.\mathsf{KeyGen}$**:** Given μ as an input, it first generates $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu)$ and $\gamma \leftarrow g(\alpha)$. It also generates crs of NIZK and $(\mathsf{fe.pk}, \mathsf{fe.msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda)$. If finally generates $\mathsf{fsk} \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}, F[\beta])$ and its commitment $\mathsf{com} \leftarrow \mathsf{Commit}(\mathsf{fsk}; r)$. The verification key is $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}, \mathsf{com}, \mathsf{aux})$ and the corresponding signing key is $\mathsf{sk} = (\mathsf{fsk}, r)$. Below, we also assume that sk implicitly includes vk.

$\mathsf{PWMSIG}'.\mathsf{Sign}$**:** Given $\mathsf{sk} = (\mathsf{fsk}, r)$ and m, it outputs a proof π of NIZK for the statement $(\mathsf{m}, \gamma, \mathsf{com})$ using $\mathsf{sk} = (\mathsf{fsk}, r)$ as the witness.

$\mathsf{PWMSIG}'.\mathsf{Vrfy}$**:** Given $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}, \mathsf{com}, \mathsf{aux})$, a message m, and a signature $\sigma = \pi$, it simply outputs the verification result of NIZK, that is, $\mathsf{NIZK.Vrfy}(\mathsf{crs}, (\mathsf{m}, \gamma, \mathsf{com}), \pi)$.

The correctness of PWMSIG' follows from the completeness of NIZK since the condition $g(\mathsf{FE.Dec}(\mathsf{fsk}, \mathsf{m})) \neq \gamma$ is satisfied for every m with overwhelming probability over the choice of α due to the pseudorandomness of PRG g.

We then move on to the construction of $\mathsf{PWMSIG}'.\mathcal{E}xtract$. $\mathsf{PWMSIG}'.\mathcal{E}xtract$ basically relies on $\mathsf{UOPF}.\mathcal{E}xtract$. To this end, all we have to do is to construct a quantum program that maps α to β with overwhelming probability, using a live signing quantum program. We introduce the following sub-routine algorithm $\mathcal{S}earchOutput$.

[9] In the actual definition, we decompose this property into the standard simulation security and the pseudorandomness of the simulator's output.
[10] For simplicity, we omit to write the commitment key and its generation algorithm in this overview.

$\mathcal{SearchOutput}$**:** It takes as input $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}, \mathsf{com}, \mathsf{aux})$, a quantum program $\widetilde{C}$, $x \in \{0,1\}^{\ell_{\mathsf{in}}}$, $i \in \{1, \cdots, \ell_{\mathsf{out}}\}$, and the threshold parameter ϵ. It estimates the probability that $\widetilde{C}$ outputs a valid signature when it is given a message that is a ciphertext of FE sampled from the following distribution D_i.

D_i**:** Generate $u \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$ and compute $\mathsf{fe.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}, (i, x, u))$. Output $\mathsf{m} := \mathsf{fe.ct}$.

If the estimation result is smaller than $\epsilon/2$, it outputs $\beta[i] = 0$; otherwise, it outputs $\beta[i] = 1$.

Then, we define $\mathcal{P}[\widetilde{C}](x)$ as the following quantum program

- It takes $x \in \{0,1\}^{\ell_{\mathsf{in}}}$ as the input.
- It does the following from $i = 1$ to $i = \ell_{\mathsf{out}}$: Compute $\beta'[i] \leftarrow \mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}_i, x, i, \epsilon)$, uncompute the process, and obtain quantum program $\widetilde{C}_{i+1}$, where $\widetilde{C}_1 := \widetilde{C}$.
- Outputs $\beta'[1] \| \cdots \| \beta'[\ell_{\mathsf{out}}]$.

We are now ready to present the description of $\mathsf{UOSIG}'.\mathcal{Extract}$.

$\mathsf{PWMSIG}'.\mathcal{Extract}$**:** Given $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}, \mathsf{com}, \mathsf{aux})$, a quantum program $\widetilde{C}$, and the threshold parameter ϵ, it first construct $\mathcal{P}[\widetilde{C}]$ and outputs $\mu' \leftarrow \mathsf{UOPF}.\mathcal{Extract}(\mathcal{P}[\widetilde{C}], \mathsf{aux})$.

Unremovability of PWMSIG' against quantum adversaries. We show the unremovability of PWMSIG' against quantum adversaries. Suppose an adversary is given $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}, \mathsf{com}, \mathsf{aux})$ and $\mathsf{sk} = (\mathsf{fsk}, r)$, and outputs a quantum program $\widetilde{C}$ and the threshold parameter ϵ. We below show that if $\widetilde{C}$ is an ϵ-live quantum signing program, that is, if we measure the success probability of $\widetilde{C}$ with respect to random messages, we obtain a measurement result greater than ϵ with overwhelming probability, the i-th execution of $\mathcal{SearchOutput}$ in $\mathcal{P}[\widetilde{C}]$ with the input α outputs $\beta[i]$ with overwhelming probability for every $i \in \{1, \cdots, \ell_{\mathsf{out}}\}$, which means that $\mathcal{P}[\widetilde{C}]$ maps α to β with overwhelming probability. Once this is proved, the unremovability of PWMSIG' follows from the correctness of UOPF.

We consider the case of $i = 1$. We first consider the case where $\beta[1] = 0$. In this case, for every $\mathsf{fe.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}, (1, \alpha, u))$, it is computationally infeasible to find a valid proof of NIZK for the statement $(\mathsf{m} = \mathsf{fe.ct}, \gamma, \mathsf{com})$ from the fact that $g(\mathsf{FE.Dec}(\mathsf{fsk}, \mathsf{m})) = g(\alpha) = \gamma$ and NIZK satisfies computational soundness. Note that com statistically binds the witness (fsk, r) used to generate the proofs. This means the result of the estimation computed in $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}, x = \alpha, 1, \epsilon)$ should be close to 0 and especially smaller than $\epsilon/2$, and $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}, x = \alpha, 1, \epsilon)$ outputs 0 if $\beta[1] = 0$. We next consider the case where $\beta[1] = 1$. In this case, $\mathsf{fe.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}, (i, \alpha, u))$ is computationally indistinguishable from a uniformly random message by the ciphertext uniformity of FE and the fact that $\mathsf{FE.Dec}(\mathsf{fsk}, \mathsf{fe.ct}) = u$ distributes uniformly at random. This means the distribution D_i defined in the description of $\mathcal{SearchOutput}$ is computationally indistinguishable from the uniform distribution on the message space if $\beta[1] = 1$. Zhandry [Zha20] showed that if two distributions are computationally indistinguishable, the estimated success probability of a quantum program with respect to one distribution is close to that with respect to the other one. By combining this with the fact that $\widetilde{C}$ is an ϵ-live quantum program, the estimated success probability in $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}, x = \alpha, 1, \epsilon)$ should be close to ϵ and especially larger than $\epsilon/2$. This means $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}, x = \alpha, 1, \epsilon)$ outputs 1 if $\beta[1] = 1$.

The above argument proves $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}, x = \alpha, 1, \epsilon)$ outputs $\beta[1]$ almost deterministically if $\widetilde{C}$ is an ϵ-live quantum program. This allows us to use gentle measurement lemma [Win99] to argue that the quantum program $\widetilde{C}_2$ obtained by uncomputation of $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}, x = \alpha, 1, \epsilon)$ is almost the same quantum program as the original $\widetilde{C}$. By using quantum union bound [Aar06], we can generalize these discussions on the output of $\mathcal{SearchOutput}(\mathsf{vk}, \widetilde{C}_i, x = \alpha, i, \epsilon)$ and quantum program $\widetilde{C}_{i+1}$ obtained by its uncomputation for every $i \in \{1, \cdots, \ell_{\mathsf{out}}\}$. Thus, we can see that the i-th execution of $\mathcal{SearchOutput}$ in $\mathcal{P}[\widetilde{C}]$ with the input α outputs $\beta[i]$ with overwhelming probability for every $i \in \{1, \cdots, \ell_{\mathsf{out}}\}$.

From the above discussions, UOSIG' satisfies unremovability against quantum adversaries.

Proof strategy for privacy and its problem. In the security game of privacy, the adversary can get quantum access to the signing oracle $\mathsf{PWMSIG}'.\mathsf{Sign}(\mathsf{sk}, \cdot)$, where $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{PWMSIG}'.\mathsf{KeyGen}(1^\lambda, \mu)$ for the secret message μ of the adversary's choice. We must ensure that the adversary cannot obtain information of α and β through the quantum oracle access. The knowledge of α and β combined with aux allows the adversary to obtain μ using $\mathsf{UOPF}.\mathcal{E}xtract$, which breaks privacy.

Our strategy towards this is to use the statistical zero-knowledge of NIZK and the security of UOPF. If the statistical zero-knowledge of NIZK guarantees that the quantum access to $\mathsf{PWMSIG}'.\mathsf{Sign}(\mathsf{sk}, \cdot)$ essentially does not leak information of α and β more than black-box access to the point function $f_{\alpha,\beta}$, we can argue that the security of UOPF protects α and β.[11] We require statistical zero-knowledge, not computational one because an adversary can obtain potentially 2^ℓ signatures by just a single quantum query to the oracle, where ℓ is the length of signed messages. We have to ensure that each one of them that is a proof of NIZK does not leak information of α and β.

However, there is a problem in this strategy. The adversary can get information of β from the signing oracle more than the black-box access to $f_{\alpha,\beta}$. Concretely, the adversary can obtain 1-bit information of β "whether $g(\mathsf{FE.Dec}(\mathsf{fsk}, \mathsf{m})) = \gamma$ or not" for any m by querying m to the signing oracle and checking whether the returned signature is valid or not. (Recall that fsk is a functional decryption key for the 1-out-of-2 OT functionality $F[\beta]$.)

Our solution: After-the-fact leakage-resilient unobfuscatable point function. Our solution to the above problem is to require leakage resilience for UOPF. More concretely, we require that the indistinguishability of points holds even if an adversary can obtain after-the-fact leakage information $h(\beta)$ of β. After-the-fact means that the adversary can choose the leakage function h after seeing its challenge input $r \in \{\alpha, R\}$ and aux. The reason why we need it is that the adversary for the privacy of our construction can obtain leakage information of β through the quantum access to the signing oracle after given vk that includes $\gamma = g(\alpha)$ and aux. After-the-fact leakage resilience is defined in the split state model. Namely, in the security game, β is considered as a concatenation of two strings $\beta_1 \in \{0,1\}^{\ell_{\mathsf{out}}}$ and $\beta_2 \in \{0,1\}^{\ell_{\mathsf{out}}}$, and after-the-fact leakage-resilient indistinguishability of points allows an adversary to obtain any local leakage $h_1(\beta_1)$ and $h_2(\beta_2)$. We emphasize that h_1 takes as input only β_1 and h_2 takes as input only β_2. Without this restriction on the locality, the after-the-fact leakage immediately allows the adversary to break the indistinguishability of points.[12] Note that the split state model is used only in the definition of indistinguishability of points. In particular, we do not need to introduce a new syntax for quantum unobfuscatable point functions. Before our work, after-the-fact leakage resilience in the split state model was considered for encryption schemes [HL11]. In fact, we achieve an after-the-fact leakage-resilient unobfuscatable point function using an after-the-fact leakage-resilient encryption scheme.

Final construction. We now present our final construction. In addition to requiring after-the-fact leakage resilience for the quantum unobfuscatable point function UOPF, we apply the following modifications to PWMSIG' and obtain our final scheme PWMSIG.

- We use two instances of FE. Namely, we generate $(\mathsf{fe.pk}_1, \mathsf{fe.msk}_1)$ and $(\mathsf{fe.pk}_2, \mathsf{fe.msk}_2)$, and generate $\mathsf{fsk}_1 \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_1, F[\beta_1])$ and $\mathsf{fsk}_2 \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_2, F[\beta_2])$, where $\beta := \beta_1 \| \beta_2$. According to this change, com is changed into a commitment of fsk_1 and fsk_2, that is, $\mathsf{com} \leftarrow \mathsf{Commit}(\mathsf{fsk}_1 \| \mathsf{fsk}_2; r)$. Moreover, the verification key is set to $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{com}, \mathsf{aux})$ and the corresponding signing key is set to $\mathsf{sk} = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.
- The relation $\mathcal{R}$ is changed so that $(x = (\mathsf{m}, \gamma, \mathsf{com}), w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)) \in \mathcal{R}$ if and only if it holds that
$$\mathsf{com} = \mathsf{Commit}(\mathsf{fsk}_1 \| \mathsf{fsk}_2; r) \wedge g(\mathsf{FE.Dec}(\mathsf{fsk}_1, \mathsf{m})) \neq \gamma \wedge g(\mathsf{FE.Dec}(\mathsf{fsk}_2, \mathsf{m})) \neq \gamma.$$
- $\mathcal{S}earch\mathcal{O}utput$ takes the additional input $d \in \{1, 2\}$ and uses $\mathsf{fe.pk}_d$ to compute $\beta_d[i]$. $\mathcal{P}[\widetilde{C}](x)$ executes $\mathcal{S}earch\mathcal{O}utput$ for every $d \in \{1, 2\}$ and $i \in \{1, \cdots, \ell_{\mathsf{out}}\}$ to compute entire bits of $\beta = \beta_1 \| \beta_2$ when given α.

[11] The verification key vk also has information of α and β, but we can ensure that they do not leak useful information of them that prevents us from using the security of UOPF, by the security of PRG and commitment. We ignore this issue here for simplicity.

[12] Concretely, we consider a leakage function $h[\mu, \mathsf{aux}, r]$ that has μ, aux, and r hardwired. It computes $\mathsf{UOPF.Extract}(f_{r,\beta}, \mathsf{aux})$ and returns 1 if and only if the result is μ. If $r = \alpha$, $h[\mu, \mathsf{aux}, r](\beta)$ is always 1, but if $r = R$, $h[\mu, \mathsf{aux}, r](\beta)$ is not necessarily 1. Thus, we can easily break the indistinguishability of points under even 1-bit leakage of β. Split state model prevents this attack.

We can prove the unremovability of PWMSIG similarly to PWMSIG'. Moreover, thanks to the after-the-fact leakage resilience of UOPF, we can also prove the privacy of PWMSIG following the above strategy using the statistical zero-knowledge of NIZK first and then relying on the security of UOPF. We prove that the after-the-fact leakage resilience against 1-bit leakage for each of β_1 and β_2 is sufficient to complete the proof. By a similar argument, we can prove the unforgeability and strong correctness of PWMSIG. For the formal proofs, see Section 5.

Achieving after-the-fact leakage-resilient unobfuscatable point function. We briefly state how to achieve an after-the-fact leakage-resilient unobfuscatable point function. Our definition requiring indistinguishability of messages and indistinguishability of points abstracts quantum unobfuscatable point function (with auxiliary information) by Alagic, Brakerski, Dulek, Schaffner [ABDS21] using quantum FHE [Mah18, Bra18] and lockable obfuscation [GKW17, WZ17]. By carefully inserting after-the-fact leakage-resilient SKE into the combination of quantum FHE and lockable obfuscation, we obtain after-the-fact leakage-resilient quantum unobfuscatable point function. The existing after-the-fact leakage-resilient SKE schemes rely on non-post-quantum assumptions such as the DDH assumption. Thus, we also propose an after-the-fact leakage-resilient SKE scheme that can be instantiated from post-quantum assumptions like the LWE assumption. In fact, our construction is based on any PKE scheme.

Removing pre-embedded restriction. We convert our pre-embedded white-box watermarking signature scheme into a standard one in a non-black-box way by using a standard EUF-CMA secure signature scheme. See Section 7 and Appendix C for the detail.

Impossibility on the universal copy protection for signatures. A copy-protected signature scheme is a digital signature scheme such that its signing key $\mathpzc{sigk}$ is a quantum state, and it satisfies the security notion that any adversary given the signing key $\mathpzc{sigk}$ cannot generate two quantum programs, both of which can generate valid signatures. We define universal copy protection for signatures as a primitive that turns any signature scheme into a copy-protected one without changing the verification key and algorithm. Such a universal copy protection is preferable to a specific copy-protected signature scheme because it can turn our signing key into copy-protected one without changing the corresponding already published verification key. The separation between EUF-CMA security and EUF-qCMA security by Boneh and Zhandry [BZ13] excludes the existence of universal copy protection for EUF-CMA secure signatures. However, there is still hope that we can have universal copy protection for EUF-qCMA secure signatures.

Unfortunately, we also exclude the existence of universal copy protection for EUF-qCMA secure signatures. More concretely, we provide a counter-example signature scheme such that

- it satisfies EUF-qCMA security,
- if we have a quantum program that can generate valid signatures, we can generate a classical program having the ability to generate valid signatures.

Clearly, any process cannot make the signing key of the scheme into a copy-protected one. We realize the counter-example using our pre-embedded white-box watermarking signature scheme together with standard EUF-qCMA signature scheme and one-way functions. For the detail, see Section 6.

1.4 More on Related Works

Watermarking. Kitagawa and Nishimaki [KN24] achieved watermarking PRFs and PKE against quantum adversaries, and Zhandry [Zha22] achieved collusion-resistant watermarking PKE against quantum adversaries. These watermarking schemes are neither signature schemes nor white-box. White-box traitor tracing [Zha21] can be seen as white-box watermarking public-key encryption. However, Zhandry's work [Zha21] has no implication to white-box watermarking *signatures* and did not study security against quantum adveraries. Yang et al. [YYAS22] present watermarking PRFs with non-black-box extraction. However, they provide neither security proof against quantum adversaries nor privacy.

Robust unobfuscatable functions and impossibility of (quantum) obfuscation. A robust unobfuscatable function [BP15, YYAS22] has the black-box unlearnability and the non-black-box learnability (a.k.a reverse engineering property). The former means that if we have only black-box access to the function, we cannot extract any information about an embedded string in the function. The latter means that if we have the description of the function and it has approximate correctness, we can extract the embedded string. *Approximate* correctness means that obfuscated circuits compute correct outputs on some small (but noticeable) fraction of its inputs.

Pre-embedded white-box watermarking, where a mark is embedded at the function generation phase, is essentially the same as robust unobfuscatable functions. In white-box watermarking, we cannot extract embedded marks by observing function's black-box input and output behavior (corresponding to black-box unlearnability). However, we can extract them from any (adversarially generated) circuit descriptions that approximately preserve the original functionality (corresponding to non-black-box learnability). In addition, *quantum* robust unobfuscatable functions are essentially the same as white-box watermarking against *quantum* adversaries. This is because the former means no QPT algorithm can output a quantum state describing quantum circuit description such that it approximately preserves the original functionality, and we cannot extract embedded information from the circuit description. Here, the approximate property of robust unobfuscatable functions is essential for watermarking since watermarking adversaries output a program with approximate correctness.

Bitansky and Paneth [BP15] constructed publicly verifiable (classical) robust unobfuscatable functions from trapdoor permutations and non-interactive commitments and used them to achieve resettably sound zero-knowledge protocols. Although no previous work pointed out, we can easily convert their *publicly verifiable* robust unobfuscatable functions into a classically robust unobfuscatable signature by using hard-core secret [BP15, Lemma 3.9] and combining standard signatures. We put an embedding string masked by an output of hard-core functions in a verification key. Hence, we can obtain a pre-embedded white-box watermarking signature against *classical* adversaries from their construction.

Alagic et al. [ABDS21] and Ananth and La Placa [AL21] presented (non-robust) quantum unobfuscatable functions. Later, Bitansky, Kellner, and Shmueli [BKS21] constructed quantum unobfuscatable functions based on post-quantum resettbaly-sound zero-knowledge arguments for NP and one-way functions. They are neither publicly verifiable, robust, nor after-the-fact leakage resilient. Their unobfuscatable functions are some sort of point functions or PRGs. Ananth and Kaleoglu [AK22] (implicitly) presented a quantum robust unobfuscatable *point function* to show an impossibility of quantum copy-protection. Their construction is neither signatures, publicly verifiable, nor after-the-fact leakage resilient. Alagic and Fefferman [AF16] showed that it is impossible to obfuscate *quantum* circuits into *reusable* states.

Impossibility of copy-protection. Aaronson [Aar09] observed that achieving copy-protection for black-box learnable functions is impossible. Ananth and La Placa [AL21] presented the impossibility of copy-protection for point functions with statistical correctness. Ananth and Kaleoglu [AK22] presented the impossibility of copy-protection for point functions with approximate correctness (in the classically-accessible random oracle model). None of these results rule out universal copy-protection for signatures.

2 Preliminaries

Notations and conventions. In this paper, standard math or sans serif font stands for classical algorithms (e.g., C or Gen) and classical variables (e.g., x or pk). Calligraphic font stands for quantum algorithms (e.g., $\mathcal{G}en$) and calligraphic font and/or the bracket notation for (mixed) quantum states (e.g., q or $|\psi\rangle$).

Let $[\ell]$ denote the set of integers $\{1,\cdots,\ell\}$, λ denote a security parameter, and $y := z$ denote that y is set, defined, or substituted by z. For a finite set X and a distribution D, $x \leftarrow X$ denotes selecting an element from X uniformly at random, $x \leftarrow D$ denotes sampling an element x according to D. Let $y \leftarrow \mathsf{A}(x)$ and $y \leftarrow \mathcal{A}(\chi)$ denote assigning to y the output of a probabilistic or deterministic algorithm A and a quantum algorithm $\mathcal{A}$ on an input x and χ, respectively. When we explicitly show that A uses randomness r, we write $y \leftarrow \mathsf{A}(x;r)$. PPT and QPT algorithms stand for probabilistic polynomial-time algorithms and polynomial-time quantum algorithms, respectively. Let negl denote a negligible function.

If $\mathcal{X}^{(b)} = \{X_\lambda^{(b)}\}_{\lambda\in\mathbb{N}}$ for $b \in \{0,1\}$ are two ensembles of random variables indexed by $\lambda \in \mathbb{N}$, we say that $\mathcal{X}^{(0)}$ and $\mathcal{X}^{(1)}$ are computationally indistinguishable (denoted by $\mathcal{X}^{(0)} \stackrel{c}{\approx} \mathcal{X}^{(1)}$) if for any polynomial-time distinguisher $\mathcal{D}$, there exists a negligible function $\mathsf{negl}(\lambda)$, such that $\left|\Pr\left[\mathcal{D}(X_\lambda^{(0)}) = 1\right] - \Pr\left[\mathcal{D}(X_\lambda^{(1)}) = 1\right]\right| = \mathsf{negl}(\lambda)$. The statistical distance between $\mathcal{X}^{(0)}$ and $\mathcal{X}^{(1)}$ over a countable set S is defined as $\mathsf{SD}(\mathcal{X}^{(0)}, \mathcal{X}^{(1)}) := \frac{1}{2}\sum_{\alpha \in S}\left|\Pr\left[X_\lambda^{(0)} = \alpha\right] - \Pr\left[X_\lambda^{(1)} = \alpha\right]\right|$. We say that $\mathcal{X}^{(0)}$ and $\mathcal{X}^{(1)}$ are statistically/perfectly indistinguishable (denoted by $\mathcal{X}^{(0)} \stackrel{s}{\approx} \mathcal{X}^{(1)}/\mathcal{X}^{(0)} \stackrel{p}{\approx} \mathcal{X}^{(1)}$) if $\mathsf{SD}(\mathcal{X}^{(0)}, \mathcal{X}^{(1)}) = \mathsf{negl}(\lambda)$ and $\mathsf{SD}(\mathcal{X}^{(0)}, \mathcal{X}^{(1)}) = 0$, respectively. We also say that $\mathcal{X}^{(0)}$ is ϵ-close to $\mathcal{X}^{(1)}$ if $\mathsf{SD}(\mathcal{X}^{(0)}, \mathcal{X}^{(1)}) \leq \epsilon$.

2.1 Quantum information.

We review some basics of quantum information in this subsection.

Definition 2.1 (Shift Distance). *For two distributions D_0, D_1, the shift distance with parameter ϵ, denoted by $\Delta^{\epsilon}_{\mathsf{Shift}}(D_0, D_1)$, is the smallest quantity δ such that for all $x \in \mathbb{R}$:*

$$\Pr[D_0 \leq x] \leq \Pr[D_1 \leq x + \epsilon] + \delta, \qquad \Pr[D_0 \geq x] \leq \Pr[D_1 \geq x - \epsilon] + \delta,$$
$$\Pr[D_1 \leq x] \leq \Pr[D_0 \leq x + \epsilon] + \delta, \qquad \Pr[D_1 \geq x] \leq \Pr[D_0 \geq x - \epsilon] + \delta.$$

For two real-valued measurements $\mathcal{M}$ and $\mathcal{N}$ over the same quantum system, the shift distance between $\mathcal{M}$ and $\mathcal{N}$ with parameter ϵ is

$$\Delta^{\epsilon}_{\mathsf{Shift}}(\mathcal{M}, \mathcal{N}) := \sup_{|\psi\rangle} \Delta^{\epsilon}_{\mathsf{Shift}}(\mathcal{M}(|\psi\rangle), \mathcal{N}(|\psi\rangle)).$$

Definition 2.2 (Quantum Program with Classical Inputs and Outputs [ALL+21]). *A quantum program with classical inputs is a pair of quantum state q and unitaries $\{\boldsymbol{U}_x\}_{x\in[N]}$ where $[N]$ is the domain, such that the state of the program evaluated on input x is equal to $\boldsymbol{U}_x q \boldsymbol{U}_x^{\dagger}$. We measure the first register of $\boldsymbol{U}_x q \boldsymbol{U}_x^{\dagger}$ to obtain an output. We say that $\{\boldsymbol{U}_x\}_{x\in[N]}$ has a compact classical description $\boldsymbol{U}$ when applying $\boldsymbol{U}_x$ can be efficiently computed given $\boldsymbol{U}$ and x.*

Lemma 2.3 (Gentle Measurement Lemma [Win99]). *Suppose a measurement on a mixed state ρ yields a particular outcome with probability $1 - \epsilon$. Then after the measurement, one can recover a state $\tilde{\rho}$ such that $\mathsf{TD}(\tilde{\rho}, \rho) \leq \sqrt{\epsilon}$.*

Lemma 2.4 (Quantum Union Bound [Aar06]). *Let ρ be a mixed state, and let $\Lambda_1, \ldots, \Lambda_T$ be binary outcome measurements. Suppose each Λ_t yields outcome 1 with probability at most ϵ when applied to ρ. Then, if we apply $\Lambda_1, \ldots, \Lambda_T$ in sequence to ρ, the probability that at least one of these measurements yields outcome 1 is at most $T\sqrt{\epsilon}$.*

Measurement Implementation. We review some notions related to measurement implementations used in the definition and the security proof.

Definition 2.5 (Projective Implementation [Zha20]). *Let:*

- *$\mathcal{D}$ be a finite set of distributions over an index set $\mathcal{I}$.*
- *$\mathcal{P} = \{\boldsymbol{P}_i\}_{i\in\mathcal{I}}$ be a POVM.*
- *$\mathcal{E} = \{\boldsymbol{E}_D\}_{D\in\mathcal{D}}$ be a projective measurement with index set $\mathcal{D}$.*

We consider the following measurement procedure.

1. *Measure under the projective measurement $\mathcal{E}$ and obtain a distribution D.*
2. *Output a random sample from the distribution D.*

We say $\mathcal{E}$ is the projective implementation of $\mathcal{P}$, denoted by $\mathsf{ProjImp}(\mathcal{P})$, if the measurement process above is equivalent to $\mathcal{P}$.

Theorem 2.6 ([Zha20, Lemma 1]). *Any binary outcome POVM $\mathcal{P} = (\boldsymbol{P}, \boldsymbol{I} - \boldsymbol{P})$ has a unique projective implementation* $\mathsf{ProjImp}(\mathcal{P})$.

Definition 2.7 (Mixture of Projetive Measurement [Zha20]). *Let $D : \mathcal{R} \to \mathcal{I}$ where $\mathcal{R}$ and $\mathcal{I}$ are some sets. Let $\{(\boldsymbol{P}_i, \boldsymbol{Q}_i)\}_{\in\mathcal{I}}$ be a collection of binary projective measurement. The mixture of projective measurements associated to $\mathcal{R}$, $\mathcal{I}$, D, and $\{(\boldsymbol{P}_i, \boldsymbol{Q}_i)\}_{\in\mathcal{I}}$ is the binary POVM $\mathcal{P}_D = (\boldsymbol{P}_D, \boldsymbol{Q}_D)$ defined as follows.*

$$\boldsymbol{P}_D = \sum_{i\in\mathcal{I}} \Pr[i \leftarrow D(R)]\boldsymbol{P}_i \qquad\qquad \boldsymbol{Q}_D = \sum_{i\in\mathcal{I}} \Pr[i \leftarrow D(R)]\boldsymbol{Q}_i,$$

where R is uniformly distributed in $\mathcal{R}$.

Definition 2.8 (Threshold Implementation [Zha20, ALL+21]). *Let*

- *$\mathcal{P} = (\boldsymbol{P}, \boldsymbol{I} - \boldsymbol{P})$ be a binary POVM*
- *$\mathcal{E}$ be the projective measurement in the first step of the measurement procedure in Definition 2.5.*
- *$t > 0$.*

A threshold implementation of $\mathcal{P}$, denoted by $\mathcal{TI}_t(\mathcal{P})$, is the following measurement procedure.

- *Apply $\mathcal{E}$ to a quantum state and obtain $(p, 1-p)$ as an outcome.*
- *Output 1 if $p \geq t$, and 0 otherwise.*

For any quantum state q, we denote by $\mathrm{Tr}[\mathcal{TI}_t(\mathcal{P})q]$ the probability that the threshold implementation applied to q outputs 1 as Coladangelo et al. did [CLLZ21]. This means that whenever $\mathcal{TI}_t(\mathcal{P})$ appears inside a trace Tr, we treat $\mathcal{TI}_t(\mathcal{P})$ as a projection onto the 1 outcome.

Lemma 2.9 ([ALL+21]). *Any binary POVM $\mathcal{P} = (\boldsymbol{P}, \boldsymbol{I} - \boldsymbol{P})$ has a threshold implementation $\mathcal{TI}_t(\mathcal{P})$ for any t.*

Theorem 2.10 ([Zha20, ALL+21]). *Let*

- *$t > 0$*
- *$\mathcal{P}$ be a collection of projective measurements indexed by some sets*
- *q be an efficiently constructible mixed state*
- *D_0 and D_1 be two efficienctly samplable and computationally indistinguishable distributions over $\mathcal{I}$.*

For any inverse polynomial ϵ, there exists a negligible function δ such that

$$\mathrm{Tr}\left[\mathcal{TI}_{t-\epsilon}(\mathcal{P}_{D_1})q\right] \geq \mathrm{Tr}\left[\mathcal{TI}_t(\mathcal{P}_{D_0})q\right] - \delta,$$

where $\mathcal{P}_{D_{\mathsf{coin}}}$ is the mixture of projective measurements associated to $\mathcal{P}$, D_{coin}, and $\mathsf{coin} \in \{0, 1\}$.

Lemma 2.11 ([ALL+21]). *For any $\epsilon, \delta, t \in (0, 1)$, any collection of projective measurements $\mathcal{P} = \{(\boldsymbol{P}_i, \boldsymbol{I} - \boldsymbol{P}_i)\}_{i\in\mathcal{I}}$ where $\mathcal{I}$ is some index set, and any distribution D over $\mathcal{I}$, there exists a measurement procedure $\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D,t}$ that satisfies the following.*

- *$\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D,t}$ implements a binary outcome measurement.*
- *For all quantum state q,*
  - $\mathrm{Tr}\left[\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D,t-\epsilon} q\right] \geq \mathrm{Tr}[\mathcal{TI}_t(\mathcal{P}_D)q] - \delta$ *and*
  - $\mathrm{Tr}[\mathcal{TI}_{t-\epsilon}(\mathcal{P}_D)q] \geq \mathrm{Tr}\left[\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D,t} q\right] - \delta$.

For simplicity, we denote the probability of the measurement outputting 1 *on* q *by* $\mathrm{Tr}\left[\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D,t}q\right]$.

- *For all qunatum state* q*, let* q' *be the post-measurement state after applying* $\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D,t}$ *on* q*, and obtaining outcome* 1*. Then, it holds* $\mathrm{Tr}[\mathcal{TI}_{t-2\epsilon}(\mathcal{P}_D)q'] \geq 1 - 2\delta$.
- *The expected running time is* $T_{\mathcal{P},D} \cdot \mathrm{poly}(1/\epsilon, 1/\log\delta)$*, where* $T_{\mathcal{P},D}$ *is the combined running time of sampling according to* D*, of mapping* i *to* $(\boldsymbol{P}_i, \boldsymbol{I} - \boldsymbol{P}_i)$*, and of implementing the projective measurement* $(\boldsymbol{P}_i, \boldsymbol{I} - \boldsymbol{P}_i)$.

We can easily obtain the following corollary from Theorem 2.10 and Lemma 2.11.

Corollary 2.12. *Let*

- $\gamma > 0$
- $\mathcal{P}$ *be a collection of projective measurements indexed by some sets*
- q *be an efficiently constructible mixed state*
- D_0 *and* D_1 *be two efficienctly samplable and computationally indistinguishable distributions over* $\mathcal{I}$.

For any inverse polynomial ϵ*, there exists a negligible function* δ *such that*

$$\mathrm{Tr}\left[\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D_1,t-3\epsilon}q\right] \geq \mathrm{Tr}\left[\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},D_0,t}q\right] - 3\delta,$$

where $\mathcal{P}_{D_{\mathsf{coin}}}$ *is the mixture of projective measurements associated to* $\mathcal{P}$*,* D_{coin}*, and* $\mathsf{coin} \in \{0,1\}$.

2.2 One-Way to Hiding (O2H) Lemma

Lemma 2.13 (O2H Lemma [AHU19]). *Let* $G, H : X \to Y$ *be functions,* z *be a string, and* $S \subseteq X$ *be a set such that* $G(x) = H(x)$ *for every* $x \notin S$*. The tuple* (G, H, S, z) *may have arbitrary joint distribution. Let* $\mathcal{A}$ *be a quantum oracle algorithm. Then we have*

$$\left|\Pr\left[\mathcal{A}^{|G\rangle}(z) \to 1\right] - \Pr\left[\mathcal{A}^{|H\rangle}(z) \to 1\right]\right| \leq 2q\sqrt{\Pr\left[x^* \in S : \mathcal{B}^{|H\rangle}(z) \to x^*\right]} \ ,$$

where q *is the number of queries for* G *and* H *made by* $\mathcal{A}$*, and* $\mathcal{B}$ *is a quantum oracle algorithm that picks* $i \leftarrow [q]$*, runs* $\mathcal{A}$ *until just before the* i*-th query made by* $\mathcal{A}$*, measures the* i*-th query, and outputs the measurement result.*

2.3 Standard Cryptographic Tools

Pseudo-Random Function. We define quantum-accessible pseudo-random function.

Definition 2.14 (Quantum-Accessible Pseudo-Random Function). *Let* $\{\mathsf{PRF}_K : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2} \mid K \in \{0,1\}^\lambda\}$ *be a family of polynomially computable functions, where* ℓ_1 *and* ℓ_2 *are some polynomials of* λ*. We say that* PRF *is a quantum-accessible pseudo-random function (QPRF) family if for any QPT adversary* $\mathcal{A}$*, it holds that*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathcal{A}}(\lambda) = \left|\Pr\left[\mathcal{A}^{|\mathsf{PRF}_K(\cdot)\rangle}(1^\lambda) \to 1 \mid K \leftarrow \{0,1\}^\lambda\right] - \Pr\left[\mathcal{A}^{|R(\cdot)\rangle}(1^\lambda) \to 1 \mid R \leftarrow \mathcal{U}\right]\right| \leq \mathsf{negl}(\lambda),$$

where $\mathcal{U}$ *is the set of all functions from* $\{0,1\}^{\ell_1}$ *to* $\{0,1\}^{\ell_2}$.

Theorem 2.15 ([Zha12]). *If there exists a OWF, there exists a QPRF.*

Commitment. We introduce the notion of statistically binding commitment with equivocal mode. This is a relaxation of injective commitment with equivocal mode introduced by Kitagawa and Nishimaki [KN23].

Definition 2.16 (Statistically Binding Commitment with Equivocal Mode). *A statistically binding commitment scheme* Com *with equivocal mode for the message space* $\mathcal{M}$ *and random coin space* $\mathcal{R}$ *is a tuple of four algorithms* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{EqSetup}, \mathsf{Open})$.

- *The setup algorithm* Setup *takes as input a security parameter* 1^λ*, and outputs a commitment key* ck.
- *The commitment algorithm* Commit *takes as input the commitment key* ck*, a message* $m \in \mathcal{M}$*, and a random coin* $r \in \mathcal{R}$*, and outputs a commitment* com.
- *The equivocation setup algorithms* $\mathsf{EqSetup}$ *takes as input a security parameter* 1^λ*, and outputs a commitment key* ck^**, a commitment* com^**, and a trapdoor* td.
- *The open algorithm* Open *takes as input the trapdoor* td*, a message* $m \in \mathcal{M}$*, and a commitment* com^**, and outputs a random coin* $r^* \in \mathcal{R}$.

We say that commitment with equivocal mode is secure if it satisfies the following two properties.

Statistically binding: *We require that*

$$\Pr[\exists m_1, m_2, r_1, r_2 \ \ s.t.\ \ m_1 \neq m_2 \ \ and \ \ \mathsf{Commit}(\mathsf{ck}, m_1; r_1) = \mathsf{Commit}(\mathsf{ck}, m_2, r_2)] = \mathsf{negl}(\lambda),$$

where $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda)$.

Trapdoor Equivocality: *For any message* $m \in \mathcal{M}$*, we have*

$$(\mathsf{ck}, \mathsf{com}, r) \stackrel{\mathsf{c}}{\approx} (\mathsf{ck}^*, \mathsf{com}^*, r^*),$$

where $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda)$*,* $r \leftarrow \mathcal{R}$*,* $\mathsf{com} \leftarrow \mathsf{Commit}(\mathsf{ck}, m; r)$*,* $(\mathsf{ck}^*, \mathsf{com}^*, \mathsf{td}) \leftarrow \mathsf{EqSetup}(1^\lambda)$*, and* $r^* \leftarrow \mathsf{Open}(\mathsf{td}, m, \mathsf{com}^*)$.

We do not explicitly require a hiding property since we do not need it in this work.

Theorem 2.17 ([KN23]). *If there exists an injective OWF with evaluation key generation algorithm, there exists statistically binding commitment with equivocal mode.*

Although Kitagawa and Nishimaki considered the injectivity proeprty [KN23, Definition 2.8] instead of the statistical binding proeprty, their construction immediately implies statistically binding. We can instantiate injective OWF with evaluation key generation algorithm with the LWE assumption [PW11, AKPW13]. See [KN24] for injective OWF with evaluation key generation algorithm.

Public-key encryption.

Definition 2.18 (PKE). *A PKE scheme* PKE *is a tuple of three algorithms* $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$*. Below, let* $\mathcal{X}$ *be the message space of* PKE.

$\mathsf{KG}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{sk})$**:** *The key generation algorithm takes a security parameter* 1^λ*, and outputs a public key* pk *and a secret key* sk.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{m}) \rightarrow \mathsf{ct}$**:** *The encryption algorithm takes a public key* pk *and a message* $\mathsf{m} \in \mathcal{X}$*, and outputs a ciphertext* ct.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \rightarrow \tilde{\mathsf{m}}$**:** *The decryption algorithm is a deterministic algorithm that takes a secret key* sk *and a ciphertext* ct*, and outputs a value* $\tilde{\mathsf{m}}$.

Correctness: *For every* $\mathsf{m} \in \mathcal{X}$, *we have*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) = \mathsf{m} \;\middle|\; \begin{array}{l}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk},\mathsf{m})\end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

Definition 2.19 (Ciphertext Pseudorandomness for PKE). *Let* $\{0,1\}^\ell$ *be the ciphertext space of* PKE. *We define the following experiment* $\mathsf{Exp}^{\mathsf{pr\text{-}ct}}_{\mathsf{PKE},\mathcal{A}}(1^\lambda,\mathsf{coin})$ *between a challenger and an adversary* $\mathcal{A}$.

1. *The challenger generates* $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$. *Then, the challenger sends* pk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *may make polynomially many encryption queries adaptively.* $\mathcal{A}$ *sends* $m \in \mathcal{M}$ *to the challenger. Then, the challenger returns* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk},m)$ *if* $\mathsf{coin} = 0$, *otherwise* $\mathsf{ct} \leftarrow \{0,1\}^\ell$.
3. $\mathcal{A}$ *outputs* $\mathsf{coin}' \in \{0,1\}$. *The challenger outputs* coin'.

We say that PKE *is pseudorandom-secure if for any QPT adversary* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{pr\text{-}ct}}_{\mathsf{PKE},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathsf{Exp}^{\mathsf{pr\text{-}ct}}_{\mathsf{PKE},\mathcal{A}}(1^\lambda,0) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{pr\text{-}ct}}_{\mathsf{PKE},\mathcal{A}}(1^\lambda,1) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Definition 2.20 (Ciphertext Uniformity for PKE). *We say that a PKE scheme* $\mathsf{PKE} = (\mathsf{KG},\mathsf{Enc},\mathsf{Dec})$ *satisfies uniformity if the distribution* $\mathsf{Enc}(\mathsf{pk},U_{\mathcal{M}})$ *is computationally indistinguishable from a uniform distribution even given* sk, *where* $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$ *and* $U_{\mathcal{M}}$ *is the uniform distribution on* $\mathcal{M}$.

Remark 2.21 (On the instantiation of PKE with ciphertext pseudorandomness and uniformity). We can easily realize a PKE scheme satisfying ciphertext pseudorandomness and uniformity. Concretely, a variant of Regev encryption [Reg09] whose ciphertext is of the form $(\boldsymbol{A}\boldsymbol{r}, \mathsf{Round}((\boldsymbol{s}^\intercal\boldsymbol{A} + \boldsymbol{e}^\intercal)\boldsymbol{r}) + b)$ satisfies these two properties, where Round is a function that outputs 1 if the input is larger than $q/2$ and otherwise outputs 0, q is the LWE modulus, and b is the plaintext bit. We use the super polynomial modulus q. Then, this variant satisfies correctness since $\boldsymbol{e}^\intercal \cdot \boldsymbol{r}$ does not affect the result of Round with overwhelming probability. It satisfies ciphertext pseudorandomness due to the LWE assumption and leftover hash lemma. It also satisfies ciphertext uniformity due to uniform randomness of b and the leftover hash lemma.

Definition 2.22 (Signature). *Let* $\mathcal{M}$ *be a message space. A signature scheme for* $\mathcal{M}$ *is a tuple of algorithms* $(\mathsf{Gen},\mathsf{Sign},\mathsf{Vrfy})$ *where:*

$\mathsf{Gen}(1^\lambda) \rightarrow (\mathsf{vk},\mathsf{sk})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a verification key* vk *and a signing key* sk.

$\mathsf{Sign}(\mathsf{sk},\mathsf{m}) \rightarrow \sigma$**:** *The signing algorithm takes as input a signing key* SK *and a message* $\mathsf{m} \in \mathcal{MSG}$ *and outputs a signature* σ.

$\mathsf{Vrfy}(\mathsf{vk},\mathsf{m},\sigma) \rightarrow 1$ **or** 0**:** *The verification algorithm takes as input a verification key* vk, *a message* m *and a signature* σ *and outputs* 1 *to indicate acceptance of the signature and* 0 *otherwise.*

Correctness: *For all* $\lambda \in \mathbb{N}$, $\mathsf{m} \in \mathcal{M}$, $(\mathsf{vk},\mathsf{sk})$ *in the range of* $\mathsf{Gen}(1^\lambda)$, *and* $\sigma \in \mathsf{Sign}(\mathsf{sk},\mathsf{m})$, *we have* $\mathsf{Vrfy}(\mathsf{vk},\mathsf{m},\sigma) = 1$.

Definition 2.23 (EUF-qCMA Security). *Let* $\mathsf{SIG} = (\mathsf{Gen},\mathsf{Sign},\mathsf{Vrfy})$ *be a signature scheme. We define the experiment* $\mathsf{Exp}^{\mathsf{euf\text{-}qcma}}_{\mathsf{SIG},\mathcal{A}}(1^\lambda)$ *between an adversary* $\mathcal{A}$ *and challenger as follows.*

1. *The challenger runs* $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, *and gives* vk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *sends a quantum state* ρ *over registers* R_1 *and* R_2 *to the challenger as a quantum signing query. The challenger picks a signing random coin* r *and applies the map*

$$|a\rangle_{\mathsf{R}_1}|b\rangle_{\mathsf{R}_2} \rightarrow |a\rangle_{\mathsf{R}_1}|b \oplus \mathsf{Sign}(\mathsf{sigk},a;r)\rangle_{\mathsf{R}_2}$$

   to ρ *and returns the resulting state to* $\mathcal{A}$. $\mathcal{A}$ *can send polynomially many queries adaptively. Let* q *be the number of queries made by* $\mathcal{A}$.

3. *At some point, $\mathcal{A}$ outputs $q+1$ pairs of message and signature $(\mathsf{m}_i, \sigma_i)_{i\in[q+1]}$ to the challenger.*

4. *The experiment outputs 1 if $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}_i, \sigma_i) = 1$ for every $i \in [q+1]$.*

We say that SIG *EUF-qCMA security if, for any QPT adversary $\mathcal{A}$, it holds that*

$$\mathsf{Adv}^{\mathsf{euf\text{-}qcma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{Exp}^{\mathsf{euf\text{-}qcma}}_{\mathsf{SIG},\mathcal{A}}(1^\lambda) = 1\right] = \mathsf{negl}(\lambda).$$

Non-interactive zero-knowledge. Let $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ be a polynomial time recognizable binary relation. For $(x, w) \in \mathcal{R}$, we call x as the statement and w as the witness. Let $\mathcal{L}$ be the corresponding NP language $\mathcal{L} = \{x \mid \exists w \text{ s.t. } (x,w) \in \mathcal{R}\}$. Below, we define a non-interactive zero-knowledge proofs for NP languages.

Definition 2.24 (NIZK Arguments (Syntax)). *A non-interactive zero-knowledge (NIZK) argument* NIZK *for the relation $\mathcal{R}$ consists of PPT algorithms* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vrfy})$.

$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$**:** *The setup algorithm takes as input the security parameter 1^λ and outputs a common reference string* crs.

$\mathsf{Prove}(\mathsf{crs}, x, w) \to \pi$**:** *The proving algorithm takes as input a common reference string* crs*, a statement x, and a witness w and outputs a proof π.*

$\mathsf{Vrfy}(\mathsf{crs}, x, \pi) \to 1/0$**:** *The verification algorithm takes as input a common reference string, a statement x, and a proof π and outputs 1 to indicate acceptance of the proof and 0 otherwise.*

Definition 2.25 (Statistical NIZK Argument). *A statistical NIZK argument* NIZK *must satisfy the following requirements.*

Completeness: *For all pairs $(x, w) \in \mathcal{R}$, if we run $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$, then we have*

$$\Pr[\mathsf{Vrfy}(\mathsf{crs}, x, \pi) = 1 \mid \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)] = 1.$$

Adaptive Exclusive Soundness: *For all QPT adversaries $\mathcal{A}$* outputting only $x \notin \mathcal{L}$, *if we run $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$, then we have*

$$\Pr\left[\mathsf{Vrfy}(\mathsf{crs}, x, \pi) = 1 \mid (x, \pi) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs})\right] = \mathsf{negl}(\lambda).$$

(Strong) Statistical Zero-Knowledge: *There exists a QPT simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that, for all unbounded adversaries $\mathcal{A}$, if we run $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\widetilde{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{Sim}_1(1^\lambda)$, then we have*

$$\left|\Pr\left[\mathcal{A}^{O_0(\mathsf{crs},\cdot,\cdot)}(1^\lambda, \mathsf{crs}) = 1\right] - \Pr\left[\mathcal{A}^{O_1(\widetilde{\mathsf{crs}},\mathsf{td},\cdot,\cdot)}(1^\lambda, \widetilde{\mathsf{crs}}) = 1\right]\right| = \mathsf{negl}(\lambda),$$

where $O_0(\mathsf{crs}, x, w)$ outputs $\mathsf{Prove}(\mathsf{crs}, x, w)$ if $(x, w) \in \mathcal{R}$ and $\perp$ otherwise, and $O_1(\widetilde{\mathsf{crs}}, \mathsf{td}, x, w)$ outputs $\mathsf{Sim}_2(\widetilde{\mathsf{crs}}, \mathsf{td}, x)$ if $(x, w) \in \mathcal{R}$ and $\perp$ otherwise. If $\mathcal{A}$ is allowed to send super-polynomially many queries to O_0 and O_1, we say strong statistical zero-knowledge. (We say strong statistical zero-knowledge with q queries when we specify the number of queries.)

Theorem 2.26 ([PS19, FR21]). *If the LWE assumption holds, there exists a statistical NIZK arguemnt system for all* NP *in the common random string model.*

Theorem 2.27 ([PS19, FR21]). *If the LWE assumption holds, there exists a strong statistical NIZK arguemnt system for all* NP *in the common random string model.*

Statistical zero-knowledge trivially implies computational zero-knowledge.

Remark 2.28 (On strong statistical ZK). Fischlin and Rohrbach [FR21, Section 5.2 in eprint ver.] used a lattice-specific variant of the well-known Feige-Lapidot-Shamir transformation [FLS99] to obtain multi-theorem statistical ZK from single-theorem statistical ZK. We use the witness indistinguishability property (implied by ZK) q times to change each answer from the zero-knowledge oracle O_0 one-by-one in the transformation where q is the number of the queries. If the advantage of the underlying single-theorem statistical ZK is sub-exponentially small (we can achieve this using long security parameters), we can apply the witness indistinguishability super-polynomially many times by complexity leveraging with an appropriate parameter setting. Hence, we can obtain Theorem 2.27 (i.e., statistical ZK holds even with super-polynomially many queries) from the statistical NIZK by Fischlin and Rohrbach [FR21] and Peikert and Shiehian [PS19].

Remark 2.29 (On adaptive soundness of statistical NIZK). Fischlin and Rohrbach consider two types of adaptive soundness. One is adaptive penalizing soundness, which is widely used in NIZK definitions. The other is adaptive exclusive soundness, which considers only adversaries that outputs only false statements given no matter what CRS. Obviously, adaptive exclusive soundness is weaker than adaptive penalized soundness. The well-known impossibility of adaptively sound statistical NIZK arguments [AF07, Pas13] holds only for adaptive *penalizing* soundness as observed by Fischlin and Rohrbach [FR21]. Canetti et al. [CCH$^+$19][13] claims that their statistical NIZK is non-adaptively sound and does not achieve adaptive (penalizing) soundness [CLW18, Section 1.1.2]. However, it is easy to observe that their statistical NIZK achieves adaptive *exclusive* soundness. As Canetti et al. [CLW18, Footnote 13] observed, the reason why the adaptive penalizing soundness does not hold for their statistical NIZK is that we cannot efficiently check a part of the winning condition (the statment output by the adversary is false) in the reduction to the CRS indistinguishability. However, if adversaries outputs only false statements, we do not need to check a statement is false. Hence, their reduction work in the adaptive *exclusive* soundness. Thus, we can obtain Theorem 2.26 from the known results.

When we use NIZK with adaptive exclusive soundness as a building block of some cryptographic scheme, a reduction to adaptive exclusive soundness (that is, an adversary for adaptive exclusive soundness) must check that a statement is not in the language by itself as we see in Section 5.5.

Lockable obfuscation. We introduce the notion of lockable obfuscation [GKW17, WZ17].

Definition 2.30 (Lockable Obfuscation). *A lockable obfuscation is a tuple of PPT algorithms* $(\mathsf{LObf}, \mathsf{Eval})$ *with a class of circuits* $\mathcal{F}$*, an input space* $\mathcal{X}$*, and a message space* $\mathcal{M}$*.*

$\mathsf{LObf}(1^\lambda, C, \mathsf{lock}, m)$**:** *The obfuscation algorithm takes as input a security parameter* 1^λ*, a circuit* $C \in \mathcal{F}$*, a lock string* $\mathsf{lock} \in \{0,1\}^{p(\lambda)}$*, and a message* $m \in \mathcal{M}$*, and outputs an obfuscated circuit* $\widetilde{P}$*.*

$\mathsf{Eval}(\widetilde{P}, x)$**:** *The evaluation algorithm takes as input a obfuscated circuit* $\widetilde{P}$ *and an input* $x \in \mathcal{X}$*, and outputs a string* m' *or* $\perp$*. We frequently use* $\widetilde{P}(x)$ *to denote* $\mathsf{Eval}(\widetilde{P}, x)$ *for ease of notations.*

Evaluation correctness: *For any* $\lambda \in \mathbb{N}, P \in \mathcal{F}, x \in \mathcal{X}$*,* $\mathsf{lock} \in \{0,1\}^{p(\lambda)}$*, and* $m \in \mathcal{M}$ *such that* $P(x) = \mathsf{lock}$*, we have*

$$\Pr\left[\mathsf{Eval}(\widetilde{P}, x) = m \;\middle|\; \widetilde{P} \leftarrow \mathsf{LObf}(1^\lambda, P, \mathsf{lock}, m)\right] = 1.$$

There exists a negligible function $\mathsf{negl}(\cdot)$ *such that for any* $P \in \mathcal{F}, x \in \mathcal{X}$*,* $\mathsf{lock} \in \{0,1\}^{p(\lambda)}$*, and* $m \in \mathcal{M}$ *such that* $P(x) \neq \mathsf{lock}$*, we have*

$$\Pr\left[\mathsf{Eval}(\widetilde{P}, x) = \perp \;\middle|\; \widetilde{P} \leftarrow \mathsf{LObf}(1^\lambda, P, \mathsf{lock}, m)\right] = 1 - \mathsf{negl}(\lambda).$$

Definition 2.31 (Simulation Security of Lockable Obfuscation). *A lockable obfuscation scheme* $\Sigma_{\mathsf{LO}} = (\mathsf{LObf}, \mathsf{Eval})$ *for a class of circuits* $\mathcal{F}$*, an input space* $\mathcal{X}$*, and a message space* $\mathcal{M}$ *is said to be secure if there exists an*

[13] The NIZK construction by Peikert and Shiehian [PS19] is based on the NIZK construction by Canetti et al. [CCH$^+$19]. More specifically, Peikert and Shiehian instantiated the correlated intractable hash in the work by Canetti et al. with the LWE assumption.

algorithm Sim *such that for any QPT adversary* $\mathcal{A}$*, the following holds*

$$\left| \Pr\left[\mathcal{A}(\widetilde{P}^{(b)}) = b \;\middle|\; \begin{array}{c} (P \in \mathcal{F}, m \in \mathcal{M}) \leftarrow \mathcal{A}(1^\lambda) \\ \mathsf{lock} \leftarrow \{0,1\}^{p(\lambda)}, b \leftarrow \{0,1\} \\ \widetilde{P}^{(0)} \leftarrow \mathsf{LObf}(1^\lambda, P, \mathsf{lock}, m) \\ \widetilde{P}^{(1)} \leftarrow \mathsf{Sim}(1^\lambda, 1^{|P|}, 1^{|m|}) \end{array} \right] - \frac{1}{2} \right| = \mathsf{negl}(\lambda).$$

Theorem 2.32 ([GKW17, WZ17]). *If the LWE assumption holds, there exists lockable obfuscation.*

(Quantum) fully homomorphic encryption.

Definition 2.33 (Quantum Fully Homomorphic Encryption with Classical Ciphertexts [Mah18, Bra18]). *A quantum fully homomorphic encryption (QFHE) with classical ciphertexts is a tuple of four algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *with a class of circuits* $\mathcal{C}$*.*

$\mathsf{Gen}(1^\lambda)$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a public key* pk *and a secret key* sk*. This is a PPT algorithm.*

$\mathsf{Enc}(\mathsf{pk}, x)$**:** *The encryption algorithm takes as input a public key* pk *and a plaintext* $x \in \{0,1\}$*, and outputs a ciphertext* ct*. For multi-bit message* $x \in \{0,1\}^\ell$*, we write* $\mathsf{Enc}(\mathsf{pk}, x)$ *to denote the bit-by-bit encryption* $(\mathsf{Enc}(\mathsf{pk}, x_1), \ldots, \mathsf{Enc}(\mathsf{pk}, x_\ell))$*. This algorithm is PPT.*

$\mathcal{Eval}(\mathsf{pk}, \mathsf{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_{\ell_{\mathsf{in}}})$**:** *The evaluation algorithm takes as input a public key* pk*, a (quantum) circuit* $\mathsf{C} \in \mathcal{C}$*, ciphertexts* $\mathsf{ct}_1, \ldots, \mathsf{ct}_{\ell_{\mathsf{in}}}$ *where* ℓ_{in} *denotes the input length of the circuit* C*, and outputs a ciphertext* ct_{C} *(this consists of* ℓ_{out} *ciphertexts where* ℓ_{out} *denotes the output length of* C*). This is a QPT algorithm.*

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:** *The decryption algorithm takes as input a secret key* sk *and a ciphertext* ct*, and outputs a message* x' *or* $\perp$*.*

In the case of classical FHE (i.e., $\mathcal{C} = \mathsf{P}/\mathsf{poly}$*), all algorithms are PPT.*

Definition 2.34 (Compactness). *A classical FHE is compact if its decryption circuit is independent of the evaluated circuit.*

Definition 2.35 (Full Homomorphism). *An FHE (or QFHE with classical ciphertexts) scheme is fully homomorphic if for any* $\mathsf{C} \in \mathcal{C}, x = (x_1, \ldots, x_{\ell_{\mathsf{in}}}) \in \{0,1\}^{\ell_{\mathsf{in}}}$*,*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}_C) = C(x) \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda) \\ \mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, x_i) \\ \mathsf{ct}_C \leftarrow \mathsf{Eval}(\mathsf{pk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_{\ell_{\mathsf{in}}}) \end{array} \right] = 1 - \mathsf{negl}(\lambda).$$

The scheme is leveled fully homomorphic if Gen *takes* 1^d *as additional input, and can only evaluate depth* d *circuits. In the QFHE with classical ciphertexts case, we use* $\mathcal{Eval}$ *instead of* Eval*.*

Definition 2.36 (Security of QFHE). *A QFHE scheme with classical ciphertexts and a class of circuits* $\mathcal{C}$ *is said to be IND-CPA secure if for any QPT adversary* $\mathcal{A}$ *and* $x_0, x_1 \in \{0,1\}^\ell$*, the following holds:*

$$\Pr\left[\mathcal{A}(1^\lambda, \mathsf{pk}, \mathsf{ct}) = 1 \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x_0) \end{array} \right] - \Pr\left[\mathcal{A}(1^\lambda, \mathsf{pk}, \mathsf{ct}) = 1 \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x_1) \end{array} \right] = \mathsf{negl}(\lambda).$$

We can consider a secret-key variant, where $\mathsf{Gen}(1^\lambda)$ outputs only a secret-key sk and Enc uses sk instead of pk.

Theorem 2.37 ([Mah18, Bra18]). *If the LWE assumption holds, and assume circular security, there exists QFHE.*

Functional encryption.

Definition 2.38 (Functional Encryption). *An FE scheme* FE *is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{SimEnc})$.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{msk})$**:** *The setup algorithm takes a security parameter* 1^λ *and outputs a public key* pk *and master secret key* msk.

$\mathsf{KG}(\mathsf{msk}, f) \to \mathsf{fsk}$**:** *The key generation algorithm* KG *takes a master secret key* msk *and a function* f*, and outputs a functional decryption key* fsk.

$\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$**:** *The encryption algorithm takes a public key* pk *and an input* x*, and outputs a ciphertext* ct.

$\mathsf{Dec}(\mathsf{fsk}, \mathsf{ct}) \to y$**:** *The decryption algorithm takes a functional decryption key* fsk *and a ciphertext* ct*, and outputs* y*.*

$\mathsf{SimEnc}(\mathsf{pk}, f, y)$**:** *The simulated encryption algorithm takes a public key* pk*, a function* f*, and a value* y*, and output a simulated ciphertext* ct.

Correctness: *We require we have that*

$$\Pr\left[\mathsf{Dec}(\mathsf{fsk}, \mathsf{ct}) = f(x) \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda), \\ \mathsf{fsk} \leftarrow \mathsf{KG}(\mathsf{msk}, f), \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

Definition 2.39 (1-Bounded Simulation Security). *We formalize the experiment* $\mathsf{Exp}^{\mathsf{1\text{-}ind}}_{\mathsf{FE},\mathcal{A}}(1^\lambda, \mathsf{coin})$ *between an adversary* $\mathcal{A}$ *and a challenger for a FE scheme* FE *as follows:*

1. *The challenger runs* $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *sends* f *and* x*. The challenger generates* $\mathsf{fsk} \leftarrow \mathsf{KG}(\mathsf{msk}, f)$*. Also, the challenger generates* $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, x)$ *if* $\mathsf{coin} = 0$ *and otherwise generate* $\mathsf{ct}^* \leftarrow \mathsf{SimEnc}(\mathsf{pk}, f, f(x))$*. The challenger sends* fsk *and* ct^* *to* $\mathcal{A}$.
3. $\mathcal{A}$ *outputs a guess* coin' *for* coin*. The challenger outputs* coin'.

We say that FE *is* 1*-bounded simulation secure if, for any QPT* $\mathcal{A}$*, it holds that*

$$\mathsf{Adv}^{\mathsf{1\text{-}sim}}_{\mathsf{FE},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{1\text{-}sim}}_{\mathsf{FE},\mathcal{A}}(1^\lambda, 0) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{1\text{-}sim}}_{\mathsf{FE},\mathcal{A}}(1^\lambda, 1) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Definition 2.40 (Ciphertext Uniformity for FE). *We say that* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{SimEnc})$ *satisfies ciphertext uniformity if for every* f*, the distribution* $\mathsf{SimEnc}(\mathsf{pk}, f, U_m)$ *is computationally indistinguishable from the uniform distribution even given* $\mathsf{fsk} \leftarrow \mathsf{KG}(\mathsf{msk}, f)$*, where* $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and* U_m *is the uniform distribution on* $\{0,1\}^m$.

We prove the following theorem in Appendix A.

Theorem 2.41. *If there exists a PKE scheme that satisfies ciphertext pseudorandomness and ciphertext uniformity, there exists FE satisfying* 1*-bounded simulation security and ciphertext uniformity for* 1*-out-of-*2 *OT functionality,*

$$F[\beta](i, x_0, x_1) = x_{\beta[i]}.$$

Since we can realize a PKE scheme satisfying ciphertext pseudorandomness and ciphertext uniformity from the LWE assumption, we obtain the following theorem.

Theorem 2.42. *Assuming the LWE assumption, there exists FE satisfying* 1*-bounded simulation security and ciphertext uniformity for* 1*-out-of-*2 *OT functionality.*

3 After-the-Fact Leakage-Resilient Quantum Unobfuscatable Point Function

In this section, we introduce the notion of after-the-fact leakage-resilient quantum unobfuscatable point function. This primitive is an essential building block of our quantum robust unobfuscatable signature scheme described in Section 5.

3.1 Definition

We present the definition of quantum unobfuscatable point function and after-the-fact leakage-resilience for it.

Definition 3.1 (Quantum Unobfuscatable Point Function). *A quantum unobfuscatable point function* UOPF *for secret message space* $\mathcal{SS}$*, input space* $\{0,1\}^{\ell_{\mathsf{in}}}$*, and output space* $\{0,1\}^{\ell_{\mathsf{out}}}$ *is a tuple of two algorithms* $(\mathsf{Gen}, \mathcal{Extract})$*.*

$\mathsf{Gen}(1^\lambda, \mu) \to (f_{\alpha,\beta}, \mathsf{aux})$**:** *The generation algorithm takes as input the security parameter and a secret message* $\mu \in \mathcal{SS}$*, and outputs a description of point function* $f_{\alpha,\beta}$ *and an auxiliary information* aux*.*

$\mathcal{Extract}(\mathcal{C}, \mathsf{aux}) \to \mu'$**:** *The extraction algorithm takes as input a quantum circuit with classical input and output* $\mathcal{C}$ *and an auxiliary information* aux*, and outputs* $\mu' \in \mathcal{SS} \cup \{\perp\}$*.*

Correctness *Let* $\mu \in \mathcal{SS}$ *and* $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{Gen}(1^\lambda, \mu)$*. Then, it satisfies the followings.*

- *For any quantum circuit with classical input and output* $\widetilde{\mathcal{C}}$*, we have* $\Pr\left[\mathcal{Extract}(\widetilde{\mathcal{C}}, \mathsf{aux}) \notin \{\mu, \perp\}\right] = \mathsf{negl}(\lambda)$*.*
- *For any quantum circuit with classical input and output* $\widetilde{\mathcal{C}}$ *that maps* α *to* β *with probability* $1 - \mathsf{negl}(\lambda)$*, we have* $\Pr\left[\mathcal{Extract}(\widetilde{\mathcal{C}}, \mathsf{aux}) = \mu\right] = 1 - \mathsf{negl}(\lambda)$*.*

Indistinguishability of messages *For any* $\mu_0, \mu_1 \in \mathcal{SS}$*, we have*

$$\left|\Pr\left[\mathcal{A}(1^\lambda, \mathsf{aux}) = 1 \;\middle|\; (f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{Gen}(1^\lambda, \mu_0)\right] - \Pr\left[\mathcal{A}(1^\lambda, \mathsf{aux}) = 1 \;\middle|\; (f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{Gen}(1^\lambda, \mu_1)\right]\right| = \mathsf{negl}(\lambda).$$

Indistinguishability of points *For any* $\mu \in \mathcal{SS}$*, we have*

$$\left|\Pr\left[\mathcal{A}(1^\lambda, \alpha, \mathsf{aux}) = 1 \;\middle|\; (f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{Gen}(1^\lambda, \mu)\right] - \Pr\left[\mathcal{A}(1^\lambda, R, \mathsf{aux}) = 1 \;\middle|\; \begin{array}{l}(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{Gen}(1^\lambda, \mu)\\ R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}\end{array}\right]\right| = \mathsf{negl}(\lambda).$$

Definition 3.2 (ℓ-After-the-Fact Leakage-Resilient Indistinguishability of Points). *Let* $\mathsf{UOPF} = (\mathsf{Gen}, \mathsf{Extract})$ *be an unobfuscatable point function for the secret message space* $\mathcal{SS}$*, input space* $\{0,1\}^{\ell_{\mathsf{in}}}$*, and output space* $\{0,1\}^{2\ell_{\mathsf{out}}}$*. We define the experiment* $\mathsf{Exp}^{\mathsf{atf\text{-}lr\text{-}uopf}}_{\mathsf{UOPF},\mathcal{A}}(1^\lambda, \ell, \mathsf{coin})$ *as follows.*

1. *The adversary* $\mathcal{A}$ *sends* μ *to the challenger.*
2. *The challenger generates* $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{Gen}(1^\lambda, \mu)$ *and* $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$*. The challenger sends* (α, aux) *if* $\mathsf{coin} = 0$ *and otherwise* (R, aux)
3. $\mathcal{A}$ *sends leakage functions* h_1, h_2 *of output length* ℓ*. The challenger returns* $h_1(\beta_1)$ *and* $h_2(\beta_2)$*, where* $\beta = \beta_1 \| \beta_2$*,* $\beta_1 \in \{0,1\}^{\ell_{\mathsf{out}}}$*, and* $\beta_2 \in \{0,1\}^{\ell_{\mathsf{out}}}$*.*
4. $\mathcal{A}$ *outputs* $\mathsf{coin}' \in \{0,1\}$*. The challenger outputs* coin'*.*

We say that UOPF *is* ℓ*-after-the-fact leakage-resilient if for any QPT* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{atf\text{-}lr\text{-}uopf}}_{\mathsf{UOPF},\mathcal{A}}(\lambda, \ell) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{atf\text{-}lr\text{-}uopf}}_{\mathsf{UOPF},\mathcal{A}}(1^\lambda, \ell, 0) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{atf\text{-}lr\text{-}uopf}}_{\mathsf{UOPF},\mathcal{A}}(1^\lambda, \ell, 1) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Theorem 3.3. *If the LWE assumption holds and there exists QFHE, there exists* 2*-after-the-fact leakage resilient quantum unobfuscatable point function.*

We prove this theorem in Appendix B.

4 Definition of White-Box Watermarking Signature

In this section, we introduce definitions for watermarking signatures.

4.1 Pre-Embedded White-Box Watermarking Signature

We first consider pre-embedded white-box watermarking signatures, where we need to embed a mark when we generate a key pair.

Definition 4.1 (Pre-Embedded Watermarking Signature). *A pre-embedded watermarking signature* PWMSIG *for the mark space* $\mathcal{MS}$ *and plaintext space* $\mathcal{MK}$ *is a tuple of four algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathcal{E}xtract)$.

$\mathsf{KeyGen}(1^\lambda, \mu) \to (\mathsf{vk}, \mathsf{sk})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and a mark* μ *and outputs a verification key* vk *and a signing key* sk.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{m}) \to \sigma$**:** *The signing algorithm takes as input a signing key* sk *and a message* m *and outputs a signature* σ. *We require that this algorithm is deterministic.*

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \sigma) \to 0/1$**:** *The verification algorithm takes as input a verification key* vk *and a signature* σ *and outputs* 0 *or* 1.

$\mathcal{E}xtract(\mathsf{vk}, \widetilde{\mathcal{C}}', \epsilon) \to \mu'$**:** *The extraction algorithm takes as input a verification key* vk, *a circuit* $\widetilde{\mathcal{C}}'$, *and a parameter* ϵ, *and outputs* $\mu' \in \mathcal{MK} \cup \{\mathsf{unmarked}\}$.

Verification Correctness: *For any message* $\mathsf{m} \in \mathcal{MS}$ *and mark* $\mu \in \mathcal{MK}$, *we have* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \mathsf{Sign}(\mathsf{sk}, \mathsf{m})) = 1$, *where* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, \mu)$.

Definition 4.2 (Strong Correctness of Marked Keys). *We define the game* $\mathsf{Expt}^{\mathsf{scorrect}}_{\mathcal{A},\mathsf{PWMSIG}}(1^\lambda)$ *as follows.*

1. *Given* 1^λ *as the initial input,* $\mathcal{A}$ *sends* $\mu \in \mathcal{MK}$ *to the challenger. The challenger generates* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, \mu)$ *and sends* vk *to* $\mathcal{A}$. $\mathcal{A}$ *can get access to the following oracle.*

   $O_{\mathtt{sign}}(\mathsf{m})$**:** *On input* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$.

2. $\mathcal{A}$ *outputs* $\mathsf{m}^* \in \mathcal{MS}$. *The challenger outputs* 1 *if* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}^*)) = 0$ *and otherwise outputs* 0.

We say that PWMSIG *satisfies strong correctness of marked keys if for every QPT* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{scorrect}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda) = \Pr\Big[\mathsf{Expt}^{\mathsf{scorrect}}_{\mathcal{A},\mathsf{PWMSIG}}(1^\lambda) = 1\Big] \leq \mathsf{negl}(\lambda).$$

Definition 4.3 (Unforgeability). *We define the game* $\mathsf{Exp}^{\mathsf{euf\text{-}cma}}_{\mathcal{A},\mathsf{PWMSIG}}(\lambda)$ *as follows.*

1. *Given* 1^λ *as the initial input,* $\mathcal{A}$ *sends* μ *to the challenger. The challenger generates* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, \mu)$, *and sends* vk *to* $\mathcal{A}$. $\mathcal{A}$ *can get access to the following oracle.*

   $O_{\mathtt{sign}}(\mathsf{m})$**:** *On input* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$. *Let* $\mathcal{Q}$ *be the set of the inputs received from* $\mathcal{A}$.

2. $\mathcal{A}$ *outputs* (m^*, σ^*). *If* $\mathsf{m}^* \notin \mathcal{Q}$, *the challenger outputs* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \sigma^*)$. *Otherwise* 0.

We say that WMSIG *satisfies unforgeability if for every QPT* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda) := \Pr\Big[\mathsf{Exp}^{\mathsf{euf\text{-}cma}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda) = 1\Big] \leq \mathsf{negl}(\lambda).$$

Definition 4.4 (Unremovability). *Let* $\epsilon \geq 0$. *We define the game* $\mathsf{Expt}^{\mathsf{urmv}}_{\mathcal{A},\mathsf{PWMSIG}}(1^\lambda, \epsilon)$ *as follows.*

1. *Given* 1^λ *as the initial input,* $\mathcal{A}$ *sends* $\mu \in \mathcal{MK}$ *to the challenger. The challenger generates* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, \mu)$ *and sends* $(\mathsf{vk}, \mathsf{sk})$ *to the adversary* $\mathcal{A}$.

2. *$\mathcal{A}$ outputs a "potentially obfuscated" quantum circuit $\widetilde{C} = (q, \boldsymbol{U})$, where $\widetilde{C}$ is a quantum program with classical inputs and outputs $\boldsymbol{U}$ is a compact classical description of $\{\boldsymbol{U}_{\mathsf{m}}\}_{\mathsf{m}\in\mathcal{MS}}$.*

Let also $\boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}$ be the unitary that maps $|a\rangle|b\rangle$ to $|a\rangle|b \oplus \mathsf{Vrfy}(\mathsf{vk},\mathsf{m},a)\rangle$. We also let $\mathcal{P} = (\boldsymbol{P}_{\mathsf{m}}, \boldsymbol{Q}_{\mathsf{m}})_{\mathsf{m}}$ be a collection of binary outcome projective measurements, where

$$\boldsymbol{P}_{\mathsf{m}} = \boldsymbol{U}_{\mathsf{m}}^{\dagger}\boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}^{\dagger}(\boldsymbol{I}\otimes|1\rangle\langle 1|)\boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}\boldsymbol{U}_{\mathsf{m}} \quad \textit{and} \quad \boldsymbol{Q}_{\mathsf{m}} = \boldsymbol{I} - \boldsymbol{P}_{\mathsf{m}}.$$

Moreover, we let $U_{\mathcal{MS}}$ be the uniform distribution over $\mathcal{MS}$. We consider the following events.

Live: *When applying the measurement $\mathcal{TI}_{\epsilon}(\mathcal{P}_{U_{\mathcal{MS}}})$ to q (and ancilla), we obtain the outcome 1, where $\mathcal{P}_{U_{\mathcal{MS}}}$ is a mixture of $\mathcal{P}$ with respect to $U_{\mathcal{MS}}$.*

$\mathsf{GoodExt}$: *When Computing $\mu' \leftarrow \mathcal{E}xtract(\mathsf{vk}, \widetilde{C}, \epsilon)$, it holds that $\mu' \neq \mathsf{unmarked}$.*

BadExt: *When Computing $\mu' \leftarrow \mathcal{E}xtract(\mathsf{vk}, \widetilde{C}, \epsilon)$, it holds that $\mu' \notin \{\mu, \mathsf{unmarked}\}$.*

We say that PWMSIG *satisfies unremovability if for every $\epsilon > 0$ and QPT $\mathcal{A}$, we have*

$$\Pr[\mathsf{BadExt}] \leq \mathsf{negl}(\lambda) \quad \textit{and} \quad \Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda).$$

Intuitively, $(\boldsymbol{P}_{\mathsf{m}}, \boldsymbol{Q}_{\mathsf{m}})$ is a projective measurement that feeds m to $\widetilde{C}$ and checks whether the outcome passes $\mathsf{Vrfy}(\mathsf{vk},\cdot)$ or not (and then uncomputes). Then, $\mathcal{P}_{U_{\mathcal{MS}}}$ can be seen as POVMs that results in 0 with the probability that $\widetilde{C}$ outputs a valid signature for a randomly chosen $\mathsf{m} \leftarrow \mathcal{MS}$. This definition says that any QPT algorithm (adversary) fails to obfuscate the signing function (key) as long as the algorithm outputs a "Live" quantum program.

Remark 4.5. Our definition follows the unremovability definition (for watermarking PRFs) by Kitagawa and Nishimaki [KN24], which originates from the traceability definition of traceable PRFs by Goyal et al. [GKWW21].

Definition 4.6 (Privacy). *We define the game $\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{A},\mathsf{PWMSIG}}(1^{\lambda})$ as follows.*

1. *Given 1^{λ} as the initial input, $\mathcal{A}$ sends $(\mu_0, \mu_1) \in \mathcal{MK}^2$ to the challenger. The challenger picks $\mathsf{coin} \leftarrow \{0,1\}$, generates $(\mathsf{vk}_{\mathsf{coin}}, \mathsf{sk}_{\mathsf{coin}}) \leftarrow \mathsf{KeyGen}(1^{\lambda}, \mu_{\mathsf{coin}})$, and sends $\mathsf{vk}_{\mathsf{coin}}$ to $\mathcal{A}$. $\mathcal{A}$ can get access to the following oracle.*

   $O_{\mathtt{qsign}}$: *On input a quantum state ρ over registers R_1 and R_2, it applies the signing unitary that maps $|a\rangle_{\mathsf{R}_1}|b\rangle_{\mathsf{R}_2}$ to $|a\rangle_{\mathsf{R}_1}|b \oplus \mathsf{Sign}(\mathsf{sk}_{\mathsf{coin}}, a)\rangle_{\mathsf{R}_2}$ to ρ and returns the resulting state. Recall that* Sign *is deterministic.*

2. *$\mathcal{A}$ outputs $\mathsf{coin}' \in \{0,1\}$. The challenger outputs coin'.*

We say that PWMSIG *satisfies privacy if for every QPT $\mathcal{A}$, we have*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{A},\mathsf{PWMSIG}}(1^{\lambda}, 0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{A},\mathsf{PWMSIG}}(1^{\lambda}, 1) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

In Definition 4.6, adversaries try to distinguish whether superpositions of signatures are generated by sk_0 or sk_1 by observing the *black-box input and output behavior* of $\mathsf{Sign}(\mathsf{sk}_0,\cdot)$ or $\mathsf{Sign}(\mathsf{sk}_1,\cdot)$. Hence, this captures privacy for white-box watermarking signatures.

Remark 4.7 (On quantum-accessible oracle). The reason why we consider the quantum-accessible oracle $O_{\mathtt{qsign}}$ rather than $O_{\mathtt{sign}}$ in Definition 4.3 is that we need the quantum-accessible oracle to prove the impossibility of universal copy-protection for signatures in Section 6.

4.2 White-Box Watermarking Signature

Although pre-embedded white-box watermarking signatures are sufficient for many applications, we might want to embed a mark after we generate a key pair. We introduce the syntax and security definitions for (non-pre-embedded) white-box watermarking signatures in this subsection.

Definition 4.8 (White-Box Watermarking Signature (Syntax)). *A watermarking signature* WMSIG *for the signature message space* $\mathcal{MS}$ *and watermarking mark space* $\mathcal{MK}$ *is a tuple of five algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathsf{Mark}, \mathcal{Extract})$.

$\mathsf{KeyGen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a verification key* vk *and a signing key* sk.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{m}) \to \sigma$**:** *The signing algorithm takes as inpuft a signing key* sk *and a message* m *and outputs a signature* σ.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \sigma) \to 0/1$**:** *The verification algorithm takes as input a verification key* vk, *a message* m, *and a signature* σ *and outputs* 0 *or* 1.

$\mathsf{Mark}(\mathsf{sk}, \mu) \to \widetilde{\mathsf{C}}$**:** *The mark algorithm takes as input a signing key* sk *and a mark* μ, *and outputs a marked signing circuit* $\widetilde{\mathsf{C}}$.

$\mathcal{Extract}(\mathsf{vk}, \widetilde{\mathcal{C}}', \epsilon, (\mathsf{m}^*, \sigma^*)) \to \mu'$**:** *The extraction algorithm takes as input a verification key* vk, *a circuit* $\widetilde{\mathcal{C}}'$, *a parameter* ϵ, *and a message-signature pair* (m^*, σ^*), *and outputs* $\mu' \in \mathcal{MK} \cup \{\mathsf{unmarked}\}$.

Verification Correctness: *For any message* $\mathsf{m} \in \mathcal{MS}$, *we have* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \mathsf{Sign}(\mathsf{sk}, \mathsf{m})) = 1$, *where* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

For any message $\mathsf{m} \in \mathcal{MS}$ *and* $\mu \in \mathcal{MK}$, *we have* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \widetilde{\mathsf{C}}(\mathsf{m})) = 1$, *where* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $\widetilde{\mathsf{C}} \leftarrow \mathsf{Mark}(\mathsf{sk}, \mu)$.

Remark 4.9 (On private marking). White-box watermarking signatures in Definition 4.8 are public marking since anyone can embed a mark. Private marking (requiring a secret mark key for Mark) is sometimes preferred than public marking in some settings since we might want to prevent adversaries from forging a watermarked signing key. As observed by Goyal et al. [GKM+19] and Kitagawa and Nishimaki [KN24], we can generically convert watermarking signatures with public marking into ones with private marking by using standard signatures.

Remark 4.10 (On inputs for $\mathcal{Extract}$). Definition 4.8 is a natural quantum variant of classical watermarking signatures except that the extraction algorithm takes as input a message-signature pair (m^*, σ^*) in our syntax. Such a pair is not used in previous works on watermarking signatures [GKM+19]. We justify using a message-signature pair in the extraction algorithm as follows.

We need to obtain many pairs of input and output to extract an embedded message from a marked function in almost all known (classical) watermarking constructions [CHN+18, BLW17, KW21, QWZ18, KW19, YAL+19, GKM+19, Nis20, BBL24]. However, obtaining such pairs from an adversarially generated quantum circuit is hard since it might collapse when we run the circuit as Kitagawa and Nishimaki argued [KN24, Section 3.1]. Kitagawa and Nishimaki introduced a public tag related to an original PRF key in the syntax of their watermarking PRFs against quantum adversaries to overcome the issue [KN24]. The pair (m^*, σ^*) plays a similar role to the public tags in watermarking PRFs against quantum adversaries. The pair is supposed to be an input-output pair of $\widetilde{\mathcal{C}}'$, that is, $\sigma^* = \widetilde{\mathcal{C}}'(\mathsf{m}^*)$. In the watermarking signature setting, it is unrealistic that we try to extract an embedded mark from a possibly pirate signing program without seeing any message-signature pair because we judge a program is suspicious when we see at least one suspicious message-signature pair. If we do not see any message-signature pair, we do not have motivation to extract an embedded mark.

Definition 4.11 (Strong Correctness of Marked Keys). *We define the game* $\mathsf{Expt}^{\mathsf{scorrect}}_{\mathcal{A},\mathsf{WMSIG}}(1^\lambda)$ *as follows.*

1. *The challenger generates* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and sends* vk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *sends* $\mu \in \mathcal{MK}$ *to the challenger. The challenger generates* $\widetilde{\mathsf{C}} \leftarrow \mathsf{Mark}(\mathsf{sk}, \mu)$. $\mathcal{A}$ *can get access to the following oracles.*

$O_{\mathtt{sign}}(\mathsf{m})$: *On input* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$.

$O_{\mathtt{msign}}(\mathsf{m})$: *On input* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \widetilde{C}(\mathsf{m})$.

3. $\mathcal{A}$ *outputs* $\mathsf{m}^* \in \mathcal{MS}$. *The challenger outputs* 1 *if* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \widetilde{C}(\mathsf{m}^*)) = 0$ *and otherwise outputs* 0.

We say that WMSIG *satisfies strong correctness of marked keys if for every QPT* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{scorrect}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda) = \Pr\left[\mathsf{Expt}^{\mathsf{scorrect}}_{\mathcal{A},\mathsf{WMSIG}}(1^\lambda) = 1\right] \leq \mathsf{negl}(\lambda).$$

Definition 4.12 (Unforgeability). *We define the game* $\mathsf{Exp}^{\mathsf{euf\text{-}cma}}_{\mathcal{A},\mathsf{WMSIG}}(\lambda)$ *as follows.*

1. *The challenger generates* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and sends* vk *to* $\mathcal{A}$. $\mathcal{A}$ *can access the following oracles.*

   $O_{\mathtt{sign}}(\mathsf{m})$: *On input* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$. *Let* $\mathcal{Q}_s$ *be the set of the inputs received from* $\mathcal{A}$.

   $O_{\mathtt{msign}}(\mathsf{m}, \mu)$: *On input* $(\mathsf{m}, \mu) \in \mathcal{MS} \times \mathcal{MK}$, *it generates* $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{sk}, \mu)$ *and returns* $\sigma \leftarrow \widetilde{C}(\mathsf{m})$. *Let* $\mathcal{Q}_m$ *be the set of the inputs (only the message part* m*) received from* $\mathcal{A}$.

2. $\mathcal{A}$ *outputs* (m^*, σ^*). *If* $\mathsf{m}^* \notin \mathcal{Q}_s \cup \mathcal{Q}_m$, *the challenger outputs* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \sigma^*)$. *Otherwise* 0.

We say that WMSIG *satisfies unforgeability if for every QPT* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{Exp}^{\mathsf{euf\text{-}cma}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda).$$

Remark 4.13. The unforgeability definition is stronger than the unforgeability definition by Goyal et al. [GKM+19]. We consider $O_{\mathtt{sign}}$ and $O_{\mathtt{msign}}$ while Goyal et al. consider only $O_{\mathtt{sign}}$. Component of signatures generated by a marked signing key could be different from that of normal signatures (see the construction in Section 7). Hence, it is natural to allow adversaries to access $O_{\mathtt{msign}}$.

Note that we do not have the setup phase for generating mark and extraction keys (i.e., no authority) unlike the definition by Goyal et al. Hence, we do not need to consider unforgeability against malicious watermarking authority.

Definition 4.14 (Unremovability). *Let* $\epsilon \geq 0$. *We define the game* $\mathsf{Expt}^{\mathsf{urmv}}_{\mathcal{A},\mathsf{WMSIG}}(1^\lambda, \epsilon)$ *as follows.*

1. *The challenger generates* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and gives* vk *to the adversary* $\mathcal{A}$.

2. $\mathcal{A}$ *gets access to the following oracles.*

   $O_{\mathtt{sign}}$: *Given* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$. $\mathcal{A}$ *can send polynomially many queries to* $O_{\mathtt{sign}}$.

   $O_{\mathtt{mark}}$: *Given* $\mu \in \mathcal{MK}$, *it returns* $\widetilde{C}' \leftarrow \mathsf{Mark}(\mathsf{sk}, \mu)$. $\mathcal{A}$ *can send only* one *query to* $O_{\mathtt{mark}}$.

3. $\mathcal{A}$ *outputs a "potentially obfuscated" quantum circuit* $\widetilde{C} = (q, \boldsymbol{U})$, *where* $\widetilde{C}$ *is a quantum program with classical inputs and outputs and* $\boldsymbol{U}$ *is a compact classical description of* $\{\boldsymbol{U}_{\mathsf{m}}\}_{\mathsf{m} \in \mathcal{MS}}$. $\mathcal{A}$ *also outputs a pair* (m^*, σ^*), *which is an input-output pair of* $\widetilde{C}$ *such that* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \sigma^*) = 1$. *If* $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \sigma^*) = 0$, *the game aborts.*

Let $\boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}$ *be the unitary that maps* $|a\rangle |b\rangle$ *to* $|a\rangle |b \oplus \mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, a)\rangle$. *We also let* $\mathcal{P} = (\boldsymbol{P}_{\mathsf{m}}, \boldsymbol{Q}_{\mathsf{m}})_{\mathsf{m}}$ *be a collection of binary outcome projective measurements, where*

$$\boldsymbol{P}_{\mathsf{m}} = \boldsymbol{U}_{\mathsf{m}}^\dagger \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}^\dagger (\boldsymbol{I} \otimes |1\rangle \langle 1|) \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}} \boldsymbol{U}_{\mathsf{m}} \quad \text{and} \quad \boldsymbol{Q}_{\mathsf{m}} = \boldsymbol{I} - \boldsymbol{P}_{\mathsf{m}}.$$

Moreover, we let $U_{\mathcal{MS}}$ *be the uniform distribution over* $\mathcal{MS}$. *We consider the following events.*

Live: *When applying the measurement* $\mathcal{TI}_\epsilon(\mathcal{P}_{U_{\mathcal{MS}}})$ *to* q *(and ancilla), we obtain the outcome* 1, *where* $\mathcal{P}_{U_{\mathcal{MS}}}$ *is a mixture of* $\mathcal{P}$ *with respect to* $U_{\mathcal{MS}}$.

GoodExt: *When Computing* $\mu' \leftarrow \mathcal{Extract}(\mathsf{vk}, \widetilde{C}, \epsilon, (\mathsf{m}^*, \sigma^*))$, *it holds that* $\mu' \neq \mathsf{unmarked}$.

BadExt: *When Computing* $\mu' \leftarrow \mathcal{Extract}(\mathsf{vk}, \widetilde{C}, \epsilon, (\mathsf{m}^*, \sigma^*))$, *it holds that* $\mu' \notin \{\mu\} \cup \{\mathsf{unmarked}\}$.

We say that WMSIG *satisfies unremovability if for every* $\epsilon > 0$ *and QPT* $\mathcal{A}$, *we have*

$$\Pr[\mathsf{BadExt}] \leq \mathsf{negl}(\lambda) \quad \text{and} \quad \Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda).$$

Remark 4.15. We can consider the setting where $\mathcal{A}$ can send polynomially many queries to $O_{\mathtt{mark}}$ (collusion-resistant setting) unlike Definition 4.14, but it is out of scope of this work.

Definition 4.16 (Privacy). *We define the game* $\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{A},\mathsf{WMSIG}}(1^\lambda, \mathsf{coin})$ *as follows.*

1. $\mathcal{A}$ *sends* $(\mathsf{vk}, \mathsf{sk})$ *and* $(\mu_0, \mu_1) \in \mathcal{MK}^2$ *to the challenger. The challenger generates* $\widetilde{C}_{\mathsf{coin}} \leftarrow \mathsf{Mark}(\mathsf{sk}, \mu_{\mathsf{coin}})$. $\mathcal{A}$ *can get access to the following oracles.*

   $O_{\mathtt{sign}}(\mathsf{m})$: *On input* $\mathsf{m} \in \mathcal{MS}$, *it returns* $\sigma \leftarrow \widetilde{C}_{\mathsf{coin}}(\mathsf{m})$.

2. $\mathcal{A}$ *outputs* coin'. *The challenger outputs* coin'.

We say that WMSIG *satisfies privacy if for every QPT* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{A},\mathsf{WMSIG}}(1^\lambda, 0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{A},\mathsf{WMSIG}}(1^\lambda, 1) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Remark 4.17. Here, we consider the strong setting where $\mathcal{A}$ can select a signature key pair $(\mathsf{vk}, \mathsf{sk})$. Hence, our definition guarantees that even the signature authority cannot break privacy. We do not need to give $O_{\mathtt{mark}}$ unlike the unremovability definition since $\mathcal{A}$ has sk and we consider public marking.

5 Pre-Embedded White-Box Watermarking Signature

In this section, we present our pre-embedded white-box watermarking signature scheme and prove its security.

5.1 Construction

We construct $\mathsf{PWMSIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathsf{Extract})$. The building blocks are as follows.

- After-the-fact leakage resilient quantum unobfuscatable point function $\mathsf{UOPF}.(\mathsf{Gen}, \mathcal{Extract})$ with the secret message space $\{0,1\}^n$, the input space $\{0,1\}^{\ell_{\mathsf{in}}}$, and the output space $\{0,1\}^{\ell_{\mathsf{out}}}$.
- FE scheme $\mathsf{FE}.(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KG}, \mathsf{Dec}, \mathsf{SimEnc})$ for the 1-ouf-of-2 OT functionality,

  $$F[\beta](i, x_0, x_1) = x_{\beta[i]}.$$

  We let $\ell := |\mathsf{fe.ct}|$ where fe.ct is a ciphertext of FE.
- PRG $g : \{0,1\}^{\ell_{\mathsf{in}}} \to \{0,1\}^{2\ell_{\mathsf{in}}}$.
- Statistically binding equivocal commitment $\mathsf{Com}.(\mathsf{Setup}, \mathsf{Commit}, \mathsf{EqSetup}, \mathsf{Open})$.
- NIZK $\mathsf{NIZK}.(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vrfy})$ for $(\mathsf{stmt}, w) \in \mathcal{R}$. The relation $\mathcal{R}$ is defined as follows. $((\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma), (\mathsf{fsk}_1, \mathsf{fsk}_2, r)) \in \mathcal{R}$ if and only if the followings are satisfied:

  $$\mathsf{Com}.\mathsf{Commit}(\mathsf{ck}, \mathsf{fsk}_1 \| \mathsf{fsk}_2; r) = \mathsf{com} \wedge g(\mathsf{FE}.\mathsf{Dec}(\mathsf{fsk}_1, \mathsf{m})) \neq \gamma \wedge g(\mathsf{FE}.\mathsf{Dec}(\mathsf{fsk}_2, \mathsf{m})) \neq \gamma.$$

- Quantum-accesible PRF $\mathsf{PRF} : \{0,1\}^\ell \to \mathcal{R}_{\mathsf{NIZK}}$, where $\mathcal{R}_{\mathsf{NIZK}}$ is the randomness space of NIZK.Prove.

The construction of PWMSIG is as follows.

$\mathsf{KeyGen}(1^\lambda, \mu)$**:**

- Generate $K \leftarrow \{0,1\}^\lambda$.
- Generate $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$.
- Generate $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu)$.
- Let $\beta = \beta_1 \| \beta_2$ and compute $\gamma \leftarrow g(\alpha)$.
- Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for $d \in [2]$.
- Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$ for $d \in [2]$.
- Generate $\mathsf{ck} \leftarrow \mathsf{Com.Setup}(1^\lambda)$ and $r \leftarrow \mathcal{R}_{\mathsf{Com}}$, and generate $\mathsf{com} \leftarrow \mathsf{Com.Commit}(\mathsf{ck}, \mathsf{fsk}_1 \| \mathsf{fsk}_2; r)$, where $\mathcal{R}_{\mathsf{Com}}$ is the ranomndess space of Com.Commit.
- Output $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\mathsf{sk} := (\mathsf{vk}, \mathsf{fsk}_1, \mathsf{fsk}_2, r, K)$.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{m} \in \{0,1\}^\ell)$**:**

- Parse $\mathsf{sk} = (\mathsf{vk}, \mathsf{fsk}_1, \mathsf{fsk}_2, r, K)$ and $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$.
- If $\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m}) = \alpha$ for some $d \in [2]$, output $\perp$. Otherwise, go to the next step.
- Generate $r_{\mathsf{prv}} \leftarrow \mathsf{PRF}_K(\mathsf{m})$.
- Compute $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and $w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.
- Output $\sigma := \pi$.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \sigma)$**:**

- Parse $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\sigma = \pi$.
- Output the result of $\mathsf{NIZK.Vrfy}(\mathsf{crs}, \mathsf{stmt}, \pi)$, where $\mathsf{stmt} = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.

$\mathcal{Extract}(\mathsf{vk}, \mathcal{C}, \epsilon)$**:**

- Parse $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\mathcal{C} = (q, \boldsymbol{U})$.
- Let $\epsilon' = \epsilon/7$, $\delta' = 2^{-\lambda}$, and $t = \epsilon - \epsilon'$.
- Define $\mathcal{P}$ and $U_{\mathcal{MS}}$ in the same way as Definition 4.4.
- Compute $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},U_{\mathcal{MS}},t}\, q$ and output unmarked if the outcome is 0. Otherwise, letting the post state be q_1^0, go to the next step.
- Construct $\boldsymbol{V}$ that is a compact description of $\{\boldsymbol{V}_x\}_x$, where $\boldsymbol{V}_x$ is a unitary that performs the following computations coherently when applied to a quantum state q.
  1. Set $q = q_1^0$.
  2. Compute $(\beta_1'[i], q_1^i) \leftarrow \mathcal{SearchOutput}(\mathsf{vk}, \boldsymbol{U}, q_1^{i-1}, x, 1, i, \epsilon)$ for every $i \in [\lambda]$.
  3. Compute $(\beta_2'[i], q_2^i) \leftarrow \mathcal{SearchOutput}(\mathsf{vk}, \boldsymbol{U}, q_2^{i-1}, x, 2, i, \epsilon)$ for every $i \in [\lambda]$, where $q_2^0 = q_1^\lambda$.
  4. Output $\beta_1'[1] \| \cdots \| \beta_1'[\lambda] \| \beta_2'[1] \| \cdots \beta_2'[\lambda]$.
- Construct a quantum program with classical input and output $\mathcal{P}[\mathcal{C}] = (q_1^0, \boldsymbol{V})$.
- Output $\mu' \leftarrow \mathsf{UOPF}.\mathcal{Extract}(\mathcal{P}[\mathcal{C}], \mathsf{aux})$.

$\mathcal{S}earch\mathcal{O}utput(\mathsf{vk}, \boldsymbol{U}, q, x, d, i, \epsilon)$

Input: $\mathsf{vk}, \boldsymbol{U}, q, x, \epsilon$.

1. Parse $\mathsf{vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$.
2. Let $\epsilon' = \epsilon/7$, $\delta' = 2^{-\lambda}$, and $t = \epsilon - 5\epsilon'$.
3. Define $\mathcal{P}$ in the same way as Definition 4.4.
4. Define D_d^i be the following distribution.
   - Generate $u_d^i \leftarrow \{0,1\}^\lambda$.
   - Output $\mathsf{fe.ct}_d^i \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}_d, (i, x, u_d^i))$.
5. Compute $\beta_d'[i] \leftarrow \mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,t}\, q$.
6. Uncompute the previous step and output $\beta_d'[i]$ and the resulting state.

Figure 1: The description of $\mathcal{S}earch\mathcal{O}utput$

Verification Correctness. Fix $\mathsf{m} \in \{0,1\}^\ell$ and $\mu \in \{0,1\}^n$. The probability that the condition "$g(\mathsf{FE.Dec}(\mathsf{fsk}_1, \mathsf{m})) = \gamma$ or $g(\mathsf{FE.Dec}(\mathsf{fsk}_2, \mathsf{m})) = \gamma$" is satisfied is negligible over the choice of α, fsk_1, and fsk_2 from the security of PRG g, where $\gamma = g(\alpha)$. Then, from the completeness of NIZK and the security of PRF, the verification correctness of UOSIG follows.

We need to prove that PWMSIG satisfies the four security requirements. We have the following theorems.

Theorem 5.1. *Assume g is a PRG, Com is a statistically binding equivocal commitment, UOPF is an unobfuscatable point function satisfying 2-after-the-fact leakage resilient indistinguishability of points, and NIZK is a NIZK satisfying computational zero-knowledge. Then, PWMSIG satisfies unforgeability.*

Theorem 5.2. *Assume g is an injective PRG, Com is a statistically binding equivocal commitment, UOPF is an unobfuscatable point function satisfying 2-after-the-fact leakage resilient indistinguishability of points, and NIZK is a NIZK satisfying computational zero-knowledge. Then, PWMSIG satisfies strong correctness of marked keys.*

Theorem 5.3. *Assume g is a PRG, Com is a statistically binding equivocal commitment, UOPF is an unobfuscatable point function satisfying indistinguishability of messages and 2-after-the-fact leakage resilient indistinguishability of points, and NIZK is a strong statistical NIZK argument for adversaries with 2^ℓ queries. Then, PWMSIG satisfies privacy.*

Theorem 5.4. *Assume UOPF satisfies correctness, Com is a statistically binding equivocal commitment, NIZK is a NIZK satisfying adaptive exclusive soundness, and FE is an FE scheme satisfying 1-bounded simulation security and ciphertext uniformity. Then, PWMSIG satisfies unremovability.*

We prove these theorems in the subsequent sections (Sections 5.2 to 5.5). Thus, we obtain the following theorem.

Theorem 5.5. *If the LWE assumption holds and QFHE exists, PWMSIG is a pre-embedded white-box watermarking signature scheme against quantum adversaries.*

5.2 Proof of Unforegability

We prove Theorem 5.1. We use the following sequence of experiments.

Hyb_0**:** This is $\mathsf{Exp}^{\mathsf{euf\text{-}cma}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda)$.

1. Given 1^λ as the initial input, $\mathcal{A}$ sends μ to the challenger. The challenger generates vk and sk as follows.
   - Generate $K \leftarrow \{0,1\}^\lambda$.
   - Generate $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$.

- Generate $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu)$.
- Let $\beta = \beta_1 \| \beta_2$ and compute $\gamma \leftarrow g(\alpha)$.
- Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for $d \in [2]$.
- Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$ for $d \in [2]$.
- Generate $\mathsf{ck} \leftarrow \mathsf{Com.Setup}(1^\lambda)$ and $r \leftarrow \mathcal{R}_{\mathsf{Com}}$, and generate $\mathsf{com} \leftarrow \mathsf{Com.Commit}(\mathsf{ck}, \mathsf{fsk}_1 \| \mathsf{fsk}_2; r)$.
- Set $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\mathsf{sk} := (\mathsf{vk}, \mathsf{fsk}_1, \mathsf{fsk}_2, r)$.

The challenger sends vk to $\mathcal{A}$.

2. $\mathcal{A}$ can get access to the following $O_{\mathtt{sign}}$.

   $O_{\mathtt{sign}}(\mathsf{m})$**:** On input m, it behaves as follows.
   - If $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m})) = \gamma$ for some $d \in [2]$, output $\perp$. Otherwise, go to the next step.
   - Generate $r_{\mathsf{prv}} \leftarrow \mathsf{PRF}_K(\mathsf{m})$.
   - Compute $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and $w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.
   - Output $\sigma := \pi$.

3. $\mathcal{A}$ outputs (m^*, σ^*). If $\mathsf{m}^* \notin \mathcal{Q}$, the challenger outputs $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \sigma^*)$, where $\mathcal{Q}$ is the list of messages queried to $O_{\mathtt{sign}}$ by $\mathcal{A}$. Otherwise, the challenger outputs 0.

We have $\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda) = \Pr[\mathsf{Hyb}_0 = 1]$.

Hyb_1**:** This is the same as Hyb_0 except that the challenger generates $(\mathsf{ck}, \mathsf{com}, \mathsf{com.td}) \leftarrow \mathsf{Com.EqSetup}(1^\lambda)$ and $r \leftarrow \mathsf{Com.Open}(\mathsf{com.td}, \mathsf{fe.fsk}_1 \| \mathsf{fe.fsk}_2, \mathsf{com})$.

We have $|\Pr[\mathsf{Hyb}_0 = 1] - \Pr[\mathsf{Hyb}_1 = 1]| = \mathsf{negl}(\lambda)$ from the trapdoor equivocal property of Com.

Hyb_2**:** This is the same as Hyb_1 except that given m, $O_{\mathtt{sign}}$ uses a truly random coin r_{prv} instead of $r_{\mathsf{prv}} \leftarrow \mathsf{PRF}_K(\mathsf{m})$ to generate the answer.

We have $|\Pr[\mathsf{Hyb}_1 = 1] - \Pr[\mathsf{Hyb}_2 = 1]| = \mathsf{negl}(\lambda)$ from the security of PRF.

Hyb_3**:** This is the same as Hyb_2 except that the challenger generates $(\mathsf{crs}, \mathsf{nizk.td}) \leftarrow \mathsf{NIZK.Sim}_1(1^\lambda)$ and $O_{\mathtt{sign}}$ returns $\pi \leftarrow \mathsf{NIZK.Sim}_2(\mathsf{crs}, \mathsf{nizk.td}, x)$ for all query m such that $g(\mathsf{FE.Dec}(\mathsf{fsk}_1, \mathsf{m})) \neq \gamma$ and $g(\mathsf{FE.Dec}(\mathsf{fsk}_2, \mathsf{m})) \neq \gamma$, where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.

We have $|\Pr[\mathsf{Hyb}_2 = 1] - \Pr[\mathsf{Hyb}_3 = 1]| = \mathsf{negl}(\lambda)$ from the zero knowledge of NIZK.

Hyb_4**:** This is the same as Hyb_3 except that $O_{\mathtt{sign}}$ returns $\pi \leftarrow \mathsf{NIZK.Sim}_2(\mathsf{crs}, \mathsf{nizk.td}, x)$ for all query m, where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.

We define the following event $\mathtt{BQ}_k$.

$\mathtt{BQ}_k$**:** In Hyb_k, $\mathcal{A}$ queries m to $O_{\mathtt{sign}}$ such that $g(\mathsf{FE.Dec}(\mathsf{fsk}_1, \mathsf{m})) = \gamma$ or $g(\mathsf{FE.Dec}(\mathsf{fsk}_2, \mathsf{m})) = \gamma$.

We have $|\Pr[\mathsf{Hyb}_3 = 1] - \Pr[\mathsf{Hyb}_4 = 1]| = \Pr[\mathtt{BQ}_4]$ since Hyb_4 is the same as Hyb_3 if $\mathtt{BQ}_4$ does not happen.

Hyb_5**:** This is the same as Hyb_4 except that the challenger generates $\gamma \leftarrow g(R)$ for $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$.

We have $|\Pr[\mathsf{Hyb}_4 = 1] - \Pr[\mathsf{Hyb}_5 = 1]| = \mathsf{negl}(\lambda)$ from the indistinguishability of points of UOPF. We also prove that $|\Pr[\mathtt{BQ}_4] - \Pr[\mathtt{BQ}_5]| = \mathsf{negl}(\lambda)$ using the after-the-fact leakage resilient indistinguishability of points of UOPF. Using $\mathcal{A}$, we construct the following $\mathcal{B}$ that attacks the after-the-fact leakage resilient indistinguishability of points of UOPF.

1. Given input 1^λ, $\mathcal{B}$ invokes $\mathcal{A}$ with the initial input 1^λ and receives μ. $\mathcal{B}$ forwards μ to its challenger, receives (r, aux), and generates vk as follows.

- Generate $(\mathsf{crs}, \mathsf{nizk.td}) \leftarrow \mathsf{NIZK.Sim}_1(1^\lambda)$.
- Compute $\gamma \leftarrow g(r)$.
- Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for every $d \in [2]$.
- Generate $(\mathsf{ck}, \mathsf{com}, \mathsf{com.td}) \leftarrow \mathsf{Com.EqSetup}(1^\lambda)$.
- Set $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$.

$\mathcal{B}$ sends vk to $\mathcal{A}$.

2. $\mathcal{B}$ simulates $O_{\mathtt{sign}}$ for $\mathcal{A}$ as Hyb_4 and Hyb_5. (The behavior of $O_{\mathtt{sign}}$ is the same in these two experiments.) This can be done by using $\mathsf{nizk.td}$.
3. When When $\mathcal{A}$ outputs (m^*, σ^*), $\mathcal{B}$ does the following. $\mathcal{B}$ outputs leakage functions $(h[\mathsf{fe.msk}_d, \gamma, \mathsf{List}])_{d\in[2]}$, where $h[\mathsf{fe.msk}_d, \gamma, \mathsf{List}]$ is described in Figure 3 and List is the list of all queries to $O_{\mathtt{sign}}$ made by $\mathcal{A}$. $\mathcal{B}$ receives leakage information $(b_1, b_2) \in \{0,1\}^2$.
4. $\mathcal{B}$ outputs 1 if $b_d = 1$ for some $d \in [2]$.

Function $h[\mathsf{fe.msk}_d, \gamma, \mathsf{List}](\beta_d)$

Constants: $\mathsf{fe.msk}_d, \gamma, \mathsf{List}$.

Input: A string β_d.

1. Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$.
2. Output 1 if there exists $\mathsf{m} \in \mathsf{List}$ such that $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m})) = \gamma$. Otherwise, output 0.

Figure 2: The description of $h[\mathsf{fe.msk}_d, \gamma, \mathsf{List}]$

$\mathcal{B}$ perfectly simulates Hyb_4 (resp. Hyb_5) if it is given α (resp. $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$). Also, $\mathcal{B}$ outputs 1 if and only if the event $\mathtt{BQ}_4$ and $\mathtt{BQ}_5$ occur in the simulated experiments. Thus, from the after-the-fact leakage resilient indistinguishability of points of UOPF, we have $|\Pr[\mathtt{BQ}_4] - \Pr[\mathtt{BQ}_5]| = \mathsf{negl}(\lambda)$.

Hyb_6**:** This is the same as Hyb_5 except that the challenger generates $\gamma \leftarrow \{0,1\}^{2\ell_{\mathsf{in}}}$ instead of $\gamma \leftarrow g(R)$.

We have $|\Pr[\mathsf{Hyb}_5 = 1] - \Pr[\mathsf{Hyb}_6 = 1]| = \mathsf{negl}(\lambda)$ and $|\Pr[\mathtt{BQ}_5] - \Pr[\mathtt{BQ}_6]| = \mathsf{negl}(\lambda)$ from the security of PRG g.

In Hyb_6 where γ is a uniformly random string, there does not exist x such that $\gamma = g(x)$ except negligible probability. Then, we have $\Pr[\mathtt{BQ}_6] = \mathsf{negl}(\lambda)$. To bound $\Pr[\mathsf{Hyb}_6 = 1]$, we introduce one more hybrid experiment.

Hyb_7**:** This is the same as Hyb_6 except that the challenger generates $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$ and $O_{\mathtt{sign}}$ returns $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w)$ for all query m, where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and $w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.

We have $|\Pr[\mathsf{Hyb}_6 = 1] - \Pr[\mathsf{Hyb}_7 = 1]| = \mathsf{negl}(\lambda)$ from the zero knowledge of NIZK. Note that in Hyb_6 and Hyb_7, γ does not have a pre-image of g and thus $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ is a true statement for all m. Therefore, we do not care about whether a queried m forms a true statement or not, and can use the zero-knowledge of NIZK. Moreover, we have $\Pr[\mathsf{Hyb}_7 = 1] = \mathsf{negl}(\lambda)$ from the adaptive exclusive soundness of NIZK.

This completes the proof. □

5.3 Proof of Strong Correctness

We prove Theorem 5.2. This proof is almost the same as that of Theorem 5.1. We use the following sequence of experiments.

Hyb_0**:** This is $\mathsf{Expt}^{\mathsf{scorrect}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda)$.

1. Given 1^λ as the initial input, $\mathcal{A}$ sends μ to the challenger. The challenger generates vk and sk as follows.
   - Generate $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$.
   - Generate $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu)$.
   - Let $\beta = \beta_1 \| \beta_2$ and compute $\gamma \leftarrow g(\alpha)$.
   - Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for $d \in [2]$.
   - Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$ for $d \in [2]$.
   - Generate $\mathsf{ck} \leftarrow \mathsf{Com.Setup}(1^\lambda)$ and $r \leftarrow \mathcal{R}_{\mathsf{com}}$, and generate $\mathsf{com} \leftarrow \mathsf{Com.Commit}(\mathsf{ck}, \mathsf{fsk}_1 \| \mathsf{fsk}_2; r)$.
   - Set $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\mathsf{sk} := (\mathsf{vk}, \mathsf{fsk}_1, \mathsf{fsk}_2, r)$.

   The challenger sends vk to $\mathcal{A}$.
2. $\mathcal{A}$ can get access to the following $O_{\mathtt{sign}}$.

   $O_{\mathtt{sign}}(\mathsf{m})$**:** On input m, it behaves as follows.
   - If $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m})) = \gamma$ for some $d \in [2]$, output $\bot$. Otherwise, go to the next step.
   - Generate $r_{\mathsf{prv}} \leftarrow \mathsf{PRF}_K(\mathsf{m})$.
   - Compute $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and $w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.
   - Output $\sigma := \pi$.
3. $\mathcal{A}$ outputs $\mathsf{m}^* \in \mathcal{MS}$. The challenger outputs 1 if $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}^*)) = 1$ and otherwise outputs 0.

We have $\mathsf{Adv}^{\mathsf{scorrect}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda) = \Pr[\mathsf{Hyb}_0 = 1]$.

Hyb_1**:** This is the same as Hyb_0 except that the challenger generates $(\mathsf{ck}, \mathsf{com}, \mathsf{com.td}) \leftarrow \mathsf{Com.EqSetup}(1^\lambda)$ and $r \leftarrow \mathsf{Com.Open}(\mathsf{com.td}, \mathsf{fe.fsk}_1 \| \mathsf{fe.fsk}_2, \mathsf{com})$.

We have $|\Pr[\mathsf{Hyb}_0 = 1] - \Pr[\mathsf{Hyb}_1 = 1]| = \mathsf{negl}(\lambda)$ from the trapdoor equivocal property of Com.

Hyb_2**:** This is the same as Hyb_1 except that given m, $O_{\mathtt{sign}}$ uses a truly random coin r_{prv} instead of $r_{\mathsf{prv}} \leftarrow \mathsf{PRF}_K(\mathsf{m})$ to generate the answer.

We have $|\Pr[\mathsf{Hyb}_1 = 1] - \Pr[\mathsf{Hyb}_2 = 1]| = \mathsf{negl}(\lambda)$ from the security of PRF.

Hyb_3**:** This is the same as Hyb_1 except that the challenger generates $(\mathsf{crs}, \mathsf{nizk.td}) \leftarrow \mathsf{NIZK.Sim}_1(1^\lambda)$ and $O_{\mathtt{sign}}$ returns $\pi \leftarrow \mathsf{NIZK.Sim}_2(\mathsf{crs}, \mathsf{nizk.td}, x)$ for all query m such that $g(\mathsf{FE.Dec}(\mathsf{fsk}_1, \mathsf{m})) \neq \gamma$ and $g(\mathsf{FE.Dec}(\mathsf{fsk}_2, \mathsf{m})) \neq \gamma$, where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.

We have $|\Pr[\mathsf{Hyb}_2 = 1] - \Pr[\mathsf{Hyb}_3 = 1]| = \mathsf{negl}(\lambda)$ from the zero knowledge of NIZK.

Hyb_4**:** This is the same as Hyb_3 except that $O_{\mathtt{sign}}$ returns $\pi \leftarrow \mathsf{NIZK.Sim}_2(\mathsf{nizk.td}, x)$ for all query m, where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.

We define the following event $\mathtt{BQ}_k$.

$\mathtt{BQ}_k$**:** In Hyb_k, $\mathcal{A}$ queries m to $O_{\mathtt{sign}}$ such that $g(\mathsf{FE.Dec}(\mathsf{fsk}_1, \mathsf{m})) = \gamma$ or $g(\mathsf{FE.Dec}(\mathsf{fsk}_2, \mathsf{m})) = \gamma$.

We have $|\Pr[\mathsf{Hyb}_3 = 1] - \Pr[\mathsf{Hyb}_4 = 1]| = \Pr[\mathtt{BQ}_4]$.

Hyb_5**:** This is the same as Hyb_4 except that the challenger generates $\gamma \leftarrow g(R)$ for $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$.

We have $|\Pr[\mathsf{Hyb}_4 = 1] - \Pr[\mathsf{Hyb}_5 = 1]| = \mathsf{negl}(\lambda)$ and $|\Pr[\mathtt{BQ}_4] - \Pr[\mathtt{BQ}_5]| = \mathsf{negl}(\lambda)$ from the indistinguishability of points and the after-the-fact leakage resilient indistinguishability of points of UOPF, respectively.

Hyb_6**:** This is the same as Hyb_5 except that the challenger generates $\gamma \leftarrow \{0,1\}^{2\ell_{\mathsf{in}}}$ instead of $\gamma \leftarrow g(R)$.

We have $|\Pr[\mathsf{Hyb}_5 = 1] - \Pr[\mathsf{Hyb}_6 = 1]| = \mathsf{negl}(\lambda)$ and $|\Pr[\mathtt{BQ}_5] - \Pr[\mathtt{BQ}_6]| = \mathsf{negl}(\lambda)$ from the security of PRG g.

In Hyb_6 where γ is a uniformly random string, there does not exist x such that $\gamma = g(x)$ except negligible probability. Then, we have $\Pr[\mathtt{BQ}_6] = \mathsf{negl}(\lambda)$. Moreover, we have $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}^*)) = 1$ for any $\mathsf{m}^* \in \mathcal{MS}$. This is because $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}^*, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}^*)) = 0$ holds only when $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m}^*)) = \gamma$ holds for some $d \in \{0,1\}$, but now there does not exist x such that $\gamma = g(x)$. This means we have $\Pr[\mathsf{Hyb}_6 = 1] = \mathsf{negl}(\lambda)$. This completes the proof. $\square$

5.4 Proof of Privacy

We prove Theorem 5.3. We use the following sequence of experiments.

Hyb_0: This is $\mathsf{Expt}^{\mathsf{priv}}_{\mathsf{PWMSIG},\mathcal{A}}(1^\lambda, \mathsf{coin})$ where $\mathsf{coin} \leftarrow \{0,1\}$ and the final output of the experiment is 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

1. Given 1^λ as the initial input, $\mathcal{A}$ sends μ_0, μ_1 to the challenger. The challenger generates vk and sk as follows.
   - Generate $K \leftarrow \{0,1\}^\lambda$.
   - Generate $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$.
   - Generate $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu_{\mathsf{coin}})$.
   - Let $\beta = \beta_1 \| \beta_2$ and compute $\gamma \leftarrow g(\alpha)$.
   - Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for $d \in [2]$.
   - Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$ for $d \in [2]$.
   - Generate $\mathsf{ck} \leftarrow \mathsf{Com.Setup}(1^\lambda)$ and $r \leftarrow \mathcal{R}_{\mathsf{Com}}$, and generate $\mathsf{com} \leftarrow \mathsf{Com.Commit}(\mathsf{ck}, \mathsf{fsk}_1 \| \mathsf{fsk}_2; r)$.
   - Set $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\mathsf{sk} := (\mathsf{vk}, \mathsf{fsk}_1, \mathsf{fsk}_2, r, K)$.

   The challenger sends vk to $\mathcal{A}$.
2. $\mathcal{A}$ can get access to the following $O_{\mathtt{qsign}}$.

   $O_{\mathtt{qsign}}$: On input two registers R_1 and R_2, it applies the signing unitary that maps $|a\rangle_{\mathsf{R}_1} |b\rangle_{\mathsf{R}_2}$ to $|a\rangle_{\mathsf{R}_1} |b \oplus \mathsf{Sign}(\mathsf{sk}, a)\rangle_{\mathsf{R}_2}$ and returns the resisters, where $\mathsf{Sign}(\mathsf{sk}, \mathsf{m})$ behaves as follows
   - If $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m})) = \gamma$ for some $d \in [2]$, output $\perp$. Otherwise, go to the next step.
   - Generate $r_{\mathsf{prv}} \leftarrow \mathsf{PRF}_K(\mathsf{m})$.
   - Compute $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and $w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.
   - Output $\sigma := \pi$.
3. $\mathcal{A}$ outputs coin'. The challenger outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

We have $\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{PWMSIG},\mathcal{A}}(\lambda) = 2\left|\Pr[\mathsf{Hyb}_0 = 1] - \frac{1}{2}\right|$.

Hyb_1: This is the same as Hyb_0 except that the challenger generates $(\mathsf{ck}, \mathsf{com}, \mathsf{com.td}) \leftarrow \mathsf{Com.EqSetup}(1^\lambda)$ and $r \leftarrow \mathsf{Com.Open}(\mathsf{com.td}, \mathsf{fe.fsk}_1 \| \mathsf{fe.fsk}_2, \mathsf{com})$.

We have $|\Pr[\mathsf{Hyb}_0 = 1] - \Pr[\mathsf{Hyb}_1 = 1]| = \mathsf{negl}(\lambda)$ from the trapdoor equivocal property of Com.

Hyb_2: This is the same as Hyb_1 except that $O_{\mathtt{qsign}}$ behaves as follows, where R is a random function.

$O_{\mathtt{qsign}}$: On input two registers R_1 and R_2, it applies the signing unitary that maps $|a\rangle_{\mathsf{R}_1} |b\rangle_{\mathsf{R}_2}$ to $|a\rangle_{\mathsf{R}_1} \left|b \oplus \mathsf{Sign}'(\mathsf{sk}, a)\right\rangle_{\mathsf{R}_2}$ and returns the resisters, where $\mathsf{Sign}'(\mathsf{sk}, \mathsf{m})$ behaves as follows

- If $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m})) = \gamma$ for some $d \in [2]$, output $\perp$. Otherwise, go to the next step.
- Generate $r_{\mathsf{prv}} \leftarrow R(\mathsf{m})$.

- Compute $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, x, w; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and $w = (\mathsf{fsk}_1, \mathsf{fsk}_2, r)$.
- Output $\sigma := \pi$.

We have $|\Pr[\mathsf{Hyb}_1 = 1] - \Pr[\mathsf{Hyb}_2 = 1]| = \mathsf{negl}(\lambda)$ from the security of PRF.

Hyb_3**:** This is the same as Hyb_2 except that the challenger generates $(\mathsf{crs}, \mathsf{nizk.td}) \leftarrow \mathsf{NIZK.Sim}_1(1^\lambda)$ and $O_{\mathtt{qsign}}$ behaves as follows, where R is a random function.

$O_{\mathtt{qsign}}$**:** On input two registers R_1 and R_2, it applies the signing unitary that maps $|a\rangle_{\mathsf{R}_1} |b\rangle_{\mathsf{R}_2}$ to $|a\rangle_{\mathsf{R}_1} |b \oplus \mathsf{Sign}''(\mathsf{sk}, a)\rangle_{\mathsf{R}_2}$ and returns the resisters, where $\mathsf{Sign}''(\mathsf{sk}, \mathsf{m})$ behaves as follows

- If $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m})) = \gamma$ for some $d \in [2]$, output $\perp$. Otherwise, go to the next step.
- Generate $r_{\mathsf{prv}} \leftarrow R(\mathsf{m})$.
- Compute $\pi \leftarrow \mathsf{NIZK.Sim}_2(\mathsf{crs}, \mathsf{nizk.td}, x; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.
- Output $\sigma := \pi$.

An unbounded adversary attacking statistical zero knowledge can simulate Sign' and Sign'' by querying the statement $(\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$ and the corresponding witness $(\mathsf{fsk}_1, \mathsf{fsk}_2, r)$ for every $\mathsf{m} \in \{0,1\}^\ell$ to its oracle, depending on which one of the real oracle and the simulated oracle the adversary gets access to. We have $|\Pr[\mathsf{Hyb}_2 = 1] - \Pr[\mathsf{Hyb}_3 = 1]| = \mathsf{negl}(\lambda)$ from NIZK's strong statistical zero-knowledge for adversaries with 2^ℓ queries.

Hyb_4**:** This is the same as Hyb_3 except that $O_{\mathtt{qsign}}$ behaves as follows, where R is a random function.

$O_{\mathtt{qsign}}$**:** On input two registers R_1 and R_2, it applies the signing unitary that maps $|a\rangle_{\mathsf{R}_1} |b\rangle_{\mathsf{R}_2}$ to $|a\rangle_{\mathsf{R}_1} |b \oplus \mathsf{Sign}'''(\mathsf{sk}, a)\rangle_{\mathsf{R}_2}$ and returns the resisters, where $\mathsf{Sign}'''(\mathsf{sk}, \mathsf{m})$ behaves as follows

- Generate $r_{\mathsf{prv}} \leftarrow R(\mathsf{m})$.
- Compute $\pi \leftarrow \mathsf{NIZK.Sim}_2(\mathsf{crs}, \mathsf{nizk.td}, x; r_{\mathsf{prv}})$ where $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m}, \gamma)$.
- Output $\sigma := \pi$.

We assume that the total number of queries to $O_{\mathtt{qsign}}$ made by $\mathcal{A}$ is q. We define p_k as follows.

p_k**:** We randomly pick $i \leftarrow [q]$. Suppose we simulate Hyb_k for $\mathcal{A}$ until just before $\mathcal{A}$ makes the i-th query to $O_{\mathtt{qsign}}$, and we measure the i-th query to $O_{\mathtt{qsign}}$ and obtain (a, b). p_k is the probability that $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, a)) = \gamma$ is satisfied for some $d \in [2]$ with the measured a.

From Lemma 2.13, we have $|\Pr[\mathsf{Hyb}_3 = 1] - \Pr[\mathsf{Hyb}_4 = 1]| = 2q\sqrt{p_4}$, where q is the number of queries to $O_{\mathtt{qsign}}$ made by $\mathcal{A}$.

Hyb_5**:** This is the same as Hyb_4 except that the challenger generates $\gamma \leftarrow g(R)$ for $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$.

We have $|\Pr[\mathsf{Hyb}_4 = 1] - \Pr[\mathsf{Hyb}_5 = 1]| = \mathsf{negl}(\lambda)$ from the indistinguishability of points of UOPF. We also prove that $|p_4 - p_5| = \mathsf{negl}(\lambda)$ using the after-the-fact leakage resilient indistinguishability of points of UOPF. Using $\mathcal{A}$, we construct the following $\mathcal{B}$ that attacks the after-the-fact leakage resilient indistinguishability of points of UOPF.

1. Given input 1^λ, $\mathcal{B}$ invokes $\mathcal{A}$ with the initial input 1^λ and obtains (μ_0, μ_1). $\mathcal{B}$ picks $i \leftarrow [q]$, forwards μ_{coin} to its challenger, receivers (r, aux), and generates vk as follows.
   - Generate $(\mathsf{crs}, \mathsf{nizk.td}) \leftarrow \mathsf{NIZK.Sim}_1(1^\lambda)$.
   - Compute $\gamma \leftarrow g(r)$.
   - Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for every $d \in [2]$.
   - Generate $(\mathsf{ck}, \mathsf{com}, \mathsf{com.td}) \leftarrow \mathsf{Com.EqSetup}(1^\lambda)$.

- Set $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$.

$\mathcal{B}$ sends vk to $\mathcal{A}$.

2. $\mathcal{B}$ simulates $O_{\mathtt{qsign}}$ for $\mathcal{A}$ as Hyb_4 just before $\mathcal{A}$ makes the i-th query. This can be done by using nizk.td.
3. When $\mathcal{A}$ outputs the i-th query to $O_{\mathtt{qsign}}$, $\mathcal{B}$ measures it, obtain the measurement result m^*, and does the following. $\mathcal{B}$ outputs leakage functions $(h[\mathsf{fe.msk}_d, \gamma, \mathsf{m}^*])_{d \in [2]}$, where $h[\mathsf{fe.msk}_d, \gamma, \mathsf{m}^*]$ is described in Figure 3. $\mathcal{B}$ receives leakage information $(b_1, b_2) \in \{0,1\}^2$.
4. $\mathcal{B}$ outputs 1 if $b_d = 1$ for some $d \in [2]$.

Function $h[\mathsf{fe.msk}_d, \gamma, \mathsf{m}^*](\beta_d)$

Constants: $\mathsf{fe.msk}_d, \gamma, \mathsf{m}^*$.

Input: A string β_d.

1. Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$.
2. Output 1 if $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m}^*)) = \gamma$. Otherwise, output 0.

Figure 3: The description of $h[\mathsf{fe.msk}_d, \gamma, \mathsf{m}^*]$

$\mathcal{B}$ perfectly simulates Hyb_4 (resp. Hyb_5) just before the i-th query to $O_{\mathtt{qsign}}$ for randomly chosen i if it is given α (resp. $R \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$). Also, $\mathcal{B}$ outputs 1 if and only if the measurement result m^* of the i-th query to $O_{\mathtt{qsign}}$ satisfies $g(\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{m}^*)) = \gamma$ for some $d \in [2]$ in the simulated experiments. Thus, from the definition of p_4 and p_5 and the after-the-fact leakage resilient indistinguishability of points of UOPF, we have $|p_4 - p_5| = \mathsf{negl}(\lambda)$.

From the indistinguishability of messages of UOPF, we have $\left|\Pr[\mathsf{Hyb}_5 = 1] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$. To bound p_5, we introduce one additional experiment.

Hyb_6**:** This is the same as Hyb_6 except that the challenger generates $\gamma \leftarrow \{0,1\}^{2\ell_{\mathsf{in}}}$ instead of $\gamma \leftarrow g(R)$.

We have $|p_5 - p_6| = \mathsf{negl}(\lambda)$ from the security of PRG g. Moreover, we have $p_6 = \mathsf{negl}(\lambda)$ since there does not exist x such that $\gamma = g(x)$ except negligible probability in Hyb_6 where γ is a uniformly random string.

This completes the proof. □

5.5 Proof of Unremovability

We prove Theorem 5.4. Let $\mathcal{A}$ be a QPT adversary attacking the unremovability of PWMSIG. The description of $\mathsf{Expt}^{\mathsf{urmv}}_{\mathcal{A},\mathsf{PWMSIG}}(\lambda, \epsilon)$ is as follows.

1. Given 1^λ as the initial input, $\mathcal{A}$ sends $\mu \in \{0,1\}^n$ to the challenger. The challenger sends $(\mathsf{vk}, \mathsf{sk})$ generated as follows.
   - Generate $K \leftarrow \{0,1\}^\lambda$.
   - Generate $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$.
   - Generate $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF.Gen}(1^\lambda, \mu)$.
   - Let $\beta = \beta_1 \| \beta_2$ and compute $\gamma \leftarrow g(\alpha)$.
   - Generate $(\mathsf{fe.pk}_d, \mathsf{fe.msk}_d) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ for $d \in [2]$.
   - Generate $\mathsf{fsk}_d \leftarrow \mathsf{FE.KG}(\mathsf{fe.msk}_d, \beta_d)$ for $d \in [2]$.
   - Generate $\mathsf{ck} \leftarrow \mathsf{Com.Setup}(1^\lambda)$ and $r \leftarrow \mathcal{R}_{\mathsf{com}}$, and generate $\mathsf{com} \leftarrow \mathsf{Com.Commit}(\mathsf{ck}, \mathsf{fsk}_1 \| \mathsf{fsk}_2; r)$.
   - Set $\mathsf{vk} := (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$ and $\mathsf{sk} := (\mathsf{vk}, \mathsf{fsk}_1, \mathsf{fsk}_2, r, K)$.
2. The adversary outputs $\widetilde{C} = (q, \boldsymbol{U})$.

We define the three events Live, GoodExt, and BadExt in the same way as Definition 4.4.

The proof of $\Pr[\mathsf{BadExt}] \leq \mathsf{negl}(\lambda)$**.** $\Pr[\mathsf{BadExt}] \leq \mathsf{negl}(\lambda)$ directly follows from the description of $\mathcal{E}xtract$ and the correctness of UOPF. ■

The proof of $\Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$**.** We define the event $\mathsf{NotAbort}$ as the event that when running $\mathcal{E}xtract(\mathsf{vk}, \widetilde{\mathcal{C}}, \epsilon)$, $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},U_{\mathcal{MS}},\epsilon-\epsilon'}q$ computed in the 4-th line of $\mathcal{E}xtract$ results in the outcome 1. From Lemma 2.11, we have $\Pr[\mathsf{NotAbort}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$. We prove that if the event $\mathsf{NotAbort}$ occurs, $\mathcal{P}[\widetilde{\mathcal{C}}] = (q_1^0, \boldsymbol{V})$ constructed when running $\mathcal{E}xtract(\mathsf{vk}, \widetilde{\mathcal{C}}, \epsilon)$ is a quantum program with classical input and output that maps α to $\beta = \beta_1 \| \beta_2$ with overwhelming probability, where q_1^0 is the state after applying $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},U_{\mathcal{MS}},\epsilon-\epsilon'}$ to q.

We show the following lemma.

Lemma 5.6. *Suppose* $\mathsf{NotAbort}$ *occurs and we apply the following computations to* q_1^0*.*

1. *Compute* $(\beta_1'[i], q_1^i) \leftarrow \mathcal{S}earchOutput(\mathsf{vk}, \boldsymbol{U}, q_1^{i-1}, \alpha, 1, i, \epsilon)$ *for every* $i \in [\lambda]$*.*
2. *Compute* $(\beta_2'[i], q_2^i) \leftarrow \mathcal{S}earchOutput(\mathsf{vk}, \boldsymbol{U}, q_2^{i-1}, \alpha, 2, i, \epsilon)$ *for every* $i \in [\lambda]$*, where* $q_2^0 = q_1^\lambda$*.*
3. *Output* $\beta_1'[1] \| \cdots \| \beta_1'[\lambda] \| \beta_2'[1] \| \cdots \beta_2'[\lambda]$*.*

Note that they are the same as the computations done by $\boldsymbol{V}_\alpha$*. Then, for every* $d \in [2]$ *and* $i \in [\ell_{\mathsf{out}}]$*, we have* $\beta_d'[i] = \beta_d[i]$ *with overwhelming probability.*

Proof of Lemma 5.6. We prove this lemma using Lemma 2.4. To this end, we below show that for any $d \in [2]$ and $i \in [\ell_{\mathsf{out}}]$, $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,\epsilon-5\epsilon'}$ applied to q_1^0 results in $\beta_d[i]$ with overwhelming probability. (Recall that $\mathcal{S}earchOutput(\mathsf{vk}, \boldsymbol{U}, q_1^{i-1}, \alpha, d, i, \epsilon)$ outputs the result of $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,\epsilon-5\epsilon'}$ applied to the input state q_d^{i-1}.)

Let $d \in [2]$ and $i \in [\ell_{\mathsf{out}}]$ be arbitrary. If $\beta_d[i] = 0$, from the statistical binding property of Com, for a sample $\mathsf{fe.ct}_d^i \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}_d, (i, \alpha, u_d^i))$ generated by D_d^i, the statement $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m} = \mathsf{fe.ct}_d^i, \gamma)$ is a false statement since $\mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{fe.ct}_d^i) = \alpha$ holds. Suppose $\mathrm{Tr}\left[\mathcal{TI}_{\epsilon-6\epsilon'}(\mathcal{P}_{D_d^i})q_1^0\right]$ is not negligible. This means that if we give a randomly generated $\mathsf{fe.ct}_d^i \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}_d, (i, \alpha, u_d^i))$ to the quantum program with classical input and output $(q_1^0, \boldsymbol{U})$, we can obtain a proof π with non-negligible probability for the false statement $x = (\mathsf{ck}, \mathsf{com}, \mathsf{m} = \mathsf{fe.ct}_d^i, \gamma)$ such that $\mathsf{NIZK.Vrfy}(\mathsf{crs}, x, \pi) = 1$, which contradict to the adaptive exclusive soundness of NIZK.[14] Therefore, we have $\mathrm{Tr}\left[\mathcal{TI}_{\epsilon-6\epsilon'}(\mathcal{P}_{D_d^i})q_1^0\right] = \mathsf{negl}(\lambda)$. Then, from Lemma 2.11, we have $\mathrm{Tr}\left[\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,\epsilon-5\epsilon'}q_1^0\right] = \mathsf{negl}(\lambda)$. This means $\beta_d'[i]$ that is the result of $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,t}q_1^0$ is 0 with overwhelming probability if $\beta_d[i] = 0$.

If $\beta_d[i] = 1$, from the 1-bounded simulation security and ciphertext uniformity of FE, randomly generated $\mathsf{fe.ct}_d^i \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}_d, (i, \alpha, u_d^i))$ is indistinguishable from a uniformly random message since $u_d^i = \mathsf{FE.Dec}(\mathsf{fsk}_d, \mathsf{fe.ct}_d^i)$ and u_d^i is a uniformly random string. In other words, if $\beta_d[i] = 1$, D_i^d is indistinguishable from $U_{\mathcal{MS}}$. By combining this fact with Theorem 2.10 and Lemma 2.11, we have

$$
\begin{aligned}
\mathrm{Tr}\left[\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,\epsilon-5\epsilon'}q_1^0\right] &\geq \mathrm{Tr}\left[\mathcal{TI}_{\epsilon-4\epsilon'}(\mathcal{P}_{D_i^d})q_1^0\right] - \mathsf{negl}(\lambda) \\
&\geq \mathrm{Tr}\left[\mathcal{TI}_{\epsilon-3\epsilon'}(\mathcal{P}_{U_{\mathcal{MS}}})q_1^0\right] - \mathsf{negl}(\lambda) \\
&\geq 1 - \mathsf{negl}(\lambda).
\end{aligned}
$$

For the third inequality, we use the third item of Lemma 2.11. This means $\beta_d'[i]$ that is the result of $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D_d^i,\epsilon-5\epsilon'}q_1^0$ is 1 with overwhelming probability if $\beta_1[1] = 1$.

[14]The reduction in this step can always output (x, π) such that x is a false statement since the reduction can generate $(\mathsf{ck}, \mathsf{com}, \mathsf{fe.msk}, f_{\alpha,\beta}, \mathsf{aux}, \gamma)$. Thus, it is a valid adversary for the adaptive exclusive soundness.

The above combined with Lemma 2.4 proves the lemma, by considering a sequence of binary outcome measurements where $(d-1)\cdot \ell_{\mathsf{out}} + i$-th one is a measurement that results in 1 if the result of $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},D^i_d,\epsilon-5\epsilon'}$ is $\beta_d[i]$. This completes the proof. $\square$

From the above discussions, we see that if the event $\mathsf{NotAbort}$ occurs, $\mathcal{P}[\widetilde{C}] = (q_1^0, \boldsymbol{V})$ maps α to $\beta = \beta_1 \| \beta_2$ with overwhelming probability. Then, from the correctness of UOPF, $\mathcal{Extract}(\mathsf{vk}, \widetilde{C}, \epsilon)$ outputs μ correctly in this case. This means $\Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{NotAbort}] - \mathsf{negl}(\lambda) \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$ holds. $\blacksquare$

We prove $\Pr[\mathsf{BadExt}] \leq \mathsf{negl}(\lambda)$ and $\Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$. Hence, we complete the proof of unremovability. $\square$

6 Impossibility of Universal Copy Protection for Signatures

In this section, we show the impossibility of universal copy protection for signatures. We first formally define the notion of universal copy protection for signatures, and then prove its impossibility.

6.1 Definitions

Definition 6.1 (Copy Protected Signature). *A copy protected signature scheme with the message space $\mathcal{M}$ is a tuple of quantum algorithms $(\mathcal{Gen}, \mathcal{Sign}, \mathsf{Vrfy})$.*

$\mathcal{Gen}(1^\lambda) \rightarrow (\mathsf{vk}, \mathcal{sigk})$**:** *The key generation algorithm takes as input the security parameter 1^λ and outputs a verification key vk and quantum signing key $\mathcal{sigk}$.*

$\mathcal{Sign}(\mathcal{sigk}, \mathsf{m}) \rightarrow \sigma$**:** *The signing algorithm takes as input $\mathcal{sigk}$ and a message $\mathsf{m} \in \mathcal{M}$ and outputs a signature σ.*

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \sigma) \rightarrow 0/1$**:** *The verification algorithm takes as input vk, m, and σ, and outputs 0 or 1.*

Verification Correctness: *For any $\mathsf{m} \in \mathcal{M}$, it holds that*

$$\Pr\left[\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \sigma) = 1 \;\middle|\; \begin{array}{l}(\mathsf{vk}, \mathcal{sigk}) \leftarrow \mathcal{Gen}(1^\lambda)\\ \sigma \leftarrow \mathcal{Sign}(\mathcal{sigk}, \mathsf{m})\end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

Remark 6.2. A copy protected signature scheme would need to satisfy reusability that ensures that a quantum signing key can be used many times to generate signatures. Since our focus is impossibility, we do not require reusability and work with a weaker definition, which makes our impossibility strong.

Definition 6.3 (Anti-Piracy for Copy Protected Signature). *Let $\mathsf{CPSIG} = (\mathcal{Gen}, \mathcal{Sign}, \mathsf{Vrfy})$ be a copy protected signature scheme with the message space $\mathcal{M}$. We consider the following security experiment $\mathsf{Exp}^{\mathsf{anti\text{-}piracy}}_{\mathsf{CPSIG},\mathcal{A}}(1^\lambda)$, where $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$.*

1. *The challenger generates $(\mathsf{vk}, \mathcal{sigk}) \leftarrow \mathcal{Gen}(1^\lambda)$ and sends $(\mathsf{vk}, \mathcal{sigk})$ to $\mathcal{A}_0$.*
2. *$\mathcal{A}_0$ creates a bipartite state q over registers R_1 and R_2. $\mathcal{A}$ sends $q[\mathsf{R}_1]$ and $q[\mathsf{R}_2]$ to $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively.*
3. *The challenger samples $\mathsf{m}_1, \mathsf{m}_2 \leftarrow \mathcal{M}$ and sends m_1 to $\mathcal{A}_1$ and m_2 to $\mathcal{A}_2$. $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively output σ_1 and σ_2. If $\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}_i, \sigma_i) = 1$ for $i \in \{1,2\}$, the challenger outputs 1, otherwise outputs 0.*

We say that CPSIG satisfies anti-piracy if for any QPT $\mathcal{A}$, it holds that

$$\mathsf{Adv}^{\mathsf{anti\text{-}piracy}}_{\mathsf{CPSIG},\mathcal{A}}(\lambda) := \Pr\Big[\mathsf{Exp}^{\mathsf{anti\text{-}piracy}}_{\mathsf{CPSIG},\mathcal{A}}(1^\lambda) = 1\Big] \leq \mathsf{negl}(\lambda).$$

We now define universal copy protection for signatures.

Definition 6.4 (Universal Copy Protection for Signatures). *A universal copy protection scheme for signatures is a tuple of quantum algorithms* $(\mathcal{UTG}, \mathcal{USign})$.

$\mathcal{UTG}(\mathsf{sigk}) \to \mathit{sigk}$**:** *The universal token generation algorithm takes as input a classical signing key of a signature scheme* sigk *and outputs a quantum signing key* sigk.

$\mathcal{USign}(\mathit{sigk}, \mathsf{m}) \to \sigma$**:** *The universal signing algorithm takes as input* sigk *and a message* $\mathsf{m} \in \mathcal{M}$ *and outputs a signature* σ.

Universal Copy Protection: *For any signature scheme* $\mathsf{SIG} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *satisfying EUF-qCMA security,* $(\mathcal{Gen}[\mathsf{Gen}, \mathcal{UTG}], \mathcal{USign}, \mathsf{Vrfy})$ *is a copy protected signature scheme satisfying anti-piracy, where* $\mathcal{Gen}[\mathsf{Gen}, \mathcal{UTG}]$ *is a quantum algorithm that takes* 1^λ *as input, run* $(\mathsf{vk}, \mathsf{sigk}) \leftarrow \mathsf{Gen}(1^\lambda)$ *and* $\mathit{sigk} \leftarrow \mathcal{UTG}(\mathsf{sigk})$*, and outputs* $(\mathsf{vk}, \mathit{sigk})$.

6.2 Counter Example Construction

We construct $\mathsf{CESIG} = (\mathsf{CE.Gen}, \mathsf{CE.Sign}, \mathsf{CE.Vrfy})$. The building blocks are as follows.

- An EUF-qCMA secure signature scheme $\mathsf{qCMASIG} = (\mathsf{qCMA.Gen}, \mathsf{qCMA.Sign}, \mathsf{qCMA.Vrfy})$.
- A pre-embedded white-box watermarking signature scheme $\mathsf{PWMSIG} = (\mathsf{PWMSIG.Gen}, \mathsf{PWMSIG.Sign}, \mathsf{PWMSIG.Vrfy}, \mathsf{PWMSIG}.\mathcal{Extract})$.
- A OWF $f : \{0,1\}^n \to \{0,1\}^m$.

The construction of CESIG is as follows.

$\mathsf{CE.Gen}(1^\lambda)$**:**

- Generate $(\mathsf{qcma.vk}, \mathsf{qcma.sigk}) \leftarrow \mathsf{qCMA.Gen}(1^\lambda)$.
- Generate $x \leftarrow \{0,1\}^n$ and compute $y \leftarrow f(x)$.
- Generate $(\mathsf{pwm.vk}, \mathsf{pwm.sigk}) \leftarrow \mathsf{PWMSIG.Gen}(1^\lambda, x)$.
- Output $\mathsf{ce.vk} := (\mathsf{qcma.vk}, \mathsf{pwm.vk}, y)$ and $\mathsf{ce.sigk} := (\mathsf{qcma.sigk}, \mathsf{pwm.sigk})$.

$\mathsf{CE.Sign}(\mathsf{ce.sigk}, \mathsf{m})$**:**

- Parse $\mathsf{ce.sigk} = (\mathsf{qcma.sigk}, \mathsf{pwm.sigk})$.
- Generate $\mathsf{qcma}.\sigma \leftarrow \mathsf{qCMA.Sign}(\mathsf{qcma.sigk}, \mathsf{m})$ and $\mathsf{pwm}.\sigma \leftarrow \mathsf{PWMSIG.Sign}(\mathsf{pwm.sigk}, \mathsf{m})$.
- Output $\mathsf{ce}.\sigma := (\mathsf{qcma}.\sigma, \mathsf{pwm}.\sigma)$.

$\mathsf{CE.Vrfy}(\mathsf{ce.vk}, \mathsf{m}, \mathsf{ce}.\sigma)$**:**

- Parse $\mathsf{ce.vk} = (\mathsf{qcma.vk}, \mathsf{pwm.vk}, y)$ and output 1 if $f(\mathsf{ce}.\sigma) = y$. Otherwise, parse $\mathsf{ce}.\sigma = (\mathsf{qcma}.\sigma, \mathsf{pwm}.\sigma)$ and go to the next step.
- Output 1 if $\mathsf{qCMA.Vrfy}(\mathsf{qcma.vk}, \mathsf{m}, \mathsf{qcma}.\sigma) = 1$ and $\mathsf{PWMSIG.Vrfy}(\mathsf{pwm.vk}, \mathsf{m}, \mathsf{pwm}.\sigma) = 1$, and otherwise output 0.

Theorem 6.5. *Assuming* $\mathsf{qCMASIG}$ *is EUF-qCMA secure,* PWMSIG *satisfies privacy, and* f *is OWFs,* CESIG *is EUF-qCMA secure.*

Theorem 6.6. *Assume* PWMSIG *satisfies unremovability. Let* $(\mathcal{UTG}, \mathcal{USign})$ *be a pair of quantum algorithms such that it meets the syntactical requirement in Definition 6.4 and* $(\mathcal{Gen}[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{USign}, \mathsf{CE.Vrfy})$ *satisfies verification correctness as copy protected signature scheme, where* $\mathcal{Gen}[\mathsf{CE.Gen}, \mathcal{UTG}]$ *is a quantum algorithm that takes as input* 1^λ*, runs* $(\mathsf{ce.vk}, \mathsf{ce.sigk}) \leftarrow \mathsf{CE.Gen}(1^\lambda)$ *and* $\mathit{sigk} \leftarrow \mathcal{UTG}(\mathsf{ce.sigk})$*, and outputs* $(\mathsf{ce.vk}, \mathit{sigk})$*. Then,* $(\mathcal{Gen}[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{USign}, \mathsf{CE.Vrfy})$ *does not satisfy anti-piracy for copy protected signature.*

Proof of Theorem 6.5. We use the following sequence of experiments.

Hyb_0**:** This is $\mathsf{Exp}^{\mathsf{euf\text{-}qcma}}_{\mathsf{CESIG},\mathcal{A}}(1^\lambda)$.

We have $\mathsf{Adv}^{\mathsf{euf\text{-}qcma}}_{\mathsf{CESIG},\mathcal{A}}(\lambda) = \Pr[\mathsf{Hyb}_0 = 1]$.

Hyb_1**:** This is the same as Hyb_0 except that the challenger generates $(\mathsf{pwm.vk}, \mathsf{pwm.sigk})$ as $(\mathsf{pwm.vk}, \mathsf{pwm.sigk}) \leftarrow \mathsf{PWMSIG.Gen}(1^\lambda, 0^n)$ instead of $(\mathsf{pwm.vk}, \mathsf{pwm.sigk}) \leftarrow \mathsf{PWMSIG.Gen}(1^\lambda, x)$.

From the privacy of PWMSIG, we have $|\Pr[\mathsf{Hyb}_0 = 1] - \Pr[\mathsf{Hyb}_1 = 1]| = \mathsf{negl}(\lambda)$.

In Hyb_1, the probability that $f(\mathsf{ce}.\sigma_i) = y$ holds for some $i \in [q+1]$ is negligible from the one-wayness of f, where q is the number of queries made by $\mathcal{A}$ and $(\mathsf{m}_i, \mathsf{ce}.\sigma_i)_{i\in[q+1]}$ is the final output of $\mathcal{A}$. Then, we can directly obtain $\Pr[\mathsf{Hyb}_1 = 1] = \mathsf{negl}(\lambda)$ from the EUF-qCMA security of qCMASIG. $\square$

Proof of Theorem 6.6. We define the following event.

`FindPreimage`**:** We execute $(\mathsf{ce.vk}, \mathit{sigk}) \leftarrow \mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}](1^\lambda)$, where $\mathsf{ce.vk} := (\mathsf{qcma.vk}, \mathsf{pwm.vk}, y)$. Next, we sample $\mathsf{m} \leftarrow \mathcal{M}$ and generate $\sigma \leftarrow \mathcal{US}ign(\mathit{sigk}, \mathsf{m})$. Then, it holds that $f(\sigma) = y$.

We consider the following two cases separately.

The case $\Pr[\mathtt{FindPreimage}]$ **is not negligible.** Consider the following adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ for the anti-piracy of $(\mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{US}ign, \mathsf{CE.Vrfy})$.

$\mathcal{A}_0$**:** Given $\mathsf{ce.vk}$ and sigk, it samples $m \leftarrow \mathcal{M}$, computes $\sigma \leftarrow \mathcal{US}ign(\mathit{sigk}, \mathsf{m})$, and sends σ to $\mathcal{A}_1$ and $\mathcal{A}_2$.

$\mathcal{A}_i(i \in \{1,2\})$**:** Given σ from $\mathcal{A}_0$ and the challenge message m_i, it outputs σ.

If the event `FindPreimage` happens in the security game, $\mathcal{A}$ wins since if $f(\sigma) = y$, $\mathsf{CE.Vrfy}(\mathsf{ce.vk}, \mathsf{m}, \sigma) = 1$ for any $\mathsf{m} \in \mathcal{M}$. Since we assume $\Pr[\mathtt{FindPreimage}]$ is not negligible, $\mathcal{A}$ breaks the anti-piracy of $(\mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{US}ign, \mathsf{CE.Vrfy})$.

The case $\Pr[\mathtt{FindPreimage}]$ **is negligible.** In this case, from the fact that $(\mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{US}ign, \mathsf{CE.Vrfy})$ satisfies verification correctness as a copy protected signature scheme, we have

$$\underset{\substack{(\mathsf{qcma.vk},\mathsf{qcma.sigk})\leftarrow\mathsf{qCMA.Gen}(1^\lambda)\\ x\leftarrow\{0,1\}^n\\ (\mathsf{pwm.vk},\mathsf{pwm.sigk})\leftarrow\mathsf{pwm.Gen}(1^\lambda,x)\\ \mathsf{ce.sigk}:=(\mathsf{qcma,sigk,pwm.sigk})\\ \mathit{sigk}\leftarrow\mathcal{UTG}(\mathsf{ce.sigk})}}{\mathbb{E}} \left[\Pr\left[\mathsf{pwm.Vrfy}(\mathsf{pwm.vk}, \mathsf{m}, \mathsf{pwm}.\sigma) = 1 \;\middle|\; \begin{array}{l} \mathsf{m} \leftarrow \mathcal{M} \\ \sigma \leftarrow \mathcal{S}ign(\mathit{sigk}, \mathsf{m}) \\ \text{Parse } \sigma = (\mathsf{qcma}.\sigma, \mathsf{pwm}.\sigma) \end{array}\right]\right]$$

$$= 1 - \mathsf{negl}(\lambda). \quad (1)$$

Consider the following adversary $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1, \mathcal{B}_2)$ for the anti-piracy of $(\mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{US}ign, \mathsf{CE.Vrfy})$.

$\mathcal{A}_0$**:** Given $\mathsf{ce.vk}$ and sigk, it constructs $\boldsymbol{U} = \{\boldsymbol{U}_m\}_{\mathsf{m}\in\mathcal{MS}}$, where $\boldsymbol{U}_m$ is the unitary that when applied to sigk and $|0...0\rangle$, computes $\sigma \leftarrow \mathcal{US}ign(\mathit{sigk}, \mathsf{m})$ and writes σ to the first register of the ancilla. It then computes $x' \leftarrow \mathsf{pwm.Extract}(\mathsf{pwm.vk}, (\mathit{sigk}, \boldsymbol{U}), 2/3)$[15] and sends x' to $\mathcal{A}_1$ and $\mathcal{A}_2$.

$\mathcal{A}_i(i \in \{1,2\})$**:** Given x' from $\mathcal{A}_0$ and the challenge message m_i, it outputs x'.

From Equation (1) and the unremovability of PWMSIG, $f(x') = y$ holds with overwhelming probability, where $\mathsf{ce.vk} = (\mathsf{qcma.vk}, \mathsf{pwm.vk}, y)$. Thus, $\mathcal{A}$ breaks the anti-piracy of $(\mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{US}ign, \mathsf{CE.Vrfy})$.

Overall, $(\mathcal{G}en[\mathsf{CE.Gen}, \mathcal{UTG}], \mathcal{US}ign, \mathsf{CE.Vrfy})$ does not satisfy anti-piracy. $\square$

[15] The choice of the threshold parameter $2/3$ is arbitrary. We can use any constant between 0 and 1.

7 White-Box Watermarking Signature

In this section, we present a non-black-box conversion from a pre-embedded white-box watermarking signature scheme constructed in Section 5 into a white-box watermarking signature scheme.

We construct $\mathsf{WMSIG} = \mathsf{WMSIG}.(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathcal{Extract})$ using the following building blocks.

- Standard signature $\mathsf{SIG}.(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy})$.
- Pre-embedded white-box watermarking signature $\mathsf{PWMSIG}.(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathcal{Extract})$ presented in Section 5.

We can use $(\mathsf{PWMSIG}.\mathsf{KeyGen}, \mathsf{PWMSIG}.\mathsf{Sign}, \mathsf{PWMSIG}.\mathsf{Vrfy})$ in a black-box way. However, we cannot use $\mathsf{PWMSIG}.\mathcal{Extract}$ in a black-box way, so we need to write down the algorithm of $\mathsf{PWMSIG}.\mathcal{Extract}$ in $\mathsf{WMSIG}.\mathcal{Extract}$.

$\mathsf{WMSIG}.\mathsf{KeyGen}(1^\lambda)$**:**

- Generate $(\mathsf{sig.vk}, \mathsf{sig.sk}) \leftarrow \mathsf{SIG}.\mathsf{KeyGen}(1^\lambda)$.
- Output $\mathsf{vk} := \mathsf{sig.vk}$ and $\mathsf{sk} := \mathsf{sig.sk}$.

$\mathsf{WMSIG}.\mathsf{Sign}(\mathsf{sk}, \mathsf{m})$**:**

- Parse $\mathsf{sk} = \mathsf{sig.sk}$.
- Generate $\mathsf{sig}.\sigma \leftarrow \mathsf{SIG}.\mathsf{Sign}(\mathsf{sig.sk}, 0\|\mathsf{m})$.
- Output $(\bot, \bot, \mathsf{sig}.\sigma)$.

$\mathsf{WMSIG}.\mathsf{Vrfy}(\mathsf{vk}, \mathsf{m}, \sigma)$**:**

- Parse $\mathsf{vk} = \mathsf{sig.vk}$ and $\sigma = (\mathsf{pwm.vk}, \mathsf{pwm}.\sigma, \mathsf{sig}.\sigma)$.
- If $\mathsf{pwm.vk} = \bot$, output $\mathsf{SIG}.\mathsf{Vrfy}(\mathsf{sig.vk}, 0\|\mathsf{m}, \mathsf{sig}.\sigma)$. Otherwise, go to the next step.
- Output 1 if $\mathsf{PWMSIG}.\mathsf{Vrfy}(\mathsf{pwm.vk}, \mathsf{m}, \mathsf{pwm}.\sigma) = 1$ and $\mathsf{SIG}.\mathsf{Vrfy}(\mathsf{sig.vk}, 1\|\mathsf{pwm.vk}, \mathsf{sig}.\sigma) = 1$, and otherwise output 0.

$\mathsf{WMSIG}.\mathsf{Mark}(\mathsf{sk}, \mu)$**:**

- Parse $\mathsf{sk} = \mathsf{sig.sk}$.
- Generate $(\mathsf{pwm.vk}, \mathsf{pwm.sk}) \leftarrow \mathsf{PWMSIG}.\mathsf{KeyGen}(1^\lambda, \mu)$.
- Generate $\mathsf{sig}.\sigma \leftarrow \mathsf{SIG}.\mathsf{Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk})$.
- Output the circuit $\widetilde{\mathsf{C}}[\mathsf{sig}.\sigma, \mathsf{pwm.vk}, \mathsf{pwm.sk}]$ that behaves as follows.
  1. Take as input m.
  2. Generate $\mathsf{pwm}.\sigma \leftarrow \mathsf{PWMSIG}.\mathsf{Sign}(\mathsf{pwm.sk}, \mathsf{m})$.
  3. Output $(\mathsf{pwm.vk}, \mathsf{pwm}.\sigma, \mathsf{sig}.\sigma)$.

$\mathsf{WMSIG}.\mathcal{Extract}(\mathsf{vk}, \mathcal{C}, \epsilon, (\mathsf{m}^*, \sigma^*))$**:**

- Let $\epsilon' = \epsilon/11$, $\delta' = 2^{-\lambda}$, $t = \epsilon - \epsilon'$, and $\tilde{t} = \epsilon - 4\epsilon'$.
- Parse $\mathsf{vk} = \mathsf{sig.vk}$, $\sigma^* = (\mathsf{pwm.vk}^*, \mathsf{pwm}.\sigma^*, \mathsf{sig}.\sigma^*)$, and $\mathcal{C} = (q, \boldsymbol{U})$.
- Let $\widetilde{\mathsf{Vrfy}}(\mathsf{vk}, \mathsf{m}, \sigma)$ be defined as follows: It parses $\mathsf{vk} = \mathsf{sig.vk}$ and $\sigma = (\mathsf{pwm.vk}, \mathsf{pwm}.\sigma, \mathsf{sig}.\sigma)$, and outputs 1 if $\mathsf{pwm.vk} = \mathsf{pwm.vk}^*$, $\mathsf{PWMSIG}.\mathsf{Vrfy}(\mathsf{pwm.vk}, \mathsf{m}, \mathsf{pwm}.\sigma) = 1$, and $\mathsf{SIG}.\mathsf{Vrfy}(\mathsf{sig.vk}, 1\|\mathsf{pwm.vk}, \mathsf{sig}.\sigma) = 1$. Otherwise, it outputs 0. If $\widetilde{\mathsf{Vrfy}}(\mathsf{vk}, \mathsf{m}^*, \sigma^*) = 0$, output unmarked.

- We let $\mathcal{P} = (\boldsymbol{P}_{\mathsf{m}}, \boldsymbol{Q}_{\mathsf{m}})_{\mathsf{m}}$ and $\widetilde{\mathcal{P}} = (\widetilde{\boldsymbol{P}}_{\mathsf{m}}, \widetilde{\boldsymbol{Q}}_{\mathsf{m}})_{\mathsf{m}}$ be collections of binary outcome projective measurements, where

$$\boldsymbol{P}_{\mathsf{m}} = \boldsymbol{U}_{\mathsf{m}}^{\dagger} \boldsymbol{U}_{\mathsf{WMSIG.Vrfy,m}}^{\dagger} (\boldsymbol{I} \otimes |1\rangle \langle 1|) \boldsymbol{U}_{\mathsf{WMSIG.Vrfy,m}} \boldsymbol{U}_{\mathsf{m}}, \quad \boldsymbol{Q}_{\mathsf{m}} = \boldsymbol{I} - \boldsymbol{P}_{\mathsf{m}}$$
$$\widetilde{\boldsymbol{P}}_{\mathsf{m}} = \boldsymbol{U}_{\mathsf{m}}^{\dagger} \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}}^{\dagger} (\boldsymbol{I} \otimes |1\rangle \langle 1|) \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \boldsymbol{U}_{\mathsf{m}}, \quad \boldsymbol{Q}_{\mathsf{m}} = \boldsymbol{I} - \widetilde{\boldsymbol{P}}_{\mathsf{m}}.$$

  We also let $U_{\mathcal{MS}}$ be the uniform distribution over $\mathcal{MS}$.
- Compute $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},U_{\mathcal{MS}},t} q$ and output unmarked if the outcome is 0. Otherwise, letting the post state be q', go to the next step.
- Compute $\mathcal{ATI}^{\epsilon',\delta'}_{\widetilde{\mathcal{P}},U_{\mathcal{MS}},\tilde{t}} q'$ and output unmarked if the outcome is 0. Otherwise, letting the post state be q_1^0, go to the next step.
- Construct $\boldsymbol{V}$ that is a compact description of $\{\boldsymbol{V}_x\}_x$, where $\boldsymbol{V}_x$ is a unitary that performs the following computations coherently when applied to a quantum state q.
  1. Set $q = q_1^0$.
  2. Compute $(\beta_1'[i], q_1^i) \leftarrow \mathcal{SearchOutput}(\mathsf{pwm.vk}^*, \boldsymbol{U}, q_1^{i-1}, x, 1, i, \epsilon)$ for every $i \in [\lambda]$.
  3. Compute $(\beta_2'[i], q_2^i) \leftarrow \mathcal{SearchOutput}(\mathsf{pwm.vk}^*, \boldsymbol{U}, q_2^{i-1}, x, 2, i, \epsilon)$ for every $i \in [\lambda]$, where $q_2^0 = q_1^\lambda$.
  4. Output $\beta_1'[1] \| \cdots \| \beta_1'[\lambda] \| \beta_2'[1] \| \cdots \beta_2'[\lambda]$.
- Construct a quantum program with classical input and output $\mathcal{P}[\mathcal{C}] = (q_1^0, \boldsymbol{V})$.
- Output $\mu' \leftarrow \mathsf{UOPF}.\mathcal{Extract}(\mathcal{P}[\mathcal{C}], \mathsf{aux})$.

$\mathcal{SearchOutput}(\mathsf{pwm.vk}, \boldsymbol{U}, q, x, d, i, \epsilon)$

Input: $\mathsf{pwm.vk}$, $\boldsymbol{U}$, q, x, ϵ.

1. Parse $\mathsf{pwm.vk} = (\mathsf{crs}, \gamma, \mathsf{fe.pk}_1, \mathsf{fe.pk}_2, \mathsf{ck}, \mathsf{com}, \mathsf{aux})$.
2. Let $\epsilon' = \epsilon/8$, $\delta' = 2^{-\lambda}$, and $t = \epsilon - 6\epsilon'$.
3. Define D_d^i be the following distribution.
   - Generate $u_d^i \leftarrow \{0,1\}^\lambda$.
   - Output $\mathsf{fe.ct}_d^i \leftarrow \mathsf{FE.Enc}(\mathsf{fe.pk}_d, (i, x, u_d^i))$.
4. Compute $\beta_d'[i] \leftarrow \mathcal{ATI}^{\epsilon',\delta'}_{\widetilde{\mathcal{P}},D_d^i,t} q$.
5. Uncompute the previous step and output $\beta_d'[i]$ and the resulting state.

Figure 4: The description of $\mathcal{SearchOutput}$

The construction idea is simple. We generate a fresh key pair of PWMSIG for each mark μ and authenticate the verification key of PWMSIG by generating a signature of SIG. Each security property except unremovability easily follow from the corresponding security property of PWMSIG and the EUF-CMA security of SIG since those security properties do not use the extraction algorithm. The analysis of unremovability requires care. We provide some extended properties of ATI for signatures whose verification consists of two verification steps in Appendix C to prove the unremovability of the construction above. The intuition is as follows. The adversary given a marked circuit cannot forge a signature under sig.vk, and a pirate circuit generated by the adversary must generate a signature passing $\widetilde{\mathsf{Vrfy}}$ that is essentially the same verification algorithm of PWMSIG. Hence, we can use the same extraction strategy as PWMSIG.

We prove the following theorem.

Theorem 7.1. *Assume* SIG *is an EUF-CMA secure signature scheme and* PWMSIG *is a pre-embedded white-box watermarking signature scheme against quantum adversaries,* WMSIG *is a white-box watermarking signature scheme against quantum adversaries.*

We need to prove the following theorems to prove Theorem 7.1.

Theorem 7.2. *Assume* PWMSIG *satisfies strong correctness of marked keys. Then,* WMSIG *satisfies strong correctness of marked keys.*

Theorem 7.3. *Assume* SIG *is EUF-CMA and* PWMSIG *is unforgeable. Then,* WMSIG *is unforgeable.*

Theorem 7.4. *Assume* SIG *is EUF-CMA and* PWMSIG *satisfies unremovability. Then,* WMSIG *satisfies unremovability.*

Theorem 7.5. PWMSIG *satisfies privacy. Then,* WMSIG *satisfies privacy.*

We prove these theorems below.

Proof of Theorem 7.2. We construct an algorithm $\mathcal{B}$ that attacks the strong correctness of marked keys of PWMSIG by using an adversary $\mathcal{A}$ that attacks the strong correctness of marked keys of WMSIG. $\mathcal{B}$ proceeds as follows.

1. $\mathcal{B}$ generates $(\mathsf{sig.vk}, \mathsf{sig.sk}) \leftarrow \mathsf{SIG.KeyGen}(1^\lambda)$ and passes $\mathsf{sig.vk}$ to $\mathcal{A}$.

2. When $\mathcal{A}$ sends μ as a challenge, $\mathcal{B}$ forwards it to its challenger and receives $\mathsf{pwm.vk}$. $\mathcal{B}$ also generates $\mathsf{sig}.\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk})$.

3. When $\mathcal{A}$ sends a query m_i to $O_{\mathtt{sign}}$, $\mathcal{B}$ generates $\mathsf{sig}.\sigma_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 0\|\mathsf{m}_i)$, and sends $(\bot, \bot, \mathsf{sig}.\sigma_i)$ to $\mathcal{A}$.

4. When $\mathcal{A}$ sends a query m_i to $O_{\mathtt{msign}}$, $\mathcal{B}$ forwards m_i to its signing oracle and receives $\mathsf{pwm}.\sigma_i \leftarrow \mathsf{PWMSIG.Sign}(\mathsf{pwm.sk}, \mathsf{m}_i)$. Then, $\mathcal{B}$ sends $(\mathsf{pwm.vk}, \mathsf{pwm}.\sigma_i, \mathsf{sig}.\sigma)$ to $\mathcal{A}$.

5. When $\mathcal{A}$ outputs m^*, $\mathcal{B}$ outputs m^*.

$\mathcal{B}$ perfectly simulates the challenger of the security game played by $\mathcal{A}$. Let $(\mathsf{pwm.vk}, \mathsf{pwm.sk}) \leftarrow \mathsf{PWMSIG.KeyGen}(1^\lambda, \mu)$ be the key pair of PWMSIG generated by the challenger of the security game played by $\mathcal{B}$. When we generate $(\mathsf{pwm.vk}, \mathsf{pwm}.\sigma, \mathsf{sig}.\sigma) \leftarrow \widetilde{C}[\mathsf{sig}.\sigma, \mathsf{pwm.vk}, \mathsf{pwm.sk}](\mathsf{m}^*)$, whether $(\mathsf{pwm.vk}, \mathsf{pwm}.\sigma, \mathsf{sig}.\sigma)$ is valid or not depends only on whether $\mathsf{pwm}.\sigma$ is valid or not since $\mathsf{sig}.\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk})$ and $\mathsf{pwm.vk} \neq \bot$. This means $\mathsf{Adv}^{\mathsf{scorrect}}_{\mathsf{PWMSIG},\mathcal{B}}(\lambda)$ is the same as $\mathsf{Adv}^{\mathsf{scorrect}}_{\mathsf{WMSIG},\mathcal{A}}(\lambda)$. This completes the proof. $\square$

Proof of Theorem 7.3. Let Reuse be an event that the adversary $\mathcal{A}$ outputs a valid forgery $(\mathsf{m}^*, (\mathsf{pwm.vk}^*, \mathsf{pwm.sig}^*, \mathsf{sig}.\sigma^*))$ such that $\mathsf{pwm.vk}^* \neq \bot$ and $\mathsf{pwm.vk}^* = \mathsf{pwm.vk}_i$ for some i, which was generated by $O_{\mathtt{msign}}$. First, we show $\Pr[\mathsf{Reuse}]$ is negligible. Suppose $\Pr[\mathsf{Reuse}]$ is non-negligible. We construct an algorithm $\mathcal{B}$ that breaks the unforgeability of PWMSIG by using the adversary $\mathcal{A}$ in the unforgeability game of WMSIG. $\mathcal{B}$ proceeds as follows.

1. $\mathcal{B}$ chooses $i^* \leftarrow [q_m]$ where q_m is the total number of queries to $O_{\mathtt{msign}}$.

2. $\mathcal{B}$ generates $(\mathsf{sig.vk}, \mathsf{sig.sk}) \leftarrow \mathsf{SIG.KeyGen}(1^\lambda)$ and sends $\mathsf{sig.vk}$ to $\mathcal{A}$.

3. When $\mathcal{A}$ sends a signing query m_i, $\mathcal{B}$ generates $\mathsf{sig}.\sigma_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 0\|\mathsf{m}_i)$ and returns $(\bot, \bot, \mathsf{sig}.\sigma_i)$ to $\mathcal{A}$.

4. When $\mathcal{A}$ send a marked signing query (m_i, μ_i), $\mathcal{B}$ does the following.

   - If it is the i^*-th marked signing query, $\mathcal{B}$ forwards μ_{i^*} to its challenger, receives $\mathsf{pwm.vk}_{i^*}$, sends m_{i^*} to its signing oracle, receives $\mathsf{pwm.sig}_{i^*} \leftarrow \mathsf{PWMSIG.Sign}(\mathsf{pwm.sk}_{i^*}, \mathsf{m}_{i^*})$, generates $\mathsf{sig}.\sigma_{i^*} \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk}_{i^*})$, and returns $(\mathsf{pwm.vk}_{i^*}, \mathsf{pwm.sig}_{i^*}, \mathsf{sig}.\sigma_{i^*})$ to $\mathcal{A}$.
   - Otherwise, $\mathcal{B}$ generates $(\mathsf{pwm.vk}_i, \mathsf{pwm.sk}_i) \leftarrow \mathsf{PWMSIG.KeyGen}(1^\lambda, \mu_i)$, $\mathsf{pwm.sig}_i \leftarrow \mathsf{PWMSIG.Sign}(\mathsf{pwm.sk}_i, \mu_i)$, and $\mathsf{sig}.\sigma_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk}_i)$, and returns $(\mathsf{pwm.vk}_i, \mathsf{pwm}.\sigma_i, \mathsf{sig}.\sigma_i)$ to $\mathcal{A}$.

5. When $\mathcal{A}$ outputs $(\mathsf{m}^*, (\mathsf{pwm.vk}^*, \mathsf{pwm}.\sigma^*, \mathsf{sig}.\sigma^*))$, $\mathcal{B}$ outputs $(\mathsf{m}^*, \mathsf{pwm}.\sigma^*)$.

If Reuse happens, $\mathsf{pwm.vk}^* = \mathsf{pwm.vk}_{i^*}$ holds with probability $1/q_m$. In addition, it holds that $\mathsf{m}^* \neq \mathsf{m}_i$ for all i and $\mathsf{PWMSIG.Vrfy}(\mathsf{pwm.vk}^*, \mathsf{m}^*, \mathsf{pwm}.\sigma^*) = 1$ by the condition of the unforgeability game of WMSIG since $\mathsf{pwm.vk}^* \neq \bot$. Thus, $(\mathsf{m}^*, \mathsf{pwm}.\sigma^*)$ is valid forgery in the unforgeability game of PWMSIG, and $\Pr[\mathsf{Reuse}]$ must be negligible.

Next, we show that we can construct an algorithm $\mathcal{B}$ that breaks EUF-CMA of SIG by using an adversary $\mathcal{A}$ that breaks unforgeability of WMSIG. $\mathcal{B}$ proceeds as follows.

1. $\mathcal{B}$ is given $\mathsf{sig.vk}$, and sends $\mathsf{sig.vk}$ to $\mathcal{A}$.
2. When $\mathcal{A}$ sends a signing query m_i, $\mathcal{B}$ sends $0\|\mathsf{m}_i$ to its signing oracle and receives $\mathsf{sig}.\sigma_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 0\|\mathsf{m}_i)$. Then, $\mathcal{B}$ returns $(\bot, \bot, \mathsf{sig}.\sigma_i)$ to $\mathcal{A}$.
3. When $\mathcal{A}$ sends a marked signing query (m_i, μ_i), $\mathcal{B}$ generates $(\mathsf{pwm.vk}_i, \mathsf{pwm.sk}_i) \leftarrow \mathsf{PWMSIG.KeyGen}(1^\lambda, \mu_i)$, sends $1\|\mathsf{pwm.vk}_i$ to its signing oracle, receives $\mathsf{sig}.\sigma_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk}_i)$. Then, $\mathcal{B}$ returns $(\mathsf{pwm.vk}_i, \mathsf{pwm.sk}_i, \mathsf{sig}.\sigma_i)$ to $\mathcal{A}$.
4. When $\mathcal{A}$ outputs $(\mathsf{m}^*, (\mathsf{pwm.vk}^*, \mathsf{pwm.sig}^*, \mathsf{sig}.\sigma^*))$, $\mathcal{B}$ outputs $(0\|\mathsf{m}^*, \mathsf{sig}.\sigma^*)$ or $(1\|\mathsf{pwm.vk}^*, \mathsf{sig}.\sigma^*)$.

We consider two cases. One is that the forgery $(\mathsf{m}^*, (\mathsf{pwm.vk}^*, \mathsf{pwm.sig}^*, \mathsf{sig}.\sigma^*))$ is valid and $\mathsf{pwm.vk}^* = \bot$ holds. The other is that the forgery $(\mathsf{m}^*, (\mathsf{pwm.vk}^*, \mathsf{pwm.sig}^*, \mathsf{sig}.\sigma^*))$ is valid and $\mathsf{pwm.vk}^* \neq \bot$ holds.

In the former case, $\mathsf{SIG.Vrfy}(\mathsf{sig.vk}, 0\|\mathsf{m}^*, \mathsf{sig}.\sigma^*) = 1$ should hold. Then, $(0\|\mathsf{m}^*, \mathsf{sig}.\sigma^*)$ is a valid forgery in the EUF-CMA game of SIG since m^* should be different from all queries m_i by $\mathcal{A}$ due to the condition of unforgeability of WMSIG. Recall that $\mathcal{B}$ sends $\{0\|\mathsf{m}_i\}_i$ and $\{1\|\mathsf{pwm.vk}_i\}_i$ to its signing oracle.

In the latter case, $\mathsf{SIG.Vrfy}(\mathsf{sig.vk}, 1\|\mathsf{pwm.vk}^*, \mathsf{sig}.\sigma^*) = 1$ should hold. Then, $(1\|\mathsf{pwm.vk}^*, \mathsf{sig}.\sigma^*)$ is a valid forgery in the EUF-CMA game of SIG since $\mathcal{B}$ sends $\{0\|\mathsf{m}_i\}_i$ and $\{1\|\mathsf{pwm.vk}_i\}_i$ to its signing oracle and $1\|\mathsf{pwm.vk}^* \neq 0\|\mathsf{m}_i$ for all i due to the prefix bit and $\mathsf{pwm.vk}^* \neq \mathsf{pwm.vk}_i$ for all i ($\Pr[\mathsf{Reuse}]$ is negligible). This completes the proof. $\square$

Proof of Theorem 7.4. $\mathsf{WMSIG}.\mathcal{Extract}$ takes as input vk $C = (q, \boldsymbol{U})$, ϵ, and (m^*, σ^*), and first applies $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},\mathcal{U}_{\mathcal{MS}},t}$ to q and then applies $\mathcal{ATI}^{\epsilon',\delta'}_{\widetilde{\mathcal{P}},\mathcal{U}_{\mathcal{MS}},\tilde{t}}$ to q', where q' is the post state of $\mathcal{ATI}^{\epsilon',\delta'}_{\mathcal{P},\mathcal{U}_{\mathcal{MS}},t}q$. Proposition C.1 guarantees that if the former results in 1, then the latter also results in 1 with overwhelming probability, and we can obtain a $(\epsilon - 4\epsilon')$-live quantum program with respect to $\widetilde{\mathsf{Vrfy}}$ that is essentially the verification algorithm of PWMSIG. Then, we can prove Theorem 7.4 almost the same way as Theorem 5.4. We omit the details. $\square$

Proof of Theorem 7.5. Suppose that $\mathcal{A}$ breaks the privacy of WMSIG. We construct an adversary $\mathcal{B}$ that breaks the privacy of PWMSIG. When $\mathcal{A}$ sends $(\mathsf{sig.vk}, \mathsf{sig.sk})$ and (μ_0, μ_1), $\mathcal{B}$ sends (μ_0, μ_1) to its challenger, receives $\mathsf{pwm.vk}$, generates $\mathsf{sig}.\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sig.sk}, 1\|\mathsf{pwm.vk})$. When $\mathcal{A}$ sends a signing query m, $\mathcal{B}$ sends $|\mathsf{m}\rangle\,|0\rangle$ to its signing oracle, receives $|\mathsf{m}\rangle\,|\mathsf{pwm.sig}\rangle$, measure $\mathsf{pwm.sig}$, and returns $(\mathsf{pwm.vk}, \mathsf{pwm}.\sigma, \mathsf{sig}.\sigma)$. $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. $\mathcal{B}$ perfectly simulate the view for $\mathcal{A}$ since if the challenger for $\mathcal{B}$ chooses $\mathsf{coin} \leftarrow \{0,1\}$, it holds that $(\mathsf{pwm.vk}, \mathsf{pwm.sk}) \leftarrow \mathsf{PWMSIG.KeyGen}(1^\lambda, \mu_{\mathsf{coin}})$ and $\mathsf{pwm.sig} \leftarrow \mathsf{PWMSIG.Sign}(\mathsf{pwm.sk}, \mathsf{m})$. Hence, $\mathcal{B}$ breaks the privacy of PWMSIG if $\mathcal{A}$ breaks the privacy of WMSIG. $\square$

References

[Aar06] Scott Aaronson. Qma/qpoly \subseteq pspace/poly: De-merlinizing quantum protocols. In *Proceedings of the 21st Annual IEEE Conference on Computational Complexity, CCC 2006, 16-20 July 2006, Prague, Czech Republic*, pages 261–273. IEEE Computer Society, 2006. (Cited on page 9, 13.)

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 229–242. IEEE Computer Society, 2009. (Cited on page 4, 12.)

[ABDS21] Gorjan Alagic, Zvika Brakerski, Yfke Dulek, and Christian Schaffner. Impossibility of quantum virtual black-box obfuscation of classical circuits. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 497–525, Virtual Event, August 2021. Springer, Cham. (Cited on page 7, 11, 12.)

[AF07] Masayuki Abe and Serge Fehr. Perfect NIZK with adaptive soundness. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 118–136. Springer, Berlin, Heidelberg, February 2007. (Cited on page 19.)

[AF16] Gorjan Alagic and Bill Fefferman. On quantum obfuscation. *CoRR (arXiv)*, abs/1602.01771, 2016. (Cited on page 12.)

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 269–295. Springer, Cham, August 2019. (Cited on page 15.)

[AK22] Prabhanjan Ananth and Fatih Kaleoglu. A note on copy-protection from random oracles. Cryptology ePrint Archive, Report 2022/1109, 2022. (Cited on page 4, 12.)

[AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 57–74. Springer, Berlin, Heidelberg, August 2013. (Cited on page 16.)

[AL21] Prabhanjan Ananth and Rolando L. La Placa. Secure software leasing. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 501–530. Springer, Cham, October 2021. (Cited on page 4, 7, 12.)

[ALL+21] Scott Aaronson, Jiahui Liu, Qipeng Liu, Mark Zhandry, and Ruizhe Zhang. New approaches for quantum copy-protection. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 526–555, Virtual Event, August 2021. Springer, Cham. (Cited on page 13, 14.)

[BBL24] Estuardo Alpirez Bock, Chris Brzuska, and Russell W. F. Lai. Simple watermarking pseudorandom functions from extractable pseudorandom generators. *IACR Communications in Cryptology*, 1(2), 2024. (Cited on page 3, 25.)

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *Journal of the ACM*, 59(2):6:1–6:48, 2012. (Cited on page 3.)

[BKS21] Nir Bitansky, Michael Kellner, and Omri Shmueli. Post-quantum resettably-sound zero knowledge. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 62–89. Springer, Cham, November 2021. (Cited on page 12.)

[BLW17] Dan Boneh, Kevin Lewi, and David J. Wu. Constraining pseudorandom functions privately. In Serge Fehr, editor, *PKC 2017, Part II*, volume 10175 of *LNCS*, pages 494–524. Springer, Berlin, Heidelberg, March 2017. (Cited on page 3, 25.)

[Bou05] Jean Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1:1–32, 2005. (Cited on page 51.)

[BP15] Nir Bitansky and Omer Paneth. On non-black-box simulation and the impossibility of approximate obfuscation. *SIAM Journal on Computing*, 44(5):1325–1383, 2015. (Cited on page 12.)

[Bra18] Zvika Brakerski. Quantum FHE (almost) as secure as classical. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 67–95. Springer, Cham, August 2018. (Cited on page 5, 11, 20.)

[BZ13] Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 361–379. Springer, Berlin, Heidelberg, August 2013. (Cited on page 6, 11.)

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019. (Cited on page 19.)

[CHN+18] Aloni Cohen, Justin Holmgren, Ryo Nishimaki, Vinod Vaikuntanathan, and Daniel Wichs. Watermarking cryptographic capabilities. *SIAM Journal on Computing*, 47(6):2157–2202, 2018. (Cited on page 3, 25.)

[CLLZ21] Andrea Coladangelo, Jiahui Liu, Qipeng Liu, and Mark Zhandry. Hidden cosets and applications to unclonable cryptography. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 556–584, Virtual Event, August 2021. Springer, Cham. (Cited on page 14.)

[CLW18] Ran Canetti, Alex Lombardi, and Daniel Wichs. Fiat-Shamir: From practice to theory, part II (NIZK and correlation intractability from circular-secure FHE). Cryptology ePrint Archive, Report 2018/1248, 2018. (Cited on page 19.)

[Cv91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, Berlin, Heidelberg, April 1991. (Cited on page 4.)

[DN21] Nico Döttling and Ryo Nishimaki. Universal proxy re-encryption. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 512–542. Springer, Cham, May 2021. (Cited on page 4.)

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008. (Cited on page 48, 49.)

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM Journal on Computing*, 29(1):1–28, 1999. (Cited on page 19.)

[FR21] Marc Fischlin and Felix Rohrbach. Single-to-multi-theorem transformations for non-interactive statistical zero-knowledge. In Juan Garay, editor, *PKC 2021, Part II*, volume 12711 of *LNCS*, pages 205–234. Springer, Cham, May 2021. (Cited on page 18, 19.)

[GKM+19] Rishab Goyal, Sam Kim, Nathan Manohar, Brent Waters, and David J. Wu. Watermarking public-key cryptographic primitives. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 367–398. Springer, Cham, August 2019. (Cited on page 3, 5, 25, 26.)

[GKW17] Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In Chris Umans, editor, *58th FOCS*, pages 612–621. IEEE Computer Society Press, October 2017. (Cited on page 11, 19, 20.)

[GKWW21] Rishab Goyal, Sam Kim, Brent Waters, and David J. Wu. Beyond software watermarking: Traitor-tracing for pseudorandom functions. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 250–280. Springer, Cham, December 2021. (Cited on page 3, 24.)

[HL11] Shai Halevi and Huijia Lin. After-the-fact leakage in public-key encryption. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 107–124. Springer, Berlin, Heidelberg, March 2011. (Cited on page 10, 49, 50, 51, 52.)

[HLWW16] Carmit Hazay, Adriana López-Alt, Hoeteck Wee, and Daniel Wichs. Leakage-resilient cryptography from minimal assumptions. *Journal of Cryptology*, 29(3):514–551, July 2016. (Cited on page 49, 51, 52.)

[KN23] Fuyuki Kitagawa and Ryo Nishimaki. One-out-of-many unclonable cryptography: Definitions, constructions, and more. Cryptology ePrint Archive, Report 2023/229, 2023. (Cited on page 16.)

[KN24] Fuyuki Kitagawa and Ryo Nishimaki. Watermarking prfs and PKE against quantum adversaries. *J. Cryptol.*, 37(3):22, 2024. (Cited on page 5, 11, 16, 24, 25.)

[KW19] Sam Kim and David J. Wu. Watermarking PRFs from lattices: Stronger security via extractable PRFs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 335–366. Springer, Cham, August 2019. (Cited on page 3, 25.)

[KW21] Sam Kim and David J. Wu. Watermarking cryptographic functionalities from standard lattice assumptions. *Journal of Cryptology*, 34(3):28, July 2021. (Cited on page 3, 25.)

[LLQZ22] Jiahui Liu, Qipeng Liu, Luowen Qian, and Mark Zhandry. Collusion resistant copy-protection for watermarkable functionalities. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 294–323. Springer, Cham, November 2022. (Cited on page 4.)

[Mah18] Urmila Mahadev. Classical homomorphic encryption for quantum circuits. In Mikkel Thorup, editor, *59th FOCS*, pages 332–338. IEEE Computer Society Press, October 2018. (Cited on page 5, 11, 20.)

[Nis20] Ryo Nishimaki. Equipping public-key cryptographic primitives with watermarking (or: A hole is to watermark). In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 179–209. Springer, Cham, November 2020. (Cited on page 3, 25.)

[NWZ16] Ryo Nishimaki, Daniel Wichs, and Mark Zhandry. Anonymous traitor tracing: How to embed arbitrary information in a key. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 388–419. Springer, Berlin, Heidelberg, May 2016. (Cited on page 3.)

[Pas13] Rafael Pass. Unprovable security of perfect NIZK and non-interactive non-malleable commitments. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 334–354. Springer, Berlin, Heidelberg, March 2013. (Cited on page 19.)

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Cham, August 2019. (Cited on page 18, 19.)

[PW11] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM Journal on Computing*, 40(6):1803–1844, 2011. (Cited on page 16.)

[QWZ18] Willy Quach, Daniel Wichs, and Giorgos Zirdelis. Watermarking PRFs under standard assumptions: Public marking and security with extraction queries. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 669–698. Springer, Cham, November 2018. (Cited on page 3, 25.)

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):34:1–34:40, 2009. (Cited on page 17.)

[Win99] A. Winter. Coding theorem and strong converse for quantum channels. *IEEE Transactions on Information Theory*, 45(7):2481–2485, 1999. (Cited on page 9, 13.)

[WZ17] Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under LWE. In Chris Umans, editor, *58th FOCS*, pages 600–611. IEEE Computer Society Press, October 2017. (Cited on page 11, 19, 20.)

[YAL+19] Rupeng Yang, Man Ho Au, Junzuo Lai, Qiuliang Xu, and Zuoxia Yu. Collusion resistant watermarking schemes for cryptographic functionalities. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 371–398. Springer, Cham, December 2019. (Cited on page 3, 25.)

[YAYX20] Rupeng Yang, Man Ho Au, Zuoxia Yu, and Qiuliang Xu. Collusion resistant watermarkable PRFs from standard assumptions. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 590–620. Springer, Cham, August 2020. (Cited on page 3.)

[YYAS22] Rupeng Yang, Zuoxia Yu, Man Ho Au, and Willy Susilo. Public-key watermarking schemes for pseudorandom functions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 637–667. Springer, Cham, August 2022. (Cited on page 3, 11, 12.)

[Zha12] Mark Zhandry. How to construct quantum random functions. In *53rd FOCS*, pages 679–687. IEEE Computer Society Press, October 2012. (Cited on page 15.)

[Zha20] Mark Zhandry. Schrödinger's pirate: How to trace a quantum decoder. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part III*, volume 12552 of *LNCS*, pages 61–91. Springer, Cham, November 2020. (Cited on page 7, 9, 13, 14.)

[Zha21] Mark Zhandry. White box traitor tracing. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 303–333, Virtual Event, August 2021. Springer, Cham. (Cited on page 3, 11.)

[Zha22] Mark Zhandry. New constructions of collapsing hashes. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 596–624. Springer, Cham, August 2022. (Cited on page 11.)

A FE with Ciphertext Uniformity for OT Functionality

We construct a FE scheme with ciphertext uniformity for the 1-out-of-2 oblivious transfer (OT) functionality

$$F[\beta](i, x_0, x_1) = x_{\beta[i]}.$$

Building blocks.

- PKE $\mathsf{PKE} = \mathsf{PKE}.(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ with ciphertext pseudorandomness and ciphertext uniformity.

Construction.

$\mathsf{Setup}(1^\lambda)$:

- Generate $(\mathsf{pk}_{j,b}, \mathsf{sk}_{j,b}) \leftarrow \mathsf{PKE}.\mathsf{KG}(1^\lambda)$ for every $j \in [n]$ and $b \in \{0,1\}$.
- Output $\mathsf{pk} = (\mathsf{pk}_{j,b})_{j,b}$ and $\mathsf{msk} = (\mathsf{sk}_{j,b})_{j,b}$.

$\mathsf{Enc}(\mathsf{pk}, (i, x_0, x_1))$:

- Parse $(\mathsf{pk}_{j,b})_{j,b} \leftarrow \mathsf{pk}$.
- Generate $s_j \leftarrow \{0,1\}^\lambda$ for every $j \in [n] \setminus \{i\}$ and compute $\mathsf{pke.ct}_{j,b} \leftarrow \mathsf{PKE}.\mathsf{Enc}(\mathsf{pk}_{j,b}, s_j)$ for every $j \in [n] \setminus \{i\}$ and $b \in \{0,1\}$.
- Set $s_{i,b} := x_b \oplus \bigoplus_{j \in [n] \setminus \{i\}} s_j$ and compute $\mathsf{pke.ct}_{i,b} \leftarrow \mathsf{PKE}.\mathsf{Enc}(\mathsf{pk}_{i,b}, s_{i,b})$ for every $b \in \{0,1\}$.
- Return $\mathsf{ct} := (\mathsf{pke.ct}_{j,b})_{j,b}$.

$\mathsf{KG}(\mathsf{msk}, \beta)$:

- Parse $(\mathsf{sk}_{j,b})_{j,b}) \leftarrow \mathsf{msk}$.
- Return $\mathsf{fsk} := (\beta, (\mathsf{sk}_{j,\beta[j]})_j)$.

$\mathsf{Dec}(\mathsf{fsk}, \mathsf{ct})$:

- Parse $(\beta, (\mathsf{sk}_i)_i) \leftarrow \mathsf{fsk}$ and $(\mathsf{pke.ct}_{j,b})_{j,b} \leftarrow \mathsf{ct}$.
- Compute $s_j \leftarrow \mathsf{PKE.Dec}(\mathsf{sk}_j, \mathsf{ct}_{j,\beta[j]})$ for every $j \in [n]$.
- Output $\bigoplus_{j\in[n]} s_j$.

$\mathsf{SimEnc}(\mathsf{pk}, \beta, y)$:

- Parse $(\mathsf{pk}_{j,b})_{j,b} \leftarrow \mathsf{pk}$.
- Generate $s_j \leftarrow \{0,1\}^\lambda$ for every $j \in [n] \setminus \{i\}$ and $s_i := y \oplus \bigoplus_{j\in[n]\setminus\{i\}} s_j$.
- For every $j \in [n]$, compute $\mathsf{pke.ct}_{j,\beta[j]} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_{j,\beta[j]}, s_j)$.
- For every $j \in [n]$, compute $\mathsf{pke.ct}_{j,1-\beta[j]} \leftarrow \{0,1\}^\ell$.
- Return $\mathsf{ct} := (\mathsf{pke.ct}_{j,b})_{j,b}$.

Correctness. By the definition of Enc and the correctness of PKE, we have $\mathsf{PKE.Dec}(\mathsf{sk}_{i,\beta[i],\mathsf{ct}_{i,\beta[i]}}) = s_{i,\beta[i]}$ for $j = i$ and $\mathsf{PKE.Dec}(\mathsf{sk}_{j,b}, \mathsf{ct}_{j,b}) = s_j$ for $j \in [n] \setminus \{i\}$. Hence, $\mathsf{Dec}(\mathsf{fsk}, \mathsf{ct})$ outputs $x_{\beta[i]}$ since Enc sets $s_{i,b} = x_b \bigoplus_{j\in[n]\setminus\{i\}} s_j$.

1-bounded simulation security. 1-bounded simulation security follows from the fact that given $\mathsf{fsk} = (\beta, (\mathsf{sk}_{j,\beta[j]})_j)$ and $\mathsf{ct} := (\mathsf{pke.ct}_{j,b})_{j,b}$, any adversary cannot distinguish $\mathsf{pke.ct}_{j,1-\beta[j]}$ from a uniformly random string for every $j \in [n]$ due to the ciphertext pseudorandomness of PKE.

Ciphertext Uniformity. Suppose we run $\mathsf{SimEnc}(\mathsf{pk}, \beta, y)$ with uniformly random y. Then, from the ciphertext uniformity of PKE, $\mathsf{pke.ct}_{j,\beta[j]} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_{j,\beta[j]}, s_j)$ distributes uniformly at random for every $j \in [n]$ even given $\{\mathsf{sk}_{j,\beta[j]}\}_j$. This completes the proof since $\{\mathsf{pke.ct}_{j,1-\beta[j]}\}_j$ are uniformly random strings in $\mathsf{SimEnc}(\mathsf{pk}, \beta, y)$.

B Construction of After-the-Fact Leakage-Resilient Quantum Unobfuscatable Point Function

We show how to realize unobfuscatable point function with after-the-fact leakage resilience. We present how to construct after-the-fact leakage-resilient SKE from any standard PKE in Appendix B.2, which is a crucial building block. Then, we present how to construct after-the-fact leakage-resilient quantum unobfuscatable point function in Appendix B.3.

B.1 Preliminaries

We use several building blocks to achieve after-the-fact leakage-resilient unobfuscatable functions. We introduce the definitions of them in this subsection.

Leakage-resilient encryption.

Definition B.1 (Average Min-Entropy [DORS08]). *The average min-entropy is defined as follows.*

$$\begin{aligned}\widetilde{H}_\infty(A \mid B) &:= -\log \mathop{\mathbb{E}}_{y\leftarrow B}[\max_x \Pr[A = x \mid B = y]] \\ &= -\log \mathop{\mathbb{E}}_{y\leftarrow B}[2^{-H_\infty(A|B=y)}].\end{aligned}$$

Definition B.2 ((k, ϵ)**-extractor** [DORS08]). *A function* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ *is an average-case* (k, ϵ)*-extractor if for all pairs of random variables* A *and* B *such that* $A \in \{0,1\}^n$ *and* $\widetilde{H}_\infty(A \mid B) \geq k$*, it holds that*

$$\mathsf{SD}((\mathsf{Ext}(A, S), S, B), (\mathcal{U}_m, S, B)) \leq \epsilon,$$

where S *is uniform over* $\{0,1\}^r$*.*

Definition B.3 (Weak Hash Proof System [HLWW16]). *A weak hash proof system (wHPS) with output space* $\mathcal{K}$ *is a tuple of four PPT algorithms* $(\mathsf{Gen}, \mathsf{Encap}, \mathsf{Encap}^*, \mathsf{Decap})$*.*

$\mathsf{Gen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$**:** *The key generation algorithm takes a security parameter* 1^λ *and outputs a public key* pk *and a secret key* sk.

$\mathsf{Encap}(\mathsf{pk}) \to (\mathsf{ct}, \mathsf{key})$**:** *The key encapsulation algorithm takes a public key* pk *and outputs a valid ciphertext* ct *encapsulating* $\mathsf{key} \in \mathcal{K}$*.*

$\mathsf{Encap}^*(\mathsf{pk}) \to \mathsf{ct}^*$**:** *The invalid key encapsulation algorithm takes a public key* pk *and outputs an invalid ciphertext* ct^*.

$\mathsf{Decap}(\mathsf{sk}, \mathsf{ct}) \to \mathsf{key}'$**:** *The decapsulation algorithm takes a secret key* sk *and a ciphertext* ct *and outputs a key* $\mathsf{key}' \in \mathcal{K}$*.*

We require a wHPS to satisfy the followings.

Correctness: *For all* $(\mathsf{pk}, \mathsf{sk})$ *in the range of* $\mathsf{Gen}(1^\lambda)$*,*

$$\Pr[\mathsf{Decap}(\mathsf{ct}, \mathsf{sk}) = \mathsf{key} \mid (\mathsf{ct}, \mathsf{key}) \leftarrow \mathsf{Encap}(\mathsf{pk})] = 1 - \mathsf{negl}(\lambda).$$

Ciphertext Indistinguishability: *If we generate* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*,* $(\mathsf{ct}, \mathsf{key}) \leftarrow \mathsf{Encap}(\mathsf{pk})$*, and* $\mathsf{ct}^* \leftarrow \mathsf{Encap}^*(\mathsf{pk})$*, it holds that*

$$(\mathsf{pk}, \mathsf{sk}, \mathsf{ct}) \overset{\mathsf{c}}{\approx} (\mathsf{pk}, \mathsf{sk}, \mathsf{ct}^*).$$

Smoothness: *If we generate* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*,* $\mathsf{ct}^* \leftarrow \mathsf{Encap}^*(\mathsf{pk})$*,* $\mathsf{key} \leftarrow \mathcal{K}$*, and set* $\mathsf{key}^* \leftarrow \mathsf{Decap}(\mathsf{sk}, \mathsf{ct}^*)$*, it holds that*

$$(\mathsf{pk}, \mathsf{ct}^*, \mathsf{key}^*) \overset{\mathsf{p}}{\approx} (\mathsf{pk}, \mathsf{ct}^*, \mathsf{key}).$$

That is, $\mathsf{key}^* \leftarrow \mathsf{Decap}(\mathsf{sk}, \mathsf{ct}^*)$ *is uniformly random over* $\mathcal{K}$*.*

Theorem B.4 ([HLWW16]). *Assume the existence of IND-CPA secure PKE. Then, for any arbitrarily large polynomial* $\ell = \ell(\lambda)$*, there exists a wHPS with output space* $\mathcal{K} = \{0,1\}^{\ell(\lambda)}$*.*

We introduce entropic security against after-the-fact leakage attacks by Halevi and Lin [HL11].

Definition B.5 (($k, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}}$)**-Entropic Leakage-Resilient Encryption [HL11]).** *Let* $\Sigma = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme. We introduce the following real game to define the view* $\mathsf{View}_{\mathcal{A}}^{\mathsf{Real}}(\Sigma)$ *as follows.*

1. *The parameters* $(k, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$ *are given. The challenger chooses a random message* $m \leftarrow \mathcal{U}_k$*, generates* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$*, and returns* pk *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *sends a pre-challenge leakage query* h_{pre}*. If the output length of* h_{pre} *is at most* ℓ_{pre}*, the challenger returns* $h_{\mathsf{pre}}(\mathsf{sk})$*. Else if, the challenger returns nothing.*
3. *If* $\mathcal{A}$ *sends a challenge query, the challenger returns* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, m)$ *to* $\mathcal{A}$*.*
4. $\mathcal{A}$ *sends a post-challenge leakage query* h_{post}*. If the output length of* h_{post} *is at most* ℓ_{post}*, the challenger returns* $h_{\mathsf{post}}(\mathsf{sk})$*. Else if, the challenger returns nothing.*

Let $(\mathsf{rand}, \mathsf{pk}^+, h_{\mathsf{pre}}(\mathsf{sk}), \mathsf{ct}, h_{\mathsf{post}}(\mathsf{sk}))$ *be the random variable describing the view of the adversary* $\mathcal{A}$ *in the game above, where* rand *is the randomness by* $\mathcal{A}$ *and* pk^+ *is* pk *and all the ciphertexts given by the encryption queries. We denote the message given at the beginning of the game by* $\mathsf{m}^{\mathsf{Real}}$.

The simulated game is the same as the real one except that Sim *is given a uniformly chosen message* $\mathsf{m}^{\mathsf{Sim}}$ *as input and interacts with* $\mathcal{A}$ *instead of the challenger. We denote the view of* $\mathcal{A}$ *interacting with* Sim *by* $\mathsf{View}_{\mathcal{A}}^{\mathsf{Sim}}(\mathsf{Sim})$.

Let δ *be another slackness parameter. We say that* Σ *is* $(k, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$*-entropic leakage-resilient if there exists a simulator* Sim *such that, for any QPT* $\mathcal{A}$*, we have the following two conditions.*

- *Indistinguishability:*

$$(\mathsf{m}^{\mathsf{Real}}, \mathsf{View}_{\mathcal{A}}^{\mathsf{Real}}(\Sigma)) \stackrel{c}{\approx} (\mathsf{m}^{\mathsf{Sim}}, \mathsf{View}_{\mathcal{A}}^{\mathsf{Sim}}(\mathsf{Sim}))$$

- *Average min-entropy of* $\mathsf{m}^{\mathsf{Sim}}$ *given* $\mathsf{View}_{\mathcal{A}}^{\mathsf{Sim}}(\mathsf{Sim})$*:*

$$\widetilde{H}_{\infty}(\mathsf{m}^{\mathsf{Sim}} \mid \mathsf{View}_{\mathcal{A}}^{\mathsf{Sim}}(\mathsf{Sim})) \geq k - \ell_{\mathsf{post}} - \delta$$

We introduce 2-split-state PKE by Halevi and Lin [HL11].

Definition B.6 (2-split-state PKE [HL11]). *A* 2*-split-state encryption is a public-key encryption scheme* $\Sigma = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *that has the following structure.*

- *The secret key consists of a pair of strings* $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$*, and the public key also consists of a pair* $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$.
- *The key generation algorithm* KG *consists of two subroutines* KG_1 *and* KG_2*, where* KG_i *outputs* $(\mathsf{pk}_i, \mathsf{sk}_i)$ *for* $i \in \{1, 2\}$.
- *The decryption algorithm* Dec *also consists of two partial decryption subroutines* Dec_1 *and* Dec_2 *and a combining subroutine* $\mathsf{CombDec}$*. Each* Dec_i *takes as input the ciphertext and* sk_i *and outputs partial decryption* p_i*. The combining subroutine* $\mathsf{CombDec}$ *takes the ciphertext and the pair* (p_1, p_2) *and recovers the plaintext.*

Definition B.7 (After-the-Fact $(\ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$**-Leakage-Resilience in the Split-State Model).** *We define the following experiment* $\mathsf{Exp}_{\Sigma,\mathcal{A}}^{\mathsf{lr\text{-}split}}(1^{\lambda}, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}}, \mathsf{coin})$ *as follows.*

1. *The challenger chooses uniformly random* $r_1, r_2 \in \{0,1\}^*$*, generates* $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KG}(1^{\lambda}; r_i)$ *for* $i = 1, 2$*, and passes* $(\mathsf{pk}_1, \mathsf{pk}_2)$ *to* $\mathcal{A}$.
2. $\mathcal{A}$ *can send an arbitrary number of leakage queries* $(h_{1,i}^{\mathsf{pre}}, h_{2,i}^{\mathsf{pre}})$ *adaptively. The challenger returns* $(h_{i,1}^{\mathsf{pre}}(\mathsf{sk}_1), h_{i,2}^{\mathsf{pre}}(\mathsf{sk}_2))$ *for the i-th query if the total output length of all the pre-challenge queries so far does not exceed* ℓ_{pre} *in each coordinate. Else if the challenger returns nothing.*
3. $\mathcal{A}$ *sends a pair* $(m_0, m_1) \in \{0,1\}^u$*. The challenger returns* $\mathsf{ct}_{\mathsf{coin}} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_{\mathsf{coin}})$.
4. $\mathcal{A}$ *can send an arbitrary number of leakage queries* $(h_{1,i}^{\mathsf{post}}, h_{2,i}^{\mathsf{post}})$ *adaptively. The challenger returns* $(h_{i,1}^{\mathsf{post}}(\mathsf{sk}_1), h_{i,2}^{\mathsf{post}}(\mathsf{sk}_2))$ *for the i-th query if the total output length of all the post-challenge queries so far does not exceed* ℓ_{post} *in each coordinate. Else if the challenger returns nothing.*
5. $\mathcal{A}$ *outputs* $\mathsf{coin}' \in \{0,1\}$*. The challenger outputs* 1 *if* $\mathsf{coin} = \mathsf{coin}'$*, otherwise* 0.

We say that a 2*-split-state encryption scheme* Σ *is* $(\ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$*-leakage-resilient in the split state model if for any QPT* $\mathcal{A}$*, we have*

$$\mathsf{Adv}_{\Sigma,\mathcal{A}}^{\mathsf{lr\text{-}split}}(\lambda, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}}) := \left|\Pr\left[\mathsf{Exp}_{\Sigma,\mathcal{A}}^{\mathsf{lr\text{-}split}}(1^{\lambda}, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}}, 0) = 1\right] - \Pr\left[\mathsf{Exp}_{\Sigma,\mathcal{A}}^{\mathsf{lr\text{-}split}}(1^{\lambda}, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}}, 1) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Theorem B.8 ([HL11]). *If there exist a $(t, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$-entropic leakage-resilient PKE scheme, there exists a 2-split-state PKE scheme that is $(\ell'_{\mathsf{pre}}, \ell'_{\mathsf{post}})$-leakage-resilient in the split-state model such that*

$$\ell'_{\mathsf{pre}} \leq \ell_{\mathsf{pre}} \textit{ and } \ell'_{\mathsf{post}} \leq \min(\ell_{\mathsf{post}} - u, t - v - 1).$$

Halevi and Lin use the two source extractor by Bourgain [Bou05] to achieve the average-case (v, ϵ)-two-source extractor[16] with $\epsilon = 2^{-u-\omega(\log \lambda)}$ and $v = \gamma t$ such that $\gamma < 1/2$, which is a building block for the theorem above. However, we do not need any computational assumption for two source extractors.

B.2 Post-Quantum Secure After-the-Fact Leakage-Resilient SKE

In this subsection, we present how to construct 2-split-state SKE/PKE that is after-the-fact leakage-resilient in the split-state model from any PKE. Although Halevi and Lin showed after-the-fact leakage-resilient PKE, they use hash proof systems [HL11]. Currently, we do not know how to instantiate hash proof system with post-quantum assumptions such as the LWE assumption.

Construction. We need entropic leakage-resilient PKE (Definition B.5) to achieve after-the-fact leakage-resilient PKE (Definition B.7). We construct a $(k, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$-entropic leakage-resilient PKE scheme from weak hash proof system. Our construction is essentially the same as that by Halevi and Lin [HL11] or by Hazay et al. [HLWW16].[17]

Ingredients.

- A wHPS $\mathsf{wHPS} = \mathsf{wHPS}.(\mathsf{Gen}, \mathsf{Encap}, \mathsf{Encap}^*, \mathsf{Decap})$ with output space $\mathcal{K} := \{0,1\}^{t_1}$.
- Average-case (t_4, δ)-strong extractor $\mathsf{Ext} : \{0,1\}^{t_1} \times \{0,1\}^{t_2} \to \{0,1\}^{t_3}$.

Our entropic leakage-resilient PKE scheme $\Sigma_{\mathsf{entrp}} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is as follows.

$\mathsf{KG}(1^\lambda)$**:**

- Generate $(\mathsf{whps.pk}, \mathsf{whps.sk}) \leftarrow \mathsf{wHPS.Gen}(1^\lambda)$.
- Output $(\mathsf{pk}, \mathsf{sk}) := (\mathsf{whps.pk.whps.sk})$.

$\mathsf{Enc}(\mathsf{pk}, m)$**:**

- Parse $\mathsf{pk} = \mathsf{whps.pk}$.
- Generate $(\mathsf{whps.ct}, \mathsf{whps.key}) \leftarrow \mathsf{wHPS.Encap}(\mathsf{whps.pk})$.
- Sample a random seed $s \in \{0,1\}^{t_2}$ and compute $\psi := \mathsf{Ext}(\mathsf{whps.key}, s) \oplus m$.
- Output $\mathsf{ct} := (\mathsf{whps.ct}, s, \psi)$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:**

- Parse $\mathsf{sk} = \mathsf{whps.sk}$ and $\mathsf{ct} = (\mathsf{whps.ct}, s, \psi)$.
- Compute $\mathsf{whps.key}' \leftarrow \mathsf{wHPS.Decap}(\mathsf{whps.sk}, \mathsf{whps.ct})$.
- Output $m' := \psi \oplus \mathsf{Ext}(\mathsf{whps.key}', s)$.

Theorem B.9. *The PKE scheme Σ_{entrp} is $(k, \ell_{\mathsf{pre}}, \ell_{\mathsf{post}})$-entropic leakage-resilient for δ' as long as these parameters satisfy the following conditions.*

$$\ell_{\mathsf{pre}} \leq \log |\mathcal{K}| - t_4 \textit{ and } \delta' \leq t_3 - \log \frac{1}{2^{-t_3} + \delta}.$$

[16] We omit the definitions of (average-case) two source extractors since they are not essential here.

[17] Halevi and Lin constructed entropic leakage-resilient PKE from hash proof system. Hazay et al. constructed leakage-resilient PKE from wHPS.

The proof is almost the same as that by Halevi and Lin [HL11]. However, we write the proof for confirmation since we use weak hash proof systems instead of hash proof systems.

Proof of Theorem B.9. Our simulator Sim is as follows. It works almost identically to the challenger in the real game except that it generates an invalid ciphertext for the challenge.

- It is given $\mathsf{m}^{\mathsf{Sim}}$, generates $(\mathsf{whps.pk}, \mathsf{whps.sk}) \leftarrow \mathsf{wHPS.Gen}(1^\lambda)$ and passes $\mathsf{whps.pk}$ to $\mathcal{A}$.
- It can answer pre/post-leakage queries $h_{\mathsf{pre}}(\cdot)$ and $h_{\mathsf{post}}(\cdot)$ since it has $\mathsf{whps.sk}$.
- It computes $\mathsf{whps.ct}^* \leftarrow \mathsf{wHPS.Encap}^*(\mathsf{whps.pk})$ for a challenge query from $\mathcal{A}$ and returns $\mathsf{ct}^* := (\mathsf{whps.ct}^*, s^*, \psi^*)$ where $s^* \leftarrow \{0,1\}^{t_2}$ and $\psi^* := \mathsf{Ext}(\mathsf{Decap}(\mathsf{whps.sk}, \mathsf{whps.ct}^*), s) \oplus \mathsf{m}^{\mathsf{Sim}}$.

We can immediately obtain the indistinguishability of Σ_{entrp} from the ciphertext indistinguishability of wHPS.

In the rest of this proof, we prove the min-entropy condition. From the smoothness of wHPS, the encapsulated key $\mathsf{whps.key}^* := \mathsf{Decap}(\mathsf{whps.sk}, \mathsf{whps.ct}^*)$ has $\log|\mathcal{K}| = t_1$ bits of min-entropy even given pk and ct^*. Hence, $(t_1 - \ell_{\mathsf{pre}}) \geq t_4$ bits of min-entropy is left in $\mathsf{whps.key}^*$ even given $\mathsf{pk} = \mathsf{whps.pk}$, ct^*, and the pre-challenge leakage. By Definition B.2, $\mathsf{Ext}(\mathsf{whps.key}^*, s)$ is δ-close to a uniform t_3-bit string even given $\mathsf{pk} = \mathsf{whps.pk}$, ct^*, the pre-challenge leakage, the seed s^*, and ψ^*. Hence, the message $\mathsf{m}^{\mathsf{Sim}}$ has at least $t_3 - \delta' \geq \log \frac{1}{2^{-t_3}+\delta}$ bits of min-entropy before the post-challenge leakage. By the post-challenge leakage, $t_3 - \delta' - \ell_{\mathsf{post}}$ bits of average min-entropy is left in $\mathsf{m}^{\mathsf{Sim}}$. This completes the proof. □

As Halevi and Lin observed, we can use an extractor with $\delta < 2^{-t_3}$, so $\delta' < 1$. Since we can see a PKE scheme as an SKE scheme, the SKE scheme derived from the result of Halevi and Lin (Theorem B.8) [HL11] and our entropic PKE scheme above satisfies the following syntax.

Definition B.10 (2-split-state SKE). *A 2-split-state encryption is a secret-key encryption scheme $\Sigma = (\mathsf{Enc}, \mathsf{Dec})$ that has the following structure.*

- *The secret key consists of a pair of uniformly random strings $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$. Hence, Σ does not have a key generation algorithm.*
- *The decryption algorithm Dec also consists of two partial decryption subroutines Dec_1 and Dec_2 and a combining subroutine $\mathsf{CombDec}$. Each Dec_i takes as input the ciphertext and sk_i and outputs partial decryption p_i. The combining subroutine $\mathsf{CombDec}$ takes the ciphertext and the pair (p_1, p_2) and recovers the plaintext.*

A key pair of wHPS by Hazay et al. [HLWW16] consists of a number of key pairs of standard PKE. In addition, randomness for key generation can be seen as a secret key in standard PKE without loss of generality (and the corresponding public key is deterministically derived from the secret key). Hence, the secret key of our entropic PKE scheme can be uniformly random strings. It is easy to see that the split-state PKE scheme by Halevi and Lin inherits this property. Thus, our 2-split-state SKE scheme has the syntax above. In the secret-key variant of Definition B.7, the adversary can access the encryption oracle $\mathsf{Enc}(\mathsf{sk}, \cdot)$ that returns a ciphertext $\mathsf{Enc}(\mathsf{sk}, m)$ for a query m through the game.

From Theorems B.4, B.8 and B.9, we obtain a 2-split-state SKE scheme that is leakage-resilient in the split state model from any PKE scheme by setting appropriate parameters.

Corollary B.11. *Assume the existence of IND-CPA secure PKE. Then, there exists 2-split-state SKE that is after-the-fact $(0,2)$-leakage-resilient in the split-state model.*

In the construction derived from entropic PKE, the leakage ration to the secret key size is bad. However, we need only 2-bit after-the-fact leakage resilience for our purpose in Section 5. We can obtain it by using a slightly long seed length for the two extractor in the construction by Halevi and Lin (say, the seed length of the two source extractor is larger than $2|\mathsf{lock}| + 6$ where lock is the lock value of lockable obfuscation described in Appendix B.3).

B.3 Construction of After-the-fact Leakage-Resilient Unobfuscatable Point Function

We construct ℓ-after-the-fact leakage-resilient unobfuscatable point function.

Ingredients.

- 2-split-state SKE scheme $\mathsf{2SKE} = \mathsf{2SKE}.(\mathsf{Enc}, \mathsf{Dec})$.
- Lockable obfuscation $\Sigma_{\mathsf{LO}} = (\mathsf{LObf}, \mathsf{Eval})$ with simulator Sim.
- QFHE scheme $\mathsf{QFHE} = \mathsf{QFHE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$.

Scheme description. Our unobfuscatable point function UOPF for the secret message space $\mathcal{SS}$, input space $\{0,1\}^{\ell_{\mathsf{in}}}$, and output space $\{0,1\}^{2\ell_{\mathsf{out}}}$ is as follows.

$\mathsf{UOPF}.\mathsf{Gen}(1^\lambda, \mu)$**:**

- Generate $\alpha \leftarrow \{0,1\}^{\ell_{\mathsf{in}}}$ and $\beta_1, \beta_2 \leftarrow \{0,1\}^{\ell_{\mathsf{out}}}$ and set $\beta = \beta_1 \| \beta_2$.
- Generate $\mathsf{lock} \leftarrow \{0,1\}^\lambda$.
- Generate $(\mathsf{qfhe.pk}, \mathsf{qfhe.sk}) \leftarrow \mathsf{QFHE}.\mathsf{Gen}(1^\lambda)$.
- Generate $\mathsf{qfhe.ct} \leftarrow \mathsf{QFHE}.\mathsf{Enc}(\mathsf{qfhe.pk}, \alpha)$.
- Generate $\mathsf{ske.ct} \leftarrow \mathsf{2SKE}.\mathsf{Enc}(\beta, \mathsf{lock})$.
- Generate $\widetilde{P} \leftarrow \mathsf{LObf}(1^\lambda, \mathsf{QFHE}.\mathsf{Dec}(\mathsf{qfhe.sk}, \cdot), \mathsf{lock}, \mu)$.
- Output $f_{\alpha,\beta}$ and $\mathsf{aux} = (\mathsf{qfhe.pk}, \mathsf{qfhe.ct}, \mathsf{ske.ct}, \widetilde{P})$.

$\mathsf{UOPF}.\mathsf{Extract}(\mathcal{C}, \mathsf{aux})$**:**

- Parse $\mathsf{aux} = (\mathsf{qfhe.pk}, \mathsf{qfhe.ct}, \mathsf{ske.ct}, \widetilde{P})$ and $\mathcal{C} = (q, \boldsymbol{U})$.
- Construct $\boldsymbol{V}$ that is a compact description of $\{\boldsymbol{V}_x\}_x$, where $\boldsymbol{V}_x$ is a unitary that performs the following computations coherently when applied to a quantum state q.
  1. Apply $\boldsymbol{U}_x$ to q, measure the first register, and obtain the result β'.
  2. Output $\mathsf{lock}' \leftarrow \mathsf{2SKE}.\mathsf{Dec}(\beta', \mathsf{ske.ct})$.
- Construct a quantum program with classical input and output $\mathcal{Q}[\mathcal{C}, \mathsf{ske.ct}] = (q, \boldsymbol{V})$.
- Compute $\mathsf{qfhe.ct}' \leftarrow \mathsf{qfhe}.\mathsf{Eval}(\mathsf{qfhe.pk}, \mathcal{Q}[\mathcal{C}, \mathsf{ske.ct}], \mathsf{qfhe.ct})$.
- Output $\mu' \leftarrow \mathsf{LO}.\mathsf{Eval}(\widetilde{P}, \mathsf{qfhe.ct}')$.

Theorem B.12. *If* $\mathsf{2SKE}$ *is* $(0, \ell_{\mathsf{post}})$*-leakage-resilient in the split state model,* Σ_{LO} *is simulation secure, and* QFHE *is IND-CPA secure,* UOPF *above is an* ℓ_{post}*-after-the-fact leakage-resilient quantum unobfuscatable point function.*

Proof. We prove the three requirements (after-the-fact leakage-resilient indistinguishability of points implies indistinguishability of points).

Correctness: Let $\mu \in \mathcal{SS}$ and $(f_{\alpha,\beta}, \mathsf{aux}) \leftarrow \mathsf{UOPF}.\mathsf{Gen}(1^\lambda, \mu)$. For any quantum circuit with classical input and ouput $\mathcal{C}$, the output of $\mathsf{UOPF}.\mathsf{Extract}(\mathcal{C}, \mathsf{aux})$ is μ or $\perp$ due to the design of $\mathsf{UOPF}.\mathsf{Extract}$ and the evaluation correctness of Σ_{LO}. Next, let $\mathcal{C}$ be a quantum circuit with classical input and output that maps α to β with overwhelming probability. Then, when we execute $\mathsf{UOPF}.\mathsf{Extract}(\mathcal{C}, \mathsf{aux})$, $\mathsf{qfhe.ct}'$ should be a ciphertext of lock with overwhelming probability. Thus, we have $\Pr[\mathsf{UOPF}.\mathsf{Extract}(\mathcal{C}, \mathsf{aux}) = \mu] = 1 - \mathsf{negl}(\lambda)$ from the correctness of Σ_{LO}.

Indistinguishability of messages: We can prove the indistinguishability of messages based on the security of $\mathsf{2SKE}$ and Σ_{LO}. Concretely, we can argue that the information of lock is hidden from the security of $\mathsf{2SKE}$. Note that β is never used except as the secret key of $\mathsf{2SKE}$. Then, the indistinguishability of messages of UOPF follows from the simulation security of Σ_{LO} since lock was erased from $\mathsf{ske.ct}$ in the previous step. Note that in this proof, $\mathsf{2SKE}$ need to satisfy only standard (one-time) indistinguishability.

ℓ_{post}**-after-the-fact leakage-resilient indistinguishability of points:** We can prove that if 2SKE is a 2-split-state SKE scheme that is after-the-fact $(0, \ell_{\mathsf{post}})$-leakage-resilient in the split-state model, Σ_{LO} is a secure lockable obfuscation, and QFHE is a secure QFHE, then UOPF satisfies ℓ_{post}-after-the-fact leakage-resilient indistinguishability of points. The proof is similar to that for indistinguishability of messages. We first argue that the information of qfhe.sk is hidden by the after-the-fact $(0, \ell_{\mathsf{post}})$-leakage-resilience of 2SKE and the simulation security of Σ_{LO}. That is, we erase lock from ske.ct by the after-the-fact $(0, \ell_{\mathsf{post}})$-leakage-resilience of 2SKE, then we erase qfhe.sk from $\widetilde{P}$ by the simulation security of Σ_{LO}. Then, the ℓ_{post}-after-the-fact leakage-resilient indistinguishability of points of UOPF follows from the security of QFHE.

$\square$

By Theorems 2.32, 2.37 and B.12 and Corollary B.11, we complete the proof of Theorem 3.3.

C Extended Projective Property of ATI

We prove a new property of ATI to prove the unremovability of the construction in Section 7. We can see $\widetilde{\mathsf{Vrfy}}$ appeared below as a sub-step of WMSIG.Vrfy in Section 7.

Notations. The following Proposition C.1 considers a signature scheme $\mathsf{SIG} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vrfy})$ for a message space $\mathcal{MS}$ with deterministic Vrfy, a modified deterministic verification algorithm $\widetilde{\mathsf{Vrfy}}$, and a sampler $\mathcal{Sample}$ for a quantum program with classical input and output. Let $\boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}$ (resp. $\boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}}$) be the unitary that maps $|a\rangle\,|b\rangle$ to $|a\rangle\,|b \oplus \mathsf{Vrfy}(\mathsf{vk},\mathsf{m},a)\rangle$ (resp. $|a\rangle \left| b \oplus \widetilde{\mathsf{Vrfy}}(\mathsf{vk},\mathsf{m},a) \right\rangle$). For an output $(|\psi\rangle, \{\boldsymbol{U}_\mathsf{m}\}_{\mathsf{m}\in\mathcal{M}})$ of $\mathcal{Sample}$, we let $\mathcal{P} = (\boldsymbol{P}_\mathsf{m}, \boldsymbol{Q}_\mathsf{m})_\mathsf{m}$ and $\widetilde{\mathcal{P}} = (\widetilde{\boldsymbol{P}}_\mathsf{m}, \widetilde{\boldsymbol{Q}}_\mathsf{m})_\mathsf{m}$ be collections of binary outcome projective measurements, where

$$\boldsymbol{P}_\mathsf{m} = \boldsymbol{U}_\mathsf{m}^\dagger \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}}^\dagger (\boldsymbol{I} \otimes |1\rangle\langle 1|) \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}} \boldsymbol{U}_\mathsf{m}, \quad \boldsymbol{Q}_\mathsf{m} = \boldsymbol{I} - \boldsymbol{P}_\mathsf{m}$$
$$\widetilde{\boldsymbol{P}}_\mathsf{m} = \boldsymbol{U}_\mathsf{m}^\dagger \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}}^\dagger (\boldsymbol{I} \otimes |1\rangle\langle 1|) \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \boldsymbol{U}_\mathsf{m}, \quad \boldsymbol{Q}_\mathsf{m} = \boldsymbol{I} - \widetilde{\boldsymbol{P}}_\mathsf{m}.$$

Finally, we let $U_{\mathcal{MS}}$ be the uniform distribution over $\mathcal{MS}$, and we define $\mathcal{P}_{U_{\mathcal{MS}}}$ and $\widetilde{\mathcal{P}}_{U_{\mathcal{MS}}}$ as the mixture of $\mathcal{P}$ and $\widetilde{\mathcal{P}}$ with respect to $U_{\mathcal{MS}}$.

Proposition C.1. *Let* $\mathsf{SIG} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a signature scheme, where* Vrfy *is deterministic. Let* $\widetilde{\mathsf{Vrfy}}$ *and* $\mathcal{Sample}$ *be a deterministic algorithm and a QPT algorithm, respectively, with the following constraints.*

- *For any* vk, m, *and* σ, *if* $\widetilde{\mathsf{Vrfy}}(\mathsf{vk},\mathsf{m},\sigma) = 1$ *holds, then* $\mathsf{Vrfy}(\mathsf{vk},\mathsf{m},\sigma) = 1$ *also holds.*
- *Any QPT algorithm* $\mathcal{A}$ *given* vk *and the classical oracle access to* $\mathsf{Sign}(\mathsf{sk},\cdot)$ *cannot find* (m,σ) *with the following conditions with non-negligible probability, where* $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$.
  - $\mathsf{Vrfy}(\mathsf{vk},\mathsf{m},\sigma) = 1$ *and* $\widetilde{\mathsf{Vrfy}}(\mathsf{vk},\mathsf{m},\sigma) = 0$.
  - $\mathcal{A}$ *did not query query* m *to the oracle* $\mathsf{Sign}(\mathsf{sk},\cdot)$.
- $\mathrm{Tr}\left[\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},U_{\mathcal{MS}},\gamma}\,|\psi\rangle\right] = 1/\mathrm{poly}(\lambda)$ *holds, where* $(|\psi\rangle, \{\boldsymbol{U}_\mathsf{m}\}_{\mathsf{m}\in\mathcal{M}}) \leftarrow \mathcal{Sample}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$ *and* $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$.

Consider the following process.

1. *Generate* $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$ *and execute* $(|\psi\rangle, \{\boldsymbol{U}_\mathsf{m}\}_{\mathsf{m}\in\mathcal{M}}) \leftarrow \mathcal{Sample}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$.
2. *Apply* $\mathcal{ATI}^{\epsilon,\delta}_{\mathcal{P},U_{\mathcal{MS}},\gamma}$ *to* $|\psi\rangle$ *and obtain the outcome* β *and the post-measurement state* $|\psi'\rangle$.

Suppose we obtain the outcome 1 *and the post-measurement state* $|\psi'\rangle$ *in the second item. Then, we have*

$$\mathrm{Tr}\Big[\mathcal{TI}_{\gamma-3\epsilon}(\widetilde{\mathcal{P}}_{U_{\mathcal{MS}}})\,|\psi'\rangle\Big] = 1-\mathsf{negl}(\lambda).$$

Proof of Proposition C.1. Let $\{|\psi_p\rangle\}_p$ and $\{|\widetilde{\psi}_q\rangle\}_q$ be the set of orthonormal eigenvectors of $\mathcal{P}_{U_{\mathcal{MS}}}$ and $\widetilde{\mathcal{P}}_{U_{\mathcal{MS}}}$, respectively. Then, we can write $|\psi'\rangle = \sum_p a_p\cdot|\psi_p\rangle$, where $\sum_p |a_p|^2 = 1$ and $\sum_{p<\gamma-2\epsilon}|a_p|^2 = \mathsf{negl}(\lambda)$. The latter condition comes from the fact that if we apply $\mathcal{TI}_{\gamma-2\epsilon}(\mathcal{P}_{U_{\mathcal{MS}}})$ to $|\psi'\rangle$, we obtain the outcome 1 with overwhelming probability (we use the third condition in the proposition and Lemma 2.11). For simplicity, we assume we can write $|\psi'\rangle = \sum_{p\geq\gamma-2\epsilon} a_p\cdot|\psi_p\rangle$. We can also write $|\psi'\rangle = \sum_{p\geq\gamma-2\epsilon} a_p\cdot|\psi_p\rangle = \sum_{p\geq\gamma-2\epsilon} a_p\cdot\sum_q b_{p,q}|\widetilde{\psi}_q\rangle = \sum_q(\sum_{p\geq\gamma-2\epsilon} a_p\cdot b_{p,q})|\widetilde{\psi}_q\rangle$, where $\sum_q|b_{p,q}|^2 = 1$ for every p. Our goal is to prove that $\sum_{q<\gamma-3\epsilon}\left|\sum_{p\geq\gamma-2\epsilon} a_p\cdot b_{p,q}\right|^2 = \mathsf{negl}(\lambda)$.

Lemma C.2. $\Pr\Big[\Big\|\mathcal{P}_{U_{\mathcal{MS}}}|\psi'\rangle - \widetilde{\mathcal{P}}_{U_{\mathcal{MS}}}|\psi'\rangle\Big\| = \mathsf{negl}(\lambda)\Big] = 1-\mathsf{negl}(\lambda)$.

We prove Lemma C.2 later and proceed the proof using it for now. We have

$$\begin{aligned}\mathcal{P}_{U_{\mathcal{MS}}}|\psi'\rangle &= \sum_{p\geq\gamma-2\epsilon} a_p\cdot p\,|\psi_p\rangle\\ &= \sum_{p\geq\gamma-2\epsilon} a_p\cdot p\sum_q b_{p,q}|\widetilde{\psi}_q\rangle\\ &= \sum_q\left(\sum_{p\geq\gamma-2\epsilon} a_p\cdot p\cdot b_{p,q}\right)|\widetilde{\psi}_q\rangle\end{aligned}$$

and

$$\begin{aligned}\widetilde{\mathcal{P}}_{U_{\mathcal{MS}}}|\psi'\rangle &= \widetilde{\mathcal{P}}_{U_{\mathcal{MS}}}\sum_q\left(\sum_{p\geq\gamma-2\epsilon} a_p\cdot b_{p,q}\right)|\widetilde{\psi}_q\rangle\\ &= \sum_q\left(\sum_{p\geq\gamma-2\epsilon} a_p\cdot b_{p,q}\right)\cdot q\,|\widetilde{\psi}_q\rangle.\end{aligned}$$

From Lemma C.2, we have $\sum_q\left|\sum_{p\geq\gamma-2\epsilon} a_p\cdot(p-q)\cdot b_{p,q}\right|^2 = \mathsf{negl}(\lambda)$ and thus have $\sum_{q<\gamma-3\epsilon}\left|\sum_{p\geq\gamma-2\epsilon} a_p\cdot(p-q)\cdot b_{p,q}\right|^2 = \mathsf{negl}(\lambda)$ with overwhelming probability. Since we have $p-q>\epsilon$ for $p\geq\gamma-2\epsilon$ and $q<\gamma-3\epsilon$, we have $\epsilon^2\cdot\sum_{q<\gamma-3\epsilon}\left|\sum_{p\geq\gamma-2\epsilon} a_p\cdot b_{p,q}\right|^2 = \mathsf{negl}(\lambda)$. Since ϵ is inverse polynomial, we obtain $\sum_{q<\gamma-3\epsilon}\left|\sum_{p\geq\gamma-2\epsilon} a_p\cdot b_{p,q}\right|^2 = \mathsf{negl}(\lambda)$. This completes the proof assuming Lemma C.2. ☐

Proof of Lemma C.2. We finally prove Lemma C.2. For any vk and m, we define the following three sets.

$A_{\mathsf{vk},\mathsf{m}}$: The set of strings σ such that $\mathsf{Vrfy}(\mathsf{vk},\mathsf{m},\sigma)=1$.

$\widetilde{A}_{\mathsf{vk},\mathsf{m}}$: The set of strings σ such that $\widetilde{\mathsf{Vrfy}}(\mathsf{vk},\mathsf{m},\sigma)=1$.

$R_{\mathsf{vk},\mathsf{m}}$: The set of strings σ such that $\mathsf{Vrfy}(\mathsf{vk},\mathsf{m},\sigma)=0$.

From the constraints on Vrfy and $\widetilde{\mathsf{Vrfy}}$, we have $\widetilde{A}_{\mathsf{vk},\mathsf{m}}\subseteq A_{\mathsf{vk},\mathsf{m}}$. For any vk and m, we can write

$$\boldsymbol{U}_{\mathsf{m}}|\psi'\rangle = \sum_{\sigma\in A_{\mathsf{vk},\mathsf{m}}\setminus\widetilde{A}_{\mathsf{vk},\mathsf{m}}}\alpha_{\mathsf{vk},\mathsf{m},\sigma}|\sigma\rangle|\phi_{\mathsf{vk},\mathsf{m},\sigma}\rangle + \sum_{\sigma\in\widetilde{A}_{\mathsf{vk},\mathsf{m}}}\alpha_{\mathsf{vk},\mathsf{m},\sigma}|\sigma\rangle|\phi_{\mathsf{vk},\mathsf{m},\sigma}\rangle + \sum_{\sigma\in R_{\mathsf{vk},\mathsf{m}}}\alpha_{\mathsf{vk},\mathsf{m},\sigma}|\sigma\rangle|\phi_{\mathsf{vk},\mathsf{m},\sigma}\rangle.$$

We define

$$\left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle = \sum_{\sigma \in A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}} \alpha_{\mathsf{vk},\mathsf{m},\sigma} \left|\sigma\right\rangle \left|\phi_{\mathsf{vk},\mathsf{m},\sigma}\right\rangle,$$
$$\left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle = \sum_{\sigma \in \widetilde{A}_{\mathsf{vk},\mathsf{m}}} \alpha_{\mathsf{vk},\mathsf{m},\sigma} \left|\sigma\right\rangle \left|\phi_{\mathsf{vk},\mathsf{m},\sigma}\right\rangle,$$
$$\left|R_{\mathsf{vk},\mathsf{m}}\right\rangle = \sum_{\sigma \in R_{\mathsf{vk},\mathsf{m}}} \alpha_{\mathsf{vk},\mathsf{m},\sigma} \left|\sigma\right\rangle \left|\phi_{\mathsf{vk},\mathsf{m},\sigma}\right\rangle.$$

We have

$$\boldsymbol{U}^{\dagger}_{\mathsf{Vrfy},\mathsf{m}} \left|1\right\rangle \left\langle 1\right| \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}} \left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle = \left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle,$$
$$\boldsymbol{U}^{\dagger}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \left|1\right\rangle \left\langle 1\right| \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle = 0,$$
$$\boldsymbol{U}^{\dagger}_{\mathsf{Vrfy},\mathsf{m}} \left|1\right\rangle \left\langle 1\right| \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}} \left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle = \left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle,$$
$$\boldsymbol{U}^{\dagger}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \left|1\right\rangle \left\langle 1\right| \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle = \left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle,$$
$$\boldsymbol{U}^{\dagger}_{\mathsf{Vrfy},\mathsf{m}} \left|1\right\rangle \left\langle 1\right| \boldsymbol{U}_{\mathsf{Vrfy},\mathsf{m}} \left|R_{\mathsf{vk},\mathsf{m}}\right\rangle = 0,$$
$$\boldsymbol{U}^{\dagger}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \left|1\right\rangle \left\langle 1\right| \boldsymbol{U}_{\widetilde{\mathsf{Vrfy}},\mathsf{m}} \left|R_{\mathsf{vk},\mathsf{m}}\right\rangle = 0.$$

Thus, we obtain

$$\begin{aligned}\mathcal{P}_{U_{\mathcal{MS}}} \left|\psi'\right\rangle - \widetilde{\mathcal{P}}_{U_{\mathcal{MS}}} \left|\psi'\right\rangle &= \frac{1}{|\mathcal{M}|} \sum_{\mathsf{m}\in\mathcal{M}} \boldsymbol{U}^{\dagger}_{\mathsf{m}} (\left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle + \left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle) - \frac{1}{|\mathcal{M}|} \sum_{\mathsf{m}\in\mathcal{M}} \boldsymbol{U}^{\dagger}_{\mathsf{m}} \left|\widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle \\ &= \frac{1}{|\mathcal{M}|} \sum_{\mathsf{m}\in\mathcal{M}} \boldsymbol{U}^{\dagger}_{\mathsf{m}} \left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle.\end{aligned}$$

For overwhelming fraction of m, $\left\|\left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle\right\| = \mathsf{negl}(\lambda)$ on average, from the fact that it is computationally hard to find (m,σ) such that $\sigma \in A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}$ without querying m to the signing oracle $\mathsf{Sign}(\mathsf{sk},\cdot)$. Then, we have

$$\begin{aligned}\left\|\frac{1}{|\mathcal{M}|} \sum_{\mathsf{m}\in\mathcal{M}} \boldsymbol{U}^{\dagger}_{\mathsf{m}} \left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle\right\|^2 &= \frac{1}{|\mathcal{M}|^2} \sum_{\mathsf{m},\mathsf{m}'\in\mathcal{M}} \left\langle A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}} \middle| \boldsymbol{U}_{\mathsf{m}} \boldsymbol{U}^{\dagger}_{\mathsf{m}'} \middle| A_{\mathsf{vk},\mathsf{m}'} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}'}\right\rangle \\ &\leq \frac{1}{|\mathcal{M}|^2} \sum_{\mathsf{m},\mathsf{m}'\in\mathcal{M}} \left\|\boldsymbol{U}^{\dagger}_{\mathsf{m}} \left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle\right\| \cdot \left\|\boldsymbol{U}^{\dagger}_{\mathsf{m}'} \left|A_{\mathsf{vk},\mathsf{m}'} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}'}\right\rangle\right\| \\ &= \frac{1}{|\mathcal{M}|^2} \sum_{\mathsf{m},\mathsf{m}'\in\mathcal{M}} \left\|\left|A_{\mathsf{vk},\mathsf{m}} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}}\right\rangle\right\| \cdot \left\|\left|A_{\mathsf{vk},\mathsf{m}'} \setminus \widetilde{A}_{\mathsf{vk},\mathsf{m}'}\right\rangle\right\| \\ &= \mathsf{negl}(\lambda).\end{aligned}$$

We use Cauchy-Schwarz for the inequality (second line). □