Watermarking PRFs against Quantum Adversaries

Fuyuki Kitagawa[1] and Ryo Nishimaki[1]

NTT Corporation, Tokyo, Japan
{fuyuki.kitagawa.yh,ryo.nishimaki.zk}@hco.ntt.co.jp

Abstract. We initiate the study of software watermarking against quantum adversaries. A quantum adversary generates a *quantum state* as a pirate software that potentially removes an embedded message from a *classical* marked software. Extracting an embedded message from quantum pirate software is difficult since measurement could irreversibly alter the quantum state. In software watermarking against classical adversaries, a message extraction algorithm crucially uses the (input-output) behavior of a classical pirate software to extract an embedded message. Even if we instantiate existing watermarking PRFs with quantum-safe building blocks, it is not clear whether they are secure against quantum adversaries due to the quantum-specific property above. Thus, we need entirely new techniques to achieve software watermarking against quantum adversaries.

In this work, we define secure watermarking PRFs for quantum adversaries (unremovability against quantum adversaries). We also present two watermarking PRFs as follows.

- We construct a privately extractable watermarking PRF against quantum adversaries from the quantum hardness of the learning with errors (LWE) problem. The marking and extraction algorithms use a public parameter and a private extraction key, respectively. The watermarking PRF is unremovable even if adversaries have (the public parameter and) access to the extraction oracle, which returns a result of extraction for a queried quantum circuit.
- We construct a publicly extractable watermarking PRF against quantum adversaries from indistinguishability obfuscation (IO) and the quantum hardness of the LWE problem. The marking and extraction algorithms use a public parameter and a public extraction key, respectively. The watermarking PRF is unremovable even if adversaries have the extraction key (and the public parameter).

We develop a quantum extraction technique to extract information (a classical string) from a quantum state without destroying the state too much. We also introduce the notion of extraction-less watermarking PRFs as a crucial building block to achieve the results above by combining the tool with our quantum extraction technique.

1 Introduction

1.1 Background

Software watermarking is a cryptographic primitive that achieves a digital analog of watermarking. A marking algorithm of software watermarking can embed an arbitrary message (bit string) into a computer software modeled as a circuit. A marked software almost preserves the functionality of the original software. An extraction algorithm of software watermarking can extract the embedded message from a marked software. Secure software watermarking should guarantee that no adversary can remove the embedded message without significantly destroying the functionality of the original software (called unremovability).

Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, and Yang [BGI+12] initiate the study of software watermarking and present the first definition of cryptographically secure software watermarking. Hopper, Molnar, and Wagner [HMW07] also study the definition of cryptographically secure watermarking for perceptual objects. However, both works do not present a secure concrete scheme. A few works study secure constructions of watermarking for cryptographic primitives [NSS99,YF11,Nis13,Nis19], but they consider only restricted removal strategies. Cohen, Holmgren, Nishimaki, Wichs, and Vaikuntanathan [CHN+18] present stronger definitions for software watermarking and the first secure watermarking schemes for cryptographic primitives *against arbitrary removal strategies*. After the celebrated work, watermarking for cryptographic primitives have been extensively studied [BLW17,KW21,QWZ18,KW19,YAL+19,GKM+19,YAYX20,Nis20].

Primary applications of watermarking are identifying ownership of objects and tracing users that distribute illegal copies. Watermarking for cryptographic primitives also has another exciting application. Aaronson, Liu, Liu, Zhandry, and Zhang [ALL+21] and Kitagawa, Nishimaki, and Yamakawa [KNY21] concurrently and independently find that we can construct secure software leasing schemes by combining watermarking with quantum cryptography.[1] Secure software leasing [AL21] is a quantum cryptographic primitive that prevents users from generating authenticated pirated copies of leased software.[2] Since watermarking has such an exciting application in quantum cryptography and quantum computers might be an imminent threat to cryptography due to rapid progress in research on quantum computing, it is natural and fascinating to study secure software watermarking in the quantum setting.

In quantum cryptography, building blocks must be quantum-safe such as lattice-based cryptography [Reg09]. However, even if we replace building blocks of existing cryptographic primitives/protocols with quantum-safe ones, we do not necessarily obtain quantum-safe cryptographic primitives/protocols [BDF+11,ARU14]. We sometimes need new proof techniques which are different from classical ones

[1] Precisely speaking, Aaronson et al. achieve copy-detection schemes [ALL+21], which are essentially the same as secure software leasing schemes.

[2] Leased software must be a quantum state since classical bit strings can be easily copied.

due to quantum specific properties such as no-cloning and superposition access [Wat09,Zha12b,Zha12a,Unr12,Zha19,CMSZ21]. Even worse, we must consider entirely different security models in some settings. Zhandry [Zha20] studies traitor tracing [CFN94] in the quantum setting as such an example. In quantum traitor tracing, an adversary can output a *quantum state* as a pirate decoder. Zhandry shows that we need new techniques for achieving quantum traitor tracing because running a quantum pirate decoder to extract information may irreversibly alter the state due to measurement.

Zhandry [Zha20] refers to software watermarking as a cryptographic primitive that has a similar issue to quantum traitor tracing. However, his work focuses only on traitor tracing and does not study software watermarking against quantum adversaries. If we use software watermarking in the quantum setting, an adversary can output a *quantum state* as a pirate circuit where an embedded message might be removed. However, previous works consider a setting where an adversary outputs a *classical* pirate circuit. It is not clear whether watermarking schemes based on quantum-safe cryptography are secure against quantum adversaries because we need an entirely new extraction algorithm to extract an embedded message from a *quantum* pirate circuit. Thus, the main question in this study is:

Can we achieve secure watermarking for cryptographic primitives against quantum adversaries?

We affirmatively answer this question in this work.

1.2 Our Result

Our main contributions are two-fold. One is the definitional work. We define watermarking for pseudorandom functions (PRFs) against quantum adversaries, where adversaries output a quantum state as a pirate circuit that distinguishes a PRF from a random function.[3] The other one is constructing the first secure watermarking PRFs against quantum adversaries. We present two watermarking PRFs as follows.

- We construct a privately extractable watermarking PRF against quantum adversaries from the quantum hardness of the learning with errors (LWE) problem. This watermarking PRF is secure in the presence of the extraction oracle and supports public marking. That is, the marking and extraction algorithms use a public parameter and secret extraction key, respectively. The watermarking PRF is unremovable even if adversaries have access to the extraction oracle, which returns a result of extraction for a queried quantum circuit.
- We construct a publicly extractable watermarking PRF against quantum adversaries from indistinguishability obfuscation (IO) and the quantum hardness of the LWE problem. This watermarking PRF also supports public

[3] This definitional choice comes from the definition of traceable PRFs [GKWW21]. See Section 1.3 and the full version for the detail.

marking. That is, the marking and extraction algorithms use a public parameter and a public extraction key, respectively. The watermarking PRF is unremovable (we do not need to consider the mark and extraction oracles since it supports public marking and public extraction).

The former and latter PRFs satisfy weak pseudorandomness and standard (strong) pseudorandomness even against a watermarking authority, respectively.

We develop a quantum extraction algorithm to achieve the results above. Zhandry [Zha20] presents a useful technique for extracting information from quantum states without destroying them too much. However, we cannot simply apply his technique to the watermarking setting. Embedded information (arbitrary string) is chosen from an exponentially large set in the watermarking setting. On the other hand, in the traitor tracing setting, we embed a user index, which could be chosen from a polynomially large set, in a decryption key. Zhandry's technique is tailored to traitor tracing based on private linear broadcast encryption (PLBE) [BSW06] where user information is chosen from a polynomially large set with linear structure. Thus, we extend Zhandry's technique [Zha20] to extract information chosen from an exponentially large set. We also introduce the notion of extraction-less watermarking as a crucial tool to achieve watermarking against quantum adversaries. This tool is a suitable building block for our quantum extraction technique in our watermarking extraction algorithm. These are our technical contributions. See Section 1.3 for the detail.

Although this paper focuses on watermarking PRFs against quantum adversaries, it is easy to extend our definitions to watermarking public-key encryption (PKE) against quantum adversaries. In particular, our construction technique easily yields watermarking PKE (where a decryption circuit is marked) schemes. However, we do not provide the detail of watermarking PKE in this paper. We will provide them in the full version.

1.3 Technical Overview

Syntax of watermarking PRF. We first review the syntax of watermarking PRF used in this work. A watermarking PRF scheme consists of five algorithms $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}, \mathcal{E}\mathit{xtract})$.[4] Setup outputs a public parameter pp and an extraction key xk. Gen is given pp and outputs a PRF key prfk and a public tag τ. Eval is the PRF evaluation algorithm that takes as an input prfk and x in the domain and outputs y. By using Mark, we can generate a marked evaluation circuit that has embedded message $\mathsf{m} \in \{0,1\}^{\ell_\mathsf{m}}$ and can be used to evaluate $\mathsf{Eval}(\mathsf{prfk}, x')$ for almost all x'. Finally, $\mathcal{E}\mathit{xtract}$ is the extraction algorithm supposed to extract the embedded message from a pirated quantum evaluation circuit generated from the marked evaluation circuit. By default, in this work, we consider the public marking setting, where anyone can execute Mark. Thus, Mark takes pp as an input. On the other hand, we consider both the private extraction and the public extraction settings. Thus, the extraction key xk used

[4] In this paper, standard math font stands for classical algorithms, and calligraphic font stands for quantum algorithms.

by $\mathcal{E}xtract$ is kept secret by an authority in the private extraction setting and made public in the public extraction setting.

In this work, we allow $\mathcal{E}xtract$ to take the public tag τ generated with the original PRF key corresponding to the pirate circuit. In reality, we execute $\mathcal{E}xtract$ for a software when a user claims that the software is illegally generated by using her/his PRF key. Thus, it is natural to expect we can use a user's public tag for extraction. Moreover, pirate circuits are distinguishers, not predictors in this work. As discussed by Goyal et al. [GKWW21], security against pirate distinguishers is much preferable compared to security against pirate predictors considered in many previous works on watermarking. In this case, it seems that such additional information fed to $\mathcal{E}xtract$ is unavoidable. For a more detailed discussion on the syntax, see the discussion in Section 3.1.

It is also natural to focus on distinguishers breaking weak pseudorandomness of PRFs when we consider pirate distinguishers instead of pirate predictors. Goyal et al. [GKWW21] already discussed this point. Thus, we focus on watermarking weak PRF in this work.

Definition of unremovability against quantum adversaries. We say that a watermarking PRF scheme satisfies unremovability if given a marked evaluation circuit $\widetilde{C}$ that has an embedded message m, any adversary cannot generate a circuit such that it is a "good enough circuit", but the extraction algorithm fails to output m. In this work, we basically follow the notion of "good enough circuit" defined by Goyal et al. [GKWW21] as stated above. Let D be the following distribution for a PRF $\mathsf{Eval}(\mathsf{prfk}, \cdot) : \mathsf{Dom} \to \mathsf{Ran}$.

D**:** Generate $b \leftarrow \{0,1\}$, $x \leftarrow \mathsf{Dom}$, and $y_0 \leftarrow \mathsf{Ran}$. Compute $y_1 \leftarrow \mathsf{Eval}(\mathsf{prfk}, x)$. Output (b, x, y_b).

A circuit is defined as good enough circuit with respect to $\mathsf{Eval}(\mathsf{prfk}, \cdot)$ if given (x, y_b) output by D, it can correctly guess b with probability significantly greater than $1/2$. In other words, a circuit is defined as good enough if the circuit breaks weak PRF security.

Below, for a distribution D' whose output is of the form (b, x, y), let $\mathcal{M}_{D'} = (\boldsymbol{M}_{D',0}, \boldsymbol{M}_{D',1})$ be binary positive operator valued measures (POVMs) that represents generating random (b, x, y) from D' and testing if a quantum circuit can guess b from (x, y). Then, for a quantum state $|\psi\rangle$, the overall distinguishing advantage of it for the above distribution D is $\langle\psi| \boldsymbol{M}_{D,0} |\psi\rangle$. Thus, a natural adaptation of the above notion of goodness for quantum circuits might be to define a quantum state $|\psi\rangle$ as good if $\langle\psi| \boldsymbol{M}_{D,0} |\psi\rangle$ is significantly greater than $1/2$. However, this notion of goodness for quantum circuits is not really meaningful. The biggest issue is that it does not consider the stateful nature of quantum programs.

This issue was previously addressed by Zhandry [Zha20] in the context of traitor tracing against quantum adversaries. In the context of classical traitor tracing or watermarking, we can assume that a pirate circuit is stateless, or can be rewound to its original state. This assumption is reasonable. If we have the software description of the pirate circuit, such a rewinding is trivial. Even if we have a hardware box in which a pirate circuit is built, it seems that such a rewinding is possible by hard reboot or cutting power. On the other hand, in the context of quantum watermarking, we have to consider that a pirate circuit is inherently stateful since it is described as a quantum state. Operations to a quantum state can alter the state, and in general, it is impossible to rewind the state into its original state. Regarding the definition of good quantum circuits above, if we can somehow compute the average success probability $\langle\psi| \boldsymbol{M}_{D,0} |\psi\rangle$ of the quantum state $|\psi\rangle$, the process can change or destroy the quantum state $|\psi\rangle$. Namely, even if we once confirm that the quantum state $|\psi\rangle$ is good by computing $\langle\psi| \boldsymbol{M}_{D,0} |\psi\rangle$, we cannot know the success probability of the quantum state even right after the computation. Clearly, the above notion of goodness is not the right notion, and we need one that captures the stateful nature of quantum programs.

In the work on traitor tracing against quantum adversaries, Zhandry [Zha20] proposed a notion of goodness for quantum programs that solves the above issue. We adopt it. For the above POVMs $\mathcal{M}_D$, let $\mathcal{M}'_D$ be the projective measurement $\{P_p\}_{p\in[0,1]}$ that projects a state onto the eigenspaces of $\boldsymbol{M}_{D,0}$, where each p is an eigenvalue of $\boldsymbol{M}_{D,0}$. $\mathcal{M}'_D$ is called projective implementation of $\mathcal{M}_D$ and denoted as $\mathsf{ProjImp}(\mathcal{M}_D)$. Zhandry showed that the following process has the same output distribution as $\mathcal{M}_D$:

1. Apply the projective measurement $\mathcal{M}'_D = \mathsf{ProjImp}(\mathcal{M}_D)$ and obtain p.
2. Output 0 with probability p and output 1 with probability $1 - p$.

Intuitively, $\mathcal{M}'_D$ project a state to an eigenvector of $\boldsymbol{M}_{D,0}$ with eigenvalue p, which can be seen as a quantum state with success probability p. Using $\mathcal{M}'_D$, Zhandry defined that a quantum circuit is Live if the outcome of the measurement $\mathcal{M}'_D$ is significantly greater than $1/2$. The notion of Live is a natural extension of the classical goodness since it collapses to the classical goodness for a classical decoder. Moreover, we can ensure that a quantum state that is tested as Live still has a high success probability. On the other hand, the above notion of goodness cannot say anything about the post-tested quantum state's success probability even if the test is passed. In this work, we use the notion of Live quantum circuits as the notion of good quantum circuits.

Difficulty of quantum watermarking PRF. From the above discussion, our goal is to construct a watermarking PRF scheme that guarantees that we can extract the embedded message correctly if a pirated quantum circuit is Live. In watermarking PRF schemes, we usually extract an embedded message by applying several tests on success probability to a pirate circuit. When a pirate circuit is a quantum state, the set of tests that we can apply is highly limited compared to a classical circuit due to the stateful nature of quantum states.

One set of tests we can apply without destroying the quantum state is $\mathsf{ProjImp}(\mathcal{M}_{D'})$ for distributions D' that are indistinguishable from D from the view of the pirate circuit.[5] We denote this set as $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$. Zhandry showed that if distributions D_1 and D_2 are indistinguishable, the outcome of $\mathsf{ProjImp}(\mathcal{M}_{D_1})$ is close to that of $\mathsf{ProjImp}(\mathcal{M}_{D_2})$. By combining this property with the projective property of projective implementations, as long as the initial quantum state is Live and we apply only tests contained in $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$, the quantum state remains Live. On the other hand, if we apply a test outside of $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$, the quantum state might be irreversibly altered. This fact is a problem since the set $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$ only is not sufficient to implement the existing widely used construction method for watermarking PRF schemes.

To see this, we briefly review the method. In watermarking PRF schemes, the number of possible embedded messages is super-polynomial, and thus we basically need to extract an embedded message in a bit-by-bit manner. In the method, such a bit-by-bit extraction is done as follows. For every $i \in [\ell_{\mathsf{m}}]$, we define two distributions $S_{i,0}$ and $S_{i,1}$ whose output is of the form (b, x, y) as D above. Then, we design a marked circuit with embedded message $\mathsf{m} \in \{0,1\}^{\ell_{\mathsf{m}}}$ so that it can be used to guess b from (x, y) with probability significantly greater than $1/2$ only for $S_{i,0}$ (resp. $S_{i,1}$) if $\mathsf{m}[i] = 0$ (resp. $\mathsf{m}[i] = 1$). The extraction algorithm can extract i-th bit of the message $\mathsf{m}[i]$ by checking for which distributions of $S_{i,0}$ and $S_{i,1}$ a pirate circuit has a high distinguishing advantage.

As stated above, we cannot use this standard method to extract a message from quantum pirate circuits. The reason is that $S_{i,0}$ and $S_{i,1}$ are typically distinguishable. This implies that at least either one of $\mathsf{ProjImp}(\mathcal{M}_{S_{i,0}})$ or $\mathsf{ProjImp}(\mathcal{M}_{S_{i,1}})$ is not contained in $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$. Since the test outside of $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$ might destroy the quantum state, we might not be able to perform the process for all i, and fail to extract the entire bits of the embedded message.

It seems that to perform the bit-by-bit extraction for a quantum state, we need to extend the set of applicable tests and come up with a new extraction method.

Our solution: Use of reverse projective property. We find that as another applicable set of tests, we have $\mathsf{ProjImp}(\mathcal{M}_{D'})$ for distributions D' that are indistinguishable from D^{rev}, where D^{rev} is the following distribution.

D^{rev}**:** Generate $b \leftarrow \{0,1\}$, $x \leftarrow \mathsf{Dom}$, and $y_0 \leftarrow \mathsf{Ran}$. Compute $y_1 \leftarrow \mathsf{Eval}(\mathsf{prfk}, x)$. Output $(1 \oplus b, x, y_b)$.

We denote the set as $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D^{\mathsf{rev}}\}$. D^{rev} is the distribution the first bit of whose output is flipped from that of D. Then, $\mathcal{M}_{D^{\mathsf{rev}}}$ can be seen as POVMs that represents generating random (b, x, y_b) from D and testing

[5] In the actual extraction process, we use an approximation of projective implementation introduced by Zhandry [Zha20] since applying a projective implementation is inefficient. In this overview, we ignore this issue for simplicity.

if a quantum circuit *cannot* guess b from (x, y_b). Thus, we see that $\mathcal{M}_{D^{\mathsf{rev}}} = (\boldsymbol{M}_{D,1}, \boldsymbol{M}_{D,0})$. Recall that $\mathcal{M}_D = (\boldsymbol{M}_{D,0}, \boldsymbol{M}_{D,1})$.

Let $D_1 \in \{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$ and D_1^{rev} be the distribution that generates $(b, x, y) \leftarrow D_1$ and outputs $(1 \oplus b, x, y)$. D_1^{rev} is a distribution contained in $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D^{\mathsf{rev}}\}$. Similarly to the relation between D and D^{rev}, if $\mathcal{M}_{D_1} = (\boldsymbol{M}_{D_1,0}, \boldsymbol{M}_{D_1,1})$, we have $\mathcal{M}_{D_1^{\mathsf{rev}}} = (\boldsymbol{M}_{D_1^{\mathsf{rev}},1}, \boldsymbol{M}_{D_1^{\mathsf{rev}},0})$. Since $\boldsymbol{M}_{D_1,0} + \boldsymbol{M}_{D_1,1} = \boldsymbol{I}$, $\boldsymbol{M}_{D_1,0}$ and $\boldsymbol{M}_{D_1,1}$ share the same set of eigenvectors, and if a vector is an eigenvector of $\boldsymbol{M}_{D_1,0}$ with eigenvalue p, then it is also an eigenvector of $\boldsymbol{M}_{D_1,1}$ with eigenvalue $1-p$. Thus, if apply $\mathsf{ProjImp}(\mathcal{M}_{D_1})$ and $\mathsf{ProjImp}(\mathcal{M}_{D_1^{\mathsf{rev}}})$ successively to a quantum state and obtain the outcomes $\widetilde{p}_1$ and $\widetilde{p}'_1$, it holds that $\widetilde{p}'_1 = 1 - \widetilde{p}_1$. We call this property the reverse projective property of the projective implementation.

Combining projective and reverse projective properties and the outcome closeness for indistinguishable distributions of the projective implementation, we see that the following key fact holds.

Key fact: As long as the initial quantum state is Live and we apply tests contained in $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$ or $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) | D' \overset{\mathsf{c}}{\approx} D^{\mathsf{rev}}\}$, the quantum state remains Live. Moreover, if the outcome of applying $\mathsf{ProjImp}(\mathcal{M}_D)$ to the initial state is p, we get the outcome close to p every time we apply a test in $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$, and we get the outcome close to $1-p$ every time we apply a test in $\{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D^{\mathsf{rev}}\}$.

In this work, we perform bit-by-bit extraction of embedded messages by using the above key fact of the projective implementation. To this end, we introduce the new notion of extraction-less watermarking PRF as an intermediate primitive.

Via extraction-less watermarking PRF. An extraction-less watermarking PRF scheme has almost the same syntax as a watermarking PRF scheme, except that it does not have an extraction algorithm $\mathcal{E}xtract$ and instead has a simulation algorithm Sim. Sim is given the extraction key xk, the public tag τ, and an index $i \in [\ell_{\mathsf{m}}]$, and outputs a tuple of the form (γ, x, y). Sim simulates outputs of D or D^{rev} for a pirate circuit depending on the message embedded to the marked circuit corresponding to the pirate circuit. More concretely, we require that from the view of the pirate circuit generated from a marked circuit with embedded message $\mathsf{m} \in \{0,1\}^{\ell_{\mathsf{m}}}$, outputs of Sim are indistinguishable from those of D if $\mathsf{m}[i] = 0$ and are indistinguishable from those of D^{rev} if $\mathsf{m}[i] = 1$ for every $i \in [\ell_{\mathsf{m}}]$. We call this security notion simulatability for mark-dependent distributions (SIM-MDD security).

By using an extraction-less watermarking PRF scheme $\mathsf{ELWMPRF}$, we construct a watermarking PRF scheme WMPRF against quantum adversaries as follows. We use $\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}$ of $\mathsf{ELWMPRF}$ as $\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}$ of WMPRF, respectively. We explain how to construct the extraction algorithm $\mathcal{E}xtract$ of WMPRF using Sim of $\mathsf{ELWMPRF}$. For every $i \in [\ell_{\mathsf{m}}]$, we define $D_{\tau,i}$ as the distribution that outputs randomly generated $(\gamma, x, y) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i)$. Given

xk, τ, and a quantum state $|\psi\rangle$, $\mathcal{E}xtract$ extracts the embedded message in the bit-by-bit manner by repeating the following process for every $i \in [\ell_{\mathsf{m}}]$.

– Apply $\mathsf{ProjImp}(\mathcal{M}_{D_{\tau,i}})$ to $|\psi_{i-1}\rangle$ and obtain the outcome $\widetilde{p}_i$, where $|\psi_0\rangle = |\psi\rangle$ and $|\psi_{i-1}\rangle$ is the state after the $(i-1)$-th loop for every $i \in [\ell_{\mathsf{m}}]$.
– Set $\mathsf{m}'_i = 0$ if $\widetilde{p}_i > 1/2$ and otherwise $\mathsf{m}'_i = 1$.

The extracted message is set to $\mathsf{m}'_1 \| \cdots \| \mathsf{m}'_{\ell_{\mathsf{m}}}$.

We show that the above construction satisfies unremovability. Suppose an adversary is given marked circuit $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$ and generates a quantum state $|\psi\rangle$, where $(\mathsf{pp}, \mathsf{xk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\mathsf{prfk}, \tau) \leftarrow \mathsf{Gen}(\mathsf{pp})$. Suppose also that $|\psi\rangle$ is Live. This assumption means that the outcome p of applying $\mathsf{ProjImp}(\mathcal{M}_D)$ to $|\psi\rangle$ is $1/2 + \epsilon$, where ϵ is an inverse polynomial. For every $i \in [\ell_{\mathsf{m}}]$, from the SIM-MDD security of $\mathsf{ELWMPRF}$, $D_{\tau,i}$ is indistinguishable from D if $\mathsf{m}[i] = 0$ and is indistinguishable from D^{rev} if $\mathsf{m}[i] = 1$. This means that $D_{\tau,i} \in \{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D\}$ if $\mathsf{m}[i] = 0$ and $D_{\tau,i} \in \{\mathsf{ProjImp}(\mathcal{M}_{D'}) \mid D' \overset{\mathsf{c}}{\approx} D^{\mathsf{rev}}\}$ if $\mathsf{m}[i] = 1$. Then, from the above key fact of the projective implementation, it holds that $\widetilde{p}_i$ is close to $1/2 + \epsilon > 1/2$ if $\mathsf{m}[i] = 0$ and is close to $1/2 - \epsilon < 1/2$ if $\mathsf{m}[i] = 1$. Therefore, we see that $\mathcal{E}xtract$ correctly extract m from $|\psi\rangle$. This means that WMPRF satisfies unremovability.

The above definition, construction, and security analysis are simplified and ignore many subtleties. The most significant point is that we use approximated projective implementations introduced by Zhandry [Zha20] instead of projective implementations in the actual construction since applying a projective implementation is an inefficient process. Moreover, though the outcomes of (approximate) projective implementations for indistinguishable distributions are close, in the actual analysis, we have to take into account that the outcomes gradually change every time we apply an (approximate) projective implementation. These issues can be solved by doing careful parameter settings.

Comparison with the work by Zhandry [Zha20]. Some readers familiar with Zhandry's work [Zha20] might think that our technique contradicts the lesson from Zhandry's work since it essentially says that once we find a large gap in success probabilities, the tested quantum pirate circuit might self-destruct. However, this is not the case. What Zhandry's work really showed is the following. Once a quantum pirate circuit itself detects that there is a large gap in success probabilities, it might self-destruct. Even if an extractor finds a large gap in success probabilities, if the tested quantum pirate circuit itself cannot detect the large gap, the pirate circuit cannot self-destruct. In Zhandry's work, whenever an extractor finds a large gap, the tested pirate circuit also detects the large gap. In our work, the tested pirate circuit cannot detect a large gap throughout the extraction process while an extractor can find it.

The reason why a pirate circuit cannot detect a large gap in our scheme even if an extractor can find it is as follows. Recall that in the above extraction process of our scheme based on an extraction-less watermarking PRF scheme, we apply $\mathsf{ProjImp}(\mathcal{M}_{D_{\tau,i}})$ to the tested pirate circuit for every $i \in [\ell_{\mathsf{m}}]$. Each $D_{\tau,i}$ outputs a tuple of the form (b, x, y) and is indistinguishable from D or D^{rev} depending on the embedded message. In the process, we apply $\mathsf{ProjImp}(\mathcal{M}_{D_{\tau,i}})$ for every $i \in [\ell_{\mathsf{m}}]$, and we get the success probability p if $D_{\tau,i}$ is indistinguishable from D and we get $1-p$ if $D_{\tau,i}$ is indistinguishable from $D^{\mathtt{rev}}$. The tested pirate circuit needs to know which of D or $D^{\mathtt{rev}}$ is indistinguishable from the distribution $D_{\tau,i}$ behind the projective implementation to know which of p or $1-p$ is the result of an application of a projective implementation. However, this is impossible. The tested pirate circuit receives only (x, y) part of $D_{\tau,i}$'s output and not b part. (Recall that the task of the pirate circuit is to guess b from (x, y).) The only difference between D and $D^{\mathtt{rev}}$ is that the first-bit b is flipped. Thus, if the b part is dropped, $D_{\tau,i}$ is, in fact, indistinguishable from both D and $D^{\mathtt{rev}}$. As a result, the pirate program cannot know which of p or $1 - p$ is the result of an application of a projective implementation. In other words, the pirate circuit cannot detect a large gap in our extraction process.

Instantiating extraction-less watermarking PRF. In the rest of this overview, we will explain how to realize extraction-less watermarking PRF.

We consider the following two settings similar to the ordinary watermarking PRF. Recall that we consider the public marking setting by default.

Private-simulatable: In this setting, the extraction key xk fed into Sim is kept secret. We require that SIM-MDD security hold under the existence of the simulation oracle that is given a public tag τ' and an index $i' \in [\ell_{\mathsf{m}}]$ and returns $\mathsf{Sim}(\mathsf{xk}, \tau', i')$. An extraction-less watermarking PRF scheme in this setting yields a watermarking PRF scheme against quantum adversaries in private-extractable setting where unremovability holds for adversaries who can access the extraction oracle.

Public-simulatable: In this setting, the extraction key xk is publicly available. An extraction-less watermarking PRF scheme in this setting yields a watermarking PRF scheme against quantum adversaries in the public-extractable setting.

We provide a construction in the first setting using private constrained PRF based on the hardness of the LWE assumption. Also, we provide a construction in the second setting based on IO and the hardness of the LWE assumption.

To give a high-level idea behind the above constructions, in this overview, we show how to construct a public-simulatable extraction-less watermarking PRF in the token-based setting [CHN+18]. In the token-based setting, we treat a marked circuit $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$ as a tamper-proof hardware token that an adversary can only access in a black-box way.

Before showing the actual construction, we explain the high-level idea. Recall that SIM-MDD security requires that an adversary $\mathcal{A}$ who is given $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$ cannot distinguish $(\gamma^*, x^*, y^*) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ from an output of D if $\mathsf{m}[i^*] = 0$ and from that of $D^{\mathtt{rev}}$ if $\mathsf{m}[i^*] = 1$. This is the same as requiring that $\mathcal{A}$ cannot distinguish $(\gamma^*, x^*, y^*) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ from that of the following distribution $D_{\mathtt{real},i^*}$. We can check that $D_{\mathtt{real},i^*}$ is identical with D if $\mathsf{m}[i^*] = 0$ and with $D^{\mathtt{rev}}$ if $\mathsf{m}[i^*] = 1$.

$D_{\texttt{real},i^*}$: Generate $\gamma \leftarrow \{0,1\}$ and $x \leftarrow \mathsf{Dom}$. Then, if $\gamma = \mathsf{m}[i^*]$, generate $y \leftarrow \mathsf{Ran}$, and otherwise, compute $y \leftarrow \mathsf{Eval}(\mathsf{prfk}, x)$. Output (γ, x, y).

Essentially, the only attack that $\mathcal{A}$ can perform is to feed x^* contained in the given tuple (γ^*, x^*, y^*) to $\widetilde{C}$ and compares the result $\widetilde{C}(x^*)$ with y^*, if we ensure that γ^*, x^* are pseudorandom. In order to make the construction immune to this attack, letting $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$ and $(\gamma^*, x^*, y^*) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$, we have to design Sim and $\widetilde{C}$ so that

– If $\gamma = \mathsf{m}[i^*]$, $\widetilde{C}(x^*)$ outputs a value different from y^*.
– If $\gamma \neq \mathsf{m}[i^*]$, $\widetilde{C}(x^*)$ outputs y^*.

We achieve these conditions as follows. First, we set (γ^*, x^*, y^*) output by $\mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ so that γ^* and y^* is random values and x^* is an encryption of $y^* \| i^* \| \gamma^*$ by a public-key encryption scheme with pseudorandom ciphertext property, where the encryption key pk is included in τ. Then, we set $\widetilde{C}$ as a token such that it has the message m and the decryption key sk corresponding to pk hardwired, and it outputs y^* if the input is decryptable and $\gamma^* \neq \mathsf{m}[i^*]$ holds for the decrypted $y^* \| i^* \| \gamma^*$, and otherwise behaves as $\mathsf{Eval}(\mathsf{prfk}, \cdot)$. The actual construction is as follows.

Let PRF be a PRF family consisting of functions $\{\mathsf{F}_{\mathsf{prfk}}(\cdot) : \{0,1\}^n \to \{0,1\}^\lambda | \mathsf{prfk}\}$, where λ is the security parameter and n is sufficiently large. Let $\mathsf{PKE} = (\mathsf{KG}, \mathsf{E}, \mathsf{D})$ be a CCA secure public-key encryption scheme satisfying pseudorandom ciphertext property. Using these ingredients, We construct an extraction-less watermarking PRF scheme $\mathsf{ELWMPRF} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}, \mathsf{Sim})$ as follows.

$\mathsf{Setup}(1^\lambda)$: In this construction, $\mathsf{pp} := \perp$ and $\mathsf{xk} := \perp$.

$\mathsf{Gen}(\mathsf{pp})$: It generates a fresh PRF key prfk of PRF and a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^\lambda)$. The PRF key is $(\mathsf{prfk}, \mathsf{sk})$ and the corresponding public tag is pk.

$\mathsf{Eval}((\mathsf{prfk}, \mathsf{sk}), x)$: It simply outputs $\mathsf{F}_{\mathsf{prfk}}(x)$.

$\mathsf{Mark}(\mathsf{pp}, (\mathsf{prfk}, \mathsf{sk}), \mathsf{m})$: It generates the following taken $\widetilde{C}[\mathsf{prfk}, \mathsf{sk}, \mathsf{m}]$.

```
Hard-Coded Constants: $\mathsf{prfk}, \mathsf{sk}, \mathsf{m}$.
Input: $x \in \{0,1\}^n$.
  1. Try to decrypt $y \| i \| \gamma \leftarrow \mathsf{D}(\mathsf{sk}, x)$ with $y \in \{0,1\}^\lambda$, $i \in [\ell_\mathsf{m}]$, and $\gamma \in \{0,1\}$.
  2. If decryption succeeds, output $y$ if $\gamma \neq \mathsf{m}[i]$ and $\mathsf{F}_{\mathsf{prfk}}(x)$ otherwise.
  3. Otherwise, output $\mathsf{F}_{\mathsf{prfk}}(x)$.
```

$\mathsf{Sim}(\mathsf{xk}, \tau, i)$: It first generates $\gamma \leftarrow \{0,1\}$ and $y \leftarrow \{0,1\}^\lambda$. Then, it parses $\tau := \mathsf{pk}$ and generates $x \leftarrow \mathsf{E}(\mathsf{pk}, y \| i \| \gamma)$. Finally, it outputs (γ, x, y).

We check that $\mathsf{ELWMPRF}$ satisfies SIM-MDD security. For simplicity, we fix the message $\mathsf{m} \in [\ell_\mathsf{m}]$ embedded into the challenge PRF key. Then, for any adversary $\mathcal{A}$ and $i^* \in [\ell_\mathsf{m}]$, SIM-MDD security requires that given $\widetilde{C}[\mathsf{prfk}, \mathsf{sk}, \mathsf{m}] \leftarrow \mathsf{Mark}(\mathsf{mk}, \mathsf{prfk}, \mathsf{m})$ and $\tau = \mathsf{pk}$, $\mathcal{A}$ cannot distinguish $(\gamma^*, x^* = \mathsf{E}(\mathsf{pk}, y^* \| i^* \| \gamma^*), y^*) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ from an output of D if $\mathsf{m}[i^*] = 0$ and is indistinguishable from D^{rev} if $\mathsf{m}[i^*] = 1$.

We consider the case of $\mathsf{m}[i^*] = 0$. We can finish the security analysis by considering the following sequence of mutually indistinguishable hybrid games, where $\mathcal{A}$ is given $(\gamma^*, x^* = \mathsf{E}(\mathsf{pk}, y^* \| i^* \| \gamma^*), y^*) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ in the first game, and on the other hand, is given $(\gamma^*, x^*, y^*) \leftarrow D$ in the last game. We first change the game so that x^* is generated as a uniformly random value instead of $x^* \leftarrow \mathsf{E}(\mathsf{pk}, y^* \| i^* \| \gamma^*)$ by using the pseudorandom ciphertext property under CCA of PKE. This is possible since the CCA oracle can simulate access to the marked token $\widetilde{C}[\mathsf{prfk}, \mathsf{sk}, \mathsf{m}]$ by $\mathcal{A}$. Then, we further change the security game so that if $\gamma^* = 1$, y^* is generated as $\mathsf{F}_{\mathsf{prfk}}(x^*)$ instead of a uniformly random value by using the pseudorandomness of PRF. Note that if $\gamma^* = 0$, y^* remains uniformly at random. We see that if $\gamma^* = 1$, the token $\widetilde{C}[\mathsf{prfk}, \mathsf{sk}, \mathsf{m}]$ never evaluate $\mathsf{F}_{\mathsf{prfk}}(x^*)$ since $\mathsf{m}[i^*] \neq \gamma^*$. Thus, this change is possible. We see that now the distribution of (γ^*, x^*, y^*) is exactly the same as that output by D. Similarly, in the case of $\mathsf{m}[i^*] = 1$, we can show that an output of $\mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ is indistinguishable from that output by D^{rev}. The only difference is that in the final step, we change the security game so that y^* is generated as $\mathsf{F}_{\mathsf{prfk}}(x^*)$ if $\gamma^* = 0$.

In the actual public-simulatable construction, we implement this idea using iO and puncturable encryption [CHN+18] instead of token and CCA secure public-key encryption. Also, in the actual secret-simulatable construction, we basically follow the same idea using private constrained PRF and secret-key encryption.

1.4 Organization

Due to the space limitation, we omit preliminaries including notations, basics on quantum informations, and definitions of standard cryptographic tools. We also omit most security proofs. See the full version of this paper for omitted contents. In Section 2, we introduce some notions of quantum measurements. In Section 3, we define watermarking PRF against quantum adversaries. In Section 4, we define extraction-less watermarking PRF. In Section 5, we show we can realize watermarking PRF against quantum adversaries from extraction-less watermarking PRF. In Section 6, we provide an instantiation of extraction-less watermarking PRF with private simulation based on the LWE assumption. In Section 7, we provide an instantiation of extraction-less watermarking PRF with public simulation based on IO and the LWE assumption.

2 Measurement Implementation

Definition 2.1 (Projective Implementation). *Let:*

- $\mathcal{P} = (\boldsymbol{P}, \boldsymbol{I} - \boldsymbol{P})$ *be a binary outcome POVM*
- D *be a finite set of distributions over outcomes* $\{0, 1\}$
- $\mathcal{E} = \{\boldsymbol{E}_D\}_{D \in \mathcal{D}}$ *be a projective measurement with index set* $\mathcal{D}$.

We define the following measurement.

1. *Measure under the projective measurement* $\mathcal{E}$ *and obtain a distribution* D *over* $\{0, 1\}$.

2. *Output a bit sampled from the distribution D.*

We say this measurement is a projective implementation of $\mathcal{P}$, denoted by $\mathsf{ProjImp}(\mathcal{P})$ *if it is equivalent to $\mathcal{P}$.*

Theorem 2.1 ([Zha20, Lemma 1]). *Any binary outcome POVM $\mathcal{P} = (\boldsymbol{P}, \boldsymbol{I} - \boldsymbol{P})$ has a projective implementation* $\mathsf{ProjImp}(\mathcal{P})$.

Definition 2.2 (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 $x \in \mathbb{R}$:*

$$\begin{aligned} &\Pr[D_0 \le x] \le \Pr[D_1 \le x + \epsilon] + \delta, &&\Pr[D_0 \ge x] \le \Pr[D_1 \ge x - \epsilon] + \delta, \\ &\Pr[D_1 \le x] \le \Pr[D_0 \le x + \epsilon] + \delta, &&\Pr[D_1 \ge x] \le \Pr[D_0 \ge x - \epsilon] + \delta. \end{aligned}$$

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

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

Definition 2.3 ((ϵ, δ)-Almost Projective [Zha20]). *A real-valued quantum measurement $\mathcal{M} = \{\boldsymbol{M}_i\}_{i \in \mathcal{I}}$ is (ϵ, δ)-almost projective if the following holds. For any quantum state $|\psi\rangle$, we apply $\mathcal{M}$ twice in a row to $|\psi\rangle$ and obtain measurement outcomes x and y, respectively. Then, $\Pr[|x - y| \le \epsilon] \ge 1 - \delta$.*

Theorem 2.2 ([Zha20, Theorem 2]). *Let D be any probability distribution and $\mathcal{P}$ be a collection of projective measurements. For any $0 < \epsilon, \delta < 1$, there exists an algorithm of measurement $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},\mathcal{D}}$ that satisfies the following.*

– $\Delta^{\epsilon}_{\mathsf{Shift}}(\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D}, \mathsf{ProjImp}(\mathcal{P}_D)) \le \delta$.
– *$\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D}$ is (ϵ, δ)-almost projective.*
– *The expected running time of $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D}$ is $T_{\mathcal{P},D} \cdot \mathrm{poly}(1/\epsilon, \log(1/\delta))$ where $T_{\mathcal{P},D}$ is the combined running time of D, the procedure mapping $i \to (\boldsymbol{P}_i, \boldsymbol{I} - \boldsymbol{P}_i)$, and the running time of measurement $(\boldsymbol{P}_i, \boldsymbol{I} - \boldsymbol{P}_i)$.*

Theorem 2.3 ([Zha20, Corollary 1]). *Let q be an efficiently constructible, potentially mixed state, and D_0, D_1 efficiently sampleable distributions. If D_0 and D_1 are computationally indistinguishable, for any inverse polynomial ϵ and any function δ, we have $\Delta^{3\epsilon}_{\mathsf{Shift}}(\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D_0}, \mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D_1}) \le 2\delta + \mathsf{negl}(\lambda)$.*

Note that the indistinguishability of D_0 and D_1 needs to hold against distinguishers who can construct q in the theorem above. However, this fact is not explicitly stated in [Zha20]. We need to care about this condition if we need secret information to construct q, and the secret information is also needed to sample an output from D_0 or D_1. We handle such a situation when analyzing the unremovability of our privately extractable watermarking PRF. In that situation, we need a secret extraction key to construct q and sample an output from D_0 and D_1.

We also define the notion of the reverse almost projective property of API.

Definition 2.4 ((ϵ,δ)-Reverse Almost Projective). *Let $\mathcal{P} = \{(\Pi_i, \boldsymbol{I} - \Pi_i)\}_i$ be a collection of binary outcome projective measurements. Let D be a distribution. We also let $\mathcal{P}^{\mathsf{rev}} = \{(\boldsymbol{I} - \Pi_i, \Pi_i)\}_i$. We say $\mathcal{API}$ is (ϵ,δ)-reverse almost projective if the following holds. For any quantum state $|\psi\rangle$, we apply $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D}$ and $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P}^{\mathsf{rev}},D}$ in a row to $|\psi\rangle$ and obtain measurement outcomes x and y, respectively. Then,* $\Pr[|(1-x)-y| \le \epsilon] \ge 1-\delta$.

We show that the measurement algorithm$\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},D}$ in Theorem 2.2 also satisfies Definition 2.4. See the full version for the proof.

3 Definition of Quantum Watermarking

We introduce definitions for watermarking PRFs against quantum adversaries in this section.

3.1 Syntax and Pseudorandomness

Definition 3.1 (Watermarking PRF). *A watermarking PRF* WMPRF *for the message space* $\mathcal{M} := \{0,1\}^{\ell_{\mathsf{m}}}$ *with domain* Dom *and range* Ran *is a tuple of five algorithms* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}, \mathcal{Extract})$.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pp}, \mathsf{xk})$**:** *The setup algorithm takes as input the security parameter and outputs a public parameter* pp *and an extraction key* xk.

$\mathsf{Gen}(\mathsf{pp}) \to (\mathsf{prfk}, \tau)$**:** *The key generation algorithm takes as input the public parameter* pp *and outputs a PRF key* prfk *and a public tag* τ.

$\mathsf{Eval}(\mathsf{prfk}, x) \to y$**:** *The evaluation algorithm takes as input a PRF key* prfk *and an input* $x \in \mathsf{Dom}$ *and outputs* $y \in \mathsf{Ran}$.

$\mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m}) \to \widetilde{C}$**:** *The mark algorithm takes as input the public parameter* pp, *a PRF key* prfk, *and a message* $\mathsf{m} \in \{0,1\}^{\ell_{\mathsf{m}}}$, *and outputs a marked evaluation circuit* $\widetilde{C}$.

$\mathcal{Extract}(\mathsf{xk}, \tau, \mathcal{C}', \epsilon) \to \mathsf{m}'$**:** *The extraction algorithm takes as input an extraction key* xk, *a tag* τ, *a quantum circuit with classical inputs and outputs* $\mathcal{C}' = (q, \boldsymbol{U})$, *and a parameter* ϵ, *and outputs* m' *where* $\mathsf{m}' \in \{0,1\}^{\ell_{\mathsf{m}}} \cup \{\mathsf{unmarked}\}$.

Evaluation Correctness: *For any message* $\mathsf{m} \in \{0,1\}^{\ell_{\mathsf{m}}}$, *it holds that*

$$\Pr\left[\widetilde{C}(x) = \mathsf{Eval}(\mathsf{prfk}, x) \;\middle|\; \begin{array}{c} (\mathsf{pp}, \mathsf{xk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{prfk}, \tau) \leftarrow \mathsf{Gen}(\mathsf{pp}) \\ \widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m}) \\ x \leftarrow \mathsf{Dom} \end{array}\right] \ge 1 - \mathsf{negl}(\lambda).$$

Remark 3.1 (On extraction correctness). Usually, a watermarking PRF scheme is required to satisfy extraction correctness that ensures that we can correctly extract the embedded mark from an honestly marked circuit. However, as observed by Quach et al. [QWZ18], if we require the extraction correctness to hold for a randomly chosen PRF key, it is implied by unremovability defined below. Note that the unremovability defined below considers a distinguisher as a pirate circuit. However, it implies the extraction correctness since we can easily transform an honestly marked circuit into a successful distinguisher. Thus, we do not explicitly require a watermarking PRF scheme to satisfy extraction correctness in this work.

Remark 3.2 (On public marking). We consider only watermarking PRFs with public marking as in Definition 3.1 since we can achieve public marking by default. The reason is as follows. Suppose that we generate pp, xk, and a marking key mk at the setup. When we generate a PRF key and a public tag at Gen, we can first generate $(\mathsf{pp}', \mathsf{xk}', \mathsf{mk}') \leftarrow \mathsf{Setup}(1^\lambda)$ from scratch (ignoring the original $(\mathsf{pp}, \mathsf{xk}, \mathsf{mk})$) and set a PRF key $\widehat{\mathsf{prfk}} := (\mathsf{prfk}', \mathsf{mk}')$ and a public tag $\widehat{\tau} := (\mathsf{pp}', \mathsf{xk}', \tau')$ where $(\mathsf{prfk}', \tau') \leftarrow \mathsf{Gen}(\mathsf{pp}')$. That is, anyone can generate a marked circuit from $\widehat{\mathsf{prfk}} = (\mathsf{prfk}', \mathsf{mk}')$ by $\mathsf{Mark}(\mathsf{mk}', \mathsf{prfk}', \mathsf{m})$. Therefore, we consider public marking by default in our model.

Discussion on syntax. Definition 3.1 is a natural quantum variant of classical watermarking PRFs except that the key generation algorithm outputs a public tag τ, and the extraction algorithm uses it. Such a public tag is not used in previous works on watermarking PRFs [CHN+18,KW21,QWZ18,KW19,YAL+19]. A public tag should not harm watermarking PRF security. We justify using τ as follows.

First, we need to obtain many pairs of input and output to extract an embedded message from a marked PRF in almost all known (classical) watermarking constructions [CHN+18,BLW17,KW21,QWZ18,KW19,YAL+19,GKM+19,Nis20]. This is because we must check whether a tested PRF circuit outputs particular values for particular inputs which *depends on the target PRF* (such particular inputs are known as marked points). Suppose marked points are fixed and do not depend on a PRF that will be marked. In that case, an adversary can easily remove an embedded message by destroying functionalities at the fixed marked points that could be revealed via a (non-target) marked PRF that an adversary generated. Recall that we consider the public marking setting. The attack was already observed by Cohen et al. [CHN+18].

Second, we consider a stronger adversary model than that in most previous works as the definition of traceable PRFs by Goyal et al. [GKWW21]. An adversary outputs a distinguisher-based pirate circuit in our security definition rather than a pirate circuit that computes an entire output of a PRF. This is a refined and realistic model, as Goyal et al. [GKWW21] argued. In this model, we cannot obtain a valid input-output pair from a pirate circuit anymore. Such a pair is typical information related to a target PRF. Goyal et al. resolve this issue by introducing a tracing key that is generated from a target PRF. Note that parameters of watermarking (pp and xk) should *not* be generated from a PRF since we consider many different PRF keys in the watermarking PRF setting.

Thus, if we would like to achieve an extraction algorithm and the stronger security notion simultaneously, an extraction algorithm should somehow take information related to a target PRF as input to correctly extract an embedded message. In the weaker adversary model, an extraction algorithm can easily obtain many valid input and output pairs by running a tested circuit many times. However, in the stronger distinguisher-based pirate circuit model, a pirate circuit outputs a single decision bit.

To resolve this issue, we introduce public tags. We think it is natural to have information related to the original PRF key in an extraction algorithm. In reality, we check a circuit when a user claims that her/his PRF key (PRF evaluation circuit) is illegally used. Thus, it is natural to expect we can use a user's public tag for extraction. This setting resembles watermarking for public-key cryptographic primitives, where a user public key is available in an extraction algorithm. In addition, public tags do not harm PRF security in our constructions. It is unclear whether we can achieve unremovability in the stronger distinguisher-based model without any syntax change (even in the classical setting). [6]

Extended pseudorandomness. We consider extended weak pseudorandomness, where weak pseudorandomness holds even if the adversary generates pp. This notion is the counterpart of extended pseudorandomness by Quach et al. [QWZ18], where pseudorandomness holds in the presence of the extraction oracle. However, our pseudorandomness holds even against an authority unlike extended pseudorandomness by Quach et al. since we allow adversaries to generate a public parameter.

Definition 3.2 (Extended Weak Pseudorandomness against Authority). *To define extended weak pseudorandomness for watermarking PRFs, we define the game* $\mathsf{Exp}^{\mathsf{ext\text{-}wprf}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda)$ *as follows.*

1. *$\mathcal{A}$ first sends pp to the challenger.*
2. *The challenger generates $(\mathsf{prfk}, \tau) \leftarrow \mathsf{Gen}(\mathsf{pp})$ and sends τ to $\mathcal{A}$.*
3. *The challenger chooses $\mathsf{coin} \leftarrow \{0,1\}$. $\mathcal{A}$ can access to the following oracles.*

   $O_{\mathtt{wprf}}$**:** *When this is invoked (no input), it returns (a,b) where $a \leftarrow \mathsf{Dom}$ and $b := \mathsf{Eval}(\mathsf{prfk}, a)$.*

   $O_{\mathtt{chall}}$**:** *When this is invoked (no input), it returns:*
   - *(a,b) where $a \leftarrow \mathsf{Dom}$ and $b := \mathsf{Eval}(\mathsf{prfk}, a)$ if $\mathsf{coin} = 0$,*
   - *(a,b) where $a \leftarrow \mathsf{Dom}$ and $b \leftarrow \mathsf{Ran}$ if $\mathsf{coin} = 1$.*

   This oracle is invoked only once.
4. *When $\mathcal{A}$ terminates with output coin', the challenger outputs 1 if $\mathsf{coin} = \mathsf{coin}'$ and 0 otherwise.*

[6] Even if we consider the weaker adversary model, the same issue appears in the quantum setting in the end. If we run a quantum circuit for an input and measure the output, the measurement could irreversibly alter the quantum state and we lost the functionality of the original quantum state. That is, there is no guarantee that we can correctly check whether a tested quantum circuit is marked or not *after* we obtain a single valid pair of input and output by running the circuit. However, as we explained above, we want to obtain information related to a target PRF for extraction. Thus, we need a public tag in the syntax in either case.

We say that WMPRF *is extended weak pseudorandom if for every QPT $\mathcal{A}$, we have*

$$\mathsf{Adv}^{\mathsf{ext\text{-}wprf}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda) = 2\left|\Pr\left[\mathsf{Exp}^{\mathsf{ext\text{-}wprf}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda) = 1\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

3.2 Unremovability against Quantum Adversaries

We define unremovability for watermarking PRFs against quantum adversaries. We first define quantum program with classical inputs and outputs and then define unremovability.

Definition 3.3 (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.*

Definition 3.4 (Unremovability for private extraction). *We consider the public marking and secret extraction setting here. Let $\epsilon \geq 0$. We define the game $\mathsf{Expt}^{\mathsf{nrmv}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda,\epsilon)$ as follows.*

1. *The challenger generates $(\mathsf{pp},\mathsf{xk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and gives pp to the adversary $\mathcal{A}$. $\mathcal{A}$ send $\mathsf{m} \in \{0,1\}^{\ell_\mathsf{m}}$ to the challenger. The challenger generates $(\mathsf{prfk},\tau) \leftarrow \mathsf{Gen}(\mathsf{pp})$, computes $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp},\mathsf{prfk},\mathsf{m})$, and sends τ and $\widetilde{C}$ to $\mathcal{A}$.*
2. *$\mathcal{A}$ can access to the following oracle.*
   O_{ext}: *On input τ' and a quantum circuit $\mathcal{C}$, it returns $\mathcal{E}xtract(\mathsf{xk},\mathcal{C},\tau',\epsilon)$.*
3. *Finally, the adversary outputs a "pirate" quantum circuit $\mathcal{C}_{\text{☠}} = (q,\boldsymbol{U})$, where $\mathcal{C}_{\text{☠}}$ is a quantum program with classical inputs and outputs whose first register (i.e., output register) is $\mathbb{C}^2$ and $\boldsymbol{U}$ is a compact classical description of $\{\boldsymbol{U}_{x,y}\}_{x\in\mathsf{Dom},y\in\mathsf{Ran}}$.*

   Let D be the following distribution.

D: *Generate $b \leftarrow \{0,1\}$, $x \leftarrow \mathsf{Dom}$, and $y_0 \leftarrow \mathsf{Ran}$. Compute $y_1 \leftarrow \mathsf{Eval}(\mathsf{prfk},x)$. Output (b,x,y_b).*

We also let $\mathcal{P} = (\boldsymbol{P}_{b,x,y},\boldsymbol{Q}_{b,x,y})_{b,x,y}$ be a collection of binary outcome projective measurements, where

$$\boldsymbol{P}_{b,x,y} = \boldsymbol{U}^\dagger_{x,y}\left|b\right\rangle\left\langle b\right|\boldsymbol{U}_{x,y} \quad \text{and} \quad \boldsymbol{Q}_{b,x,y} = \boldsymbol{I} - \boldsymbol{P}_{b,x,y}.$$

Moreover, we let $\mathcal{M}_D = (\boldsymbol{P}_D,\boldsymbol{Q}_D)$ be binary outcome POVMs, where

$$\boldsymbol{P}_D = \sum_{r\in\mathcal{R}}\frac{1}{|\mathcal{R}|}\boldsymbol{P}_{D(r)} \quad \text{and} \quad \boldsymbol{Q}_D = \boldsymbol{I} - \boldsymbol{P}_D.$$

Live: *When applying the measurement $\mathsf{ProjImp}(\mathcal{M}_D)$ to q, we obtain a value p such that $p \geq \frac{1}{2} + \epsilon$.*

$\mathsf{GoodExt}$: *When Computing* $\mathsf{m}' \leftarrow \mathcal{E}xtract(\mathsf{xk}, \mathcal{C}_{\skull}, \tau, \epsilon)$*, it holds that* $\mathsf{m}' \neq \mathsf{unmarked}$.
BadExt: *When Computing* $\mathsf{m}' \leftarrow \mathcal{E}xtract(\mathsf{xk}, \mathcal{C}_{\skull}, \tau, \epsilon)$*, it holds that* $\mathsf{m}' \notin \{\mathsf{m}, \mathsf{unmarked}\}$.

We say that WMPRF *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, $(\boldsymbol{P}_{b,x,y}, \boldsymbol{Q}_{b,x,y})$ is a projective measurement that feeds (x, y) to $\mathcal{C}_{\skull}$ and checks whether the outcome is b or not (and then uncomputes). Then, $\mathcal{M}_D$ can be seen as POVMs that results in 0 with the probability that $\mathcal{C}_{\skull}$ can correctly guess b from (x, y_b) for (b, x, y_b) generated randomly from D.

Remark 3.3 (On attack model). We check whether $\mathcal{C}_{\skull}$ correctly distinguishes a real PRF value from a random value or not by applying $\mathsf{ProjImp}(\mathcal{M}_D)$ to q. This attack model follows the refined and more realistic attack model by Goyal et al. [GKWW21]. The adversary outputs a pirate circuit that computes an entire PRF value in all previous works except their work.

The distinguisher-based pirate circuit model is compatible with the (quantum) pirate decoder model of traitor tracing. Thus, our attack model also follows the attack model of quantum traitor tracing (the black box projection model) by Zhandry [Zha20, Section 4.2].[7]

As in the traitor tracing setting [Zha20], $\mathsf{ProjImp}(\mathcal{M}_D)$ is inefficient in general. We can handle this issue as Zhandry did. We will use an approximate version of $\mathsf{ProjImp}(\mathcal{M}_D)$ to achieve an efficient reduction. In addition, we cannot apply both $\mathsf{ProjImp}(\mathcal{M}_D)$ and $\mathcal{E}xtract$ to $\mathcal{C}_{\skull}$ simultaneously. However, the condition $\Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$ claims that an embedded mark cannot be removed as long as the pirate circuit is alive. This fits the spirit of watermarking. See Zhandry's paper [Zha20, Section 4] for more discussion on the models.

Remark 3.4 (On selective message). As we see in Definition 3.4, we consider the selective setting for private extraction case, where $\mathcal{A}$ must send the target message m to the challenger before $\mathcal{A}$ accesses to the oracle $O_{\mathtt{ext}}$ and after pp is given. This is the same setting as that by Quach et al. [QWZ18]. We can consider the fully adaptive setting, where $\mathcal{A}$ can send the target message m after it accesses to the oracle $O_{\mathtt{ext}}$, as Kim and Wu [KW19]. However, our privately extractable watermarking PRF satisfies only selective security. Thus, we write only the selective variant for the private extraction case.

Definition 3.5 (Unremovability for Public Extraction). *This is the same as Definition 3.4 except we use the game* $\mathsf{Exp}^{\mathsf{pub\text{-}ext\text{-}nrmv}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda, \epsilon)$ *defined in the same way as* $\mathsf{Expt}^{\mathsf{nrmv}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda, \epsilon)$ *except the following differences.*

- *In item 1,* $\mathcal{A}$ *is given* xk *together with* pp.
- *Item 2 is removed.*

[7] In the watermarking setting, an extraction algorithm can take the description of a pirate circuit as input (corresponding to the software decoder model [Zha20, Section 4.2]), unlike the black-box tracing model of traitor tracing. However, we use a pirate circuit in the black box way for our extraction algorithms. Thus, we follow the black box projection model by Zhandry [Zha20].

4 Definition of Extraction-Less Watermarking

We introduce the notion of extraction-less watermarking PRF as an intermediate primitive towards watermarking PRFs secure against quantum adversaries.

4.1 Syntax and Pseudorandomness

Definition 4.1 (Extraction-Less Watermarking PRF). *An extraction-less watermarking PRF* WMPRF *for the message space* $\{0,1\}^{\ell_{\mathsf{m}}}$ *with domain* Dom *and range* Ran *is a tuple of five algorithms* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}, \mathsf{Sim})$*, where the first four algorithms have the same input/output behavior as those defined in Definition 3.1 and* Sim *has the following input/output behavior.*

$\mathsf{Sim}(\mathsf{xk}, \tau, i) \rightarrow (\gamma, x, y)$**:** *The simulation algorithm* Sim *takes as input the extraction key* xk*, a tag* τ*, and an index* i*, and outputs a tuple* (γ, x, y)*.*

Evaluation Correctness: *It is defined in exactly the same way as the evaluation correctness for watermarking PRF defined in Definition 3.1.*

Extended pseudorandomness. Extended pseudorandomness for extraction-less watermarking PRF is defined in exactly the same way as that for watermarking PRF, that is Definition 3.2.

4.2 Simulatability for Mark-Dependent Distributions (SIM-MDD Security)

We introduce the security notion for extraction-less watermarking PRF that we call simulatability for mark-dependent distributions. Let D and $D^{\texttt{rev}}$ be the following distributions.

D**:** Generate $b \leftarrow \{0,1\}$, $x \leftarrow \mathsf{Dom}$, and $y_0 \leftarrow \mathsf{Ran}$. Compute $y_1 \leftarrow \mathsf{Eval}(\mathsf{prfk}, x)$. Output (b, x, y_b).

$D^{\texttt{rev}}$**:** Generate $(b, x, y) \leftarrow D$. Output $(1 \oplus b, x, y)$.

Namely, D is the distribution that outputs a random value if the first bit $b = 0$ and a PRF evaluation if the first bit $b = 1$, and $D^{\texttt{rev}}$ is its opposite (i.e., a PRF evaluation if $b = 0$ and a random value if $b = 1$). SIM-MDD security is a security notion that guarantees that an adversary given $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{mk}, \mathsf{prfk}, \mathsf{m})$ cannot distinguish an output of $\mathsf{Sim}(\mathsf{xk}, \tau, i)$ from that of D if $\mathsf{m}[i] = 0$ and from that of $D^{\texttt{rev}}$ if $\mathsf{m}[i] = 1$.

Definition 4.2 (SIM-MDD Security with Private Simulation). *To define SIM-MDD security with private simulation, we define the game* $\mathsf{Expt}^{\mathsf{sim\text{-}mdd}}_{i^*, \mathcal{A}, \mathsf{WMPRF}}(\lambda)$ *as follows, where* $i^* \in [\ell_{\mathsf{m}}]$*.*

1. *The challenger generates* $(\mathsf{pp}, \mathsf{xk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* pp *to* $\mathcal{A}$*.* $\mathcal{A}$ *sends* $\mathsf{m} \in \{0,1\}^{\ell_{\mathsf{m}}}$ *to the challenger. The challenger generates* $(\mathsf{prfk}, \tau) \leftarrow \mathsf{Gen}(\mathsf{pp})$ *and computes* $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{mk}, \mathsf{prfk}, \mathsf{m})$*. The challenger sends* τ *and* $\widetilde{C}$ *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *can access to the following oracle.*

$O_{\mathtt{sim}}$: *On input* τ' *and* $i' \in [\ell_{\mathsf{m}}]$, *it returns* $\mathsf{Sim}(\mathsf{xk}, \tau', i')$.

3. *Let* $D_{\mathtt{real},i^*}$ *be the following distribution. Note that* $D_{\mathtt{real},i^*}$ *is identical with* D *if* $\mathsf{m}[i^*] = 0$ *and with* $D^{\mathtt{rev}}$ *if* $\mathsf{m}[i^*] = 1$.

   $D_{\mathtt{real},i^*}$: *Generate* $\gamma \leftarrow \{0,1\}$ *and* $x \leftarrow \mathsf{Dom}$. *Then, if* $\gamma = \mathsf{m}[i^*]$, *generate* $y \leftarrow \mathsf{Ran}$, *and otherwise, compute* $y \leftarrow \mathsf{Eval}(\mathsf{prfk}, x)$. *Output* (γ, x, y).

   The challenger generates $\mathsf{coin} \leftarrow \{0,1\}$. *If* $\mathsf{coin} = 0$, *the challenger samples* $(\gamma, x, y) \leftarrow D_{\mathtt{real},i^*}$. *If* $\mathsf{coin} = 1$, *the challenger generates* $(\gamma, x, y) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$. *The challenger sends* (γ, x, y) *to* $\mathcal{A}$.
4. *When* $\mathcal{A}$ *terminates with output* coin', *the challenger outputs* 1 *if* $\mathsf{coin} = \mathsf{coin}'$ *and* 0 *otherwise.*

Note that $\mathcal{A}$ *is not allowed to access to* $O_{\mathtt{sim}}$ *after* $\mathcal{A}$ *is given* (γ, x, y).

We say that WMPRF *is SIM-MDD secure if for every* $i^* \in [\ell_{\mathsf{m}}]$ *and QPT* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathsf{sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda) = 2\left|\Pr\left[\mathsf{Expt}^{\mathsf{sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda) = 1\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

We consider the selective setting above as unremovability for private extraction in Definition 3.4 since we use SIM-MDD security with private simulation to achieve unremovability for private simulation.

Remark 4.1 (On multi challenge security). We can prove that the above definition implies the multi-challenge variant where polynomially many outputs of $\mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ are required to be indistinguishable from those of $D_{\mathtt{real},i^*}$. This is done by hybrid arguments where outputs of $\mathsf{Sim}(\mathsf{xk}, \tau, i^*)$ are simulated using $O_{\mathtt{sim}}$ and those of $D_{\mathtt{real},i^*}$ are simulated using $\widetilde{C}$. To apply Theorem 2.3, we need the multi challenge variant. However, we consider the single challenge variant due to the implication above. A similar remark is applied to the variants of SIM-MDD security introduced below.

SIM-MDD security with private simulation under the $\mathcal{API}$ *oracle.* Let the $\mathcal{API}$ oracle be an oracle that is given $(\epsilon, \delta, \tau', i')$ and a quantum state q, and returns the result of $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},\mathsf{D}_{\tau',i'}}(q)$ and the post measurement state, where $\mathcal{P}$ is defined in the same way as that in Definition 3.4 and $\mathsf{D}_{\tau',i'}$ be the distribution that outputs randomly generated $(\gamma, x, y) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau', i')$. The $\mathcal{API}$ oracle cannot be simulated using the simulation oracle $O_{\mathtt{sim}}$ since we need superposition of outputs of Sim to compute $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},\mathsf{D}_{\tau',i'}}(q)$. When constructing watermarking PRFs with private simulation from extraction-less watermarking PRFs, the underlying extraction-less watermarking PRF scheme needs to satisfy SIM-MDD security with private simulation under the $\mathcal{API}$ oracle that we call QSIM-MDD security with private simulation. The reason is as follows. In the security analysis of the construction, the indistinguishability guarantee provided by SIM-MDD security needs to hold for an adversary against the resulting watermarking scheme who can access the extraction oracle. This means that it also needs to hold for an adversary who can access the $\mathcal{API}$ oracle since $\mathcal{API}$ is repeatedly invoked in the extraction algorithm of the resulting scheme.

Fortunately, as we will see, we can generically convert an extraction-less watermarking PRF scheme satisfying SIM-MDD security with private simulation into one satisfying QSIM-MDD security with private simulation, using QPRFs. Thus, when realizing an extraction-less watermarking PRF scheme as an intermediate step towards privately extractable watermarking PRFs, we can concentrate on realizing one satisfying SIM-MDD security with private simulation.

Remark 4.2. There is a similar issue in the traitor tracing setting. If PLBE is a secret-key based one, we need a counterpart of QSIM-MDD in secret-key based PLBE to achieve traitor tracing with a secret tracing algorithm against quantum adversaries by using Zhandry's framework [Zha20]. Note that Zhandry focuses on public-key based PLBE in his work [Zha20].

Definition 4.3 (QSIM-MDD Security with Private Simulation). *Let* $\mathsf{D}_{\tau,i}$ *be a distribution defined as follows.*

$\mathsf{D}_{\tau,i}$**:** *Output* $(\gamma, x, y) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i)$.

Then, we define the game $\mathsf{Exp}^{\mathsf{q\text{-}sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda)$ *in the same way as* $\mathsf{Exp}^{\mathsf{sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda)$ *except that in addition to* $O_{\mathtt{sim}}$*,* $\mathcal{A}$ *can access to the following oracle in the step 2.*

$O_{\mathtt{api}}$**:** *On input* $(\epsilon, \delta, \tau', i')$ *and a quantum state* q*, it returns the result of* $\mathcal{API}^{\epsilon,\delta}_{\mathcal{P},\mathsf{D}_{\tau',i'}}(q)$ *and the post measurement state, where* $\mathcal{P}$ *is defined in the same way as that in Definition 3.4.*

We say that WMPRF *is QSIM-MDD secure with private simulation if for every* $i^* \in [\ell_{\mathsf{m}}]$ *and QPT* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{q\text{-}sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda) = 2\left|\Pr\left[\mathsf{Exp}^{\mathsf{q\text{-}sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda) = 1\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

We have the following theorem.

Theorem 4.1. *Assume there exists an extraction-less watermarking PRF scheme satisfying SIM-MDD security with private simulation and a QPRF. Then, there exists an extraction-less watermarking PRF scheme satisfying QSIM-MDD security with private simulation.*

We prove this theorem in the full version.

Definition 4.4 (SIM-MDD Security with Public Simulation). *We define the game* $\mathsf{Exp}^{\mathsf{sim\text{-}mdd\text{-}pub}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda)$ *in the same way as* $\mathsf{Expt}^{\mathsf{sim\text{-}mdd}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda)$ *except the following differences, where* $i^* \in [\ell_{\mathsf{m}}]$.

– *In item 1,* $\mathcal{A}$ *is given* xk *together with* pp.
– *Item 2 is removed.*

We say that WMPRF *satisfies SIM-MDD security with public simulation if for every* $i^* \in [\ell_{\mathsf{m}}]$ *and QPT* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{sim\text{-}mdd\text{-}pub}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda) = 2\left|\Pr\left[\mathsf{Exp}^{\mathsf{sim\text{-}mdd\text{-}pub}}_{i^*,\mathcal{A},\mathsf{WMPRF}}(\lambda) = 1\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

5 Watermarking PRF from Extraction-Less Watermarking PRF

We show how to construct watermarking PRF secure against quantum adversaries from extraction-less watermarking PRF.

Let $\mathsf{ELWMPRF} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Eval}, \mathsf{Mark}, \mathsf{Sim})$ be an extraction-less watermarking PRF scheme whose message space is $\{0,1\}^{\ell_{\mathsf{m}}+1}$. We construct a watermarking PRF scheme $\mathsf{WMPRF} = (\mathsf{WM.Setup}, \mathsf{WM.Gen}, \mathsf{WM.Eval}, \mathsf{WM.Mark}, \mathcal{E}xtract)$ whose message space is $\{0,1\}^{\ell_{\mathsf{m}}}$ as follows. We use Setup, Gen, and Eval as $\mathsf{WM.Setup}$, $\mathsf{WM.Gen}$, and $\mathsf{WM.Eval}$, respectively. Thus, the domain and range of WMPRF are the same as those of $\mathsf{ELWMPRF}$. Also, we construct $\mathsf{WM.Mark}$ and $\mathcal{E}xtract$ as follows.

$\mathsf{WM.Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$**:**

- Output $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m}\|0)$.

$\mathcal{E}xtract(\mathsf{xk}, \mathcal{C}, \tau, \epsilon)$**:**

- Let $\epsilon' = \epsilon/4(\ell_{\mathsf{m}}+1)$ and $\delta' = 2^{-\lambda}$.
- Parse $(q, \boldsymbol{U}) \leftarrow \mathcal{C}$.
- Let $\mathcal{P}$ be defined in the same way as that in Definition 3.4 and $D_{\tau,i}$ be the following distribution for every $i \in [\ell_{\mathsf{m}}+1]$.

  $D_{\tau,i}$**:** Output $(\gamma, x, y) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i)$.
- Compute $\widetilde{p}_{\ell_{\mathsf{m}}+1} \leftarrow \mathcal{API}^{\epsilon',\delta'}_{\mathcal{P},D_{\tau,\ell_{\mathsf{m}}+1}}(q)$. If $\widetilde{p}_{\ell_{\mathsf{m}}+1} < \frac{1}{2}+\epsilon-4\epsilon'$, return $\mathsf{unmarked}$. Otherwise, letting q_0 be the post-measurement state, go to the next step.
- For all $i \in [\ell_{\mathsf{m}}]$, do the following.
  1. Compute $\widetilde{p}_i \leftarrow \mathcal{API}^{\epsilon',\delta'}_{\mathcal{P},D_{\tau,i}}(q_{i-1})$. Let q_i be the post-measurement state.
  2. If $\widetilde{p}_i > \frac{1}{2}+\epsilon-4(i+1)\epsilon'$, set $\mathsf{m}'_i = 0$. If $\widetilde{p}_i < \frac{1}{2}-\epsilon+4(i+1)\epsilon'$, set $\mathsf{m}'_i = 1$. Otherwise, exit the loop and output $\mathsf{m}' = 0^{\ell_{\mathsf{m}}}$.
- Output $\mathsf{m}' = \mathsf{m}'_1\|\cdots\|\mathsf{m}'_{\ell_{\mathsf{m}}}$.

We have the following theorems.

Theorem 5.1. *If* $\mathsf{ELWMPRF}$ *satisfies extended weak pseudorandomness against authority, then so does* WMPRF.

Theorem 5.2. *If* $\mathsf{ELWMPRF}$ *is an extraction-less watermarking PRF that satisfies QSIM-MDD security,* WMPRF *is a privately extractable watermarking PRF.*

Theorem 5.3. *If* $\mathsf{ELWMPRF}$ *is an extraction-less watermarking PRF that satisfies SIM-MDD security with public simulation,* WMPRF *is a publicly extractable watermarking PRF.*

It is clear that Theorem 5.1 holds since the evaluation algorithm of WMPRF is the same as that of $\mathsf{ELWMPRF}$ and extended weak pseudorandomness is insensitive to how the marking and extraction algorithms are defined. Thus, we omit a formal proof.

The proofs of Theorems 5.2 and 5.3 are almost the same. Thus, we only provide the proof for the former, and omit the proof for the latter.

Proof of Theorem 5.2. Let $\epsilon > 0$. Let $\mathcal{A}$ be a QPT adversary attacking the unremovability of WMPRF. The description of $\mathsf{Expt}^{\mathsf{nrmv}}_{\mathcal{A},\mathsf{WMPRF}}(\lambda, \epsilon)$ is as follows.

1. The challenger generates $(\mathsf{pp}, \mathsf{xk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and gives pp to the adversary $\mathcal{A}$. $\mathcal{A}$ sends $\mathsf{m} \in \{0,1\}^{\ell_\mathsf{m}}$ to the challenger. The challenger generates $(\mathsf{prfk}, \tau) \leftarrow \mathsf{Gen}(\mathsf{pp})$, computes $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m}\|0)$, and sends $\widetilde{C}$ to $\mathcal{A}$.
2. $\mathcal{A}$ can access to the following oracle.

   $O_{\mathtt{ext}}$: On input τ' and a quantum circuit $\mathcal{C}$, it returns $\mathcal{Extract}(\mathsf{xk}, \mathcal{C}, \tau', \epsilon)$.
3. Finally, the adversary outputs a quantum circuit $\mathcal{C}_{☠} = (q, \boldsymbol{U})$.

We define D, $\mathcal{P}$, $\mathcal{M}_D$, and the three events Live, $\mathsf{GoodExt}$, and BadExt in the same way as Definition 3.4.

The proof of $\Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$. $\mathcal{Extract}$ outputs $\mathsf{unmarked}$ if and only if $\widetilde{p}_{\ell+1} < \frac{1}{2} + \epsilon - 4\epsilon'$, that is we have $\Pr[\mathsf{GoodExt}] = \Pr\left[\widetilde{p}_{\ell+1} \geq \frac{1}{2} + \epsilon - 4\epsilon'\right]$. Let p the probability obtained by applying $\mathsf{ProjImp}(\mathcal{M}_D)$ to q. Then, we have $\Pr[\mathsf{Live}] = \Pr\left[p \geq \frac{1}{2} + \epsilon\right]$. Let $\widetilde{p}$ be the outcome obtained if we apply $\mathcal{API}^{\epsilon',\delta'}_{\mathcal{P},D}$ to q. From the property of $\mathcal{API}$, we have

$$\Pr[\mathsf{Live}] = \Pr\left[p \geq \frac{1}{2} + \epsilon\right] \leq \Pr\left[\widetilde{p} \geq \frac{1}{2} + \epsilon - \epsilon'\right] + \mathsf{negl}(\lambda).$$

D and $D_{\tau,\ell_\mathsf{m}+1}$ are computationally indistinguishable from the QSIM-MDD security of $\mathsf{ELWMPRF}$ since outputs of $\mathsf{Sim}(\mathsf{xk}, \tau, i)$ is indistinguishable from those of D if $\mathsf{m}[i] = 0$. This indistinguishability holds even under the existence of $O_{\mathtt{api}}$. Then, from Theorem 2.3, we have

$$\Pr\left[\widetilde{p} \geq \frac{1}{2} + \epsilon - \epsilon'\right] \leq \Pr\left[\widetilde{p}_{\ell+1} \geq \frac{1}{2} + \epsilon - 4\epsilon'\right] + \mathsf{negl}(\lambda) = \Pr[\mathsf{GoodExt}] + \mathsf{negl}(\lambda).$$

By combining the above two equations, we obtain $\Pr[\mathsf{GoodExt}] \geq \Pr[\mathsf{Live}] - \mathsf{negl}(\lambda)$.

The reason D and $D_{\tau,\ell+1}$ need to be computationally indistinguishable under the existence of $O_{\mathtt{api}}$ to apply Theorem 2.3 is as follows. In this application of Theorem 2.3, the quantum state appeared in the statement of it is set as q contained in the quantum circuit $\mathcal{C}$ output by $\mathcal{A}$. Then, Theorem 2.3 (implicitly) requires that D and $D_{\tau,\ell+1}$ be indistinguishable for distinguishers who can construct q. To construct q, we need to execute $\mathcal{A}$ who can access to $O_{\mathtt{ext}}$ in which $\mathcal{API}$ is repeatedly executed. This is the reason D and $D_{\tau,\ell+1}$ need to be indistinguishable under the existence of $O_{\mathtt{api}}$.

The proof of $\Pr[\mathsf{BadExt}] \leq \mathsf{negl}(\lambda)$. We define the event BadExt_i as follows for every $i \in [\ell_\mathsf{m}]$.

BadExt_i: When Running $\mathcal{Extract}(\mathsf{xk}, \mathcal{C}_{☠}, \tau^*, \epsilon)$, the following conditions hold.

- $\widetilde{p}_{\ell+1} \geq \frac{1}{2} + \epsilon - 4\epsilon'$ holds.
- $\mathsf{m}'_j = \mathsf{m}_j$ holds for every $j \in [i-1]$.
- $\mathcal{Extract}$ exits the i-th loop or $\mathsf{m}'_i \neq \mathsf{m}_i$ holds.

Then, we have $\Pr[\mathsf{BadExt}] \leq \sum_{i\in[\ell]} \Pr[\mathsf{BadExt}_i]$. Below, we estimate $\Pr[\mathsf{BadExt}_i]$.

We first consider the case of $\mathsf{m}_{i-1} = 0$ and $\mathsf{m}_i = 0$. Assume $\mathsf{m}'_{i-1} = \mathsf{m}_{i-1} = 0$ holds. Then, we have $\widetilde{p}_{i-1} > \frac{1}{2} + \epsilon - 4i\epsilon'$. Let $\widetilde{p}'_{i-1} \leftarrow \mathcal{API}^{\epsilon',\delta'}_{\mathcal{P},D_{\tau,i-1}}(q_{i-1})$. From, the almost-projective property of $\mathcal{API}$, we have

$$\Pr\left[\widetilde{p}'_{i-1} > \frac{1}{2} + \epsilon - 4i\epsilon' - \epsilon'\right] \geq 1 - \delta'.$$

When $\mathsf{m}_{i-1} = 0$ and $\mathsf{m}_i = 0$, $D_{\tau,i-1}$ and $D_{\tau,i}$ are computationally indistinguishable since both of them are computationally indistinguishable from D by the QSIM-MDD security of $\mathsf{ELWMPRF}$. This indistinguishability holds under the existence of $O_{\mathtt{api}}$. Thus, from Theorem 2.3, we have

$$1 - \delta' \leq \Pr\left[\widetilde{p}'_{i-1} > \frac{1}{2} + \epsilon - (4i+1)\epsilon'\right] \leq \Pr\left[\widetilde{p}_i > \frac{1}{2} + \epsilon - 4(i+1)\epsilon'\right] + \mathsf{negl}(\lambda).$$

This means that $\Pr[\mathsf{BadExt}_i] = \mathsf{negl}(\lambda)$ in this case. Note that the reason the indistinguishability of $D_{\tau,i-1}$ and $D_{\tau,i}$ needs to hold under $O_{\mathtt{api}}$ is that Theorem 2.3 requires it hold for distinguishers who can construct q_{i-1}.

Next, we consider the case of $\mathsf{m}_{i-1} = 0$ and $\mathsf{m}_i = 1$. Assume $\mathsf{m}'_{i-1} = \mathsf{m}_{i-1} = 0$ holds. Then, we have $\widetilde{p}_{i-1} > \frac{1}{2} + \epsilon - 4i\epsilon'$. We then define an additional distribution $D^{\mathtt{rev}}_{\tau,i}$ as follows.

$D^{\mathtt{rev}}_{\tau,i}$: Generate $(\gamma, x, y) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i)$. Output $(1 \oplus \gamma, x, y)$.

That is, the first bit of the output is flipped from $\mathsf{D}_{\tau,i}$. Then, for any random coin r, we have $(\boldsymbol{P}_{D^{\mathtt{rev}}_{\tau,i}(r)}, \boldsymbol{Q}_{D^{\mathtt{rev}}_{\tau,i}(r)}) = (\boldsymbol{Q}_{\mathsf{D}_{\tau,i}(r)}, \boldsymbol{P}_{\mathsf{D}_{\tau,i}(r)})$. This is because we have $\boldsymbol{Q}_{b,x,y} = \boldsymbol{I} - \boldsymbol{P}_{b,x,y} = \boldsymbol{P}_{1\oplus b,x,y}$ for any tuple (b, x, y). Therefore, $\mathcal{API}^{\epsilon',\delta'}_{\mathcal{P},D^{\mathtt{rev}}_{\tau,i-1}}$ is exactly the same process as $\mathcal{API}^{\epsilon',\delta'}_{\mathcal{P}^{\mathtt{rev}},\mathsf{D}_{\tau,i-1}}$. Let $\widetilde{p}'_{i-1} \leftarrow \mathcal{API}^{\epsilon',\delta'}_{\mathcal{P},D^{\mathtt{rev}}_{\tau,i-1}}(q_{i-1})$. From, the reverse-almost-projective property of $\mathcal{API}$, we have

$$\Pr\left[\widetilde{p}'_{i-1} < \frac{1}{2} - \epsilon + 4i\epsilon' + \epsilon'\right] \geq 1 - \delta'.$$

When $\mathsf{m}_{i-1} = 0$ and $\mathsf{m}_i = 1$, $D^{\mathtt{rev}}_{\tau,i-1}$ and $D_{\tau,i}$ are computationally indistinguishable since both of them are computationally indistinguishable from the following distribution $D^{\mathtt{rev}}$ by the QSIM-MDD security of $\mathsf{ELWMPRF}$.

$D^{\mathtt{rev}}$: Generate $(\gamma, x, y) \leftarrow D$. Output $(1 \oplus \gamma, x, y)$.

This indistinguishability holds under the existence of $O_{\mathtt{api}}$. Thus, from Theorem 2.3, we have

$$1 - \delta' \leq \Pr\left[\widetilde{p}'_{i-1} < \frac{1}{2} - \epsilon + (4i+1)\epsilon'\right] \leq \Pr\left[\widetilde{p}_i < \frac{1}{2} - \epsilon + 4(i+1)\epsilon'\right] + \mathsf{negl}(\lambda).$$

This means that $\Pr[\mathsf{BadExt}_i] = \mathsf{negl}(\lambda)$ also in this case. Note that the reason the indistinguishability of $D^{\mathtt{rev}}_{\tau,i-1}$ and $D_{\tau,i}$ needs to hold under $O_{\mathtt{api}}$ is that Theorem 2.3 requires it hold for distinguishers who can construct q_{i-1}.

Similarly, we can prove that $\Pr[\mathsf{BadExt}_i] = \mathsf{negl}(\lambda)$ holds in the case of $(\mathsf{m}_{i-1}, \mathsf{m}_i) = (1, 0)$ and $(\mathsf{m}_{i-1}, \mathsf{m}_i) = (1, 1)$.

Overall, we see that $\Pr[\mathsf{BadExt}] = \mathsf{negl}(\lambda)$ holds in all cases. ■

6 Extraction-Less Watermarking PRF from LWE

We present an extraction-less watermarking PRF, denoted by $\mathsf{PRF}_{\mathsf{cprf}}$, whose message space is $\{0,1\}^{\ell_{\mathsf{m}}}$ with domain $\{0,1\}^n$ and range $\{0,1\}^m$. We use the following tools, which can be instantiated with the QLWE assumption:

- Private CPRF $\mathsf{CPRF} = (\mathsf{CPRF.Setup}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.CEval})$. For ease of notation, we denote CPRF evaluation circuit $\mathsf{CPRF.Eval}(\mathsf{msk}, \cdot)$ and constrained evaluation circuits $\mathsf{CPRF.CEval}(\mathsf{sk}_f, \cdot)$ by $\mathsf{G} : \{0,1\}^n \to \{0,1\}^m$ and $\mathsf{G}_{\notin \mathcal{V}} : \{0,1\}^n \to \{0,1\}^m$, respectively, where $x \in \mathcal{V}$ iff $f(x) = 1$.
- SKE scheme $\mathsf{SKE} = (\mathsf{SKE.Gen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$. The plaintext space and ciphertext space of SKE are $\{0,1\}^{\ell_{\mathsf{ske}}}$ and $\{0,1\}^n$, respectively, where $\ell_{\mathsf{ske}} = \log \ell_{\mathsf{m}} + 1$.
- PKE scheme $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. The plaintext space of PKE is $\{0,1\}^{2\lambda}$.

Construction overview. We already explained the high-level idea for how to realize extraction-less watermarking PRFs in Section 1.3. However, the construction of $\mathsf{PRF}_{\mathsf{cprf}}$ requires some additional efforts. Thus, before providing the actual construction, we provide a high-level overview of $\mathsf{PRF}_{\mathsf{cprf}}$.

Recall that letting $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$ and $(\gamma^*, x^*, y^*) \leftarrow \mathsf{Sim}(\mathsf{xk}, \tau, i^*)$, we have to design Sim and $\widetilde{C}$ so that

- If $\gamma = \mathsf{m}[i^*]$, $\widetilde{C}(x^*)$ outputs a value different from y^*.
- If $\gamma \neq \mathsf{m}[i^*]$, $\widetilde{C}(x^*)$ outputs y^*.

In the token-based construction idea, we achieve these conditions by setting x^* as an encryption of $y^* \| i^* \| \gamma^*$ and designing $\widetilde{C}$ as a token such that it outputs y^* if the input is decryptable and $\gamma^* \neq \mathsf{m}[i^*]$ holds for the decrypted value $y^* \| i^* \| \gamma^*$, and otherwise behaves as the original evaluation circuit. However, in $\mathsf{PRF}_{\mathsf{cprf}}$, we use a constrained evaluation circuit of CPRF as $\widetilde{C}$, and thus we cannot program output values for specific inputs. Intuitively, it seems that Sim needs to use the original PRF key prfk to achieve the above two conditions.

To solve the issue, we adopt the idea used by Quach et al. [QWZ18]. In $\mathsf{PRF}_{\mathsf{cprf}}$, the setup algorithm Setup generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ of PKE, and sets $\mathsf{pp} = \mathsf{pk}$ and $\mathsf{xk} = \mathsf{sk}$. Then, the PRF key generation algorithm is given pk, generates $\mathsf{G} \leftarrow \mathsf{CPRF.Setup}(1^\lambda, 1^\kappa)$ along with $\mathsf{ske.k} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$, and sets the public tag τ as an encryption of $(\mathsf{G}, \mathsf{ske.k})$ under pk. The evaluation algorithm of $\mathsf{PRF}_{\mathsf{cprf}}$ is simply that of CPRF.

Now, we explain how to design Sim and $\widetilde{C} \leftarrow \mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$ to satisfy the above two conditions. Given $\mathsf{xk} = \mathsf{sk}$, $\tau = \mathsf{Enc}(\mathsf{pk}, \mathsf{prfk})$ and i, Sim is able to extract $\mathsf{prfk} = (\mathsf{G}, \mathsf{ske.k})$. Then, Sim generates $\gamma \leftarrow \{0,1\}$ and sets $x \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.k}, i \| \gamma)$ and $y \leftarrow \mathsf{G}(x)$. We set $\widetilde{C}$ as a constrained version of G for a circuit D that outputs 1 if the input x is decryptable by $\mathsf{ske.k}$ and $\gamma = \mathsf{m}[i]$ holds for decrypted value $i \| \gamma$, and otherwise outputs 0. For an input x, the constrained version of G outputs the correct output $\mathsf{G}(x)$ if and only if $D(x) = 0$. We can check that $\mathsf{PRF}_{\mathsf{cprf}}$ satisfies the above two conditions.

The above construction does not satisfy extended weak pseudorandomness against authority since the authority can extract the original CPRF key G by

$\mathsf{xk} = \mathsf{sk}$. However, this problem can be fixed by constraining G. We see that Sim needs to evaluate G for valid ciphertexts of SKE. Thus, to implement the above mechanism, it is sufficient to set the public tag τ as an encryption of $\mathsf{ske.k}$ and a constrained version of G for a circuit $D_{\mathtt{auth}}$ that output 0 if and only if the input is decryptable by $\mathsf{ske.k}$. Then, the authority can only extract such a constrained key. By requiring sparseness for SKE, the constrained key cannot be used to break the pseudorandomness of $\mathsf{PRF}_{\mathsf{cprf}}$ for random inputs. This means that $\mathsf{PRF}_{\mathsf{cprf}}$ satisfies extended weak pseudorandomness against an authority. Note that we only need a single-key CPRF for $\mathsf{PRF}_{\mathsf{cprf}}$ since either a user or the authority (not both) is a malicious entity in security games.

The description of $\mathsf{PRF}_{\mathsf{cprf}}$ is as follows.

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

$\mathsf{Gen}(\mathsf{pp})$:
- Parse $\mathsf{pp} = \mathsf{pk}$.
- Generate $\mathsf{G} \leftarrow \mathsf{CPRF.Setup}(1^\lambda, 1^\kappa)$. In our construction, κ is the size of circuit $D[\mathsf{ske.k}, \mathsf{m}]$ described in Figure 2, which depends on ℓ_{m} (and λ).
- Generate $\mathsf{ske.k} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$.
- Construct a circuit $D_{\mathtt{auth}}[\mathsf{ske.k}]$ described in Figure 1.
- Compute $\mathsf{G}_{\notin \mathcal{V}_{\mathtt{auth}}} := \mathsf{CPRF.Constrain}(\mathsf{G}, D_{\mathtt{auth}}[\mathsf{ske.k}])$, where $\mathcal{V}_{\mathtt{auth}} \subset \{0,1\}^n$ is a set such that $x \in \mathcal{V}_{\mathtt{auth}}$ iff $D_{\mathtt{auth}}[\mathsf{ske.k}](x) = 1$.
- Output $\mathsf{prfk} := (\mathsf{G}, \mathsf{ske.k})$ and $\tau \leftarrow \mathsf{Enc}(\mathsf{pk}, (\mathsf{G}_{\notin \mathcal{V}_{\mathtt{auth}}}, \mathsf{ske.k}))$.

$\mathsf{Eval}(\mathsf{prfk}, x \in \{0,1\}^n)$: Recall that G is a keyed CPRF evaluation circuit.
- Parse $\mathsf{prfk} = (\mathsf{G}, \mathsf{ske.k})$.
- Output $y := \mathsf{G}(x)$.

$\mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m})$:
- Parse $\mathsf{pp} = \mathsf{pk}$ and $\mathsf{prfk} = (\mathsf{G}, \mathsf{ske.k})$.
- Construct a circuit $D[\mathsf{ske.k}, \mathsf{m}]$ described in Figure 2.
- Compute $\mathsf{G}_{\notin \mathcal{V}} \leftarrow \mathsf{CPRF.Constrain}(\mathsf{G}, D[\mathsf{ske.k}, \mathsf{m}])$, where $\mathcal{V} \subset \{0,1\}^n$ is a set such that $x \in \mathcal{V}$ iff $D[\mathsf{ske.k}, \mathsf{m}](x) = 1$.
- Output $\widetilde{C} = \mathsf{G}_{\notin \mathcal{V}}$.

$\mathsf{Sim}(\mathsf{xk}, \tau, i)$:
- Parse $\mathsf{xk} = \mathsf{sk}$.
- Compute $(\mathsf{G}_{\notin \mathcal{V}_{\mathtt{auth}}}, \mathsf{ske.k}) \leftarrow \mathsf{Dec}(\mathsf{sk}, \tau)$.
- Choose $\gamma \leftarrow \{0,1\}$.
- Compute $x \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.k}, i \| \gamma)$ and $y \leftarrow \mathsf{G}_{\notin \mathcal{V}_{\mathtt{auth}}}(x)$.
- Output (γ, x, y).

The evaluation correctness of $\mathsf{PRF}_{\mathsf{cprf}}$ follows from the sparseness of SKE and the correctness of CPRF. For the security of $\mathsf{PRF}_{\mathsf{cprf}}$, we have the following theorems.

Theorem 6.1. SKE *is a secure SKE scheme with pseudorandom ciphertext,* CPRF *is a selectively single-key private CPRF,* PKE *is a CCA secure PKE scheme, then* $\mathsf{PRF}_{\mathsf{cprf}}$ *is an extraction-less watermarking PRF satisfying SIM-MDD security.*

```
Circuit $D_{\mathtt{auth}}[\mathsf{ske.k}]$
Constants: An SKE key $\mathsf{ske.k}$, and a message $\mathsf{m}$.
Input: A string $x \in \{0,1\}^n$.
  1. Compute $d \leftarrow \mathsf{SKE.Dec}(\mathsf{ske.k}, x)$.
  2. Output 0 if $d \neq \bot$ and 1 otherwise.
```

Fig. 1: The description of $D_{\mathtt{auth}}$

```
Circuit $D[\mathsf{ske.k}, \mathsf{m}]$
Constants: An SKE key $\mathsf{ske.k}$, and a message $\mathsf{m}$.
Input: A string $x \in \{0,1\}^n$.
  1. Compute $d \leftarrow \mathsf{SKE.Dec}(\mathsf{ske.k}, x)$.
  2. If $d \neq \bot$, do the following
     (a) Parse $d = i \| \gamma$, where $i \in [\ell_{\mathsf{m}}]$ and $\gamma \in \{0,1\}$.
     (b) If $\gamma = \mathsf{m}[i]$, output 1. Otherwise, output 0.
  3. Otherwise output 0.
```

Fig. 2: The description of D

Theorem 6.2. *If* CPRF *is a selective single-key private CPRF,* $\mathsf{PRF}_{\mathsf{cprf}}$ *satisfies extended weak pseudorandomness.*

We prove Theorems 6.1 and 6.2 in the full version.

7 Extraction-Less Watermarking PRF with Public Simulation from IO

We construct an extraction-less watermarking PRF satisfying SIM-MDD security with public simulation. In the construction, we use puncturable encryption (PE) [CHN+18]. We provide the definition of PE in the full version.

We describe our extraction-less watermarking PRF $\mathsf{PRF}_{\mathsf{io}}$ for message space $\{0,1\}^{\ell_{\mathsf{m}}}$ with domain $\{0,1\}^{\ell_{\mathsf{in}}}$ and range $\{0,1\}^{\ell_{\mathsf{out}}}$ below. We use the following tools:

- PPRF $\mathsf{PRF} = \mathsf{PRF}.(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Puncture})$. We denote a PRF evaluation circuit $\mathsf{PRF.Eval}_{\mathsf{prfk}}(\cdot)$ by $\mathsf{F} : \{0,1\}^{\ell_{\mathsf{in}}} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}}$, a PRF evaluation circuit with punctured key $\mathsf{PRF.Eval}_{\mathsf{prfk}_{\neq x}}(\cdot)$ by $\mathsf{F}_{\neq x}$ (that is, we omit prfk and simply write $\mathsf{F}(\cdot)$ instead of $\mathsf{F}_{\mathsf{prfk}}(\cdot)$) for ease of notations.
- PE scheme $\mathsf{PE} = \mathsf{PE}.(\mathsf{Gen}, \mathsf{Puncture}, \mathsf{Enc}, \mathsf{Dec})$. The plaintext and ciphertext space of PE are $\{0,1\}^{\ell_{\mathsf{pt}}}$ and $\{0,1\}^{\ell_{\mathsf{ct}}}$, respectively, where $\ell_{\mathsf{pt}} = \ell + \log \ell_{\mathsf{m}} + 1$ and $\ell_{\mathsf{in}} := \ell_{\mathsf{ct}}$ ($\ell_{\mathsf{ct}} = \mathrm{poly}(\ell, \log \ell_{\mathsf{m}})$).
- Indistinguishability obfuscator $i\mathcal{O}$.

```
Circuit $D[\mathsf{F}, \mathsf{pe.dk}, \mathsf{m}]$
Constants: A PRF $\mathsf{F}$, a PE decryption key $\mathsf{pe.dk}$, and a message $\mathsf{m}$.
Input: A string $x \in \{0,1\}^{\ell_{\mathsf{in}}}$.
  1. Compute $d \leftarrow \mathsf{PE.Dec}(\mathsf{pe.dk}, x)$.
  2. If $d \neq \bot$, do the following
     (a) Parse $d = s \| i \| \gamma$, where $s \in \{0,1\}^{\ell}$, $i \in [\ell_{\mathsf{m}}]$, and $\gamma \in \{0,1\}$.
     (b) If $\mathsf{m}[i] \neq \gamma$, output $\mathsf{PRG}(s)$. Otherwise, output $\mathsf{F}(x)$.
  3. Otherwise, output $\mathsf{F}(x)$.
```

Fig. 3: The description of D

– PRG $\mathsf{PRG} : \{0,1\}^{\ell} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}}$.

$\mathsf{Setup}(1^{\lambda})$**:**
- Output $(\mathsf{pp}, \mathsf{xk}) := (\bot, \bot)$.

$\mathsf{Gen}(\mathsf{pp})$**:**
- Parse $\mathsf{pp} = \bot$.
- Compute $\mathsf{F} \leftarrow \mathsf{PRF.Gen}(1^{\lambda})$.
- Generate $(\mathsf{pe.ek}, \mathsf{pe.dk}) \leftarrow \mathsf{PE.Gen}(1^{\lambda})$.
- Output $\mathsf{prfk} := (\mathsf{F}, \mathsf{pe.dk})$ and $\tau := \mathsf{pe.ek}$.

$\mathsf{Eval}(\mathsf{prfk}, x \in \{0,1\}^{\ell_{\mathsf{in}}})$**:**
- Parse $\mathsf{prfk} = (\mathsf{F}, \mathsf{pe.dk})$.
- Compute and output $y \leftarrow \mathsf{F}(x)$.

$\mathsf{Mark}(\mathsf{pp}, \mathsf{prfk}, \mathsf{m} \in \{0,1\}^{\ell_{\mathsf{m}}})$**:**
- Parse $\mathsf{pp} = \bot$ and $\mathsf{prfk} = (\mathsf{F}, \mathsf{pe.dk})$.
- Construct a circuit $D[\mathsf{F}, \mathsf{pe.dk}, \mathsf{m}]$ described in Figure 3.
- Compute and output $\widetilde{C} := i\mathcal{O}(D[\mathsf{F}, \mathsf{pe.dk}, \mathsf{m}])$.

$\mathsf{Sim}(\mathsf{xk}, \tau, i)$**:**
- Parse $\mathsf{xk} = \bot$ and $\tau = \mathsf{pe.ek}$.
- Choose $\gamma \leftarrow \{0,1\}$ and $s \leftarrow \{0,1\}^{\ell}$.
- Compute $y := \mathsf{PRG}(s)$.
- Compute $x \leftarrow \mathsf{PE.Enc}(\mathsf{pe.ek}, s \| i \| \gamma)$.
- Output (γ, x, y)

The size of the circuit D is appropriately padded to be the maximum size of all modified circuits, which will appear in the security proof.

The evaluation correctness of $\mathsf{PRF_{io}}$ immediately follows from the sparseness of PE and the functionality of $i\mathcal{O}$.[8] $\mathsf{PRF_{io}}$ trivially satisfies pseudorandomness (against an authority) since Setup outputs nothing, τ is a public key $\mathsf{pe.ek}$, and Eval is independent of $(\mathsf{pe.ek}, \mathsf{pe.dk})$ ($\mathsf{pe.dk}$ is not used in Eval). Moreover, we have the following theorem.

[8] In fact, $\mathsf{PRF_{io}}$ satisfies a stronger evaluation correctness than one written in Definition 4.1. The evaluation correctness holds even for any PRF key prfk and input $x \in \mathsf{Dom}$ like the statistical correctness by Cohen et al. [CHN+18].

Theorem 7.1. *If* PRF *is a secure PPRF,* PRG *is a secure PRG,* PE *is a secure PE with strong ciphertext pseudorandomness, and $i\mathcal{O}$ is a secure IO, then* $\mathsf{PRF}_{\mathsf{io}}$ *is an extraction-less watermarking PRF satisfying SIM-MDD security with public simulation.*

We prove Theorem 7.1 in the full version.

References

AL21. P. Ananth and R. L. La Placa. Secure Software Leasing. In *EUROCRYPT 2021, Part II*, pages 501–530. 2021.

ALL+21. S. Aaronson, J. Liu, Q. Liu, M. Zhandry, and R. Zhang. New Approaches for Quantum Copy-Protection. In *CRYPTO 2021, Part I*, pages 526–555, Virtual Event, 2021.

ARU14. A. Ambainis, A. Rosmanis, and D. Unruh. Quantum Attacks on Classical Proof Systems: The Hardness of Quantum Rewinding. In *55th FOCS*, pages 474–483. 2014.

BDF+11. D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random Oracles in a Quantum World. In *ASIACRYPT 2011*, pages 41–69. 2011.

BGI+12. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. *Journal of the ACM*, 59(2):6:1–6:48, 2012.

BLW17. D. Boneh, K. Lewi, and D. J. Wu. Constraining Pseudorandom Functions Privately. In *PKC 2017, Part II*, pages 494–524. 2017.

BSW06. D. Boneh, A. Sahai, and B. Waters. Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In *EUROCRYPT 2006*, pages 573–592. 2006.

CFN94. B. Chor, A. Fiat, and M. Naor. Tracing Traitors. In *CRYPTO'94*, pages 257–270. 1994.

CHN+18. A. Cohen, J. Holmgren, R. Nishimaki, V. Vaikuntanathan, and D. Wichs. Watermarking Cryptographic Capabilities. *SIAM Journal on Computing*, 47(6):2157–2202, 2018.

CMSZ21. A. Chiesa, F. Ma, N. Spooner, and M. Zhandry. Post-Quantum Succinct Arguments: Breaking the Quantum Rewinding Barrier. In *FOCS 2021*. 2021.

GKM+19. R. Goyal, S. Kim, N. Manohar, B. Waters, and D. J. Wu. Watermarking Public-Key Cryptographic Primitives. In *CRYPTO 2019, Part III*, pages 367–398. 2019.

GKWW21. R. Goyal, S. Kim, B. Waters, and D. J. Wu. Beyond Software Watermarking: Traitor-Tracing for Pseudorandom Functions. In *Asiacrypt 2021*, Lecture Notes in Computer Science. 2021.

HMW07. N. Hopper, D. Molnar, and D. Wagner. From Weak to Strong Watermarking. In *TCC 2007*, pages 362–382. 2007.

KNY21. F. Kitagawa, R. Nishimaki, and T. Yamakawa. Secure Software Leasing from Standard Assumptions. In *TCC 2021*, LNCS. 2021.

KW19. S. Kim and D. J. Wu. Watermarking PRFs from Lattices: Stronger Security via Extractable PRFs. In *CRYPTO 2019, Part III*, pages 335–366. 2019.

KW21. S. Kim and D. J. Wu. Watermarking Cryptographic Functionalities from Standard Lattice Assumptions. *J. Cryptol.*, 34(3):28, 2021.

Nis13. R. Nishimaki. How to Watermark Cryptographic Functions. In *EUROCRYPT 2013*, pages 111–125. 2013.

Nis19. R. Nishimaki. How to Watermark Cryptographic Functions by Bilinear Maps. *IEICE Transactions*, 102-A(1):99–113, 2019.

Nis20. R. Nishimaki. Equipping Public-Key Cryptographic Primitives with Watermarking (or: A Hole Is to Watermark). In *TCC 2020, Part I*, pages 179–209. 2020.

NSS99. D. Naccache, A. Shamir, and J. P. Stern. How to Copyright a Function? In *PKC'99*, pages 188–196. 1999.

QWZ18. W. Quach, D. Wichs, and G. Zirdelis. Watermarking PRFs Under Standard Assumptions: Public Marking and Security with Extraction Queries. In *TCC 2018, Part II*, pages 669–698. 2018.

Reg09. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):34:1–34:40, 2009.

Unr12. D. Unruh. Quantum Proofs of Knowledge. In *EUROCRYPT 2012*, pages 135–152. 2012.

Wat09. J. Watrous. Zero-Knowledge against Quantum Attacks. *SIAM J. Comput.*, 39(1):25–58, 2009.

YAL$^+$19. R. Yang, M. H. Au, J. Lai, Q. Xu, and Z. Yu. Collusion Resistant Watermarking Schemes for Cryptographic Functionalities. In *ASIACRYPT 2019, Part I*, pages 371–398. 2019.

YAYX20. R. Yang, M. H. Au, Z. Yu, and Q. Xu. Collusion Resistant Watermarkable PRFs from Standard Assumptions. In *CRYPTO 2020, Part I*, pages 590–620. 2020.

YF11. M. Yoshida and T. Fujiwara. Toward Digital Watermarking for Cryptographic Data. *IEICE Transactions*, 94-A(1):270–272, 2011.

Zha12a. M. Zhandry. How to Construct Quantum Random Functions. In *53rd FOCS*, pages 679–687. 2012.

Zha12b. M. Zhandry. Secure Identity-Based Encryption in the Quantum Random Oracle Model. In *CRYPTO 2012*, pages 758–775. 2012.

Zha19. M. Zhandry. How to Record Quantum Queries, and Applications to Quantum Indifferentiability. In *CRYPTO 2019, Part II*, pages 239–268. 2019.

Zha20. M. Zhandry. Schrödinger's Pirate: How to Trace a Quantum Decoder. In *TCC 2020, Part III*, pages 61–91. 2020.