Hidden Cosets and Applications to Unclonable Cryptography

Andrea Coladangelo[1], Jiahui Liu[2], Qipeng Liu[3], and Mark Zhandry[4]

[1]UC Berkeley, Simons Institute for the Theory of Computing & qBraid
[2]University of Texas at Austin
[3]Princeton University
[4]Princeton University & NTT Research

July 12, 2021

Abstract

In 2012, Aaronson and Christiano introduced the idea of *hidden subspace states* to build public-key quantum money [STOC '12]. Since then, this idea has been applied to realize several other cryptographic primitives which enjoy some form of unclonability.

In this work, we study a generalization of hidden subspace states to hidden *coset* states. This notion was considered independently by Vidick and Zhang [Eurocrypt '21], in the context of proofs of quantum knowledge from quantum money schemes. We explore unclonable properties of coset states and several applications:

- We show that, assuming indistinguishability obfuscation (iO), hidden coset states possess a certain *direct product hardness* property, which immediately implies a tokenized signature scheme in the plain model. Previously, a tokenized signature scheme was known only relative to an oracle, from a work of Ben-David and Sattath [QCrypt '17].
- Combining a tokenized signature scheme with extractable witness encryption, we give a construction of an unclonable decryption scheme in the plain model. The latter primitive was recently proposed by Georgiou and Zhandry [ePrint '20], who gave a construction relative to a classical oracle.
- We conjecture that coset states satisfy a certain natural (information-theoretic) monogamy-of-entanglement property. Assuming this conjecture is true, we remove the requirement for extractable witness encryption in our unclonable decryption construction, by relying instead on compute-and-compare obfuscation for the class of unpredictable distributions. As potential evidence in support of the monogamy conjecture, we prove a weaker version of this monogamy property, which we believe will still be of independent interest.
- Finally, we give a construction of a copy-protection scheme for pseudorandom functions (PRFs) in the plain model. Our scheme is secure either assuming iO, OWF and extractable witness encryption, or assuming $\mathsf{iO}, \mathsf{OWF}$, compute-and-compare obfuscation for the class of unpredictable distributions, and the conjectured monogamy property mentioned above. This is the first example of a copy-protection scheme with provable security in the plain model for a class of functions that is not evasive.

Contents

1 Introduction

The no-cloning principle of quantum mechanics asserts that quantum information cannot be generically copied. This principle has profound consequences in quantum cryptography, as it puts a fundamental restriction on the possible strategies that a malicious party can implement. One of these consequences is that quantum information enables cryptographic tasks that are provably impossible to realize classically, the most famous example being information-theoretically secure key distribution [BB84].

Beyond this, the no-cloning principle opens up an exciting avenue to realize cryptographic tasks which enjoy some form of *unclonability*, e.g. quantum money [Wie83, AC12, FGH$^+$12, Zha19a, Kan18], quantum tokens for digital signatures [BS16], copy-protection of programs [Aar09, ALL$^+$20, CMP20], and more recently unclonable encryption [Got02, BL19] and decryption [GZ20].

In this work, we revisit the *hidden subspace* idea proposed by Aaronson and Christiano, which has been employed towards several of the applications above. We propose a generalization of this idea, which involves hidden *cosets* (affine subspaces), and we show applications of this to signature tokens, unclonable decryption and copy-protection.

Given a subspace $A \subseteq \mathbb{F}_2^n$, the corresponding *subspace state* is defined as a uniform superposition over all strings in the subspace A, i.e.

$$|A\rangle := \frac{1}{\sqrt{|A|}} \sum_{x \in A} |x\rangle \ ,$$

The first property that makes this state useful is that applying a Hadamard on all qubits creates a uniform superposition over all strings in $A^\perp$, the orthogonal complement of A, i.e. $H^{\otimes n} |A\rangle = |A^\perp\rangle$.

The second property, which is crucial for constructing unclonable primitives with some form of verification, is the following. Given one copy of $|A\rangle$, where $A \subseteq F_2^n$ is uniformly random of

dimension $n/2$, it is impossible to produce two copies of $|A\rangle$ except with negligible probability. As shown by [AC12], unclonability holds even when given quantum access to oracles for membership in A and $A^\perp$, as long as the number of queries is polynomially bounded. On the other hand, such membership oracles allow for verifying the state $|A\rangle$, leading to publicly-verifiable quantum money, where the verification procedure is the following:

- Given an alleged quantum money state $|\psi\rangle$, query the oracle for membership in A on input $|\psi\rangle$. Measure the outcome register, and verify that the outcome is 1.
- If so, apply $H^{\otimes n}$ to the query register, and query the oracle for membership in $A^\perp$. Measure the outcome register, and accept the money state if the outcome is 1.

It is not difficult to see that the unique state that passes this verification procedure is $|A\rangle$.

In order to obtain a quantum money scheme in the plain model (without oracles), Aaronson and Christiano suggest instantiating the oracles with some form of program obfuscation. This vision is realized subsequently in [Zha19a], where access to the oracles for subspace membership is replaced by a suitable obfuscation of the membership programs, which can be built from indistinguishability obfuscation (iO). More precisely, Zhandry shows that, letting P_A and $P_{A^\perp}$ be programs that check membership in A and $A^\perp$ respectively, any computationally bounded adversary who receives a uniformly random subspace state $|A\rangle$ together with $\mathsf{iO}(P_A)$ and $\mathsf{iO}(P_{A^\perp})$ cannot produce two copies of $|A\rangle$ except with negligible probability.

The subspace state idea was later employed to obtain *quantum tokens* for digital signatures [BS16]. What these are is best explained by the (award-winning) infographic in [BS16] (see the ancillary arXiv files there). Concisely, a quantum signature token allows Alice to provide Bob with the ability to sign *one and only one* message in her name, where such signature can be publicly verified using Alice's public key. The construction of quantum tokens for digital signatures from [BS16] is the following.

- Alice samples a uniformly random subspace $A \subseteq \mathbb{F}_2^n$, which constitutes her secret key. A signature token is the state $|A\rangle$.
- Anyone in possession of a token $|A\rangle$ can sign message 0 by outputting a string $v \in A$ (this can be obtained by measuring $|A\rangle$ in the computational basis), and can sign message 1 by outputting a string $w \in A^\perp$ (this can be done by measuring $|A\rangle$ in the Hadamard basis).
- Signatures can be publicly verified assuming quantum access to an oracle for subspace membership in A and in $A^\perp$ (such access can be thought of as Alice's public key).

In order to guarantee security of the scheme, i.e. that Bob cannot produce a valid signature for more than one message, Ben-David and Sattath prove the following strengthening of the original property proven by Aaronson and Christiano. Namely, they show that any query-bounded adversary with quantum access to oracles for membership in A and $A^\perp$ cannot produce, except with negligible probability, a pair (v, w) where $v \in A \setminus \{0\}$ and $w \in A^\perp \setminus \{0\}$. We refer to this property as a *direct product hardness* property.

The natural step to obtain a signature token scheme in the plain model is to instantiate the subspace membership oracles using iO, analogously to the quantum money application. However, unlike for the case of quantum money, here one runs into a technical barrier, which we expand upon in Section 2.1. Thus, a signature token scheme is not known in the plain model, and this has remained an open question since [BS16].

In general, a similar difficulty in obtaining schemes that are secure in the plain model as opposed to an oracle model seems prevalent in works about other unclonable primitives. For example, in the case of copy-protection of programs, we know that copy-protection of a large class of evasive programs, namely compute-and-compare programs, is possible with provable non-trivial security against fully malicious adversaries in the quantum random oracle model (QROM) [CMP20]. Other results achieving provable security in the plain model are secure only against a restricted class of adversaries [AP21, KNY20, BJL$^+$21]. To make the contrast between plain model and oracle model even more stark, all unlearnable programs can be copy-protected assuming access to (highly structured) oracles [ALL$^+$20], but we know, on the other hand, that a copy-protection scheme for all unlearnable programs in the plain model does not exist (assuming Learning With Errors is hard for quantum computers) [AP21].

Likewise, for the recently proposed task of unclonable decryption, the only currently known scheme is secure only in a model with access to subspace membership oracles [GZ20].

1.1 Our Results

We study a generalization of subspace states, which we refer to as *coset* states. This notion has also been studied independently in a work of Vidick and Zhang [VZ21], in the context of proofs of quantum knowledge from quantum money schemes.

For $A \subseteq \mathbb{F}_2^n$, and $s, s' \in \mathbb{F}_2^n$, the corresponding coset state is:

$$|A_{s,s'}\rangle := \sum_{x \in A} (-1)^{\langle x, s' \rangle} |x + s\rangle \ ,$$

where here $\langle x, s' \rangle$ denotes the inner product of x and s'. In the computational basis, the quantum state is a superposition over all elements in the coset $A + s$, while, in the Hadamard basis, it is a superposition over all elements in $A^\perp + s'$. Let P_{A+s} and $P_{A^\perp+s'}$ be programs that check membership in the cosets $A + s$ and $A^\perp + s'$ respectively. To check if a state $|\psi\rangle$ is a coset state with respect to A, s, s', one can compute P_{A+s} in the computational basis, and check that the outcome is 1; then, apply $H^{\otimes n}$ followed by $P_{A^\perp+s'}$, and check that the outcome is 1.

Computational Direct Product Hardness. Our first technical result is establishing a *computational direct product hardness* property in the plain model, assuming post-quantum iO and one-way functions.

Theorem 1.1 (Informal)**.** *Any quantum polynomial-time adversary who receives $|A_{s,s'}\rangle$ and programs $\mathsf{iO}(P_{A+s})$ and $\mathsf{iO}(P_{A^\perp+s'})$ for uniformly random $A \subseteq \mathbb{F}_2^n$, $s, s' \in \mathbb{F}_2^n$, cannot produce a pair $(v, w) \in (A + s) \times (A^\perp + s')$, except with negligible probability in n.*

As we mentioned earlier, this is in contrast to regular subspace states, for which a similar direct product hardness is currently not known in the plain model, but only in a model with access to subspace membership oracles.

We then apply this property to obtain the following primitives.

Signature Tokens. Our direct product hardness immediately implies a *signature token* scheme in the plain model (from post-quantum iO and one-way functions), thus resolving the main question left open in [BS16].

Theorem 1.2 (Informal)**.** *Assuming post-quantum* iO *and one-way functions, there exists a signature token scheme.*

In this signature token scheme, the public verification key is the pair $(\mathsf{iO}(P_{A+s}), \mathsf{iO}(P_{A^\perp+s'}))$, and a signature token is the coset state $|A_{s,s'}\rangle$. Producing signatures for both messages 0 and 1 is equivalent to finding elements in both $A + s$ and $A^\perp + s'$, which violates our computational direct product hardness property.

Unclonable Decryption. Unclonable decryption, also known as *single-decryptor encryption*, was introduced in [GZ20]. Informally, a single-decryptor encryption scheme is a (public-key) encryption scheme in which the secret key is a *quantum state*. The scheme satisfies a standard notion of security (in our case, CPA security), as well as the following additional security guarantee: no efficient quantum algorithm with one decryption key is able to produce two working decryption keys. We build a single-decryptor encryption scheme using a signature tokens scheme and extractable witness encryption in a black-box way. By leveraging our previous result about the existence of a signature token scheme in the plain model, we are able to prove security without the need for the structured oracles used in the original construction of [GZ20].

Theorem 1.3 (Informal)**.** *Assuming post-quantum* iO*, one-way functions, and extractable witness encryption, there exists a public-key single-decryptor encryption scheme.*

Copy-protection of PRFs. The notion of a copy-protection scheme was introduced by Aaronson in [Aar09] and recently explored further in [AP21, CMP20, ALL$^+$20, BJL$^+$21].

In a copy-protection scheme, the vendor of a classical program wishes to provide a user the ability to run the program on any input, while ensuring that the functionality cannot be "pirated": informally, the adversary, given one copy of the program, cannot produce two programs that enable evaluating the program correctly.

Copy-protection is trivially impossible classically, since classical information can always be copied. This impossibility can be in principle circumvented if the classical program is encoded in a quantum state, due to the no-cloning principle. However, positive results have so far been limited. A copy-protection scheme [CMP20] is known for a class of evasive programs, known as compute-and-compare programs, with provable non-trivial security against fully malicious adversaries in the Quantum Random Oracle Model (QROM). Other schemes in the plain model are only secure against restricted classes of adversaries (which behave honestly in certain parts of the protocol) [AP21, KNY20, BJL$^+$21]. Copy-protection schemes for more general functionalities are known [ALL$^+$20], but these are only secure assuming very structured oracles (which depend on the functionality that is being copy-protected).

In this work, we present a copy-protection scheme for a family of pseudorandom functions (PRFs). In such a scheme, for any classical key K for the PRF, anyone in possession of a *quantum* key ρ_K is able to evaluate $PRF(K, x)$ on any input x.

The copy-protection property that our scheme satisfies is that given a quantum key ρ_K, no efficient algorithm can produce two (possibly entangled) keys such that these two keys allow for simultaneous correct evaluation on uniformly random inputs, with noticeable probability.

Similarly to the unclonable decryption scheme, our copy-protection scheme is secure assuming post-quantum iO, one-way functions, and extractable witness encryption.

Theorem 1.4 (Informal)**.** *Assuming post-quantum* iO*, one-way functions, and extractable witness encryption, there exists a copy-protection scheme for a family of PRFs.*

We remark that our scheme requires a particular kind of PRFs, namely puncturing and extracting with small enough error (we refer to Section 7.2 for precise definitions). However, PRFs satisfying these properties can be built from just one-way functions.

The existence of extractable witness encryption is considered to be a very strong assumption. In particular, it was shown to be impossible in general (under a special-purpose obfuscation conjecture) [GGHW17]. However, we emphasize that no provably secure copy-protection schemes with standard malicious security in the plain model are known at all. Given the central role of PRFs in the construction of many other cryptographic primitives, we expect that our copy-protection scheme, and the techniques developed along the way, will play an important role as a building block to realize *unclonable* versions of other primitives.

To avoid the use of extractable witness encryption, we put forth a (information-theoretic) conjecture about a *monogamy of entanglement* property of coset states, which we discuss below. Assuming this conjecture is true, we show that both unclonable decryption and copy-protection of PRFs can be constructed *without* extractable witness encryption, by relying instead on compute-and-compare obfuscation [WZ17, GKW17] (more details on the latter can be found in Section 3.3).

Theorem 1.5 (Informal)**.** *Assuming post-quantum* iO*, one-way functions, and obfuscation of compute-and-compare programs against unpredictable distributions, there exist: (i) a public-key single-decryptor encryption scheme, and (ii) a copy-protection scheme for a family of PRFs.*

As potential evidence in support of the monogamy-of-entanglement conjecture, we prove a weaker version of the monogamy of entanglement property, which we believe will still be of independent interest (more details on this are below).

Remark 1.6. *While* iO *was recently constructed based on widely-believed computational assumptions [JLS20], the latter construction is not quantum resistant, and the situation is less clear quantumly. However, several works have proposed candidate post-quantum obfuscation schemes [BGMZ18, WW20, BDGM20], and based on these works* iO *seems plausible in the post-quantum setting as well.*

Remark 1.7. *Compute-and-compare obfuscation against unpredictable distributions is known to exist assuming LWE (or* iO*) and assuming the existence of Extremely Lossy Functions (ELFs) [Zha19b] [WZ17, GKW17]. Unfortunately, the only known constructions of ELFs rely on hardness assumptions that are broken by quantum computers (exponential hardness of decisional Diffie-Hellman). To remedy this, we give a construction of computate-and-compare obfuscation against* sub-exponentially *unpredictable distributions, from plain LWE (see Theorem 3.3, and its proof in Appendix B). The latter weaker obfuscation is sufficient to prove security of our single-decryptor encryption scheme, and copy-protection scheme for PRFs, if one additionally assumes* sub-exponentially *secure* iO *and one-way functions.*

Monogamy-of-Entanglement. As previously mentioned, we conjecture that coset states additionally satisfy a certain (information-theoretic) *monogamy of entanglement* property, similar to the one satisfied by BB84 states, which is studied extensively in [TFKW13]. Unlike the monogamy

property of BB84 states, the monogamy property we put forth is well-suited for applications with public verification, in a sense made more precise below.

This monogamy property states that Alice, Bob and Charlie cannot cooperatively win the following game with a challenger, except with negligible probability. The challenger first prepares a uniformly random coset state $|A_{s,s'}\rangle$ and gives the state to Alice. Alice outputs two (possibly entangled) quantum states and sends them to Bob and Charlie respectively. Finally, Bob and Charlie both get the description of the subspace A. The game is won if Bob outputs a vector in $A + s$ and Charlie outputs a vector in $A^\perp + s'$.

Notice that if Alice were told A before she had to send the quantum states to Bob and Charlie, then she could recover s and s' (efficiently) given $|A_{s,s'}\rangle$. Crucially, A is only revealed to Bob and Charlie *after* Alice has sent them the quantum states (analogously to the usual monogamy-of-entanglement game based on BB84 states, where θ is only revealed to Bob and Charlie after they receive their states from Alice.).

We note that the hardness of this game is an *information-theoretic* conjecture. As such, there is hope that it can be proven unconditionally.

Under this conjecture, we show that the problem remains hard (computationally) even if Alice additionally receives the programs $\mathsf{iO}(P_{A+s})$ and $\mathsf{iO}(P_{A^\perp+s'})$. Based on this result, we then obtain unclonable decryption and copy-protection of PRFs from post-quantum iO and one-way functions, and compute-and-compare obfuscation against unpredictable distributions. We thus remove the need for extractable witness encryption (more details on this are provided in the technical overview, Section 2.1).

As evidence in support of our conjecture, we prove a weaker information-theoretic monogamy property, namely that Alice, Bob and Charlie cannot win at a monogamy game that is identical to the one described above, except that at the last step, Bob and Charlie are each required to return a pair in $(A + s) \times (A^\perp + s')$, instead of a single element each. Since coset states have more algebraic structure than BB84 states, a more refined analysis is required to prove this (weaker) property compared to that of [TFKW13]. We again extend this monogamy result to the case where Alice receives programs $\mathsf{iO}(P_{A+s})$ and $\mathsf{iO}(P_{A^\perp+s'})$.

We emphasize that our monogamy result for coset states differs from the similar monogamy result for BB84 states in one crucial way: the result still holds when Alice receives programs that allow her to verify the correctness of her state (namely $\mathsf{iO}(P_{A+s})$ and $\mathsf{iO}(P_{A^\perp+s'})$). This is not the case for the BB84 monogamy result. In fact, Lutomirski [Lut10] showed that an adversary who is given $|x^\theta\rangle$ and a public verification oracle that outputs 1 if the input state is correct and 0 otherwise, can efficiently copy the state $|x^\theta\rangle$. At the core of this difference is the fact that coset states are highly entangled, whereas strings of BB84 states have no entanglement at all.

For this reason, we believe that the monogamy property of coset states may be of independent interest, and may find application in contexts where public verification of states is important.

2 Technical Overview

2.1 Computational Direct Product Hardness for Coset States

Our first technical contribution is to establish a *computational* direct product hardness property for coset states. In this section, we aim to give some intuition for the barrier to proving such a property for regular subspace states, and why resorting to coset states helps.

We establish the following: a computationally bounded adversary who receives $|A_{s,s'}\rangle$ and programs $\mathsf{iO}(P_{A+s})$ and $\mathsf{iO}(P_{A^\perp+s'})$ for uniformly random A, s, s', cannot produce a pair (v, w), where $v \in A + s$ and $w \in A^\perp + s'$, except with negligible probability.

The first version of this direct product hardness property involved regular subspace states, and was *information-theoretic*. It was proven by Ben-David and Sattath [BS16], and it established the following: given a uniformly random subspace state $|A\rangle$, where $A \subseteq \mathbb{F}_2^n$ has dimension $n/2$, no adversary can produce a pair of vectors v, w such that $v \in A$ and $w \in A^\perp$ respectively, even with access to oracles for membership in A and in $A^\perp$.

The first successful instantiation of the membership oracles in the plain model is due to Zhandry, in the context of public-key quantum money [Zha19a]. Zhandry showed that replacing the membership oracles with indistinguishability obfuscations of the membership programs P_A and $P_{A^\perp}$ is sufficient to prevent an adversary from copying the subspace state, and thus is sufficient for public-key quantum money. In what follows, we provide some intuition as to how one proves this "computational no-cloning" property, and why the same proof idea does not extend naturally to the direct product hardness property for regular subspace states.

In [Zha19a], Zhandry shows that iO realizes what he refers to as a *subspace-hiding obfuscator*. A subspace hiding obfuscator shO has the property that any computationally bounded adversary who chooses a subspace A cannot distinguish between $\mathsf{shO}(P_A)$ and $\mathsf{shO}(P_B)$ for a uniformly random superspace B of A (of not too large dimension). In turn, a subspace hiding obfuscator can then be used to show that an adversary who receives $|A\rangle$, $\mathsf{shO}(P_A)$ and $\mathsf{shO}(P_{A^\perp})$, for a uniformly random A, cannot produce two copies of $|A\rangle$. This is done in the following way. For the rest of the section, we assume that $A \subseteq \mathbb{F}_2^n$ has dimension $n/2$.

- Replace $\mathsf{shO}(P_A)$ with $\mathsf{shO}(P_B)$ for a uniformly random superspace B of A, where $\dim(B) = \frac{3}{4}n$. Replace $\mathsf{shO}(P_{A^\perp})$ with $\mathsf{shO}(P_C)$ for a uniformly random superspace C of $A^\perp$, where $\dim(C) = \frac{3}{4}n$.

- Argue that the task of copying a subspace state $|A\rangle$, for a uniformly random subspace $C^\perp \subseteq A \subseteq B$ (even knowing B and C directly) is just as hard as the task of copying a uniformly random subspace state of dimension $|A'\rangle \subseteq \mathbb{F}_2^{n/2}$ where $\dim(A') = \frac{n}{4}$. The intuition for this is that knowing $C^\perp$ fixes $\frac{n}{4}$ dimensions out of the $\frac{n}{2}$ original dimensions of A. Then, you can think of the first copying task as equivalent to the second up to a change of basis. Such reduction completely removes the adversary's knowledge about the membership programs.

- The latter task is of course hard (it would even be hard with access to membership oracles for A' and $A'^\perp$).

One can try to apply the same idea to prove a *computational direct product hardness property* for subspace states, where the task is no longer to copy $|A\rangle$, but rather we wish to show that a bounded adversary receiving $|A\rangle$ and programs $\mathsf{iO}(P_A)$ and $\mathsf{iO}(P_{A^\perp})$, for uniformly random A, cannot produce a pair (v, w), where $v \in A$ and $w \in A^\perp$. Applying the same replacements as above using shO allows us to reduce this task to the task of finding a pair of vectors in $A \times A^\perp$ given $|A\rangle$,B, C, such that $C^\perp \subseteq A \subseteq B$. Unfortunately, unlike in the case of copying, this task is easy, because any pair of vectors in $C^\perp \times B^\perp$ also belongs to $A \times A^\perp$. This is the technical hurdle that ones runs into when trying to apply the proof idea from [Zha19a] to obtain a computational direct hardness property for subspace states.

Our first result is that we overcome this hurdle by using coset states. In the case of cosets, the natural analog of the argument above results in a replacement of the program that checks membership in $A+s$ with a program that checks membership in $B+s$. Similarly, we replace $A^\perp+s'$ with $C+s'$. The crucial observation is that, since $B+s = B+s+t$ for any $t \in B$, the programs P_{B+s} and P_{B+s+t} are functionally equivalent. So, an adversary who receives $\mathsf{iO}(P_{B+s})$ cannot distinguish this from $\mathsf{iO}(P_{B+s+t})$ for any t. We can thus argue that t functions as a randomizing mask that prevents the adversary from guessing s and finding a vector in $A+s$.

Signature Tokens. The computational direct product hardness immediately gives a signature token scheme in the plain model:

- Alice samples a key (A, s, s') uniformly at random. This constitutes her secret key. The verification key is $(\mathsf{iO}(P_{A+s}), \mathsf{iO}(P_{A^\perp+s'}))$. A signature token is $|A_{s,s'}\rangle$.
- Anyone in possession of a token can sign message 0 by outputting a string $v \in A+s$ (this can be obtained by measuring the token in the computational basis), and can sign message 1 by outputting a string $w \in A^\perp + s'$ (this can be done by measuring the token in the Hadamard basis).
- Signatures can be publicly verified using Alice's public key.

If an algorithm produces both signatures for messages 0 and 1, it finds vectors $v \in A+s$ and $w \in A^\perp + s'$, which violates computational direct product hardness.

2.2 Unclonable Decryption

Our second result is an *unclonable decryption* scheme (also known as a *single-decryptor encryption* scheme [GZ20] - we will use the two terms interchangeably in the rest of the paper) from black-box use of a signature token scheme and extractable witness encryption. This construction removes the need for structured oracles, as used in the construction of [GZ20].

Additionally, we show that, assuming the conjectured monogamy property described in Section 1.1, we obtain an unclonable decryption scheme from just iO and post-quantum one-way functions, where iO is used to construct obfuscators for both subspace-membership programs and compute-and-compare programs [GKW17, WZ17].

In this overview, we focus on the construction from the monogamy property, as we think it is conceptually more interesting.

Recall that a single-decryptor encryption scheme is a public-key encryption scheme in which the secret key is a quantum state. On top of the usual encryption security notions, one can define "single-decryptor" security: this requires that it is not possible for an adversary who is given the secret key to produce two (possibly entangled) decryption keys, which both enable simultaneous successful decryption of ciphertexts. A simplified version of our single-decryptor encryption scheme is the following. Let $n \in \mathbb{N}$.

- The key generation procedure samples uniformly at random $A \subseteq \mathbb{F}_2^n$, with $\dim(A) = \frac{n}{2}$ and $s, s' \in \mathbb{F}_2^n$ uniformly at random. The public key is the pair $(\mathsf{iO}(P_{A+s}), \mathsf{iO}(P_{A^\perp+s'}))$. The (quantum) secret key is the coset state $|A_{s,s'}\rangle$.

- To encrypt a message m, sample uniformly $r \leftarrow \{0,1\}$, and set $R = \mathsf{iO}(P_{A+s})$ if $r = 0$ and $R = \mathsf{iO}(P_{A^\perp+s'})$ if $r = 1$. Then, let C be the following program:

  C: on input v, output the message m if $R(v) = 1$ and otherwise output $\perp$.

  The ciphertext is then $(r, \mathsf{iO}(C))$.

- To decrypt a ciphertext $(r, \mathsf{iO}(C))$ with the quantum key $|A_{s,s'}\rangle$, one simply runs the program $\mathsf{iO}(C)$ coherently on input $|A_{s,s'}\rangle$ if $r = 0$, and on $H^{\otimes n}|A_{s,s'}\rangle$ if $r = 1$.

In the full scheme, we actually amplify security by sampling $r \leftarrow \{0,1\}^\lambda$, and having λ coset states, but we choose to keep the presentation in this section as simple as possible.

The high level idea for single-decryptor security is the following. Assume for the moment that iO were an ideal obfuscator (we will argue after this that iO is good enough). Consider a pirate who receives a secret key, produces two copies of it, and gives one to Bob and the other to Charlie. Suppose both Bob and Charlie can decrypt ciphertexts $(r, \mathsf{iO}(C))$ correctly with probability close to 1, over the randomness in the choice of r (which is crucially chosen only after Bob and Charlie have received their copies). Then, there must be some efficient quantum algorithm, which uses Bob's (resp. Charlie's) auxiliary quantum information (whatever state he has received from the pirate), and is able to output a vector in $A+s$. This is because in the case of $r = 0$, the program C outputs the plaintext message m exclusively on inputs $v \in A+s$. Similarly, there must be an algorithm that outputs a vector in $A^\perp + s'$ starting from Bob's (resp. Charlie's) auxiliary quantum information. Notice that this doesn't imply that Bob can *simultaneously* output a pair in $(A+s) \times (A^\perp + s')$, because explicitly recovering a vector in one coset might destroy the auxiliary quantum information preventing recovery of a vector in the other (and this very fact is of course crucial to the direct product hardness). Hence, in order to argue that it is not possible for both Bob and Charlie to be decrypting with probability close to 1, we have to use the fact that Bob and Charlie have separate auxiliary quantum information, and that each of them can recover vectors in $A+s$ or $A^\perp+s'$, which means that this can be done simultaneously, now violating the direct product hardness property.

The crux of the security proof is establishing that iO is a good enough obfuscator to enable this argument to go through.

To this end, we first notice that there is an alternative way of computing membership in $A+s$, which is functionally equivalent to the program C defined above.

Let $\mathsf{Can}_A(s)$ be a function that computes the lexicographically smallest vector in $A+s$ (think of this as a representative of the coset). It is not hard to see that a vector t is in $A+s$ if and only if $\mathsf{Can}_A(t) = \mathsf{Can}_A(s)$. Also Can_A is efficiently computable given A. Therefore, a functionally equivalent program to C, in the case that $r = 0$, is:

$\widetilde{C}$: on input v, output m if $\mathsf{Can}_A(v) = \mathsf{Can}_A(s)$, otherwise output $\perp$.

By the security of iO, an adversary can't distinguish $\mathsf{iO}(C)$ from $\mathsf{iO}(\widetilde{C})$.

The key insight is that now the program $\widetilde{C}$ is a *compute-and-compare* program [GKW17, WZ17]. The latter is a program described by three parameters: an efficiently computable function f, a target y and an output z. The program outputs z on input x if $f(x) = y$, and otherwise outputs $\perp$. In our case, $f = \mathsf{Can}_A$, $y = \mathsf{Can}_A(s)$, and $z = m$. Goyal et al. [GKW17] and Wichs et al. [WZ17] show that, assuming LWE or assuming iO and certain PRGs, a compute-and-compare program can be obfuscated provided y is (computationally) unpredictable given the function f and the auxiliary information. More precisely, the obfuscation guarantee is that the obfuscated compute-and-compare program is indistinguishable from the obfuscation of a (simulated) program that outputs zero on

every input (notice, as a sanity check, that if y is unpredictable given f, then the compute-and-compare program must output zero almost everywhere as well). We will provide more discussion on compute-and-compare obfuscation for unpredictable distributions in the presence of quantum auxiliary input in Section 3.3 and Appendix B.

- By the security of iO, we can replace the ciphertext $(0, \mathsf{iO}(C))$, with the ciphertext $(0, \mathsf{iO}(\mathsf{CC.Obf}(\widetilde{C})))$ where $\mathsf{CC.Obf}$ is an obfuscator for compute-and-compare programs (this is because C has the same functionality as $\mathsf{CC.Obf}(\widetilde{C})$).
- By the security of $\mathsf{CC.Obf}$, we can replace the latter with $(0, \mathsf{iO}(\mathsf{CC.Obf}(Z)))$, where Z is the zero program. It is clearly impossible to decrypt from the latter, since no information about the message is present.

Thus, assuming iO cannot be broken, a Bob that is able to decrypt implies an adversary breaking the compute-and-compare obfuscation. This implies that there must be an efficient algorithm that can predict $y = \mathsf{Can}_A(s)$ with non-negligible probability given the function Can_A and the auxiliary information received by Bob. Similarly for Charlie.

Therefore, if Bob and Charlie, with their own quantum auxiliary information, can both independently decrypt respectively $(0, \mathsf{iO}(C))$ and $(1, \mathsf{iO}(C'))$ with high probability (where here C and C' only differ in that the former releases the encrypted message on input a vector in $A + s$, and C' on input a vector in $A^\perp + s'$), then there exist efficient quantum algorithms for Bob and Charlie that take as input the descriptions of $\mathsf{Can}_A(\cdot)$ and $\mathsf{Can}_{A^\perp}(\cdot)$ respectively (or of the subspace A), and their respective auxiliary information, and recover $\mathsf{Can}_A(s)$ and $\mathsf{Can}_{A^\perp}(s')$ respectively with non-negligible probability. Since $\mathsf{Can}_A(s) \in A + s$ and $\mathsf{Can}_{A^\perp}(s') \in A^\perp + s'$, this violates the strong monogamy property of coset states described in Section 1.1.

Recall that this states that Alice, Bob and Charlie cannot cooperatively win the following game with a challenger, except with negligible probability. The challenger first prepares a uniformly random coset state $|A_{s,s'}\rangle$ and gives the state to Alice. Alice outputs two (possibly entangled) quantum states and sends them to Bob and Charlie respectively. Finally, Bob and Charlie both get the description of the subspace A. The game is won if Bob outputs a vector in $A + s$ and Charlie outputs a vector in $A^\perp + s'$. Crucially, in this monogamy property, Bob and Charlie will both receive the description of the subspace A in the final stage, yet it is still not possible for both of them to be simultaneously successful.

What allows to deduce the existence of efficient extracting algorithms is the fact that the obfuscation of compute-and-compare programs from [GKW17, WZ17] holds provided y is computationally unpredictable given f (and the auxiliary information). Thus, an algorithm that breaks the obfuscation property implies an efficient algorithm that outputs y (with noticeable probability) given f (and the auxiliary information).

In our other construction from signature tokens and extractable witness encryption, one can directly reduce unclonable decryption security to direct product hardness. We do not discuss the details of this construction in this section, instead we refer the reader to Section 6.5.

2.3 Copy-Protecting PRFs

Our last contribution is the construction of copy-protected PRFs assuming post-quantum iO, one-way functions and the monogamy property we discussed in the previous section. Alternatively just

as for unclonable decryption, we can do away with the monogamy property by assuming extractable witness encryption.

A copy-protectable PRF is a regular PRF $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^{m'}$, except that it is augmented with a *quantum key* generation procedure, which we refer to as $\mathsf{QKeyGen}$. This takes as input the classical PRF key K and outputs a quantum state ρ_K. The state ρ_K allows to efficiently compute $F(K,x)$ on any input x (where correctness holds with overwhelming probability). Beyond the standard PRF security, the copy-protected PRF satisfies the following additional security guarantee: any computationally bounded adversary that receives ρ_K cannot process ρ_K into two states, such that each state enables efficient evaluation of $F(K,\cdot)$ on uniformly random inputs.

A simplified version of our construction has the following structure. For the rest of the section, we take all subspaces to be of $\mathbb{F}_2^n$ with dimension $n/2$.

- The quantum key generation procedure $\mathsf{QKeyGen}$ takes as input a classical PRF key K and outputs a quantum key. The latter consists of a number of uniformly sampled coset states $|(A_i)_{s_i,s'_i}\rangle$, for $i \in [\lambda]$, together with a (classical) *obfuscation* of the classical program P that operates as follows. P takes an input of the form $(x, v_1, \ldots, v_\lambda)$; checks that each vector v_i belongs to the correct coset ($A_i + s_i$ if $x_i = 0$, and $A_i^\perp + s'_i$ if $x_i = 1$); if so, outputs the value $F(K,x)$, otherwise outputs $\perp$.

- A party in possession of the quantum key can evaluate the PRF on input x as follows: for each i such that $x_i = 1$, apply $H^{\otimes n}$ to $|(A_i)_{s_i,s'_i}\rangle$. Measure each resulting coset state in the standard basis to obtain vectors $v_1, \ldots, v_\lambda$. Run the obfuscated program on input $(x, v_1, \ldots, v_\lambda)$.

Notice that the program has the classical PRF key K hardcoded, as well as the values A_i, s_i, s'_i, so giving the program in the clear to the adversary would be completely insecure: once the adversary knows the key K, he can trivially copy the functionality $F(K,\cdot)$; and even if the key K is hidden by the obfuscation, but the A_i, s_i, s'_i are known, a copy of the (classical) obfuscated program P, together with the A_i, s_i, s'_i is sufficient to evaluate $F(K,\cdot)$ on any input.

So, the hope is that an appropriate obfuscation will be sufficient to hide all of these parameters. If this is the case, then the intuition for why the scheme is secure is that in order for two parties to simultaneously evaluate correctly on uniformly random inputs, each party should be able to produce a vector in $A_i + s$ or in $A_i^\perp + s'_i$. If the two parties accomplish this separately, then this implies that it is possible to simultaneously extract a vector in $A_i + s_i$ and one in $A_i^\perp + s'_i$, which should not be possible. [1]

We will use iO to obfuscate the program P. In the next part of this overview, we will discuss how we are able to deal with the fact that the PRF key K and the cosets are hardcoded in the program P. First of all, we describe a bit more precisely the copy-protection security that we wish to achieve. The latter is captured by the following security game between a challenger and an adversary (A, B, C):

- The challenger samples a uniformly random PRF key K and runs $\mathsf{QKeyGen}$ to generate ρ_K. Sends ρ_K to A.

- A sends quantum registers to two spatially separated parties B and C.

[1] Again, we point out that we could not draw this conclusion if only a single party were able to do the following two things, each with non-negligible probability: produce a vector in $A + s_i$ and produce a vector in $A^\perp + s'_i$. This is because in a quantum world, being able to perform two tasks with good probability, does not imply being able to perform both tasks simultaneously. So it is crucial that both parties are able to separately recover the vectors.

- The challenger samples uniformly random inputs x, x' to $F(K, \cdot)$. Sends x to B and x' to C.
- B and C return y and y' respectively to the challenger.

(A, B, C) wins if $y = F(K, x)$ and $y' = F(K, x')$.

Since the obfuscation we are using is not VBB, but only iO, there are two potential issues with security. B and C could be returning correct answers not because they are able to produce vectors in the appropriate cosets, but because:

(i) $\mathsf{iO}(P)$ leaks information about the PRF key K.

(ii) $\mathsf{iO}(P)$ leaks information about the cosets.

We handle issue (i) via a delicate "puncturing" argument [SW14]. At a high level, a puncturable PRF F is a PRF augmented with a procedure that takes a key K and an input value x, and produces a "punctured" key $K \setminus \{x\}$, which enables evaluation of $F(K, \cdot)$ at any point other than x. The security guarantee is that a computationally bounded adversary possessing the punctured key $K \setminus \{x\}$ cannot distinguish between $F(K, x)$ and a uniformly random value (more generally, one can puncture the key at any polynomially sized set of points). Puncturable PRFs can be obtained from OWFs using the [GGM86] construction [BW13].

By puncturing K precisely at the challenge inputs x and x', one is able to hardcode a punctured PRF key $K \setminus \{x, x'\}$ in the program P, instead of K, and setting the output of program P at x to uniformly random z and z', instead of to $F(K, x)$ and $F(K, x')$ respectively. The full argument is technical, and relies on the "hidden trigger" technique introduced in [SW14], which allows the "puncturing" technique to work even when the program P is generated before x and x' are sampled.

Once we have replaced the outputs of the program P on the challenge inputs x, x' with uniformly random outputs z, z', we can handle issue (ii) in a similar way to the case of unclonable decryption in the previous section.

By the security of iO, we can replace the behaviour of program P at x by a suitable functionally equivalent compute-and-compare program that checks membership in the appropriate cosets. We then replace this by an obfuscation of the same compute-and-compare program, and finally by an obfuscation of the zero program. We can then perform a similar reduction as in the previous section from an adversary breaking copy-protection security (and thus the security of the compute-and-compare obfuscation) to an adversary breaking the monogamy of entanglement game described in the previous section.

As in the previous section, we can replace the reliance on the conjectured monogamy property by extractable witness encryption. In fact, formally, we directly reduce the security of our copy-protected PRFs to the security of our unclonable decryption scheme.

3 Preliminaries

In this paper, we use λ to denote security parameters. We denote a function belonging to the class of polynomial functions by $\mathsf{poly}(\cdot)$. We say a function $f(\cdot) : \mathbb{N} \to \mathbb{R}^+$ is negligible if for all constant $c > 0$, $f(n) < \frac{1}{n^c}$ for all large enough n. We use $\mathsf{negl}(\cdot)$ to denote a negligible function. We say a function $f(\cdot) : \mathbb{N} \to \mathbb{R}^+$ is sub-exponential if there exists a constant $0 < c < 1$, such that $f(n) = 2^{n^c}$ for all large enough n. We use $\mathsf{subexp}(\cdot)$ to denote a sub-exponential function.

When we refer to a probabilistic algorithm $\mathcal{A}$, sometimes we need to specify the randomness r used by $\mathcal{A}$ when running on some input x. We write this as $\mathcal{A}(x;r)$.

For a finite set S, we use $x \leftarrow S$ to denote uniform sampling of x from the set S. We denote $[n] = \{1, 2, \cdots, n\}$. A binary string $x \in \{0,1\}^\ell$ is represented as $x_1 x_2 \cdots x_\ell$. For two strings x, y, $x||y$ is the concatenation of x and y.

We refer to a probabilistic polynomial-time algorithm as PPT, and we refer to a quantum polynomial-time algorithm as QPT.

We will assume familiarity with basic quantum information and computation concepts. We refer the reader to Appendix A.1 and [NC02] for a reference.

3.1 Pseudorandom Functions

For the rest of this paper, we will assume that all of the classical cryptographic primitives used are post-quantum (i.e. secure against quantum adversaries), and we sometimes omit mentioning this for convenience, except in formal definitions and theorems.

Definition 3.1 (PRF)**.** *A pseudorandom function (PRF) is a function $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$, where $\{0,1\}^k$ is the key space, and $\{0,1\}^n$ and $\{0,1\}^m$ are the domain and range. k, n and m are implicity functions of a security parameter λ. The following should hold:*

- *For every $K \in \{0,1\}^k$, $F(K,\cdot)$ is efficiently computable;*
- *PRF security: no efficient quantum adversary $\mathcal{A}$ making quantum queries can distinguish between a truly random function and the function $F(K,\cdot)$; that is for every such $\mathcal{A}$, there exists a negligible function* negl*,*

$$\left| \Pr_{K \leftarrow \{0,1\}^k} \left[\mathcal{A}^{F(K,\cdot)}() = 1 \right] - \Pr_{O : \{0,1\}^n \to \{0,1\}^m} \left[\mathcal{A}^{O}() = 1 \right] \right| \leq \mathsf{negl}(\lambda)$$

3.2 Indistinguishability Obfuscation

Definition 3.2 (Indistinguishability Obfuscator (iO) [BGI+01, GGH+16, SW14])**.** *A uniform PPT machine* iO *is an indistinguishability obfuscator for a circuit class $\{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ if the following conditions are satisfied:*

- *For all λ, all $C \in \mathcal{C}_\lambda$, all inputs x, we have*

$$\Pr\left[\widehat{C}(x) = C(x) \,|\, \widehat{C} \leftarrow \mathsf{iO}(1^\lambda, C) \right] = 1$$

- *(Post-quantum security): For all (not necessarily uniform) QPT adversaries (Samp, D), the following holds: if $\Pr[\forall x, C_0(x) = C_1(x) \,:\, (C_0, C_1, \sigma) \leftarrow \mathsf{Samp}(1^\lambda)] > 1 - \alpha(\lambda)$ for some negligible function α, then there exists a negligible function β such that:*

$$\Bigg| \Pr\left[D(\sigma, \mathsf{iO}(1^\lambda, C_0)) = 1 \,:\, (C_0, C_1, \sigma) \leftarrow \mathsf{Samp}(1^\lambda) \right]$$
$$- \Pr\left[D(\sigma, \mathsf{iO}(1^\lambda, C_1)) = 1 \,:\, (C_0, C_1, \sigma) \leftarrow \mathsf{Samp}(1^\lambda) \right] \Bigg| \leq \beta(\lambda)$$

Whenever we assume the existence of iO in the rest of the paper, we refer to iO for the class of polynomial-size circuits, i.e. when $\mathcal{C}_\lambda$ is the collection of all circuits of size at most λ.

We will also make use of the stronger notion of *sub-exponentially secure* iO. By the latter, we mean that the distinguishing advantage above is $1/\mathsf{subexp}$ for some sub-exponential function subexp, instead of negligible (while the adversary is still QPT).

Similarly, we will also make use of sub-exponentially secure one-way functions. For the latter, the advantage is again $1/\mathsf{subexp}$ (and the adversary is QPT).

3.3 Compute-and-Compare Obfuscation

Definition 3.3 (Compute-and-Compare Program)**.** *Given a function $f : \{0,1\}^{\ell_{\mathsf{in}}} \to \{0,1\}^{\ell_{\mathsf{out}}}$ along with a target value $y \in \{0,1\}^{\ell_{\mathsf{out}}}$ and a message $z \in \{0,1\}^{\ell_{\mathsf{msg}}}$, we define the compute-and-compare program:*

$$\mathsf{CC}[f,y,z](x) = \begin{cases} z & \text{if } f(x) = y \\ \bot & \text{otherwise} \end{cases}$$

We define the following class of *unpredictable distributions* over pairs of the form $(\mathsf{CC}[f,y,z],\mathsf{aux})$, where aux is auxiliary quantum information. These distributions are such that y is computationally unpredictable given f and aux.

Definition 3.4 (Unpredictable Distributions)**.** *We say that a family of distributions $D = \{D_\lambda\}$ where D_λ is a distribution over pairs of the form $(\mathsf{CC}[f,y,z],\mathsf{aux})$ where aux is a quantum state, belongs to the class of* unpredictable distributions *if the following holds. There exists a negligible function negl such that, for all QPT algorithms $\mathcal{A}$,*

$$\Pr_{(\mathsf{CC}[f,y,z],\mathsf{aux})\leftarrow D_\lambda}\Big[A(1^\lambda, f, \mathsf{aux}) = y\Big] \leq \mathsf{negl}(\lambda).$$

We further define the class of *sub-exponentially unpredictable distributions*, where we require the guessing probability to be inverse sub-exponential in the security parameter.

Definition 3.5 (Sub-Exponentially Unpredictable Distributions)**.** *We say that a family of distributions $D = \{D_\lambda\}$ where D_λ is a distribution over pairs of the form $(\mathsf{CC}[f,y,z],\mathsf{aux})$ where aux is a quantum state, belongs to the class of* sub-exponentially unpredictable distributions *if the following holds. There exists a sub-exponential function subexp such that, for all QPT algorithms $\mathcal{A}$,*

$$\Pr_{(\mathsf{CC}[f,y,z],\mathsf{aux})\leftarrow D_\lambda}\Big[A(1^\lambda, f, \mathsf{aux}) = y\Big] \leq 1/\mathsf{subexp}(\lambda).$$

We assume that a program P has an associated set of parameters $P.\mathsf{param}$ (e.g input size, output size, circuit size, etc.), which we are not required to hide.

Definition 3.6 (Compute-and-Compare Obfuscation)**.** *A PPT algorithm $\mathsf{CC.Obf}$ is an obfuscator for the class of unpredictable distributions (or sub-exponentially unpredictable distributions) if for any family of distributions $D = \{D_\lambda\}$ belonging to the class, the following holds:*

- *Functionality Preserving: there exists a negligible function negl such that for all λ, every program P in the support of D_λ,*

$$\Pr[\forall x,\ \widetilde{P}(x) = P(x),\ \widetilde{P} \leftarrow \mathsf{CC.Obf}(1^\lambda, P)] \geq 1 - \mathsf{negl}(\lambda)$$

- *Distributional Indistinguishability: there exists an efficient simulator* Sim *such that:*

$$(\mathsf{CC.Obf}(1^\lambda, P), \mathsf{aux}) \approx_c (\mathsf{Sim}(1^\lambda, P.\mathsf{param}), \mathsf{aux})$$

where $(P, \mathsf{aux}) \leftarrow D_\lambda$.

Combining the results of [WZ17, GKW17] with those of [Zha19b], we have the following two theorems. For the proofs and discussions, we refer the readers to Appendix B. Note that although Theorem B.2 is a strictly stronger statement, currently we do not know of any post-quantum construction for ELFs.

Theorem B.1. *Assuming the existence of post-quantum* iO *and the quantum hardness of LWE, there exist obfuscators for sub-exponentially unpredictable distributions, as in Definition 3.6.*

Theorem B.2. *Assuming the existence of post-quantum* iO *and post-quantum extremely lossy functions (ELFs), there exist obfuscators as in Definition 3.6. for any unpredictable distributions.*

3.4 Subspace Hiding Obfuscation

Subspace-hiding obfuscation was introduced by Zhandry [Zha19a] as a key component in constructing public-key quantum money. This notion requires that the obfuscation of a circuit that computes membership in a subspace A is indistinguishable from the obfuscation of a circuit that computes membership in a uniformly random superspace of A (of dimension sufficiently far from the full dimension). The formal definition is as follows.

Definition 3.7 ([Zha19a])**.** *A subspace hiding obfuscator (shO) for a field* $\mathbb{F}$ *and dimensions* d_0, d_1 *is a PPT algorithm* shO *such that:*

- ***Input.*** shO *takes as input the description of a linear subspace* $S \subseteq \mathbb{F}^n$ *of dimension* $d \in \{d_0, d_1\}$.

  For concreteness, we will assume S *is given as a matrix whose rows form a basis for* S.

- ***Output.*** shO *outputs a circuit* $\hat{S}$ *that computes membership in* S*. Precisely, let* $S(x)$ *be the function that decides membership in* S*. Then there exists a negligible function* negl*,*

$$\Pr[\hat{S}(x) = S(x) \quad \forall x : \hat{S} \leftarrow \mathsf{shO}(S)] \geq 1 - \mathsf{negl}(n)$$

- ***Security.*** *For security, consider the following game between an adversary and a challenger.*
  - *The adversary submits to the challenger a subspace* S_0 *of dimension* d_0.
  - *The challenger samples a uniformly random subspace* $S_1 \subseteq \mathbb{F}^n$ *of dimension* d_1 *such that* $S_0 \subseteq S_1$.
    It then runs $\hat{S} \leftarrow \mathsf{shO}(S_b)$*, and gives* $\hat{S}$ *to the adversary.*
  - *The adversary makes a guess* b' *for* b.

  shO *is secure if all QPT adversaries have negligible advantage in this game.*

Zhandry [Zha19a] gives a construction of a subspace hiding obfuscator based on one-way functions and iO.

Theorem 3.8 (Theorem 6.3 in [Zha19a])**.** *If injective one-way functions exist, then any indistinguishability obfuscator, appropriately padded, is also a subspace hiding obfuscator for field* $\mathbb{F}$ *and dimensions* d_0, d_1*, as long as* $|\mathbb{F}|^{n-d_1}$ *is exponential.*

3.5 Extractable Witness Encryption

In this subsection, we describe the primitive of witness encryption [GGHW17] with extractable security, which will we use in our construction of unclonable decryption in Section 6.5.

Definition 3.9 (Extractable Witness Encryption)**.** *An extractable witness encryption scheme for an NP relation R is a pair of algorithms* $(\mathsf{Enc}, \mathsf{Dec})$*:*

- $\mathsf{Enc}(1^\lambda, x, m) \to \mathsf{ct}$: *takes as input a security parameter λ in unary, an instance x and a message m, and outputs a ciphertext* ct*.*
- $\mathsf{Dec}(\mathsf{ct}, w) \to m/\bot$: *takes as input a ciphertext* ct *and a witness w and outputs a message m or $\bot$ (for decryption failure).*

The scheme satisfies the following:

Correctness: *For any security parameter $\lambda \in \mathbb{N}$, for any $m \in \{0,1\}$, for any x and w such that $R(x,w) = 1$, we have that:*

$$\Pr[\mathsf{Dec}(\mathsf{Enc}(1^\lambda, x, m), w] = m] = 1$$

Extractable Security: *For any QPT adversary $\mathcal{A}$, polynomial-time sampler $(x, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda)$ and for any polynomial $q(\cdot)$, there exists a QPT extractor E and a polynomial $p(\cdot)$, such that:*

$$\Pr\left[\mathcal{A}(1^\lambda, x, \mathsf{ct}, \mathsf{aux}) = m \middle| \begin{array}{c} m \leftarrow \{0,1\}, (x, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda), \\ \mathsf{ct} \leftarrow \mathsf{Enc}(1^\lambda, x, m) \end{array}\right] \geq \frac{1}{2} + \frac{1}{q(\lambda)}$$

$$\to \Pr\left[E(1^\lambda, x, \mathsf{aux}) = w \text{ s.t. } R(x,w) = 1 : (x, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda)\right] \geq \frac{1}{p(\lambda)}.$$

3.6 Testing Quantum Adversaries: Projective Implementation

In this section, we include several definitions about measurements, which are relevant to testing whether quantum adversaries are successful in the security games of Section 6.2. Part of this section is taken verbatim from [ALL+20]. As this section only pertains directly to our security definitions for unclonable decryption schemes, the reader can skip ahead, and return to this section when reading Section 6.2. In particular, this section is not needed to understand Sections 4 and 5.

In classical cryptographic security games, the challenger typically gets some information from the adversary and checks if this information satisfies certain properties. However, in a setting where the adversary is required to return *quantum* information to the challenger, classical definitions of "testing" whether a quantum state returned by the adversary satisfies certain properties may result in various failures as discussed in [Zha20], as this state may be in a superposition of "successful" and "unsuccessful" adversaries. We provide here a short description of some of the difficulties in the quantum setting, and we refer the reader to [Zha20] for a more in-depth discussion.

As an example, consider a security game in which an adversary is required to return some information to a challenger, which enables evaluation of a program on any input. Such a scenario is natural in copy-protection, where the adversary (a "pirate") attempts to create two copies of a copy-protected program, given just a single copy (and one can think of these two copies as being returned to the challenger for testing).

Naturally, one would consider a copy-protected program to be "good" if it enables correct evaluation on all inputs, or at least on a large fraction of all inputs. Testing correct evaluation on all inputs is of course not possible efficiently (not even classically). Instead, one would typically have the challenger *estimate* the fraction of correct evaluations to high statistical confidence by picking a large enough number of inputs uniformly at random (or from an appropriate distribution), running the copy-protected program on these inputs, and computing the fraction of correct evaluations. Unfortunately, such a test does not easily translate to the quantum setting. The reason is that the challenger only gets a single copy of the program, which in a quantum world cannot be generically copied. Moreover, in general, each evaluation may alter the copy-protected program in an irreversible way (if the outcome of the evaluation is not deterministic). Thus, estimating the fraction of inputs on which the copy-protected program received from the adversary evaluates correctly is not in general possible. For instance, consider an adversary who sends a state $\frac{1}{\sqrt{2}}|P_0\rangle + \frac{1}{\sqrt{2}}|P_1\rangle$ to the challenger, where $|P_0\rangle$ is a copy-protected program that evaluates perfectly on every input, and $|P_1\rangle$ is a useless program. Using this state, evaluation is successful on any input with probability $1/2$. Thus, even a single evaluation collapses the state either to $|P_0\rangle$ or to $|P_1\rangle$, preventing the challenger from performing subsequent evaluations on the original state. In fact, it is impossible to have a generic procedure that estimates the "average success probability of evalutation" to very high precision, as this would imply a procedure that distinguishes between the state $\frac{1}{\sqrt{2}}|P_0\rangle + \frac{1}{\sqrt{2}}|P_1\rangle$ and the state $|P_0\rangle$ almost perfectly, which is impossible since the two states have large overlap.

Projective Implementation Motivated by the discussion above, [Zha20] formalizes a new measurement procedure for testing a state received by an adversary. We will be adopting this procedure when defining security of single-decryptor encryption schemes in Section 6.2.

Consider the following procedure as a binary POVM $\mathcal{P}$ acting on an alleged-copy-protected program ρ: sample a uniformly random input x, evaluates the copy-protected program on x, and checks if the output is correct. In a nutshell, the new procedure consists of applying an appropriate projective measurement which *measures* the success probability of the tested state ρ under $\mathcal{P}$, and to output "accept" if the success probability is high enough. Of course, such measurement will not be able extract the exact success probability of ρ, as this is impossible from we have argued in the discussion above. Rather, the measurement will output a success probability from a finite set, such that the expected value of the output matches the true success probability of ρ. We will now describe this procedure in more detail.

The starting point is that a POVM specifies exactly the probability distribution over outcomes $\{0,1\}$ ("success" or "failure") on any copy-protected program, but it does not uniquely determine the post-measurement state. Zhandry shows that, for any binary POVM $\mathcal{P} = (P, I-P)$, there exists a particularly nice implementation of $\mathcal{P}$ which is projective, and such that the post-measurement state is an eigenvector of P. In particular, Zhandry observes that there exists a projective measurement $\mathcal{E}$ which *measures* the success probability of a state with respect to $\mathcal{P}$. More precisely,

- $\mathcal{E}$ outputs a *distribution* D of the form $(p, 1-p)$ from a finite set of distribution over outcomes $\{0,1\}$. (we stress that $\mathcal{E}$ actually outputs a distribution).
- The post-measurement state upon obtaining outcome $(p, 1-p)$ is an *eigenvector* (or a mixture of eigenvectors) of P with eigenvalue p.

A measurement $\mathcal{E}$ which satisfies these properties is the measurement in the common eigenbasis of P and $I - P$ (such common eigenbasis exists since P and $I - P$ commute).

Note that since $\mathcal{E}$ is projective, we are guaranteed that applying the same measurement twice will yield the same outcome. Thus, what we obtain from applying $\mathcal{E}$ is a state with a "well-defined" success probability with respect to $\mathcal{P}$: we know exactly how good the leftover program is with respect to the initial testing procedure $\mathcal{P}$.

Formally, to complete the implementation of $\mathcal{P}$, after having applied $\mathcal{E}$, one outputs the bit 1 with probability p, and the bit 0 with probability $1 - p$. This is summarized in the following definition.

Definition 3.10 (Projective Implementation of a POVM)**.** *Let $\mathcal{P} = (P, Q)$ be a binary outcome POVM. Let $\mathcal{D}$ be a finite set of distributions $(p, 1-p)$ over outcomes $\{0, 1\}$. Let $\mathcal{E} = \{E_p\}_{(p,1-p)\in\mathcal{D}}$ be a projective measurement with index set $\mathcal{D}$. Consider the following measurement procedure:*

(i) Apply the projective measurement $\mathcal{E}$ and obtain as outcome a distribution $(p, 1-p)$ over $\{0, 1\}$;

(ii) Output a bit according to this distribution, i.e. output 1 w.p p and output 0 w.p $1 - p$.

We say the above measurement procedure is a projective implementation of $\mathcal{P}$, which we denote by $\mathsf{ProjImp}(\mathcal{P})$, if it is equivalent to $\mathcal{P}$ (i.e. it produces the same probability distribution over outcomes).

Zhandry shows that any binary POVM has a projective implementation, as in the previous definition.

Lemma 3.11 (Adapted from Lemma 1 in [Zha20])**.** *Any binary outcome POVM $\mathcal{P} = (P, Q)$ has a projective implementation $\mathsf{ProjImp}(\mathcal{P})$.*

Moreover, if the outcome is a distribution $(p, 1-p)$ when measuring under $\mathcal{E}$, the collapsed state ρ' is a mixture of eigenvectors of P with eigenvalue p, and it is also a mixture of eigenvectors of Q with eigenvalue $1 - p$.

As anticipated, the procedure that we will eventually use to test a state received from the adversary will be to:

(i) *Measure* the success probability of the state,

(ii) Accept if the outcome is large enough.

As you may guess at this point, we will employ the projective measurement $\mathcal{E}$ defined previously for step (i). We call this variant of the projective implementation a *threshold implementation*.

Threshold Implementation The concept of threshold implementation of a POVM was proposed by Zhandry, and formalized by Aaronson, Liu, Liu, Zhandry and Zhang [ALL$^+$20]. The following is a formal definition.

Definition 3.12 (Threshold Implementation)**.** *Let $\mathcal{P} = (P, Q)$ be a binary POVM. Let $\mathsf{ProjImp}(\mathcal{P})$ be a projective implementation of $\mathcal{P}$, and let $\mathcal{E}$ be the projective measurement in the first step of $\mathsf{ProjImp}(\mathcal{P})$ (using the same notation as in Definition 3.10). Let $\gamma > 0$. We refer to the following measurement procedure as a* threshold implementation *of $\mathcal{P}$ with parameter γ, and we denote is as $\mathsf{TI}_\gamma(\mathcal{P})$.*

- *Apply the projective measurement $\mathcal{E}$, and obtain as outcome a vector $(p, 1-p)$;*
- *Output a bit according to the distribution $(p, 1-p)$: output 1 if $p \geq \gamma$, and 0 otherwise.*

For simplicity, for any quantum state ρ, we denote by $\mathrm{Tr}[\mathsf{TI}_\gamma(\mathcal{P})\,\rho]$ the probability that the threshold implementation applied to ρ **outputs 1**. Thus, whenever $\mathsf{TI}_\gamma(\mathcal{P})$ appears inside a trace Tr, we treat $\mathsf{TI}_\gamma(\mathcal{P})$ as a projection onto the 1 outcome (i.e. the space spanned by eigenvectors of P with eigenvalue at least γ).

Similarly to Lemma 3.11, we have the following lemma.

Lemma 3.13. *Any binary outcome POVM $\mathcal{P} = (P, Q)$ has a threshold implementation $\mathsf{TI}_\gamma(\mathcal{P})$ for any γ.*

In this work, we are interested in threshold implementations of POVMs with a particular structure. These POVMs represent a challenger's test of a quantum state received from an adversary in a security game (like the POVM described earlier for testing whether a program evaluates correctly on a uniformly random input). These POVMs have the following structure:

- Sample a projective measurement from a set of projective measurements $\mathcal{I}$, according to some distribution D over $\mathcal{I}$.
- Apply this projective measurement.

We refer to POVMs of this form as *mixtures of projective measurements*. The following is a formal definition.

Definition 3.14 (Mixture of Projective Measurements)**.** *Let $\mathcal{R}$, $\mathcal{I}$ be sets. Let $D : \mathcal{R} \to \mathcal{I}$. Let $\{(P_i, Q_i)\}_{i\in I}$ be a collection of binary projective measurements. The* mixture of projective measurements *associated to $\mathcal{R}$, $\mathcal{I}$, D and $\{(P_i, Q_i)\}_{i\in I}$ is the binary POVM $\mathcal{P}_D = (P_D, Q_D)$ defined as follows:*

$$P_D = \sum_{i\in\mathcal{I}} \Pr[i \leftarrow D(R)]\, P_i, \qquad Q_D = \sum_{i\in\mathcal{I}} \Pr[i \leftarrow D(R)]\, Q_i,$$

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

In other words, $\mathcal{P}_D$ is implemented in the following way: sample randomness $r \leftarrow \mathcal{R}$, compute the index $i = D(r)$, and apply the projective measurement (P_i, Q_i). Thus, for any quantum state ρ, $\mathrm{Tr}[P_D\rho]$ is the probability that a projective measurement (P_i, Q_i), sampled according to the distribution induced by D, applied to ρ outputs 1.

The following lemma will be important in the proof of security for our single-decryptor encryption scheme in Section 6.

Informally, the lemma states the following. Let $\mathcal{P}_{D_0}$ and $\mathcal{P}_{D_1}$ be two mixtures of projective measurements, where D_0 and D_1 are two computationally indistinguishable distributions. Let $\gamma, \gamma' > 0$ be inverse-polynomially close. Then for any (efficiently constructible) state ρ, the probabilities of obtaining outcome 1 upon measuring $\mathsf{TI}_\gamma(\mathcal{P}_{D_0})$ and $\mathsf{TI}_{\gamma'}(\mathcal{P}_{D_1})$ respectively are negligibly close.

Theorem 3.15 (Theorem 6.5 in [Zha20]). *Let $\gamma > 0$. Let $\mathcal{P}$ be a collection of projective measurements indexed by some set $\mathcal{I}$. Let ρ be an efficiently constructible mixed state, and let D_0, D_1 be two efficiently sampleable and computationally indistinguishable distributions over $\mathcal{I}$. For any inverse polynomial ϵ, there exists a negligible function δ such that*

$$\mathrm{Tr}[\mathsf{TI}_{\gamma-\epsilon}(\mathcal{P}_{D_1})\rho] \geq \mathrm{Tr}[\mathsf{TI}_{\gamma}(\mathcal{P}_{D_0})\rho] - \delta\,,$$

where $\mathcal{P}_{D_i}$ is the mixture of projective measurements associated to $\mathcal{P}$ and D_i.

Approximating Threshold Implementation *Projective* and *threshold* implementations of POVMs are unfortunately not efficiently computable in general.

However, they can be approximated if the POVM is a mixture of projective measurements, as shown by Zhandry [Zha20], using a technique first introduced by Marriott and Watrous [MW05] in the context of error reduction for quantum Arthur-Merlin games.

We will make use of the following lemma from a subsequent work of Aaronson et al. [ALL+20].

Lemma 3.16 (Corollary 1 in [ALL+20]). *For any $\epsilon, \delta, \gamma \in (0,1)$, any collection of projective measurements $\mathcal{P} = \{(P_i, Q_i)\}_{i\in\mathcal{I}}$, where $\mathcal{I}$ is some index set, and any distribution D over $\mathcal{I}$, there exists a measurement procedure $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ that satisfies the following:*

- *$\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ implements a binary outcome measurement. For simplicity, we denote the probability of the measurement **outputting 1** on ρ by $\mathrm{Tr}[\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}\,\rho]$.*
- *For all quantum states ρ, $\mathrm{Tr}[\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}\,\rho] \geq \mathrm{Tr}[\mathsf{TI}_{\gamma}(\mathcal{P}_D)\,\rho] - \delta$.*
- *For all quantum states ρ, let ρ' be the post-measurement state after applying $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ on ρ, and obtaining outcome 1. Then, $\mathrm{Tr}[\mathsf{TI}_{\gamma-2\epsilon}(\mathcal{P}_D)\,\rho'] \geq 1 - 2\delta$.*
- *The expected running time is $T_{\mathcal{P},D} \cdot \mathsf{poly}(1/\epsilon, 1/(\log\delta))$, where $T_{\mathcal{P},D}$ is the combined running time of sampling according to D, of mapping i to (P_i, Q_i), and of implementing the projective measurement (P_i, Q_i).*

Intuitively the corollary says that if a quantum state ρ has weight p on eigenvectors with eigenvalues at least γ, then the measurement $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ will produce with probability at least $p - \delta$ a post-measurement state which has weight $1 - 2\delta$ on eigenvectors with eigenvalues at least $\gamma - 2\epsilon$. Moreover, the running time for implementing $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ is proportional to $\mathsf{poly}(1/\epsilon, 1/(\log\delta))$, which is a polynomial in λ as long as ϵ is any inverse polynomial and δ is any inverse sub-exponential function.

Crucially for applications to single-decryption encryption and copy-protection, the above lemma can be generalized to pairs of POVMs on bipartite states.

Lemma 3.17 (Lemma 3 in [ALL+20]). *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two collections of projective measurements, indexed by elements of $\mathcal{I}$, and let D_1 and D_2 be probability distributions over $\mathcal{I}$.*

For any $\epsilon, \delta, \gamma \in (0,1)$, let $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D_1,\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D_2,\gamma}$ be the measuring algorithms above. They satisfy:

- *For any bipartite (possibly entangled, mixed) quantum state $\rho \in \mathcal{H}_1 \otimes \mathcal{H}_2$,*

$$\mathrm{Tr}\left[\left(\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D_1,\gamma} \otimes \mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D_2,\gamma}\right)\rho\right] \geq \mathrm{Tr}\left[\left(\mathsf{TI}_\gamma(\mathcal{P}_{1,D_1}) \otimes \mathsf{TI}_\gamma(\mathcal{P}_{2,D_2})\right)\rho\right] - 2\delta,$$

where $\mathcal{P}_{i,D_i}$ is the mixture of projective measurement corresponding to $\mathcal{P}_i, D_i$.

- *For any (possibly entangled, mixed) quantum state $\rho \in \mathcal{H}_1 \otimes \mathcal{H}_2$, let ρ' be the (normalized) post-measurement state after applying the measurements $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D_1,\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D_2,\gamma}$ to ρ and obtaining outcomes 1 for both. Then,*

$$\mathrm{Tr}\left[\left(\mathsf{TI}_{\gamma-2\epsilon}(\mathcal{P}_{1,D_1}) \otimes \mathsf{TI}_{\gamma-2\epsilon}(\mathcal{P}_{2,D_2})\right)\rho'\right] \geq 1 - 4\delta.$$

4 Coset States

This section is organized as follows. In Section 4.1, we introduce coset states. In Section 4.2, we show that coset states satisfy both an information-theoretic and a computational *direct product hardness* property. The latter immediately yields a signature token scheme in the plain model assuming iO, (this is described in Section 5). In Section 4.3 we show that coset states satisfy both an information-theoretic *monogamy of entanglement* property (analogous to that satisfied by BB84 states [TFKW13]), and a computational monogamy of entanglement property. The latter is used in Section 6.5 to obtain an unclonable decryption scheme from iO and extractable witness encryption. In Section 4.4, we describe a *strong version* of the monogamy property, which we conjecture to be true. The latter is used in Section 6.3 to obtain an unclonable decryption scheme which does not assume extractable witness encryption.

4.1 Definitions

In this subsection, we provide the basic definitions and properties of coset states.

For any subspace A, its complement is $A^\perp = \{b \in \mathbb{F}^n \,|\, \langle a, b\rangle \bmod 2 = 0\,,\, \forall a \in A\}$. It satisfies $\dim(A) + \dim(A^\perp) = n$. We also let $|A| = 2^{\dim(A)}$ denote the size of the subspace A.

Definition 4.1 (Subspace States)**.** *For any subspace $A \subseteq \mathbb{F}_2^n$, the subspace state $|A\rangle$ is defined as*

$$|A\rangle = \frac{1}{\sqrt{|A|}} \sum_{a \in A} |a\rangle \ .$$

Note that given A, the subspace state $|A\rangle$ can be constructed efficiently.

Definition 4.2 (Coset States)**.** *For any subspace $A \subseteq \mathbb{F}_2^n$ and vectors $s, s' \in \mathbb{F}_2^n$, the coset state $|A_{s,s'}\rangle$ is defined as:*

$$|A_{s,s'}\rangle = \frac{1}{\sqrt{|A|}} \sum_{a \in A} (-1)^{\langle s', a\rangle} |a + s\rangle \ .$$

Note that by applying $H^{\otimes n}$, which is QFT for $\mathbb{F}_2^n$, to the state $|A_{s,s'}\rangle$, one obtains exactly $|A^\perp_{s',s}\rangle$.

Additionally, note that given $|A\rangle$ and s, s', one can efficiently construct $|A_{s,s'}\rangle$ as follows:

$$\sum_a |a\rangle \xrightarrow{\text{add } s} \sum_a |a+s\rangle \xrightarrow{H^{\otimes n}} \sum_{a' \in A^\perp} (-1)^{\langle a', s\rangle} |a'\rangle$$
$$\xrightarrow{\text{adding } s'} \sum_{a' \in A^\perp} (-1)^{\langle a', s\rangle} |a' + s'\rangle \xrightarrow{H^{\otimes n}} \sum_{a \in A} (-1)^{\langle a, s'\rangle} |a + s\rangle$$

For a subspace A and vectors s, s', we define $A + s = \{v + s : v \in A\}$, and $A^\perp + s' = \{v + s' : v \in A^\perp\}$.

It is also convenient for later sections to define a canonical representative, with respect to subspace A, of the coset $A + s$.

Definition 4.3 (Canonical representative of a coset)**.** *For a subspace A, we define the function $\mathsf{Can}_A(\cdot)$ such that $\mathsf{Can}_A(s)$ is the lexicographically smallest vector contained in $A + s$ (we call this the canonical representative of coset $A + s$).*

Note that if $\tilde{s} \in A + s$, then $\mathsf{Can}_A(s) = \mathsf{Can}_A(\tilde{s})$. Also note that Can_A is polynomial-time computable given the description of A. The algorithm to compute Can_A is the following:

1. Initialize the answer to be empty.
2. In the first step, let the first entry of the answer be 0 and check if a vector starting with 0 is in $A + s$. This can be done efficiently by solving a linear system (by knowing A and s). If such a vector is not in $A + s$, let the first entry of the answer be 1.
3. Iterate the same procedure for all entries, and output the answer.

When it is clear from the context, for ease of notation, we will write $A + s$ to mean the *program* that checks membership in $A + s$. For example, we will often write $\mathsf{iO}(A + s)$ to mean an iO obfuscation of the program that checks membership in $A + s$.

The following equivalences, which follow straightforwardly from the security of iO, will be useful in our security proofs later on.

Lemma 4.4. *For any subspace $A \subseteq \mathbb{F}_2^n$,*

- $\mathsf{iO}(A + s) \approx_c \mathsf{iO}(\mathsf{shO}_A(\cdot - s))$,

  where $\mathsf{shO}_A()$ denotes the program $\mathsf{shO}(A)$, and shO is the subspace hiding obfuscator defined in Section 3.4. So, $\mathsf{shO}_A(\cdot - s)$ is the program that on input x, runs program $\mathsf{shO}(A)$ on input $x - s$.

- $\mathsf{iO}(A + s) \approx_c \mathsf{iO}(\mathsf{CC}[\mathsf{Can}_A, \mathsf{Can}_A(s)])$,

  where recall that $\mathsf{CC}[\mathsf{Can}_A, \mathsf{Can}_A(s)]$ refers to the compute-and-compare program which on input x outputs 1 if and only if $\mathsf{Can}_A(x) = \mathsf{Can}_A(s)$.

4.2 Direct Product Hardness

In this section, we argue that coset states satisfy both an information-theoretic and a computational direct product hardness property.

4.2.1 Information-Theoretic Direct Product Hardness

Theorem 4.5. *Let $A \subseteq \mathbb{F}_2^n$ be a uniformly random subspace of dimension $n/2$, and s, s' be uniformly random in $\mathbb{F}_2^n$. Let $\epsilon > 0$ be such that $1/\epsilon = o(2^{n/2})$. Given one copy of $|A_{s,s'}\rangle$, and a quantum membership oracle for $A + s$ and $A^\perp + s'$, an adversary needs $\Omega(\sqrt{\epsilon}2^{n/2})$ queries to output a pair (v, w) such that $v \in A + s$ and $w \in A^\perp + s'$ with probability at least ϵ.*

The proof is a simple random self-reduction to the analogous statement from Ben-David and Sattath [BS16] for regular subspace states. The proof is given in Section 4.2.3.

4.2.2 Computational direct product hardness

Next, we present the computational version of the direct product hardness property. This establishes that Theorem 4.5 still holds, even if an adversary is given iO obfuscations of the subspace membership checking programs.

Theorem 4.6. *Assume the existence of post-quantum iO and one-way function. Let $A \subseteq \mathbb{F}_2^n$ be a uniformly random subspace of dimension $n/2$, and s, s' be uniformly random in $\mathbb{F}_2^n$. Given one copy of $|A_{s,s'}\rangle$, $\mathsf{iO}(A+s)$ and $\mathsf{iO}(A^\perp + s')$, any polynomial time adversary outputs a pair (v, w) such that $v \in A + s$ and $w \in A^\perp + s'$ with negligible probability.*

4.2.3 Proof of Theorem 4.5

We first present the theorem from Ben-David and Sattath [BS16].

Theorem 4.7 ([BS16])**.** *Let $A \subseteq \mathbb{F}_2^n$ be a uniformly random subspace of dimension $n/2$, and let $\epsilon > 0$ be such that $1/\epsilon = o(2^{n/2})$. Given one copy of $|A\rangle$, and a quantum membership oracle for A and $A^\perp$, an adversary needs $\Omega(\sqrt{\epsilon}2^{n/2})$ queries to output a pair (v, w) such that $v \in A \setminus \{0\}$ and $w \in A^\perp \setminus \{0\}$ with probability ϵ.*

Proof of Theorem 4.5. Let $\mathcal{A}$ be an adversary for Theorem 4.5 who suceeds with probability p, we construct an adversary $\mathcal{A}'$ for Theorem 4.7 with almost the same success probability making the same number of queries. $\mathcal{A}'$ proceeds as follows.

- $\mathcal{A}'$ receives $|A\rangle$ for some $A \subseteq \mathbb{F}_2^n$. Samples s, s' uniformly at random, and creates the state $|A_{s,s'}\rangle$.
- $\mathcal{A}'$ gives $|A_{s,s'}\rangle$ as input to $\mathcal{A}$. $\mathcal{A}$ also needs to get access to oracle $A + s$ and $A^\perp + s'$. $\mathcal{A}'$ can simulate them by having access to $A, A^\perp$ and knowing s, s'. It receives v, w in return from $\mathcal{A}$. $\mathcal{A}'$ outputs $(v - s, w - s')$.

With probability p, $\mathcal{A}$ returns v, w such that $v \in A + s$ and $w \in A^\perp + s'$. Thus the output of $\mathcal{A}'$ $(v - s, w - s')$ is such that $v - s \in A$ and $w - s' \in A^\perp$. All that is left to argue is that with overwhelming probability $v - s \neq 0$ and $w - s \neq 0$. Note that there are $2^{n/2} \cdot 2^{n/2}$ pairs $(\tilde{s}, \tilde{s}')$ such that $|A_{\tilde{s},\tilde{s}'}\rangle = |A_{s,s'}\rangle$, since translating s and s' by an element in A and $A^\perp$ respectively does not affect the state. Note further that only $2^{n/2+1} - 1$ pairs are such that $v - \tilde{s} = 0$ or $w - \tilde{s}' = 0$. Since s and s' are sampled uniformly at random, the probability that $v - s = 0$ or $w - s' = 0$ is $\frac{2^{n/2+1}-1}{2^n}$, which is negligible. □

4.2.4 Proof of Theorem 4.6

Proof. We consider the following hybrids.

- Hyb 0: This is the game of Theorem 4.6: $A \subseteq \mathbb{F}_2^n$, s, s' are sampled uniformly at random. $\mathcal{A}$ receives $\mathsf{iO}(A+s)$, $\mathsf{iO}(A^\perp + s')$, and $|A_{s,s'}\rangle$. $\mathcal{A}$ wins if it returns $(v,w) \in (A+s) \times (A^\perp + s')$.
- Hyb 1: Same as Hyb 0 except $\mathcal{A}$ gets $\mathsf{iO}(\mathsf{shO}_A(\cdot - s))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$. Recall that shO_A is the program $\mathsf{shO}(A)$, and so $\mathsf{shO}_A(\cdot - s))$ is the program that on input x, runs program $\mathsf{shO}(A)$ on input $x - s$.
- Hyb 2: Same as Hyb 1 except $\mathcal{A}$ gets $\mathsf{iO}(\mathsf{shO}_B(\cdot - s))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$, for a uniformly random superspace B of A, of dimension $3/4n$.
- Hyb 3: Same as Hyb 2 except for the following. The challenger samples s, s', A, and a uniformly random superspace B of A as before. The challenger sets $t = s + w_B$, where $w_B \leftarrow B$. Sends $\mathsf{iO}(\mathsf{shO}_B(\cdot - t))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$ to $\mathcal{A}$.
- Hyb 4: Same as Hyb 3 except $\mathcal{A}$ gets $\mathsf{iO}(\mathsf{shO}_B(\cdot - t))$, $\mathsf{iO}(\mathsf{shO}_{A^\perp}(\cdot - s'))$ and $|A_{s,s'}\rangle$.
- Hyb 5: Same as Hyb 4 except $\mathcal{A}$ gets $\mathsf{iO}(\mathsf{shO}_B(\cdot - t))$, $\mathsf{iO}(\mathsf{shO}_{C^\perp}(\cdot - s'))$ and $|A_{s,s'}\rangle$, for a uniformly random superspace $A^\perp \subseteq C^\perp$ of dimension $3n/4$.
- Hyb 6: Same as Hyb 5 except for the following. The challenger sets $t' = s' + w_{C^\perp}$, where $w_{C^\perp} \leftarrow C^\perp$. $\mathcal{A}$ gets $\mathsf{iO}(\mathsf{shO}_B(\cdot - t))$, $\mathsf{iO}(\mathsf{shO}_{C^\perp}(\cdot - t'))$ and $|A_{s,s'}\rangle$.
- Hyb 7: Same as Hyb 6 except the challenger sends B, C, t, t' in the clear to $\mathcal{A}$.

Claim 4.8. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \text{ wins in Hyb 1}] - \Pr[\mathcal{A} \text{ wins in Hyb 0}]| = \mathsf{negl}(\lambda)\,.$$

Proof. Suppose for a contradiction there was a QPT adversary $\mathcal{A}$ such that:

$$|\Pr[\mathcal{A} \text{ wins in Hyb 1}] - \Pr[\mathcal{A} \text{ wins in Hyb 0}]| \tag{1}$$

is non-negligible. Such an adversary can be used to construct $\mathcal{A}'$ which distinguishes $\mathsf{iO}(A+s)$ from $\mathsf{iO}(\mathsf{shO}_A(\cdot - s))$, which is impossible by the security of the (outer) iO, since $A + s$ and $\mathsf{shO}_A(\cdot - s)$ compute the same functionality.

Fix n, let $A \subseteq \mathbb{F}_2^n$, $s, s' \in \mathbb{F}_2^n$ be such that the difference in (1) is maximized. Suppose $\Pr[\mathcal{A} \text{ wins in Hyb 1}] > \Pr[\mathcal{A} \text{ wins in Hyb 0}]$, the other case being similar.

$\mathcal{A}'$ proceeds as follows:

- Receives as a challenge a circuit P which is either $\mathsf{iO}(A+s)$ or $\mathsf{iO}(\mathsf{shO}_A(\cdot - s))$. Creates the state $|A_{s,s'}\rangle$. Gives P, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$ as input to $\mathcal{A}$.
- $\mathcal{A}$ returns a pair (v,w). If $v \in A+s$ and $w \in A^\perp + s'$, then $\mathcal{A}'$ guesses that $P = \mathsf{iO}(\mathsf{shO}_A(\cdot - s))$, otherwise that $P = \mathsf{iO}(A+s)$.

It is straightforward to verify that $\mathcal{A}'$ succeeds at distinguishing with non-negligible probability. □

Claim 4.9. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \textit{ wins in Hyb 2}] - \Pr[\mathcal{A} \textit{ wins in Hyb 1}]| = \mathsf{negl}(\lambda)\,.$$

Proof. Suppose for a contradiction there was a QPT adversary $\mathcal{A}$ such that:

$$|\Pr[\mathcal{A} \text{ wins in Hyb 2}] - \Pr[\mathcal{A} \text{ wins in Hyb 1}]|\,,$$

is non-negligible.

We argue that $\mathcal{A}$ can be used to construct an adversary $\mathcal{A}'$ that breaks the security of shO. Fix n. Suppose $\Pr[\mathcal{A} \text{ wins in Hyb 2}] > \Pr[\mathcal{A} \text{ wins in Hyb 1}]$, the other case being similar. $\mathcal{A}'$ proceeds as follows:

- Sample $A \subseteq \mathbb{F}_2^n$ uniformly at random. Send A to the challenger.
- The challenger returns a program P which is either shO_A or shO_B. $\mathcal{A}'$ samples uniformly $s, s' \in \mathbb{F}_2^n$, and creates the state $|A_{s,s'}\rangle$. Gives $\mathsf{iO}(P(\cdot - s))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$ as input to $\mathcal{A}$.
- $\mathcal{A}$ returns a pair (v, w). If $v \in A + s$ and $w \in A^\perp + s'$, then $\mathcal{A}'$ guesses that $P = \mathsf{shO}_B$, otherwise that $P = \mathsf{shO}_A$.

It is straightforward to verify that $\mathcal{A}'$ succeeds at the security game for shO with non-negligible advantage. □

Claim 4.10. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \textit{ wins in Hyb 3}] - \Pr[\mathcal{A} \textit{ wins in Hyb 2}]| = \mathsf{negl}(\lambda)\,.$$

Proof. The proof is similar to the proof of Lemma 4.8, and follows from the security of iO and the fact that $\mathsf{shO}_B(\cdot - s)$ and $\mathsf{shO}_B(\cdot - t)$ compute the same functionality. This is because for any vector $w_B \in B$, $B + w_b$ is the same subspace as B. □

Claim 4.11. *For any QPT adversary $\mathcal{A}$, and $j = 4, 5, 6$, we have*

$$|\Pr[\mathcal{A} \textit{ wins in Hyb j}] - \Pr[\mathcal{A} \textit{ wins in Hyb (j-1)}]| = \mathsf{negl}(\lambda)$$

Proof. The proofs are analogous to those of Lemmas 4.8, 4.9, 4.10. □

Lemma 4.12. *For any QPT adversary $\mathcal{A}$ for Hyb 6, there exists an adversary $\mathcal{A}'$ for Hyb 7 such that*

$$\Pr[\mathcal{A}' \textit{ wins in Hyb 7}] \geq \Pr[\mathcal{A} \textit{ wins in Hyb 6}]\,.$$

Proof. This is immediate. □

Lemma 4.13. *For any (unbounded) adversary $\mathcal{A}$,*

$$\Pr[\mathcal{A} \textit{ wins in Hyb 7}] = \mathsf{negl}(\lambda)\,.$$

Proof. Suppose there exists an adversary $\mathcal{A}$ for Hyb 7 that wins with probability p.

We first show that, without loss of generality, one can take B to be the subspace of vectors such that the last $n/4$ entries are zero (and the rest are free), and one can take C to be such that the last $3/4n$ entries are zero (and the rest are free). We construct the following adversary $\mathcal{A}'$ for the game where B and C have the special form above with trailing zeros, call these B_* and C_*, from an adversary $\mathcal{A}$ for the game of Hyb 7.

- $\mathcal{A}'$ receives a state $|A_{s,s'}\rangle$, together with t and t', for some $C_* \subseteq A \subseteq B_*$, where $t = s + w_{B_*}$ for $w_{B_*} \leftarrow B_*$, and $t' = s' + w_{C_*^\perp}$, where $w_{C_*^\perp} \leftarrow C_*^\perp$.
- $\mathcal{A}'$ picks uniformly random subspaces B and C of dimension $\frac{3}{4}n$ and $\frac{n}{4}$ respectively such that $C \subseteq B$, and a uniformly random isomorphism $\mathcal{T}$ mapping C_* to C and B_* to B (which can be sampled efficiently). We think of $\mathcal{T}$ as a change-of-basis matrix (in particular when we take its transpose). $\mathcal{A}'$ applies to $|A_{s,s'}\rangle$ the unitary $U_\mathcal{T}$ which acts as $\mathcal{T}$ on the standard basis elements. $\mathcal{A}'$ gives $U_\mathcal{T}|A\rangle$ to $\mathcal{A}$ together with B, C, $\mathcal{T}(t)$ and $(\mathcal{T}^{-1})^T(t')$. $\mathcal{A}'$ receives a pair (v, w) from $\mathcal{A}$. $\mathcal{A}'$ outputs $(\mathcal{T}^{-1}(v), \mathcal{T}^T(w))$.

First, notice that

$$\begin{aligned} U_\mathcal{T}|A_{s,s'}\rangle &= U_\mathcal{T} \sum_{v\in A} (-1)^{\langle v,s'\rangle} |v+s\rangle \\ &= \sum_{v\in A} (-1)^{\langle v,s'\rangle} |\mathcal{T}(v)+\mathcal{T}(s)\rangle \\ &= \sum_{w\in\mathcal{T}(\mathcal{A})} (-1)^{\langle \mathcal{T}^{-1}(w),s'\rangle} |w+\mathcal{T}(s)\rangle \\ &= \sum_{w\in\mathcal{T}(A)} (-1)^{\langle w,(\mathcal{T}^{-1})^T(s')\rangle} |w+\mathcal{T}(s)\rangle \\ &= |\mathcal{T}(A)_{z,z'}\rangle \ , \end{aligned}$$

where $z = \mathcal{T}(s)$ and $z' = (\mathcal{T}^{-1})^T(s')$.

Notice that $\mathcal{T}(A)$ is a uniformly random subspace between C and B, and that z and z' are uniformly random vectors in $\mathbb{F}_2^n$. Moreover, we argue that:

(i) $\mathcal{T}(t)$ is distributed as a uniformly random element of $z + B$.

(ii) $(\mathcal{T}^{-1})^T(t')$ is distributed as a uniformly random element of $z' + C^\perp$.

For (i), notice that

$$\mathcal{T}(t) = \mathcal{T}(s + w_{B_*}) = \mathcal{T}(s) + \mathcal{T}(w_{B_*}) = z + \mathcal{T}(w_{B_*}) \, ,$$

where w_{B_*} is uniformly random in B_*. Since $\mathcal{T}$ is an isomorphism with $\mathcal{T}(B_*) = B$, then $\mathcal{T}(w_{B_*})$ is uniformly random in B. Thus, $\mathcal{T}(t)$ is distributed as a uniformly random element in $z + B$.

For (ii), notice that

$$(\mathcal{T}^{-1})^T(t') = (\mathcal{T}^{-1})^T(s' + w_{C_*^\perp}) = (\mathcal{T}^{-1})^T(s') + (\mathcal{T}^{-1})^T(w_{C_*^\perp}) = z' + (\mathcal{T}^{-1})^T(w_{C_*^\perp}) \, ,$$

where $w_{C_*^\perp}$ is uniformly random in $C_*^\perp$. We claim that $(\mathcal{T}^{-1})^T(w_{C_*^\perp})$ is uniformly random in $C^\perp$. Notice, first, that the latter belongs to $C^\perp$. Let $x \in C$, then

$$\langle (\mathcal{T}^{-1})^T(w_{C_*^\perp}), x\rangle = \langle w_{C_*^\perp}, \mathcal{T}^{-1}(x)\rangle = 0\,,$$

where the last equality follows because $w_{C_*^\perp} \in C_*^\perp$, and $\mathcal{T}^{-1}(C) = C_*$. The claim follows from the fact that $(\mathcal{T}^{-1})^T$ is a bijection.

Hence, $\mathcal{A}$ receives inputs from the correct distribution, and thus, with probability p, $\mathcal{A}$ returns a pair (v, w) such that $v \in \mathcal{T}(A) + z$ and $w \in \mathcal{T}(A)^\perp + z'$, where $z = \mathcal{T}(s)$ and $z' \in (\mathcal{T}^{-1})^T(s')$. $\mathcal{A}'$ returns $(v', w') = (\mathcal{T}^{-1}(v), \mathcal{T}^T(w))$.

Notice that:

- If $v \in \mathcal{T}(A) + z$, where $z = \mathcal{T}(s)$, then $\mathcal{T}^{-1}(v) \in A + s$.
- If $w \in \mathcal{T}(A)^\perp + z'$, where $z' \in (\mathcal{T}^{-1})^T(s')$, then $\mathcal{T}^T(w) \in A^\perp + s'$. This is because, for any $u \in A$:
$$\langle \mathcal{T}^T(w), u\rangle = \langle w, \mathcal{T}(u)\rangle = 0\,.$$

Thus, with probability p, $\mathcal{A}'$ returns a pair (v', w') where $v' \in A + s$ and $w' \in A^\perp + s'$, as desired.

So, we can now assume that B is the space of vectors such that the last $\frac{n}{4}$ entries are zero, and C is the space of vectors such that the last $\frac{3}{4}n$ entries are zero. Notice then that the sampled subspace A is uniformly random subspace subject to the last $\frac{n}{4}$ entries being zero, and the first $\frac{n}{4}$ entries being free. From an adversary $\mathcal{A}$ for Hybrid 7 with such B and C, we will construct an adversary $\mathcal{A}'$ for the information-theoretic direct-product game where the ambient subspace is $\mathbb{F}_2^{n'}$, where $n' = \frac{n}{2}$.

- $\mathcal{A}'$ receives $|A_{s,s'}\rangle$, for uniformly random $A \subseteq \mathbb{F}_2^{n'}$ of dimension $n'/2$ and uniformly random $s, s' \in \mathbb{F}_2^{n'}$. $\mathcal{A}'$ samples $\tilde{s}, \tilde{s}', \hat{s}, \hat{s}' \leftarrow \mathbb{F}_2^{\frac{n}{4}}$.
  Let $|\phi\rangle = \frac{1}{2^{n/8}} \sum_{x\in\{0,1\}^{n/4}} (-1)^{\langle x, \tilde{s}'\rangle} |x + \tilde{s}\rangle$. $\mathcal{A}'_0$ creates the state
  $$|W\rangle = |\phi\rangle \otimes |A_{s,s'}\rangle \otimes |\hat{s}\rangle\ ,$$
  $\mathcal{A}'$ gives to $\mathcal{A}$ as input the state $|W\rangle$, together with $t = 0^{3n/4}||\hat{s} + w_B$ for $w_B \leftarrow B$ and $t' = \hat{s}'||0^{3n/4} + w_{C^\perp}$, for $w_{C^\perp} \leftarrow C^\perp$. $\mathcal{A}$ returns a pair $(v, w) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$. Let $v' = [v]_{\frac{n}{4}+1, \frac{3}{4}n} \in \mathbb{F}_2^{n/2}$ be the "middle" $n/2$ entries of v. Let $w' = [w]_{\frac{n}{4}+1, \frac{3}{4}n} \in \mathbb{F}_2^{n/2}$. $\mathcal{A}'$ outputs (v', w').

Notice that

$$\begin{aligned}
|W\rangle &= |\phi\rangle \otimes |A_{s,s'}\rangle \otimes |\hat{s}\rangle \\
&= \sum_{x\in\{0,1\}^{n/4}, v\in A} (-1)^{\langle x, \tilde{s}'\rangle} (-1)^{\langle v, s'\rangle} \Big| (x + \tilde{s})||(v + s)||\hat{s}\Big\rangle \\
&= \sum_{x\in\{0,1\}^{n/4}, v\in A} (-1)^{\langle (x||v||0^{n/4}), (\tilde{s}'||s'||\hat{s}')\rangle} \Big| x||v||0^{n/4} + \tilde{s}||s||\hat{s}\Big\rangle \\
&= \sum_{w\in\tilde{A}} (-1)^{\langle w, z'\rangle} |w + z\rangle = |\tilde{A}_{z,z'}\rangle\ ,
\end{aligned}$$

where $z = \tilde{s}||s||\hat{s}$, $z' = \tilde{s}'||s'||\hat{s}'$, and $\tilde{A} \subseteq \mathbb{F}_2^n$ is the subspace in which the first $n/4$ entries are free, the middle $n/2$ entries belong to subspace A, and the last $n/4$ entries are zero (notice that there is a freedom for the choice of s' in the above calculation).

Notice that the subspace $\tilde{A}$, when averaging over the choice of A, is distributed precisely as in the game of Hybrid 7 (with the special choice of B and C); z, z' are uniformly random in $\mathbb{F}_2^n$; t is uniformly random from $z + B$, and t' is uniformly random from $z' + C^\perp$. Thus, with probability p, $\mathcal{A}$ returns to $\mathcal{A}'$ a pair (v, w) such that $v \in \tilde{A} + z$ and $w \in \tilde{A}^\perp + z'$. It follows that, with probability p, the answer (v', w') returned by $\mathcal{A}'$ is such that $v' \in A + s$ and $w' \in A^\perp + s'$.

Thus, by Theorem 4.5, we deduce that p must be negligible.

□

Therefore, we have shown that the advantage in distinguishing Hybrid 0 and Hybrid 6 is negligible, and the success probability in Hybrid 6 is at most the success probability in Hybrid 7, which is negligible). Hence, the probability of success in the original game is also negligible. □

4.3 Monogamy-of-Entanglement Property

In this subsection, we argue that coset states satisfy an information-theoretic and a computational monogamy-of-entanglement property. We will not make use of these properties directly, instead we will have to rely on a stronger conjectured monogamy-of-entanglement property, which is presented in subsection 4.4. Thus, the properties that we prove in this subsection serve merely as "evidence" in support of the stronger conjecture.

4.3.1 Information-Theoretic Monogamy-of-Entanglement

Let $n \in \mathbb{N}$. Consider the following game between a challenger and an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$.

- The challenger picks a uniformly random subspace $A \subseteq \mathbb{F}_2^n$ of dimension $\frac{n}{2}$, and two uniformly random elements $s, s' \in \mathbb{F}_2^n$. Sends $|A_{s,s'}\rangle$ to $\mathcal{A}_0$.
- $\mathcal{A}_0$ creates a bipartite state on registers B and C. Then, $\mathcal{A}_0$ sends register B to $\mathcal{A}_1$, and C to $\mathcal{A}_2$.
- The description of A is then sent to both $\mathcal{A}_1, \mathcal{A}_2$.
- $\mathcal{A}_1$ and $\mathcal{A}_2$ return respectively (s_1, s_1') and (s_2, s_2').

$(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ wins if, for $i \in \{1, 2\}$, $s_i \in A + s$ and $s_i' \in A^\perp + s'$.

Let $\mathsf{ITMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n)$ be a random variable which takes the value 1 if the game above is won by adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, and takes the value 0 otherwise. We have the following theorem.

Theorem 4.14. *There exists a sub-exponential function* subexp *such that, for any (unbounded) adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$\Pr[\mathsf{ITMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n) = 1] \leq 1/\mathsf{subexp}(n)\,.$$

We refer the reader to Appendix C.1 for the proof.

4.3.2 Computational monogamy

We describe a computational version of the monogamy game from the previous section. In the computational version, $\mathcal{A}_0$ additionally receives the programs $\mathsf{iO}(A+s)$ and $\mathsf{iO}(A'+s')$. The game is between a challenger and an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$.

- The challenger picks a uniformly random subspace $A \subseteq \mathbb{F}^n$ of dimension $\frac{n}{2}$, and two uniformly random elements $s, s' \in \mathbb{F}_2^n$. It sends $|A_{s,s'}\rangle$, $\mathsf{iO}(A+s)$, and $\mathsf{iO}(A^\perp + s')$ to $\mathcal{A}_0$.
- $\mathcal{A}_0$ creates a bipartite state on registers B and C. Then, $\mathcal{A}_0$ sends register B to $\mathcal{A}_1$, and C to $\mathcal{A}_2$.
- The description of A is then sent to both $\mathcal{A}_1, \mathcal{A}_2$.
- $\mathcal{A}_1$ and $\mathcal{A}_2$ return respectively (s_1, s_1') and (s_2, s_2').

$(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ wins if, for $i \in \{1,2\}$, $s_i \in A + s$ and $s_i' \in A^\perp + s'$.

Let $\mathsf{CompMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n)$ be a random variable which takes the value 1 if the game above is won by adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, and takes the value 0 otherwise.

Theorem 4.15. *Assume the existence of post-quantum* iO *and one-way function, there exists a negligible function* $\mathsf{negl}(\cdot)$*, for any QPT adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$\Pr[\mathsf{CompMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n) = 1] = \mathsf{negl}(n)\,.$$

The proof is very similar to the proof of Theorem 4.6. We refer the reader to Appendix C.2 for the full details.

4.4 Conjectured Strong Monogamy Property

In this section, we describe a stronger version of the monogamy property, which we conjecture to hold. The monogamy property is a slight (but significant) variation of the one stated in the last section (which we proved to be true). Recall that there $\mathcal{A}_1$ and $\mathcal{A}_2$ are required to return pairs (s_1, s_1') and (s_2, s_2') respectively, such that both $s_1, s_2 \in A + s$ and $s_1', s_2' \in A^\perp + s'$. Now, we require that it is hard for $\mathcal{A}_1$ and $\mathcal{A}_2$ to even return a single string s_1 and s_2 respectively such that $s_1 \in A + s$ and $s_2 \in A^\perp + s'$.

Formally, consider the following game between a challenger and an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$.

- The challenger picks a uniformly random subspace $A \subseteq \mathbb{F}_2^n$ of dimension $\frac{n}{2}$, and two uniformly random elements $s, s' \in \mathbb{F}_2^n$. It sends $|A_{s,s'}\rangle$ to $\mathcal{A}_0$.
- $\mathcal{A}_0$ creates a bipartite state on registers B and C. Then, $\mathcal{A}_0$ sends register B to $\mathcal{A}_1$, and C to $\mathcal{A}_2$.
- The description of A is then sent to both $\mathcal{A}_1, \mathcal{A}_2$.
- $\mathcal{A}_1$ and $\mathcal{A}_2$ return respectively s_1 and s_2.

Let $\mathsf{ITStrongMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n)$ be a random variable which takes the value 1 if the game above is won by adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, and takes the value 0 otherwise. We conjecture the following:

Conjecture 4.16. *There exists a sub-exponential function* subexp *such that, for any (unbounded) adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$\Pr[\mathsf{ITStrongMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n) = 1] \leq 1/\mathsf{subexp}(n)\,.$$

Assuming the conjecture is true, and assuming post-quantum iO and one-way functions, we are able to prove the following computational strong monogamy statement. Consider a game between a challenger and an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, which is identical to the one described above except that all $\mathcal{A}_0$ additionally gets the membership checking programs $\mathsf{iO}(A + s)$ and $\mathsf{iO}(A^\perp + s')$.

- The challenger picks a uniformly random subspace $A \subseteq \mathbb{F}_2^n$ of dimension $\frac{n}{2}$, and two uniformly random elements $s, s' \in \mathbb{F}_2^n$. It sends $|A_{s,s'}\rangle$, $\mathsf{iO}(A + s)$, and $\mathsf{iO}(A^\perp + s')$ to $\mathcal{A}_0$.
- $\mathcal{A}_0$ creates a bipartite state on registers B and C. Then, $\mathcal{A}_0$ sends register B to $\mathcal{A}_1$, and C to $\mathcal{A}_2$.
- The description of A is then sent to both $\mathcal{A}_1, \mathcal{A}_2$.
- $\mathcal{A}_1$ and $\mathcal{A}_2$ return respectively s_1 and s_2.

$(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ wins if, for $s_1 \in A + s$ and $s_2 \in A^\perp + s'$.

Let $\mathsf{CompStrongMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n)$ be a random variable which takes the value 1 if the game above is won by adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, and takes the value 0 otherwise.

Theorem 4.17. *Assuming Conjecture 4.16 holds, and assuming the existence of post-quantum* iO *and one-way functions, then there exists a negligible function* $\mathsf{negl}(\cdot)$*, for any QPT adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$\Pr[\mathsf{CompStrongMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n) = 1] = \mathsf{negl}(n)\,.$$

We can further show a 'sub-exponential strong monogamy property' if we additionally assume sub-exponentially secure iO and one-way functions.

Theorem 4.18. *Assuming Conjecture 4.16 holds, and assuming the existence of sub-exponentially secure post-quantum* iO *and one-way functions, then for any QPT adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$\Pr[\mathsf{CompStrongMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), n) = 1] \leq 1/\mathsf{subexp}(n)\,.$$

The proof is almost identical to that of Theorem 4.15, therefore we omit the proof here and refer to the proof of Theorem 4.15.

In the rest of the work, whenever we mention 'strong monogamy property' or 'strong monogamy-of-entanglement property', we refer to the computational monogamy property in Theorem 4.17 above. Whenever we mention 'sub-exponentially strong monogamy property' or 'sub-exponentially strong monogamy-of-entanglement property', we refer to the computational monogamy property in Theorem 4.18.

5 Tokenized Signature Scheme from iO

In this section, we present a construction for tokenized signatures with unforgeability security based on the computational direct product hardness(Theorem 4.6). We improved upon the scheme in [BS16] by removing the need of (highly structured) oracles or post-quantum VBB obfuscation.

5.1 Definitions

Definition 5.1 (Tokenized signature scheme)**.** *A tokenized signature (TS) scheme consists of a tuple of QPT algorithms* $(\mathsf{KeyGen}, \mathsf{TokenGen}, \mathsf{Sign}, \mathsf{Verify})$ *with the following properties:*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$*: Takes as input* 1^λ*, where* λ *is a security parameter, and outputs a secret key, public (verification) key pair* $(\mathsf{sk}, \mathsf{pk})$*.*
- $\mathsf{TokenGen}(\mathsf{sk}) \to |\mathsf{tk}\rangle$*: Takes as input a secret key* sk *and outputs a signing token* $|\mathsf{tk}\rangle$*.*
- $\mathsf{Sign}(m, |\mathsf{tk}\rangle) \to (m, \mathsf{sig})/\bot$*: Takes as input a message* $m \in \{0,1\}^*$ *and a token* $|\mathsf{tk}\rangle$*, and outputs either a message, signature pair* (m, sig) *or* $\bot$*.*
- $\mathsf{Verify}(\mathsf{pk}, m, \mathsf{sig}) \to 0/1$*: Takes as input an verification key, an alleged message, signature pair* (m, sig)*, and outputs* 0 *("reject") or* 1 *("accept").*
- $\mathsf{Revoke}(\mathsf{pk}, |\mathsf{tk}\rangle) \to 0/1$*: Takes in public key* pk *and a claimed token* $|\mathsf{tk}\rangle$*, and outputs* 0 *("reject") or* 1 *("accept").*

These algorithms satisfy the following. First is correctness. There exists a negligible function $\mathsf{negl}(\cdot)$, for any $\lambda \in \mathbb{N}$, $m \in \{0,1\}^*$,

$$\Pr[\mathsf{Verify}(\mathsf{pk}, m, \mathsf{sig}) = 1 : (m, \mathsf{sig}) \leftarrow \mathsf{Sign}(m, |\mathsf{tk}\rangle), |\mathsf{tk}\rangle \leftarrow \mathsf{TokenGen}(\mathsf{sk}), (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)] \geq 1 - \mathsf{negl}(\lambda)\,.$$

Definition 5.2 (Length restricted TS scheme)**.** *A TS scheme is* r*-restricted if it holds only for* $m \in \{0,1\}^r$*. We refer to a scheme that is* 1*-restricted as a one-bit TS scheme.*

Notation-wise, we introduce an additional algorithm Verify_ℓ. The latter takes as input a public key pk and ℓ pairs $(m_\ell, \mathsf{sig}_\ell), \ldots, (m_\ell, \mathsf{sig}_\ell)$. It checks that $m_i \neq m_j$ for all $i \neq j$, and $\mathsf{Verify}(m_i, \mathsf{sig}_i) = 1$ for all $i \in [\ell]$; it outputs 1 if and only if they all hold.

Next we define unforgeability.

Definition 5.3 (1-Unforgeability)**.** *A TS scheme is* 1*-unforgeable if for every QPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, for every* λ*:*

$$\Pr\left[\begin{array}{c} (m_0, \mathsf{sig}_0, m_1, \mathsf{sig}_1) \leftarrow \mathcal{A}(\mathsf{pk}, |\mathsf{tk}\rangle) \\ \mathsf{Verify}_2(\mathsf{pk}, m_0, \mathsf{sig}_0, m_1, \mathsf{sig}_1) = 1 \end{array} : \begin{array}{c} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ |\mathsf{tk}\rangle \leftarrow \mathsf{TokenGen}(\mathsf{sk}) \end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

Definition 5.4 (Unforgeability)**.** *A TS scheme is unforgeable if for every QPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, for every* λ*,* $l = \mathsf{poly}(\lambda)$*:*

$$\Pr\left[\begin{array}{c} \{m_i, \mathsf{sig}_i\}_{i\in[l+1]} \leftarrow \mathcal{A}(\mathsf{pk}, \{|\mathsf{tk}_i\rangle\}_{i\in[l]}) \\ \mathsf{Verify}_{l+1}(\mathsf{pk}, \{m_i, \mathsf{sig}_i\}_{i\in[l+1]}) = 1 \end{array} : \begin{array}{c} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ |\mathsf{tk}_1\rangle \leftarrow \mathsf{TokenGen}(\mathsf{sk}) \\ \vdots \\ |\mathsf{tk}_l\rangle \leftarrow \mathsf{TokenGen}(\mathsf{sk}) \end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

Finally we have revocability.

Definition 5.5 (Revocability)**.** *A revocable tokenized signature scheme satisfies:*

- *Correctness:*

  $$\Pr\left[\mathsf{Revoke}(\mathsf{pk}, |\mathsf{tk}\rangle) = 1 \middle| (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda), |\mathsf{tk}\rangle \leftarrow \mathsf{TokenGen}(\mathsf{sk})\right] = 1.$$

- *Revocability: For every $\ell \leq \mathsf{poly}(\lambda), t \leq \ell$, and every QPT $\mathcal{A}$ with ℓ signing tokens $|\mathsf{tk}_1\rangle \otimes \cdots \otimes |\mathsf{tk}_\ell\rangle$ and pk, which has generated t signatures $(m_1, \mathsf{sig}_1), \cdots, (m_t, \mathsf{sig}_t)$ and a state σ:*

  $$\Pr\left[\mathsf{Verify}_t(\mathsf{pk}, (m_1, \mathsf{sig}_1), \cdots, (m_t, \mathsf{sig}_t)) = 1 \wedge \mathsf{Revoke}_{\ell-t+1}(\sigma) = 1\right] \leq \mathsf{negl}(\lambda)$$

  Here $\mathsf{Revoke}_{\ell-t+1}$ means applying Revoke on all $\ell - t + 1$ registers of σ, and outputs 1 if they all output 1.

The revocability property follows straightforwardly from unforgeability [BS16]. Thus to show a construction is secure, we only need to focus on proving unforgeability. The following theorem says 1-unforgeability is sufficient to achieve a full blown TS scheme.

Theorem 5.6 ([BS16])**.** *A one-bit 1-unforgeable TS scheme implies a (full blown) TS scheme, assuming the existence of a quantum-secure digital signature scheme.*

In the next section, we give our construction of a one-bit 1-unforgeable TS scheme from coset states.

5.2 Tokenized Signature Construction

Construction.

- $\mathsf{KeyGen}(1^\lambda)$*: Set $n = \mathsf{poly}(\lambda)$. Sample uniformly $A \subseteq \mathbb{F}_2^n$. Sample $s, s' \leftarrow \mathbb{F}_2^n$. Output $\mathsf{sk} = (A, s, s')$ (where by A we mean a description of the subspace A) and $\mathsf{pk} = (\mathsf{iO}(A + s), \mathsf{iO}(A^\perp + s'))$.*
- $\mathsf{TokenGen}(\mathsf{sk})$*: Takes as input sk of the form (A, s, s'). Outputs $|\mathsf{tk}\rangle = |A_{s,s'}\rangle$.*
- $\mathsf{Sign}(m, |\mathsf{tk}\rangle)$*: Takes as input $m \in \{0, 1\}$ and a state $|\mathsf{tk}\rangle$ on n qubits. Compute $H^{\otimes n}|\mathsf{tk}\rangle$ if $m = 1$, otherwise do nothing to the quantum state. It then measures in the standard basis. Let sig be the outcome. Output (m, sig).*
- $\mathsf{Verify}(\mathsf{pk}, (m, \mathsf{sig}))$*: Parse pk as $\mathsf{pk} = (C_0, C_1)$ where C_0 and C_1 are circuits. Output $C_m(\mathsf{sig})$.*
- $\mathsf{Revoke}(\mathsf{pk}, |\mathsf{tk}\rangle)$*: Parse pk as $\mathsf{pk} = (C_0, C_1)$. Then:*
  - *Coherently compute C_0 on input $|\mathsf{tk}\rangle$, and measure the output of the circuit. If the latter is 1, uncompute C_0, and proceed to the next step. Otherwise halt and output 0.*
  - *Apply $H^{\otimes n}$. Coherently compute C_1 and measure the output of the circuit. If the latter is 1, output 1.*

Theorem 5.7. *Assuming post-quantum iO and one-way function, the scheme of Construction 5.2 is a one-bit 1-unforgeable tokenized signature scheme.*

Proof. Security follows immediately from Theorem 4.6. □

Corollary 5.8. *Assuming post-quantum iO, one-way function(which implies digital signature) and a quantum-secure digital signature scheme, there exists a (full blown) tokenized signature scheme.*

Proof. This is an immediate consequence of Theorems 5.6 and 5.7. □

6 Single-Decryptor Encryption

In this section, we formally introduce unclonable decryption, i.e. single-decryptor encryption [GZ20]. Then we describe two constructions and prove their security.

Our first construction (Section 6.3) relies on the strong monogamy-of-entanglement property (Conjecture 4.16), the existence of post-quantum one-way function, indistinguishability obfuscation and compute-and-compare obfuscation for (sub-exponentially) unpredictable distributions (whose existence has been discussed in Section 3.3 and Appendix B). Our second construction (Section 6.5) has a similar structure. It does not rely on the strong monogamy-of-entanglement property for coset states, but on the (weaker) direct product hardness property (Theorem 4.6). However, the construction additionally relies on a much stronger cryptographic primitive – post-quantum extractable witness encryption (as well post-quantum one-way functions and indistinguishability obfuscation).

6.1 Definitions

Definition 6.1 (Single-Decryptor Encryption Scheme)**.** *A single-decryptor encryption scheme consists of the following efficient algorithms:*

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$: *a (classical) probabilistic algorithm that takes as input a security parameter λ and outputs a classical secret key sk and public key pk.*
- $\mathsf{QKeyGen}(\mathsf{sk}) \to \rho_{\mathsf{sk}}$: *a quantum algorithm that takes as input a secret key sk and outputs a quantum secret key ρ_{sk}.*
- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$: *a (classical) probabilistic algorithm that takes as input a public key pk, a message m and outputs a classical ciphertext ct.*
- $\mathsf{Dec}(\rho_{\mathsf{sk}}, \mathsf{ct}) \to m/\bot$: *a quantum algorithm that takes as input a quantum secret key ρ_{sk} and a ciphertext ct, and outputs a message m or a decryption failure symbol $\bot$.*

A secure single-decryptor encryption scheme should satisfy the following:

Correctness: There exists a negligible function $\mathsf{negl}(\cdot)$, for all $\lambda \in \mathbb{N}$, for all $m \in \mathcal{M}$,

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

Note that correctness implies that a honestly generated quantum decryption key can be used to decrypt correctly polynomially many times, from the gentle measurement lemma [Aar05].

CPA Security: The scheme should satisfy (post-quantum) CPA security, i.e. indistinguishability under chosen-plaintext attacks: for every (stateful) QPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{c} \mathcal{A}(\mathsf{ct}) = b : \\ \end{array} \begin{array}{c} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ ((m_0, m_1) \in \mathcal{M}^2) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pk}) \\ b \leftarrow \{0,1\}; \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

Anti-Piracy Security Next, we define anti-piracy security via the anti-piracy game below. Recall that, intuitively, anti-piracy security says that it is infeasible for a pirate who receives a quantum secret key to produce two quantum keys, which both allow successful decryption. This can be formalized into ways:

- (*CPA-style anti-piracy*) We can ask the pirate to provide a pair of messages (m_0, m_1) along with two quantum secret keys, and we test whether the two keys allow to (simultanoeusly) distinguish encryptions of m_0 and m_1.
- (*random challenge anti-piracy*) We do *not* ask the pirate to provide a pair of plaintext messages, but only a pair of quantum secret keys, and we test whether the two quantum secret keys allow for simultaneous decryption of encryptions of uniformly random messages.

The reader might expect that, similarly to standard definitions of encryption security, the former implies the latter, i.e. that CPA-security (it is infeasible to distinguish encryptions of chosen plaintexts with better than negligible advantage) implies that it is infeasible to decrypt uniformly random challenges with non-negligible probability. However, for the case of anti-piracy security, this implication does not hold, as we explain in more detail in Appendix D.4. This subtlety essentially arises due to the fact that there are two parties involved, having to simultaneously make the correct guess. Therefore, we will state both definitions here, and we will later argue that our construction satisfies both.

In Section 6.2, we will introduce an even stronger definition of CPA-style anti-piracy (and a stronger definition for random challenge anti-piracy in Appendix D.3). We will eventually prove that our constructions satisfy both of the strong definitions. We chose to start our presentation of unclonable decryption with the definitions in this section since they are much more intuitive than the stronger version of Section 6.2.

In order to describe the security games, it is convenient to first introduce the concept of a *quantum decryptor*. The following definition is implicitly with respect to some single-decryptor encryption scheme $(\mathsf{Setup}, \mathsf{QKeyGen}, \mathsf{Enc}, \mathsf{Dec})$.

Definition 6.2 (Quantum decryptor)**.** *A* quantum decryptor *for ciphertexts of length n, is a pair (ρ, U) where ρ is a state, and U is a general quantum circuit acting on $n + m$ qubits, where m is the number of qubits of ρ.*

For a ciphertext c of length n, we say that we run the quantum decryptor (ρ, U) on ciphertext c to mean that we execute the circuit U on inputs $|c\rangle$ and ρ.

We are now ready to describe the CPA-style anti-piracy game.

Definition 6.3 (Anti-Piracy Game, CPA-style)**.** *Let $\lambda \in \mathbb{N}^+$. The CPA-style anti-piracy game is the following game between a challenger and an adversary $\mathcal{A}$.*

1. ***Setup Phase**: The challenger samples keys $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$.*
2. ***Quantum Key Generation Phase**: The challenger sends $\mathcal{A}$ the classical public key pk and one copy of quantum decryption key $\rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$.*
3. ***Output Phase**: $\mathcal{A}$ outputs a pair of distinct messages (m_0, m_1). It also outputs a (possibly mixed and entangled) state σ over two registers R_1, R_2 and two general quantum circuits U_1 and U_2. We interpret $\mathcal{A}$'s output as two (possibly entangled) quantum decryptors $\mathsf{D}_1 = (\sigma[R_1], U_1)$ and $\mathsf{D}_2 = (\sigma[R_2], U_2)$.*

4. ***Challenge Phase:*** *The challenger samples b_1, b_2 and r_1, r_2 uniformly at random and generates ciphertexts $c_1 = \mathsf{Enc}(\mathsf{pk}, m_{b_1}; r_1)$ and $c_2 = \mathsf{Enc}(\mathsf{pk}, m_{b_2}; r_2)$. The challenger runs quantum decryptor D_1 on c_1 and D_2 on c_2, and checks that D_1 outputs m_{b_1} and D_2 outputs m_{b_2}. If so, the challenger outputs 1 (the game is won by the adversary), otherwise outputs 0.*

We denote by $\mathsf{AntiPiracyCPA}(1^\lambda, \mathcal{A})$ a random variable for the output of the game.

Note that an adversary can succeed in this game with probability at least 1/2. It simply gives ρ_{sk} to the first quantum decryptor and the second decryptor randomly guesses the plaintext.

We remark that one could have equivalently formulated this definition by having the pirate send registers R_1 and R_2 to two separated parties Bob and Charlie, who then receive ciphertexts from the challenger sampled as in the Challenge Phase above. The two formulations are equivalent upon identifying the quantum circuits U_1 and U_2.

Definition 6.4 (Anti-Piracy Security, CPA-style)**.** *Let $\gamma : \mathbb{N}^+ \to [0, 1]$. A single-decryptor encryption scheme satisfies γ-anti-piracy security, if for any QPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that the following holds for all $\lambda \in \mathbb{N}$:*

$$\Pr\left[b = 1, b \leftarrow \mathsf{AntiPiracyCPA}(1^\lambda, \mathcal{A})\right] \leq \frac{1}{2} + \gamma(\lambda) + \mathsf{negl}(\lambda) \tag{2}$$

Unless specified otherwise, when discussing anti-piracy security of an unclonable encryption scheme in this work, we refer to CPA-style anti-piracy security.

It is not difficult to show that if γ-anti-piracy security holds for all inverse poly γ, then this directly implies CPA security (we refer the reader to the appendix (Appendix D.1) for the proof of this implication).

Next, we define an anti-piracy game with *random challenge plaintexts*. This quantifies how well an efficient adversary can produce two "quantum decryptors" both of which enable successful decryption of encryptions of uniformly random plaintexts. This security notion will be directly useful in the security proof for copy-protection of PRFs in Section 7.

Definition 6.5 (Anti-Piracy Game, with random challenge plaintexts)**.** *Let $\lambda \in \mathbb{N}^+$. The anti-piracy game with random challenge plaintexts is the following game between a challenger and an adversary $\mathcal{A}$.*

1. ***Setup Phase****: The challenger samples keys $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$.*
2. ***Quantum Key Generation Phase****: The challenger sends $\mathcal{A}$ the classical public key pk and one copy of quantum decryption key $\rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$.*
3. ***Output Phase****: $\mathcal{A}$ outputs a (possibly mixed and entangled) state σ over two registers R_1, R_2 and two general quantum circuits U_1 and U_2. We interpret $\mathcal{A}$'s output as two (possibly entangled) quantum decryptors $\mathsf{D}_1 = (\sigma[R_1], U_1)$ and $\mathsf{D}_2 = (\sigma[R_2], U_2)$.*
4. ***Challenge Phase:*** *The challenger samples $m_1, m_2 \leftarrow \mathcal{M}$ and r_1, r_2 uniformly at random, and generates ciphertexts $c_1 = \mathsf{Enc}(\mathsf{pk}, m_1; r_1)$ and $c_2 = \mathsf{Enc}(\mathsf{pk}, m_2; r_2)$. The challenger runs quantum decryptor D_1 on c_1 and D_2 on c_2, and checks that D_1 outputs m_1 and D_2 outputs m_2. If so, the challenger outputs 1 (the game is won by the adversary), otherwise outputs 0.*

We denote by $\mathsf{AntiPiracyGuess}(1^\lambda, \mathcal{A})$ *a random variable for the output of the game.*

Note that an adversary can succeed in this game with probability at least $1/|\mathcal{M}|$. The adversary simply gives ρ_{sk} to the first quantum decryptor and the second decryptor randomly guesses the plaintext.

Definition 6.6 (Anti-Piracy Security, with random challenge plaintexts)**.** *Let* $\gamma : \mathbb{N}^+ \to [0,1]$*. A single-decryptor encryption scheme satisfies* γ*-anti-piracy security with random challenge plaintexts, if for any QPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that the following holds for all* $\lambda \in \mathbb{N}$*:*

$$\Pr\left[b = 1, b \leftarrow \mathsf{AntiPiracyGuess}(1^\lambda, \mathcal{A})\right] \leq \frac{1}{|\mathcal{M}|} + \gamma(\lambda) + \mathsf{negl}(\lambda) \tag{3}$$

where $\mathcal{M}$ *is the message space.*

Remark 6.7. *In the rest of the section, we will mainly focus on Definition 6.4 and the stronger version of it from the next section. We will appeal to Definition 6.6 when we prove security of our copy-protection scheme for PRFs.*

6.2 Strong Anti-Piracy Security

The stronger definition of anti-piracy security that we introduce in this section is more technically involved, and less intuitive, than the definitions in the previous section, but is easier to work with when proving security of our constructions. This section relies on preliminary concepts introduced in Section 3.6. We will refer to the anti-piracy security notions defined in the previous section as *regular* anti-piracy" to distinguish them from *strong* anti-piracy defined in this section.

In order to describe the anti-piracy game in this section, we first need to formalize a procedure to test good quantum decryptors and the notion of a *good quantum decryptor*. Again, the following definitions are implicitly with respect to some single-decryptor encryption scheme $(\mathsf{Setup}, \mathsf{QKeyGen}, \mathsf{Enc}, \mathsf{Dec})$.

We first describe a procedure to test good quantum decryptors. The procedure is parametrized by a threshold value γ. We are guaranteed that, if the procedure passes, then the post-measurement state is a γ-good decryptor.

Definition 6.8 (Testing a quantum decryptor)**.** *Let* $\gamma \in [0,1]$*. Let* pk *be a public key, and* (m_0, m_1) *a pair of messages. We refer to the following procedure as a test for a* γ*-good quantum decryptor with respect to* pk *and* (m_0, m_1)*:*

- *The procedure takes as input a quantum decryptor* (ρ, U)*.*
- *Let* $\mathcal{P} = (P, I - P)$ *be the following mixture of projective measurements (in the sense of Definition 3.14) acting on some quantum state* ρ'*:*
  - *Sample a uniform* $b \leftarrow \{0,1\}$*. Compute* $c \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$*.*
  - *Run the quantum decryptor* (ρ', U) *on input* c*. Check whether the outcome is* m_b*. If so, output* 1*, otherwise output* 0*.*
- *Let* $\mathsf{TI}_{1/2+\gamma}(\mathcal{P})$ *be the threshold implementation of* $\mathcal{P}$ *with threshold value* $\frac{1}{2} + \gamma$*, as defined in Definition 3.12. Run* $\mathsf{TI}_{1/2+\gamma}(\mathcal{P})$ *on* ρ*, and output the outcome. If the output is* 1*, we say that the test passed, otherwise the test failed.*

By Lemma 3.13, we have the following corollary.

Corollary 6.9 (γ-good Decryptor)**.** *Let $\gamma \in [0,1]$. Let (ρ, U) be a quantum decryptor. Let $\mathsf{TI}_{1/2+\gamma}(\mathcal{P})$ be the test for a γ-good decryptor defined above. Then, the post-measurement state conditioned on output 1 is a mixture of states which are in the span of all eigenvectors of P with eigenvalues at least $1/2+\gamma$. We refer to the latter state as a γ-good decryptor with respect to (m_0, m_1).*

Now we are ready to define the strong γ-anti-piracy game.

Definition 6.10 (Strong Anti-Piracy Game)**.** *Let $\lambda \in \mathbb{N}^+$, and $\gamma \in [0,1]$. The strong γ-anti-piracy game is the following game between a challenger and an adversary $\mathcal{A}$.*

1. ***Setup Phase**: The challenger samples keys $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$.*
2. ***Quantum Key Generation Phase**: The challenger sends $\mathcal{A}$ the classical public key pk and one copy of quantum decryption key $\rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$.*
3. ***Output Phase**: $\mathcal{A}$ outputs a pair of distinct messages (m_0, m_1). It also outputs a (possibly mixed and entangled) state σ over two registers R_1, R_2 and two general quantum circuits U_1 and U_2. We interpret $\mathcal{A}$'s output as two (possibly entangled) quantum decryptors $\mathsf{D}_1 = (\sigma[R_1], U_1)$ and $\mathsf{D}_2 = (\sigma[R_2], U_2)$.*
4. ***Challenge Phase**: The challenger runs the test for a γ-good decryptor with respect to pk and (m_0, m_1) on D_1 and D_2. The challenger outputs 1 if both tests pass, otherwise outputs 0.*

We denote by $\mathsf{StrongAntiPiracy}(1^\lambda, \gamma, \mathcal{A})$ a random variable for the output of the game.

Definition 6.11 (Strong Anti-Piracy-Security)**.** *Let $\gamma : \mathbb{N}^+ \to [0,1]$. A single-decryptor encryption scheme satisfies strong γ-anti-piracy security, if for any QPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that the following holds for all $\lambda \in \mathbb{N}$:*

$$\Pr\left[b = 1, b \leftarrow \mathsf{StrongAntiPiracy}(1^\lambda, \gamma(\lambda), \mathcal{A})\right] \leq \mathsf{negl}(\lambda) \tag{4}$$

Definition 6.11 implies Definition 6.4.

Theorem 6.12. *Let $\gamma : \mathbb{N}^+ \to [0,1]$. Suppose a single-decryptor encryption scheme satisfies strong γ-anti-piracy security (Definition 6.11). Then, it also satisfies γ-anti-piracy security (Definition 6.4).*

Proof. We refer the reader to appendix (Appendix D.2) for the proof. □

In a similar way, one can define a stronger version of random challenge anti-piracy security (Definition 6.6). We leave the details to (Appendix D.3).

6.3 Construction from Strong Monogamy Property

In this section, we give our first construction of a single-decryptor encryption scheme, whose security relies on the strong monogamy-of-entanglement property from Section 4.4.

In the rest of the paper, to simplify notation, whenever it is clear from the context, we will denote a program that checks membership in a set S simply by S.

Construction 1.

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$:
  - *Sample κ random $(n/2)$-dimensional subspaces $A_i \subseteq \mathbb{F}_2^n$ for $i = 1, 2, \cdots, \kappa$, where $n = \lambda$ and $\kappa = \kappa(\lambda)$ is a polynomial in λ.*
  - *For each $i \in [\kappa]$, choose two uniformly random vectors $s_i, s'_i \in \mathbb{F}_2^n$.*
  - *Prepare the programs $\mathsf{iO}(A_i + s_i)$ and $\mathsf{iO}(A_i^\perp + s'_i)$ (where we assume that the programs $A_i + s_i$ and $A_i^\perp + s'_i$ are padded to some appropriate length).*
  - *Output $\mathsf{sk} = \{A_i, s_i, s'_i\}_{i\in[\kappa]}, \mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A_i^\perp + s'_i)\}_{i\in[\kappa]}$.*
- $\mathsf{QKeyGen}(\mathsf{sk}) \to \rho_{\mathsf{sk}}$: *on input $\mathsf{sk} = \{A_i, s_i, s'_i\}_{i\in[\kappa]}$, output the "quantum secret key" $\rho_{\mathsf{sk}} = \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\kappa]}$. Recall that each $|A_{i,s_i,s'_i}\rangle$ is*

$$|A_{i,s_i,s'_i}\rangle = \frac{1}{\sqrt{|A_i|}} \sum_{a\in A_i} (-1)^{\langle a, s'_i\rangle} |a + s_i\rangle .$$

- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$: *on input a public key $\mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A_i^\perp + s'_i)\}_{i\in[\kappa]}$ and message m:*
  - *Sample a uniformly random string $r \leftarrow \{0,1\}^\kappa$.*
  - *Let r_i be the i-th bit of r. Define $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$. Let $P_{m,r}$ be the following program:*

> On input $u = u_1||u_2||\cdots||u_\kappa$ (where each $u_i \in \mathbb{F}_2^n$):
> 1. If for all $i \in [\kappa]$, $R_i^{r_i}(u_i) = 1$:
> Output m
> 2. Else:
> Output $\perp$

Figure 1: Program $P_{m,r}$

  - *Let $\hat{P}_{m,r} = \mathsf{iO}(P_{m,r})$. Output ciphertext $\mathsf{ct} = (\hat{P}_{m,r}, r)$.*
- $\mathsf{Dec}(\rho_{\mathsf{sk}}, \mathsf{ct}) \to m/\perp$: *on input $\rho_{\mathsf{sk}} = \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\kappa]}$ and $\mathsf{ct} = (\hat{P}_{m,r}, r)$:*
  - *For each $i \in [\kappa]$, if $r_i = 1$, apply $H^{\otimes n}$ to the i-th state $|A_{i,s_i,s'_i}\rangle$; if $r_i = 0$, leave the i-th state $|A_{i,s_i,s'_i}\rangle$ unchanged. Denote the resulting state by ρ^*_{sk}.*
  - *Evaluate the program $\hat{P}_{m,r}$ on input ρ^*_{sk} in superposition; measure the evaluation register and denote the outcome by m'. Output m'.*
  - *Rewind by applying the operations in the first step again.*

Correctness. Honest evaluation applies $H^{\otimes n}$ to $|A_{i,s_i,s'_i}\rangle$ whenever $r_i = 1$. Clearly, the coherent evaluation of $\mathsf{iO}(A_i + s_i)$ on $|A_{i,s_i,s'_i}\rangle$ always outputs 1, and likewise the coherent evaluation of $\mathsf{iO}(A_i^{\perp} + s'_i)$ on $H^{\otimes n}|A_{i,s_i,s'_i}\rangle$ also always outputs 1. Therefore, by definition of $\hat{\mathsf{P}}_{m,r}$, the evaluation $\hat{\mathsf{P}}_{m,r}(\rho^*_{\mathsf{sk}})$ outputs m with probability 1.

Theorem 6.13 (Strong Anti-Piracy)**.** *Assuming the existence of post-quantum* iO*, one-way functions, compute-and-compare obfuscation for the class of unpredictable distributions (as in Definition 3.6), and the strong monogamy-of-entanglement property (Conjecture 4.16), the single-decryptor encryption scheme of Construction 1 has strong* γ*-anti-piracy security for any inverse polynomial* γ*.*

Similarly, assuming the existence of post-quantum sub-exponentially secure iO *and one-way functions, the quantum hardness of LWE and assuming the strong monogamy-of-entanglement property (Conjecture 4.16), the single-decryptor encryption scheme of Construction 1 has strong* γ*-anti-piracy security for any inverse polynomial* γ*.*

In the above theorem, ELFs and the quantum hardness of LWE are for building the corresponding compute-and-compare obfuscation (see Theorem B.2 and Theorem B.1). We will prove this theorem in Section 6.4.

We remark that this does *not* immediately imply that there exists a negligible γ such that strong γ-anti-piracy holds. The slightly subtle reason is that the parameter γ in strong γ-anti-piracy is actually a parameter of the security game (rather than a measure of the success probability of an adversary in the game).

From Theorem 6.12, we know that strong γ-anti-piracy security implies regular γ-anti-piracy security. Thus, for any inverse-polynomial γ, the scheme of Construction 1 has regular γ-anti-piracy security. For regular anti-piracy security, it is straightforward to see that a scheme that satisfies the notion for any inverse-polynomial γ, also satisfies it for $\gamma = 0$. Thus, we have the following.

Corollary 6.14 (Regular Anti-Piracy)**.** *Assuming the existence of post-quantum* iO*, one-way functions, compute-and-compare obfuscation for the class of unpredictable distributions (as in Definition 3.6), and the strong monogamy-of-entanglement property (Conjecture 4.16), the single-decryptor encryption scheme of Construction 1 has regular* γ*-anti-piracy security for* $\gamma = 0$*.*

Similarly, assuming the existence of post-quantum sub-exponentially secure iO *and one-way functions, the quantum hardness of LWE and assuming the strong monogamy-of-entanglement property (Conjecture 4.16), the single-decryptor encryption scheme of Construction 1 has regular* γ*-anti-piracy security for* $\gamma = 0$*.*

As mentioned earlier, it is not clear whether anti-piracy security, CPA-style (Definition 6.3) implies anti-piracy with random challenge inputs (Definition 6.5). Thus, we will also separately prove the latter, since in Section 7 we will reduce security of our PRF copy-protection scheme to it.

Theorem 6.15 (Regular Anti-Piracy, For Random Challenge Plaintexts)**.** *Assuming the existence of post-quantum* iO*, one-way functions, compute-and-compare obfuscation for the class of unpredictable distributions (as in Definition 3.6), and the strong monogamy-of-entanglement property (Conjecture 4.16), the single-decryptor encryption scheme has* γ*-anti-piracy security against random challenge plaintexts for* $\gamma = 0$*.*

Similarly, assuming the existence of post-quantum sub-exponentially secure iO *and one-way functions, the quantum hardness of LWE and assuming the strong monogamy-of-entanglement property*

(Conjecture 4.16), the single-decryptor encryption scheme has γ-anti-piracy security against random challenge plaintexts for $\gamma = 0$.

Proof. We refer the reader to Appendix D.3. The proof follows a similar outline as the proof of CPA-style anti-piracy. □

6.4 Proof of Strong Anti-Piracy Security of Construction 1

In this section, we prove Theorem 6.13. We only focus on the first half of the theorem, as the second half follows the same outline of the first one. The only differences between them are:

- They either base on strong monogamy-of-entanglement or *sub-exponentially* strong monogamy-of-entanglement.
- Thus, they rely on either compute-and-compare obfuscation for any unpredictable distribution (post-quantum ELFs) or sub-exponentially unpredictable distributions (the quantum hardness of LWE).

The proof proceeds via two hybrids. We will mark changes between consecutive hybrids in red. We denote the advantage of adversary $\mathcal{A}$ in Hybrid i by $\mathsf{Adv}_{\mathcal{A},i}$. Let γ be any inverse-polynomial.

Hybrid 0. This is the strong γ-anti-piracy game from Definition 6.10:

1. The challenger samples $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and sends pk to adversary $\mathcal{A}$. Let $\mathsf{sk} = \{A_i, s_i, s'_i\}_{i\in[\kappa]}$ and $\mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A_i^\perp + s'_i)\}_{i\in[\kappa]}$.
2. The challenger prepares the quantum key $\rho_{\mathsf{sk}} = \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\kappa]} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$, and sends ρ_{sk} to $\mathcal{A}$.
3. $\mathcal{A}$ produces a quantum state σ over two registers R_1, R_2, unitaries U_1, U_2. and a pair of messages (m_0, m_1). Sends this information to the challenger.
4. For $i \in \{1, 2\}$, let $\mathcal{P}_{i,D}$ be the following mixture of projective measurements acting on some quantum state ρ':
   - Sample a uniform $b \leftarrow \{0, 1\}$. Compute $c \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$.
   - Run the quantum decryptor (ρ', U_i) on input c. Check whether the outcome is m_b. If so, output 1, otherwise output 0.

   Formally, let D be the distribution over pairs (b, c) of (bit, ciphertext) defined in the first bullet point, and let $\mathcal{P}_i = \{M^i_{(b,c)}\}_{b,c}$ be a collection of projective measurements where $M^i_{(b,c)}$ is the projective measurement described in the second bullet point. Then, $\mathcal{P}_{i,D}$ is the mixture of projective measurements associated to D and $\mathcal{P}_i$ (as in Definition 3.14).
5. The challenger runs $\mathsf{TI}_{\frac{1}{2}+\gamma}(\mathcal{P}_{1,D})$ and $\mathsf{TI}_{\frac{1}{2}+\gamma}(\mathcal{P}_{2,D})$ on quantum decryptors $(\sigma[R_1], U_1)$ and $(\sigma[R_2], U_2)$ respectively. $\mathcal{A}$ wins if both measurements output 1.

Hybrid 1. Hybrid 1 is the same as Hybrid 0, except in step 5 the challenger runs the *approximate* threshold implementations $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D,\frac{1}{2}+\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D,\frac{1}{2}+\gamma}$, where $\epsilon = \frac{\gamma}{4}$, and δ is some negligible function of λ.

1. The challenger samples $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and sends pk to adversary $\mathcal{A}$. Let $\mathsf{sk} = \{A_i, s_i, s'_i\}_{i\in[\kappa]}$ and $\mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A_i^\perp + s'_i)\}_{i\in[\kappa]}$.
2. The challenger prepares the quantum key $\rho_{\mathsf{sk}} = \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\kappa]} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$, and sends ρ_{sk} to $\mathcal{A}$.
3. $\mathcal{A}$ produces a quantum state σ over two registers R_1, R_2, unitaries U_1, U_2. and a pair of messages (m_0, m_1). Sends this information to the challenger.
4. Let $\mathcal{P}_{1,D}$ and $\mathcal{P}_{2,D}$ be as in Hybrid 0.
5. The challenger runs $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D,\frac{1}{2}+\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D,\frac{1}{2}+\gamma}$ on $(\sigma[R_1], U_1)$ and $(\sigma[R_2], U_2)$. $\mathcal{A}$ wins if both measurements output 1.

By Lemma 3.17, we have $\mathsf{Adv}_{\mathcal{A},1} \geq \mathsf{Adv}_{\mathcal{A},0} - 2\delta$. Moreover, by Lemma 3.16, for each $i \in \{1,2\}$, $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_i,D,\frac{1}{2}+\gamma}$ runs in time $\mathsf{poly}(\log(1/\delta), 1/\epsilon)$. The latter is polynomial for our choice of ϵ and δ.

We complete the proof of Theorem 6.13 by showing that the advantage $\mathsf{Adv}_{\mathcal{A},1}$ in Hybrid 1 is negligible.

Lemma 6.16. $\mathsf{Adv}_{\mathcal{A},1}$ *is negligible.*

Proof. Suppose for a contradiction that $\mathsf{Adv}_{\mathcal{A},1}$ is non-negligible. Then, applying $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D,\frac{1}{2}+\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D,\frac{1}{2}+\gamma}$ on $\sigma[R_1]$ and $\sigma[R_2]$ results in two outcomes 1 with non-negligible probability. Let σ' be the bipartite state conditioned on both outcomes being 1.

From the second bullet point of Lemma 3.17, we have the following:

$$\mathrm{Tr}\left[\left(\mathsf{TI}_{\frac{1}{2}+\gamma-2\epsilon}(\mathcal{P}_{1,D}) \otimes \mathsf{TI}_{\frac{1}{2}+\gamma-2\epsilon}(\mathcal{P}_{2,D})\right)\sigma'\right] \geq 1 - 4\delta, \tag{5}$$

where recall that, for ease of notation, when we write TI inside of a trace we are referring to the projection on the 1 outcome. The observation says that the collapsed state $\sigma'[R_1]$ is negligibly close to being a $(\gamma - 2\epsilon)$-good decryptor with respect to ciphertexts generated according to distribution D. Similarly for $\sigma'[R_2]$.

We then define a different but computationally close distribution D' over pairs (b, c) of (bit, ciphertext). Let (m_0, m_1) be the pair of messages chosen by $\mathcal{A}$.

1. Sample $b \leftarrow \{0,1\}$ and $r \leftarrow \{0,1\}^\kappa$.
2. Let $\mathsf{Can}_{i,0}(\cdot) = \mathsf{Can}_{A_i}(\cdot)$ and $\mathsf{Can}_{i,1}(\cdot) = \mathsf{Can}_{A_i^\perp}(\cdot)$ where $\mathsf{Can}_{A_i}(\cdot), \mathsf{Can}_{A_i^\perp}(\cdot)$ are the functions defined in Definition 4.3.
3. Define function f as follows:

$$f(u_1, \cdots, u_\kappa) = \mathsf{Can}_{1,r_1}(u_1)||\cdots||\mathsf{Can}_{\kappa,r_\kappa}(u_\kappa).$$

Let $s_{i,0} = s_i$ and $s_{i,1} = s'_i$. Let the "lock value" y be the following:

$$y = \mathsf{Can}_{1,r_1}(s_{1,r_1})||\cdots||\mathsf{Can}_{\kappa,r_\kappa}(s_{\kappa,r_\kappa}).$$

Let $C_{m_b,r}$ be the compute-and-compare program $\mathsf{CC}[f, y, m_b]$.

4. Run the obfuscation algorithm $\mathsf{CC.Obf}$ on $C_{m_b,r}$ and obtain the obfuscated program $\widetilde{\mathsf{CC}}_{m_b,r} = \mathsf{CC.Obf}(C_{m_b,r})$. Let $\widehat{\mathsf{CC}}_{m_b,r} = \mathsf{iO}(\widetilde{\mathsf{CC}}_{m_b,r})$.

5. Let $c = (\widehat{\mathsf{CC}}_{m_b,r}, r)$. Output (b, c).

Since the programs $C_{m_b,r}$ and $P_{m_b,r}$ (from Fig. 1) are functionally equivalent, the two distributions D and D' are computationally indistinguishable assuming post-quantum security of iO. A direct application of Theorem 3.15, together with (5), gives the following corollary. Let $\mathcal{P}_{1,D'}$ be the same mixture of projective measurements as $\mathcal{P}_{1,D}$, except that pairs of (bit, ciphertext) are sampled according to distribution D' instead of D.

Corollary 6.17. *Let D, D' be the distributions defined above. Let σ' be the post-measurement state conditioned on $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D,\frac{1}{2}+\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D,\frac{1}{2}+\gamma}$ both outputting 1 on $\sigma[R_1]$ and $\sigma[R_2]$. For any inverse polynomial ϵ', there exists a negligible function δ' such that:*

$$\mathrm{Tr}\left[\left(\mathsf{TI}_{\frac{1}{2}+\gamma-2\epsilon-\epsilon'}(\mathcal{P}_{1,D'}) \otimes \mathsf{TI}_{\frac{1}{2}+\gamma-2\epsilon-\epsilon'}(\mathcal{P}_{2,D'})\right)\sigma'\right] \geq 1 - 4\delta - \delta'.$$

Intuitively, the above corollary says that if $\sigma'[R_1]$ and $\sigma'[R_2]$ are both negligibly close to being $(\gamma - 2\epsilon)$-good decryptors with respect to ciphertexts generated according to D, then, for any inverse polynomial ϵ', they are also negligibly close to being $(\gamma - 2\epsilon - \epsilon')$-good decryptors with respect to ciphertexts generated according to D'. By setting $\epsilon' = \epsilon = \frac{\gamma}{4}$, we have that there exists a negligible function negl such that:

$$\mathrm{Tr}\left[\left(\mathsf{TI}_{\frac{1}{2}+\frac{\gamma}{4}}(\mathcal{P}_{1,D'}) \otimes \mathsf{TI}_{\frac{1}{2}+\frac{\gamma}{4}}(\mathcal{P}_{2,D'})\right)\sigma'\right] \geq 1 - \mathsf{negl}(\lambda). \tag{6}$$

The rest of the proof amounts to showing that all of the above implies that there exists an efficient algorithm breaking the computational strong monogamy-of-entanglement property. Before giving the full details, we provide a sketch of how the proof proceeds.

(i) First notice that if the "lock value" y, as defined in the description of D', is computationally unpredictable given the quantum decryptor $\sigma[R_1]$ (and the additional classical auxiliary information), then the security of compute-and-compare obfuscation implies that we can replace D' with a distribution D'' that contains no information at all about the plaintext (with respect to which no quantum decryptor can have any advantage beyond random guessing).

(ii) From (6), we know that conditioned on the (approximate) threshold implementation measurement accepting on both sides (which happens with non-negligible probability), each side is (close to) a $\frac{\gamma}{4}$-good decryptor with respect to D'. Notice that this implies the existence of an efficient algorithm that takes $\sigma[R_1]$ as auxiliary information, and distinguishes between D' and D''. By the security of compute-and-compare obfuscation, this implies that the lock value in the left ciphertext must be predictable given $\sigma[R_1]$ (and the classical auxiliary information). Similarly the lock value in the right ciphertext must be predictable given $\sigma[R_2]$ (and the classical auxiliary information).

(iii) Since lock values consist of concatenations of canonical representatives of either the coset $A_i + s_i$ or $A_i^\perp + s'_i$, one would like to conclude that it is possible to extract (with non-negligible probability) one representative on each side (i.e. one using the information in register R_1, and one using the information in register R_2). However, one has to be cautious, since this deduction does not work in general! In fact, successfully extracting on R_1's side might destroy the (entangled) quantum information on R_2's side, preventing a successful simultaneous extraction.

(iv) The key is that if each side is (close to) a $\frac{\gamma}{4}$-good decryptor, then no matter what measurement is performed on the left side, and no matter what outcome is obtained, the state on the right side is still in the support of $\frac{\gamma}{4}$-good decryptors. This means that extraction will still succeed with non-negligible probability on the right side. This implies a strategy that succeeds at extracting canonical representatives simultaneously on both sides. Finally, since for each i the choice of whether to encrypt using $A_i + s_i$ or $A_i^\perp + s'_i$ is independent and uniformly random, with overwhelming probability there will be some i such that the extracting algorithm will recover s_i and s'_i simultaneously, breaking the strong monogamy-of-entanglement property.

Extracting from register R_1. Let $\mathcal{P}_{1,D'} = (P_{1,D'}, I - P_{1,D'})$. Recall that from Equation (6) we have

$$\mathrm{Tr}[\mathsf{TI}_{1/2+\frac{\gamma}{4}}(\mathcal{P}_{1,D'})\,\sigma'[R_1]] \geq 1 - \mathsf{negl}(\lambda).$$

This implies that $\sigma'[R_1]$ has $1 - \mathsf{negl}(\lambda)$ weight over eigenvectors of $P_{1,D'}$ whose eigenvalues are at least $1/2 + \frac{\gamma}{4}$. Therefore,

$$\mathrm{Tr}[P_{1,D'}\,\sigma'[R_1]] \geq \frac{1}{2} + \frac{\gamma}{4} - \mathsf{negl}. \tag{7}$$

Hence, if we view $(\sigma'[R_1], U_1)$ as a quantum decryptor, its advantage on challenges sampled from D' is noticeably greater than random guessing.

Let Sim be an efficient simulator for the compute-and-compare obfuscation scheme that we employ (using the notation of Definition 3.6). We define D'' to be the following distribution over pairs (b, c) of (bit, ciphertext).

- Let $\widetilde{\mathsf{Sim}} = \mathsf{Sim}(1^\lambda, \mathsf{param})$ where param consists of the parameters of the compute-and-compare program being obfuscated in the description of D' (input size, output size, circuit size - these are the parameters of $C_{m_b,r}$).
- Let $c = (\mathsf{iO}(\widetilde{\mathsf{Sim}}), r)$. Output (b, c).

Because the simulated ciphertext generated in D'' is independent of m_b, the quantum decryptor $(\sigma'[R_1], U_1)$ cannot enable guessing b with better than $1/2$ probability. More concretely, let $\mathcal{P}_{1,D''} = (P_{1,D''}, P_{2,D''})$ be the mixture of projective measurements which is the same as $\mathcal{P}_{1,D'}$ except pairs of (bit, ciphertext) are sampled according to D'' instead of D'. Then,

$$\mathrm{Tr}[P_{1,D''}\,\sigma'[R_1]] = \frac{1}{2}. \tag{8}$$

Since the quantum decryptor $(\sigma'[R_1], U_1)$ behaves noticeably differently on distributions D' and D'', it can be used as a distinguisher for the two related distributions $\widehat{D}'$ and $\widehat{D}''$, defined as follows (these implicitly depend on some adversary $\mathcal{A}$):

1. $\widehat{D}'$: a distribution over pairs of programs and auxiliary information

$$(C, \mathsf{AUX}),$$

where $\mathsf{AUX} = (\mathsf{pk}, m_b, r, \sigma'[R_1], U_1)$, with the latter being sampled like the homonymous parameters in Hybrid 0, and $C = \widetilde{\mathsf{CC}}_{m_b,r}$ is an obfuscated compute-and-compare as in D'.

2. $\widehat{D}''$: a distribution over pairs of programs and auxiliary information

$$(C, \mathsf{AUX}),$$

where $\mathsf{AUX} = (\mathsf{pk}, m_b, r, \sigma'[R_1], U_1)$, with the latter being sampled like the homonymous parameters in Hybrid 0, and $C = \widetilde{\mathsf{Sim}}$ as in D''.

To distinguish $\widehat{D}'$ and $\widehat{D}''$, a distinguisher runs the quantum decryptor $(\sigma'[R_1], U_1)$ on input $(\mathsf{iO}(C), r)$ and checks whether the output is equal to m_b. By the definition of $\widehat{D}'$ and $\widehat{D}''$, it is straightforward to see that in the first case this procedure is equivalent to performing the measurement $\mathcal{P}_{1,D'}$ (recall that the latter depends on U_1) on $\sigma'[R_1]$, and in the second case this procedure is equivalent to performing the measurement $\mathcal{P}_{1,D''}$ on $\sigma'[R_1]$. By (7) and (8), the advantage of this distinguisher is at least $\frac{\gamma}{4} - \mathsf{negl}$, which is noticeable.

Thus, by the security of compute-and-compare obfuscation, the distribution $\widehat{D}'$ over pairs of (compute-and-compare program, auxiliary information) is not computationally unpredictable (as in Definition 3.4). In particular, the contrapositive of Definition 3.4 is that there exists an efficient algorithm $\mathcal{M}_1$ that succeeds at the following with non-negligible probability:

- Let $(\mathsf{CC}[f, y, m], \mathsf{AUX}) \leftarrow \widehat{D}'$.
- $\mathcal{M}_1$ receives f and AUX, and outputs y'. $\mathcal{M}_1$ is successful if $y' = y$.

In our case, for a fixed $\{A_i, s_i, s_i'\}_{i\in[\kappa]}$ and r, the function f is defined as:

$$f(u_1, \cdots, u_\kappa) = \mathsf{Can}_{1,r_1}(u_1)||\cdots||\mathsf{Can}_{\kappa,r_\kappa}(u_\kappa)$$

Notice that this function is efficiently computable given descriptions of the subspaces A_i. Thus, in our case, the contrapositive of Definition 3.4 says that there exists an adversary which receives the description of the function f and AUX, which in particular includes the description of the subspaces A_i, and is able to guess the appropriate coset representatives, depending on the bits of r. The existence of this adversary will be crucial in our reduction to an adversary for the computational strong monogamy-of-entanglement game. Notice that in the monogamy-of-entanglement game, each of the two parties $\mathcal{A}_1$ and $\mathcal{A}_2$ receives the descriptions of the subspaces A_i, but not of the cosets, which they have to guess.

Extracting on register R_2. If two registers are entangled, then performing measurements on one register will generally result in destruction of the quantum information on the other side. We show that this does not happen in our case, and that one can extract coset representatives from register R_1, in a way that the leftover state on R_2 also allows for extraction of coset representatives. Recall the definition of γ-good decryptor from Corollary 6.9. Informally, a quantum decryptor (ρ, U) is a γ-good decryptor with respect to a distribution D over pairs of (bit, ciphertext) if ρ is a mixture of states which are all in the span of eigenvectors of the γ-good decryptor test with respect to D, with eigenvalues greater than $\frac{1}{2} + \gamma$.

Claim 6.18. *Let ρ be a bipartite state on registers R_1 and R_2. Let U_1 and U_2 be general quantum circuits acting respectively on R_1 and R_2 (plus a register containing ciphertexts). Suppose that $(\rho[R_1], U_1)$ and $(\rho[R_2], U_2)$ are both γ-good decryptors with respect to a distribution D over pairs of (bit, ciphertext). Let M be any POVM on R_1. Then, the post-measurement state on R_2 (together with U_2) conditioned on any outcome is still a γ-good decryptor with respect to distribution D.*

Proof. Assume ρ is a pure state. The general statement follows from the fact that a mixed state is a convex mixture of pure states.

We can write ρ in an eigenbasis of products of eigenvectors of $P_{1,D}$ and $P_{2,D}$. The hypothesis that both $(\rho[R_1], U_1)$ and $(\rho[R_2], U_2)$ are both γ-good decryptors implies that ρ can be written as follows: $\rho = \sum_{i,j:p_i,q_j \geq 1/2+\gamma} \alpha_{i,j} |x_i\rangle \otimes |y_j\rangle$, where x_i is an eigenvector of $P_{1,D}$ with eigenvalue p_i and y_j is an eigenvector of $P_{2,D}$ with eigenvalue q_j. In particular, note that only the eigenvectors corresponding to eigenvalues $p_i, q_j \geq 1/2+\gamma$ have non-zero weight. Finally, notice that applying any POVM M on R_1 (or in general any quantum operation on R_1) does not change the support of the resulting traced out state on R_2: the support still consists of eigenvectors of $P_{2,D}$ with eigenvalues $\geq 1/2 + \gamma$.

□

Now, let σ' be as defined in Corollary 6.17. Then, Equation (6) implies that both $\sigma'[R_1]$ and $\sigma'[R_2]$ are negligibly close to being $\frac{\gamma}{4}$-good decryptors with respect to D'. By Claim 6.18 (together with simple triangle inequalities) this implies that, conditioned on algorithm $\mathcal{M}_1$ successfully outputting the lock value (which happens with non-negligible probability), the remaining state $\sigma''[R_2]$ is still a $\frac{\gamma}{4}$-good decryptor with respect to D'.

By the same argument as for $\mathcal{M}_1$, there exists an algorithm $\mathcal{M}_2$ that takes the description of the function f and auxiliary information $\mathsf{AUX} = (\mathsf{pk}, m_b, r, \sigma''[R_2], U_2)$ and outputs the lock value with non-negligible probability. Thus, $\mathcal{M}_1$ and $\mathcal{M}_2$ simultaneously output a lock value with non-negligible probability.

Breaking the strong monogamy-of-entanglement property. We now give a formal description of an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ breaking the strong monogamy-of-entanglement property (Theorem 4.17) of coset states, given an adversary $\mathcal{A}$ that breaks the γ-anti-piracy-game, and algorithms $\mathcal{M}_1$ and $\mathcal{M}_2$ as described above.

1. The challenger picks a uniformly random subspace $A \subseteq \mathbb{F}_2^n$ and two uniformly random elements $s, s' \in \mathbb{F}_2^n$. It sends $|A_{s,s'}\rangle$ and $\mathsf{iO}(A + s), \mathsf{iO}(A^\perp + s')$ to $\mathcal{A}_0$.

2. $\mathcal{A}_0$ simulates the game in Hybrid 1:

   - Samples $i^* \leftarrow [\kappa]$. Generates $\rho_{\mathsf{sk}} = \{|A_i, s_i, s_i'\rangle\}_{i\in[\kappa]}$ and $\mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A^\perp + s_i')\}_{i\in[\kappa]}$ where A_i, s_i, s_i' are uniformly random except $|A_{i^*,s_{i^*},s'_{i^*}}\rangle = |A_{s,s'}\rangle$, $\mathsf{iO}(A_{i^*} + s_{i^*}) = \mathsf{iO}(A + s)$ and $\mathsf{iO}(A_{i^*}^\perp + s'_{i^*}) = \mathsf{iO}(A^\perp + s')$.
   - $\mathcal{A}_0$ gives ρ_{sk} and pk to the adversary $\mathcal{A}$ for the γ-anti-piracy game. $\mathcal{A}_0$ obtains $\sigma[R_1], \sigma[R_2]$ together with U_1, U_2 and (m_0, m_1). $\mathcal{A}_0$ applies $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_1,D,\frac{1}{2}+\gamma}$ and $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}_2,D,\frac{1}{2}+\gamma}$ to σ, where D is the distribution over pairs of (bit, ciphertext) defined in Hybrid 0 (which is efficiently sampleable given pk.) If if any of the two outcomes is 0, $\mathcal{A}_0$ halts. If both outcomes are 1, let the collpased state be σ'.

It sends $\sigma'[R_1], \mathsf{pk}$ to $\mathcal{A}_1$ and $\sigma'[R_2], \mathsf{pk}$ to $\mathcal{A}_2$. Both $\mathcal{A}_1, \mathcal{A}_2$ also get the description of A_i (for all $i \neq i^*$).

3. The challenger gives the description of A (equivalently, A_{i^*}) to $\mathcal{A}_1$ and $\mathcal{A}_2$.

4. $\mathcal{A}_1$ samples m_{b_1,R_1} as a uniformly random message in $\{m_0, m_1\}$ and $r_{R_1} \leftarrow \{0,1\}^\kappa$. Let $\mathsf{AUX}_{R_1} = (\mathsf{pk}, m_{b_1,R_1}, r_{R_1}, \sigma'[R_1])$ and f_{R_1} be the function corresponding to r_{R_1}. Because $\mathcal{A}_1$ gets the description of all A_i, the description of f_{R_1} is efficiently computable. It runs $\mathcal{M}_1$ on $f_{R_1}, \mathsf{AUX}_{R_1}$ and gets the outcome y_{R_1}.

5. Similarly for $\mathcal{A}_2$, it prepares $f_{R_2}, \mathsf{AUX}_{R_2}$, runs $\mathcal{M}_2$ and gets the outcome y_{R_2}.

It follows from the previous analysis that, with non-negligible probability, both y_{R_1} and y_{R_2} are correct lock values. Since $r_{R_1,i^*} \neq r_{R_2,i^*}$ with overwhelming probability, this violates the strong monogamy-of-entanglement property.

□

Thus, the advantage in Hybrid 1 is negligible. This implies that the advantage in Hybrid 0 is also negligible, which concludes the proof of Theorem 6.13. □

6.5 Construction from Extractable Witness Encryption

In this section, we give an alternative construction of a single-decryptor encryption scheme. This construction uses a quantum signature token scheme as a black box. The construction is conceptually very similar to that of Section 6.3, but it uses extractable witness encryption instead of compute-and-compare obfuscation to deduce simultaneous extraction. Because the extraction guarantee from extractable witness encryption is stronger than the one from compute-and-compare obfuscation (we elaborate on this difference in Section 6.6), we do not need to reduce security of the scheme to the strong monogamy-of-entanglement property, but instead we are able to reduce security of the scheme to security of the signature token scheme (which, recall, is a primitive that we show how to construct using the computational direct product hardness property of coset states in Section 5).

In the following construction, let $\mathsf{WE} = (\mathsf{WE.Enc}, \mathsf{WE.Enc})$ be an extractable witness encryption scheme (as in Definition 3.9), and let $\mathsf{TS} = (\mathsf{TS.KeyGen}, \mathsf{TS.TokenGen}, \mathsf{TS.Sign}, \mathsf{TS.Verify})$ be an unforgeable signature token scheme (as in Definitions 5.1 and 5.4). The construction below works to encrypt single bit messages, but can be extended to messages of polynomial length without loss of generality.

Construction 2.

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$*: Let* $\kappa = \kappa(\lambda)$ *be a polynomial.*
  - *For each* $i \in [\kappa]$*, compute* $(\mathsf{TS.sk}_i, \mathsf{TS.pk}_i) \leftarrow \mathsf{TS.KeyGen}(1^\lambda)$.
  - *Output* $\mathsf{pk} = \{\mathsf{TS.pk}_i\}_{i\in[\kappa]}$ *and* $\mathsf{sk} = \{\mathsf{TS.sk}_i\}_{i\in[\kappa]}$.
- $\mathsf{QKeyGen}(\mathsf{sk}) \to \rho_{\mathsf{sk}}$: *On input* $\mathsf{sk} = \{\mathsf{TS.sk}_i\}_{i\in[\kappa]}$*:*
  - *For* $i \in [\kappa]$*, compute* $|\mathsf{tk}_i\rangle \leftarrow \mathsf{TS.TokenGen}(\mathsf{TS.sk}_i)$
  - *Output* $\rho_{\mathsf{sk}} = \{|\mathsf{tk}_i\rangle\}_{i\in[\kappa]}$

- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$*: On input a public key* $\mathsf{pk} = \{\mathsf{TS.pk}_i\}_{i\in[\kappa]}$ *and a message* $m \in \{0,1\}$*;*
  - *Sample a random string* $r \leftarrow \{0,1\}^\kappa$
  - *Compute* $\mathsf{ct}_{r,m} \leftarrow \mathsf{WE.Enc}(1^n, r, m)$*, where* r *is an instance of the language* L*, defined by the following* NP *relation* R_L*. In what follows, let* w *be parsed as* $w = w_1||\cdots||w_\kappa$*, for* w_i*'s of the appropriate length.*

$$R_L(r,w) = \begin{cases} 1 & \text{if } \mathsf{TS.Verify}(\mathsf{TS.pk}_i, r_i, w_i) = 1 \text{ for all } i \in [\kappa], \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

    That is, each w_i *should be a valid signature of* r_i*.*
  - *Output the ciphertext* $\mathsf{ct} = (\mathsf{ct}_{r,m}, r)$*.*
- $\mathsf{Dec}(\mathsf{ct}, \rho_{\mathsf{sk}}) \to m/\perp$*: On input a ciphertext* $\mathsf{ct} = (\mathsf{ct}_{r,m}, r)$ *and a quantum secret key* $\rho_{\mathsf{sk}} = \{|\mathsf{tk}_i\rangle\}_{i\in[\kappa]}$*.*
  - *For each* $i \in [\kappa]$*, sign message* r_i *by running* $(r_i, \mathsf{sig}_i) \leftarrow \mathsf{TS.Sign}(|\mathsf{tk}_i\rangle, r_i)$*. Let* $w = \mathsf{sig}_1||\cdots||\mathsf{sig}_\kappa$*.*
  - *Output* $m/\perp \leftarrow \mathsf{WE.Dec}(\mathsf{ct}_{r,m}, \mathsf{sig}_1||\cdots||\mathsf{sig}_\kappa)$*.*

  Note in the decryption algorithm Dec*, we run* TS.Sign *and* WE.Dec *coherently, so that (by the gentle measurement lemma) an honest user can rewind and use the quantum key polynomially many times.*

6.6 Security of Construction 2

The proofs of security are straightforward, and similar to the proofs given in [GZ20], except that here we have a new definition of γ-anti-piracy security, and we use a tokenized signature scheme instead of a one-shot signature scheme. The proof also resembles our proof of security for the construction from the strong monogamy-of-entanglement property.

We sketch the proofs here and omit some details.

Correctness and Efficiency. It is straightforward to see that all procedures are efficient and that correctness follows from the correctness of the WE and TS schemes.

CPA Security. CPA security relies on extractable security of the witness encryption scheme and on unforgeability of the tokenized signature scheme. Suppose that there exists a QPT adversary $\mathcal{A}$ that succeeds with non-negligible probability in its CPA security game, by the extractable security of witness encryption, there exists an extractor that extracts witness $w = \mathsf{sig}_i||\cdots||\mathsf{sig}_\kappa$, where each sig_i is the signature of a random bit r_i that can TS.Verify. This clearly violates the unforgeability of TS, since the adversary $\mathcal{A}$ in CPA security game is not given any tokens.

(Strong) γ-Anti-Piracy. Strong γ-anti-piracy security for any inverse-polynomial γ also follows from extractable security of the witness encryption scheme and unforgeability of tokenized signature scheme scheme.

Suppose that there exists a QPT adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ that succeeds with non-negligible probability in the γ-anti-piracy game. Then, with non-negligible probability over the randomness of the challenger, $\mathcal{A}_0$ outputs a state σ such that $\sigma[R_1]$ and $\sigma[R_2]$ simultaneously pass the γ-good decryptor test with non-negligible probability. Therefore, by applying the corresponding approximation threshold implementation (ATI), the resulting state $\sigma'[R_1]$ and $\sigma'[R_2]$ are negligibly close to being $(\gamma - \epsilon)$-good decryptors, for any inverse polynomial ϵ.

Since $(\gamma - \epsilon)$ is inverse-polynomial, then by the extractable security of the witness encryption scheme, there must exist an extractor E_1 on the R_1 side that extracts, with non-negligible probability, a witness $w_1 = \mathsf{sig}_{1,1}||\cdots||\mathsf{sig}_{1,\kappa}$ (where each $\mathsf{sig}_{1,i}$ is a signature for bit $r_{R_1,i}$). Similarly, by Claim 6.18, there also exists an extractor E_2 on the R_2 side that extracts witness $w_2 = \mathsf{sig}_{2,1}||\cdots||\mathsf{sig}_{2,\kappa}$ (where each $\mathsf{sig}_{2,i}$ is a signature for bit $r_{R_2,i}$) from the leftover state after extraction on R_1. Since r_{R_1}, r_{R_2} are independently sampled, with probability $(1 - 1/2^\kappa)$, there exists a position i^* where $r_{R_1,i^*} \neq r_{R_2,i^*}$. We can then construct an adversary that breaks the 1-unforgeability of tokenized signatures by getting one token $|\mathsf{tk}_{i^*}\rangle$ and successfully producing signatures on two different messages $r_{R_1,i^*} \neq r_{R_2,i^*}$.

7 Copy-Protection of Pseudorandom Functions

In this section, we formally define copy-protection of pseudorandom functions. Then, we describe a construction that essentially builds on the single-decryptor encryption scheme described in Section 6.3 (together with post-quantum sub-exponentially secure one-way functions and iO). The same construction can be based on the single-decryptor encryption scheme from Section 6.5, but we omit the details to avoid redundancy. In Section 7.2, we give definitions of certain families of PRFs which we use in our construction. We remark that all of the PRFs that we use can be constructed from post-quantum one-way functions.

7.1 Definitions

In what follows, the PRF $F : [K] \times [N] \to [M]$, implicitly depends on a security parameter λ. We denote by $\mathsf{Setup}(\cdot)$ the procedure which on input 1^λ, outputs a PRF key.

Definition 7.1 (Copy-Protection of PRF)**.** *A copy-protection scheme for a PRF $F : [K] \times [N] \to [M]$ consists of the following QPT algorithms:*

$\mathsf{QKeyGen}(K)$*: takes a key K and outputs a quantum key ρ_K;*

$\mathsf{Eval}(\rho_K, x)$*: takes a quantum key ρ_K and an input $x \in [N]$. It outputs a classical string $y \in [M]$.*

A copy-protection scheme should satisfy the following properties:

Definition 7.2 (Correctness)**.** *There exists a negligible function $\mathsf{negl}(\cdot)$, such that for all λ, all $K \leftarrow \mathsf{Setup}(1^\lambda)$, all inputs x,*

$$\Pr[\mathsf{Eval}(\rho_K, x) = F(K, x) : \rho_K \leftarrow \mathsf{QKeyGen}(K)] \geq 1 - \mathsf{negl}(\lambda)\,.$$

Note that the correctness property implies that the evaluation procedure has an "almost unique" output. This means that the PRF can be evaluated (and rewound) polynomially many times, without disturbing the quantum key ρ_K, except negligibly.

Definition 7.3 (Anti-Piracy Security)**.** *Let $\lambda \in \mathbb{N}^+$. Consider the following game between a challenger and an adversary $\mathcal{A}$:*

1. *The challenger samples $K \leftarrow \mathsf{Setup}(1^\lambda)$ and $\rho_K \leftarrow \mathsf{QKeyGen}(K)$. It gives ρ_K to $\mathcal{A}$;*
2. *$\mathcal{A}$ returns to the challenger a bipartite state σ on registers R_1 and R_2, as well as general quantum circuits U_1 and U_2.*
3. *The challenger samples uniformly random $u, w \leftarrow [N]$. Then runs U_1 on input $(\sigma[R_1], u)$, and runs U_2 on input $(\sigma[R_2], w)$. The outcome of the game is 1 if and only if the outputs are $F(K, u)$ and $F(K, w)$ respectively.*

Denote by $\mathsf{CopyProtectionGame}(1^\lambda, \mathcal{A})$ a random variable for the output of the game.

We say the scheme has anti-piracy security if for every polynomial-time quantum algorithm $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, for all $\lambda \in \mathbb{N}^+$,

$$\Pr\left[b = 1, b \leftarrow \mathsf{CopyProtectionGame}(1^\lambda, \mathcal{A})\right] = \mathsf{negl}(\lambda)\,.$$

We give a stronger anti-piracy definition, which is an indistinguishability definition and is specifically for copy-protecting PRFs. We will show that our construction also satisfies this definition.

Definition 7.4 (Indistinguishability Anti-Piracy Security for PRF)**.** *Let $\lambda \in \mathbb{N}^+$. Consider the following game between a challenger and an adversary $\mathcal{A}$:*

1. *The challenger runs $K \leftarrow \mathsf{Setup}(1^\lambda)$, and $\rho_K \leftarrow \mathsf{QKeyGen}(K)$. It gives ρ_K to $\mathcal{A}$;*
2. *$\mathcal{A}$ returns to the challenger a bipartite state σ on registers R_1 and R_2, as well as general quantum circuits U_1 and U_2.*
3. *The challenger samples two uniformly random inputs $u, w \leftarrow [N]$ and two uniformly random strings $y_1, y_2 \leftarrow [M]$ (these are of the same length as the PRF output).*
4. *The challenger flips two coins independently: $b_1, b_2 \leftarrow \{0, 1\}$. If $b_1 = 0$, it gives $(u, F(K, u), \sigma[R_1])$ as input to U_1; else it gives $(u, y_1, \sigma[R_1])$ as input to U_1. Let b'_1 be the output. Similarly, if $b_2 = 0$, it gives $(w, F(K, w), \sigma[R_2])$ as input to U_2; else it gives $(w, y_2, \sigma[R_2])$ as input to U_2. Let b'_2 be the output.*
5. *The outcome of the game is 1 if $b'_1 = b_1$ and $b'_2 = b_2$.*

Denote by $\mathsf{IndCopyProtectionGame}(1^\lambda, \mathcal{A})$ a random variable for the output of the game.

We say the scheme has indistinguishability anti-piracy security if for every polynomial-time quantum algorithm $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, for all $\lambda \in \mathbb{N}^+$,

$$\Pr\left[b = 1, b \leftarrow \mathsf{IndCopyProtectionGame}(1^\lambda, \mathcal{A})\right] = \frac{1}{2} + \mathsf{negl}(\lambda)\,.$$

Similarly to the relationship between CPA-style unclonable decryption (Definition 6.3) and anti-piracy with random challenge inputs (Definition 6.5), it is not clear whether Definition 7.4 implies Definition 7.3 (this subtlety arises due to the fact that there are two parties involved, having to simultaneously make the correct guess). Thus, we will give separate statements and security proofs in the next section.

7.2 Preliminaries: Puncturable PRFs and related notions

A *puncturable* PRF is a PRF augmented with a procedure that allows to "puncture" a PRF key K at a set of points S, in such a way that the adversary with the punctured key can evaluate the PRF at all points except the points in S. Moreover, even given the punctured key, an adversary cannot distinguish between a uniformly random value and the evaluation of the PRF at a point S with respect to the original unpunctured key. Formally:

Definition 7.5 ((Post-quantum) Puncturable PRF)**.** *A PRF family $F : \{0,1\}^{n(\lambda)} \to \{0,1\}^{m(\lambda)}$ with key generation procedure KeyGen_F is said to be puncturable if there exists an algorithm $\mathsf{Puncture}_F$, satisfying the following conditions:*

- ***Functionality preserved under puncturing:*** *Let $S \subseteq \{0,1\}^{n(\lambda)}$. For all $x \in \{0,1\}^{n(\lambda)}$ where $x \notin S$, we have that:*

$$\Pr[F(K,x) = F(K_S,x) : K \leftarrow \mathsf{KeyGen}(1^\lambda), K_S \leftarrow \mathsf{Puncture}_F(K,S)] = 1$$

- ***Pseudorandom at punctured points:*** *For every QPT adversary (A_1, A_2), there exists a negligible function negl such that the following holds. Consider an experiment where $K \leftarrow \mathsf{KeyGen}_F(1^\lambda)$, $(S,\sigma) \leftarrow A_1(1^\lambda)$, and $K_S \leftarrow \mathsf{Puncture}_F(K,S)$. Then, for all $x \in S$,*

$$\left|\Pr[A_2(\sigma, K_S, S, F(K,x)) = 1] - \Pr_{r \leftarrow \{0,1\}^{m(\lambda)}}[A_2(\sigma, K_S, S, r) = 1]\right| \leq \mathsf{negl}(\lambda)$$

Definition 7.6. *A statistically injective (puncturable) PRF family with (negligible) failure probability $\epsilon(\cdot)$ is a (puncturable) PRF family F such that with probability $1 - \epsilon(\lambda)$ over the random choice of key $K \leftarrow \mathsf{KeyGen}_F(1^\lambda)$, we have that $F(K,\cdot)$ is injective.*

We will also make use of extracting PRFs: these are PRFs that are strong extractors on their inputs in the following sense.

Definition 7.7 (Extracting PRF)**.** *An extracting (puncturable) PRF with error $\epsilon(\cdot)$ for min-entropy $k(\cdot)$ is a (puncturable) PRF F mapping $n(\lambda)$ bits to $m(\lambda)$ bits such that for all λ, if X is any distribution over $n(\lambda)$ bits with min-entropy greater than $k(\lambda)$, then the statistical distance between $(K, F(K,X))$ and $(K, r \leftarrow \{0,1\}^{m(\lambda)})$ is at most $\epsilon(\cdot)$, where $K \leftarrow \mathsf{KeyGen}(1^\lambda)$.*

Puncturable PRFs can be straightforwardly built by modifying the [GGM86] tree-based construction of PRFs, which only assumes one-way functions. [SW14] showed that puncturable statistically injective PRFs and extracting puncturable PRFs with the required input-output size can be built from one-way functions as well. These constructions can all be made post-quantum as shown in [Zha12]. Thus, the following theorems from [SW14] hold also against bounded quantum adversaries.

Theorem 7.8 ([SW14] Theorem 1, [GGM86])**.** *If post-quantum one-way functions exist, then for all efficiently computable functions $n(\lambda)$ and $m(\lambda)$, there exists a post-quantum puncturable PRF family that maps $n(\lambda)$ bits to $m(\lambda)$ bits.*

Theorem 7.9 ([SW14] Theorem 2)**.** *If post-quantum one-way functions exist, then for all efficiently computable functions $n(\lambda)$, $m(\lambda)$, and $e(\lambda)$ such that $m(\lambda) \geq 2n(\lambda) + e(\lambda)$, there exists a post-quantum puncturable statistically injective PRF family with failure probability $2^{-e(\lambda)}$ that maps $n(\lambda)$ bits to $m(\lambda)$ bits.*

Theorem 7.10 ([SW14] Theorem 3)**.** *If post-quantum one-way functions exist, then for all efficiently computable functions $n(\lambda)$, $m(\lambda)$, $k(\lambda)$, and $e(\lambda)$ such that $n(\lambda) \geq k(\lambda) \geq m(\lambda) + 2e(\lambda) + 2$, there exists a post-quantum extracting puncturable PRF family that maps $n(\lambda)$ bits to $m(\lambda)$ bits with error $2^{-e(\lambda)}$ for min-entropy $k(\lambda)$.*

7.3 Construction

In this section, we describe a construction of a copy-protection scheme for a class of PRFs. We will eventually reduce security of this construction to security of the single-decryptor encryption scheme of Section 6.3, and we will therefore inherit the same assumptions. A similar construction can be based on the single-decryptor encryption scheme of Section 6.5.

Let λ be the security parameter. Our construction copy-protects a PRF $F_1 : [K_\lambda] \times [N_\lambda] \to [M_\lambda]$ where $N = 2^{n(\lambda)}$ and $M = 2^{m(\lambda)}$, for some polynomials $n(\lambda)$ and $m(\lambda)$, satisfying $n(\lambda) \geq m(\lambda) + 2\lambda + 4$. For convenience, we will omit writing the dependence on λ, when it is clear from the context. Moreover, F_1 should be a puncturable extracting PRF with error $2^{-\lambda-1}$ for min-entropy $k(\lambda) = n(\lambda)$ (i.e., a uniform distribution over all possible inputs). By Theorem 7.10, such PRFs exist assuming post-quantum one-way functions.

In our construction, we will parse the input x to $F_1(K_1, \cdot)$ as three substrings $x_0||x_1||x_2$, where each x_i is of length ℓ_i for $i \in \{0, 1, 2\}$ and $n = \ell_0 + \ell_1 + \ell_2$. $\ell_2 - \ell_0$ should also be large enough (we will specify later how large). Our copy-protection construction for F_1 will make use of the following additional building blocks:

1. A puncturable statistically injective PRF F_2 with failure probability $2^{-\lambda}$ that accepts inputs of length ℓ_2 and outputs strings of length ℓ_1. By Theorem 7.9, such a PRF exists assuming one-way functions exist, and as long as $\ell_1 \geq 2\ell_2 + \lambda$.

2. A puncturable PRF F_3 that accepts inputs of length ℓ_1 and outputs strings of length ℓ_2. By Lemma 7.14 in [SW14], assuming one-way functions exist, F_3 is a puncturable PRF.

Note that PRF F_1 is the PRF that we will copy-protect. The PRFs F_2 and F_3 are just building blocks in the construction.

Next, we describe a copy-protection scheme for the PRF F_1, using the above building blocks. The description is contained in Figures 2 and 3.

The program P, described in Figure 3, takes as input x and ℓ_0 vectors $v_1, \cdots, v_{\ell_0}$, and has two modes. If x is not in the sparse hidden trigger set (not passing the 'if' check in the first line), the program is in the *normal mode*: it outputs a PRF evaluation of x if and only if every v_i is in the appropriate coset. Otherwise, the program is in the *hidden trigger mode*. It will compute a circuit Q' from the input x and output $Q'(v_1, \cdots, v_{\ell_0})$. On almost all inputs except a sparse set of hidden triggers, the program runs in its normal mode. For $i \in [l_0]$, define the programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s_i')$ (where the inputs to iO should be appropriately padded).

We prove the following theorem:

Theorem 7.11. *Assuming the existence of post-quantum iO, one-way functions, compute-and-compare obfuscation for the class of unpredictable distributions (as in Definition 3.6), and the strong monogamy-of-entanglement property (Conjecture 4.16), our construction satisfies anti-piracy security (as in Definition 7.3).*

$\mathsf{QKeyGen}(K_1)$: Sample uniformly random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. Sample PRF keys K_2, K_3 for F_2, F_3. Let P be the program described in Figure 3. Output the quantum key $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$.

$\mathsf{Eval}(\rho_K, x)$: Let $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$. Parse x as $x = x_0||x_1||x_2$ where x_0 is of length ℓ_0. For all $i \in [\ell_0]$, if $x_{0,i}$ is 1, apply $H^{\otimes n}$ to $|A_{i,s_i,s'_i}\rangle$. Otherwise, leave the state unchanged.

Let σ be the resulting state (which can be interpreted as a superposition over tuples of l_0 vectors). Run $\mathsf{iO}(P)$ coherently on input x and σ, and measure the final output register to obtain y.

Figure 2: Quantum copy-protection scheme for PRFs.

Hardcoded: Keys K_1, K_2, K_3, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:

   Hidden Trigger Mode: Treat Q' as a (classical) circuit and output $Q'(v_1, \cdots, v_{\ell_0})$.

2. Otherwise, check if the following holds: for all $i \in [\ell_0]$, $R_i^{x_{0,i}}(v_i) = 1$ (where $x_{0,i}$ is the i-th bit of x_0).

   Normal Mode: If so, output $F_1(K_1, x)$. Otherwise, output $\perp$.

Figure 3: Program P

Similarly, assuming the existence of post-quantum sub-exponentially secure iO *and one-way functions, the quantum hardness of LWE and assuming the strong monogamy-of-entanglement property (Conjecture 4.16), our construction satisfies anti-piracy security.*

We show correctness of our construction in Section 7.4, and anti-piracy security in Section 7.5.

The following theorem states that our construction also satisfies Definition 7.4.

Theorem 7.12. *Assuming the existence of post-quantum* iO*, one-way functions, compute-and-compare obfuscation for the class of unpredictable distributions (as in Definition 3.6), and the strong monogamy-of-entanglement property (Conjecture 4.16), our construction satisfies indistinguishability-based anti-piracy security (as in Definition 7.4).*

Similarly, assuming the existence of post-quantum sub-exponentially secure iO *and one-way functions, the quantum hardness of LWE and assuming the strong monogamy-of-entanglement property (Conjecture 4.16), our construction satisfies indistinguishability-based anti-piracy security.*

We include the proof of the latter theorem in Appendix F.

7.4 Proof of Correctness

First, it is easy to see that all procedures are efficient. We then show that our construction satisfies correctness.

Lemma 7.13. *The above construction has correctness.*

Proof. First, we observe that for an input x, keys K_2, K_2, if the step 1 check in the program P is not met, then the output of $\mathsf{Eval}(\rho_K, x)$ will be the same as $F_1(K_1, \cdot)$ with probability 1.

Therefore, let us assume there exists a fixed input $x^* = x_0^* || x_1^* || x_2^*$ such that for an inverse polynomial fraction of possible keys K_2, K_3, the step 1 check is passed. Define $\hat{x}_2^*$ be the first ℓ_0 bits of x_2^* and $\hat{F}_3(K_3, \cdot)$ be the function that outputs the first ℓ_0 bits of $F_3(K_3, \cdot)$. $\hat{F}_3$ is a PRF because it is a truncation of another PRF F_3. To pass the step 1 check, $(x_0^*, x_1^*, \hat{x}_2^*)$ should at least satisfy:

$$\hat{F}_3(K_3, x_1^*) \oplus x_0^* = \hat{x}_2^*.$$

Thus, for an inverse polynomial fraction of K_3, the above equation holds. This gives a non-uniform algorithm for breaking the security of $\hat{F}_3$ and violates the security of F_3 as a consequence: given oracle access to $\hat{F}_3(K_3, \cdot)$ for a random K_3, or a truly random function $f(\cdot)$, the algorithm simply queries on x_1^* and checks if the output is $x_0^* \oplus \hat{x}_2^*$; if yes, it outputs 1 (indicating the function is $\hat{F}_3(K_3, \cdot)$); otherwise, it outputs 0 (indicating the function is a truly random funtion). Since the above equation holds for an inverse polynomial fraction of K_3, our non-uniform algorithm succeeds with an inverse polynomial probability.

Since non-uniform security of PRFs can be based on non-uniform security of OWFs, the correctness of our construction relies on the existence of non-uniform secure post-quantum OWFs. □

7.5 Proof of Anti-Piracy Security

In this subsection, we prove the anti-piracy security. Before proving anti-piracy, we give the following helper lemma from [SW14].

Lemma 7.14 (Lemma 1 in [SW14])**.** *Except with negligible probability over the choice of the key* K_2*, the following two statements hold:*

1. *For any fixed x_1, there exists at most one pair (x_0, x_2) that will cause the step 1 check in Program P to pass.*

2. *There are at most 2^{ℓ_2} values of x that can cause the step 1 check to pass.*

The proof will exploit the sparse hidden triggers in the program P. Intuitively, we want to show that sampling a unifomly random input is indistinguishable from sampling an element from the sparse hidden trigger set. Then, we will reduce an adversary that successfully evaluates on hidden triggers to an adversary that breaks the single decryptor encryption scheme of Section 6.

Definition 7.15 (Hidden Trigger Inputs)**.** *An input x is a hidden trigger input of the program P (defined in Figure 3) if it makes the step 1 check in the program be satisfied.*

We will prove a lemma says that no efficient algorithm, given the quantum key, can distinguish between the following two cases: (i) sample two uniformly random inputs, and (ii) sample two inputs in the hidden trigger set.

Before describing the lemma, we describe an efficient procedure which takes as input an input/output pair for F_1, PRF keys K_2, K_3, descriptions of cosets, and produces a hidden trigger input.

Definition 7.16. *The procedure* $\mathsf{GenTrigger}$ *takes as input x_0 (of length ℓ_0), y (of length m, where m is the length of the output of F_1), two PRF keys K_2, K_3 and hidden cosets $\{A_i, s_i, s'_i\}_{i\in[\ell_0]}$:*

1. *Let Q be the program (padded to length $\ell_2 - \ell_0$) that takes as input $v_1, \cdots, v_{\ell_0}$ and outputs y if and only if for every input v_i, if $x_{0,i} = 0$, then v_i is in $A_i + s_i$ and otherwise it is in $A_i^{\perp} + s'_i$.*

2. *$x'_1 \leftarrow F_2(K_2, x_0||Q)$;*

3. *$x'_2 \leftarrow F_3(K_3, x'_1) \oplus (x_0||Q)$.*

4. *Output $x' = x_0||x'_1||x'_2$.*

Note that for any x_0, y, $\mathsf{GenTrigger}$ will produce an input x' such that it starts with x_0 and the evaluation of P on input x' and valid vectors $v_1, \cdots, v_{\ell_0}$ is y.

The following lemma says that any efficient algorithm cannot distinguish if it gets two inputs sampled uniformly at random, or two hidden trigger inputs (sampled according to Definition 7.16):

Lemma 7.17. *Assuming post-quantum* iO *and one-way functions, any efficient QPT algorithm $\mathcal{A}$ cannot win the following game with non-negligible advantage:*

- *A challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and prepares a quantum key $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$ (where recall that P has keys K_1, K_2, K_3 hardcoded).*

- *The challenger then samples a random input $u \leftarrow [N]$. Let $y_u = F_1(K_1, u)$. Parse the input as $u = u_0||u_1||u_2$.*

  Let $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.

- *Similarly, it samples a random input $w \leftarrow [N]$. Let $y_w = F_1(K_1, w)$. Parse the input as $w = w_0||w_1||w_2$.*

  Let $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.

- *The challenger flips a coin b, and sends (ρ_K, u, w) or (ρ_K, u', w') to $\mathcal{A}$, depending on the outcome. $\mathcal{A}$ wins if it guesses b.*

One might wonder whether it is sufficient to just show a version of the above lemma which says that any efficient algorithm cannot distinguish if it gets *one* uniformly random input or *one* random hidden trigger input, and use a hybrid argument to show indistinguishability in the case of two samples. However, this is not the case, as one cannot efficiently sample a random hidden trigger input when given only the public information in the security game (in particular $\mathsf{GenTrigger}$ requires knowing K_2, K_3), and so the typical reduction would not go through.

Next, we show that if Lemma 7.17 holds, then our construction satisfies anti-piracy security Theorem 7.11. After this, to finish the proof, we will only need to prove Lemma 7.17. The core of the latter proof is the "hidden trigger" technique used in [SW14], which we will prove in Appendix E.

Proof for Theorem 7.11. We mark the changes between the current hybrid and the previous in red.

Hybrid 0. Hybrid 0 is the original anti-piracy security game.

1. The challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P)) \leftarrow \mathsf{QKeyGen}(K_1)$. Note that here P hardcodes K_1, K_2, K_3.
2. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
3. The challenger also prepares two inputs u, w as follows:
   - It samples u uniformly at random. Let $y_u = F_1(K_1, u)$.
   - It samples w uniformly at random. Let $y_w = F_1(K_1, w)$.
4. The outcome of the game is 1 if and only if both quantum programs successfully produce y_u and y_w respectively.

Hybrid 1 The difference between Hybrids 0 and 1 corresponds exactly to the two cases that the adversary needs to distinguish between in the game of Lemma 7.17.

1. The challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P)) \leftarrow \mathsf{QKeyGen}(K_1)$. Note that here P hardcodes K_1, K_2, K_3.
2. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
3. The challenger also prepares two inputs u', w' as follows:
   - It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u = F_1(K_1, u)$. Let $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
   - It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w = F_1(K_1, w)$. Let $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.

4. The outcome of the game is 1 if and only if both quantum programs successfully produce y_u and y_w respectively.

Assume there exists an algorithm that distinguishes Hybrid 0 and 1 with *non-negligible* probability $\epsilon(\lambda)$, then these exists an algorithm that breaks the game in Lemma 7.17 with probability $\epsilon(\lambda) - \mathsf{negl}(\lambda)$.

The reduction algorithm receives ρ_k and u, w or u', w' from the challenger in Lemma 7.17; it computes y_u, y_w using $\mathsf{iO}(P)$ on the received inputs respectively and gives them to the quantum decryptor states $\sigma[R_1], \sigma[R_2]$. If they both decrypt correctly, then the reduction outputs 0 (i.e. it guess that sampling was uniform), otherwise it outputs 1 (i.e. it guesses that hidden trigger inputs were sampled).

Hybrid 2. In this hybrid, if $u_0 \neq w_0$ (which happens with overwhelming probability), $F_1(K_1, u)$ and $F_1(K_1, w)$ are replaced with truly random strings. Since both inputs have enough min-entropy $\ell_1+\ell_2 \geq m+2\lambda+4$ (as $u_1||u_2$ and $w_1||w_2$ are completely uniform and not given to the adversary) and F_1 is an extracting puncturable PRF, both outcomes y_u, y_w are statistically close to independently random outcomes. Thus, Hybrid 1 and Hybrid 2 are statistically close.

1. The challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P)) \leftarrow \mathsf{QKeyGen}(K_1)$. Note that here P hardcodes K_1, K_2, K_3.
2. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
3. The challenger also prepares two inputs u', w' as follows:
   - It samples u_0 uniformly at random. It then samples a uniformly random y_u. Let $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
   - It samples w_0 uniformly at random. It then samples a uniformly random y_w. Let $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
4. The outcome of the game is 1 if and only if both quantum programs successfully produce y_u and y_w respectively.

Hybrid 3. The game in this hybrid has exactly the same distribution as that of Hybrid 2 (in the sense that all sampled values are distributed identically). We only change the order in which some values are sampled, and recognize that certain procedures become identical to encryptions in our single-decryptor encryption scheme from Section 6. Thus, $\mathcal{A}$ wins the game with the same probability as in Hybrid 2.

1. The challenger first samples $\{A_i, s_i, s'_i\}_{i\in[\ell_0]}$ and prepares the quantum states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}$. It treat the the quantum states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}$ as the quantum decryption key ρ_{sk} for our single-decryptor encryption scheme and the secret key sk is $\{A_i, s_i, s'_i\}_{i\in[\ell_0]}$. Similarly, let $\mathsf{pk} = \{R^0_i, R^1_i\}_{i\in[\ell_0]}$ where $R^0_i = \mathsf{iO}(A_i + s_i)$ and $R^1_i = \mathsf{iO}(A^\perp_i + s'_i)$.
2. It samples y_u, y_w uniformly at random. Let $(u_0, Q_0) \leftarrow \mathsf{Enc}(\mathsf{pk}, y_u)$ and $(w_0, Q_1) \leftarrow \mathsf{Enc}(\mathsf{pk}, y_w)$ where $\mathsf{Enc}(\mathsf{pk}, \cdot)$ is the encryption algorithm of the single-decryptor encryption scheme of Construction 1.

3. The challenger sets $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$.

4. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.

5. The challenger also prepares two inputs u', w' as follows (as $\mathsf{GenTrigger}$ does):

   - Let $u'_1 \leftarrow F_2(K_2, u_0||Q_0)$ and $u'_2 \leftarrow F_3(K_3, u'_1) \oplus (u_0||Q_0)$. Let $u' = u_0||u'_1||u'_2$.
   - Let $w'_1 \leftarrow F_2(K_2, w_0||Q_1)$ and $w'_2 \leftarrow F_3(K_3, w'_1) \oplus (w_0||Q_1)$. Let $w' = w_0||w'_1||w'_2$.

6. The outcome of the game is 1 if and only if both quantum programs successfully produce y_u and y_w respectively.

Note that the only differences of Hybrids 2 and 3 are the orders of executions. Namely, in Hybrid 3, $\{A_i, s_i, s'_i\}$ are sampled much earlier than when ρ_k is prepared. Similarly, the obfuscation programs sampled in $\mathsf{GenTrigger}$ are now sampled much earlier than sampling u' and w'. We write Hybrid 3 in a way that is similar to the weak anti-piracy security game of the single-decryptor encryption scheme of Construction 1.

Given an algorithm $\mathcal{A}$ that wins the game in Hybrid 3 with non-negligible probability $\gamma(\lambda)$, we can build another algorithm $\mathcal{B}$ that breaks the (regular) γ-anti-piracy security with random challenge plaintexts (see Definition 6.6) of the underlying single-decryptor encryption scheme.

- $\mathcal{B}$ plays as the challenger in the game of Hybrid 3.
- $\mathcal{B}$ receives $\rho_{\mathsf{sk}} = \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}$ and $\mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A_i^{\perp} + s'_i)\}_{i\in[\ell_0]}$ in the anti-piracy game of single-decryptor encryption.
- $\mathcal{B}$ prepares K_1, K_2, K_3 and the program P. Let $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$.
- $\mathcal{B}$ gives ρ_K to $\mathcal{A}$, and $\mathcal{A}$ prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
- $\mathcal{B}$ outputs the decryptors $(\sigma[R_1], \mathsf{P}_1)$ and $(\sigma[R_2], \mathsf{P}_2)$, where P_1 and P_2 are defined as follows: on input $(\rho_1, \mathsf{ct}_1 = (u_0||Q_1))$ and $(\rho_2, \mathsf{ct}_2 = (w_0||Q_2))$ respectively (where ct_1 and ct_2 represent encryptions of random y_u and y_w), P_1 and P_2 behave respectively as follows:
  - P_1: Let $u'_1 \leftarrow F_2(K_2, u_0||Q_0)$ and $u'_2 \leftarrow F_3(K_3, u'_1) \oplus (u_0||Q_0)$. Let $u' = u_0||u'_1||u'_2$. Run (ρ_1, U_1) on u'.
  - P_2: Let $w'_1 \leftarrow F_2(K_2, w_0||Q_1)$ and $w'_2 \leftarrow F_3(K_3, w'_1) \oplus (w_0||Q_1)$. Let $w' = w_0||w'_1||w'_2$. Run (ρ_2, U_2) on w' respectively.

We know that whenever $\mathcal{A}$ succeeds in the game of Hyb 3, it outputs y_u, y_w correctly. Thus, the programs prepared by $\mathcal{B}$ successfully decrypts encryptions of uniformly random plaintexts. Thus, $\mathcal{B}$ breaks γ-anti-piracy security with random challenge plaintexts. □

References

[Aar05] Scott Aaronson. "Limitations of Quantum Advice and One-Way Communication". In: *Theory of Computing* 1.1 (2005), pp. 1–28. DOI: 10.4086/toc.2005.v001a001. URL: https://doi.org/10.4086/toc.2005.v001a001.

[Aar09] Scott Aaronson. "Quantum copy-protection and quantum money". In: *2009 24th Annual IEEE Conference on Computational Complexity*. IEEE. 2009, pp. 229–242.

[AC02] Mark Adcock and Richard Cleve. "A quantum Goldreich-Levin theorem with cryptographic applications". In: *Annual Symposium on Theoretical Aspects of Computer Science*. Springer. 2002, pp. 323–334.

[AC12] Scott Aaronson and Paul Christiano. "Quantum money from hidden subspaces". In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. ACM. 2012, pp. 41–60.

[ALL+20] Scott Aaronson, Jiahui Liu, Qipeng Liu, Mark Zhandry, and Ruizhe Zhang. *New approaches for quantum copy-protection*. 2020.

[AP21] Prabhanjan Ananth and Rolando L. La Placa. *Secure Software Leasing*. 2021.

[BB84] Charles H Bennett and Gilles Brassard. *Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing*. 1984.

[BDGM20] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. *Factoring and Pairings are not Necessary for iO: Circular-Secure LWE Suffices*. Cryptology ePrint Archive, Report 2020/1024. https://eprint.iacr.org/2020/1024. 2020.

[BGI+01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. "On the (im) possibility of obfuscating programs". In: *Annual International Cryptology Conference*. Springer. 2001, pp. 1–18.

[BGMZ18] James Bartusek, Jiaxin Guan, Fermi Ma, and Mark Zhandry. "Preventing Zeroizing Attacks on GGH15". In: *Proceedings of TCC 2018*. 2018.

[BJL+21] Anne Broadbent, Stacey Jeffery, Sébastien Lord, Supartha Podder, and Aarthi Sundaram. *Secure Software Leasing Without Assumptions*. 2021. arXiv: 2101.12739 [quant-ph].

[BL19] Anne Broadbent and Sébastien Lord. "Uncloneable Quantum Encryption via Random Oracles". In: *IACR Cryptology ePrint Archive* 2019 (2019), p. 257.

[BS16] Shalev Ben-David and Or Sattath. "Quantum tokens for digital signatures". In: *arXiv preprint arXiv:1609.09047* (2016).

[BW13] Dan Boneh and Brent Waters. "Constrained pseudorandom functions and their applications". In: *International conference on the theory and application of cryptology and information security*. Springer. 2013, pp. 280–300.

[CMP20] Andrea Coladangelo, Christian Majenz, and Alexander Poremba. *Quantum copy-protection of compute-and-compare programs in the quantum random oracle model*. 2020. arXiv: 2009.13865 [quant-ph].

[FGH+12] Edward Farhi, David Gosset, Avinatan Hassidim, Andrew Lutomirski, and Peter Shor. "Quantum money from knots". In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. 2012, pp. 276–289.

[GGH+16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. "Candidate indistinguishability obfuscation and functional encryption for all circuits". In: *SIAM Journal on Computing* 45.3 (2016), pp. 882–929.

[GGHW17] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. "On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input". In: *Algorithmica* 79.4 (2017), pp. 1353–1373.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. "How to Construct Random Functions". In: *J. ACM* 33.4 (Aug. 1986), pp. 792–807. ISSN: 0004-5411. DOI: 10.1145/6490.6503. URL: https://doi.org/10.1145/6490.6503.

[GKW17] Rishab Goyal, Venkata Koppula, and Brent Waters. "Lockable obfuscation". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 612–621.

[GL89] Oded Goldreich and Leonid A Levin. "A hard-core predicate for all one-way functions". In: *Proceedings of the twenty-first annual ACM symposium on Theory of computing*. 1989, pp. 25–32.

[Got02] Daniel Gottesman. "Uncloneable encryption". In: *arXiv preprint quant-ph/0210062* (2002).

[GZ20] Marios Georgiou and Mark Zhandry. *Unclonable Decryption Keys*. Cryptology ePrint Archive, Report 2020/877. https://eprint.iacr.org/2020/877. 2020.

[JLS20] Aayush Jain, Huijia Lin, and Amit Sahai. *Indistinguishability Obfuscation from Well-Founded Assumptions*. Cryptology ePrint Archive, Report 2020/1003. https://eprint.iacr.org/2020/1003. 2020.

[Kan18] Daniel M Kane. "Quantum money from modular forms". In: *arXiv preprint arXiv:1809.05925* (2018).

[KNY20] Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. *Secure Software Leasing from Standard Assumptions*. 2020. arXiv: 2010.11186 [quant-ph].

[Lut10] Andrew Lutomirski. "An online attack against Wiesner's quantum money". In: *arXiv preprint arXiv:1010.0256* (2010).

[MW05] Chris Marriott and John Watrous. "Quantum arthur–merlin games". In: *computational complexity* 14.2 (2005), pp. 122–152.

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

[PW11] Chris Peikert and Brent Waters. "Lossy trapdoor functions and their applications". In: *SIAM Journal on Computing* 40.6 (2011), pp. 1803–1844.

[SW14] Amit Sahai and Brent Waters. "How to use indistinguishability obfuscation: deniable encryption, and more". In: *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 2014, pp. 475–484.

[TFKW13] Marco Tomamichel, Serge Fehr, Jędrzej Kaniewski, and Stephanie Wehner. "A monogamy-of-entanglement game with applications to device-independent quantum cryptography". In: *New Journal of Physics* 15.10 (2013), p. 103002.

[VZ21] Thomas Vidick and Tina Zhang. "Classical proofs of quantum knowledge". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2021, pp. 630–660.

[Wie83] Stephen Wiesner. "Conjugate coding". In: *ACM Sigact News* 15.1 (1983), pp. 78–88.

[WW20] Hoeteck Wee and Daniel Wichs. *Candidate Obfuscation via Oblivious LWE Sampling*. Cryptology ePrint Archive, Report 2020/1042. `https://eprint.iacr.org/2020/1042`. 2020.

[WZ17] Daniel Wichs and Giorgos Zirdelis. "Obfuscating compute-and-compare programs under LWE". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 600–611.

[Zha12] Mark Zhandry. "How to Construct Quantum Random Functions". In: *Proceedings of the 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*. FOCS '12. USA: IEEE Computer Society, 2012, pp. 679–687. ISBN: 9780769548746. DOI: 10.1109/FOCS.2012.37. URL: https://doi.org/10.1109/FOCS.2012.37.

[Zha19a] Mark Zhandry. "Quantum lightning never strikes the same state twice". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2019, pp. 408–438.

[Zha19b] Mark Zhandry. "The magic of ELFs". In: *Journal of Cryptology* 32.3 (2019), pp. 825–866.

[Zha20] Mark Zhandry. *Schrödinger's Pirate: How To Trace a Quantum Decoder*. Cryptology ePrint Archive, Report 2020/1191. `https://eprint.iacr.org/2020/1191`. 2020.

A Additional Preliminaries

A.1 Quantum Computation and Information

A quantum system Q is defined over a finite set B of classical states. In this work we will consider $B = \{0,1\}^n$. A **pure state** over Q is a unit vector in $\mathbb{C}^{|B|}$, which assigns a complex number to each element in B. In other words, let $|\phi\rangle$ be a pure state in Q, we can write $|\phi\rangle$ as:

$$|\phi\rangle = \sum_{x \in B} \alpha_x |x\rangle$$

where $\sum_{x\in B} |\alpha_x|^2 = 1$ and $\{|x\rangle\}_{x\in B}$ is called the "**computational basis**" of $\mathbb{C}^{|B|}$. The computational basis forms an orthonormal basis of $\mathbb{C}^{|B|}$.

Given two quantum systems R_1 over B_1 and R_2 over B_2, we can define a **product** quantum system $R_1 \otimes R_2$ over the set $B_1 \times B_2$. Given $|\phi_1\rangle \in R_1$ and $|\phi_2\rangle \in R_2$, we can define the product state $|\phi_1\rangle \otimes |\phi_2\rangle \in R_1 \otimes R_2$.

We say $|\phi\rangle \in R_1 \otimes R_2$ is **entangled** if there does not exist $|\phi_1\rangle \in R_1$ and $|\phi_2\rangle \in R_2$ such that $|\phi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle$. For example, consider $B_1 = B_2 = \{0,1\}$ and $R_1 = R_2 = \mathbb{C}^2$, $|\phi\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$ is entangled. Otherwise, we say $|\phi\rangle$ is un-entangled.

A mixed state is a collection of pure states $|\phi_i\rangle$ for $i \in [n]$, each with associated probability p_i, with the condition $p_i \in [0,1]$ and $\sum_{i=1}^n p_i = 1$. A mixed state can also be represented by the density matrix: $\rho := \sum_{i=1}^n p_i |\phi_i\rangle \langle\phi_i|$.

Partial Trace. For two subsystems R_1 and R_2 making up the composite system described by the density matrix ρ. The partial trace over the R_2 subsystem, denoted Tr_{R_2}, is defined as $\mathrm{Tr}_{R_2}[\rho] := \sum_j (I_{R_1} \otimes \langle j|_{R_2})\rho(I_{R_1} \otimes |j\rangle_{R_2})$. where $\{|j\rangle\}$ is any orthonormal basis for subsystem R_2.

For a quantum state σ over two registers R_1, R_2, we denote the state in R_1 as $\sigma[R_1]$, where $\sigma[R_1] = \mathrm{Tr}_{R_2}[\sigma]$ is a partial trace of σ. Similarly, we denote $\sigma[R_2] = \mathrm{Tr}_{R_1}[\sigma]$.

Purification of mixed states. For a mixed state ρ over system Q, there exists another space Q' and a pure state $|\psi\rangle$ over $Q \otimes Q'$ such that ρ is a partial trace of $|\psi\rangle\langle\psi|$ with respect to Q'.

A pure state $|\phi\rangle$ can be manipulated by a unitary transformation U. The resulting state $|\phi'\rangle = U|\phi\rangle$.

We can extract information from a state $|\phi\rangle$ by performing a **measurement**. A measurement specifies an orthonormal basis, typically the computational basis, and the probability of getting result x is $|\langle x|\phi\rangle|^2$. After the measurement, $|\phi\rangle$ "collapses" to the state $|x\rangle$ if the result is x.

For example, given the pure state $|\phi\rangle = \frac{3}{5}|0\rangle + \frac{4}{5}|1\rangle$ measured under $\{|0\rangle, |1\rangle\}$, with probability $9/25$ the result is 0 and $|\phi\rangle$ collapses to $|0\rangle$; with probability $16/25$ the result is 1 and $|\phi\rangle$ collapses to $|1\rangle$.

We finally assume a quantum computer can implement any unitary transformation (by using these basic gates, Hadamard, phase, CNOT and $\frac{\pi}{8}$ gates), especially the following two unitary transformations:

- **Classical Computation:** Given a function $f : X \to Y$, one can implement a unitary U_f over $\mathbb{C}^{|X|\cdot|Y|} \to \mathbb{C}^{|X|\cdot|Y|}$ such that for any $|\phi\rangle = \sum_{x\in X, y\in Y} \alpha_{x,y}|x, y\rangle$,

$$U_f|\phi\rangle = \sum_{x\in X, y\in Y} \alpha_{x,y}|x, y \oplus f(x)\rangle$$

  Here, $\oplus$ is a commutative group operation defined over Y.

- **Quantum Fourier Transform:** Let $N = 2^n$. Given a quantum state $|\phi\rangle = \sum_{i=0}^{2^n-1} x_i|i\rangle$, by applying only $O(n^2)$ basic gates, one can compute $|\psi\rangle = \sum_{i=0}^{2^n-1} y_i|i\rangle$ where the sequence $\{y_i\}_{i=0}^{2^n-1}$ is the sequence achieved by applying the classical Fourier transform QFT_N to the sequence $\{x_i\}_{i=0}^{2^n-1}$:

$$y_k = \frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} x_i \omega_n^{ik}$$

  where $\omega_n = e^{2\pi i/N}$, i is the imaginary unit.

  One property of QFT is that by preparing $|0^n\rangle$ and applying QFT_2 to each qubit, $(\mathsf{QFT}_2|0\rangle)^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle$ which is a uniform superposition over all possible $x \in \{0,1\}^n$.

For convenience, we sometimes omit writing the normalization of a pure state.

B Compute-and-Compare Obfuscation for (Sub-Exponentially) Unpredictable Distributions

In this section, we prove compute-and-compare obfuscation for sub-exponentially unpredictable distributions exists assuming the existence of post-quantum iO and the quantum hardness of LWE.

We show a similar statement about compute-and-compare obfuscation for any unpredictable distributions assuming post-quantum iO and post-quantum extremely lossy functions. We focus on the first result and it extends to the second case with little effort.

Our proof follows the steps below:

1. Assuming the quantum hardness of LWE, there exist lossy functions with any sub-linear residual leakage [PW11].
2. Assuming lossy functions with any sub-linear residual leakage, there exist PRGs with sub-exponentially unpredictable seeds (quantum auxiliary input). The proof constitutes that of [Zha19b], with the building block ELFs (extremely loss functions) being replaced with plain lossy functions and the last step of invoking Goldriech-Levin [GL89] being replaced with a quantum version of Goldreich-Levin [AC02]. For the quantum version of Goldreich-Levin, we prove a variant which holds against quantum auxiliary input.
3. Finally, assuming PRGs with sub-exponentially unpredictable seeds and post-quantum iO, there exists compute-and-compare obfuscation for sub-exponentially unpredictable distributions [WZ17].

As proved in [WZ17], in Step 3, we can build such compute-and-compare obfuscation solely based on the quantum hardness of LWE. However, as all the constructions in this work require iO as a building block, we focus on the simpler construction which is based on iO. Thus, we have the following theorem:

Theorem B.1. *Assuming the existence of post-quantum* iO *and the quantum hardness of LWE, there exist obfuscators as in Definition 3.6. for sub-exponentially unpredictable distributions.*

In the rest of the section, we will introduce all building blocks and prove Step 2. Step 1 and 3 follow directly from previous work.

Similarly, we can prove the following theorem:

Theorem B.2. *Assuming the existence of post-quantum* iO *and post-quantum extremely lossy functions, there exist obfuscators as in Definition 3.6. for any unpredictable distributions.*

Theorem B.2 directly follows all three steps above without even replacing the building block ELFs with plain lossy functions. Thus, we omit the proof here. However, currently we do not know any post-quantum construction for extremly lossy functions.

B.1 Preliminaries

We first introduce lossy functions. For the purpose of this work, we ignore the need of trapdoors in the definition. The definition is taken verbatim from [PW11].

Define the following quantities: the security parameter is λ, $n(\lambda) = \mathsf{poly}(\lambda)$ represents the input length of the function, $m(\lambda) = \mathsf{poly}(\lambda)$ represents the output length and $k(\lambda) \leq n(\lambda)$ represents the lossiness of the collection. For convenience, we also define the residual leakage $r(\lambda) := n(\lambda) - k(\lambda)$. For all these quantities, we often omit the dependence on λ.

Definition B.3 (Lossy Functions [PW11])**.** *A collection of (n, k)-lossy functions is given by a tuple of (possibly probabilistic) polynomial-time algorithms $(S_{\mathsf{lf}}, F_{\mathsf{lf}})$ having the properties below. For notational convenience, define the algorithms $S_{\mathsf{inj}}(\cdot) := S_{\mathsf{lf}}(\cdot, 1)$ and $S_{\mathsf{lossy}}(\cdot) := S_{\mathsf{lf}}(\cdot, 0)$.*

1. Easy to sample an injective function: $S_{\mathsf{inj}}(1^\lambda)$ *outputs* s *where* s *is a function index, with overwhelming probability,* $F_{\mathsf{lf}}(s,\cdot)$ *computes a (deterministic) injective function* $f_s(\cdot)$ *over the domain* $\{0,1\}^{n(\lambda)}$.

   For notational convenience, we assume $S_{\mathsf{inj}}(1^\lambda)$ *samples a function description* $f_s(\cdot)$.

2. Easy to sample a lossy function: $S_{\mathsf{lossy}}(1^\lambda)$ *outputs* s *where* s *is a function index,* $F_{\mathsf{lf}}(s,\cdot)$ *computes a (deterministic) function* $f_s(\cdot)$ *over the domain* $\{0,1\}^{n(\lambda)}$ *whose image has size at most* $2^r = 2^{n-k}$*, with overwhelming probability.*

   For notational convenience, we also assume $S_{\mathsf{lossy}}(1^\lambda)$ *samples a function description* $f_s(\cdot)$.

3. Hard to distinguish injective from lossy: *the outputs (function descriptions) of* $S_{\mathsf{inj}}(1^\lambda)$ *and* $S_{\mathsf{lossy}}(1^\lambda)$ *are computationally indistinguishable.*

Theorem B.4 (Theorem 6.4, [PW11])**.** *Assuming* $\mathsf{LWE}_{q,\chi}$ *is hard for some* q, χ*, there exists a collection of* (n,k) *lossy functions where the residual leakage* r *is* $r = n^c$ *for any constant* $c > 0$.

Remark B.5. *This can be done by carefully choosing parameters* $c_1 = n^{1-c}$*,* c_2 *as some constant,* $c_3 = 1/c$ *in Theorem 6.4 of [PW11].*

Remark B.6. *For the definition of extremely lossy functions,*

- *In bullet (2):* $S_{\mathsf{lossy}}(1^\lambda)$ *takes another parameter* $r \in [2^n]$ *and* f_s *sampled from* $S_{\mathsf{lossy}}(1^\lambda, r)$ *has image size* r*, with overwhelming probability;*
- *In bullet (3): For any polynomial* p *and inverse polynomial function* δ *(in* n*), there is a polynomial* q *such that: for any adversary* $\mathcal{A}$ *running in time at most* p*, and any* $r \in [q(n), M]$*, it can not distinguish the outputs of* $S_{\mathsf{inj}}(1^\lambda)$ *between* $S_{\mathsf{lossy}}(1^\lambda)$ *with advantage more than* δ.

Second, we introduce PRGs with sub-exponentially unpredictable seeds [Zha19b, WZ17].

Definition B.7 (PRG with Sub-Exponentially Unpredictable Seeds [Zha19b])**.** *A family of pseudorandom generators* $H : \mathcal{X} \to \mathcal{Y}$ *is secure for sub-exponentially unpredictable seeds if, for any sub-exponentially unpredictable distribution on* $(X, \mathcal{H}_Z)$*, no efficient adversary can distinguish* $(H, \rho_z, H(x))$ *from* (H, ρ_z, S) *where* $(x, \rho_z) \leftarrow D$ *and* $S \leftarrow U_Y$*, where* ρ_z *is a quantum auxiliary input.*

The following theorem follows from Appendix A in [WZ17]. Moreover, iO in the theorem statement can be further replaced with LWE using the construction in their work.

Theorem B.8 ([WZ17])**.** *Assuming PRGs with sub-exponentially unpredictable seeds and* iO*, there exists compute-and-compare obfuscation for sub-exponentially unpredictable distributions.*

B.2 PRGs with Sub-Exponentially Unpredictable Seeds

To prove Theorem B.1, we only need to prove PRGs with sub-exponentially unpredictable seeds can be built from lossy functions.

Most of the proof follows [Zha19b], except we are working with plain lossy functions (not extremely lossy functions), sub-exponentially unpredictable distributions and potentially quantum auxiliary information. We first look at the construction.

Construction 3. *Let q be the input length and m be the output length. Let λ be a security parameter. We will consider inputs x as q-dimensional vectors $\mathbf{x} \in \mathbb{F}_2^q$. Let LF be a lossy function (with some sub-linear residual leakage, which will be specified later). Let $M = 2^{m+\lambda+1}$, and let n be the output length of the lossy function. Set $N = 2^n$. Let ℓ be some polynomial in m, λ to be determined later. First, we will construct a function H' as follows.*

Choose random $f_1, \ldots, f_\ell \leftarrow \mathsf{LF}.S_{\mathsf{inj}}(1^\lambda)$ where $f_i : [M] \to [N]$, and let $h_1, \ldots, h_{\ell-1} : [N] \to [M/2] = [2^{m+\lambda}]$ and $h_\ell : [N] \to [2^m]$ be sampled from pairwise independent and uniform function families. Define $\mathbf{f} = \{f_1, \ldots, f_\ell\}$ and $\mathbf{h} = \{h_1, \ldots, h_\ell\}$. Define $H'_i : \{0,1\}^i \to [M/2]$ (and $H'_\ell : \{0,1\}^\ell \to [2^m]$) as follows:

- $H'_0() = 1 \in [2^{m+\lambda}]$
- $H'_i(\mathbf{b}_{[1,i-1]}, b_i)$: *compute* $y_i = H'_{i-1}(\mathbf{b}_{[1,i-1]})$, $z_i \leftarrow f_i(y_i || b_i)$, *and output* $y_{i+1} \leftarrow h_i(z_i)$

Then we set $H' = H'_\ell$. To define H, choose a random matrix $\mathbf{R} \in \mathbb{F}_2^{\ell \times q}$. The description of H consists of $\mathbf{f}, \mathbf{h}, \mathbf{R}$. We set $H(x) = H'(\mathbf{R} \cdot \mathbf{x})$. A diagram of H is given in Figure 4.

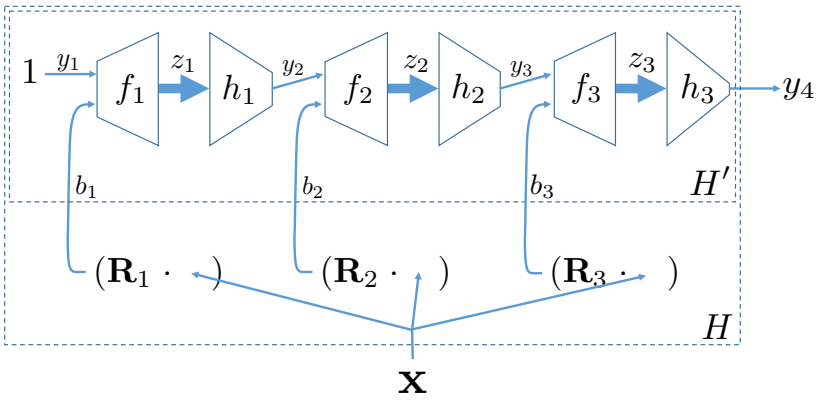

Figure 4: An example instantiation for $\ell = 3$, from [Zha19b].

We have the following theorem, which will finish the proof for Theorem B.1.

Theorem B.9. *Assuming lossy functions with sub-linear residual leakage, there exist PRGs with sub-exponentially unpredictable seeds (quantum auxiliary input).*

First, we have the claim from [Zha19b].

Claim B.10 (Claim 6.4, [Zha19b])**.** *If $\ell \geq m + \lambda$, and if $\mathbf{b}$ is drawn uniformly at random, then $(H', H'(\mathbf{b}))$ is statistically close to (H', R) where R is uniformly random in $[2^m]$.*

We will thus set $\ell = m + \lambda$ in our construction of H'. We now present our main theorem (Theorem B.9) of the section.

Theorem B.9. We show that H in Construction 3 is a pseudorandom generator that is secure for sub-exponentially unpredictable seeds.

Let λ be the security parameter. Let $(x, \rho_z) \leftarrow D$ be a sub-exponentially unpredictable distribution where $|x| = q$ (the input length of the PRG). In other words, there is no efficient algorithm

that given ρ_z, outputs x with probability more than $2^{-\lambda^{c_1}}$ for some constant $0 < c_1 \leq 1$. Let LF be a $(m + \lambda + 1, k)$-lossy function with sub-linear residual leakage r such that $2^{-r} \gg 2^{-\lambda^{c_1}}$. Note that there always exists a constant c such that $r = 2^{(m+\lambda+1)^c} \ll 2^{\lambda^{c_1}}$.

Recall that $H(\mathbf{x}) = H'(\mathbf{R} \cdot \mathbf{x})$, and that $H'(\mathbf{b})$ is statistically close to random when $\mathbf{b}$ is random (by Claim B.10). Therefore, it suffices to show that the following distributions are indistinguishable:

$$(\mathbf{f}, \mathbf{h}, \mathbf{R}, \rho_z, H'(\mathbf{R} \cdot \mathbf{x})) \text{ v.s. } (\mathbf{f}, \mathbf{h}, \mathbf{R}, \rho_z, H'(\mathbf{b})) \text{ for a uniformly random } \mathbf{b}.$$

Suppose an adversary $\mathcal{A}$ has non-negligible advantage ϵ in distinguishing the two distributions. Define $\mathbf{b}^{(i)}$ so that the first i bits of $\mathbf{b}^{(i)}$ are equal to the first i bits of $\mathbf{R} \cdot \mathbf{x}$, and the remaining $\ell - i$ bits are chosen uniformly at random independently of $\mathbf{x}$. Define **Hybrid** i to be the case where $\mathcal{A}$ is given the distribution $(\mathbf{f}, \mathbf{h}, \mathbf{R}, \rho_z, H'(\mathbf{b}^{(i)}))$.

We know that $\mathcal{A}$ distinguishes **Hybrid 0** from **Hybrid** ℓ with probability ϵ. Choose an i uniformly at random from $[\ell]$. Then the adversary distinguishes **Hybrid** $(i-1)$ from **Hybrid** i with expected advantage at least ϵ/ℓ. Next, observe that since bits $i+1$ through ℓ are random in either case, they can be simulated independently of the challenge. Moreover, $H'(\mathbf{b})$ can be computed given $H'_{i-1}(\mathbf{b}_{[i-1]})$, b_i (be random or equal to $\mathbf{R}_i, \mathbf{x}$), and the random $b_{i+1}, \ldots, b_\ell$. Thus, we can construct an adversary $\mathcal{A}'$ that distinguishes the following distributions:

$$(i, \mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x}), \mathbf{R}_i, \mathbf{R}_i \cdot \mathbf{x}) \text{ and } (i, \mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x}), \mathbf{R}_i, b_i)$$

with advantage ϵ/ℓ, where i is chosen randomly in $[\ell]$, $\mathbf{R}_{[i-1]}$ consists of the first $i-1$ rows of $\mathbf{R}$, $\mathbf{R}_i$ is the ith row of $\mathbf{R}$, and b_i is a random bit.

$\mathcal{A}'$ cannot distinguish f_i generated as $\mathsf{LF}.S_{\mathsf{lossy}}(1^\lambda)$ from the honest f_i generated from $\mathsf{LF}.S_{\mathsf{inj}}(1^\lambda)$, except with negligible probability. This means, if we generate $f_i \leftarrow \mathsf{LF}.S_{\mathsf{lossy}}(1^\lambda)$, we have that $\mathcal{A}'$ still distinguishes the distributions

$$(i, \mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x}), \mathbf{R}_i, \mathbf{R}_i \cdot \mathbf{x}) \text{ and } (i, \mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x}), \mathbf{R}_i, b_i) \quad (10)$$

with advantage $\epsilon' = \epsilon/\ell - 2 \cdot \mathsf{negl}$. Thus, given $(\mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x}), \mathbf{R}_i)$, $\mathcal{A}'$ is able to compute $\mathbf{R}_i \cdot \mathbf{x}$ with probability $\frac{1}{2} + \epsilon'$. Note that ϵ' is non-negligible.

Now fix $\mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}$, which fixes H'_{i-1}. Let $y_i = H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x})$. Notice that since $\mathbf{f}, \mathbf{h}$ are fixed, there are at most 2^r possible values for y_i. We now make the following claim:

Claim B.11. *Let $\mathcal{D}$ be a sub-exponentially unpredictable distribution on $\mathcal{X} \times \mathcal{H}_Z$, with guessing probability no more than $2^{-\lambda^{c_1}}$. Suppose $T : \mathcal{X} \to \mathcal{R}$ is drawn from a family $\mathcal{T}$ of efficient functions where the size of the image of T is 2^r. Then the following distribution is also computationally unpredictable: $(x, (T, \rho_z, T(x)))$ where $T \leftarrow \mathcal{T}$, $(x, \rho_z) \leftarrow \mathcal{D}$, with guessing probability no more than $2^r \cdot 2^{-\lambda^{c_1}}$ (as long as $r \ll \lambda^{c_1}$).*

Proof. Suppose we have an efficient adversary $\mathcal{B}$ that predicts x with non-negligible probability γ given $T, \rho_z, T(x)$, and suppose T has image size 2^r. We then construct a new adversary $\mathcal{C}$ that, given x, samples a random T, samples $(x', \rho_{z'}) \leftarrow \mathcal{D}$, and sets $a = T(x')$. It then runs $\mathcal{B}(T, \rho_z, a)$ to get a string x'', which it outputs. Notice that a is sampled from the same distribution as $T(x)$, so with probability at least $1/2^r$, $a = T(x)$. In this case, $x'' = x$ with probability γ. Therefore, $\mathcal{C}$ outputs x with probability $\gamma/2^r$, which can not be greater than $2^{-\lambda^{c_1}}$. Thus, γ is at most $2^r \cdot 2^{-\lambda^{c_1}}$. □

Using Claim B.11 with $T = H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x})$, we see that $(\mathbf{x}, (i, \mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x})))$ is computationally unpredictable. Moreover, $\mathbf{R}_i \cdot \mathbf{x}$ is a Goldreich-Levin [GL89] hardcore bit. We rely on the following lemma, which we will prove in the next section:

Lemma B.12 (Quantum Goldreich-Levin)**.** *If there exists a quantum algorithm, that given a random r and an auxiliary quantum input $|\psi_x\rangle$, it computes $\langle x, r\rangle$ with probability at least $1/2+\epsilon$ (where the probability is taken over the choice of x and random r); then there exists a quantum algorithm that takes $|\psi_x\rangle$ and extracts x with probability $4 \cdot \epsilon^2$.*

The same lemma holds if the quantum auxiliary input is a mixed state, by convexity.

Applying the quantum Goldreich-Levin theorem to the computationally unpredictable distribution $(\mathbf{x}, (i, \mathbf{f}, \mathbf{h}, \mathbf{R}_{[i-1]}, \rho_z, H'_{i-1}(\mathbf{R}_{[i-1]} \cdot \mathbf{x}))\ )$, we see that there exists an algorithm that extracts x with probability at least $4 \cdot \epsilon'^2$. This contradicts the computationally unpredictability of the underlying distribution, proving Theorem B.9.

□

B.3 Quantum Goldreich-Levin, with Quantum Auxiliary Input

In this section, we prove the final step:

Lemma B.12. *If there exists a quantum algorithm, that given random r and auxiliary quantum input $|\psi_x\rangle$, it computes $\langle x, r\rangle$ with probability at least $1/2 + \epsilon$ (where the probability is taken over the choice of x and random r); then there exists a quantum algorithm that takes $|\psi_x\rangle$ and extracts x with probability $4 \cdot \epsilon^2$.*

The same lemma holds if the quantum auxiliary input is a mixed state, by convexity.

The proof is the same as that in [AC02], but quantum auxiliary input about x is considered.

Proof. Assume there exists a unitary U, given r and an auxiliary quantum state $|\psi_x\rangle$, it computes $\langle x, r\rangle$ with probability more than $1/2 + \epsilon$. Since r is classical information, U can be modeled as: read r, applies U_r. For every x, r, we have:

$$\begin{aligned} U\,|r\rangle\,|\psi_x\rangle\,|\mathbf{0}^m\rangle &= |r\rangle\,U_r\,|\psi_x\rangle\,|\mathbf{0}^m\rangle \\ &= |r\rangle\left(\alpha_{x,r}\,|\langle x, r\rangle\rangle\,|\phi_{x,r}\rangle + \beta_{x,r}\,|\overline{\langle x, r\rangle}\rangle\,|\phi'_{x,r}\rangle\right) = |r\rangle\,|\Phi_{x,r}\rangle\,, \end{aligned}$$

where $|\mathbf{0}^m\rangle$ is the working space, $\alpha_{x,r}$ is the coefficient for computing $\langle x, r\rangle$ correctly and $\beta_{x,r}$ for an incorrect answer.

Let ϵ_x be the probability that the quantum algorithm answers correctly on x and R be the space of all r, we have the success probability as:

$$\mathbb{E}_r\left[|\alpha_{x,r}|^2\right] = \frac{1}{|R|}\sum_r |\alpha_{x,r}|^2 = 1/2 + \epsilon_x.$$

Now we fix an x and r. Our algorithm for extracting x does the following: it starts with the state above, then (1.) it applies a Z-gate(phase-flip gate) to get

$$\begin{aligned} &|r\rangle\left(\alpha_{x,r}(-1)^{\langle x,r\rangle}\,|\langle x, r\rangle\rangle\,|\phi_{x,r}\rangle + \beta_{x,r}(-1)^{\overline{\langle x,r\rangle}}\,|\overline{\langle x, r\rangle}\rangle\,|\phi'_{x,r}\rangle\right) \\ =\,&|r\rangle\,(-1)^{\langle x,r\rangle}\left(\alpha_{x,r}\,|\langle x, r\rangle\rangle\,|\phi_{x,r}\rangle - \beta_{x,r}\,|\overline{\langle x, r\rangle}\rangle\,|\phi'_{x,r}\rangle\right) \\ =\,&|r\rangle\,|\Phi'_{x,r}\rangle\,. \end{aligned}$$

Then (2.) it applies $U^\dagger$, because we have $\langle \Phi_{x,r} \,|\, \Phi'_{x,r} \rangle = (-1)^{\langle x,r \rangle}(|\alpha_{x,r}|^2 - |\beta_{x,r}|^2)$,

$$U^\dagger \,|r\rangle\, |\Phi'_{x,r}\rangle = |r\rangle \left((-1)^{\langle x,r \rangle}(|\alpha_{x,r}|^2 - |\beta_{x,r}|^2)\, |\psi_x\rangle\, |\mathbf{0}^m\rangle + |\mathsf{err}_{x,r}\rangle \right),$$

where $|\mathsf{err}_{x,r}\rangle$ is orthogonal to $|\psi_x\rangle\, |\mathbf{0}^m\rangle$.

In the first two step, we actually compute everything over a uniform superposition of r. Next (3.) it applies QFT on r register,

$$\begin{aligned} &\mathsf{QFT}\, \frac{1}{\sqrt{|R|}} \sum_r |r\rangle \left((-1)^{\langle x,r \rangle}(|\alpha_{x,r}|^2 - |\beta_{x,r}|^2)\, |\psi_x\rangle\, |\mathbf{0}^m\rangle + |\mathsf{err}_{x,r}\rangle \right) \\ =&\frac{1}{|R|} \sum_y \sum_r |y\rangle\, (-1)^{\langle y,r \rangle} \left((-1)^{\langle x,r \rangle}(|\alpha_{x,r}|^2 - |\beta_{x,r}|^2)\, |\psi_x\rangle\, |\mathbf{0}^m\rangle + |\mathsf{err}_{x,r}\rangle \right). \end{aligned}$$

Therefore, the phase on $|x\rangle\, |\psi_x\rangle\, |\mathbf{0}^m\rangle$ is at least,

$$\frac{1}{|R|} \sum_r \left(|\alpha_{x,r}|^2 - |\beta_{x,r}|^2 \right) \geq 2 \cdot \epsilon_x.$$

It measures r register and with probability at least $4 \cdot \epsilon_x^2$, it extracts x.

By convexity, the quantum algorithm succeeds in extracting x is at least $4 \cdot \epsilon^2$. □

C Proofs of Coset State Properties

C.1 Proof for Theorem 4.14

Proof of Theorem 4.14. In this section, we prove Theorem 4.14, the information-theoretic monogamy property of coset states. The proof resembles the proof of monogamy for BB84 states in [TFKW13]. However, the extra algebraic structure of subspace states requires a more refined analysis. We first state the lemmas that are required for the main theorem.

Assume $A \subseteq \mathbb{F}_2^n$ is of dimension $n/2$. We use $\mathsf{CS}(A)$ to denote the set of all cosets of A. Since $\dim(A) = n/2$, $|\mathsf{CS}(A)| = 2^{n/2}$. Note that if $A + s \neq A + s_0$, then they are disjoint. Because each coset $A + s$ of A has a canonical form, which is $\mathsf{Can}_A(s)$, we will identify $\mathsf{CS}(A)$ with the set of all canonical vectors (where cosets are identified with their canonical vectors).

We use R_2^n to denote the set of all subspaces of dimension $n/2$ in $\mathbb{F}_2^n$.

Lemma C.1. *Fixing a subspace A, the coset states $|A_{s,s'}\rangle$ and $|A_{s_0,s'_0}\rangle$ are orthogonal if and only if $A + s \neq A + s_0$ or $A' + s' \neq A' + s'_0$.*

Proof. If $A + s \neq A + s_0$, then $|A_{s,s'}\rangle$ has support over $A + s$ but $|A_{s_0,s'_0}\rangle$ has support over $A + s_0$. Because they have disjoint support, it is easy to see they are orthogonal.

If $A' + s' \neq A' + s'_0$, we can apply QFT and use the same argument in the Fourier domain. □

Lemma C.2. *Fixing A, $|A_{s,s'}\rangle$ for all $s, s' \in \mathsf{CS}(A), \mathsf{CS}(A^\perp)$ form a basis.*

Proof. We already know that the states $|A_{s,s'}\rangle$ and $|A_{s_0,s'_0}\rangle$ are orthogonal if $s, s' \neq s_0, s'_0$. Since there are total $2^{n/2} \times 2^{n/2} = 2^n$ states $|A_{s,s'}\rangle$, they form a basis. □

Lemma C.3. *Fixing A, $\frac{1}{2^{n/2}}\sum_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}|A_{s,s'},A_{s,-s'}\rangle = \frac{1}{2^{n/2}}\sum_{v\in\mathbb{F}_2^n}|v,v\rangle$. In other words, the summation is independent of A and it is an EPR pair.*

Proof.

$$\begin{aligned}\sum_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}|A_{s,s'},A_{s,-s'}\rangle &= \frac{1}{|A||A^\perp|}\sum_{s,s'\in\mathbb{F}_2^n}|A_{s,s'},A_{s,-s'}\rangle\\ &= \frac{1}{|A||A^\perp|}\sum_{s,s'\in\mathbb{F}_2^n}\frac{1}{|A|}\sum_{\substack{a\in A\\ b\in A}}(-1)^{\langle a-b,s'\rangle}|a+s\rangle|b+s\rangle\\ &= \frac{2^n}{|A||A^\perp|}\sum_{s\in\mathbb{F}_2^n}\frac{1}{|A|}\sum_{a\in A}|a+s\rangle|a+s\rangle\\ &= \sum_{s\in S}|s\rangle|s\rangle\end{aligned}$$

where the first equality comes from the fact that for any vectors $s_0\in A+s$ and $s_0'\in A^\perp+s'$, $|A_{s,s'}\rangle = |A_{s_0,s_0'}\rangle$. □

We want to prove the following statement:

Theorem C.4. *Fix $n\in\mathbb{N}$. For any Hilbert spaces $\mathcal{H}_B,\mathcal{H}_C$, any collections of POVMs*

$$\left\{\left\{P^A_{s,s'}\right\}_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}\right\}_{A\in R_2^n}\ \text{and}\ \left\{\left\{Q^A_{s,s'}\right\}_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}\right\}_{A\in R_2^n}$$

on the Hilbert spaces, and any CPTP map that maps $|A_{s,s'}\rangle\langle A_{s,s'}|$ into $\mathcal{D}(\mathcal{H}_B)\otimes\mathcal{D}(\mathcal{H}_C)$, we have that,

$$\mathbb{E}_{A\in R_2^n}\mathbb{E}_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}\mathsf{Tr}\left[\left(P^A_{s,s'}\otimes Q^A_{s,s'}\right)\cdot\Phi(|A_{s,s'}\rangle\langle A_{s,s'}|)\right]\leq 1/\mathsf{subexp}(n)$$

where subexp *is a sub-exponential function.*

Note that this bound directly gives Theorem 4.14, since both parties in Theorem 4.14 get the description of A, by applying $\mathsf{Can}_A(\cdot)$, one could map any vectors in $A+s$ and $A^\perp+s'$ to $\mathsf{Can}_A(s)$ and $\mathsf{Can}_{A^\perp}(s')$.

To prove Theorem 4.14 (and the above Theorem C.4), we present the following theorem about the monogamy game.

Theorem C.5. *Fix $n\in\mathbb{N}$. For any Hilbert spaces $\mathcal{H}_B,\mathcal{H}_C$, any collections of POVMs*

$$\left\{\left\{P^A_{s,s'}\right\}_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}\right\}_{A\in R_2^n}\ \text{and}\ \left\{\left\{Q^A_{s,s'}\right\}_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}\right\}_{A\in R_2^n}$$

on the Hilbert spaces, and any state ρ, we have

$$\mathbb{E}_{A\in R_2^n}\sum_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}\mathsf{Tr}\left[\left(|A_{s,s'}\rangle\langle A_{s,s'}|\otimes P^A_{s,s'}\otimes Q^A_{s,s'}\right)\cdot\rho\right]\leq 1/\mathsf{subexp}(n)$$

where subexp *is a sub-exponential function.*

Next, we show that to prove Theorem 4.14, we only need to show Theorem C.5.

Lemma C.6. *Theorem C.5 implies Theorem C.4 (and hence Theorem 4.14).*

Proof. For convenience, let $S = \mathbb{F}_2^n$. Assume there exists a strategy for the game in Theorem C.4 which achieves advantage δ. We construct a strategy (preparing ρ and POVMs) for the game in Theorem C.5 which achieves the same advantage.

1. Prepare the state $\rho = \frac{1}{|S|}(I \otimes \Phi) \sum_{s,s' \in S} |s,s\rangle\langle s',s'|$, which is equal to the following (for any subspace A) by Lemma C.3,

$$(I \otimes \Phi) \sum_{s,s' \in S} |s,s\rangle\langle s',s'| = (I \otimes \Phi) \sum_{\substack{s_1,s_1' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp) \\ s_2,s_2' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)}} |A_{s_1,s_1'}, A_{s_1,-s_1'}\rangle\langle A_{s_2,s_2'}, A_{s_2,-s_2'}|$$
$$= \sum_{\substack{s_1,s_1' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp) \\ s_2,s_2' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)}} |A_{s_1,s_1'}\rangle\langle A_{s_2,s_2'}| \otimes \Phi\left(|A_{s_1,-s_1'}\rangle\langle A_{s_2,-s_2'}|\right)$$

2. $\overline{P}^A_{s,s'} = P^A_{s,-s'}$ and $\overline{Q}^A_{s,s'} = Q^A_{s,-s'}$ where P, Q are POVMs for the game in Theorem C.4 and $\overline{P}, \overline{Q}$ are the POVMs for the game in Theorem C.5.

Thus, we have that the advantage is,

$$\begin{aligned}
&\mathbb{E}_{A \in R_2^n} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[\left(|A_{s,s'}\rangle\langle A_{s,s'}| \otimes \overline{P}^A_{s,s'} \otimes \overline{Q}^A_{s,s'}\right) \cdot \rho\right] \\
=&\mathbb{E}_{A \in R_2^n} \frac{1}{|S|} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[|A_{s,s'}\rangle\langle A_{s,s'}| \otimes \left(\left(\overline{P}^A_{s,s'} \otimes \overline{Q}^A_{s,s'}\right) \cdot \Phi\left(|A_{s,-s'}\rangle\langle A_{s,-s'}|\right)\right)\right] \\
=&\mathbb{E}_{A \in R_2^n} \frac{1}{|S|} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[|A_{s,s'}\rangle\langle A_{s,s'}| \otimes \left(\left(P^A_{s,-s'} \otimes Q^A_{s,-s'}\right) \cdot \Phi\left(|A_{s,-s'}\rangle\langle A_{s,-s'}|\right)\right)\right] \\
=&\mathbb{E}_{A \in R_2^n} \mathbb{E}_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[\left(P^A_{s,s'} \otimes Q^A_{s,s'}\right) \cdot \Phi(|A_{s,s'}\rangle\langle A_{s,s'}|)\right] = \delta
\end{aligned}$$

□

Without loss of generality, we can assume that the adversary's strategy is pure (see more discussion in Lemma 9, [TFKW13]). In other words, all $P^A_{s,s'}$ and $Q^A_{s,s'}$ are projections.

Proof of Theorem C.5. First, we define Π^A as

$$\Pi^A = \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} |A_{s,s'}\rangle\langle A_{s,s'}| \otimes P^A_{s,s'} \otimes Q^A_{s,s'}$$

Note that Π^A is a projection. By definition, the advantage is

$$\begin{aligned}
&\frac{1}{|R_2^n|} \sum_{A \in R_2^n} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[|A_{s,s'}\rangle\langle A_{s,s'}| \otimes P^A_{s,s'} \otimes Q^A_{s,s'} \cdot \rho\right] \\
\leq&\mathbb{E}_{v_1,\cdots,v_n} \left[\frac{1}{\binom{n}{n/2}} \sum_{A \in \mathsf{span}_{n/2}(v_1,\cdots,v_n)} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[|A_{s,s'}\rangle\langle A_{s,s'}| \otimes P^A_{s,s'} \otimes Q^A_{s,s'} \cdot \rho\right]\right]
\end{aligned}$$

where $(v_1, \cdots, v_n)$ range over all possible bases of the space, and $\mathsf{span}_{n/2}(v_1, \cdots, v_n)$ is the set of all subspaces spanned by exactly $n/2$ vectors in $(v_1, \cdots, v_n)$.

In other words, we decompose the sampling procedure of R_2^n into two steps: (1) sample a random basis; (2) choose $n/2$ vectors in the basis.

Then we have, for any fixed basis $v_1, \cdots, v_n$,

$$\begin{aligned}
&\frac{1}{\binom{n}{n/2}} \sum_{A \in \mathsf{span}_{n/2}(v_1,\cdots,v_n)} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} \mathsf{Tr}\left[|A_{s,s'}\rangle\langle A_{s,s'}| \otimes P^A_{s,s'} \otimes Q^A_{s,s'} \cdot \rho\right] \\
=& \frac{1}{\binom{n}{n/2}} \mathsf{Tr}\left[\sum_{A \in \mathsf{span}_{n/2}(v_1,\cdots,v_n)} \sum_{s,s' \in \mathsf{CS}(A),\mathsf{CS}(A^\perp)} |A_{s,s'}\rangle\langle A_{s,s'}| \otimes P^A_{s,s'} \otimes Q^A_{s,s'} \cdot \rho\right] \\
\leq& \frac{1}{\binom{n}{n/2}} \left| \sum_{A \in \mathsf{span}_{n/2}(v_1,\cdots,v_n)} \Pi^A \right|
\end{aligned}$$

where $|\cdot|$ is the ∞-Schatten norm.

Lemma C.7. *For every fixed basis $v_1, \cdots, v_n \in \mathbb{F}_2^n$, we have*

$$\frac{1}{\binom{n}{n/2}} \left| \sum_{A \in \mathsf{span}_{n/2}(v_1,\cdots,v_n)} \Pi^A \right| \leq \frac{1}{\binom{n}{n/2}} \sum_{t=0}^{n/2} \binom{n/2}{t}^2 2^{-t} = O\left(2^{-\sqrt{n}}\right)$$

If we can prove the above lemma, we finished our proof for Theorem C.5.

Proof of Lemma C.7. We first show the upper bound is sub-exponentially small. By the fact that $\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ for all $1 \leq k \leq n$, we have:

$$\begin{aligned}
\frac{1}{\binom{n}{n/2}} \sum_{t=1}^{n/2} \binom{n/2}{t}^2 2^{-t} \leq& \frac{1}{\binom{n}{n/2}} \sum_{t=1}^{\sqrt{n}} \binom{n/2}{t}^2 + 2^{-\sqrt{n}} \\
\leq& \frac{1}{2^{n/2}} \sum_{t=1}^{\sqrt{n}} \left(\frac{en}{2t}\right)^{2t} + 2^{-\sqrt{n}} \\
\leq& \frac{\sqrt{n}}{2^{n/2}} \cdot \left(\frac{en}{2}\right)^{\sqrt{n}} + 2^{-\sqrt{n}} \\
=& \exp(-\Omega(n - \sqrt{n}\log n)) + 2^{-\sqrt{n}}
\end{aligned}$$

Next, we prove the remaining part of the lemma. The idea is similar to that in [TFKW13].

We require the following lemma.

Lemma C.8 (Lemma 2 in [TFKW13])**.** *Let $A_1, A_2, \cdots, A_N \in P(\mathcal{H})$ (positive semi-definite operators on $\mathcal{H}$), and let $\{\pi_k\}_{k\in[N]}$ be a set of N mutually orthogonal permutations of $[N]$. Then,*

$$\left| \sum_{i \in [N]} A_i \right| \leq \sum_{k \in [N]} \max_{i \in [N]} \left| \sqrt{A_i} \sqrt{A_{\pi_k(i)}} \right|.$$

The set $\{\pi_k\}_{k\in[N]}$ is called a set of mutually orthogonal permutations, if for every $\pi \neq \pi'$ in the set, $\pi(i) \neq \pi'(i)$ for all $i \in [N]$.

Fixing basis $v_1, \cdots, v_n$, there are a total of $\binom{n}{n/2}$ subspaces that can be sampled by picking a subset of $\{v_1, \cdots, v_n\}$ of size $n/2$. So, in our case, $N = \binom{n}{n/2}$.

We define a collection of permutations on $\mathsf{span}_{n/2}(v_1, \cdots, v_n)$. Roughly speaking, the mapping does the following. It picks some basis vectors in the subspace and swaps them with basis vectors that are not used.

- Recall that each subspace A is described as $\{u_1, \cdots, u_{n/2}\}, \{u_{n/2+1}, \cdots, u_n\}$ where the subspace is spanned by $u_1, \cdots, u_{n/2}$. $\{u_{n/2+1}, \cdots, u_n\}$ are the vectors in $\{v_i\}_{i\in[n]}$ that are not in the subspace.
- The mapping is described by two subsets $I \subseteq \{1, 2, \cdots, n/2\}$ and $I' \subseteq \{n/2+1, \cdots, n\}$ of the same size.
- The mapping does the following:

$$\begin{aligned}\pi_{I,I'}\left(\{u_1, \cdots, u_{n/2}\}, \{u_{n/2+1}, \cdots, u_n\}\right) =& \{u_1, \cdots, u_{n/2}\} - \bigcup_{i\in I}\{u_i\} + \bigcup_{i'\in I'}\{u_{i'}\}, \\ & \{u_{n/2+1}, \cdots, u_n\} + \bigcup_{i\in I}\{u_i\} - \bigcup_{i'\in I'}\{u_{i'}\}\end{aligned}$$

  In other words, it removes the vectors with indices in I and adds vectors with indices in I'.

There are a total of $\sum_{t=0}^{n/2}\binom{n/2}{t}^2 = \binom{n}{n/2} = N$ permutations $\pi_{I,I'}$. They are mutually orthogonal: if $(I, I') \neq (I_0, I_0')$, for all subspace $A = \{u_1, \cdots, u_{n/2}\}, \{u_{n/2+1}, \cdots, u_n\}$, it is easy to see that $\pi_{I,I'}(A) \neq \pi_{I_0,I_0'}(A)$. Therefore, by Lemma C.8 and Π^A are all projections, we have

$$\frac{1}{\binom{n}{n/2}}\left|\sum_{A\in\mathsf{span}_{n/2}(v_1,\cdots,v_n)}\Pi^A\right| \leq \frac{1}{\binom{n}{n/2}}\sum_{\pi_{I,I'}}\max_{A\in\mathsf{span}_{n/2}(v_1,\cdots,v_n)}\left|\Pi^A\Pi^{\pi_{I,I'}(A)}\right|$$

Because Π^A is a projection, $\sqrt{\Pi^A} = \Pi^A$ for all A.

Next, we prove the following claim: for every $A, A' \in \mathsf{span}_{n/2}(v_1, \cdots, v_n)$, $|\Pi^A\Pi^{A'}| \leq 2^{\dim(A\cap A')-n/2}$.

Define

$$\begin{aligned}\overline{\Pi}^A &= \sum_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}|A_{s,s'}\rangle\langle A_{s,s'}| \otimes P^A_{s,s'} \otimes I \\ \overline{\Pi}^{A'} &= \sum_{s,s'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}|A_{s,s'}\rangle\langle A_{s,s'}| \otimes I \otimes Q^A_{s,s'}\end{aligned}$$

From the fact that (1) for two semi-definite operators A, B such that $A \leq B$, their ∞-Schatten norm satisfies $|A| \leq |B|$; (2) for a semi-definite operator A, $|A|^2 = |AA^\dagger|$, we have:

$$|\Pi^A\Pi^{A'}|^2 \leq |\overline{\Pi}^A\overline{\Pi}^{A'}|^2 = \left|\overline{\Pi}^A\overline{\Pi}^{A'}\overline{\Pi}^{A'}\overline{\Pi}^A\right| = \left|\overline{\Pi}^A\overline{\Pi}^{A'}\overline{\Pi}^A\right|$$

Then we have,

$$\begin{aligned}\overline{\Pi}^{A}\overline{\Pi}^{A'}\overline{\Pi}^{A} = &\sum_{\substack{s_1,s_1'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)\\ s_2,s_2'\in\mathsf{CS}(A'),\mathsf{CS}(A'^\perp)\\ s_3,s_3'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)}} |A_{s_1,s_1'}\rangle\langle A_{s_1,s_1'}|A'_{s_2,s_2'}\rangle\langle A'_{s_2,s_2'}|A_{s_3,s_3'}\rangle\langle A_{s_3,s_3'}| \\ &\qquad\qquad \otimes P^A_{s_1,s_1'}P^A_{s_3,s_3'}\otimes Q^{A'}_{s_2,s_2'} \\ = &\sum_{\substack{s_1,s_1'\in\mathsf{CS}(A),\mathsf{CS}(A^\perp)\\ s_2,s_2'\in\mathsf{CS}(A'),\mathsf{CS}(A'^\perp)}} |\langle A_{s_1,s_1'}|A'_{s_2,s_2'}\rangle|^2\cdot |A_{s_1,s_1'}\rangle\langle A_{s_1,s_1'}|\otimes P^A_{s_1,s_1'}\otimes Q^{A'}_{s_2,s_2'}\end{aligned}$$

Since for all s_1, s_1', s_2, s_2', $|A_{s_1,s_1'}\rangle\langle A_{s_1,s_1'}|\otimes P^A_{s_1,s_1'}\otimes Q^{A'}_{s_2,s_2'}$ are projections, its Schatten-∞ norm is bounded by the largest $|\langle A_{s_1,s_1'}|A'_{s_2,s_2'}\rangle|^2$.

$$|\langle A_{s_1,s_1'}|A'_{s_2,s_2'}\rangle| \le \frac{1}{2^{n/2}}\sum_{a\in S}[a\in A+s_1\ \wedge\ a\in A'+s_2] = 2^{\dim(A\cap A')}/2^{n/2}$$

This is because, for all basis vectors outside of $A\cap A'$, their coefficient is determined by s_1, s_2. Therefore, the only degree of freedom comes from the basis in $A\cap A'$.

Overall, $|\Pi^A\Pi^{A'}| \le 2^{\dim(A\cap A')}/2^{n/2}$. Thus,

$$\begin{aligned}\frac{1}{\binom{n}{n/2}}\sum_{\pi_{I,I'}}\max_{A\in\mathsf{span}_{n/2}(v_1,\cdots,v_n)}\left|\Pi^A\Pi^{\pi_{I,I'}(A)}\right| &\le \frac{1}{\binom{n}{n/2}}\sum_{t=0}^{n/2}\binom{n/2}{t}^2 2^{t-n/2} \\ &= \frac{1}{\binom{n}{n/2}}\sum_{t=0}^{n/2}\binom{n/2}{t}^2 2^{-t}\end{aligned}$$

Thus, we proved Lemma C.7. □

This completes the proof of Theorem C.5, and thus of Theorem 4.14. □

C.2 Proof of Theorem 4.15

Proof. We consider the following hybrids.

- Hyb 0: This is the original security game $\mathsf{CompMonogamy}$.
- Hyb 1: Same as Hyb 0 except $\mathcal{A}_0$ gets $\mathsf{iO}(\mathsf{shO}(A)(\cdot - s))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$, for a uniformly random superspace B of A, of dimension $3/4n$.
- Hyb 2: Same as Hyb 1 except $\mathcal{A}_0$ gets $\mathsf{iO}(\mathsf{shO}(B)(\cdot - s))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$, for a uniformly random superspace B of A, of dimension $3/4n$.
- Hyb 3: Same as Hyb 2 except for the following. The challenger samples s, s', A, and a uniformly random superspace B of A as before. The challenger sets $t = s + w_B$, where $w_B \leftarrow B$. Sends $\mathsf{iO}(\mathsf{shO}(B)(\cdot - t))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$ to $\mathcal{A}_0$.
- Hyb 4: Same as Hyb 3 except $\mathcal{A}_0$ gets $\mathsf{iO}(\mathsf{shO}(B)(\cdot - t))$, $\mathsf{iO}(\mathsf{shO}(A^\perp)(\cdot - s'))$ and $|A_{s,s'}\rangle$.

- Hyb 5: Same as Hyb 4 except $\mathcal{A}_0$ gets $\mathsf{iO}(\mathsf{shO}(B)(\cdot - t))$, $\mathsf{iO}(\mathsf{shO}(C^\perp)(\cdot - s'))$ and $|A_{s,s'}\rangle$, for a uniformly random subspace $C \subseteq A$ of dimension $n/4$.
- Hyb 6: Same as Hyb 5 except for the following. The challenger sets $t' = s' + w_{C^\perp}$, where $w_{C^\perp} \leftarrow C^\perp$. $\mathcal{A}_0$ gets $\mathsf{iO}(\mathsf{shO}(B)(\cdot - t))$, $\mathsf{iO}(\mathsf{shO}(C^\perp)(\cdot - t'))$ and $|A_{s,s'}\rangle$.
- Hyb 7: Same as Hyb 6 except the challenger sends B, C, t, t' in the clear to $\mathcal{A}_0$.

Lemma C.9. *For any QPT adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$|\Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \textit{ wins in Hyb 1}] - \Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \textit{ wins in Hyb 0}]| = \mathsf{negl}(\lambda)\,.$$

Proof. Suppose for a contradiction there was a QPT adversary $\mathcal{A}$ such that:

$$|\Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb 1}] - \Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb 0}]| \tag{11}$$

is non-negligible.

Such an adversary can be used to construct $\mathcal{A}'$ which distinguishes $\mathsf{iO}(A+s)$ from $\mathsf{iO}(\mathsf{shO}(A)(\cdot - s)$, which is impossible by the security of the (outer) iO, since $A + s$ and $\mathsf{iO}(A)(\cdot - s)$ compute the same functionality.

Fix n, let $A \subseteq \mathbb{F}_2^n$, $s, s' \in \mathbb{F}_2^n$ be such that the difference in (11) is maximized. Suppose $\Pr[\mathcal{A} \text{ wins in Hyb 1}] > \Pr[\mathcal{A} \text{ wins in Hyb 0}]$, the other case being similar.

$\mathcal{A}'$ proceeds as follows:

- Receives as a challenge a circuit P which is either $\mathsf{iO}(A + s)$ or $\mathsf{iO}(\mathsf{shO}(A)(\cdot - s))$. Creates the state $|A_{s,s'}\rangle$. Gives P, $\mathsf{shO}(A^\perp + s')$ and $|A_{s,s'}\rangle$ as input to $\mathcal{A}_0$.
- $\mathcal{A}_0$ returns a bipartite state. $\mathcal{A}'$ forwards the first register to $\mathcal{A}_1$ and the second to $\mathcal{A}_2$. $\mathcal{A}_1$ returns (s_1, s_1') and $\mathcal{A}_2$ returns (s_2, s_2'). $\mathcal{A}'$ checks If $s_1, s_2 \in A + s$ and $s_1', s_2' \in A^\perp + s'$. If so, $\mathcal{A}'$ guesses that $P = \mathsf{iO}(\mathsf{shO}(A)(\cdot - s))$, otherwise that $P = \mathsf{iO}(A + s)$.

It is straightforward to verify that $\mathcal{A}'$ succeeds at distinguishing with non-negligible probability. □

Lemma C.10. *For any QPT adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$|\Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \textit{ wins in Hyb 2}] - \Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \textit{ wins in Hyb 1}]| = \mathsf{negl}(\lambda)\,.$$

Proof. Suppose for a contradiction there was a QPT adversary $\mathcal{A}$ such that:

$$|\Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb 2}] - \Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb 1}]|\,,$$

is non-neglibile.

We argue that $\mathcal{A}$ can be used to construct an adversary $\mathcal{A}'$ that breaks the security of shO.

Fix n. Suppose $\Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb 2}] > \Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb 1}]$, the other case being similar.

$\mathcal{A}'$ proceeds as follows:

- Sample $A \subseteq \mathbb{F}_2^n$ uniformly at random. Send A to the challenger.

- The challenger returns a program P which is either $\mathsf{shO}(A)$ or $\mathsf{shO}(B)$ for a uniformly sampled superspace B. $\mathcal{A}'$ samples uniformly $s, s' \in \mathbb{F}_2^n$, and creates the state $|A_{s,s'}\rangle$. Gives $\mathsf{iO}(P(\cdot - s))$, $\mathsf{iO}(A^\perp + s')$ and $|A_{s,s'}\rangle$ as input to $\mathcal{A}_0$. The latter returns a bipartite state. $\mathcal{A}'$ forwards the first register to $\mathcal{A}_1$ and the second register to $\mathcal{A}_2$.
- $\mathcal{A}_1$ returns a pair (s_1, s_1') and $\mathcal{A}_2$ returns a pair (s_2, s_2'). $\mathcal{A}'$ checks that $s_1, s_2 \in A + s$ and $s_1', s_2' \in A^\perp + s'$. If so, then $\mathcal{A}'$ guesses that $P = \mathsf{shO}(B)$, otherwise that $P = \mathsf{shO}(A)$.

It is straightforward to verify that $\mathcal{A}'$ succeeds at the security game for shO with non-negligible advantage. □

Lemma C.11. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \text{ wins in Hyb } 3] - \Pr[\mathcal{A} \text{ wins in Hyb } 2]| = \mathsf{negl}(\lambda)\,.$$

Proof. The proof is similar to the proof of Lemma C.9, and follows from the security of iO and the fact that $\mathsf{shO}(B)(\cdot - s)$ and $\mathsf{shO}(B)(\cdot - t)$ compute the same functionality. □

Lemma C.12. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \text{ wins in Hyb } 4] - \Pr[\mathcal{A} \text{ wins in Hyb } 3]| = \mathsf{negl}(\lambda)\,.$$

Proof. The proof is analogous to that of Lemma C.9. □

Lemma C.13. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \text{ wins in Hyb } 5] - \Pr[\mathcal{A} \text{ wins in Hyb } 4]| = \mathsf{negl}(\lambda)\,.$$

Proof. The proof is analogous to that of Lemma C.10. □

Lemma C.14. *For any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A} \text{ wins in Hyb } 6] - \Pr[\mathcal{A} \text{ wins in Hyb } 5]| = \mathsf{negl}(\lambda)\,.$$

Proof. The proof is analogous to that of Lemma C.11. □

Lemma C.15. *For any QPT adversary $\mathcal{A}$ for Hyb 6, there exists an adversary $\mathcal{A}'$ for Hyb 7 such that*

$$\Pr[\mathcal{A}' \text{ wins in Hyb } 7] \geq \Pr[\mathcal{A} \text{ wins in Hyb } 6]\,.$$

Proof. This is immediate. □

Lemma C.16. *For any adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$,*

$$\Pr[(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2) \text{ wins in Hyb } 7] = \mathsf{negl}(\lambda)\,.$$

Proof. Suppose there exists an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ for Hyb 7 that wins with probability p.

We first show that, without loss of generality, one can take B to be the subspace of vectors such that the last $n/4$ entries are zero (and the rest are free), and one can take C to be such that the last $3/4n$ entries are zero (and the rest are free). We construct the following adversary $(\mathcal{A}_0', \mathcal{A}_1', \mathcal{A}_2')$ for the game where B and C have the special form above with trailing zeros, call these B_* and C_*, from an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ for the game of Hyb 7.

- $\mathcal{A}'_0$ receives a state $|A_{s,s'}\rangle$, together with t and t', for some $C_* \subseteq A \subseteq B_*$, where $t = s + w_{B_*}$ for $w_{B_*} \leftarrow B_*$, and $t' = s' + w_{C_*^\perp}$, where $w_{C_*^\perp} \leftarrow C_*^\perp$.

- $\mathcal{A}'_0$ picks uniformly random subspaces B and C of dimension $\frac{3}{4}n$ and $\frac{n}{4}$ respectively such that $C \subseteq B$, and a uniformly random isomorphism $\mathcal{T}$ mapping C_* to C and B_* to B. We think of $\mathcal{T}$ as a change-of-basis matrix. $\mathcal{A}'_0$ applies to $|A_{s,s'}\rangle$ the unitary $U_\mathcal{T}$ which acts as $\mathcal{T}$ on the standard basis elements. $\mathcal{A}'_0$ gives $U_\mathcal{T}|A\rangle$ to $\mathcal{A}_0$ together with B, C, $\mathcal{T}(t)$ and $(\mathcal{T}^{-1})^T(t')$. $\mathcal{A}'_0$ receives a bipartite state from $\mathcal{A}_0$. Forwards the first register to $\mathcal{A}'_1$ and the second register to $\mathcal{A}'_2$.

- $\mathcal{A}'_1$ forwards the received register to $\mathcal{A}_1$, and receives a pair (s_1, s'_1) as output. $\mathcal{A}'_1$ returns $(\mathcal{T}^{-1}(s_1), \mathcal{T}^T(s'_1))$ to the challenger. $\mathcal{A}'_2$ proceeds analogously.

First, notice that

$$\begin{aligned} U_\mathcal{T}|A_{s,s'}\rangle &= U_\mathcal{T} \sum_{v\in A} (-1)^{\langle v,s'\rangle} |v+s\rangle \\ &= \sum_{v\in A} (-1)^{\langle v,s'\rangle} |\mathcal{T}(v)+\mathcal{T}(s)\rangle \\ &= \sum_{w\in\mathcal{T}(\mathcal{A})} (-1)^{\langle \mathcal{T}^{-1}(w),s'\rangle} |w+\mathcal{T}(s)\rangle \\ &= \sum_{w\in\mathcal{T}(A)} (-1)^{\langle w,(\mathcal{T}^{-1})^T(s')\rangle} |w+\mathcal{T}(s)\rangle \\ &= |\mathcal{T}(A)_{z,z'}\rangle \ , \end{aligned}$$

where $z = \mathcal{T}(s)$ and $z' = (\mathcal{T}^{-1})^T(s')$.

Notice that $\mathcal{T}(A)$ is a uniformly random subspace between C and B, and that z and z' are uniformly random vectors in $\mathbb{F}_2^n$. Moreover, we argue that:

(i) $\mathcal{T}(t)$ is distributed as a uniformly random element of $z + B$.

(ii) $(\mathcal{T}^{-1})^T(t')$ is distributed as a uniformly random element of $z' + C^\perp$.

For (i), notice that

$$\mathcal{T}(t) = \mathcal{T}(s + w_{B_*}) = \mathcal{T}(s) + \mathcal{T}(w_{B_*}) = z + \mathcal{T}(w_{B_*})\,,$$

where w_{B_*} is uniformly random in B_*. Since $\mathcal{T}$ is an isomorphism with $\mathcal{T}(B_*) = B$, then $\mathcal{T}(w_{B_*})$ is uniformly random in B. Thus, $\mathcal{T}(t)$ is distributed as a uniformly random element in $z + B$.

For (ii), notice that

$$(\mathcal{T}^{-1})^T(t') = (\mathcal{T}^{-1})^T(s' + w_{C_*^\perp}) = (\mathcal{T}^{-1})^T(s') + (\mathcal{T}^{-1})^T(w_{C_*^\perp}) = z' + (\mathcal{T}^{-1})^T(w_{C_*^\perp})\,,$$

where $w_{C_*^\perp}$ is uniformly random in $C_*^\perp$. We claim that $(\mathcal{T}^{-1})^T(w_{C_*^\perp})$ is uniformly random in $C^\perp$. Notice, first, that the latter belongs to $C^\perp$. Let $x \in C$, then

$$\langle (\mathcal{T}^{-1})^T(w_{C_*^\perp}), x\rangle = \langle w_{C_*^\perp}, \mathcal{T}^{-1}(x)\rangle = 0\,,$$

where the last equality follows because $w_{C_*^\perp} \in C_*^\perp$, and $\mathcal{T}^{-1}(C) = C_*$. The claim follows from the fact that $(\mathcal{T}^{-1})^T$ is a bijection.

Hence, $\mathcal{A}_0$ receives inputs from the correct distribution, and thus, with probability p, both $\mathcal{A}_1$ and $\mathcal{A}_2$ return the pair $(z = \mathcal{T}(s), z'(\mathcal{T}^{-1})^T(s'))$. Thus, with probability p, $\mathcal{A}'_1$ and $\mathcal{A}'_2$ both return $(\mathcal{T}^{-1}(z), \mathcal{T}^T(z')) = (s, s')$ to the challenger, as desired.

So, we can now assume that B is the space of vectors such that the last $\frac{n}{4}$ entries are zero, and C is the space of vectors such that the last $\frac{3}{4}n$ entries are zero. Notice then that the sampled subspace A is uniformly random subspace subject to the last $\frac{n}{4}$ entries being zero, and the first $\frac{n}{4}$ entries being free. From an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ for Hybrid 7 with such B and C, we will construct an adversary $(\mathcal{A}'_0, \mathcal{A}'_1, \mathcal{A}'_2)$ for the information-theoretic monogamy where the ambient subspace is $\mathbb{F}_2^{n'}$, where $n' = \frac{n}{2}$.

- $\mathcal{A}'_0$ receives $|A_{s,s'}\rangle$, for uniformly random $A \subseteq \mathbb{F}_2^{n'}$ of dimension $n'/2$ and uniformly random $s, s' \in \mathbb{F}_2^{n'}$. $\mathcal{A}'_0$ samples $\tilde{s}, \tilde{s}', \hat{s}, \hat{s}' \leftarrow \mathbb{F}_2^{\frac{n}{4}}$. Let $|\phi\rangle = \frac{1}{2^{n/8}} \sum_{x \in \{0,1\}^{n/4}} (-1)^{\langle x, \tilde{s}' \rangle} |x + \tilde{s}\rangle$. $\mathcal{A}'_0$ creates the state
$$|W\rangle = |\phi\rangle \otimes |A_{s,s'}\rangle \otimes |\hat{s}\rangle \ ,$$
$\mathcal{A}'_0$ gives to $\mathcal{A}_0$ as input the state $|W\rangle$, together with $t = 0^{3n/4}||\hat{s} + w_B$ for $w_B \leftarrow B$ and $t' = \hat{s}'||0^{3n/4} + w_{C^\perp}$, for $w_{C^\perp} \leftarrow C^\perp$. $\mathcal{A}_0$ returns a bipartite state. $\mathcal{A}'_0$ forwards the first register to $\mathcal{A}'_1$ and the second register to $\mathcal{A}'_2$.

- $\mathcal{A}'_1$ receives A from the challenger. $\mathcal{A}'_1$ sends to $\mathcal{A}_1$ the previously received register, together with the the subspace $A' \subseteq \mathbb{F}_2^n$ whose first $n/4$ entries are free, the last $n/4$ entries are zero, and the middle $n/2$ entries belong to A. $\mathcal{A}_1$ returns a pair $(s_1, s'_1) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$. Let $r_1 = [s_1]_{\frac{n}{4}+1, \frac{3}{4}n} \in \mathbb{F}_2^{n/2}$ be the "middle" $n/2$ entries of s_1. Let $r_1 = [s'_1]_{\frac{n}{4}+1, \frac{3}{4}n} \in \mathbb{F}_2^{n/2}$. $\mathcal{A}'_1$ outputs (r_1, r'_1).

- $\mathcal{A}'_2$ receives A from the challenger. $\mathcal{A}'_2$ sends to $\mathcal{A}_2$ the previously received register, together with the the subspace $A' \subseteq \mathbb{F}_2^n$ whose first $n/4$ entries are free, the last $n/4$ entries are zero, and the middle $n/2$ entries belong to A. $\mathcal{A}_2$ returns a pair $(s_2, s'_2) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$. Let $r_2 = [s_2]_{\frac{n}{4}+1, \frac{3}{4}n} \in \mathbb{F}_2^{n/2}$ be the "middle" $n/2$ entries of s_2. Let $r'_2 = [s'_2]_{\frac{n}{4}+1, \frac{3}{4}n} \in \mathbb{F}_2^{n/2}$. $\mathcal{A}'_2$ outputs (r_2, r'_2).

Notice that

$$\begin{aligned}
|W\rangle &= |\phi\rangle \otimes |A_{s,s'}\rangle \otimes |\tilde{s}\rangle \\
&= \sum_{x \in \{0,1\}^{n/4}, v \in A} (-1)^{\langle x, \tilde{s}' \rangle} (-1)^{\langle v, s' \rangle} \Big| (x + \tilde{s})||(v + s)||\hat{s} \Big\rangle \\
&= \sum_{x \in \{0,1\}^{n/4}, v \in A} (-1)^{\langle (x||v||0^{n/4}), (\tilde{s}'||s'||\hat{s}') \rangle} \Big| x||v||0^{n/4} + \tilde{s}||s||\hat{s} \Big\rangle \\
&= \sum_{w \in \tilde{A}} (-1)^{\langle w, z' \rangle} |w + z\rangle = |\tilde{A}_{z,z'}\rangle \ ,
\end{aligned}$$

where $z = \tilde{s}||s||\hat{s}$, $z' = \tilde{s}'||s'||\hat{s}'$, and $\tilde{A} \subseteq \mathbb{F}_2^n$ is the subspace in which the first $n/4$ entries are free, the middle $n/2$ entries belong to subspace A, and the last $n/4$ entries are zero.

Notice that the subspace $\tilde{A}$, when averaging over the choice of A, is distributed precisely as in the game of Hybrid 7 (with the special choice of B and C); z, z' are uniformly random in $\mathbb{F}_2^n$; t is uniformly random from $z + B$, and t' is uniformly random from $z' + C^\perp$. It follows that, with probability p, the answers returned by $\mathcal{A}'_1$ and $\mathcal{A}'_2$ are both correct.

From the information-theoretic security of the monogamy game, Theorem 4.14, it follows that p must be negligible. □

D More Discussions On Anti-Piracy Security

D.1 Anti-Piracy Implies CPA Security

Lemma D.1. *If a single-decryptor encryption scheme satisfies CPA-style γ-anti-piracy security (Definition 6.4) for all inverse poly γ, it also satisfies CPA security.*

Proof. Let $\mathcal{A}$ be an adversary that breaks CPA security with advantage δ. We construct the following adversary $\mathcal{B}$ that breaks its CPA-style $(\delta/2)$-anti-piracy security.

$\mathcal{B}$ upon receiving a public key pk and a quantum key ρ_{sk}, it prepares the following programs:

- It runs the stateful adversary $\mathcal{A}$ on $(1^\lambda, \mathsf{pk})$, it outputs (m_0, m_1).
- Let $(\sigma[R_1], U_1)$ be the stateful algorithm $\mathcal{A}$ in the CPA security game (after outputting (m_0, m_1)), except when it outputs a bit b, it outputs m_b; $(\sigma[R_2], U_2)$ be the honest decryption algorithm using ρ_{sk}; $\mathsf{aux} = (m_0, m_1)$ be the output of $\mathcal{A}$.

First, we observe that $\sigma[R_1]$ and $\sigma[R_2]$ are un-entangled. For $(\sigma[R_1], U_1)$, because $\mathcal{A}$ wins CPA games with advantage δ, here it also outputs the correct message with probability $1/2 + \delta$. For $(\sigma[R_2], U_2)$, by the correctness of the scheme, it outputs the correct message with probability $1 - \mathsf{negl}(\lambda)$. Overall, $\mathcal{B}$ wins the game with probability $1/2 + \delta - \mathsf{negl}(\lambda) \gg 1/2 + \delta/2$. □

D.2 Strong Anti-Piracy Implies Regular Definition

In this section, we show the notion of strong anti-piracy security from Definition 6.11 implies that from Definition 6.4.

Proof. Assume a single-decryptor encryption scheme satisfies the strong notation of anti-piracy. For any adversary $\mathcal{A}$, consider the game $\mathsf{StrongAntiPiracy}$:

- At the beginning of the game, the challenger takes a security parameter λ and obtains keys $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$.
- The challenger sends $\mathcal{A}$ public-key pk and one copy of decryption key ρ_{sk} corresponding to pk.
- $\mathcal{A}$ finally outputs two (entangled) quantum decryptors $\mathsf{D}_1 = (\sigma[R_1], U_1)$ and $\mathsf{D}_2 = (\sigma[R_2], U_2)$ and $\mathsf{aux} = (m_0, m_1)$ $(m_0 \neq m_1)$
- The challenger outputs 1 (for $\mathcal{A}$ winning) if and only if *both* quantum decryptors $\mathsf{D}_1, \mathsf{D}_2$ are tested to be γ-good with respect to pk and aux.

It (the challenger outputs 1) happens with only negligible probability. In other words, with overwhelming probability over the distribution of $(\mathsf{sk}, \mathsf{pk})$ and (m_0, m_1), by applying projective measurement (the projective measurement $\mathcal{E}_1, \mathcal{E}_2$ inside both threshold implementations) and obtaining (d_1, d_0) and (d_1', d_0') for $\mathsf{D}_1, \mathsf{D}_2$ respecitvely, at least one of d_1, d_1' is smaller than $\frac{1}{2} + \gamma$, by Definition 6.8 (the definition of γ-good decryptor).

Also note that, in Definition 6.4, the game only differs in the test phase,

- The first three steps are identical to those in the above game.
- The challenger samples b_1, b_2 and r_1, r_2 uniformly at random and generates ciphertexts $c_1 = \mathsf{Enc}(\mathsf{pk}, m_{b_1}; r_1)$ and $c_2 = \mathsf{Enc}(\mathsf{pk}, m_{b_2}; r_2)$. The challenger runs D_1 on c_1 and D_2 on c_2 and it outputs 1 (the game is won by the adversary) if and only if D_1 outputs m_{b_1} and D_2 outputs m_{b_2}.

By the definition of projective measurement (Definition 3.10), the distribution of the second game, can be computed by its projective measurement. In other words, the test phase can be computed in the following equivalent way:

- Apply the projective measurement $\mathcal{E}_1, \mathcal{E}_2$ and obtain (d_1, d_0) and (d_1', d_0') for $\mathsf{D}_1, \mathsf{D}_2$ respectively. The challenger then samples two bits b_1, b_2 independently, where $b_1 = 1$ with probability d_1 and $b_2 = 1$ with probability d_1'. It outputs 1 if and only if $b_1 = b_2 = 1$.

Since we know that with overwhelming probability over the distribution of $(\mathsf{sk}, \mathsf{pk})$ and (m_0, m_1), by applying projective measurement and obtaining (d_1, d_0) and (d_1', d_0') for $\mathsf{D}_1, \mathsf{D}_2$ respecitvely, at least one of d_1, d_1' is smaller than $\frac{1}{2} + \gamma$, we can bound the probability of succeeding in the second game.

$$\begin{aligned}\Pr[\mathcal{A} \text{ succeeds}] &\leq 1 \cdot \Pr\left[d_1 \geq \frac{1}{2} + \gamma \wedge d_1' \geq \frac{1}{2} + \gamma\right] \\ &\quad + \left(\frac{1}{2} + \gamma\right) \cdot \Pr\left[d_1 \leq \frac{1}{2} + \gamma \vee d_1' \leq \frac{1}{2} + \gamma\right] \\ &\leq \mathsf{negl}(\lambda) + \left(\frac{1}{2} + \gamma\right)\end{aligned}$$

Therefore, it also satisfies the weak definition (Definition 6.4). □

D.3 Strong Anti-Piracy, with Random Challenge Plaintexts

Definition D.2 (Testing a quantum decryptor, with random challenge plaintexts)**.** *Let $\gamma \in [0, 1]$. Let pk be a public key. We refer to the following procedure as a γ-good test for a quantum decryptor with respect to pk and random challenge plaintexts:*

- *The procedure takes as input a quantum decryptor (ρ, U).*
- *Let $\mathcal{P} = (P, I - P)$ be the following mixture of projective measurements (in the sense of Definition 3.14) acting on some quantum state ρ':*
  - *Sample a uniform random message $m \leftarrow \mathcal{M}$. Compute $c \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$.*

– *Run the quantum decryptor* (ρ', U) *on input* c*. Check whether the outcome is* m*. If so, output* 1*, otherwise output* 0*.*

- *Let* $\mathsf{TI}_{1/|\mathcal{M}|+\gamma}(\mathcal{P})$ *be the threshold implementation of* $\mathcal{P}$ *with threshold value* $\frac{1}{|\mathcal{M}|}+\gamma$*, as defined in Definition 3.12. Run* $\mathsf{TI}_{1/|\mathcal{M}|+\gamma}(\mathcal{P})$ *on* (ρ, U)*, and output the outcome. If the output is* 1*, we say that the test passed, otherwise the test failed.*

Now we are ready to define the strong γ-anti-piracy game.

Definition D.3 ((Strong) γ-Anti-Piracy Game, with Random Challenge Plaintexts)**.** *A strong anti-piracy security game (for random plaintexts) for adversary* $\mathcal{A}$ *is denoted as* $\mathsf{StrongAntiPiracyGuess}(1^\lambda)$*, which consists of the following steps:*

1. ***Setup Phase****: At the beginning of the game, the challenger takes a security parameter* λ *and obtains keys* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$*.*
2. ***Quantum Key Generation Phase****: The challenger sends* $\mathcal{A}$ *the public-key* pk *and one copy of decryption key* ρ_{sk}*.*
3. ***Output Phase****: Finally,* $\mathcal{A}$ *outputs a (possibly mixed and entangled) state* σ *over two registers* R_1, R_2 *and two quantum circuits* (U_1, U_2)*. They can be viewed as two quantum decryptors* $\mathsf{D}_1 = (\sigma[R_1], U_1)$ *and* $\mathsf{D}_2 = (\sigma[R_2], U_2)$*.*
4. ***Challenge Phase****: The challenger outputs 1 (for* $\mathcal{A}$*'s winning) if and only if* both *quantum decryptors* $\mathsf{D}_1, \mathsf{D}_2$ *are tested to be* γ*-good with respect to* pk *and random challenge plaintexts.*

Definition D.4 ((Strong) γ-Anti-Piracy-Security)**.** *A single-decryptor encryption scheme satisfies strong* γ*-anti-piracy security against random plaintexts, if for any QPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that the following holds for all* $\lambda \in \mathbb{N}$*:*

$$\Pr\left[b = 1, b \leftarrow \mathsf{StrongAntiPiracyGuess}(1^\lambda)\right] \leq \mathsf{negl}(\lambda) \tag{12}$$

We claim Definition D.4 implies Definition 6.6. The proof is done in the same way as that in Appendix D.2. We omit the proof here.

To prove both our constructions satisfy the strong anti-piracy security against random messages:

- Construction based on strong monogamy property: the proof works in the exactly same way, except the compute-and-compare program $\mathsf{CC}[f, y, m_b]$ for a uniform bit b should be replaced with $\mathsf{CC}[f, y, m]$ for a uniformly random message m.
- Construction based on extractable witness encryption: the proof works in the exact same way.

D.4 Comparing Definition 6.4 with Definition 6.6

In Section 6, we define two anti-piracy security, namely Definition 6.4 for chosen plaintexts and Definition 6.6 for random plaintexts. In this section, we discuss their relationship.

One would hope that anti-piracy security against chosen plaintexts (Definition 6.4) implies anti-piracy security against random plaintexts (Definition 6.6), which is an analogue of security against chosen plaintext attack implies security against random plaintext attack (decrypting encryptions of

random messages). However, we realize that it is unlikely to be the case for anti-piracy security. Although it is not a formal proof, this intuition explains where thinks might fail.

Consider an adversary that breaks Definition 6.6. Assume it outputs the following decryptor state:

$$\left(\sqrt{\gamma}\,|\mathsf{good}\rangle + \sqrt{1-\gamma}\,|\mathsf{bad}\rangle\right)^{\otimes 2},$$

where $|\mathsf{good}\rangle$ is a perfect decryptor state and $|\mathsf{bad}\rangle$ is a garbage state that is orthogonal to $|\mathsf{good}\rangle$. It is easy to see that it breaks Definition 6.6 with advantage γ^2. However, each side can only win the CPA security game with advantage at most $1/2+\gamma$ independently. Therefore, its advantage for Definition 6.4 is $(1/2+\gamma)^2$, which is smaller than the trivial advantage $1/2$.

E Proof of Lemma 7.17

We are going to show that an adversary can not distinguish a pair of uniformly inputs from a pair of hidden trigger inputs by a sequence of hybrids. For simplicity, we first show the following lemma about the indistinguishability of a single random input or a single hidden trigger input. We will then show how the proof for Lemma E.1 translates to a proof for Lemma 7.17 easily.

Note that one can not get Lemma 7.17 by simply applying Lemma E.1 twice, as one can not sample a random hidden trigger input by only given the public information in the security game ($\mathsf{GenTrigger}$ requires knowing K_2, K_3), which is essentially required.

Lemma E.1. *Assuming post-quantum iO and one-way functions, for every efficient QPT algorithm $\mathcal{A}$, it can not distinguish the following two cases with non-negligible advantage:*

- *A challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and prepares a quantum key $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$. Here P hardcodes K_1, K_2, K_3.*
- *It samples a random input $u \leftarrow [N]$. Let $y = F_1(K_1, u)$. Parse the input as $u = u_0||u_1||u_2$.*
- *Let $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.*
- *It flips a coin b and outputs (ρ_K, u) or (ρ_K, u') depending on the coin.*

Note that we will mark the changes between the current hybrid and the previous hybrid in red.

Proof of Lemma E.1 .

Hybrid 0. This is the original game where the input is sampled either uniformly at random or sampled as a hidden triggers input.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y = F_1(K_1, u)$.

4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.

5. Generate the program P as in Figure 5. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin b.

Hardcoded: Keys K_1, K_2, K_3, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:
   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
2. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.
   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 5: Program P (same as Figure 3)

Hybrid 1 In this hybrid, the key K_1 in the program P is punctured at u, u'. The indistinguishability of Hybrid 0 and Hybrid 1 comes from the security of indistinguishability obfuscation.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to length $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y = F_1(K_1, u)$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. Generate the program P as in Figure 6. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin.

Hardcoded: Constants u, u'; Keys $K_1 \setminus \{u, u'\}, K_2, K_3$, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $x = u$ or u', it outputs $Q(v_1, \cdots, v_{\ell_0})$.
2. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:
   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
3. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.
   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 6: Program P

Note that starting from this hybrid, whenever we mention K_1 *inside a program* P, we mean to use the punctured key $K_1 \setminus \{u, u'\}$. Similar notations of punctured keys K_2, K_3 inside other programs will appear in the upcoming hybrids.

Hybrid 2. In this hybrid, the value of $F_1(K_1, u)$ is replaced with a uniformly random output. The indistinguishability of Hybrid 1 and Hybrid 2 comes from the pseudorandomness at punctured points of a puncturable PRF.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to length $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.

2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.

3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y \leftarrow [M]$.

4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q be the obfuscation program during the execution of $\mathsf{GenTrigger}$.

5. Generate the program P as in Figure 6. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin.

Hybrid 3. In this hybrid, the check on the second line will be skipped if x_1 is equal to u_1 or u'_1. By Lemma 2 of [SW14], adding this check does not affect its functionality, except with negligible probability.

The lemma says, to skip the check on the second line, x_1 will be equal to one of $\{u_1, u'_1\}$. To see why it does not change the functionality of the program, by Lemma 7.14 and for all but negligible fraction of all keys K_2, if $x_1 = u'_1$, there is only one way to make the check satisfied and the input is u_0, u'_2. This input $u' = u_0||u'_1||u'_2$ is already handled in the first line. Therefore, the functionality does not change.

After this change, $F_3(K_3, \cdot)$ will never be executed on those inputs. We can then puncture the key K_3 on them. The indistinguishability comes from the security of iO.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to length $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. Generate the program P as in Figure 7. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin.

Hardcoded: Constants u, u'; Keys $K_1 \setminus \{u, u'\}, K_2, K_3 \setminus \{u_1, u'_1\}$, R_i^0, R_i^1 for all $i \in [\ell_0]$. On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $x = u$ or u', it outputs $Q(v_1 \cdots, v_{\ell_0})$.
2. If $x_1 = u_1$ or u'_1, skip this check. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:

   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
3. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.

   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 7: Program P

Hybrid 4. In this hybrid, before checking $x_1 = F_2(K_2, x_0'||Q')$, it checks if $x_0'||Q' \neq u_0||Q$. Because if $x_0'||Q' = u_0||Q$ and the last check $x_1 = F_2(K_2, x_0'||Q')$ is also satisfied, we know that

$$x_1 = F_2(K_2, x_0'||Q') = F_2(K_2, u_0||Q) = u_1' \quad \text{(by the definition of } \mathsf{GenTrigger}\text{)}.$$

Therefore the step 2 will be skipped (by the first check). Thus, we can puncture K_2 at $u_0||Q$ The indistinguishability also comes from the security of iO.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s_i' for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s_i')$ (padded to length $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s_i'}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s_i'\}_{i\in[\ell_0]})$. Let Q be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. Generate the program P as in Figure 8. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin.

Hardcoded: Constants u, u'; Keys $K_1 \setminus \{u, u'\}$, $K_2 \setminus \{u_0||Q\}$, $K_3 \setminus \{u_1, u_1'\}$, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $x = u$ or u', it outputs $Q(v_1 \cdots)$.
2. If $x_1 = u_1$ or u_1', skip this check. If $F_3(K_3, x_1) \oplus x_2 = x_0'||Q'$ and $x_0 = x_0'$ and $x_0'||Q' \neq u_0||Q$, then also check $x_1 = F_2(K_2, x_0'||Q')$:

   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
3. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.

   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 8: Program P

Hybrid 5. In this hybrid, since the key K_2 has been punctured at $u_0||Q$, we can replace the evaluation of $F_2(K_2, \cdot)$ at the input with a uniformly random value. The indistinguishability comes from the pseudorandomness of the underlying puncturable PRF F_2.

We expand the $\mathsf{GenTrigger}$ procedure.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^{\perp} + s'_i)$ (padded to length $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$ as follows:
   (a) Let Q be the obfuscation of the program (padded to length $\ell_2 - \ell_0$) that takes inputs $v_1, \cdots, v_{\ell_0}$ and outputs y if and only if for every input v_i, if $u_{0,i} = 0$, then v_i is in $A_i + s_i$ and otherwise it is in $A_i^{\perp} + s'_i$.
   (b) $u'_1 \leftarrow [2^{\ell_1}]$ (since $F_2(K_2, u_0||Q)$ has been replaced with a uniformly random value).
   (c) $u'_2 \leftarrow F_3(K_3, u'_1) \oplus (u_0||Q)$.
   (d) It outputs $u' = u_0||u'_1||u'_2$.
5. Generate the program P as in Figure 8. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin.

Hybrid 6. In this hybrid, since the key K_3 has been punctured at u'_1, we can replace the evaluation of $F_3(K_3, \cdot)$ at u'_1 with a uniformly random value. The indistinguishability comes from the pseudorandomness of the underlying puncturable PRF F_3.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to length $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y \leftarrow [M]$.
4. It samples u' as follows:
   (a) $u'_1 \leftarrow [2^{\ell_1}]$;
   (b) $u'_2 \leftarrow [2^{\ell_2}]$.
   (c) It outputs $u' = u_0||u'_1||u'_2$.
5. Generate the program P as in Figure 8. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then u or u' depending on a random coin.

In this hybrids, u, u' are sampled independently, uniformly at random and they are symmetric in the program. The distributions for $b = 0$ and $b = 1$ are identical and even unbounded adversary can not distinguish these two cases. Therefore we finish the proof for Lemma E.1.

Remark E.2. *The program P depends on Q_u. Although Q_u is indexed by u, it only depends on u_0. Thus, the distributions for $b = 0$ and $b = 1$ are identical*

□

Finishing the proof for Lemma 7.17. The only difference between Lemma 7.17 and Lemma E.1 is the number of inputs sampled: either a single input u (or u') or a pair of independent inputs u, w (or u', w').

All hybrids for Lemma 7.17 are the same for the corresponding hybrids for Lemma E.1, except two inputs are sampled. Thus every time K_1, K_2 or K_3 are punctured according to u or u' in the proof of Lemma E.1, K_1, K_2 or K_3 are punctured *twice* according to both u, u' and w, w' in the proof of Lemma 7.17.

We are now giving the proof. If indistinguishability of some hybrid is not explained, it follows from the same reason as that in the corresponding hybrid in the proof of Lemma E.1.

Hybrid 0. The original game where both u, w are sampled uniformly at random or u', w' are random hidden trigger inputs.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u = F_1(K_1, u)$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w = F_1(K_1, w)$.
6. It samples $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
7. Generate the program P as in Figure 9. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

Hardcoded: Keys K_1, K_2, K_3, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:
   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
2. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.
   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 9: Program P

Hybrid 1. In this hybrid, the key K_1 in the program P is punctured at u, u', w, w'.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u = F_1(K_1, u)$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q_u be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w = F_1(K_1, w)$.
6. It samples $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q_w be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
7. Generate the program P as in Figure 10. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

Hardcoded: Keys $K_1 \setminus \{u, u', w, w'\}, K_2, K_3, R_i^0, R_i^1$ for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $x = u$ or u', it outputs $Q_u(v_1, \cdots, v_{\ell_0})$. If $x = w$ or w', it outputs $Q_w(v_1, \cdots, v_{\ell_0})$.
2. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:
   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
3. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.
   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 10: Program P

Hybrid 2. In this hybrid, y_u and y_w are sampled uniformly at random. Note that as long as $u \neq w$ (with overwhelming probability), we can apply the pseudorandomness at punctured points of a puncturable PRF.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q_u be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w \leftarrow [M]$.
6. It samples $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q_w be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
7. Generate the program P as in Figure 10. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

Hybrid 3. In this hybrid, P is changed by checking if x_1 is equal to u_1, u'_1, w_1 or w'_1. Moreover, K_3 is punctured at these points.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q_u be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w \leftarrow [M]$.
6. It samples $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$. Let Q_w be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
7. Generate the program P as in Figure 11. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

Hardcoded: Keys $K_1 \setminus \{u, u', w, w'\}, K_2, K_3 \setminus \{u_1, u'_1, w_1, w'_1\}$, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $x = u$ or u', it outputs $Q_u(v_1, \cdots, v_{\ell_0})$. If $x = w$ or w', it outputs $Q_w(v_1, \cdots, v_{\ell_0})$.
2. If x_1 is equal to u_1, u'_1, w_1 or w'_1, then skip this check. If $F_3(K_3, x_1) \oplus x_2 = x'_0||Q'$ and $x_0 = x'_0$ and $x_1 = F_2(K_2, x'_0||Q')$:

   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
3. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.

   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 11: Program P

Hybrid 4. In this hybrid, before checking $x_1 = F_2(K_2, x_0'||Q')$, it checks if $x_0'||Q' \neq u_0||Q_u$ and $x_0'||Q' \neq w_0||Q_w$. Because if $x_0'||Q' = u_0||Q_u$ and the last check $x_1 = F_2(K_2, x_0'||Q')$ is also satisfied, we know that

$$x_1 = F_2(K_2, x_0'||Q') = F_2(K_2, u_0||Q_u) = u_1' \quad \text{(by the definition of } \mathsf{GenTrigger}\text{)}.$$

Therefore the step 2 will be skipped (by the first check). Similarly, if $x_0'||Q' = w_0||Q_w$ and the last check $x_1 = F_2(K_2, x_0'||Q')$ is also satisfied, we know that

$$x_1 = F_2(K_2, x_0'||Q') = F_2(K_2, w_0||Q_w) = w_1' \quad \text{(by the definition of } \mathsf{GenTrigger}\text{)}.$$

Finally, we puncture K_2 at $u_0||Q_u$ and $w_0||Q_w$.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s_i' for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s_i')$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s_i'}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s_i'\}_{i\in[\ell_0]})$. Let Q_u be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w \leftarrow [M]$.
6. It samples $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s_i'\}_{i\in[\ell_0]})$. Let Q_w be the obfuscation program during the execution of $\mathsf{GenTrigger}$.
7. Generate the program P as in Figure 12. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

Hardcoded: Keys $K_1 \setminus \{u, u', w, w'\}, K_2 \setminus \{u_0||Q_u, w_0||Q_w\}, K_3 \setminus \{u_1, u_1', w_1, w_1'\}$, R_i^0, R_i^1 for all $i \in [\ell_0]$.
On input $x = x_0||x_1||x_2$ and vectors $v_1, \cdots, v_{\ell_0}$:

1. If $x = u$ or u', it outputs $Q_u(v_1, \cdots, v_{\ell_0})$. If $x = w$ or w', it outputs $Q_w(v_1, \cdots, v_{\ell_0})$.
2. If x_1 is equal to u_1, u_1', w_1 or w_1', then skip this check. If $F_3(K_3, x_1) \oplus x_2 = x_0'||Q'$ and $x_0 = x_0'$ and $x_0'||Q' \neq u_0||Q_u$ and $x_0'||Q' \neq w_0||Q_w$ and $x_1 = F_2(K_2, x_0'||Q')$:

   It treats Q' as a circuit and outputs $Q'(v_1, \cdots, v_{\ell_0})$.
3. Otherwise, it checks if the following holds: for all $i \in [\ell_0]$, $R^{x_{0,i}}(v_i) = 1$.

   If they all hold, outputs $F_1(K_1, x)$. Otherwise, outputs $\perp$.

Figure 12: Program P

Hybrid 5. In this hybrid, since the key K_2 has been punctured at $u_0||Q_u$ and $w_0||Q_w$, we can replace the evaluation of $F_2(K_2,\cdot)$ at these two inputs with uniformly random values, as long as $u_0 \neq w_0$ (with overwhelming probability). The indistinguishability comes from the pseudorandomness of the underlying puncturable PRF F_2.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u \leftarrow [M]$.
4. It samples $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$ as follows:
   (a) Let Q_u be the obfuscation of the program (padded to length $\ell_2 - \ell_0$) that takes inputs $v_1, \cdots, v_{\ell_0}$ and outputs y_u if and only if for every input v_i, if $u_{0,i} = 0$, then v_i is in $A_i + s_i$ and otherwise it is in $A_i^\perp + s'_i$.
   (b) $u'_1 \leftarrow [2^{\ell_1}]$ (since $F_2(K_2, u_0||Q_u)$ has been replaced with a uniformly random value).
   (c) $u'_2 \leftarrow F_3(K_3, u'_1) \oplus (u_0||Q_u)$.
   (d) It outputs $u' = u_0||u'_1||u'_2$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w \leftarrow [M]$.
6. It samples $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$ as follows:
   (a) Let Q_w be the obfuscation of the program (padded to length $\ell_2 - \ell_0$) that takes inputs $v_1, \cdots, v_{\ell_0}$ and outputs y_w if and only if for every input v_i, if $w_{0,i} = 0$, then v_i is in $A_i + s_i$ and otherwise it is in $A_i^\perp + s'_i$.
   (b) $w'_1 \leftarrow [2^{\ell_1}]$ (since $F_2(K_2, w_0||Q_w)$ has been replaced with a uniformly random value).
   (c) $w'_2 \leftarrow F_3(K_3, w'_1) \oplus (w_0||Q_w)$.
   (d) It outputs $w' = w_0||w'_1||w'_2$.
7. Generate the program P as in Figure 12. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

Hybrid 6. In this hybrid, since the key K_3 has been punctured at u'_1, w'_1, we can replace the evaluation of $F_3(K_3, \cdot)$ at u'_1, w'_1 with uniformly random values (as long as $u'_1 \neq w'_1$, which happens with overwhelming probability). The indistinguishability comes from the pseudorandomness of the underlying puncturable PRF F_3.

1. It samples random subspaces A_i of dimension $\lambda/2$ and vectors s_i, s'_i for $i = 1, 2, \cdots, \ell_0$. It then prepares programs $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^{\perp} + s'_i)$ (padded to the length upper bound $\ell_2 - \ell_0$). It prepares the quantum state $|\psi\rangle = \bigotimes_i |A_{i,s_i,s'_i}\rangle$.
2. It then samples keys K_1, K_2, K_3 for F_1, F_2, F_3.
3. It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u \leftarrow [M]$.
4. It samples u' as follows:
   (a) $u'_1 \leftarrow [2^{\ell_1}]$.
   (b) $u'_2 \leftarrow [2^{\ell_2}]$.
   (c) It outputs $u' = u_0||u'_1||u'_2$.
5. It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w \leftarrow [M]$.
6. It samples w' as follows:
   (a) $w'_1 \leftarrow [2^{\ell_1}]$.
   (b) $w'_2 \leftarrow [2^{\ell_2}]$.
   (c) It outputs $w' = w_0||w'_1||w'_2$.
7. Generate the program P as in Figure 12. The adversary is given $(|\psi\rangle, \mathsf{iO}(P))$ and then (u, w) or (u', w') depending on a random coin b.

In this hybrids, u, u', w, w' are sampled independently, uniformly at random and they are symmetric in the program. The distributions for $b = 0$ and $b = 1$ are identical and even unbounded adversary can not distinguish these two cases. Therefore we finish the proof for Lemma 7.17.

Remark E.3. *The program P depends on Q_u and Q_w. Although Q_u and Q_w are indexed by u and w, they only depend on u_0, w_0 respectively. Thus, the distributions for $b = 0$ and $b = 1$ are identical*

□

F Proof of Theorem 7.12

The proof for Theorem 7.12 is similar to proof for Theorem 7.11, but has some main differences in the final reduction. We highlight the changes made: the red-colored parts are the differences between the latter hybrid and the former hybrid; the blue-colored parts are differences between original hybrids in proof for Theorem 7.11 and these new hybrids.

Hybrid 0. Hybrid 0 is the original anti-piracy indistinguihsability security game.

1. A challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and prepares a quantum key $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$. Here P hardcodes K_1, K_2, K_3.
2. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
3. The challenger also prepares two inputs u, w as follows:
   - It samples u uniformly at random. Let $y_u = F_1(K_1, u)$.
   - It samples w uniformly at random. Let $y_w = F_1(K_1, w)$.
4. It samples y'_u, y'_w uniformly at random.
5. The challenger then samples uniform coins $b_0, b_1 \leftarrow \{0,1\}$. If $b_0 = 0$, give (u, y_u) to quantum program $(U_1, \sigma[R_1])$; else give (u, y'_u) to quantum program $(U_1, \sigma[R_1])$. Similarly, if $b_1 = 0$, give (w, y_w) to quantum program $(U_2, \sigma[R_2])$; else give (w, y'_w).
6. The outcome of the game is 1 if and only if both quantum programs successfully produce $b'_0 = b_0$ and $b'_1 = b_1$ respectively.

Hybrid 1 The changes between Hybrid 0 and 1 are exactly the two cases the adversary needs to distinguish between in the game of Lemma 7.17. Assume there exists an algorithm that distinguishes Hybrid 0 and 1 with *non-negligible* probability $\epsilon(\lambda)$, then these exists an algorithm that breaks the game in Lemma 7.17 with probability $\epsilon(\lambda) - \mathsf{negl}(\lambda)$.

The reduction algorithm receives ρ_k and u, w or u', w' from the challenger in Lemma 7.17; it computes y_u, y_w using $\mathsf{iO}(P)$ on the received inputs respectively and gives them to the quantum decryptor states $\sigma[R_1], \sigma[R_2]$. If they both output the guess correctly, then the reduction outputs 0 for u, w, otherwise it outputs 1 for u', w'.

1. A challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and prepares a quantum key $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$. Here P hardcodes K_1, K_2, K_3.
2. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
3. The challenger also prepares two inputs u', w' as follows:
   - It samples $u = u_0||u_1||u_2$ uniformly at random. Let $y_u = F_1(K_1, u)$. Let $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
   - It samples $w = w_0||w_1||w_2$ uniformly at random. Let $y_w = F_1(K_1, w)$. Let $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
4. It samples y'_u, y'_w uniformly at random.
5. The challenger then samples uniform coins $b_0, b_1 \leftarrow \{0,1\}$. If $b_0 = 0$, give (u', y_u) to quantum program $(U_1, \sigma[R_1])$; else give (u', y'_u) to quantum program $(U_1, \sigma[R_1])$. Similarly, if $b_1 = 0$, give (w', y_w) to quantum program $(U_2, \sigma[R_2])$; else give (w', y'_w).
6. The outcome of the game is 1 if and only if both quantum programs successfully produce $b'_0 = b_0$ and $b'_1 = b_1$ respectively.

Hybrid 2. In this hybrid, if $u_0 \neq w_0$ (which happens with overwhelming probability), $F_1(K_1, u)$ and $F_1(K_1, w)$ can be replaced with truly random strings. Since both inputs have enough min-entropy $\ell_1 + \ell_2 \geq m + 2\lambda + 4$ (as $u_1||u_2$ and $w_1||w_2$ are completely uniform and not given to the adversary) and F_1 is an extracting puncturable PRF, both outcomes y_u, y_w are statistically close to independently random outcomes. Thus, Hybrid 1 and Hybrid 2 are statistically close.

1. A challenger samples $K_1 \leftarrow \mathsf{Setup}(1^\lambda)$ and prepares a quantum key $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$. Here P hardcodes K_1, K_2, K_3.

2. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.

3. The challenger also prepares two inputs u', w' as follows:

   - It samples u_0 uniformly at random. It then samples a uniformly random y_u. Let $u' \leftarrow \mathsf{GenTrigger}(u_0, y_u, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.
   - It samples w_0 uniformly at random. It then samples a uniformly random y_w. Let $w' \leftarrow \mathsf{GenTrigger}(w_0, y_w, K_2, K_3, \{A_i, s_i, s'_i\}_{i\in[\ell_0]})$.

4. It samples y'_u, y'_w uniformly at random.

5. The challenger then samples uniform coins $b_0, b_1 \leftarrow \{0,1\}$. If $b_0 = 0$, give (u, y_u) to quantum program $(U_1, \sigma[R_1])$; else give (u, y'_u) to quantum program $(U_1, \sigma[R_1])$. Similarly, if $b_1 = 0$, give (w, y_w) to quantum program $(U_2, \sigma[R_2])$; else give (w, y'_w).

6. The outcome of the game is 1 if and only if both quantum programs successfully produce $b'_0 = b_0$ and $b'_1 = b_1$ respectively.

Hybrid 3.

1. A challenger first samples $\{A_i, s_i, s'_i\}_{i\in[\ell_0]}$ and prepares the quantum states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}$. It treat the the quantum states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}$ as the quantum decryption key ρ_{sk} for our single-decryptor encryption scheme and the secret key sk is $\{A_i, s_i, s'_i\}_{i\in[\ell_0]}$. Similarly, let $\mathsf{pk} = \{R_i^0, R_i^1\}_{i\in[\ell_0]}$ where $R_i^0 = \mathsf{iO}(A_i + s_i)$ and $R_i^1 = \mathsf{iO}(A_i^\perp + s'_i)$.

2. It samples y_u, y_w uniformly at random. It also samples y'_u, y'_w uniformly at random.

3. Then it flips two random coins $b_0, b_1 \leftarrow \{0,1\}$. If $b_0 = 1$, let $(u_0, Q_0) \leftarrow \mathsf{Enc}(\mathsf{pk}, y_u)$; else let $(u_0, Q_0) \leftarrow \mathsf{Enc}(\mathsf{pk}, y'_u)$. Similarly, if $b_1 = 1$, let $(w_0, Q_1) \leftarrow \mathsf{Enc}(\mathsf{pk}, y_w)$; else, let $(w_0, Q_1) \leftarrow \mathsf{Enc}(\mathsf{pk}, y'_w)$. $\mathsf{Enc}(\mathsf{pk}, \cdot)$ is the encryption algorithm of the underlying single-decryptor encryption scheme using pk.

4. The challenger constructs the program P which hardcodes K_1, K_2, K_3. It then prepares ρ_K, which is $(\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$.

5. $\mathcal{A}$ upon receiving ρ_K, it runs and prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.

6. The challenger also prepares two inputs u', w' as follows (as $\mathsf{GenTrigger}$ does):

- Let $u'_1 \leftarrow F_2(K_2, u_0||Q_0)$ and $u'_2 \leftarrow F_3(K_3, u'_1) \oplus (u_0||Q_0)$. Let $u' = u_0||u'_1||u'_2$.
- Let $w'_1 \leftarrow F_2(K_2, w_0||Q_1)$ and $w'_2 \leftarrow F_3(K_3, w'_1) \oplus (w_0||Q_1)$. Let $w' = w_0||w'_1||w'_2$.

7. The challenger again samples uniform coins $\delta_0, \delta_1 \leftarrow \{0,1\}$. If $\delta_0 = 0$, give (u', y_u) to quantum program $(U_1, \sigma[R_1])$; else give (u', y'_u) to quantum program $(U_1, \sigma[R_1])$. Similarly, if $\delta_1 = 0$, give (w', y_w) to quantum program $(U_2, \sigma[R_2])$; else give (w', y'_w).

8. The outcome of the game is 1 if both quantum programs successfully produce the answers below respectively:

   - If $(U_1, \sigma[R_1])$ outputs 0 and $b_0 = \delta_0$, or if it outputs 1 and $b_0 \neq \delta_0$, then $(U_1, \sigma[R_1])$ succeeds. Otherwise it fails.
   - If $(U_2, \sigma[R_2])$ outputs 0 and $b_1 = \delta_1$, or if it outputs 1 and $b_1 \neq \delta_1$, then $(U_2, \sigma[R_2])$ succeeds. Otherwise it fails.

Note that the only differences of Hyb 2 and Hyb 3 are the orders of executions and that the challenger prepares u', w' from one of (y_u, y'_u) and one of (y_w, y'_w) respectively, instead of preparing them from y_u, y_w first and choosing random y'_u, y'_w later. Since we will check if the random coins match in Step 8 of Hybrid 3, the game is essentially the same to an adversary as the game in Hybrid 2.

Given an algorithm $\mathcal{A}$ that wins the indistinguishability anti-piracy game for PRF in Hybrid 3 with non-negligible probability $\gamma(\lambda)$, we can build another algorithm $\mathcal{B}$ that breaks the (regular) CPA-style γ-anti-piracy security (see Definition 6.4) of the underlying single-decryptor encryption scheme.

- $\mathcal{B}$ plays as the challenger in the game of Hybrid 3.
- $\mathcal{B}$ will get $\rho_{\mathsf{sk}} = \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}$ and $\mathsf{pk} = \{\mathsf{iO}(A_i + s_i), \mathsf{iO}(A_i^{\perp} + s'_i)\}_{i\in[\ell_0]}$ in the anti-piracy game.
- $\mathcal{B}$ prepares K_1, K_2, K_3 and the program P. Let $\rho_K = (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\ell_0]}, \mathsf{iO}(P))$.
- $\mathcal{B}$ gives ρ_K to $\mathcal{A}$ and $\mathcal{A}$ prepares a pair of (potentially entangled) quantum states $\sigma[R_1], \sigma[R_2]$ as well as quantum circuits U_1, U_2.
- $\mathcal{B}$ also samples uniform random y_u, y'_u, y_w, y'_w and sends (y_u, y'_u) and (y_w, y'_w) as the challenge plaintexts for the two quantum programs, to the challenger of single-decryptor encryption anti-piracy game.
- $\mathcal{B}$ then creates quantum programs $\mathsf{P}_1, \mathsf{P}_2$ which will do the following steps.
- P_1 receives challenge ciphertext $\mathsf{ct}_0 = (u_0, Q_0)$ (which will be encryption of either y_u or y'_u), and P_2 receives challenge ciphertext $\mathsf{ct}_1 = (w_0, Q_1)$ (which will be encryption of either y_w or y'_w). They each independently prepares u', w' as follows :
  - Let $u'_1 \leftarrow F_2(K_2, u_0||Q_0)$ and $u'_2 \leftarrow F_3(K_3, u'_1) \oplus (u_0||Q_0)$. Let $u' = u_0||u'_1||u'_2$.
  - Let $w'_1 \leftarrow F_2(K_2, w_0||Q_1)$ and $w'_2 \leftarrow F_3(K_3, w'_1) \oplus (w_0||Q_1)$. Let $w' = w_0||w'_1||w'_2$.

- P_1 gives either (u', y_u) or (u', y'_u) depending on a random coin $\delta_0 \leftarrow \{0,1\}$, to $(\sigma[R_1], U_1)$; P_2 gives either (w', y_w) or (w', y'_w) depending on random coin $\delta_1 \leftarrow \{0,1\}$, to $(\sigma[R_2], U_2)$.
- Then P_1 and P_2 respectively run $(\sigma[R_1], U_1)$ and $(\sigma[R_2], U_2)$ on their challenge received to output their answers a_1 and a_2.
- Finally, depending on answers received and the coins δ_0, δ_1, P_1 and P_2 does the following:

  For P_1: if $(\sigma[R_1], U_1)$ outputs 0(which means the program thinks it receives an input and its PRF evaluation):

  - if $\delta_0 = 0$: P_1 outputs 0 (for encryption of y_u) to the challenger.
  - else, $\delta_0 = 1$: P_1 outputs 1 (for encryption of y'_u) to the challenger.

  If $(\sigma[R_1], U_1)$ outputs 1 (which means the program thinks it receives an input and a random value):

  - if $\delta_0 = 0$: P_1 outputs 1 (for encryption of y'_u) to the challenger.
  - else, $\delta_0 = 1$: P_1 outputs 0 (for encryption of y_u) to the challenger.

  Similarly on the P_2 and $(\sigma[R_2], U_2)$ side.

We observe that the advantage of $\mathcal{B}$ in the CPA-style γ-anti-piracy game of single-decryptor encryption is the same as advantage of $\mathcal{A}$ in the indistinguishability anti-piracy game for PRF. $\square$