Semi-Quantum Copy-Protection and More

Céline Chevalier[1,2], Paul Hermouet[1,2], and Quoc-Huy Vu[3]

[1] DIENS, École normale supérieure, PSL University, CNRS, INRIA, Paris, France
[2] CRED, Université Panthéon-Assas, Paris II, France
[3] LIP6, Sorbonne Université, Paris, France
{celine.chevalier, paul.hermouet, quoc.huy.vu}@ens.fr

Abstract. Properties of quantum mechanics have enabled the emergence of quantum cryptographic protocols achieving important goals which are proven to be impossible classically. Unfortunately, this usually comes at the cost of needing quantum power from every party in the protocol, while arguably a more realistic scenario would be a network of classical clients, classically interacting with a quantum server.
In this paper, we focus on copy-protection, which is a quantum primitive that allows a program to be evaluated, but not copied, and has shown interest especially due to its links to other unclonable cryptographic primitives. Our main contribution is to show how to dequantize existing quantum copy-protection from hidden coset states, by giving a construction for classically-instructed remote state preparation for coset states. We also present the first secure copy-protection scheme for point-functions in the plain model, to which our dequantizer can be applied.

1 Introduction

Quantum mechanical effects have enabled the construction of cryptographic primitives that are impossible classically. In particular, the no-cloning principle of quantum mechanics, which means that an unknown quantum state cannot be copied in general, has given rise to many wonderful primitives such as quantum money [Wie83, AC12], quantum lightning [Zha19], quantum copy-protection [Aar09], one-shot signatures [BS17, AGKZ20], secure software leasing [AL21], unclonable encryption [BL20] and many more. By standard definition, these quantum primitives can be seen as a two-party protocol requiring quantum communication to transfer the quantumly encoded program between parties, and of course, local quantum computation from both parties. Besides the fact that there is a fundamental difference between classical and quantum communication, a more realistic and practical scenario would be a *classical* communication network with classical clients interacting with a single powerful quantum server. With only classical communication, however, these notions of quantum cryptography become unusable.

[1] This is called *hybrid quantum cryptography* in [AGKZ20].

Semi-quantum cryptography.[1] Ideally, for both theoretical and practical reasons, we might want to minimize the required model and use local quantum computation and only classical communication. In this research direction, an emerging field of "dequantizing" quantum cryptographic protocols has shown that it is possible to use local quantum computation and classical communication to obtain cryptographic constructions which are otherwise classically impossible [Mah18b, BCM+18, RS20, AGKZ20, KNY21, HMNY21, Shm22a, Shm22b, GMP22]. Building on techniques introduced in [Mah18b, BCM+18, GV19], recent work [GMP22] has shown how to construct a *classically-instructed parallel remote state preparation of BB84 states* which then can be used to dequantize quantum copy-protection for certain narrow classes of functions [CMP20]. These protocols rely on the idea of conjugate coding, which is based on four states, also referred to as *BB84 states*: $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$ where $|\pm\rangle := \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$.

Quantum cryptography from coset states. Given a subspace $A \subseteq \mathbb{F}_2^n$, the corresponding *subspace state* is defined as a uniform superposition over all vectors in the subspace A, i.e., $|A\rangle := \frac{1}{\sqrt{|A|}} \sum_{v \in A} |v\rangle$. The idea of using hidden subspace state to construct quantum cryptographic primitives was first proposed by Aaronson and Christiano in [AC12] in the oracle model where the parties have access to some membership checking oracles. This idea was realized subsequently in the plain model using indistinguishability obfuscation by Zhandry [Zha19]. The subspace state idea was later generalized to *coset states* in [CLLZ21, VZ21], which can be seen as quantum one-time pad encrypted subspace states. Formally, for a subspace $A \subseteq \mathbb{F}_2^n$ and two vectors $s, s' \in \mathbb{F}_2^n$, the corresponding coset state is defined as $|A_{s,s'}\rangle := \frac{1}{\sqrt{|A|}} \sum_{x \in A} (-1)^{\langle x, s' \rangle} |x + s\rangle$. The coset state idea has shown a broad range of applications to signature tokens, unclonable decryptors, copy-protection [CLLZ21], classical proof of quantum knowledge [VZ21], public semi-quantum money [Shm22a], semi-quantum signature tokens [Shm22b], and unclonable encryption [AKL+22].

Perhaps the most striking example of quantum primitives is the notion of *quantum copy-protection*, introduced by Aaronson [Aar09]. Informally, quantum copy-protection allows a program to be encoded in a quantum state in such a way that the program can be evaluated, but not copied. It is also interesting to highlight the relation between copy-protection and other unclonable cryptography functionalities. Indeed, Ananth and Kaleoglu [AK21] show that the existence of an unclonable encryption scheme with a strong security property implies the existence of copy-protection of point-functions. Sattath and Wyborski [SW22] show that copy-protection of a certain family of functions allows for the construction of unclonable decryptors. We then explore the possibility of constructing semi-quantum copy-protection.

(Semi-)Quantum copy-protection from coset states. To the best of our knowledge, all known provably secure copy-protection schemes with standard

malicious security are based on hidden coset states [CLLZ21, AKL+22].[2] In these protocols, the basic idea is to encode the program into random hidden coset states and send these states as copy-protection of the program to the receiver.

The main distinction between random BB84 states and coset states is the (un)learnability with verification oracles: when the verification oracles are accessible, the former is learnable while the latter is unlearnable. This explains why coset states have more applications, mostly in the public-key setting. On the other hand, known constructions of semi-quantum cryptography from coset states are dequantized using an application-specific approach [Shm22a, Shm22b], and there is no construction of semi-quantum copy-protection. In contrast, [GMP22]'s construction of semi-quantum cryptography from BB84 states is generic and applicable to many existing constructions (including the one for copy-protection of point functions [CMP20]). However, the disadvantage of [GMP22]'s construction is that the dequantized protocols only have inverse polynomial security, due to the inverse polynomial soundness of the [GMP22] protocol itself.

In this work, we therefore focus on the application of coset states to quantum copy-protection and ask the following question:

Can we achieve semi-quantum copy-protection from coset states, with standard security?

1.1 Our Results

We answer this open question affirmatively. In fact, we give a construction for *classically-instructed remote state preparation for coset states*, based on the existence of indistinguishability obfuscation for classical circuits, and on that the Learning With Errors [Reg05] problem.

Theorem 1 (Informal). *Assume that LWE is sub-exponentially hard for quantum computers and that indistinguishability obfuscation for classical circuits exists with sub-exponential security against quantum polynomial-time adversaries. Then, there is a classically-instructed remote state preparation protocol for coset states with negligible soundness (as defined in Definition 2).*

Our protocol is a multi-round protocol between classical Alice and quantum polynomial-time Bob that allows Alice to delegate the construction of hidden coset states to Bob. Furthermore, Alice knows the description of the constructed coset states (which reside on Bob's device), while Bob himself does not, and no-cloning also applies to these states.[3] Hence, the situation at the end of this protocol is equivalent to one where Alice sent hidden coset states to Bob, allowing us to dequantize existing quantum copy-protection from coset states in a generic and modular way. We note that our remote coset state preparation protocol can also be

[2] The only known exception is the construction of copy-protection of single-bit point functions in the quantum random oracle model based on BB84 states [AKL23]. In this work, we focus only on constructions in the plain model.

[3] These coset states actually satisfy a strong monogamy of entanglement property, which we describe later in Section 2.

applied to dequantizing other coset-state based quantum protocols (for example, public-key quantum money [Shm22a] and tokenized signatures [Shm22b]), which may be of independent interest.

Corollary 1 (Informal). *Assume that LWE is sub-exponentially hard for quantum computers and that indistinguishability obfuscation for classical circuits exists with sub-exponential security against quantum polynomial-time adversaries. Then, there is a semi-quantum copy-protection from coset states for certain class of functions.*

To broaden the applicability of our semi-quantum protocol, we also present in this work a copy-protection of point functions in the plain model, whose security is also based on the idea of hidden coset states. We note that until now, no copy-protection scheme for point functions in the plain model with negligible security was known.

Theorem 2 (Informal). *Assume the existence of post-quantum indistinguishability obfuscation, one-way functions, and compute-and-compare obfuscation for the class of unpredictable distributions. Then, there is a secure copy-protection scheme for point-functions in the plain model.*

In fact, our copy-protection scheme is almost identical to that for pseudorandom functions given in [CLLZ21]. We observe that by making few modifications to their proof, we obtain a copy-protection of point functions with a non-trivial challenge distribution in the security definition.

1.2 Organization

The remaining of the paper is organized as follows. In Section 2 we explain the main ideas in our construction and the overview of the proof of soundness of our protocol. The preliminaries are given in Section 3. In Section 4, we present our construction of classically-instructed remote state preparation for coset states with correctness proof. The formal proof of soundness of our protocol is given in Section 5. These parts contain the main technical contribution of our paper. In the final section (Section 6), we present our construction for quantum copy-protection of point functions, and show how to apply our remote state preparation protocol to obtain a semi-quantum copy-protection construction.

1.3 Related Work

One can see our protocol as an interactive protocol between a classical verifier and an (untrusted) prover, in which the verifier classically instructs the prover to prepare some hidden quantum states, which *satisfy certain properties*. This is highly relevant to a series of works starting with [BCM+18, Mah18b, Mah18b] that have developed techniques to allow the verifier to force the prover to *behave in a certain way*. We note that the former kind of protocols is implied by the latter, while the other direction is not true. For example, in our protocol, it is

still possible that the prover does not behave in an expected way, but its output at the end of the protocol still satisfies the defined property. In the following, these protocols are termed as *semi-quantum* protocols.

In a breakthrough result [Mah18b], Mahadev introduced a protocol that allows a classical verifier to verifiably delegate a quantum computation to an untrusted quantum prover. The key ingredient of Mahadev's protocol is a *measurement* protocol, which allows the client to delegate single-qubit measurements in the standard or Hadamard basis to a quantum prover, and be able to efficiently verify the measurement outcome, assuming that the prover cannot break certain cryptographic assumptions.[4] This yields the first kind of semi-quantum protocols, so-called *prepare-and-measure protocols*. These protocols involve a quantum prover preparing and sending a quantum state to the verifier and the verifier performing single-qubit measurements on this state. One can use Mahadev's measurement protocol to delegate these quantum measurements to the prover itself. This is in contrast to the other type of quantum protocols: *prepare-and-send* protocols, in which the verifier prepares and sends quantum states to the prover.

It turns out that replacing the quantum communication of prepare-and-send protocols is significantly harder than doing so for prepare-and-measure protocols. At a high level, the main difference is the following: Mahadev's measurement protocol shows that *there exists* a quantum state that is consistent with the distribution of measurement outcomes reported by the prover. (One can think about this similarly as the *proof of membership* in an interactive proof system.) In contrast, if we want to replace the step of the verifier sending a physical quantum state to the prover, we need to show that the prover has *actually constructed* a certain quantum state, not just that such a quantum state exists. This is done by establishing a *rigidity* argument. The idea of rigidity, first formally introduced by Mayers and Yao [MY04], is that certain games can be used to "self-test" quantum states: if such a game is won with high enough probability, then the self-test property tells us that the players must hold some quantum state, up to local isometry. In our context, a game is a model of the protocol under consideration, and the game is won if the prover passes the client's verification.

Lying strictly between the two notions of "*there exists*" and "*actually constructed*" is the notion of *classical proof of quantum knowledge* formalized in [VZ21]. They also show that indeed Mahadev's measurement protocol achieves this stronger notion of *proof of knowledge*, in the sense that the prover "*knows*" the state it is measuring, not just that it exists mathematically. The same property also holds for a coset states prepare-and-send protocol: if the verifier sends a hidden coset state to the prover, later on the prover can prove that it "knows" the received coset state. Although this setting seems close to our protocol, we note that the proof given in [VZ21] does not directly carry on to our setting, where we replace quantum communication by classical communication.

[4] These cryptographic assumptions can be based on the quantum hardness of the Learning with Errors problem [Reg05].

The first semi-quantum protocol that provably forces a quantum prover to prepare a certain quantum state is the single-qubit remote state preparation protocol of [GV19] (see also [CCKW19] for a related result). [MV21] gives a protocol that allows a classical verifier to certify that a quantum prover must have prepared and measured a Bell state, i.e., an entangled 2-qubit quantum state. Finally, [GMP22], by developing new techniques to show a n-fold parallel rigidity proof, gives the first parallel remote BB84 state preparation protocol. Their proof technique is the backbone of our soundness proof presented later in Section 5. The most interesting point of the [GMP22] protocol is that it allows to dequantize a number of BB84 states-based quantum cryptographic primitives, yielding a generic and modular way of translating these protocols to a setting where only classical communication is used. The downside of the [GMP22] protocol is that it only achieves *inverse polynomial* soundness, which means that their dequantized protocols can only achieve *inverse polynomial* security at most, even if the original quantum protocols have negligible security.

In addition to this line of work focused on rigidity statements, application-specific semi-quantum protocols were considered for private-key quantum money [RS20], certifiable deletion of quantum encryption [HMNY21], secure software leasing [KNY21], public-key quantum money [Shm22a] and tokenized signature [Shm22b]. The common points of these protocols are that: (i) their approaches are less generic and modular than the [GMP22] protocol and the protocol we present in this work; (ii) new analysis are required for each application. However, we note that all these application-specific semi-quantum protocols achieve *negligible* security, as they do not prove that the prover in their protocol behave in a certain way, but only that the output of the prover at the end satisfies certain properties. This is also the approach that we take in this work, which we describe in more details in Section 2.

Readers who are familiar with the context might think that our dequantization of coset state generation protocol with monogamy-of-entanglement property can be achieved readily from previous works by Shmueli ([Shm22a, Shm22b]). However, this is not quite true, due to the fact that Shmueli's works only show coset state delegation protocols with *direct product hardness* properties, and not *monogamy-of-entanglement* properties. These two kinds of unclonability properties are very different in their nature: while *direct product hardness* can be used in the constructions of quantum money and tokenized signatures (where there is only a single adversary playing the unclonability game), it cannot be used in the context of quantum copy-protection (where there are two or more adversaries simultaneously playing the unclonability game). In fact, *direct product hardness* does not imply *monogamy-of-entanglement*, and the former will be trivially broken if one casts it under the monogamy-of-entanglement game. We refer the readers to [CLLZ21] for detailed definitions and applications of these two unclonability properties.

Copy-Protection of Point Functions. The first construction for copy-protection of point functions was presented in [CMP20], which is based on BB84 states and its security is proven in the quantum random oracle model. However, [CMP20]'s

construction only achieves *constant* security. If we consider a weaker security notion, so-called *secure software leasing* [AL21], [BJL+21] shows that we can even construct secure software leasing of point functions unconditionally in the plain model with negligible security.

On the other hand, copy-protection with negligible security against malicious adversaries are only known for pseudorandom functions, single-decryptor and point functions [CLLZ21, AKL+22]. While the copy-protection schemes for pseudorandom functions and single-decryptor are secure in the plain model [CLLZ21], the security of the construction for point functions is proven in the quantum random oracle model [AKL+22]. These latter constructions are all based on the idea of hidden coset states. In a recent work, [AKL23] gives another construction for copy-protection of single-bit point functions in the quantum random oracle model based on BB84 states. In this work, we build upon the copy-protection construction of pseudorandom functions [CLLZ21] to construct copy-protection of point functions in the plain model.

Acknowledgements

This work was supported in part by the French ANR projects CryptiQ (ANR-18-CE39-0015) and SecNISQ (ANR-21-CE47-0014).

2 Technical Overview

2.1 Our Semi-Quantum Copy-Protection Protocol

Security Requirements. We first start with security analysis of known coset states-based quantum protocols. In particular, we focus on the copy-protection of pseudorandom functions scheme in the plain model and the single-decryptor scheme in the plain model presented in [CLLZ21]. The security of these constructions reduce to a *monogamy of entanglement* property of coset states [CLLZ21, CV22]. Informally, this property states that a triple of quantum algorithms Alice, Bob and Charlie cannot cooperatively win the following monogamy 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. No communication is allowed between Bob and Charlie. 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 $A^{\perp} + s'$, where $A^{\perp}$ denote the dual subspace of A.

If our goal is to design a semi-quantum protocol for preparing coset states such that it can be used in a plug-and-play manner for the aforementioned protocols, our protocol needs to have the following properties:

- *Correctness.* If the prover is honest, at the end of the protocol execution, the prover must have a hidden coset state $|A_{s,s'}\rangle$ in its registers.

- *Soundness.* No (computationally bounded) prover after interacting with the classical verifier in the protocol, can win the monogamy of entanglement game described above (with a single modification in the first step of the game: instead of sending the coset state to the prover, we run the protocol). For a formal definition of the soundness, see Definition 2. We note that the soundness property also implies the blindness property: an untrusted prover cannot know the description of A and s, s' through the interaction.

The first attempt. Having described all requirements needed, we now turn into our protocol construction. Our starting point is the recent public semi-quantum money in the plain model introduced by Shmueli in [Shm22a], which uses hybrid quantum homomorphic encryption (QFHE)[5] and indistinguishability obfuscation ($i\mathcal{O}$) as the building blocks. The scheme is as follows.

1. The classical verifier $\mathcal{V}$ samples a random $\frac{\lambda}{2}$-dimensional subspace $A \subseteq \mathbb{F}_2^\lambda$ (represented by a matrix $\mathbf{M}_A \in \{0,1\}^{\frac{\lambda}{2}\times\lambda}$), and sends to the prover $\mathcal{P}$ $(\mathbf{M}_A^{p_x}, \mathsf{ct}_{p_x})$, an encryption of the matrix $\mathbf{M}_A$ under QFHE.
2. $\mathcal{P}$ homomorphically evaluates the circuit C, which is a quantum circuit that gets as input the classical description of a subspace $A \subseteq \mathbb{F}_2^\lambda$ and generates a uniform superposition over A. $\mathcal{P}$ obtains a homomorphically evaluated ciphertext
$$(|A_{x,z}\rangle, \mathsf{ct}_{x,z}) \leftarrow \mathsf{QFHE.Eval}\big(\mathsf{pk}, (\mathbf{M}_A^{p_x}, \mathsf{ct}_{p_x}), C\big),$$
and sends the classical part $\mathsf{ct}_{x,z}$ to $\mathcal{V}$.
3. $\mathcal{V}$ decrypts $(x,z) \leftarrow \mathsf{QFHE.Decrypt}(\mathsf{sk}, \mathsf{ct}_{x,z})$ and sends obfuscated membership check programs $i\mathcal{O}(A+x)$, $i\mathcal{O}(A^\perp + z)$ to $\mathcal{P}$.

Unfortunately, there is an efficient "splitting" attack that breaks the monogamy game described above (even if the adversary does not receive the description of A in the question phase) (see [Shm22b] for the description of the attack). Indeed, in the construction of semi-quantum tokenized signatures [Shm22b], which is also based on the [Shm22a] construction above, the author also needs to overcome this problem by carefully changing the security property required for the signature setting. We forego the details of his approach, but we note that his approach is not applicable in our setting. The main difference is that in our setting, there are two simultaneous non-communicating adversaries that also receive the description of A in the monogamy game.

The second attempt: running self-testing protocol under QFHE. We make an important observation: in the semi-quantum protocol for preparing BB84 states (also called a self-testing protocol) of [GMP22], instead of asking the prover $\mathcal{P}$ to prepare its own states (which are polynomially many $|+\rangle$ states

[5] A hybrid QFHE scheme is one where every encryption of a quantum state $|\psi\rangle$ consists of a quantum one-time pad encryption of $|\psi\rangle$ with Pauli keys $(x,z) \in \{0,1\}^*$, and $\mathsf{ct}_{x,z}$ which is a classical FHE encryption of the Pauli keys.

if $\mathcal{P}$ is honest), the verifier $\mathcal{V}$ can send the input to $\mathcal{P}$ using QFHE. In particular, $\mathcal{V}$ sends encryption of $\mathbf{M}_0$, which is the all-zero matrix. $\mathcal{P}$ homomorphic evaluates a quantum circuit C on the received ciphertext such that if the input matrix is all-zero, C evaluates to a uniform superposition over $\mathbb{F}_2^\lambda$, which is product of $|+\rangle$ states. Under QFHE encryption, the quantum part of the evaluated ciphertext is product of random $|\pm\rangle$ states. $\mathcal{P}$ then uses this in the [GMP22] self-testing protocol. We will see that an honest prover $\mathcal{P}$ using product of $|+\rangle$ states as in the [GMP22] protocol or $\mathcal{P}$ using product of $|\pm\rangle$ states does not change the correctness of the protocol, while its soundness is maintained (since the soundness does not depend on which input the prover has used in the protocol execution).

We now briefly give a description of the [GMP22] protocol, and refer the reader to their paper and Section 4.1 for more details. The verifier first runs a number of *test* rounds, where the prover is asked to measure its entire quantum state. These test rounds are used by the verifier to check whether the prover behaves as intended. Once the verifier is convinced of this, the verifier runs a *preparation* round. Test and preparation rounds are indistinguishable from the point of view of the prover, except that unlike in a test round, in a preparation the prover is not asked to measure its final state. Essentially, the [GMP22] protocol can be seen as a 1-over-n cut-and-choose protocol, in which the verifier run n rounds of testing, and 1 round of preparation from $n+1$ indistinguishable instances. The soundness statement for the test rounds is a self-testing statement, which characterizes which states and measurements the prover used in the protocol. The soundness of the [GMP22] protocol follows from that of the test rounds via a statistical cut-and-choose argument. In the following, we focus on the test sub-protocol.

The main cryptographic primitive underlying the [GMP22] protocol (as well as other self-testing protocols [GV19, MV21]) is the so-called extended noisy trapdoor claw-free function (ENTCF) family[6], which can be constructed assuming the quantum hardness of LWE [Mah18b]. An ENTCF family is a family of functions indexed by a set of keys $\mathcal{K}_0 \cup \mathcal{K}_1$. $\mathcal{K}_0$ and $\mathcal{K}_1$ are disjoint sets of keys with the property that the two sets are computationally indistinguishable.

We first describe the single-qubit remote preparation protocol from [GV19], as the [GMP22] test protocol is a n-fold parallel of [GV19].

1. For a given *basis choice* $\theta \in \{0,1\}$ (where "0" corresponds to the computational and "1" to the Hadamard basis), the verifier $\mathcal{V}$ samples a key $k \in \mathcal{K}_\theta$, alongside some trapdoor information t. $\mathcal{V}$ sends k to the prover $\mathcal{P}$ and keeps t private.
2. The verifier and prover then interact classically.
3. For us, the most relevant part is the last round of the protocol, i.e., the last message from the verifier to the prover and back. Before the last round, the remaining quantum state of an *honest* prover is the single-qubit state $|v\rangle_\theta$ for $v \in \{0,1\}$, where $|v\rangle_\theta$ is a conjugate encoding of v in the basis θ: if $\theta = 0$, $|v\rangle_\theta = |v\rangle$, otherwise $|v\rangle_\theta = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^v |1\rangle)$. From the transcript and the

[6] We refer the reader to [Mah18b, Section 4] for further details on ENTCF families.

trapdoor information, the verifier can compute v; in contrast, the prover, which does not know the trapdoor, cannot efficiently compute θ or v. In the last round, the verifier sends θ to the prover, who returns $v' \in \{0,1\}$; the verifier then checks whether $v' = v$. The honest prover would generate v' by measuring its remaining qubit $|v\rangle_\theta$ in the basis θ and therefore always pass the verifier's check.

In the [GMP22] test protocol, $\mathcal{V}$ runs n independent copies of [GV19] in parallel, except that the basis choice θ_i is the same for each copy. Next, from the [GMP22] protocol, we describe a self-testing protocol for coset states.

Assume that now the verifier has private input which is a description of a coset state (A, x, z). We modify the verification procedure of the [GMP22] test protocol in the last round as follows. Let $\vec{v}$ be the last message sent by $\mathcal{P}$ to $\mathcal{V}$ in the protocol above. If $\theta = 0$ (note that the basis choice is the same for n copies), $\mathcal{V}$ checks if $\vec{v} \in A + x$, otherwise, it decodes[7] $\vec{v}$ to get a vector $\vec{w}$ and checks if $\vec{w} \in A^\perp + z$. An honest prover would use the coset state $|A_{x,z}\rangle$, which it obtains after running the [Shm22a] protocol described above, as the input to this self-testing protocol. The honest prover would have measured its state in the computational basis when $\theta = 0$, and in the Hadamard basis when $\theta = 1$. Thus, any honest prover would pass this self-testing protocol for coset states with probability 1.

The crucial point is that, since the prover's input in both the [GMP22] self-testing protocol and the self-testing protocol for coset states described above is encrypted under QFHE, and the fact that the two protocols are identical from the prover point's of view (except the last verification procedure, which is hidden from the prover), the two protocols are computationally indistinguishable. In other words, any computationally bounded prover cannot distinguish if it is playing in the [GMP22] self-testing protocol or the coset-state self-testing protocol. This allows us to carry the rigidity argument of the [GMP22] protocol to our setting. We elaborate more on this later in Section 2.2. For time being, let's say we have showed that if the prover $\mathcal{P}$ passes the verification, it must have "used" a coset state in the self-testing protocol (with inverse polynomial soundness).

However, our ultimate goal is to perform a remote state preparation protocol (and not just self-testing). Our final step would be to run this coset-state self-testing protocol in the n-over-$2n$ cut-and-choose fashion: the verifier first sends $2n$ encrypted coset states and $|+\rangle$ to the prover, and it picks n instances uniformly at random for the self-testing protocol. The remaining n instances are used as the output of the final protocol. Building on the simple but powerful "quantum cut-and-choose" formalism of Bouman and Fehr [BF10], we can show that if the prover passes all the test instances, it must have at least 1 coset state in its registers at the end of the protocol (with inverse polynomial soundness). Notably, we will show that even if we only obtain inverse polynomial soundness at this step, our final protocol still achieves negligible security for a monogamy of entanglement

[7] We omit the details of this decoding procedure, and refer the reader to Section 4.1. We note that with the trapdoor t, this procedure can be implemented efficiently by the verifier.

game, which is the main property used in many copy-protection schemes. (An overview of the soundness proof is given below in Section 2.2.)

Our final protocol. Our final protocol (Protocol 5) works as follows:

(1) The verifier first sends homomorphic encryption that allows the prover to either construct coset states or BB84 states.
(2) The prover is asked to homomorphically evaluate the instructed circuits and return classical encryption of the one-time pads of the homomorphic encryption, and keep the quantum parts.
(3) Next, the prover and the verifier run a number of self-testing rounds (Protocol 3), in which each test round consists of testing either BB84 states (Protocol 1) or coset states (Protocol 2), forming several test blocks. (In particular, a test block consists of a number of BB84 states testing rounds, and one coset states testing round.) All the BB84 states are consumed after this step, while only half of the coset states are consumed.
(4) Once the verifier is convinced, the verifier runs the coset states generation round on the remaining half of the coset states, in which the verifier sends back to the prover obfuscation of the membership checking programs. The final state of the prover can then be used in coset states based constructions. To be more precise, the output state of a single run of our protocol would satisfy the monogamy of entanglement property that we described above. If a quantum copy-protection scheme requires n random coset states, we can simply run our protocol n times (with independent randomness for each instance).

2.2 Soundness Proof

We now give a brief intuition for the soundness of the protocol.

Rigidity argument for the [GMP22] self-testing protocol. Since the soundness proof uses the rigidity argument of the [GMP22] protocol as the backbone, we first give a short sketch of it. Recall that the [GMP22] protocol is a n-fold parallel of [GV19]. The main technical challenge and the bulk of [GMP22] work is to establish that the prover must treat all the parallel copies of the protocol independently, that is, to show that its (a priori uncharacterized) Hilbert space can be partitioned into n identical subspaces, one for each copy of the protocol.

Consider the last round of the [GMP22] self-testing protocol: at the start, the prover has a state $\sigma^{(\theta,\vec{v})}$, which it produced as a result of the previous rounds of the protocol. Upon receiving $\theta \in \{0,1\}$ the prover measures a binary observable Z_i (if $\theta = 0$) or X_i (if $\theta = 1$) and returns the outcome v_i', one for each copy. Let $Z(\vec{a}) \coloneqq Z_1^{a_1} \cdots Z_n^{a_n}$, similarly for $X(\vec{b})$. The main goal of the [GMP22] soundness proof is to show that when acting on the prover's (unknown) state $\sigma^{(\theta)}$ (where $\sigma^{(\theta)}$ is like $\sigma^{(\theta,\vec{v})}$, but averaged over all $\vec{v}$), the operators $\{Z(\vec{a})X(\vec{b})\}$ behave

essentially like Pauli operators. Formally, this means showing that on average over $\vec{a}, \vec{b} \in \{0,1\}^n$,

$$\mathrm{Tr}\Big[Z(\vec{a})X(\vec{b})Z(\vec{a})X(\vec{b})\sigma^{(\theta)}\Big] \approx (-1)^{\vec{a}\cdot\vec{b}}. \tag{1}$$

This is done through the following steps[8]:

(1) Defining inefficient observables $\tilde{X}_i = (-1)^{v_i} X_i$, where v_i is the i-th bit of the verifier's string $\vec{v}$ (Definition 28). This observable depends on v_i, which requires the trapdoor information to be computed efficiently. Intuitively, while X_i describes the prover's answer, $\tilde{X}_i$ describes whether that answer is accepted by the verifier. Later in the proof, we will show Equation (1) with $\tilde{X}$ instead of X. We note that the Z observable can be efficiently implemented without the trapdoor, and the verifier can also use Z to verify the answer of the prover.
(2) Extending the family of states $\{\sigma^{(\theta)}\}_{\theta\in\{0,1\}}$ to a larger family of "counterfactual states" $\{\sigma^{(\vec{\theta})}\}_{\vec{\theta}\in\{0,1\}^n}$, which are defined as the states the prover would have prepared if the verifier had sent keys $k_i \in \mathcal{K}_{\theta_i}$ for different θ_i. The key point here is that the states $\{\sigma^{(\vec{\theta})}\}_{\vec{\theta}}$ are computationally indistinguishable by the properties of ENTCF families.
(3) Showing various commutation and anti-commutation relations for the observables $Z(\vec{a})$ and $\tilde{X}(\vec{b})$ using the counterfactual states $\sigma^{(\vec{\theta})}$. For example, to show that Z_i and $\tilde{X}_j$ commute, we would choose a $\vec{\theta}$ with $\theta_i = 0$ and $\theta_j = 1$ since the verifier can check the outcomes of "Z-type observables" for $\theta = 0$ and "X-type observables" for $\theta = 1$. Then, we can relate these statements back to the prover's actual states $\sigma^{(\theta)}$ using the computational indistinguishability of $\{\sigma^{(\vec{\theta})}\}$.
(4) Combining the commutation and anti-commutation statements from the previous step to show that the observables $\{Z(\vec{a})\tilde{X}(\vec{b})\}$ behave like Pauli observables on $\sigma^{(\theta=1)}$ (Lemma 10).
(5) Explicitly defining an isometry $\tilde{V}$ which can be shown to map $\{Z(\vec{a})\tilde{X}(\vec{b})\}$ to the corresponding Pauli observables (Definition 30). Furthermore, by using this $\tilde{V}$, we can also define a modified isometry V that maps the efficient observables $\{Z(\vec{a})X(\vec{b})\}$ to the corresponding Pauli observables (Lemma 11). This proves Equation (1) and finishes the proof.

Rigidity argument for our coset-state self-testing protocol. Using the [GMP22] rigidity argument, we now turn into our coset-state self-testing protocol. Crucially, since the two protocols are identical from the prover's point of view, and the fact that the input of the prover is encrypted, Equation (1) also carries to the coset-state self-testing. Specifically, it means that under the isometry V, the prover's observables in the coset-state self-testing protocol also behave like Pauli observables (Lemma 14). Roughly speaking, the isometry "teleport" the prover's state into a "concrete" state by means of EPR pairs. In our case, the

[8] This sketch is described in [GMP22], we briefly recall it here, and refer the reader to that paper for more details.

concrete state would be (close to) a mixed state of a vector $v \in A + x$ if $\theta = 0$, or a vector $v' \in A^\perp + z$ if $\theta = 1$ (up to some classical post-processing), for a coset state instance (A, x, z) (Lemma 16).

This means that we can fix a prover $\mathcal{P}$ and consider a "hypothetical" quantum verifier, which runs the protocol in superposition with $\mathcal{P}$, that is, we do not measure to get the prover's classical message as in the original protocol, but only do a projective measurement at the end for the verification. Then under the isometry V, if $\theta = 0$, we should obtain a state that is close to $|A + x\rangle$, and if $\theta = 1$, a state that is close to $|A^\perp + z\rangle$. In other words, consider that we run $\mathcal{P}$ with $\theta = 0$ in superposition, check the obtained state is $|A + x\rangle$, then undo the prover computation (described by a unitary), then run $\mathcal{P}$ with $\theta = 1$ in superposition, check the obtained state is $|A^\perp + z\rangle$. If both checks pass, it is easy to see that the prover must have a coset state $|A_{x,z}\rangle$ in its registers.

Note that this does not constitute a classical verification of QFHE. What it says is that after the evaluation and if $\mathcal{P}$ passes verification with overwhelming probability it is necessary that it must have a coset state in its register up to an isometry.

We stress that the above rigidity statement has $1/\mathsf{poly}(n)$ closeness, due to the $1/\mathsf{poly}(n)$ closeness in the rigidity argument of the [GMP22] protocol.

Going from self-testing to remote state preparation. We then simply run the self-testing protocol sequentially in the cut-and-choose style. Say we have $2N$ coset state instances, and we run the self-testing protocol over N instances, chosen uniformly at random. The remaining N instances are the output of the protocol. By a particular "quantum sample-and-estimate" strategy defined in [BF10], it means that after running the self-testing rounds, the prover has at least one coset state $|A_{x,z}\rangle$ among N remaining coset state instances in its registers, with inverse polynomial closeness. We can write the prover's state at this step as (inverse polynomially δ-close to) $|A_{x,z}\rangle \otimes \rho$, where ρ can depend on the protocol's transcript and the encryption of (A, x, z) (Proposition 2).

Establishing a monogamy of entanglement property. In this final step, we want to show that now if the prover involves in a monogamy of entanglement game, it would have negligible probability of winning. The security game is defined as follows (Definition 2).

1. The prover and the verifier jointly execute our semi-quantum protocol to obtain (supposedly) N coset states, which are hidden but kept by the prover.
2. The prover and the verifier play the monogamy game using the output of the semi-quantum protocol:
   (a) The prover splits its state into a bipartite state and sends each part to Bob and Charlie, respectively. No communication is allowed between Bob and Charlie.
   (b) The verifier sends the description of the *subspace* to both Bob and Charlie.
   (c) Bob and Charlie are asked to output N vectors belonging to N cosets (for Bob), and N dual cosets (for Charlie).

However, our current situation is different from the standard monogamy game setting in which the prover only has the coset state, while here the prover also has an auxiliary state that depends on the coset state description. (Even worse, it might be possible that the prover can have two copies of the coset state after the interactive protocol.) The proof of the standard monogamy game does not carry over directly. Hence, for our monogamy of entanglement proof, new ideas are needed.

Injecting quantumness into the reduction. Our idea is to consider an intermediate game as follows.

1. The prover and the verifier jointly execute our semi-quantum protocol.
2. After finishing the protocol execution, the verifier asks the prover to send it a coset state among the remaining coset state instances uniformly at random.
3. Upon receiving a quantum state from the prover, the verifier verifies whether the received state is indeed the expected coset state, then it sends it back unmodified to the prover.
4. The prover and the verifier play the monogamy game.

Here we make few notes. First, with probability $\frac{1}{N}$ the coset state instance that the verifier asked is (A, x, z). It is easy to see that with probability $\frac{(1-\delta)}{N}$, which is non-negligible, any adversary for the original security experiment can be turned into an adversary for this experiment with identical winning probability. Secondly, defining this intermediate game is possible because of our rigidity argument above. Indeed, only in this step we inject quantumness into the reduction and make it a quantum verifier.

The proof continues with the following steps (which are formally described as a series of hybrids in the proof of Theorem 3).

- We make another important observation is that when considering only the coset instance (A, x, z), it is exactly the same as the public-key semi-quantum protocol introduced by Shmueli in [Shm22a]. We then follow proof strategies in previous works and carefully modify the experiment to remove the QFHE secret key (corresponding to this coset instance (A, x, z)) from the reduction. This is essentially done by changing the obfuscated membership checking programs sent to the prover in the last step of the protocol, using the following two techniques: subspace-hiding obfuscation [Zha19], and complexity leveraging to blindly sample the obfuscations [Shm22a]. To use Shmueli's complexity leveraging technique, we will need sub-exponential security of the building blocks (which include the QFHE and the indistinguishability obfuscation).
- Then we make a final change in the reduction: upon receiving the coset state from the prover and if the check passes, the verifier keeps the received coset state in its internal memory, and send back to the prover another random coset state $|A'_{x',z'}\rangle$. In the monogamy game, instead of sending a description

of A (as a basis matrix), the verifier sends a description of A'. Note that now the winning condition is also changed subject to this change in the challenge coset. We can think of $|A'_{x',z'}\rangle$ as the challenge of the original monogamy of entanglement game (with quantum communication).

- In this final experiment, if the prover managed to win the monogamy game, it means that Bob has successfully output a vector $v \in A' + x'$, and Charlie has successfully output a vector $w \in A'^{\perp} + z'$. The verifier then outputs v, w and wins the monogamy game with quantum communication. We conclude that no efficient prover can win this experiment except with negligible probability.
- The last part of the proof is to show that this final experiment is computationally indistinguishable from the previous experiment (in which the QFHE secret key was removed). We do this by invoking the security of the QFHE. However, there is a subtlety that needs to be taken care of. That is, even if we do not use the QFHE secret key in the reduction at this step, the adversary still receives predicate programs on the ciphertext, which are the obfuscated membership checking programs. Thus, we cannot simply send a uniformly random coset state $|A'_{x',z'}\rangle$ to the prover. In the protocol, we change the obfuscation programs so that both $|A_{x,z}\rangle$ and $|A'_{x',z'}\rangle$ make the programs accept. We refer to the formal construction and proof for the description of how these obfuscation programs are generated. Once this is shown, we can complete the proof.

2.3 Copy-Protection of Point Functions

Next, we give some intuition and obstacles behind our construction of copy-protection of point functions in this section. A natural idea to construct copy-protection of point functions would be as follows. In order to protect a point y, sample a pseudorandom function (PRF) secret key k, protect it using the PRF copy-protection scheme to get ρ_{k}, and let the copy-protection of y be $(\rho_{\mathsf{k}}, \mathsf{PRF}(\mathsf{k}, y))$. The evaluation of an input x then consists, given a copy-protection of a point (ρ, z), on using the PRF copy-protection's evaluation procedure to compute $\mathsf{PRF}(\mathsf{k}, x)$ and returning whether the outcome is z or not.

At first glance, it may look like this idea has good chances to result in a secure scheme, since copy-protection of PRFs with a strong security property was shown in [CLLZ21]. This so-called *indistinguishable anti-piracy security* of copy-protection of PRFs is defined as a game between a challenger and three adversaries Alice, Bob and Charlie. Bob and Charlie are not given a challenge only, but a pair (x, z) such that z is either $\mathsf{PRF}(\mathsf{k}, x)$, or a uniformly random string, depending on the challenger's random coins. Then they are asked to return the value of the coins.

Unfortunately, we do not know how to reduce the security of our protocol directly to the indistinguishable anti-piracy security of copy-protection of PRFs, and thus we need to make few modifications to the indistinguishable anti-piracy game described above in order to carry out the reduction: (i) first, we change the definition by allowing Alice to have access to the z part of each challenge

pair before she sends the bipartite state to Bob and Charlie; (ii) secondly, it is no longer the z part of the challenge that is either real or random, but the x part. More precisely, the challenge pairs (x, z) become such that z is an image of the PRF and x is either its pre-image or a uniformly random string. Furthermore, the image value z is sent to Alice and thus it is the same for both Bob and Charlie, only the x values are sampled independently based on the challenger's random coins.

Even though we conjecture that our construction is secure if the underlying copy-protection of PRF scheme has security with respect to these modifications, it turns out that we have incompatible distributions when we do the reduction. The reason is essentially that, in order to prove the security, we do a reduction to the monogamy of entanglement of coset states (Definition 19). In this game, Bob and Charlie must each return a vector from a different space, which - in the reduction - they extract from the challenge they are given. Because of our last change, they receive the same challenge with probability 1/4, and then they return the same vector with probability 1/4, which is too much to proceed with the reduction.

It turns out that, we could not find a way to circumvent this issue with this distribution, because as long as Bob and Charlie can be given the same challenge with large enough probability, the same problem occurs when we try to do the reduction. However, if we choose the distribution of [CMP20], Bob and Charlie never receive the same challenge, and thus extract vectors in different spaces with non-negligible probability. We refer the reader to Section 6 for a formal description of the construction, and we note that even though the challenge distribution that we use is less ideal, it is still a non-trivial challenge distribution in the context of copy-protection of point functions.

3 Preliminaries

Notation. Throughout this paper, λ denotes the security parameter. The notation $\mathsf{negl}(\lambda)$ denotes any function f such that $f(\lambda) = \lambda^{-\omega(1)}$, and $\mathsf{poly}(\lambda)$ denotes any function f such that $f(\lambda) = \mathcal{O}(\lambda^c)$ for some $c > 0$. For $a, b \in \mathbb{R}$, $[a, b] \coloneqq \{\, x \in \mathbb{R} \mid a \leq x \leq b \,\}$ and $[\![a, b]\!] \coloneqq \{\, x \in \mathbb{Z} \mid a \leq x \leq b \,\}$ will denote the closed real and integer interval with endpoints a and b. With an abuse of notation, we will write $[\![n]\!]$ as shorthand for $[\![0, n-1]\!]$. For a set $I = \{i_1, \ldots, i_\ell\} \subseteq [\![n]\!]$ and a n-bit string $x \in \{0,1\}^n$, we write $x|_I \coloneqq x_{i_1} \cdots x_{i_\ell}$. When sampling uniformly at random a value a from a set $\mathcal{U}$, we employ the notation $a \xleftarrow{\$} \mathcal{U}$. When sampling a value a from a probabilistic algorithm $\mathcal{A}$, we employ the notation $a \leftarrow \mathcal{A}$. Let $|\cdot|$ denote either the length of a string, or the cardinal of a finite set, or the absolute value. By PPT we mean a polynomial-time non-uniform family of probabilistic circuits, and by QPT we mean a polynomial-time family of quantum circuits. For a probabilistic algorithm f, we write $f(x; r)$ to denote the computation of f on input x with randomness r drawn uniformly at random. We sometimes omit the randomness and just write $f(x)$.

3.1 Quantum Computation

We assume familiarity with quantum information and computation, and refer to [NC11] and Appendix A.1 for the definition of basic concepts.

We use $\mathcal{H}$ to denote an arbitrary finite-dimensional Hilbert space, and use indices to differentiate between distinct spaces. The map $\mathrm{Tr} : \mathcal{L}(\mathcal{H}) \to \mathbb{C}$ denotes the trace, and $\mathrm{Tr}_B : \mathcal{L}(\mathcal{H}_A \otimes \mathcal{H}_B) \to \mathcal{L}(\mathcal{H}_A)$ is the partial trace over subsystem B. $\mathrm{Pos}(\mathcal{H})$ denotes the set of positive semidefinite operators on $\mathcal{H}$, and $\mathcal{D}(\mathcal{H}) = \{A \in \mathrm{Pos}(\mathcal{H}) \mid \mathrm{Tr}[A] = 1\}$ is the set of density matrices on $\mathcal{H}$.

The single qubit Pauli operators are $\sigma_X = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ and $\sigma_Z = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$. The Hadamard basis states are written as $|(-)^b\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b |1\rangle)$.

An observable on $\mathcal{H}$ is a Hermitian linear operator on $\mathcal{H}$. A binary observable is an observable that only has eigenvalues $\in \{-1, 1\}$. For a binary observable O and $b \in \{0, 1\}$, we denote by $O^{(b)}$ the projector onto the $(-1)^b$-eigenspace of O. For any procedure which takes a quantum state as input and produces a bit (or more generally an integer) as output, e.g., by measuring the input state, we denote the probability distribution over outputs b on input state ψ by $\Pr[\, b \,|\, \psi \,]$.

We will borrow the notation of [GMP22, MV21], and also include some technical lemmas from the preliminaries of those papers, which are used later in our proof for semi-quantum copy-protection construction in Section 4.

3.1.1 Sampling in a Quantum Population

In this paper, we also use a generic framework presented in [BF10] for analyzing cut-and-choose strategies applied to quantum states. We briefly recall the framework in Appendix A.3.

3.2 Cryptographic Primitives

We will use several cryptographic primitives in this paper: (i) indistinguishability obfuscation $i\mathcal{O}$, (ii) pseudorandom functions PRF, (iii) leveled hybrid quantum fully homomorphic encryption $\mathsf{QFHE} := \langle \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{QOTP}, \mathsf{Eval}, \mathsf{Decrypt} \rangle$. We refer to Appendix A for formal definitions of these primitives.

3.3 Extended Trapdoor Claw-free Functions

Our remote state preparation protocol is based on a cryptographic primitive called extended noisy trapdoor claw free function families (ENTCF families), which are defined in [Mah18b, Section 4] and can be constructed from the Learning with Errors assumption [Reg05, BCM+18]. We use the same notation as in [Mah18b, Section 4], with the exception that we write $\mathcal{K}_0$ instead of $\mathcal{K}_{\mathcal{G}}$ and $\mathcal{K}_1$ instead of $\mathcal{K}_{\mathcal{F}}$. In addition, we also define the following functions for convenience:

Definition 1 (Decoding maps, [MV21, Definition 2.1]).

1. *For a key $k \in \mathcal{K}_0 \cup \mathcal{K}_1$, an image $y \in \mathcal{Y}$, a bit $b \in \{0, 1\}$, and a pre-image $x \in \mathcal{X}$, we define $\mathsf{Chk}(k, y, b, x)$ to return 1 if $y \in \mathrm{Supp}(f_{k,b}(x))$, and 0 otherwise. (This definition is as in [Mah18b, Definition 4.1 and 4.2].)*

2. *For a key $k \in \mathcal{K}_0$ and a $y \in \mathcal{Y}$, we define $\hat{b}(k,y)$ by the condition $y \in \cup_x \mathrm{Supp}\Big(f_{k,\hat{b}(k,y)}(x)\Big)$. (This is well-defined because $f_{k,0}$ and $f_{k,1}$ form an injective pair.)*

3. *For a key $k \in \mathcal{K}_0 \cup \mathcal{K}_1$ and a $y \in \mathcal{Y}$, we define $\hat{x}_b(k,y)$ by the condition $y \in \mathrm{Supp}(f_{k,b}(\hat{x}_b(k,y)))$, and $\hat{x}_b(k,y) = \bot$ if $y \notin \cup_x \mathrm{Supp}(f_{k,b}(x))$. For $k \in \mathcal{K}_0$, we also use the shorthand $\hat{x}(k,y) \coloneqq \hat{x}_{\hat{b}(k,y)}(k,y)$.*

4. *For a key $k \in \mathcal{K}_1$, a $y \in \mathcal{Y}$, and a $d \in \{0,1\}^w$, we define $\hat{u}(k,y,d)$ by the condition $d \cdot (\hat{x}_0(k,y) \oplus \hat{x}_1(k,y)) = \hat{u}(k,y,d)$.*

The above decoding maps applied to vector inputs are understood to act in an element-wise fashion. For example, for $\vec{k} \in \mathcal{K}_1^{\times n}$, $\vec{y} \in \mathcal{Y}^{\times n}$, and $\vec{d} \in \{0,1\}^{w \times n}$, we denote by $\hat{u}(\vec{k},\vec{y},\vec{d}) \in \{0,1\}^n$ the string defined by $\Big(\hat{u}(\vec{k},\vec{y},\vec{d})\Big)_i \coloneqq \hat{u}(k_i,y_i,d_i)$.

4 Semi-Quantum Copy-Protection

In this section, we introduce our protocol for remote hidden coset state preparation in Section 4.1, followed by proof of correctness in Section 4.2 and proof of soundness in Section 5.

4.1 Construction

Notation. Our Protocol 1 and Protocol 2 will be (almost) a parallel repetition of a sub-protocol. We make use of vector notation to denote tuples of items corresponding to the different copies of the sub-protocol. For example, if each of the n parallel sub-protocols requires a key k_i, we denote $\vec{k} = (k_1, \ldots, k_n)$. A function that takes as input a single value can be extended to input vectors in the obvious way: for example, if f takes as input a single key k, then we write $f(\vec{k})$ for the vector $(f(k_1), \ldots, f(k_n))$. We will also use $\vec{0}$ and $\vec{1}$ for the bit strings consisting only of 0 and 1, respectively (and whose length will be clear from the context), and $\vec{1}^i \in \{0,1\}^n$ for the bit string whose i-th bit is 1 and whose remaining bits are 0. Let n the length of a vector in a coset state (i.e., if $v \in A$ then $|v| = n$). In our constructions below, we set $n \coloneqq 2\lambda$.

Ingredients. Our constructions use the following building blocks:

- A quantum hybrid fully homomorphic encryption scheme $\mathsf{QFHE} \coloneqq \langle \mathsf{KeyGen}, \mathsf{QOTP}, \mathsf{Encrypt}, \mathsf{Eval}, \mathsf{Decrypt} \rangle$, with sub-exponential advantage security.
- A post-quantum secure indistinguishability obfuscation scheme $i\mathcal{O}$.
- A post-quantum secure extended noisy trapdoor claw-free function (ENTCF) family $(\mathcal{F}, \mathcal{G})$.

Our main protocol's construction is given in Protocol 5. The protocol involves two parties: a QPT prover (or receiver, denoted as $\mathcal{P}$), and a PPT verifier (or sender, denoted as $\mathcal{V}$).

Protocol 1: Semi-Quantum Copy-Protection: BB84 Test Round

Input. The verifier initially receives Pauli keys (α, β) with $\alpha, \beta \in \{0,1\}^n$ as private inputs.

1. The verifier selects a uniformly random basis $\theta \xleftarrow{\$} \{0,1\}$, where 0 corresponds to the computational and 1 to the Hadamard basis.
2. The verifier samples keys and trapdoors $\{(k_i, t_i)\}_{i=1}^n$ by computing $(k_i, t_i) \leftarrow \mathsf{Gen}_{\mathcal{K}_\theta}(1^\lambda)$. The verifier then sends $\{k_i\}_{i=1}^n$ to the prover (but keeps the trapdoors $\{t_i\}_{i=1}^n$ private).
3. The verifier receives $\{y_i\}_{i=1}^n$ where $y_i \in \mathcal{Y}$ from the prover.
4. The verifier selects a round type $\in \{\text{pre-image round}, \text{Hadamard round}\}$ uniformly at random and sends the round type to the prover.
   (a) For a *pre-image round*: the verifier receives $\{(b_i, x_i)\}_{i=1}^n$ from the prover, with $b_i \in \{0,1\}$, and $x_i \in \mathcal{X}$. The verifier sets $\mathtt{flag}_{\mathtt{bb84}} \leftarrow \mathtt{flag}_{\mathtt{Pre}}$ and aborts if $\mathsf{Chk}(k_i, t_i, b_i, x_i) = 0$ for any $i \in [\![1, n]\!]$.
   (b) For a *Hadamard round*: the verifier receives $\{d_i\}_{i=1}^n$ from the prover with $d_i \in \{0,1\}^w$ (for some w depends on the security parameter λ). The verifier sends $q = \theta$ to the prover, and receives answers $\{v_i\}_{i=1}^n$ with $v_i \in \{0,1\}$. The verifier performs the following:
      - If $q = \theta = 0$, set $\mathtt{flag}_{\mathtt{bb84}} \leftarrow \mathtt{flag}_{\mathtt{Had}}$ and abort if $\hat{b}(k_i, y_i) \neq v_i$ for some $i \in [\![1, n]\!]$.
      - If $q = \theta = 1$, set $\mathtt{flag}_{\mathtt{bb84}} \leftarrow \mathtt{flag}_{\mathtt{Had}}$ and abort if $\hat{u}(k_i, y_i, d_i) \neq v_i \oplus \beta_i$ for some $i \in [\![1, n]\!]$.

Protocol 2: Semi-Quantum Copy-Protection: Coset-state Test Round

Input. The verifier initially receives a subspace $A \subseteq \mathbb{F}_2^n$ and Pauli keys (α, β) with $\alpha, \beta \in \{0,1\}^n$ as private inputs.

1. The verifier selects a uniformly random basis $\theta \xleftarrow{\$} \{0,1\}$, where 0 corresponds to the computational and 1 to the Hadamard basis.
2. The verifier samples keys and trapdoors $\{(k_i, t_i)\}_{i=1}^n$ by computing $(k_i, t_i) \leftarrow \mathsf{Gen}_{\mathcal{K}_\theta}(1^\lambda)$. The verifier then sends $\{k_i\}_{i=1}^n$ to the prover (but keeps the trapdoors $\{t_i\}_{i=1}^n$ private).
3. The verifier receives $\{y_i\}_{i=1}^n$ where $y_i \in \mathcal{Y}$ from the prover.
4. The verifier sends "Hadamard round" as the round type to the prover.
5. The verifier receives $\{d_i\}_{i=1}^n$ from the prover with $d_i \in \{0,1\}^w$ (for some w depends on the security parameter λ). The verifier sends $q = \theta$ to the prover, and receives answers $\{v_i\}_{i=1}^n$ with $v_i \in \{0,1\}$.
   The verifier performs the following:
   - If $q = \theta = 0$, let $\vec{v} \coloneqq v_1 \dots v_n$. Set $\mathtt{flag}_{\mathtt{coset}} \leftarrow \mathtt{flag}_{\mathtt{Had}}$ and abort if $\vec{v} \notin A + \alpha$.

- If $q = \theta = 1$, let $s_i \leftarrow v_i \oplus \hat{u}(k_i, y_i, d_i)$ and let $s := s_1 \dots s_n$. Set $\texttt{flag}_{\texttt{coset}} \leftarrow \texttt{flag}_{\texttt{Had}}$ and abort if $\vec{s} \notin A^{\perp} + \beta$.

Protocol 3: Semi-Quantum Copy-Protection: Self-Testing

Let M^2 the maximum number of test rounds (for $M \in \mathbb{N}$).
Input. The verifier initially receives a subspace $A \subseteq \mathbb{F}_2^n$ and Pauli keys (α', β') and $\{(\alpha_i, \beta_i)\}_{i=1}^{M^2}$ with $\alpha', \beta', \alpha_i, \beta_i \in \{0,1\}^n$ as private inputs. Note that (A, α', β') corresponds to one coset-state instance, and $\{(\alpha_i, \beta_i)\}_{i=1}^{M^2}$ corresponds to M^2 BB84 instances.

1. The verifier privately samples $B \xleftarrow{\$} [\![1, M-1]\!]$ (this determines the number of BB84 test rounds that will be performed).
2. The verifier performs BM executions of Protocol 1 (with corresponding private inputs $\{(\alpha_i, \beta_i)\}$) with the prover. The verifier aborts if Protocol 1 aborts for some execution.
3. The verifier privately samples $R \xleftarrow{\$} [\![1, M]\!]$ and executes Protocol 1 with the prover $R-1$ times (with corresponding private inputs $\{(\alpha_i, \beta_i)\}$). Then the verifier executes Protocol 2 with the prover (with private inputs (A, α', β')) and aborts if Protocol 2 aborts.

Protocol 4: Semi-Quantum Copy-Protection: Self-Testing (with Soundness Amplification)

Let $N := \lambda$ the number of iterations.
Input. The verifier initially receives $\{(A_i, \alpha_i', \beta_i')\}_{i=1}^{N}$ and $\{(\alpha_i, \beta_i)\}_{i=1}^{NM^2}$ as private inputs. Each tuple in the first set corresponds to a coset-state instance, and each tuple in the second set corresponds to a BB84 instance.
The verifier and the prover sequentially run Protocol 3 N times as follows.

1. For each run, the verifier and the prover interactively run Protocol 3 with one coset state instance $(A_i, \alpha_i', \beta_i')$ and M^2 BB84 instances $\{(\alpha_i, \beta_i)\}_{i=1}^{M^2}$, each is picked uniformly at random from the input sets. (If some instance has been picked before, it will be excluded).
2. The verifier aborts unless Protocol 3 does not abort in all N iterations.

Protocol 5: Semi-Quantum Copy-Protection: Main Protocol

Verifier's preparation.

1. **Coset-state instances.** For each $i \in [\![1, 2N]\!]$, the verifier samples a random $\frac{n}{2}$-dimensional subspace $S_i \subseteq \mathbb{F}_2^n$, described by a matrix $\mathbf{M}_{S_i} \in \{0,1\}^{\frac{n}{2} \times n}$. Samples Pauli keys $p_{\alpha_i} \xleftarrow{\$} \{0,1\}^{\frac{n^2}{2}}$ to encrypt $\mathbf{M}_{S_i}^{p_{\alpha_i}} \leftarrow \mathsf{QFHE.QOTP}(p_{\alpha_i}, \mathbf{M}_{S_i})$, and then $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{QFHE.KeyGen}(1^\lambda, 1^{\ell(\lambda)})$ for some polynomial $\ell(\cdot)$, $\mathsf{ct}_i \leftarrow \mathsf{QFHE.Encrypt}(\mathsf{pk}_i, p_{\alpha_i})$.
2. **n-qubit BB84 instances.** For each $i \in [\![1, NM^2]\!]$, the verifier samples Pauli keys $p_{\alpha_i} \xleftarrow{\$} \{0,1\}^{\frac{n^2}{2}}$ to encrypt $\mathbf{M}_0^{p_{\alpha_i}} \leftarrow \mathsf{QFHE.QOTP}(p_{\alpha_i}, \mathbf{M}_0)$

(here, $\mathbf{M}_0$ is the all-zero vector of length $\frac{n^2}{2}$), and then $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{QFHE.KeyGen}(1^\lambda, 1^{\ell(\lambda)})$, $\mathsf{ct}_i \leftarrow \mathsf{QFHE.Encrypt}(\mathsf{pk}_i, p_{\alpha_i})$.

3. For each index $i \in [\![1, 2N + NM^2]\!]$, the verifier picks uniformly at random one instance from either the set of (encrypted) coset states or the set of (encrypted) n-qubit BB84 states prepared above. For each index i, denote the i-th instance as $(\mathsf{pk}_i, \mathbf{M}^{p_{\alpha_i}}, \mathsf{ct}_i)$ with secrets (sk_i, S_i). (If this instance is from the set of n-qubit BB84 states, we understand that $S_i = \mathbf{M}_0$.)

4. The verifier sends $\{\mathsf{pk}_i, \mathbf{M}^{p_{\alpha_i}}, \mathsf{ct}_i\}_{i=1}^{2N+NM^2}$ to the prover.

Prover's homomorphic evaluation

5. Let C the quantum circuit that for an input matrix $\mathbf{M} \in \{0,1\}^{\frac{n}{2} \times n}$, outputs a uniform superposition of its row span, except that if $\mathbf{M} = \mathbf{M}_0$, it outputs a uniform superposition of all vectors in the space $\mathbb{F}_2^n$. The prover homomorphically evaluates C for each $i \in [\![1, 2N + NM^2]\!]$: $(|S_{i,\alpha_i,\beta_i}\rangle, \mathsf{ct}_{i,\alpha_i,\beta_i}) \leftarrow \mathsf{QFHE.Eval}\big(\mathsf{pk}_i, (\mathbf{M}^{p_{\alpha_i}}, \mathsf{ct}_i), C\big)$, saves the quantum part $|S_{i,\alpha_i,\beta_i}\rangle$ and sends the classical part $\mathsf{ct}_{i,\alpha_i,\beta_i}$ to the verifier.

Self-testing for the prover.

6. For each $i \in [\![1, 2N + NM^2]\!]$, the verifier decrypts $(\alpha_i, \beta_i) \leftarrow \mathsf{QFHE.Decrypt}(\mathsf{sk}_i, \mathsf{ct}_{i,\alpha_i,\beta_i})$. For all coset-state instances, if $\alpha_i \in S_i$, the protocol is terminated.

7. The verifier then runs Protocol 4 with these NM^2 prepared BB84 instances and N coset-state instances, where each coset-state instance is picked uniformly at random among $2N$ prepared instances. (If some instance has been picked before, it will be excluded). It aborts if Protocol 4 aborts.

Coset-state generation.

8. The verifier samples a random $\frac{n}{2}$-dimensional coset $(\hat{S}, \hat{\alpha}, \hat{\beta}) \subseteq \mathbb{F}_2^n$ independently.[a] Let $\mathbf{M}_{\hat{S}}, \mathbf{M}_{\hat{S}^\perp} \in \{0,1\}^{\frac{n}{2} \times n}$ bases for $\hat{S}$ and $\hat{S}^\perp$, respectively.

9. Let T the set of indexes of the remaining N instances of the coset-states which have not been used in the self-testing protocol above. For each $i \in T$, the verifier does the following:
   (a) Let $\mathbf{M}_{S_i^\perp} \in \{0,1\}^{\frac{n}{2} \times n}$ a basis for $S_i^\perp$ (as a matrix). Compute indistinguishability obfuscations $P_{0,i} \leftarrow i\mathcal{O}\left(i\mathcal{O}(\mathbf{M}_{S_i} + \alpha_i) \vee i\mathcal{O}(\mathbf{M}_{\hat{S}} + \hat{\alpha})\right)$ and $P_{1,i} \leftarrow i\mathcal{O}\left(i\mathcal{O}(\mathbf{M}_{S_i^\perp} + \beta_i) \vee i\mathcal{O}(\mathbf{M}_{\hat{S}^\perp} + \hat{\beta})\right)$, all with appropriate padding.[b]
   (b) Record $\{(\alpha_i, \beta_i, S_i)\}_{i \in T}$.
   (c) Send T and $\{P_{0,i}, P_{1,i}\}_{i \in T}$ to the prover.

The output of the prover is $\{P_{0,i}, P_{1,i}, |S_{i,\alpha_i,\beta_i}\rangle\}_{i \in T}$ where $|T| = N$.

[a] This step is merely an artifact that we will need later for the security proof.

[b] Here, we understand that for any two programs C, C' with binary output, $i\mathcal{O}(C \vee C')(x)$ outputs $C(x) \vee C'(x)$.

Notation. For each execution of Protocol 5, we abuse the notation and denote $(|A_{s,s'}\rangle, \mathsf{R}^0, \mathsf{R}^1)$ the state obtained by the receiver, where R^b the obfuscated membership checking programs, computed by concatenating all the obfuscated programs $P_{b,i}$ in Protocol 5, and (A, s, s') the "coset" (which in fact consists of polynomial many different real cosets) obtained by the sender. That is, we consider the whole output state of the protocol as a single unclonable state (which we also call "coset state"). This notation will only be used later when we describe the applications of our protocol in the context of semi-quantum copy-protection.[9]

4.2 Proof of Correctness

Proposition 1. *There exists a* QPT *prover that is accepted in Protocol 5 with probability negligibly close to* 1 *in the security parameter* λ*. Furthermore, the final* quantum *state of such a prover at the end of Protocol 5 is (negligibly close to) a product of* N *hidden coset states:*

$$\bigotimes_{i\in T} |S_{i,\alpha_i,\beta_i}\rangle\,, \tag{2}$$

where $\{(S_i, \alpha_i, \beta_i)\}_{i\in T}$ *are recorded by the verifier at the end of Protocol 5.*

Proof. The proof of correctness includes three steps: (1) If the prover ran honestly then its output after the homomorphic evaluation step has negligible trace distance to (QOTP encrypted of) BB84 states and coset states; (2) The self-test protocol passes (that is, the protocol does not terminate at this step) with probability negligibly close to 1; (3) In the last step of coset-state generation, after discarding all BB84 states, the output of the prover at the end of Protocol 5 has negligible trace distance to the state described in Equation (2). We give a full proof in Appendix B.1. □

5 Proof of Soundness

In this section, we prove soundness of Protocol 5, following the steps outlined in Section 2.2:

1. First, we show a ridigity argument (with inverse polynomial soundness) for our self-testing protocol (Protocol 3) in Section 5.1.
2. We show that any malicious prover in our remote state preparation protocol must have also constructed a hidden random coset state up to some inverse polynomial error. This final step reduces to a particular "quantum sample-and-estimate strategy", which is a quantum counterpart of the classical "cut-and-choose" as defined by Bouman and Fehr [BF10].

[9] We thank an anonymous reviewer for pointing out this approach, so that our protocol is indeed applicable to copy-protection in a plug-and-play manner.

3. We then show the soundness of our final protocol (Protocol 5), which is stated as a monogamy of entanglement game, in Section 5.2. Notably, even if our rigidity statement achieves only inverse polynomial soundness, we show that our protocol achieves negligible security in this monogamy game.

Informally, a prover that succeeds in Protocol 5 has negligible probability of winning a monogamy of entanglement game for coset states, which is formally stated as Theorem 3. This means that if we consider the output of our final protocol as a single unclonable state, the situation at the end of Protocol 5 is essentially identical to one in which the verifier has sent a hidden coset state to the prover via a quantum channel, whose security is based on the monogamy of entanglement of coset states defined in Definition 19.

5.1 Self-Testing Protocol Soundness

In order to prove Proposition 2, we first show a rigidity argument for our self-testing protocol (Protocol 3). We state in this section the following proposition.

Proposition 2. *For any $\lambda \in \mathbb{N}$, there exist choices $M = \mathsf{poly}(\lambda)$ and $\delta = 1/\mathsf{poly}(\lambda)$ such that if the verifier executes Protocol 5 with an efficient quantum prover whose success probability is lower-bounded by an inverse polynomial, the following holds. We denote by ϕ^T_{SVP} the verifier and prover's joint final state at the end of Protocol 5, where T is the set of coset states obtained by the verifier, S is set to $|\bot\rangle\langle\bot|$ by the verifier if the protocol aborts and $|\top\rangle\langle\top|$ otherwise, V is the register in which the verifier records the set T, and P is the prover's registers. Then, denoting the probability of success as $\Pr[\top]$, and writing*

$$\phi^T_{SVP} = \Pr[\top]|\top\rangle\langle\top|_S \otimes \phi^T_{VP|\top} + (1-\Pr[\top])|\bot\rangle\langle\bot| \otimes \phi^T_{VP|\bot}.$$

Then there exists a coset instance (A, α, β) in T such that the state $\phi^T_{VP|\top}$ conditioned on acceptance satisfies:

$$\phi^T_{VP|\top} \overset{c}{\approx}_{1/\mathsf{poly}(\lambda)} |T\rangle\langle T|_V \otimes |A_{\alpha,\beta}\rangle\langle A_{\alpha,\beta}| \otimes \rho, \tag{3}$$

for some auxiliary state ρ.

Due to the space limitations, we defer the proof of this proposition to Appendix B.2.

5.2 Soundness of Protocol 5

We now formally define the notion of soundness for our protocol, which is described as a coset monogamy game similar to Definition 19.

Definition 2 (Soundness). *For any QPT prover $\mathcal{P} = \{\mathcal{P}_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$ interacting with a PPT verifier $\mathcal{V}$ in Protocol 5, after which $\mathcal{V}$ outputs $\{S_i, \alpha_i, \beta_i\}_{i\in T}$ and $\mathcal{P}$ outputs a state ψ, let $(\{S_i, \alpha_i, \beta_i\}_{i\in T}, \psi) \leftarrow \langle \mathcal{P}_\lambda(\rho_\lambda), \mathcal{V}(1^\lambda)\rangle$ denote this interaction. The prover (now modeled as a triple algorithm $(\mathcal{P}, \mathcal{B}, \mathcal{C})$) then interacts with the verifier in the following monogamy game.*

1. *The prover applies a CPTP map to split ψ into a bipartite state ψ_{BC}; it sends the register B to $\mathcal{B}$ and the register C to $\mathcal{C}$. No communication is allowed between $\mathcal{B}$ and $\mathcal{C}$ after this phase.*
2. *Question. The verifier sends the description of $\{S_i\}_{i\in T}$, to both $\mathcal{B}$ and $\mathcal{C}$.*
3. *Answer. $\mathcal{B}$ returns $s_1^{(i)} \in \mathbb{F}_2^n$ and $\mathcal{C}$ returns $s_2^{(i)} \in \mathbb{F}_2^n$ for all $i \in T$.*

The prover $(\mathcal{P}, \mathcal{B}, \mathcal{C})$ wins if and only if $s_1^{(i)} \in S_i + \alpha_i$ and $s_2^{(i)} \in S_i^{\perp} + \beta_i$ for all $i \in T$. Let $\mathsf{SMCosetMonogamy}(\mathcal{P}, \lambda)$ be a random variable which takes the value 1 if the game above is won by the prover $(\mathcal{P}, \mathcal{B}, \mathcal{C})$, and takes the value 0 otherwise.

The protocol is secure if the winning probability of any QPT *adversary is negligible. Formally, for any* QPT *malicious prover, the protocol is* computationally sound *if we have*

$$\Pr[\mathsf{SMCosetMonogamy}(\mathcal{P}, \lambda) = 1] \leq \mathsf{negl}(\lambda).$$

Theorem 3. *Protocol 5 is computationally sound, according to Definition 2.*

Proof. Let $\mathcal{P} = \{\mathcal{P}_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$ a quantum polynomial time adversary that succeeds in the game SMCosetMonogamy with some non-negligible probability $\varepsilon = \{\varepsilon_\lambda\}_{\lambda\in\mathbb{N}}$. Let $(\{S_i, \alpha_i, \beta_i\}_{i\in T}, \psi) \leftarrow \langle \mathcal{P}_\lambda(\rho_\lambda), \mathcal{V}(1^\lambda)\rangle$. This means that $\mathcal{P} = (\mathcal{P}, \mathcal{B}, \mathcal{C})$ is able to output a pair $(s_1^{(i)}, s_2^{(i)}) \in (S_i + \alpha_i) \times (S_i^{\perp} + \beta_i)$ for all $i \in T$ in the monogamy game defined in Definition 2.

Let $\delta' \in (0, 1]$ the sub-exponential security level of the QFHE (that is, any QPT adversary cannot break the semantic security of the QFHE with advantage bigger than $2^{\lambda^{\delta'}}$), and denote $\delta := \frac{\delta'}{2}$.

We next describe a sequence of hybrid experiments.[10]

Game G_0: This is the original experiment.

We define G_0 as the original attack, where $\mathcal{P}$ interacts with the verifier in Protocol 5 and wins the monogamy game SMCosetMonogamy. We say G_0 is successful if $\mathsf{SMCosetMonogamy}(\mathcal{P}, \lambda) = 1$. The experiment G_0 is thus successful with probability ε.

Game G_1: Changing the success definition of the experiment.

Pick a random index $i \in T$, for shorthand, denote this coset instance as (S, α, β), and the adversary's corresponding output in the monogamy game is (s_1, s_2). In the current hybrid, the experiment is defined to be successful if $s_1 \in S + \alpha$ and $s_2 \in S^{\perp} + \beta$. In particular, in the current hybrid, we only consider the monogamy game for a random instance among $|T|$ coset instances. (The other instances are not considered). Apparently, G_1 is successful with probability at least ε. From now on, we only consider this coset instance in later hybrids, and all the changes are only applied to this instance.

[10] Some hybrids follow from the proof strategy given in [Shm22a].

Game G_2: Injecting quantum communication into the interaction between the prover and the verifier.

This hybrid is identical to G_1 except that now we consider the verifier as a QPT algorithm instead of a PPT algorithm, and we make an additional round of interaction using quantum communication in the protocol. (Think about the verifier now as a QPT challenger of the experiment.) In particular, right after the last step of Protocol 5 (step 9c), we ask the prover to send the coset state $|S_{\alpha,\beta}\rangle$ to the verifier. Denote this state as $|\$\rangle$. The verifier then does the following:

- Verify the received coset state:
  (a) Checks that the output qubit of the computation $i\mathcal{O}(S+\alpha)(|\$\rangle)$[11] is 1.
  (b) Execute Hadamard transform $\mathsf{H}^{\otimes\lambda}$ on $|\$\rangle$ to obtain $|\$'\rangle$ and then check the output qubit of the computation $i\mathcal{O}(S^{\perp}+\beta)(|\$'\rangle)$ is 1.
- If any of these checks returns 0, abort and declare the game as a failure.
- Execute $\mathsf{H}^{\otimes\lambda}$ again on $|\$'\rangle$ to obtain $|\$''\rangle$ and send $|\$''\rangle$ back to the prover.

From Proposition 2, it follows that with probability at least $1/|T|$, the adversary's output state ϕ is inverse polynomially ϵ-close to $|S_{\alpha,\beta}\rangle\otimes\rho$ for some auxiliary state ρ. It means that when it is asked, the adversary can always send a state $|\$\rangle$ that is inverse polynomially ϵ-close to $|S_{\alpha,\beta}\rangle$ to the challenger.

Note that the quantum verification described above executes only on the register containing $|\$\rangle$ and thus commutes with any other quantum operation on a register entangled with it at the point where $\mathcal{P}$ finishes executing the real protocol Protocol 5. Thus after finishing the above additional interaction, the adversary's state is unchanged, if the verification passed.

The probability that the adversary does not fail in the experiment is $1-\epsilon$. It is then clear that, for any adversary that wins the G_1 with probability ε, it wins G_2 with probability at least $\varepsilon' := \varepsilon(1-\epsilon)/|T|$. Thus, the success probability of G_2 is ε' for some non-negligible ε'.

Game G_3: Removing subspace information from obfuscated circuits.

This hybrid is identical to G_2, with the only difference is that when the verifier returns the obfuscations P_0, P_1 in the last step of Protocol 5 (Step 9c), the obfuscations are changed: We sample two random $(\lambda-\lambda^{\delta})$-dimensional subspaces $T_0, T_1 \subseteq \mathbb{F}_2^{\lambda}$ subjected to $T_1^{\perp} \subseteq S \subseteq T_0$. The verifier uses $i\mathcal{O}(T_0+\alpha)$ instead of $i\mathcal{O}(S+\alpha)$, and $i\mathcal{O}(T_1+\beta)$ instead of $i\mathcal{O}(S^{\perp}+\beta)$.

It is easy to see that any QPT distinguisher between G_2 and G_3 can be transformed into a QPT distinguisher between obfuscations of the original functions $S+\alpha, S^{\perp}+\beta$ and obfuscations of $T_0+\alpha, T_1+\beta$. By the subspace hiding property of indistinguishability obfuscators (Lemma 6), the success probabilities of G_2 and G_3 are thus negligibly close. Thus the successful probability of G_3 is at least $\varepsilon' - \mathsf{negl}(\lambda)$.

Game G_4: Computing the obfuscations with less information on α, β.

This hybrid is identical to G_3, with a modification in the way we check membership in each of the cosets: Let B_0 a basis for T_0, and B_1 a basis for $T_1^{\perp}$,

[11] We are running a classical function on a quantum input, which can be interpreted as running a classical function in superposition.

and let $y_\alpha, y_\beta \in \{0,1\}^{\lambda-\lambda^\delta}$ defined as $y_\alpha := B_0 \cdot \alpha$ and $y_\beta := B_1 \cdot \beta$. $i\mathcal{O}(T_0 + \alpha)$ is changed to be an obfuscation of a circuit that for an input $u \in \{0,1\}^\lambda$ checks whether $B_0 \cdot u = y_\alpha$. $i\mathcal{O}(T_1 + \beta)$ is changed to be an obfuscation of a circuit that for an input $u \in \{0,1\}^\lambda$ checks whether $B_1 \cdot u = y_\beta$.

One can verify that the functionality of the obfuscated circuits $i\mathcal{O}(T_0 + \alpha)$, $i\mathcal{O}(T_1 + \beta)$ did not change, and thus by the security of the indistinguishability obfuscation schemes, the distributions are indistinguishable and the success probability of G_4 is $\varepsilon' - \mathsf{negl}(\lambda)$.

Game G_5: Reordering the sampling process of the subspaces S, T_0, T_1.

This hybrid is identical to G_4, except that we change the order of the subspaces sampling process. In the previous hybrid, we sample a random $\frac{\lambda}{2}$-dimensional subspace $S \subseteq \mathbb{F}_2^\lambda$ then two random $(\lambda-\lambda^\delta)$-dimensional subspaces T_0, T_1 subjected to $T_1^\perp \subseteq S \subseteq T_0$. In the current hybrid, we first sample two random $(\lambda - \lambda^\delta)$-dimensional subspaces $T_0, T_1 \subseteq \mathbb{F}_2^n$ subjected to $T_1^\perp \subseteq T_0$, then sample a random $\frac{\lambda}{2}$-dimensional subspace $S \subseteq \mathbb{F}_2^n$ subjected to $T_1^\perp \subseteq S \subseteq T_0$.

Since the distribution of (S, T_0, T_1) in both hybrids are identical, the success probability of G_5 is $\varepsilon' - \mathsf{negl}(\lambda)$.

Game G_6: Fixing the subspace T_0, T_1.

In the subspace sampling process described in the previous hybrid, T_0 and T_1 are sampled before everything else. Thus we can perform an averaging argument on the sampling of T_0, T_1 to take the samples that maximize the success probability of the previous hybrid. Fix these samples of T_0, T_1 and define G_6 with respect to these samples. It is clear that the success probability of G_6 is $\varepsilon' - \mathsf{negl}(\lambda)$.

Game G_7: Removing the QFHE secret key from the reduction.

This hybrid is identical to G_6 with one change: In step 6, when the verifier decrypts the QFHE classical part to get the Pauli keys α, β, the current hybrid does not decrypt to get α, β and instead it samples uniformly random $\alpha', \beta' \in \{0,1\}^\lambda$ and computes $y'_\alpha := B_0 \cdot \alpha', y'_\beta := B_1 \cdot \beta'$. The verifier then use these strings as y_α, y_β in the construction of the obfuscations $i\mathcal{O}(T_0 + \alpha)$, $i\mathcal{O}(T_1 + \beta)$, respectively.

We note that this change is only done for the specific coset instance under the consideration, for the other instances, the verifier still decrypts normally using the corresponding QFHE secret key.

Since α', β' are chosen uniformly at random, for fixed bases B_0, B_1, y'_α, y'_β are also uniformly random. Observe that conditioned on the probabilistic event $y'_\alpha = y_\alpha$ and $y'_\beta = y_\beta$ (for which to happen, the probability is exactly $2^{-2\lambda^\delta}$), the current and previous hybrids distribute identically. It follows that the success probability in G_7 is at least $2^{-2\lambda^\delta} \cdot (\varepsilon' - \mathsf{negl}(\lambda)) > 2^{-3\lambda^\delta}$.

Game G_8: Clearing all given knowledge on S and reducing to the original monogamy of entanglement game defined in Definition 19.

This hybrid is identical to G_7, except that we make two additional changes as follows.

– In the additional quantum communication round that we added after the end of Protocol 5 (see hybrid G_2), instead of sending back the original state $|\$\rangle$,

the verifier send $|\hat{S}_{\hat{\alpha},\hat{\beta}}\rangle$. Recall that the coset $(\hat{S},\hat{\alpha},\hat{\beta})$ is the one the verifier sampled independently in step 8.

- In the step 2 in the monogamy game (Definition 2), when the challenger (i.e., the verifier) sends the description of the subspace S to both adversaries $\mathcal{B},\mathcal{C}$, it sends $\hat{S}$ instead.
- Consequently, the winning condition is changed to be that $\mathcal{B}$ outputs a vector in $\hat{S}+\hat{\alpha}$ and $\mathcal{C}$ outputs a vector in $\hat{S}^{\perp}+\hat{\beta}$.

We make few observations on the distribution in the current hybrid. First, in order to execute G_8, there is no need to know the secret key (corresponding to the coset instance under the consideration) of the QFHE scheme. However, one needs to care when invoking the semantic security of the QFHE, because even there is no need for the secret key, the adversary is still given a "predicate" check on the ciphertext, that is the obfuscation. Thus, to use the security of the QFHE, it is necessary to use two plaintexts such that the obfuscation evaluation on the ciphertext of these two plaintexts are identical. Our obfuscations $(P_{0,i},P_{1,i})$ were generated so that this condition is satisfied.

Secondly, the obfuscation distribution does not change from the description above, and we can see that in the previous hybrid, the adversary obtains a quantum one-time pad encryption of $|S\rangle$, while in the current hybrid, the adversary obtains a quantum one-time pad of $|\hat{S}\rangle$. More precisely, the adversary in the current hybrid receives an encryption of $|\hat{S}\rangle$ that is $|\hat{S}_{\hat{\alpha},\hat{\beta}}\rangle$ and an encryption of some Pauli keys (α,β) that are different from $(\hat{\alpha},\hat{\beta})$ with overwhelming probability. But because of the semantic security of $\mathsf{QFHE.Encrypt}$ (see Definition 16), this is indistinguishable from having $|\hat{S}_{\hat{\alpha},\hat{\beta}}\rangle$ and an actual encryption of $(\hat{\alpha},\hat{\beta})$.

From these observations, it follows that we can invoke the security of the QFHE to argue the indistinguishability of the current and previous hybrids, and in particular the indistinguishability between their success probabilities. Using the sub-exponential-advantage security of the QFHE, we have the success probability of G_8 is $> 2^{-3\lambda^{\delta}} - 2^{-2\lambda^{\delta'}} > 2^{-3\lambda^{\delta}-1}$.

At this point of the proof, we can reduce the success probability of an adversary in G_8 to the monogamy of entanglement game defined in Definition 19. We note that the coset game in Definition 19 can achieve sub-exponentially negligible security, say $2^{-4\lambda^{\delta}}$, if we assume sub-exponential security of the building blocks (i.e., the indistinguishability obfuscation scheme). Now, any QPT adversary of G_8 can be used to construct a QPT adversary for the coset game defined in Definition 19 as follows. Specifically, the reduction receives a challenge coset state $|\hat{S}_{\hat{\alpha},\hat{\beta}}\rangle$ and the obfuscated membership checking programs $i\mathcal{O}(\hat{S}+\hat{\alpha}), i\mathcal{O}(\hat{S}^{\perp}+\hat{\beta})$ from its challenger in the coset game in Definition 19. The reduction runs Protocol 5 with the adversary. Note that the reduction (playing the role of the verifier in Protocol 5) only needs $i\mathcal{O}(\hat{S}+\hat{\alpha})$ and $i\mathcal{O}(\hat{S}^{\perp}+\hat{\beta})$ to perfectly simulate the protocol with the adversary. Furthermore, it uses $|\hat{S}_{\hat{\alpha},\hat{\beta}}\rangle$ in the experiment described above instead of generating the state on its own, when it needs to send a coset state back to the adversary. When the reduction receives $\hat{S}$ from its challenger, it sends $\hat{S}$ to $\mathcal{B},\mathcal{C}$, and finally the reduction outputs

whatever $\mathcal{B}$ and $\mathcal{C}$ output. (Formally, the reduction now consists of two non-communicating reductions, each interacts with $\mathcal{B}$ and $\mathcal{C}$ respectively.) This is exactly in contradiction to strong monogamy of entanglement security as we presented above. □

6 (Semi-Quantum) Copy-Protection of Point Functions

In this section, we give a construction of quantum copy-protection of point functions and we also show how to instantiate a semi-quantum copy-protection scheme by applying our remote state preparation protocol Protocol 5 to this quantum copy-protection scheme. We note that our semi-quantum copy-protection scheme is interactive, while its quantum version is non-interactive.

We first recall security definition of (semi-)quantum copy-protection of point functions in Section 6.1, and present our constructions in Section 6.2, followed by a sketch of security proofs. (The formal proofs are given in Appendix D.)

6.1 Definition

Recall that a point functions family $\{\mathsf{PF}_y\}_{y\in\mathcal{X}}$ is indexed by points $y \in \mathcal{X}$ and a point function PF_y returns 1 on input y and 0 on any other input.

We give a security definition for copy-protection of point functions by instantiating the general definition of copy-protection (see Appendix A.9) with the following family: $\mathcal{F} \coloneqq \{\mathsf{PF}_y\}_{y\in\{0,1\}^n}$, where each function $f = \mathsf{PF}_y$ in the family is described by $d_f \coloneqq y$. For the anti-piracy security, we will consider the function distribution $\mathcal{D}_f \coloneqq \mathcal{U}(\{0,1\}^n)$: the uniform distribution over $\{0,1\}^n$; and the family of distributions $\mathcal{X} \coloneqq \{\mathcal{X}_y\}_{y\in\{0,1\}^n}$ such that for any $y \in \{0,1\}^n$, $\mathcal{X}_y$:

- samples $x \xleftarrow{\$} \{0,1\}^n$ and yields (y, x) with probability 1/3;
- samples $x \xleftarrow{\$} \{0,1\}^n$ and yields (x, y) with probability 1/3;
- samples $x, x' \xleftarrow{\$} \{0,1\}^n$ and yields (x, x') with probability 1/3.

6.2 Constructions

Let $\{\mathsf{PF}_y\}_{y\in\{0,1\}^n}$ be the family to be copy-protected, where $n \coloneqq n(\lambda)$ is a polynomial in λ. We define ℓ_0, ℓ_1, ℓ_2 such that $n = \ell_0 + \ell_1 + \ell_2$ and $\ell_2 - \ell_0$ is large enough. For this construction, we need three pseudorandom functions (PRFs):

- A puncturable extracting PRF $\mathsf{PRF}_1 : \mathcal{K}_1 \times \{0,1\}^n \to \{0,1\}^m$ with error $2^{-\lambda-1}$, where m is a polynomial in λ and $n \geq m + 2\lambda + 4$.
- A puncturable injective PRF $\mathsf{PRF}_2 : \mathcal{K}_2 \times \{0,1\}^{\ell_2} \to \{0,1\}^{\ell_1}$ with failure probability $2^{-\lambda}$, with $\ell_1 \geq 2\ell_2 + \lambda$.
- A puncturable PRF $\mathsf{PRF}_3 : \mathcal{K}_3 \times \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$.

Hardcoded: Keys $(\mathsf{k}_1, \mathsf{k}_2, \mathsf{k}_3) \in \mathcal{K}_1 \times \mathcal{K}_2 \times \mathcal{K}_3$, programs $\mathsf{R}_i^0, \mathsf{R}_i^1$ for all $i \in [\![1, \ell_0]\!]$.
On input $x = x_0 \| x_1 \| x_2$ and vectors $v_0, v_1, \cdots, v_{\ell_0}$ where each $v_i \in \mathbb{F}_2^n$, do the following:

1. (**Hidden Trigger Mode**) If $\mathsf{PRF}_3(\mathsf{k}_3, x_1) \oplus x_2 = x_0 \| \mathsf{Q}'$ and $x_1 = \mathsf{PRF}_2(\mathsf{k}_2, x_0 \| \mathsf{Q}')$: treat Q' as a classical circuit and output $\mathsf{Q}'(v_1, \cdots, v_{\ell_0})$.
2. (**Normal Mode**) If for all $i \in [\![1, \ell_0]\!]$, $\mathsf{R}_i^{x_i}(v_i) = 1$, then output $\mathsf{PRF}_1(\mathsf{k}_1, x)$. Otherwise, output $\perp$.

Fig. 1. Program P.

Construction 1 (Quantum Copy-Protection of Point Functions).

- $\rho_y \leftarrow \mathsf{PF.Protect}(y)$:
  - *Sample ℓ_0 random coset states $\{|A_{i,s_i,s_i'}\rangle\}_{i \in [\![1,\ell_0]\!]}$, where each subspace $A_i \subseteq \mathbb{F}_2^n$ if of dimension $\frac{n}{2}$.*
  - *For each coset state $|A_{i,s_i,s_i'}\rangle$, prepare the obfuscated membership programs $\mathsf{R}_i^0 = i\mathcal{O}(A_i + s_i)$ and $\mathsf{R}_i^1 = i\mathcal{O}(A_i^\perp + s_i')$.*
  - *Sample $k_i \leftarrow \mathsf{PRF}_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1, 2, 3\}$.*
  - *Prepare the program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$, where P is described in Figure 1.*
  - *Compute $z \coloneqq \mathsf{PRF}_1(k_1, y)$.*
  - *Return $\rho_y \coloneqq \left(\{|A_{i,s_i,s_i'}\rangle\}_{i \in [\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$.*
- $m \leftarrow \mathsf{PF.Eval}(\rho_y, x)$:
  - *Parse $\rho_y = \left(\{|A_{i,s_i,s_i'}\rangle\}_{i \in [\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$.*
  - *Parse x as $x \coloneqq x_0 \| x_1 \| x_2$.*
  - *For each $i \in [\![1, \ell_0]\!]$, if $x_{0,i} = 1$, apply $\mathsf{H}^{\otimes n}$ to $|A_{i,u_i,u_i'}\rangle$; if $x_{0,i} = 0$, leave the state unchanged.*
  - *Let σ be the resulting state (which can be interpreted as a superposition over tuples of ℓ_0 vectors). Run $\hat{\mathsf{P}}$ coherently on input x and σ, and measure the final output register to obtain z'.*
  - *Return 1 if $z' = z$, otherwise return 0.*

The semi-quantum copy-protection scheme for point functions is presented in Construction 2, which is essentially obtained by applying our compiler in Section 4 to Construction 1.

Construction 2 (Semi-Quantum Copy-Protection of Point Functions).

- $\mathsf{PF.Protect}(y)$: *This is now an interactive protocol between a sender and a receiver. The sender does the following:*

– *Run Protocol 5 independently ℓ_0 times with the receiver to obtain*

$$\left(\{A_i, s_i, s'_i\}_{i\in[\![1,\ell_0]\!]}, \{(\mathsf{R}^0_i, \mathsf{R}^1_i)\}_{i\in[\![1,\ell_0]\!]}\right).$$

The receiver obtains the corresponding $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}$.

– *Sample PRF keys k_i for PRF_i with $i \in \{1,2,3\}$.*

– *Prepare the program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$, where P is described in Figure 1.*

– *Compute $z := \mathsf{PRF}_1(\mathsf{k}_1, y)$.*

– *Send $\left(\hat{\mathsf{P}}, z\right)$ to the receiver.*

- $m \leftarrow \mathsf{PF.Eval}(\rho_y, x)$*:*
  – *Parse $\rho_y = \left(\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$.*
  – *Parse x as $x := x_0\|x_1\|x_2$.*
  – *For each $i \in [\![1,\ell_0]\!]$, if $x_{0,i} = 1$, apply $\mathsf{H}^{\otimes n}$ to $|A_{i,u_i,u'_i}\rangle$; if $x_{0,i} = 0$, leave the state unchanged.*
  – *Let σ be the resulting state (which can be interpreted as a superposition over tuples of ℓ_0 vectors). Run $\hat{\mathsf{P}}$ coherently on input x and σ, and measure the final output register to obtain z'.*
  – *Return 1 if $z' = z$, otherwise return 0.*

Theorem 4. *Assuming the existence of post-quantum indistinguishability obfuscation, one-way functions, and compute-and-compare obfuscation for the class of unpredictable distributions, Construction 1 and Construction 2 have* correctness *and* anti-piracy security*.*

The correctness of our protocols follows directly from the correctness of the copy-protection of PRFs construction of [CLLZ21, Lemma 7.13].

The security of our protocols relies on a new security notion for (semi-quantum) single-decryptors. We recall its definitions and introduce this new security notion – which we call anti-piracy security (real-or-random style) – in Appendix C. We show that the [CLLZ21]'s single-decryptor scheme also achieves this new security definition. The security proof of our constructions then follows the same strategy as that of copy-protection of PRFs given in [CLLZ21], except that we reduce security to our new single-decryptors definitions. We refer the reader to Appendix D for a detailed proof.

References

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

AC12. Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *44th ACM STOC*, pages 41–60. ACM Press, May 2012.

AGKZ20. Ryan Amos, Marios Georgiou, Aggelos Kiayias, and Mark Zhandry. One-shot signatures and applications to hybrid quantum/classical authentication. In *52nd ACM STOC*, pages 255–268. ACM Press, June 2020.

AK21. Prabhanjan Ananth and Fatih Kaleoglu. Unclonable encryption, revisited. In *TCC 2021, Part I*, LNCS 13042, pages 299–329. Springer, Heidelberg, November 2021.

AKL+22. Prabhanjan Ananth, Fatih Kaleoglu, Xingjian Li, Qipeng Liu, and Mark Zhandry. On the feasibility of unclonable encryption, and more. In *CRYPTO 2022, Part II*, LNCS 13508, pages 212–241. Springer, Heidelberg, August 2022.

AKL23. Prabhanjan Ananth, Fatih Kaleoglu, and Qipeng Liu. Cloning games: A general framework for unclonable primitives. Cryptology ePrint Archive, Paper 2023/128, 2023.

AL21. Prabhanjan Ananth and Rolando L. La Placa. Secure software leasing. In *EUROCRYPT 2021, Part II*, LNCS 12697, pages 501–530. Springer, Heidelberg, October 2021.

BCM+18. Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh V. Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. In *59th FOCS*, pages 320–331. IEEE Computer Society Press, October 2018.

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, 2020.

BF10. Niek J. Bouman and Serge Fehr. Sampling in a quantum population, and applications. In *CRYPTO 2010*, LNCS 6223, pages 724–741. Springer, Heidelberg, August 2010.

BGI+01. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *CRYPTO 2001*, LNCS 2139, pages 1–18. Springer, Heidelberg, August 2001.

BGI14. Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *PKC 2014*, LNCS 8383, pages 501–519. Springer, Heidelberg, March 2014.

BJL+21. Anne Broadbent, Stacey Jeffery, Sébastien Lord, Supartha Podder, and Aarthi Sundaram. Secure software leasing without assumptions. In *TCC 2021, Part I*, LNCS 13042, pages 90–120. Springer, Heidelberg, November 2021.

BL20. Anne Broadbent and Sébastien Lord. Uncloneable Quantum Encryption via Oracles. 158:4:1–4:22, 2020.

Bra18. Zvika Brakerski. Quantum FHE (almost) as secure as classical. In *CRYPTO 2018, Part III*, LNCS 10993, pages 67–95. Springer, Heidelberg, August 2018.

BS17. Shalev Ben-David and Or Sattath. Quantum tokens for digital signatures. Cryptology ePrint Archive, Report 2017/094, 2017.

BW13. Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT 2013, Part II*, LNCS 8270, pages 280–300. Springer, Heidelberg, December 2013.

CCKW19. Alexandru Cojocaru, Léo Colisson, Elham Kashefi, and Petros Wallden. QFactory: Classically-instructed remote secret qubits preparation. In *ASIACRYPT 2019, Part I*, LNCS 11921, pages 615–645. Springer, Heidelberg, December 2019.

CLLZ21. Andrea Coladangelo, Jiahui Liu, Qipeng Liu, and Mark Zhandry. Hidden cosets and applications to unclonable cryptography. In *CRYPTO 2021, Part I*, LNCS 12825, pages 556–584, Virtual Event, August 2021. Springer, Heidelberg.

CMP20. Andrea Coladangelo, Christian Majenz, and Alexander Poremba. Quantum copy-protection of compute-and-compare programs in the quantum random oracle model. Cryptology ePrint Archive, Report 2020/1194, 2020.

CV22. Eric Culf and Thomas Vidick. A monogamy-of-entanglement game for subspace coset states. *Quantum*, 6:791, 2022.

GGM84. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984.

GMP22. Alexandru Gheorghiu, Tony Metger, and Alexander Poremba. Quantum cryptography with classical communication: parallel remote state preparation for copy-protection, verification, and more. Cryptology ePrint Archive, Report 2022/122, 2022.

GP20. Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. Cryptology ePrint Archive, Report 2020/1010, 2020.

GV19. Alexandru Gheorghiu and Thomas Vidick. Computationally-secure and composable remote state preparation. In *60th FOCS*, pages 1024–1033. IEEE Computer Society Press, November 2019.

HILL99. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

HMNY21. Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Quantum encryption with certified deletion, revisited: Public key, attribute-based, and classical communication. In *ASIACRYPT 2021, Part I*, LNCS 13090, pages 606–636. Springer, Heidelberg, December 2021.

KNY21. Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. Secure software leasing from standard assumptions. In *TCC 2021, Part I*, LNCS 13042, pages 31–61. Springer, Heidelberg, November 2021.

KPTZ13. Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *ACM CCS 2013*, pages 669–684. ACM Press, November 2013.

Mah18a. Urmila Mahadev. Classical homomorphic encryption for quantum circuits. In *59th FOCS*, pages 332–338. IEEE Computer Society Press, October 2018.

Mah18b. Urmila Mahadev. Classical verification of quantum computations. In *59th FOCS*, pages 259–267. IEEE Computer Society Press, October 2018.

MV21. Tony Metger and Thomas Vidick. Self-testing of a single quantum device under computational assumptions. In *ITCS 2021*, volume 185, pages 19:1–19:12. LIPIcs, January 2021.

MY04. Dominic Mayers and Andrew Yao. Self testing quantum apparatus. *Quantum Info. Comput.*, 4(4):273–286, jul 2004.

NC11. Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, USA, 10th edition, 2011.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

RS20. Roy Radian and Or Sattath. Semi-quantum money. Cryptology ePrint Archive, Report 2020/414, 2020.

Shm22a. Omri Shmueli. Public-key quantum money with a classical bank. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 790–803, 2022.

Shm22b. Omri Shmueli. Semi-quantum tokenized signatures. In *CRYPTO 2022, Part I*, LNCS 13507, pages 296–319. Springer, Heidelberg, August 2022.

SW22. Or Sattath and Shai Wyborski. Uncloneable decryptors from quantum copy-protection, 2022.

VZ21. Thomas Vidick and Tina Zhang. Classical proofs of quantum knowledge. In *EUROCRYPT 2021, Part II*, LNCS 12697, pages 630–660. Springer, Heidelberg, October 2021.

Wie83. Stephen Wiesner. Conjugate coding. *ACM Sigact News*, 15(1):78–88, 1983.

Wil11. Mark M Wilde. From classical to quantum shannon theory. *arXiv preprint arXiv:1106.1445*, 2011.

Zha19. Mark Zhandry. Quantum lightning never strikes the same state twice. In *EUROCRYPT 2019, Part III*, LNCS 11478, pages 408–438. Springer, Heidelberg, May 2019.

Supplementary Material

A Preliminaries

A.1 Quantum Computation

Quantum gates. We refer to the following well-known unitary gates:

- *Pauli gates:* $\mathsf{X} : |a\rangle \mapsto |1-a\rangle$, $\mathsf{Z} : |a\rangle \mapsto (-1)^a |a\rangle$ and $\mathsf{Y} \coloneqq i\mathsf{XZ}$, for each $a \in \{0,1\}$.
- *Hadamard gate:* $\mathsf{H} : |a\rangle \mapsto \frac{1}{\sqrt{2}} |0\rangle + \frac{(-1)^a}{\sqrt{2}} |1\rangle$, for each $a \in \{0,1\}$.
- *Rotation gates:* $\mathsf{R}_\phi : |a\rangle \mapsto e^{ia\phi} |a\rangle$, for each $a \in \{0,1\}$. We obtain the T gate where $\phi = \frac{\pi}{4}$, the phase gate P where $\phi = \frac{\pi}{2}$.
- *Controlled gates:* for any k-qubit unitary quantum gate U, we define the controlled-U as: $\mathsf{Ctrl\text{-}U} : |a\rangle |x\rangle \mapsto |a\rangle \mathsf{U}^a |x\rangle$, for each $a \in \{0,1\}$ and $x \in \{0,1\}^k$. In particular, we write the controlled-NOT gate as $\mathsf{CNOT} : |a\rangle |b\rangle \mapsto |a\rangle |b \oplus a\rangle$.
- *Toffoli gates:* $\mathsf{CCNOT} : |a,b,c\rangle \mapsto |a,b,c \oplus (a \cdot b)\rangle$ for each $(a,b,c) \in \{0,1\}^3$.

A.1.1 Efficiency in the Quantum Setting

Definition 3 (Efficiency).

(i) ***Efficient unitaries:*** *a family of unitaries $\{U_\lambda \in \mathcal{U}(\mathcal{H}_\lambda)\}_{\lambda\in\mathbb{N}}$ is efficient if there exists a (classical) polynomial-time Turing machine M that, on input 1^λ, outputs a description of a circuit (with a fixed gate set) that implements the unitary.*

(ii) ***Efficient isometries:*** *a family of isometries $\{V_\lambda : \mathcal{H}_{A_\lambda} \to \mathcal{H}_{B_\lambda}\}_{\lambda\in\mathbb{N}}$ is efficient if there exists an efficient family of unitaries $\{U_\lambda \in \mathcal{U}(\mathcal{H}_{B_\lambda})\}_{\lambda\in\mathbb{N}}$ such that $V_\lambda = U_\lambda(\mathcal{I}_{A_\lambda} \otimes |0_{k(\lambda)}\rangle)$, where $k(\lambda) = \dim(\mathcal{H}_{B_\lambda}) - \dim(\mathcal{H}_{A_\lambda})$.*

(iii) ***Efficient observables:*** *a family of binary observables $\{Z_\lambda : \mathrm{Herm}(\mathcal{H}_{A_\lambda})\}_{\lambda\in\mathbb{N}}$ is efficient if there exists a family of Hilbert spaces $\mathcal{H}_{B_\lambda}$ with $\dim(\mathcal{H}_{B_\lambda}) = \mathsf{poly}(\lambda)$, and a family of efficient unitaries $\{U_\lambda \in \mathcal{U}(\mathcal{H}_{A_\lambda} \otimes \mathcal{H}_{B_\lambda})\}_{\lambda\in\mathbb{N}}$ such that for any $|\psi\rangle_A \in \mathcal{H}_A$:*

$$U^\dagger(\sigma_Z \otimes \mathcal{I})U_\lambda(|\psi\rangle_A |0\rangle_B) = (Z_\lambda |\psi\rangle_A) \otimes |0\rangle_B . \tag{4}$$

(iv) ***Efficient measurements:*** *a family of measurements $\{M_\lambda = \{M_\lambda^{(i)} \in \mathcal{L}(\mathcal{H}_{A_\lambda})\}_{i\in\mathcal{A}}\}_{\lambda\in\mathbb{N}}$ is efficient if the isometry*

$$|\psi\rangle \mapsto \sum_{i\in\mathcal{A}} |i\rangle \otimes M_\lambda^{(i)} |\psi\rangle \tag{5}$$

is efficient.

A.1.2 Distance Measures

Definition 4 (Norms). *Let $A \in \mathcal{L}(\mathcal{H})$ with singular values $\lambda_1, \ldots, \lambda_n \geq 0$. Then, the trace norm is defined as*

$$\|A\|_1 = \sum_i \lambda_i \,.$$

Definition 5 (Trace distance). *For two quantum states $\rho, \sigma \in \mathrm{Pos}(\mathcal{H})$, the trace distance between them is*

$$\Delta(\rho, \sigma) \coloneqq \frac{1}{2}\|\rho - \sigma\|_1.$$

Definition 6 (Approximate equality, [MV21, Definition 2.8 and Definition 2.14]). *We overload the symbol "$\approx$" in the following ways (leaving the dependence on the security parameter implicit in the quantities on the left):*

1. **Complex numbers:** *For $a, b \in \mathbb{C}$ we define:*

$$a \approx_\epsilon b \iff |a - b| = O(\epsilon) + \mathsf{negl}(\lambda)\,.$$

2. **Operators:** *For $A, B \in \mathcal{L}(\mathcal{H})$, we define:*

$$A \approx_\epsilon B \iff \|A - B\|_1^2 = O(\epsilon) + \mathsf{negl}(\lambda)\,.$$

   (We will most frequently use this for (possibly subnormalised) quantum states $A, B \in \mathrm{Pos}(\mathcal{H})$.)

3. **Operators on a state:** *For $A, B \in \mathcal{L}(\mathcal{H})$ and $\psi \in \mathrm{Pos}(\mathcal{H})$, we define:*

$$A \approx_{\epsilon,\psi} B \iff \mathrm{Tr}\big[(A-B)^\dagger (A-B)\psi\big] = O(\epsilon) + \mathsf{negl}(\lambda)\,.$$

4. **Computationally indistinguishable states:** *For two (families of not necessarily normalised) states $\psi, \psi' \in \mathrm{Pos}(\mathcal{H})$ which are computationally indistinguishable up to δ (i.e., no QPT distinguisher has advantage exceeding δ in distinguishing ψ from ψ'[12]), we write:*

$$\psi \overset{c}{\approx}_\delta \psi'\,.$$

   We can also define computational indistinguishability with respect to non-uniform QPT *algorithms with* quantum advice*, denoted by $\mathcal{A} \coloneqq \{\mathcal{A}_\lambda, \phi_\lambda\}_{\lambda\in\mathbb{N}}$, where each $\mathcal{A}_\lambda$ is the classical description of a $\mathsf{poly}(\lambda)$-size quantum circuit, and ϕ_λ is some (not necessarily efficiently computable) non-uniform $\mathsf{poly}(\lambda)$-qubit quantum advice. In this work, we implicitly consider computational indistinguishability with respect to non-uniform QPT adversaries with quantum advice, unless stated explicitly otherwise.*

[12] A distinguisher $\mathcal{D}$ is a CPTP map from the input state to a classical single-qubit state (i.e. a distribution over $\{0,1\}$). The distinguishability is the trace distance between $\mathcal{D}(\psi)$ and $\mathcal{D}(\psi')$.

If we write $\approx_0$, we mean that the quantities are negligibly close. All asymptotic statements are understood to be in the limits $\epsilon \to 0$ and $\lambda \to \infty$.

We include a copy of some technical lemmas on state-dependent operator relations using computational indistinguishability from [MV21] in Appendix A.2 for the reader's convenience.

A.2 Properties of the State-Dependent Distance

A feature of the state-dependent distance is that if two operators are close in the state-dependent distance, we can replace one operator by the other *acting on either side of the state.*

Lemma 1 (Replacement lemma [MV21, Lemma 2.21]). *Let $\psi \in \mathrm{Pos}(\mathcal{H})$, and $A, B, C \in \mathcal{L}(\mathcal{H})$. If $A \approx_{\epsilon,\psi} B$ and $\|C\|_\infty = O(1)$, then*

$$\mathrm{Tr}[CA\psi] \approx_{\epsilon^{1/2}} \mathrm{Tr}[CB\psi] \,, \tag{6}$$
$$\mathrm{Tr}[AC\psi] \approx_{\epsilon^{1/2}} \mathrm{Tr}[BC\psi] \,. \tag{7}$$

Lemma 2 ([MV21, Lemma 2.22]). *Let $A, B \in \mathcal{L}(\mathcal{H})$ be linear operators, $C \in \mathcal{L}(\mathcal{H})$ a linear operator with constant operator norm, and $\psi \in \mathrm{Pos}(\mathcal{H})$ with $\mathrm{Tr}[\psi] \leq 1$. Then, the following holds:*

$$A \approx_{\epsilon,\psi} B \implies A\,\psi\,C \approx_\epsilon B\,\psi\,C \quad \text{and} \quad C\,\psi\,A^\dagger \approx_\epsilon C\,\psi\,B^\dagger \,. \tag{8}$$

The following lemma allows us to replace computationally indistinguishable states with one another in the state-dependent distance. This means that if two states are computationally indistinguishable and a state-dependent operator relation holds for one of the states, we can "lift" this relation to the other state, provided the operators are efficient.

Lemma 3 (Lifting lemma [MV21, Lemma 2.25]). *Let $\psi, \psi' \in \mathcal{D}(\mathcal{H})$ such that $\psi \overset{c}{\approx}_\delta \psi'$. Let $\mathcal{H}'$ be another Hilbert space with $\dim(\mathcal{H}') \geq \dim(\mathcal{H})$. For this case, let $\psi, \psi' \in \mathcal{D}(\mathcal{H}')$ such that $\psi \overset{c}{\approx}_\delta \psi'$. Let A be an efficient binary observable on $\mathcal{H}$, B an efficient binary observable on $\mathcal{H}'$, and $V : \mathcal{H} \to \mathcal{H}'$ an efficient isometry. Then:*

$$VAV^\dagger \approx_{\epsilon,\,\psi} B \implies VAV^\dagger \approx_{\epsilon^{1/4}+\delta,\psi'} B \,. \tag{9}$$

Finally, we recall some further miscellaneous properties of the state-dependent distance.

Lemma 4 ([MV21, Lemma 2.18]). *Let $\psi_i \in \mathrm{Pos}(\mathcal{H})$ for $i \in \{1, \ldots, n\}$ with constant n, and $A, B \in \mathcal{L}(\mathcal{H})$. Define $\psi = \sum_i \psi_i$. Then:*

$$\forall i \in [\![1, n]\!] : \; A \approx_{\epsilon,\psi_i} B \text{ iff } A \approx_{\epsilon,\psi} B \tag{10}$$

Lemma 5 ([MV21, Lemma 2.24]). *Let $\mathcal{H}_1, \mathcal{H}_2$ be Hilbert spaces with $\dim(\mathcal{H}_1) \leq \dim(\mathcal{H}_2)$ and $V : \mathcal{H}_1 \to \mathcal{H}_2$ an isometry. Let A and B be binary observables on $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively, $\psi \in \mathrm{Pos}(\mathcal{H}_1)$, and $\epsilon \geq 0$. Then for any $b \in \{0,1\}$:*

$$V^\dagger B V \approx_{\epsilon,\psi} A \implies V^\dagger B^{(b)} V \approx_{\epsilon,\psi} A^{(b)}\,, \tag{11}$$

$$B \approx_{\epsilon, V\psi V^\dagger} VAV^\dagger \implies B^{(b)} \approx_{\epsilon, V\psi V^\dagger} VA^{(b)}V^\dagger\,. \tag{12}$$

A.3 Sampling in a Quantum Population

In this section, we describe a generic framework presented in [BF10] for analyzing cut-and-choose strategies applied to quantum states.

A.3.1 Classical Sampling Stratiegies

Let $q \coloneqq (q_1, \ldots, q_n) \in \Omega^n$ be a string of length n. We consider the problem of estimating the relative Hamming weight of a substring $\omega(q|_{\bar{t}})$ by only looking at the substring $q|_t$ of q, for a subset $t \subset [\![1, n]\!]$. We consider sampling strategies $\Psi \coloneqq (P_T, P_S, f)$, where P_T is an (independently sampled) distribution over subsets $t \subseteq [\![1, n]\!]$, P_S is a distribution over seeds $s \in S$, and $f : \{(t, v) : t \subset [\![1, n]\!], v \in \Omega^t\} \times S \to \mathbb{R}$ is a function that takes the subset t, the substring v, and a seed s, and outputs an estimate for the relative Hamming weight of the remaining string. For a fixed subset t, seed s, and a parameter δ, define $B^\delta_{t,s}(\Psi) \subseteq \Omega^n$ as

$$B^\delta_{t,s} \coloneqq \{b \in \Omega^n : |\omega(b|_{\bar{t}}) - f(t, b|_t, s)| < \delta\}.$$

Then we define the *classical error probability* of strategy Ψ as follows.

Definition 7 (Classical Error Probability). *The classical error probability of a sampling strategy $\Psi \coloneqq (P_T, P_S, f)$ is defined as the following value, parameterized by $0 < \delta < 1$:*

$$\varepsilon^\delta_{\mathsf{classical}}(\Psi) \coloneqq \max_{q \in \Omega^n} \Pr_{t \leftarrow P_T, s \leftarrow P_S} \left[q \notin B^\delta_{t,s}(\Psi)\right].$$

A.3.2 Quantum Sampling Strategies

Now, let $A \coloneqq A_1, \ldots, A_n$ be an n-partite quantum system where the state space of each system A_i equals $\mathcal{H}_{A_i} = \mathbb{C}^d$ with $d = |\Omega|$, and let $\{|a\rangle\}_{a \in \Omega}$ be a fixed orthonormal basis of $\mathbb{C}^d$. A may be entangled with another system E, and we write the purified state on A and E as $|\psi\rangle_{AE}$. We consider the problem of testing whether the state on A is close to the all-zero reference state $|0\rangle_{A_1} \ldots |0\rangle_{A_n}$. There is a natural way to apply any sampling strategy $\Psi = (P_T, P_S, f)$ to this setting: sample t, s according to P_T, P_S, measure subsystems A_i for $i \in [\![1, t]\!]$ in basis $\{|a\rangle\}_a$ to observe $q|_t \in \Omega^{|t|}$, and compute an estimate $f(t, q|_t, s)$.

In order to analyze the effect of this strategy, we first consider the mixed state on registers T (holding the subset t), S (holding the seed s), and A, E that results from sampling t and s according to $P_{TS} \coloneqq P_T P_S$

$$\rho_{TSAE} \coloneqq \sum_{t,s} P_{TS}(t,s) \left|t,s\right\rangle \left\langle t,s\right|_{TS} \otimes \left|\psi\right\rangle \left\langle\psi\right|_{AE}.$$

Next, we compare this state to an *ideal* state, parameterized by $0 < \delta < 1$, of the form

$$\widetilde{\rho}_{TSAE} \coloneqq \sum_{t,s} P_{TS}(t,s) \left|t,s\right\rangle \left\langle t,s\right|_{TS} \otimes \left|\widetilde{\psi}^{ts}\right\rangle \left\langle\widetilde{\psi}^{ts}\right|_{AE} \text{ with } \left|\psi^{ts}\right\rangle_{AE} \in \mathsf{span}\left(B^{\delta}_{t,s}\right) \otimes \mathcal{H}_E,$$

where

$$\mathsf{span}\left(B^{\delta}_{t,s}\right) \coloneqq \mathsf{span}\left(\{\left|b\right\rangle : b \in B^{\delta}_{t,s}\}\right) = \mathsf{span}\left(\{\left|b\right\rangle : \left|\omega(b|_{\overline{t}}) - f(t, b|_t, s)\right| < \delta\}\right).$$

That is, $\widetilde{\rho}_{TSAE}$ is a state such that it holds *with certainty* that the state on registers $A|_{\overline{t}}E$, after having measured $A|_t$ and observing $q|_t$, is in a superposition of states with relative Hamming weight δ-close to $f(t, q|_t, s)$. This leads us to the definition of the *quantum error probability* of strategy Ψ.

Definition 8 (Quantum Error Probability). *The quantum error probability of a sampling strategy $\Psi \coloneqq (P_T, P_S, f)$ is defined as the following value, parameterized by $0 < \delta < 1$:*

$$\varepsilon^{\delta}_{\mathsf{quantum}}(\Psi) \coloneqq \max_{\mathcal{H}_E} \max_{\left|\psi\right\rangle_{AE}} \min_{\widetilde{\rho}_{TSAE}} \Delta\left(\rho_{TSAE}, \widetilde{\rho}_{TSAE}\right),$$

where the first max is over all finite-dimensional registers E, the second max is over all state $\left|\psi\right\rangle_{AE}$ and the min is over all ideal state $\widetilde{\rho}_{TSAE}$ of the form described above.

Finally, we relate the classical and quantum error probabilities.

Theorem 5 ([BF10]). *For any sampling strategy Ψ and $\delta > 0$,*

$$\varepsilon^{\delta}_{\mathsf{quantum}}(\Psi) \leq \sqrt{\varepsilon^{\delta}_{\mathsf{classical}}(\Psi)}.$$

Remark 1. The results presented here immediately generalize from the all-zero reference state $\left|0\right\rangle \ldots \left|0\right\rangle$ to an arbitrary reference state $\left|\varphi\right\rangle_A$ of the form $\left|\varphi\right\rangle_A = U_1 \left|0\right\rangle \ldots U_n \left|0\right\rangle$ for unitary operators U_i acting on $\mathbb{C}^d$. Indeed, the generalization follows simply by a suitable change of basis, defined by the U_i's.

In this work, we will only need to analyze one simple sample-and-estimate strategy $\Psi_{\mathsf{uniform}} \coloneqq (P_T, P_S, f)$, where P_T is the uniform distribution over subsets $t \subseteq [\![1, n]\!]$, P_S is empty and $f(t, q|_t) = \omega(q|_t)$. That is, f receives a uniformly random subset $q|_t$ of q, and outputs the relative Hamming weight of $q|_t$ as its guess for the relative Hamming weight of $q|_{\overline{t}}$. The classical error probability of this strategy can be bound using Hoeffding inequalities, which is done in [BF10, Appendix B.3], where it is shown to be bounded by $4\exp(\frac{-n\delta^2}{32})$ for parameter δ. Thus, we have the following corollary of Theorem 5.

Corollary 2. *The quantum error probability of* Ψ_{uniform} *with parameter* δ *is*

$$\varepsilon^{\delta}_{\mathsf{quantum}}(\Psi_{\mathsf{uniform}}) \leq 2\exp(\frac{-n\delta^2}{64}).$$

A.4 Indistinguishability Obfuscation

Definition 9 (Indistinguishability Obfuscator [BGI+01]). *A uniform* PPT *machine* $i\mathcal{O}$ *is called an indistinguishability obfuscator for a classical circuit class* $\{\mathcal{C}_\lambda\}$ *if the following conditions are satisfied:*

- *For all security parameters* $\lambda \in \mathbb{N}$*, for all* $C \in \mathcal{C}_\lambda$*, for all input* x*, we have that*
$$\Pr[C'(x) = C(x) \mid C' \leftarrow i\mathcal{O}(\lambda, C)] = 1.$$
- *For any (not necessarily uniform) distinguisher* $\mathcal{D}$*, for all security parameters* $\lambda \in \mathbb{N}$*, for all pairs of circuits* $C_0, C_1 \in \mathcal{C}_\lambda$*, we have that if* $C_0(x) = C_1(x)$ *for all inputs* x*, then*
$$\mathsf{Adv}^{i\mathcal{O}}(\lambda, \mathcal{A}) \coloneqq |\Pr[\mathcal{D}(i\mathcal{O}(\lambda, C_0)) = 1] - \Pr[\mathcal{D}(i\mathcal{O}(\lambda, C_1)) = 1]| \leq \mathsf{negl}(\lambda).$$

We further say that $i\mathcal{O}$ *is* δ*-secure, for some concrete negligible function* $\delta(\lambda)$*, if for all* QPT *adversaries* $\mathcal{A}$*, the advantage* $\mathsf{Adv}^{i\mathcal{O}}(\lambda, \mathcal{A})$ *is smaller than* $\delta(\lambda)^{\Omega(1)}$*.*

Quantum-secure instantiations. There has been recent progress in constructing quantum-secure indistinguishability obfuscation schemes [GP20, BDGM20] from cryptographic assumptions that conjecturally hold against quantum adversaries.

In [Zha19, Shm22a], it is shown that indistinguishability obfuscation schemes have the property of *subspace hiding*.

Lemma 6 ([Zha19, Shm22a]). *Let iO an indistinguishability obfuscation scheme, and assume that injective one-way functions exist. Let* $S = \{S_\lambda\}_{\lambda\in\mathbb{N}}$ *a subspace* $S \subseteq \mathbb{F}_2^\lambda$*. For a subspace* S'*, denote by* $C_{S'}$ *a classical circuit that checks membership in* S'*. Then, for every constant* $\delta \in (0, 1]$ *we have the following indistinguishability:*

$$\{i\mathcal{O}(C_{S_\lambda})\}_{\lambda\in\mathbb{N}} \overset{c}{\approx}_0 \{i\mathcal{O}(C_T) \mid T \xleftarrow{\$} \mathcal{S}_{S_\lambda}\}_{\lambda\in\mathbb{N}},$$

where $\mathcal{S}_{S_\lambda}$ *is the set of all subspaces of dimension* $\lambda - \lambda^\delta$ *that contain* S_λ*.*

A.5 Compute-and-Compare Obfuscation

Definition 10 (Compute-and-Compare Program).

A compute-and-compare program $\mathsf{CC}[f, y, m]$*, parametrized by a function* f*, a lock-value* y *and a message* m*, is such that* $\mathsf{CC}[f, y, m](x) = m$ *if* $f(x) = y$ *and* $\mathsf{CC}[f, y, m](x) = \perp$ *otherwise.*

Definition 11 (Unpredictable Distribution).

Let $\mathcal{D}$ be a distribution over pairs of the form $(\mathsf{CC}[f, y, m], \mathsf{aux})$ where $\mathsf{CC}[f, y, m]$ is a compute-and-compare program and aux some (possibly quantum) auxiliary information.

We say $\mathcal{D}$ is an unpredictable distribution if no QPT algorithm can learn y, given f and aux with non-negligible probability, where $(\mathsf{CC}[f, y, m]), \mathsf{aux} \leftarrow \mathcal{D}$.

More formally, for all QPT algorithm A,

$$\Pr\big[A(1^\lambda, f, \mathsf{aux}) = y\big] \leq \mathsf{negl}(\lambda)$$

where $(\mathsf{CC}[f, y, m], \mathsf{aux}) \leftarrow \mathcal{D}$.

Definition 12 (Compute-and-Compare Obfuscator). *An efficient algorithm ($\mathsf{CC} - \mathsf{Obf}$) is said to be a compute-and-compare obfuscator for a family of unpredictable distributions $\mathcal{D} = \{\mathcal{D}_\lambda\}$ if for all $\mathcal{D}_\lambda \in \mathcal{D}$:*

- $\mathsf{CC} - \mathsf{Obf}$ *is functionality preserving:*

$$\Pr\big[\mathsf{CC} - \mathsf{Obf}(1^\lambda, \mathsf{CC}[f, y, m])(x) = \mathsf{CC}[f, y, m](x)\big] \geq 1 - \mathsf{negl}(\lambda)$$

- $\mathsf{CC} - \mathsf{Obf}$ *has distributional indistinguishability: there exists an efficient simulator $\mathcal{S}im$ such that*

$$\big\{\mathsf{CC} - \mathsf{Obf}(1^\lambda, C), \mathsf{aux}\big\} \approx \big\{\mathcal{S}im(1^\lambda, C.param), \mathsf{aux}\big\}$$

A.6 Puncturable Pseudorandom Function

A pseudorandom function (PRF) system [GGM84] consists of a keyed function F and a set of keys $\mathcal{K}$ such that for a randomly chosen key $k \in \mathcal{K}$, the output of the function $F(k, x)$ for any input x in the input space $\mathcal{X}$ "looks" random to a QPT adversary, even when given a polynomially many evaluations of $F(k, \cdot)$. Puncturable PRFs have an additional property that some keys can be generated *punctured* at some point, so that they allow to evaluate the PRF at all points except for the punctured points. Furthermore, even with the punctured key, the PRF evaluation at a punctured point still looks random.

Punctured PRFs are originally introduced in [BW13, BGI14, KPTZ13], who observed that it is possible to construct such puncturable PRFs for the construction from [GGM84], which can be based on any one-way function [HILL99].

Definition 13 (Puncturable Pseudorandom Function). *A pseudorandom function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is a* puncturable pseudorandom function *if there is an addition key space $\mathcal{K}_p$ and three PPT algorithms $\mathsf{pPRF} = \langle \mathsf{KeyGen}, \mathsf{Puncture}, \mathsf{Eval} \rangle$ such that:*

- $\mathsf{k} \leftarrow \mathsf{KeyGen}(1^\lambda)$. *The key generation algorithm KeyGen takes the security parameter 1^λ as input and outputs a random key $\mathsf{k} \in \mathcal{K}$.*
- $\mathsf{k}\{x\} \leftarrow \mathsf{Puncture}(\mathsf{k}, x)$. *The puncturing algorithm $\mathsf{Puncture}$ takes as input a PRF key $\mathsf{k} \in \mathcal{K}$ and $x \in \mathcal{X}$, and outputs a key $\mathsf{k}\{x\} \in \mathcal{K}_p$.*

- $y \leftarrow \mathsf{Eval}(\mathsf{k}\{x\}, x')$. *The evaluation algorithm takes as input a punctured key* $\mathsf{k}\{x\} \in \mathcal{K}_p$ *and* $x' \in \mathcal{X}$*, and outputs a classical string* $y \in \mathcal{Y}$*.*

We require the following properties of pPRF*.*

- ***Functionality preserved under puncturing.*** *For all* $\lambda \in \mathbb{N}$*, for all* $x \in \mathcal{X}$*,*

$$\Pr\left[\forall x' \in \mathcal{X} \setminus \{x\} : \mathsf{Eval}(\mathsf{k}\{x\}, x') = \mathsf{Eval}(\mathsf{k}, x') \;\middle|\; \begin{matrix} \mathsf{k} \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda) \\ \mathsf{k}\{x\} \xleftarrow{\$} \mathsf{Puncture}(\mathsf{k}, x) \end{matrix}\right] = 1.$$

- ***Pseudorandom at punctured points.*** *For every* QPT *adversary* $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$*, and every* $\lambda \in \mathbb{N}$*, the following holds:*

$$\left|\Pr\left[1 \leftarrow \mathcal{A}_2(\mathsf{k}\{x^*\}, y, \tau) \;\middle|\; \begin{matrix} (x^*, \tau) \leftarrow \mathcal{A}_1(1^\lambda, \tau) \\ \mathsf{k} \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda) \\ \mathsf{k}\{x^*\} \xleftarrow{\$} \mathsf{Puncture}(\mathsf{k}, x^*) \\ y \leftarrow \mathsf{Eval}(\mathsf{k}, x^*) \end{matrix}\right] - \Pr\left[1 \leftarrow \mathcal{A}_2(\mathsf{k}\{x^*\}, y, \tau) \;\middle|\; \begin{matrix} (x^*, \tau) \leftarrow \mathcal{A}_1(1^\lambda, \tau) \\ \mathsf{k} \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda) \\ \mathsf{k}\{x^*\} \xleftarrow{\$} \mathsf{Puncture}(\mathsf{k}, x^*) \\ y \xleftarrow{\$} \mathcal{Y} \end{matrix}\right]\right| \leq \mathsf{negl}(\lambda),$$

where the probability is taken over the randomness of KeyGen*,* Puncture*, and* $\mathcal{A}_1$*.*

Denote the above probability as $\mathsf{Adv}^{\mathsf{pPRF}}(\lambda, \mathcal{A})$*. We further say that* pPRF *is* δ*-secure, for some concrete negligible function* $\delta(\lambda)$*, if for all* QPT *adversaries* $\mathcal{A}$*, the advantage* $\mathsf{Adv}^{\mathsf{pPRF}}(\lambda, \mathcal{A})$ *is smaller than* $\delta(\lambda)^{\Omega(1)}$*.*

The following definitions are taken from [CLLZ21].

Definition 14 (Statistically injective PRF). *A statistically injective (puncturable) PRF family with (negligible) failure probability* $\varepsilon(\cdot)$ *is a (puncturable) PRF family* PRF *such that with probability* $1 - \varepsilon(\lambda)$ *over the random choice of key* $\mathsf{k} \leftarrow \mathsf{KeyGen}(1^\lambda)$*, we have that* $\mathsf{PRF}(\mathsf{k}, \cdot)$ *is injective.*

Definition 15 (Extracting PRF). *An extracting (puncturable) PRF with error* $\varepsilon(\cdot)$ *for min-entropy* $k(\cdot)$ *is a (puncturable) PRF* PRF *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* $(\mathsf{k}, \mathsf{PRF}(\mathsf{k}, X))$ *and* $(\mathsf{k}, r \leftarrow \{0,1\}^{m(\lambda)})$ *is at most* $\varepsilon(\cdot)$*, where* $\mathsf{k} \leftarrow \mathsf{KeyGen}(1^\lambda)$*.*

A.7 Leveled Hybrid Quantum Fully Homomorphic Encryption

We rely on quantum fully homomorphic encryption of a specific structure, which was defined in [Shm22a].

Definition 16 (Leveled Hybrid Quantum Fully Homomorphic Encryption). *A hybrid leveled quantum fully homomorphic encryption scheme is given by* $\mathsf{QFHE} \coloneqq \langle \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{QOTP}, \mathsf{Eval}, \mathsf{Decrypt} \rangle$ *with the following syntax:*

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^\ell)$. *A* PPT *algorithm that given a security parameter* $\lambda \in \mathbb{N}$ *and target circuit bound* $\ell \in \mathbb{N}$*, outputs a classical key pair* $(\mathsf{pk}, \mathsf{sk})$.
- $|\psi\rangle^{(x,z)} \leftarrow \mathsf{QOTP}((x,z), |\psi\rangle)$. *A* QPT *algorithm that takes as input an* n*-qubit quantum state* $|\psi\rangle$ *and classical strings as quantum OTPs* $x, z \in \{0,1\}^n$ *and outputs its QOTP transformation* $|\psi\rangle^{(x,z)} \coloneqq (\otimes_{i \in [\![n]\!]} \mathsf{Z}^{z_i}) \cdot (\otimes_{i \in [\![n]\!]} \mathsf{X}^{x_i}) |\psi\rangle$. *We often call these one-time pads* (x,z) *the Pauli keys. Furthermore, if* $|\psi\rangle$ *is a* classical *string* m*, we ignore the Pauli key* z *and write* $\mathsf{QOTP}(x, m)$ *whose output is* $x \oplus m$.
- $\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$. *A* PPT *algorithm that takes as input a classical string* $x \in \{0,1\}^*$ *and the public key* pk *and outputs a classical ciphertext* ct.
- $x \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$. *A* PPT *algorithm that takes as input a classical ciphertext* ct *and the secret key* sk *and outputs a classical string* x.
- $(|\phi\rangle^{(x',z')}, \mathsf{ct}_{x',z'}) \leftarrow \mathsf{Eval}(\mathsf{pk}, (|\psi\rangle^{(x,z)}, \mathsf{ct}_{x,z}), C)$. *A* QPT *algorithm that takes as input a general quantum circuit* C*, a quantum one-time pad encrypted state* $|\psi\rangle^{(x,z)}$ *and a classical ciphertext* $\mathsf{ct}_{x,z}$ *of the pads. The evaluation outputs a QOTP encryption of some quantum state* $|\phi\rangle$ *encrypted under new keys* (x', z') *and a classical ciphertext* $\mathsf{ct}_{x',z'}$.

The scheme satisfies the following.

- ***Semantic Security.*** *For every polynomials* $m(\cdot), \ell(\cdot)$*, and* QPT *algorithm* $\mathcal{A} \coloneqq \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda \in \mathbb{N}}$ *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\left| \Pr\left[1 \leftarrow \mathcal{A}_2(m_0 \oplus x, \mathsf{ct}_x) \;\middle|\; \begin{array}{r} (m_0, m_1) \leftarrow \mathcal{A}_1(1^\lambda) \\ (\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda, 1^{\ell(\lambda)}) \\ x \xleftarrow{\$} \{0,1\}^{m(\lambda)} \\ \mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x) \end{array} \right] - \Pr\left[1 \leftarrow \mathcal{A}_2(m_1 \oplus x, \mathsf{ct}_x) \;\middle|\; \begin{array}{r} (m_0, m_1) \leftarrow \mathcal{A}_1(1^\lambda) \\ (\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda, 1^{\ell(\lambda)}) \\ x \xleftarrow{\$} \{0,1\}^{m(\lambda)} \\ \mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x) \end{array} \right] \right| \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

  where $\lambda \in \mathbb{N}$ *and* $m_0, m_1 \in \{0,1\}^{m(\lambda)}$.
  - *Denote the above probability as* $\mathsf{Adv}^{\mathsf{QFHE}}(\lambda, \mathcal{A})$*. We further say that* QFHE *is* δ*-secure, for some concrete negligible function* $\delta(\lambda)$*, if for all* QPT *adversaries* $\mathcal{A}$*, the advantage* $\mathsf{Adv}^{\mathsf{QFHE}}(\lambda, \mathcal{A})$ *is smaller than* $\delta(\lambda)^{\Omega(1)}$.
- ***Homomorphism.*** *For every polynomial* $\ell \coloneqq \ell(\lambda)$ *there is a negligible function* $\mathsf{negl}(\cdot)$ *such that the following holds. Let* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^\ell)$*, let* x, z *equal-length strings, let* $\mathsf{ct}_{x,z} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, (x,z))$*, let* C *a quantum circuit of size* $\leq \ell$*, let* $|\psi\rangle$ *a* $|x|$*-qubit state input for* C*. Then,* $\Delta(D_0, D_1) \leq \mathsf{negl}(\lambda)$*, where* D_0, D_1 *are defined as follows.*

– D_0: *The output state is* $|\psi'\rangle \leftarrow C(|\psi\rangle)$.

– D_1: *The output state generated by first evaluating* $(|\phi\rangle^{(x',z')}, \mathsf{ct}_{x',z'}) \leftarrow \mathsf{Eval}\left(\mathsf{pk}, (|\psi\rangle^{(x,z)}, \mathsf{ct}_{x,z}), C\right)$, *and then decrypting* $(x', z') \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{x',z'})$, $|\phi\rangle \leftarrow \mathsf{QOTP}\left((x', z'), |\phi\rangle^{(x',z')}\right)$.

Quantum-secure instantiations. Quantum Leveled Fully-Homomorphic encryption with the hybrid structure follows from the work of Mahadev [Mah18a] and Brakerski [Bra18], and can be based on the quantum hardness of Learning with Errors [Reg05]. Consequently, constructing QFHE that has hybrid structure, leveled, and has sub-exponential advantage security can be based on assuming LWE with sub-exponential indistinguishability.

A.8 Coset States

This section is taken verbatim from [CLLZ21].

For any subspace $A \subseteq \mathbb{F}_2^n$, its complement is $A^\perp \coloneqq \{b \in \mathbb{F}_2^n \mid \langle a, b\rangle = 0, \forall a \in A\}$. We have that $\dim(A) + \dim(A^\perp) = n$. We also let $|A| \coloneqq 2^{\dim(A)}$ denote the size of the subspace A.

Definition 17 (Subspace States). *For any subspace* $A \subseteq \mathbb{F}_2^n$, *the subspace state* $|A\rangle$ *is defined as*

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

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

Definition 18 (Coset States). *For any subspace* $A \subseteq \mathbb{F}_2^n$, *vectors* $s, s' \in \mathbb{F}_2^n$, *the coset state* $|A_{s,s'}\rangle$ *is defined as*

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

Note that given $|A\rangle$ and s, s', the coset state $|A_{s,s'}\rangle$ can be constructed efficiently.

Furthermore, for a subspace A and vectors s, s', we define $A + s \coloneqq \{v + s \mid v \in A\}$, and $A^\perp + s' \coloneqq \{w + s' \mid w \in A^\perp\}$.

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 $iO(A + s)$ to mean an indistinguishability obfuscation of the program that checks membership in $A + s$.

A.8.1 Strong Monogamy-of-Entanglement Property

Coset states satisfy the following strong monogamy-of-entanglement property, which will be used as the main tool in our construction for copy-protection.

Definition 19 (Coset-Monogamy Game [CLLZ21, CV22]). *The coset mongamy game between a challenger and a* QPT *adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ *is defined as follows.*

1. *Preparation. The challenger picks a uniformly random subspace* $A \subseteq \mathbb{F}_2^\lambda$ *of dimension* $\frac{\lambda}{2}$*, and two uniformly random vectors* $s, s' \in \mathbb{F}_2^n$*. It sends* $|A, s, s'\rangle, i\mathcal{O}(A+s), i\mathcal{O}(A^\perp + s')$ *to the adversary* $\mathcal{A}_0$*.*
2. *The adversary applies a quantum channel:* $\Phi : \mathcal{H}_A \to \mathcal{H}_B \otimes \mathcal{H}_C$ *where* $\mathcal{H}_A = (\mathbb{C}^2)^{\otimes\lambda}$ *and* $\mathcal{H}_B, \mathcal{H}_C$ *are arbitrary. It then computes* $\rho_{BC} := \Phi(|A_{s,s'}\rangle\langle A_{s,s'}| \otimes |i\mathcal{O}(A+s), i\mathcal{O}(A^\perp+s')\rangle\langle i\mathcal{O}(A+s), i\mathcal{O}(A^\perp+s')|)$*. It sends registers* B *to* $\mathcal{A}_1$ *and* C *to* $\mathcal{A}_2$*, respectively.*
3. *Question. The challenger sends the description of* A*, in the form of a basis for it, to both* $\mathcal{A}_1$ *and* $\mathcal{A}_2$*.*
4. *Answer.* $\mathcal{A}_1$ *returns* $s_1 \in \mathbb{F}_2^n$ *and* $\mathcal{A}_2$ *returns* $s_2 \in \mathbb{F}_2^n$*.*

The adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ *wins if and only if* $s_1 \in A + s$ *and* $s_2 \in A^\perp + s'$*. Let* $\mathsf{CosetMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), \lambda)$ *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 6 ([CLLZ21, Theorem 4.18]). *Assuming the existence of post-quantum indistinguishability obfuscation and one-way functions, then there exists a negligible function* $\mathsf{negl}(\cdot)$*, for any* QPT *adversary* $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*,*

$$\Pr\left[\,\mathsf{CosetMonogamy}((\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2), \lambda)\,\right] \leq \mathsf{negl}(\lambda).$$

A.9 Copy-Protection

In the following, we assume that $\mathcal{F}$ is such that each function f in the family has the same domain $\mathcal{X}$ and the same codomain $\mathcal{Y}$ and has a classical description d_f (polynomial in λ) that allows for an efficient computation of f.

Definition 20 (Copy-Protection Scheme of a Family $\mathcal{F}$). *A copy-protection scheme is a tuple of algorithms* $\langle \mathsf{Protect}, \mathsf{Eval}\rangle$ *with the following properties:*

- $\rho_f \leftarrow \mathsf{Protect}(1^\lambda, d_f)$*. On input the description* d_f *of a function* $f \in \mathcal{F}$*, the quantum protection algorithm outputs a quantum state* ρ_f*.*
- $y \leftarrow \mathsf{Eval}(1^\lambda, \rho, x)$*. On input a quantum state* ρ *and an input* $x \in \mathcal{X}$*, the quantum evaluation algorithm outputs an image* $y \in \mathcal{Y}$*.*

We ask a copy-protection scheme to have *correctness* and *anti-piracy* security. We define these two notions below.

Definition 21 (Correctness of a Copy-Protection Scheme). *A copy-protection scheme has* correctness *if the quantum protection of a function f computes f on every x with overwhelming probability.*

$$\forall f \in \mathcal{F},\ \forall x \in \mathcal{X},\ \Pr\left[\mathsf{Eval}(1^\lambda, \rho_f, x) = f(x)\ :\ \rho_f \leftarrow \mathsf{Protect}(1^\lambda, d_f)\right]$$

Piracy Game of Copy-Protection. We present below a *piracy game for copy-protection*, parameterized by a copy-protection scheme $\mathsf{CP} = \langle \mathsf{Protect}, \mathsf{Eval}\rangle$, a security parameter λ, a function distribution $\mathcal{D}_f$ and a family of distribution $\mathcal{X} = \{\mathcal{X}_f\}_{f\in\mathcal{F}}$. This game is between a challenger and an adversary represented by three algorithms $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$.

- **Setup phase:** The challenger samples $f \leftarrow \mathcal{F}$, then sends $\rho_f \leftarrow \mathsf{Protect}(1^\lambda, d_f)$ to $\mathcal{A}_0$.
- **Splitting phase:** $\mathcal{A}_0$ prepares a bipartite quantum state σ_{12}, then sends σ_1 to $\mathcal{A}_1$ and σ_2 to $\mathcal{A}_2$.
- **Challenge phase:** The challenger samples $(x_1, x_2) \leftarrow \mathcal{X}_f$, then sends x_1 to $\mathcal{A}_1$ and x_2 to $\mathcal{A}_2$.
- **Answer phase:** $\mathcal{A}_1$ returns y_1 and $\mathcal{A}_2$ returns y_2.

For $i \in \{1, 2\}$, we say that $\mathcal{A}_i$ answers correctly if $y_i = f(x_i)$. $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2$ win the game if both $\mathcal{A}_1$ and $\mathcal{A}_2$ answer correctly.

We denote the random variable that indicates whether an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ wins the game or not as $\mathsf{CP\text{-}AP}^{\langle \mathsf{Protect},\mathsf{Eval}\rangle}_{\mathcal{D}_f,\mathcal{X}}(1^\lambda, (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2))$.

Trivial Adversary. As noted in [CMP20] and [AKL+22], an adversary can always win the game with a trivial probability (that we define formally in the next paragraph) by applying the following strategy: $\mathcal{A}_0$ just forwards the quantum protection state to $\mathcal{A}_1$ or $\mathcal{A}_2$ and nothing to the other one. The one who receives the state can answer the challenge with probability close to 1 using the Eval algorithm, and the other one returns the optimal answer given their challenge.

Thus, given a family $\mathcal{F}$, a function distribution $\mathcal{D}_f$ and a family of challenge distribution $\mathcal{X} = \{\mathcal{X}_f\}_{f\in\mathcal{F}}$, we define the trivial probability of winning the piracy game as

$$p^{trivial}_{D_f,\{X_f\}_{f\in\mathcal{F}}} := \max_{i\in\{1,2\}}\left[\mathop{\mathbb{E}}_{d_f\in D_f}\left(\max_{y\in\mathcal{Y}} \Pr[y \mid x_i]\right)\right]$$

Definition 22 (δ-Anti-Piracy Security). *A copy-protection scheme $\langle \mathsf{Protect}, \mathsf{Eval}\rangle$ has* δ-anti-piracy security *with respect to the function distribution $\mathcal{D}_f$ and the family of challenge distribution $\mathcal{X} = \{\mathcal{X}_f\}_{f\in\mathcal{F}}$ if no QPT adversary $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2$ can win the anti-piracy game parametrized by the function distribution $\mathcal{D}_f$ and the family of challenge distribution $\mathcal{X}$ with probability significantly greater than δ.*

More precisely, for any QPT adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$

$$\Pr\left[\mathsf{CP\text{-}AP}^{\langle \mathsf{Protect},\mathsf{Eval}\rangle}_{\mathcal{D}_f,\mathcal{X}}(1^\lambda, (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)) = 1\right] \leq \delta + \mathsf{negl}(\lambda).$$

Definition 23 (Anti-Piracy Security). *Whenever a single-decryptor has* δ-anti-piracy security *with respect to a function distribution* $\mathcal{D}_f$ *and a family of challenge distribution* $\mathcal{X} = \{\mathcal{X}_f\}_{f\in\mathcal{F}}$ *with* $\delta = p^{trivial}_{\mathcal{D}_f,\mathcal{X}}$*, we simply write that it has* anti-piracy security.

Remark 2. For ease of notations, we will use f and d_f indifferently, and we will not write the dependance on λ when clear from the context.

B Semi-Quantum Copy-Protection

B.1 Proof of Correctness

Proof of Proposition 1. The proof of correctness includes three steps: (1) If the prover ran honestly then its output after the homomorphic evaluation step has negligible trace distance to (QOTP encrypted of) BB84 states and coset states; (2) The self-test protocol passes (that is, the protocol does not terminate at this step) with probability negligibly close to 1; (3) In the last step of coset-state generation, after discarding all BB84 states, the output of the prover at the end of Protocol 5 has negligible trace distance to the state described in Equation (2).

We describe the honest strategy. By the statistical correctness of the homomorphic encryption, at the end of step 5 of Protocol 5, the i-th quantum state that an honest prover holds in its quantum-evaluated registers has negligible trace distance to either $\bigotimes_{j=1}^{n} |(-1)^{\beta_{i,j}}\rangle$ (if the corresponding instance is a n-qubit BB84 state) or $|S_{i,\alpha_i,\beta_i}\rangle$ (if the corresponding instance is a coset state). That is, this negligible distance holds with probability 1 over the previous messages of the protocol.

For each coset-state instance i, we claim that the probability for such honest prover to have $\alpha_i \in S_i$ is negligible. It follows from the fact that if $\alpha_i \in S_i$, we have that $|S_{i,\alpha_i,\beta_i}\rangle = |S_{i,0,\beta_i}\rangle$. By just measuring this state in the computational basis, we get a non-zero vector $s \in S_i$ with overwhelming probability, even without knowing S_i or the QFHE secret key. This violates the semantic security of the QFHE, because S_i is a subspace of dimension $\frac{n}{2}$ chosen uniformly at random, for any vector $s \in \mathbb{F}_2^n$, the probability that $s \in S_i$ is negligible. It means that Protocol 5 terminates at step 6 with negligible probability.

Next, we show that an honest prover succeeds in the self-test rounds of Protocol 5 with probability negligibly close to 1. An honest prover behaves the same way in each execution of Protocol 1 and Protocol 2. Hence, to show that an honest prover succeeds in Protocol 3 with probability negligibly close to 1, it suffices to describe honest strategies for Protocol 1 and Protocol 2 that succeed with probability negligibly close to 1. We note that Protocol 4 is N sequential repetition of Protocol 3, and thus the completeness of Protocol 4 is also negligibly close to 1.

Claim 1. *There exists a* QPT *prover that is accepted in Protocol 1 with probability negligibly close to* 1.

Proof. In Protocol 1, the prover receives n keys $k_1, \ldots, k_n$ and returns answers for each key k_j individually. Since the verifier's checks are independent for each j, we only need to describe an honest procedure for one key k_j that succeeds in the verifier's checks for that j with probability negligibly close to 1. The honest strategy for a single key k_j is adapted from the one in [GV19, MV21, GMP22]. We spell out the details below.

From now on, for simplicity, we drop the subscript i and understand that we are considering the i-th instance. First, note that at the beginning of Protocol 1, for a given key $k_j \in \mathcal{K}_\theta$, the prover is having the state $|(-)^{\beta_j}\rangle = \mathsf{Z}^{\beta_j} |(-)^0\rangle$ in his quantum registers. The prover then adjoins a uniform superposition over all $x \in \mathcal{X}$, evaluate f_{k_j} in superposition to obtain the following state:

$$\frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{b\in\{0,1\}} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}} \sqrt{f_{k_j,b}(x)(y)} \mathsf{Z}^{\beta_j} |b\rangle |x\rangle |y\rangle$$

Preparing this state can be efficiently done (up to negligible error) using the Samp procedure from the definition of ENTCF families ([BCM+18, Definition 3.1] and [Mah18b, Definition 4.2]). The prover then measures the "image register" (i.e., the register that stores y) to obtain an image $y_j \in \mathcal{Y}$ and sends this back to the verifier. The post-measurement state for each j is

$$\begin{cases} |\hat{b}(k_j, y_j)\rangle |\hat{x}(k_j, y_j)\rangle & \text{if } k_j \in \mathcal{K}_0, \\ \frac{1}{\sqrt{2}} \left(|0\rangle |\hat{x}_0(k_j, y_j)\rangle + (-1)^{\beta_j} |1\rangle |\hat{x}_1(k_j, y_j)\rangle\right) & \text{if } k_j \in \mathcal{K}_1. \end{cases} \tag{13}$$

If the verifier selects a "pre-image round", the prover measures both registers in the computational basis and returns the result. From the states in Equation (13) it is clear that the prover succeeds with probability negligibly close to 1 in the pre-image round.

If the verifier selects a "Hadamard round", the prover measures the "x-register" in the Hadamard basis to obtain d_j and returns this to the verifier. We introduce the shorthand $b_j \coloneqq \hat{b}(k_j, y_j)$ and $u_j \coloneqq \hat{u}(k_j, y_j, d_j)$. At this point, the prover's state for each j is (up to a global phase):

$$\begin{cases} |b_j\rangle & \text{if } k_j \in \mathcal{K}_0, \\ |(-)^{u_j \oplus \beta_j}\rangle & \text{if } k_j \in \mathcal{K}_1. \end{cases} \tag{14}$$

The prover now receives a question $q = \theta$ and measures the remaining qubit in the computational basis if $q = 0$ and in the Hadamard basis if $q = 1$. Then it is clear from the expression for the prover's remaining qubit in Equation (14) that the prover will pass the verifier's check. □

Claim 2. *There exists a* QPT *prover that is accepted in Protocol 2 with probability negligibly close to* 1.

Proof. At the beginning of each instance of Protocol 2, the prover is having the state $|A_{\alpha,\beta}\rangle$ with $\alpha, \beta \in \{0,1\}^n$. The honest strategy for the prover in Protocol 2

is similar to the honest strategy for Protocol 1 described in Claim 1: the prover uses the ENTCF family to commits to each qubit of the state $|A_{\alpha,\beta}\rangle$ using the corresponding function key. A formal description of the commitment process is given in [Mah18b, Section 5.1]. In the last round, if $q = \theta = 0$, the prover measures each qubit in the computational basis and in the Hadamard basis if $q = \theta = 1$. It equivalents to either measure the state $|A_{\alpha,\beta}\rangle$ in the computational basis if $q = 0$ and in the Hadamard basis if $q = 1$.

Since the prover applies the same strategy for each qubit in the state, here we describe the state commitment process for the j-th qubit of the state $|A_{\alpha,\beta}\rangle$. For a given key $k_j \in \mathcal{K}_\theta$, we can write the prover's coset state as

$$\sum_{b_j \in \{0,1\}} \gamma_{b_j} |b_j\rangle |\psi_{b_j}\rangle$$

The prover then adjoins a uniform superposition over all $x \in \mathcal{X}$, evaluate f_{k_j} in superposition to obtain

$$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{b_j \in \{0,1\}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \gamma_{b_j} \sqrt{f_{k_j,b_j}(x)(y)} |b_j\rangle |\psi_{b_j}\rangle |x\rangle |y\rangle \tag{15}$$

The prover then measures the "y-register" to obtain an image $y_j \in \mathcal{Y}$ and sends this back to the verifier. The post-measurement state for each j is

$$\begin{cases} |\hat{b}(k_j, y_j)\rangle |\psi_{b_j}\rangle |\hat{x}(k_j, y_j)\rangle & \text{if } k_j \in \mathcal{K}_0, \\ \sum_{b_j \in \{0,1\}} \gamma_{b_j} |b_j\rangle |\psi_{b_j}\rangle |\hat{x}_{b_j}(k_j, y_j)\rangle & \text{if } k_j \in \mathcal{K}_1. \end{cases} \tag{16}$$

We note that the verifier always sends "Hadamard round" as the round type in Protocol 2. The prover measures the "x-register" in the Hadamard basis to obtain d_j and returns this to the verifier. The prover now receives a question $q = \theta$ and measures the j-th qubit in the computational basis if $q = 0$ and in the Hadamard basis if $q = 1$. Recall that we denote $u_j := \hat{u}(k_j, y_j, d_j)$. At this point, the prover's state (before the measurement) is (up to a global phase):

$$\begin{cases} |b_j\rangle |\psi_{b_j}\rangle & \text{if } k_j \in \mathcal{K}_0, \\ (\mathsf{X}^{u_j}\mathsf{H} \otimes \mathcal{I}) |b_j\rangle |\psi_{b_j}\rangle & \text{if } k_j \in \mathcal{K}_1. \end{cases} \tag{17}$$

The prover measures the j-th qubit and returns a bit v_j to the verifier. It is clear from Equation (17) that: (1) if the coset state is measured in the computational basis (corresponding to the case $q = 0$), the verifier obtains a vector $v \in A + \alpha$; or (2) the coset state is measured in the Hadamard basis (corresponding to the case $q = 1$), the verifier obtains a vector $s \in A^\perp + \beta$. This concludes the proof of the claim. □

Having described the honest behavior for the self-test step, we finish the proof of correctness. □

B.2 Proof of Soundness: Proof of Proposition 2

The rigidity argument we establish in this section for Protocol 3 will be based on the n-fold parallel rigidity proof from [GMP22]. We will make frequent use of some technical lemmas from the proof of that paper.

B.2.1 Devices

We model the actions of a general prover by a "device". This formalizes all possible actions that can be taken by the prover to compute his answers to the verifier in Protocol 1 and Protocol 2. By Naimark's theorem, up to adding dimensions to the prover's Hilbert space, we can assume without loss of generality that the prover only performs projective measurements (instead of more general POVMs).

Definition 24 (Devices [GMP22]). *A device $D \coloneqq (S, \Pi, M, P)$ is specified by the following:*

1. *A set $S = \{\psi^{(\vec{\theta})}\}_{\vec{\theta}\in\{0,1\}^n}$ of states $\psi^{(\vec{\theta})} \in \mathcal{D}(\mathcal{H}_D \otimes \mathcal{H}_Y)$, where $\dim(\mathcal{H}_Y) = |\mathcal{Y}|^n$ and the states are classical on $\mathcal{H}_Y$:*

$$\psi^{(\vec{\theta})} = \sum_{\vec{y}\in\mathcal{Y}^n} \psi_{\vec{y}}^{(\vec{\theta})} \otimes |\vec{y}\rangle\langle\vec{y}|_Y \,. \tag{18}$$

   In the context of Protocol 1 and Protocol 2, $\psi^{(\vec{\theta})}$ is the prover's state after returning $\vec{y}$ for the case where the verifier makes basis choices $\vec{\theta}$.[13] *Each $\psi^{(\vec{\theta})}$ also implicitly depends on the specific keys chosen by the verifier (not just the basis choice $\vec{\theta}$); all the statements we make hold on average over key choices (for a fixed basis choice $\vec{\theta}$). Furthermore, since Protocol 1 and Protocol 2 are actually used as sub-protocols in a bigger protocol (Protocol 5), $\psi^{(\vec{\theta})}$ also depends on all messages exchanged (before the executions of these sub-protocols) in Protocol 5; for clarity we suppress this dependence from the notation, as we will see later these dependencies do not affect the rigidity proofs of these sub-protocols.*

2. *In the case of Protocol 1, a projective measurement Π on $\mathcal{H}_D \otimes \mathcal{H}_Y$:*

$$\Pi = \left\{ \Pi^{(\vec{b},\vec{x})} = \sum_{\vec{y}} \Pi_{\vec{y}}^{(\vec{b},\vec{x})} \otimes |\vec{y}\rangle\langle\vec{y}|_Y \right\}_{\vec{b}\in\{0,1\}^n;\, \vec{x}\in\mathcal{X}^n} . \tag{19}$$

   This is the measurement used by the prover to compute his answer $(\vec{b}, \vec{x})$ in the pre-image challenge.

[13] In Protocol 1, the only two basis choices are $\vec{\theta} = \vec{0}$ and $\vec{\theta} = \vec{1}$. However, $\psi^{(\vec{\theta})}$ is still well-defined as the state that the prover (who is defined in terms of the quantum circuits he runs on a given input) would prepare if given keys of basis choice $\vec{\theta}$, even though this never occurs in Protocol 1. This is different from Protocol 2, as it is crucial for the verifier's procedure in Protocol 2 to use only $\vec{0}$ or $\vec{1}$ as the basis choice. Otherwise the protocol would be "undefined".

3. *In the case of Protocol 2, Π is the identity operator $\mathcal{I}$ on $\mathcal{H}_D \otimes \mathcal{H}_Y$. This is because in Protocol 2, there is no pre-image challenge.*

4. *A projective measurement M on $\mathcal{H}_D \otimes \mathcal{H}_Y$:*

$$M = \left\{ M^{(\vec{d})} = \sum_{\vec{y}} M_{\vec{y}}^{(\vec{d})} \otimes |\vec{y}\rangle\langle\vec{y}|_Y \right\}_{\vec{d}\in\{0,1\}^{w\times n}} . \tag{20}$$

This is the measurement used by the prover to compute his answer $\vec{d}$ in the Hadamard challenge. We use an additional Hilbert spaces $\mathcal{H}_R$ to record the outcomes of measuring M and write the post-measurement state after applying M to $\psi^{(\vec{\theta})}$ as

$$\sigma^{(\vec{\theta})} \coloneqq \sum_{\vec{y},\vec{d}} M_{\vec{y}}^{(\vec{d})} \psi_{\vec{y}}^{(\vec{\theta})} M_{\vec{y}}^{(\vec{d})} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}|_{YR} . \tag{21}$$

5. *A set $P = \{P_q\}$, where for each $q \in \{0,1\}$, P_q is a projective measurement on $\mathcal{H}_D \otimes \mathcal{H}_Y \otimes \mathcal{H}_R$:*

$$P_q = \left\{ P_q^{(\vec{v})} = \sum_{\vec{y},\vec{d}} P_{q,\vec{y},\vec{d}}^{(\vec{v})} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}|_{YR} \right\}_{\vec{v}\in\{0,1\}^n} . \tag{22}$$

In the context of Protocol 1 and Protocol 2, given question q, the prover will measure $\{P_q^{(\vec{v})}\}$ and return the outcome $\vec{v}$ as his answer.

Definition 25 (Efficient devices). *A device is called* efficient *if the states $\psi^{(\vec{\theta})}$ can be prepared efficiently and the measurements Π, M, and P_q can be performed efficiently (in the sense of Definition 3).*

B.2.2 Success Probabilities of a Device

During the self-testing protocol (Protocol 3), the verifier applies certain checks to the answers given by the prover. If the prover fails these checks, the verifier sets a flag to $\texttt{flag}_{\texttt{Pre}}$ or $\texttt{flag}_{\texttt{Had}}$ then aborts. Here, we define the probabilities that the prover passes these checks and relate these probabilities in both protocols Protocol 1 and Protocol 2.

Definition 26 (Success probabilities). *For any device $D \coloneqq (S, \Pi, M, P)$ we define $\gamma_P(D_{\texttt{bb84}})$ as the device's failure probability in a pre-image round, $\gamma_H(D_{\texttt{bb84}})$ as the failure probability in a Hadamard round in Protocol 1 and $\gamma_H(D_{\texttt{coset}})$ as the failure probability in a Hadamard round in Protocol 2:*

$$\gamma_P(D_{\texttt{bb84}}) \coloneqq \Pr\left[\texttt{flag}_{\texttt{bb84}} = \texttt{flag}_{\texttt{Pre}} \mid \textit{round type = pre-image round}\right] , \tag{23}$$

$$\gamma_H(D_{\texttt{bb84}}) \coloneqq \Pr\left[\texttt{flag}_{\texttt{bb84}} = \texttt{flag}_{\texttt{Had}} \mid \textit{round type = Hadamard round}\right] , \tag{24}$$

$$\gamma_H(D_{\texttt{coset}}) \coloneqq \Pr\left[\texttt{flag}_{\texttt{coset}} = \texttt{flag}_{\texttt{Had}}\right] . \tag{25}$$

Next, we give the definition of a perfect prover in Protocol 1. Informally, a perfect prover is accepted by the verifier in a pre-image round with probability negligibly close to 1.

Definition 27 (Perfect device in Protocol 1). *We call a device D* perfect *if $\gamma_P(D_{\mathsf{bb84}}) = \mathsf{negl}(\lambda)$.*

The following lemma says that for any device in Protocol 1 that has a non-negligible failure probability in the pre-image test, there is another perfect device that is "close" to the original one in the sense that its measurements are the same as for the original device and its states only differ by $O(\gamma_P(D))$. By using this lemma, for the rest of the rigidity proof, it suffices to only consider perfect devices: for any arbitrary device, we can first make a reduction to the corresponding perfect device at the cost of incurring an approximation error of $O(\gamma_P(D))$, and then apply our soundness proof to the perfect device.

Lemma 7 ([GMP22, Lemma 4.9]). *Let $D = (S, \Pi, M, P)$ be an efficient device in Protocol 1 with $\gamma_P(D_{\mathsf{bb84}}) < 1$, where $S = \left\{\psi^{(\vec{\theta})}\right\}$. Then there exists an efficient* perfect *device $D' = (S', \Pi, M, P)$, which uses the same measurements Π, M, P and whose states $S' = \left\{\psi'^{(\vec{\theta})}\right\}$ satisfy for any $\vec{\theta} \in \{0,1\}^n$:*

$$\psi'^{(\vec{\theta})} \approx_{\gamma_P(D_{\mathsf{bb84}})} \psi^{(\vec{\theta})}. \tag{26}$$

Proof. The proof of this lemma uses essentially the same technique to that of [MV21, Lemma 4.13], which in turn based on [Mah18b, Claim 7.2]. We give a sketch of the proof for correctness. A construction of D' is as follows. D' first prepares the states $\psi^{(\vec{\theta})}$ as D does, then applies the efficient unitary U_Π associated with the measurement Π:

$$|0\rangle\langle 0|_R \otimes \psi^{(\vec{\theta})} \overset{U_\Pi}{\longmapsto} |\vec{b},\vec{x}\rangle\langle\vec{b},\vec{x}|_R \otimes \Pi^{(\vec{b},\vec{x})}\psi^{(\vec{\theta})}\Pi^{(\vec{b},\vec{x})}. \tag{27}$$

Now D' coherently evaluates the (efficient) Chk-function on the Y-register of $\Pi^{(\vec{b},\vec{x})}\psi^{(\vec{\theta})}\Pi^{(\vec{b},\vec{x})}$ and the new register containing (b_i, x_i) for all $i \in [\![1,n]\!]$. If Chk succeeds, D' applies $U_\Pi^\dagger$ to the state, traces out the ancillary register R, and uses this as $\psi'^{(\vec{\theta})}$. Otherwise, D' repeats the process up to polynomially (in the security parameter) many times, and aborts if the Chk procedure never succeeds. Since $\gamma_P(D_{\mathsf{bb84}})$ is defined as the maximum failure probability of the pre-image test, and the Chk procedure fails if the pre-image check fails on any qubit, the probability of the Chk procedure failing is at most $n \cdot \gamma_P(D_{\mathsf{bb84}}) = O(\gamma_P(D_{\mathsf{bb84}}))$ by a union bound.

If $1 - \gamma_P(D_{\mathsf{bb84}})$ is negligible, the trace distance bound between $\psi^{(\vec{\theta})}$ and $\psi'^{(\vec{\theta})}$ is trivially satisfied. If $1 - \gamma_P(D_{\mathsf{bb84}})$ is non-negligible, the probability that Chk fails polynomially many times is negligible. Furthermore, by definition of the ENTCF family, the Chk procedure requires only the function key and not the trapdoor, which implies that it can be computed efficiently by the prover D'. It means that D' is efficient and perfect.

Fix $\vec{\theta}$. By Definition 6, we need to show $\left\|\psi'^{(\vec{\theta})} - \psi^{(\vec{\theta})}\right\|_1 \approx_{\gamma_P(D_{\mathsf{bb84}})^{1/2}} 0$. Since the probability of the Chk to succeed is at least $1 - O(\gamma_P(D_{\mathsf{bb84}}))$, by the gentle measurement lemma ([Wil11]), the post-measurement state after Chk has succeeded is $O(\gamma_P(D_{\mathsf{bb84}})^{1/2})$-close in trace distance to $U_\Pi(|0\rangle\langle 0|_R \otimes \psi^{(\vec{\theta})})U_\Pi^\dagger$. Because the trace distance is unitarily invariant, this implies that the state $\psi'^{(\vec{\theta})}$ is also $O(\gamma_P(D_{\mathsf{bb84}})^{1/2})$-close in trace distance to $\psi^{(\vec{\theta})}$. □

B.2.3 Rigidity Proof of Protocol 1

The rigidity proof of Protocol 1 follows identically from that of [GMP22]. In this section, we recall definitions and related technical lemmas from [GMP22] that are needed for our proof later. The main difference lies in the last verification procedure, in which our verification procedure also involves the Pauli keys from the QFHE. However, one can easily inspect their proof and see that this difference does not change most part of the proof. This essentially follows from the fact that the one-time pads (and generally, the homomorphic enryption) are independent of all the messages and verifier's secrets in the execution of Protocol 1, it only is used in the verification of the verifier as its secret input. When the difference appears, we will re-prove the lemma with respect to our protocol.

Definition 28 (Observables). *For a device $D \coloneqq (S, \Pi, M, P)$ with projective measurements as in Definition 24 and $\vec{\beta} \in \{0,1\}^n$, we define the following binary observables:*

$$Z_i = \sum_{\vec{v}} (-1)^{v_i} P_0^{(\vec{v})}, \tag{28}$$

$$X_i = \sum_{\vec{v}} (-1)^{v_i} P_1^{(\vec{v})}, \tag{29}$$

$$\tilde{X}_i = \sum_{\vec{v},\vec{y},\vec{d}} (-1)^{\beta_i \oplus v_i \oplus \hat{u}(k_i,y_i,d_i)} P_{1,\vec{y},\vec{d}}^{(\vec{v})} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}|_{YR}. \tag{30}$$

We further use the following notation for products of observables: for $\vec{a} \in \{0,1\}^n$, we define

$$Z(\vec{a}) \coloneqq Z_1^{a_1} \dots Z_n^{a_n} = \sum_{\vec{v}} (-1)^{\vec{a}\cdot\vec{v}} P_0^{(\vec{v})}, \tag{31}$$

and likewise for $X(\vec{a})$ and $\tilde{X}(\vec{a})$. It is easy to see that

$$\tilde{X}(\vec{a})_{\vec{y},\vec{d}} = (-1)^{\vec{a}\cdot(\vec{\beta}\oplus\hat{u}(\vec{k},\vec{y},\vec{d}))} X(\vec{a})_{\vec{y},\vec{d}}. \tag{32}$$

Remark 3. $\tilde{X}_i$ is not an observable that an efficient prover can implement because it depends on $\hat{u}(k,y,d)$, which requires the trapdoor information to be computed efficiently, and the Pauli key β, which the prover only has an encryption of it. Intuitively, while X_i describes the prover's answer, $\tilde{X}_i$ describes whether that answer is accepted by the verifier.

Definition 29 (Partial post-measurement states). *For $k \in \mathcal{K}_0 \cup \mathcal{K}_1$, $v \in \{0,1\}$ and $\beta \in \{0,1\}$ define the set $V_{\beta,k,v} \subseteq \mathcal{Y} \times \{0,1\}^w$ by the following condition:*

$$(y,d) \in V_{\beta,k,v} \text{ iff } \begin{cases} \hat{b}(k,y) = v & \text{if } k \in \mathcal{K}_0\,, \\ \hat{u}(k,y,d) = v \oplus \beta & \text{if } k \in \mathcal{K}_1\,. \end{cases} \tag{33}$$

Then for $\vec{\beta}, \vec{k}, \vec{\theta}, \vec{v}$ we define

$$\sigma^{(\vec{\beta},\vec{\theta},\vec{v})} = \sum_{y_1,d_1 \in V_{\beta_1,k_1,v_1}} \cdots \sum_{y_n,d_n \in V_{\beta_n,k_n,v_n}} \sigma^{(\vec{\theta})}_{\vec{y},\vec{d}} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}|\,. \tag{34}$$

Further for $\vec{a} \in \{0,1\}^n$ we define

$$\sigma^{(\vec{\beta},\vec{\theta},v,\vec{a})} \coloneqq \sum_{\vec{v}\colon \vec{v}\cdot\vec{a}=v} \sigma^{(\vec{\beta},\vec{\theta},\vec{v})}\,. \tag{35}$$

Remark 4. In the following, once $\vec{\beta}$ is fixed, we can drop $\vec{\beta}$ from these notations and simply write $\sigma^{(\vec{\theta},\vec{v})}$ and $\sigma^{(\vec{\theta},v,\vec{a})}$. The reason is that as we explained above, the involvement of $\vec{\beta}$ is primarily a technicality needed because of our protocol construction, but does not affect the modular proofs we present here. Another way to see it is to consider $\vec{\beta}$ as a part of the trapdoor information $\vec{t}$. Then we can write $\hat{u}'(k,y,d) \coloneqq \hat{u}(k,y,d) \oplus \beta$ and define $(y,d) \in V_{k,v}$ if $\hat{u}'(k,y,d) = v$ when $k \in \mathcal{K}_1$. For any statement involving these states, we understand that there is some $\vec{\beta}$ known by the verifier and these states are defined with respect to this $\vec{\beta}$.

Intuitively, when $\vec{\theta} = \vec{0}$, then for any $\vec{a} \in \{0,1\}^n$, $\sigma^{(\vec{0},v,\vec{a})}$ is that part of the state $\sigma^{(\vec{0})}$ for which the honest device would receive outcome v when measuring the observable $Z(\vec{a})$. The following lemma shows what outcomes a successful device must produce when measuring the observables from Definition 28 on the partial post-measurement states from Definition 29.

Lemma 8 ([GMP22, Corollary 4.18]). *Consider an efficient device $D = (S, \Pi, M, P)$ and a bit $v \in \{0,1\}$.*

1. *For any $\vec{\theta}, \vec{a} \in \{0,1\}^n$ such that $\theta_i = 0$ if $a_i = 1$, then:*

$$Z(\vec{a}) \approx_{\gamma_H(D_{\mathsf{bb84}}),\sigma^{(\vec{\theta},v,\vec{a})}} (-1)^v \mathcal{I}\,. \tag{36}$$

2. *For any $\vec{\theta}, \vec{a} \in \{0,1\}^n$ such that $\theta_i = 1$ if $a_i = 1$, then:*

$$X(\vec{a}) \approx_{\gamma_H(D_{\mathsf{bb84}}),\sigma^{(\vec{\theta},v,\vec{a})}} (-1)^v \mathcal{I}\,. \tag{37}$$

Next, we define isometries $\tilde{V}, V$ which can be shown to map the prover's observables to the corresponding Pauli observables.

Definition 30 (Rounding isometries [GMP22]). *For a device D with associated Hilbert space $\mathcal{H}_D$ and $\vec{y} \in \mathcal{Y}^{\times n}$, $d \in \{0,1\}^{w \times n}$, we define the isometry $\tilde{V}_{y,d} : \mathcal{H}_D \to \mathcal{H}_D \otimes \mathcal{H}_A \otimes \mathcal{H}_Q$ by the following action on an arbitrary state $|\varphi\rangle_D$:*

$$\tilde{V}_{\vec{y},\vec{d}}|\varphi\rangle_D \coloneqq \mathop{\mathbb{E}}_{\vec{a},\vec{b}\in\{0,1\}^n} \left(\left(\tilde{X}(\vec{a})_{\vec{y},\vec{d}} Z(\vec{b})_{\vec{y},\vec{d}}\right)_D \otimes \left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_A\right)|\varphi\rangle_D \otimes \left(|\Phi^+\rangle^{\otimes n}\right)_{AQ}, \tag{38}$$

where $|\Phi^+\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$ denotes an EPR pair, and $\left(|\Phi^+\rangle^{\otimes n}\right)_{AQ}$ is distributed between A and Q such that every EPR pair has one qubit in either system. We can combine the different $V_{y,d}$ into one isometry

$$\tilde{V} \coloneqq \sum_{\vec{y},\vec{d}} \tilde{V}_{\vec{y},\vec{d}} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}| : \mathcal{H}_D \otimes \mathcal{H}_Y \otimes \mathcal{H}_R \to \mathcal{H}_D \otimes \mathcal{H}_Y \otimes \mathcal{H}_R \otimes \mathcal{H}_A \otimes \mathcal{H}_Q\,. \tag{39}$$

We similarly define

$$V_{\vec{y},\vec{d}}|\varphi\rangle_D \coloneqq \mathop{\mathbb{E}}_{\vec{a},\vec{b}\in\{0,1\}^n} \left(\left(X(\vec{a})_{\vec{y},\vec{d}} Z(\vec{b})_{\vec{y},\vec{d}}\right)_D \otimes \left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_A\right)|\varphi\rangle_D \otimes \left(|\Phi^+\rangle^{\otimes n}\right)_{AQ} \tag{40}$$

and

$$V \coloneqq \sum_{\vec{y},\vec{d}} V_{\vec{y},\vec{d}} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}|\,. \tag{41}$$

The following lemma relates $\tilde{V}$ and V.

Lemma 9. *For any keys $\vec{k} \in \mathcal{K}_1^n$ and $\vec{\beta} \in \{0,1\}^n$:*

$$V_{\vec{y},\vec{d}} = \sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)_A \otimes \sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)_Q \tilde{V}_{\vec{y},\vec{d}}. \tag{42}$$

Proof. For any state $|\varphi\rangle_D$, we have:

$$\begin{aligned}
&\sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)_A \otimes \sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)_Q \tilde{V}_{\vec{y},\vec{d}}|\varphi\rangle_D \\
&= \mathop{\mathbb{E}}_{a,b\in\{0,1\}^n} \left(\tilde{X}(\vec{a})_{\vec{y},\vec{d}} Z(\vec{b})_{\vec{y},\vec{d}}\right)_D |\varphi\rangle_D \otimes \\
&\qquad \left[\left(\sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_A \otimes \sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)_Q \left(|\Phi^+\rangle^{\otimes n}\right)_{AQ}\right]
\end{aligned}$$

Repeatedly using that $(\sigma_Z)_A |\Phi^+\rangle_{AQ} = (\sigma_Z)_Q |\Phi^+\rangle_{AQ}$:

$$\begin{aligned}
&= \mathop{\mathbb{E}}_{a,b\in\{0,1\}^n} \left(\tilde{X}(\vec{a})_{\vec{y},\vec{d}} Z(\vec{b})_{\vec{y},\vec{d}}\right)_D |\varphi\rangle_D \otimes \\
&\qquad \left[\left(\sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)\sigma_X(\vec{a})\sigma_Z(\vec{b})\sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d}) \oplus \vec{\beta}\right)\right)_A \left(|\Phi^+\rangle^{\otimes n}\right)_{AQ}\right]
\end{aligned}$$

Since $\sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d})\oplus\vec{\beta}\right)\sigma_X(\vec{a})\sigma_Z(\vec{b})\sigma_Z\left(\hat{u}(\vec{k},\vec{y},\vec{d})\oplus\vec{\beta}\right) = (-1)^{\vec{a}\cdot\left(\hat{u}(\vec{k},\vec{y},\vec{d})\oplus\vec{\beta}\right)}\sigma_X(\vec{a})\sigma_Z(\vec{b})$:

$$= \mathop{\mathbb{E}}_{a,b\in\{0,1\}^n}\left((-1)^{\vec{a}\cdot\left(\hat{u}(\vec{k},\vec{y},\vec{d})\oplus\vec{\beta}\right)}\tilde{X}(\vec{a})_{\vec{y},\vec{d}}Z(\vec{b})_{\vec{y},\vec{d}}\right)_D|\varphi\rangle_D\otimes\left[\left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_A\left(|\Phi^+\rangle^{\otimes n}\right)_{AQ}\right]$$

Recalling from Definition 28 that $(-1)^{\vec{a}\cdot\left(\hat{u}(\vec{k},\vec{y},\vec{d})\oplus\vec{\beta}\right)}\tilde{X}(\vec{a})_{\vec{y},\vec{d}} = X(\vec{a})_{\vec{y},\vec{d}}$:

$$= \mathop{\mathbb{E}}_{a,b\in\{0,1\}^n}\left(X(\vec{a})_{\vec{y},\vec{d}}Z(\vec{b})_{\vec{y},\vec{d}}\right)_D|\varphi\rangle_D\otimes\left[\left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_A\left(|\Phi^+\rangle^{\otimes n}\right)_{AQ}\right]$$
$$= V|\varphi\rangle_D\,. \qquad \square$$

We then show that the isometry $\tilde{V}$ maps the observables $\tilde{X}(\vec{a})Z(\vec{b})$ to the corresponding Pauli observables.

Lemma 10 ([GMP22, Lemma 4.28]). *For an efficient perfect device $D = (S,\Pi,M,P)$ and any $\vec{a},\vec{b}\in\{0,1\}^n$ we have*

$$\mathrm{Tr}\left[\tilde{V}^\dagger\left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)^\dagger_Q\tilde{V}\tilde{X}(\vec{a})_{DYR}Z(\vec{b})_{DYR}\sigma^{(\vec{1})}_{DYR}\right]\approx_{n^{1/2}\gamma_H(D_{\text{bb84}})^{1/8}}1\,. \tag{43}$$

By combining Lemma 9 and Lemma 10 we can show that the isometry V maps the observables $X(\vec{a})Z(\vec{b})$ to the corresponding Pauli observables.

Lemma 11 ([GMP22, Proposition 4.29]). *For an efficient perfect device $D = (S,\Pi,M,P)$ and any $\vec{a},\vec{b}\in\{0,1\}^n$ we have*

$$VX(\vec{a})Z(\vec{b})V^\dagger\approx_{n^{1/2}\gamma_H(D_{\text{bb84}})^{1/8},V\sigma^{(\vec{1})}V^\dagger}\left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_Q\otimes\mathcal{I}_{YRDA}\,. \tag{44}$$

B.2.4 Rigidity Proof of Protocol 2

Having established a characterization of the prover's observables $X(\vec{a})Z(\vec{b})$ in Protocol 1, we now use this to characterize the prover's behavior in Protocol 2.

Step 1: Modeling. First, we introduce the corresponding notion of post-measurement states for an efficient device of Protocol 2. Note that the two protocols are identical from the prover's point of view when the round type is the Hadamard round, and the marginal observables from Definition 28 are defined for Hadamard round. Thus we can use the same notation of marginal observables from Definition 28 (in particular, we only need the efficient observables $X(\vec{a})$ and $Z(\vec{b})$) for an efficient device in Protocol 2.

Definition 31. *For $\vec{k}\in(\mathcal{K}_0\cup\mathcal{K}_1)^n$, $\vec{v}\in\{0,1\}^n$ and $A\subseteq\mathbb{F}_2^n$, $\vec{\alpha},\vec{\beta}\in\{0,1\}^n$ define the set $V_{A,\vec{\alpha},\vec{\beta},\vec{k},\vec{v}}\subseteq\mathcal{Y}^n\times\{0,1\}^{w\times n}$ by the following condition:*

$$(\vec{y},\vec{d})\in V_{A,\vec{\alpha},\vec{\beta},\vec{k},\vec{v}}\ \text{iff}\ \begin{cases}\hat{b}(\vec{k},\vec{y})=\vec{v}\in A+\vec{\alpha} & \text{if } \vec{k}\in\mathcal{K}_0^n\,,\\ \hat{u}(\vec{k},\vec{y},\vec{d})\oplus\vec{v}\in A^\perp+\vec{\beta} & \text{if } \vec{k}\in\mathcal{K}_1^n\,.\end{cases} \tag{45}$$

Then for $\vec{\alpha}, \vec{\beta}, \vec{k}, \vec{\theta} \in \{\vec{0}, \vec{1}\}, \vec{v}$ we define

$$\sigma^{(A,\vec{\alpha},\vec{\beta},\vec{\theta},\vec{v})} = \sum_{\vec{y},\vec{d} \in V_{A,\vec{\alpha},\vec{\beta},\vec{k},\vec{v}}} \sigma^{(\vec{\theta})}_{\vec{y},\vec{d}} \otimes |\vec{y},\vec{d}\rangle\langle\vec{y},\vec{d}| \,. \tag{46}$$

By the same argument as in Remark 4, we can write $\sigma^{(\vec{\theta},\vec{v})}$ for simplicity.

We note that different from Definition 29, we only consider two basis choices $\vec{\theta} = \vec{0}$ or $\vec{\theta} = \vec{1}$, whereas the post-measurement states in Definition 29 can be defined with respect to any basis choice. Similar to Lemma 8, we analyze what outcomes a successful device must produce when measuring the observables from Definition 28 on the post-measurement states from Definition 31.

Lemma 12. *For any efficient device $D = (S, \Pi, M, P)$, a coset state description (A, α, β):*

$$\sum_{\vec{v} \in S_0} \mathrm{Tr}\left[Z_i^{(v_i)} \sigma^{(\vec{0},\vec{v})}\right] \approx_{\gamma_H(D_{\mathsf{coset}})} 1 \,, \tag{47}$$

$$\sum_{\vec{v} \in S_1} \mathrm{Tr}\left[X_i^{(v_i)} \sigma^{(\vec{1},\vec{v})}\right] \approx_{\gamma_H(D_{\mathsf{coset}})} 1 \,, \tag{48}$$

where $S_0 \coloneqq A + \alpha$ and $S_1 \coloneqq A^\perp + \beta - \hat{u}(\vec{k}, \vec{y}, \vec{d})$.

Proof. We first prove Equation (47). Since the case $q = \theta = 0$ occurs with probability $1/2$ in Protocol 2, the device's failure probability in this case can be at most $2\gamma_H(D_{\mathsf{coset}})$. Furthermore, since the device only succeeds if $v_i = \hat{b}(k_i, y_i)$ and $\vec{v} \in A + \alpha$ for all $i \in [\![1, n]\!]$ in the protocol, it means that the device succeeds with probability at least $1 - 2\gamma_H(D)$. Now comparing the definition of $\sigma^{(\vec{0},\vec{v})}$ with the verifier's checks in the protocol, this means that for all $i \in [\![1, n]\!]$:

$$\sum_{v \in S_0} \mathrm{Tr}\left[Z_i^{(v_i)} \sigma^{(\vec{0},\vec{v})}\right] \geq 1 - 2\gamma_H(D) \,.$$

For the inequality in the other direction, we note that since $Z_i^{(v_i)}$ is a projector, we immediately have

$$\sum_{\vec{v} \in S_0} \mathrm{Tr}\left[Z_i^{(v_i)} \sigma^{(\vec{0},\vec{v})}\right] \leq \sum_{\vec{v} \in S_0} \mathrm{Tr}\left[\sigma^{(\vec{0},\vec{v})}\right] = \mathrm{Tr}\left[\sigma^{(\vec{0})}\right] = 1,$$

finishing the proof of Equation (47).

The proof of Equation (48) is completely analogous, combining with the fact that if $\vec{v} + \hat{u}(\vec{k}, \vec{y}, \vec{d}) \in A^\perp + \beta$ iff $\vec{v} \in A^\perp + \beta - \hat{u}(\vec{k}, \vec{y}, \vec{d})$. □

Step 2: Relating Protocol 1 and Protocol 2. We relate the prover's operators and states in Protocol 1 and Protocol 2 by the following lemmas.

Lemma 13. *For any efficient devices D, D' with the notation given in Definition 24. Assume that D is a device of Protocol 1 with corresponding states $(\psi^{(\vec{\theta})}, \sigma^{(\vec{\theta})})$ and D' is a device of Protocol 2 with corresponding states $(\psi'^{(\vec{\theta}')}, \sigma'^{(\vec{\theta}')})$. Then*

$$\psi^{(\vec{\theta})} \overset{c}{\approx}_0 \psi'^{(\vec{\theta}')}\,, \tag{49}$$

and

$$\sigma^{(\vec{\theta})} \overset{c}{\approx}_0 \sigma^{(\vec{\theta}')}\,. \tag{50}$$

Proof. At the beginning of each protocol's execution: in Protocol 1, the device's state is (encrypted) BB84 states, while in Protocol 2, the device's state is (encrypted) coset states. Furthermore, note that executing Protocol 1 or Protocol 2 does not require the secret key of the QFHE encryption scheme. Equation (49) then follows directly from semantic security of the QFHE encryption scheme.

In Protocol 2, the verifier never sends a "pre-image round" challenge. In Protocol 1, the round type is chosen uniformly at random, so with probability $\frac{1}{2}$, the round type is "Hadamard round". In this case, the execution of two protocols are identical from the prover's point of view. Since the prover is efficient, Equation (50) also follows. □

We then obtain the following relation between the success probabilities of devices in Protocol 1 and Protocol 2.

Corollary 3. *For any efficient device $D \coloneqq (S, \Pi, M, P)$:*

$$\gamma_H(D_{\texttt{bb84}}) \overset{c}{\approx}_0 2\gamma_H(D_{\texttt{coset}}). \tag{51}$$

Remark 5. Due to the relation in Equation (51) and the definition of the "$\approx$"-notation (Definition 6), from now on, we drop the subscript and simply write $\gamma_H(D)$ when it is clear from the context.

Combining Corollary 3 and Lemma 13, using the same isometry V defined in Definition 30, we can "lift" the approximate-equality relations described in Lemma 11 for an efficient device in Protocol 1 to an efficient device in Protocol 2.

Lemma 14. *For an efficient perfect device $D = (S, \Pi, M, P)$ in Protocol 2 and any $\vec{a}, \vec{b} \in \{0,1\}^n$ we have*

$$VX(\vec{a})Z(\vec{b})V^\dagger \approx_{n^{1/8}\gamma_H(D)^{1/32}, V\sigma^{(\vec{1})}V^\dagger} \left(\sigma_X(\vec{a})\sigma_Z(\vec{b})\right)_Q \otimes \mathcal{I}_{YRDA}\,. \tag{52}$$

Proof. The lemma follows directly from the lifting lemma (Item 6 of Lemma 3) and the fact that the isometry V and the operators X, Z are efficient. Using the notation from Lemma 3, we have $\delta = 0$, $\varepsilon = n^{1/2}\gamma_H(D)^{1/8}$, the isometry is V, the observable A is $X(\vec{a})Z(\vec{b})$, the observable B is $\sigma_X(\vec{a})\sigma_z(\vec{b}) \otimes \mathcal{I}$. The two states are $V\sigma'^{(\vec{1})}V^\dagger$ of a device in Protocol 1 and $V\sigma^{(\vec{1})}V^\dagger$ of a device in Protocol 2. □

Step 3: Rigidity. We first prove the following technical lemma.

Lemma 15. *For an efficient device $D = (S, \Pi, M, P)$, a coset state description (A, α, β):*

$$\sum_{\vec{v}\in S_0} |\vec{v}\rangle\langle\vec{v}| \otimes (\sigma_{Z,i}^{(v_i)})_Q \approx_{\varepsilon, \sum_{\vec{v}'\in S_0} |\vec{v}'\rangle\langle\vec{v}'|\otimes V\sigma^{\vec{0},\vec{v}'}V^\dagger} \mathcal{I}\,, \tag{53}$$

$$\sum_{\vec{v}\in S_1} |\vec{v}\rangle\langle\vec{v}| \otimes (\sigma_{X,i}^{(v_i)})_Q \approx_{\varepsilon, \sum_{\vec{v}'\in S_1} |\vec{v}'\rangle\langle\vec{v}'|\otimes V\sigma^{\vec{1},\vec{v}'}V^\dagger} \mathcal{I}\,, \tag{54}$$

where $S_0 = A + \alpha$, $S_1 = A^\perp + \beta - \hat{u}(\vec{k}, \vec{y}, \vec{d})$ and the approximation factor ε will be clarified later in the proof.

Proof. We first prove the first statement. It is easy to check that $\sum_{\vec{v}\in V} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{Z,i}^{(v_i)}\right)_Q$ is a projector, so we can expand the definition of the state-dependent distance and compute:

$$\begin{aligned}
&\mathrm{Tr}\left[\left(\sum_{\vec{v}\in S_0} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{Z,i}^{(v_i)}\right)_Q - \mathcal{I}\right)^\dagger \left(\sum_{\vec{v}\in S_0} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{Z,i}^{(v_i)}\right)_Q - \mathcal{I}\right) \sum_{\vec{v}'\in S_0} |\vec{v}'\rangle\langle\vec{v}'| \otimes V\sigma^{(\vec{0},\vec{v}')}V^\dagger\right] \\
&= \mathrm{Tr}\left[\left(\mathcal{I} - \sum_{\vec{v}\in S_0} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{Z,i}^{(v_i)}\right)_Q\right) \sum_{\vec{v}'\in S_0} |\vec{v}'\rangle\langle\vec{v}'| \otimes V\sigma^{(\vec{0},\vec{v}')}V^\dagger\right] \\
&= 1 - \sum_{\vec{v}\in S_0} \mathrm{Tr}\left[\left(|\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{Z,i}^{(v_i)}\right)_Q\right) \sum_{\vec{v}'\in S_0} |\vec{v}'\rangle\langle\vec{v}'| \otimes V\sigma^{(\vec{0},\vec{v}')}V^\dagger\right] \\
&= 1 - \sum_{\vec{v}\in S_0} \mathrm{Tr}\left[\left(\sigma_{Z,i}^{(v_i)}\right)_Q V\sigma^{(\vec{0},\vec{v})}V^\dagger\right],
\end{aligned}$$

To show the first part of the lemma, we need to show that

$$\sum_{\vec{v}\in S_0} \mathrm{Tr}\left[\left(\sigma_{Z,i}^{(v_i)}\right)_Q V\sigma^{(\vec{0},\vec{v})}V^\dagger\right] \approx_\varepsilon 1\,. \tag{55}$$

For this, recall from Lemma 14 that we have

$$VZ_iV^\dagger \approx_{n^{1/8}\gamma_H(D)^{1/32}, V\sigma^{(\vec{1})}V^\dagger} (\sigma_{Z,i})_Q \otimes \mathcal{I}_{YRDA}\,. \tag{56}$$

For shorthand, write $\gamma := n^{1/8}\gamma_H(D)^{1/32}$. Since V and Z_i are efficient, by the lifting lemma (Lemma 3) and the fact that $\sigma^{(\vec{0})} \overset{c}{\approx}_0 \sigma^{(\vec{1})}$, this implies that:

$$VZ_iV^\dagger \approx_{\gamma^{1/4}, V\sigma^{(\vec{0})}V^\dagger} (\sigma_{Z,i})_Q \otimes \mathcal{I}_{YRDA}\,. \tag{57}$$

Using Lemma 4 and Lemma 5, we get:

$$\sum_{\vec{v}\in S_0} VZ_i^{(v_i)}V^\dagger \approx_{\gamma^{1/4}, \sum_{\vec{v}\in S_0} V\sigma^{(\vec{0},\vec{v})}V^\dagger} \sum_{\vec{v}\in S_0} \left(\sigma_{Z,i}^{(v_i)}\right)_Q \otimes \mathcal{I}_{YRDA}\,. \tag{58}$$

Using the replacement lemma (Lemma 1), we obtain

$$\sum_{\vec{v}\in S_0} \mathrm{Tr}\left[\left(\sigma_{Z,i}^{(v_i)}\right)_Q V\sigma^{(\vec{0},v_i,\vec{1}^i)}V^\dagger\right] \approx_{\gamma^{1/8}} \sum_{\vec{v}\in S_0} \mathrm{Tr}\left[VZ_i^{(v_i)}V^\dagger V\sigma^{(\vec{0},\vec{v})}V^\dagger\right] \tag{59}$$

$$= \sum_{\vec{v}\in S_0} \mathrm{Tr}\left[Z_i^{(v_i)}\sigma^{(\vec{0},\vec{v})}\right] \tag{60}$$

$$\approx_{\gamma_H(D)} 1\,, \tag{61}$$

where the last line follows from Equation (47). Set $\varepsilon := \gamma^{1/8}$, this finishes the proof of the first statement.

For the second statement, we can perform the same calculation, but use Equation (48). □

Lemma 16. *For an efficient perfect device $D = (S, \Pi, M, P)$, a coset state description (A, α, β) and $\vec{\theta} \in \{\vec{0}, \vec{1}\}$, there exists a set of subnormalized states $\{\rho_i^{(\vec{\theta},\vec{v})}\}_{\vec{v}\in S_i}$ where S_i for $i \in \{0,1\}$ are defined as in Lemma 15 such that*

$$\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \approx_{2n\varepsilon} \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left((\mathsf{H}^{\otimes n})^i|\vec{v}\rangle\langle\vec{v}|(\mathsf{H}^{\otimes n})^i\right)_Q \otimes \rho_i^{(\vec{\theta},\vec{v})}\,, \tag{62}$$

where $i = 0$ if $\vec{\theta} = \vec{0}$ and $i = 1$ if $\vec{\theta} = \vec{1}$.

Proof. We define the shorthand

$$M(\theta) = \begin{cases} Z & \text{if } \theta = 0\,, \\ X & \text{if } \theta = 1\,. \end{cases}$$

Applying Lemma 15 and Lemma 2 to get

$$\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes V\sigma^{(\vec{\theta},\vec{v})}V^\dagger$$

$$\approx_\varepsilon \left(\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{M(\theta_1),1}^{(v_1)}\right)_Q\right) \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \left(\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma_{M(\theta_1),1}^{(v_1)}\right)_Q\right)$$

We repeat this for the remaining indices $j = 2, \ldots, n$. Since there are in total n steps, the total approximation error will be $n\varepsilon$. We then have

$$\begin{aligned}
&\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \\
&\approx_{n\varepsilon} \left(\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma^{(v_1)}_{M(\theta_1),1}\right)_Q\right) \cdots \left(\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma^{(v_n)}_{M(\theta_n),n}\right)_Q\right) \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \\
&\qquad\qquad \left(\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma^{(v_1)}_{M(\theta_1),1}\right)_Q\right) \cdots \left(\sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\sigma^{(v_n)}_{M(\theta_n),n}\right)_Q\right) \\
&= \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left(\prod_j \sigma^{(v_j)}_{M(\theta_j),j}\right)_Q V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \left(\prod_j \sigma^{(v_j)}_{M(\theta_j),j}\right)_Q .
\end{aligned}$$

Now noting that $\prod_j \sigma^{(v_j)}_{M(\theta_j),j} = (\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i$, we obtain

$$\begin{aligned}
&= \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left((\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i\right)_Q \otimes \left(\langle v| (\mathsf{H}^{\otimes n})^i\right)_Q V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \left((\mathsf{H}^{\otimes n})^i |v\rangle\right)_Q \\
&= \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left((\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i\right)_Q \\
&\qquad \otimes \mathrm{Tr}_Q\left[\left((\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i\right)_Q V\sigma^{(\vec{\theta},\vec{v})}V^\dagger \left((\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i\right)_Q\right]
\end{aligned}$$

Analogously to how we added the factors $\prod_j \sigma^{(v_j)}_{M(\theta_j),j}$ in a previous step, we can now replace the factors $\left((\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i\right)_Q$ inside the partial trace by identity, resulting in

$$\approx_{2n\varepsilon} \sum_{\vec{v}\in S_i} |\vec{v}\rangle\langle\vec{v}| \otimes \left((\mathsf{H}^{\otimes n})^i |\vec{v}\rangle\langle\vec{v}| (\mathsf{H}^{\otimes n})^i\right)_Q \otimes \mathrm{Tr}_Q\left[V\sigma^{(\vec{\theta},\vec{v})}V^\dagger\right] .$$

We then obtain the desired statement by defining

$$\rho_i^{(\vec{\theta},\vec{v})} := \mathrm{Tr}_Q\left[V\sigma^{(\vec{\theta},\vec{v})}V^\dagger\right] , \tag{63}$$

with $i = 0$ if $\vec{\theta} = \vec{0}$ and $i = 1$ if $\vec{\theta} = \vec{1}$. □

What Lemma 16 says is that up to an isometry, with inverse polynomial error, the device's state must be (information-theoretically) close to a mixed state of vectors in S_i, tensored with an auxiliary state $\rho_i^{(\vec{\theta},\vec{v})}$. We note that it is not hard to show that $\rho_0^{(\vec{0},\vec{v})} \overset{c}{\approx}_0 \rho_1^{(\vec{1},\vec{v})}$. (Though it is not necessary for our soundness proof.)

Furthermore, from the statement of Lemma 16, for a fixed efficient device D, if we run Protocol 2 "coherently" in superposition, then

(i) when $\vec{\theta} = \vec{0}$, the device's state must be in superposition of all vectors in S_0, that is $|A + \alpha\rangle$,

(ii) when $\vec{\theta} = \vec{0}$, the device's state must be in superposition of all vectors in S_1. By applying a correction (XOR-ing the register Q with $\hat{u}(\vec{k}, \vec{y}, \vec{d})$), the state would be $|A^{\perp} + \beta\rangle$.

Thus, with the verifier in Protocol 2, we obtain efficient projective measurements to characterize the prover's initial state. Formally, let O_0 be the following process: run Protocol 2 in superposition (without measuring any intermediate messages such as y, d, v) with the basis choice $\vec{\theta} = \vec{1}$ and check if the register Q at the end of the protocol is $|A + \alpha\rangle$. O_1 is defined analogously for $\vec{\theta} = \vec{1}$, and it applies a correction by XORing the register Q with $\hat{u}(\vec{k}, \vec{y}, \vec{d})$ and check if the register Q at the end is $|A^{\perp} + \beta\rangle$. We obtain the main technical lemma.

Lemma 17. *For any efficient device D, the initial state of the device ψ must be close to (up to some inverse polynomial error) $|A_{\alpha,\beta}\rangle \otimes \rho$:*

$$\psi \approx_{4n\varepsilon} |A_{\alpha,\beta}\rangle \otimes \rho. \tag{64}$$

Proof. Let U_0 and U_1 be the efficient unitaries corresponding to operators O_0 and O_1 defined above. Fix a device D. We first apply $U_0\psi$ and record the output to an ancilla register. If the output is 1, apply the inverse $U_0^{\dagger}$ to obtain ψ'. Finally apply $U_1\psi'$. If the output is 1, by the definition of U_i (and O_i), the lemma follows. Note that for each application of U_i, the approximation error is $2n\varepsilon$ which comes from Lemma 16. □

B.2.5 Rigidity Proof of Protocol 3

We are now ready to prove the rigidity of Protocol 3, namely that any efficient quantum prover that does not cause the protocol to abort must have the initial state close to a hidden coset state.

Lemma 18. *For any $\lambda \in \mathbb{N}$, there exist choices $M = \mathsf{poly}(\lambda)$ and $\delta = 1/\mathsf{poly}(\lambda)$ such that if the verifier executes Protocol 3 with an efficient quantum prover whose success probability is lower-bounded by an inverse polynomial, the following holds. Let (A, α, β) the private input of the verifier for the coset instance. Denoting the probability that the protocol does not abort as $\Pr[\top]$, and let ψ the initial state of the prover. Then, with probability $\Pr[\top]$, we have*

$$\psi \overset{c}{\approx}_{\varepsilon} |A_{\alpha,\beta}\rangle \otimes \rho, \tag{65}$$

for some auxiliary state ρ, and the approximation error ε is inverse polynomial on the security parameter λ.

Proof. Essentially, we can see Protocol 3 as a cut-and-choose protocol in which the number of evaluation instances is 1 and the number of check instances is $M^2 - 1$. We then can reduce this lemma to Lemma 17 using the same argument as in [GMP22, Theorem 4.33]. We omit the details. □

Remark 6. We make few comments on the inverse polynomial soundness.[14] First of all, what the soundness lemma (Lemma 18) says is effectively the same as a typical self-testing statement, which is that: if the prover succeeds with probability $1-\varepsilon$ in the protocol, the state it used in the protocol must be, up to an isometry, $\mathsf{poly}(\varepsilon)$-close to ideal (in our setting, the closeness is measured by computational distinguishability rather than trace distance, as in typical self-testing settings). Now, in practice, we would have to estimate ε by doing many runs of the protocol. In particular, we would need about $1/\varepsilon^2$ repetitions to have high (that is, $1-\mathsf{negl}(\lambda)$) confidence that the prover's success probability is $1-\varepsilon$. This implies that if we want ε to be negligible, we would have to do superpolynomial-many repetitions of the protocol and since this is not efficient, we are limited to $\varepsilon = 1/\mathsf{poly}(\lambda)$. It is from doing this $1/\varepsilon^2$ repetitions that we go from the original self-testing statement (Lemma 17) to the statement that characterizes the prover's state in the actual protocol.

We now finish this section with the proof of Proposition 2.

Proof of Proposition 2. Since in the final protocol (Protocol 5), we run N instances over $2N$ possible instances of the self-testing protocol (Protocol 3) (in the cut-and-choose fashion), we can invoke techniques developed in [BF10] to relate quantum sampling to classical sampling and conclude Proposition 2.

In particular, consider the following interaction between a quantum prover $\mathcal{P}$ and a challenger $\mathcal{V}$.

1. $\mathcal{P}$ and $\mathcal{V}$ jointly execute Protocol 5. Let $\overline{T}$ be the set of N indices chosen uniformly at random by $\mathcal{V}$ in N runs of the self-testing protocol.
2. Let X_i be the outcome of each of N runs of the self-testing protocol. $\mathcal{V}$ verifies that $X_i = \mathsf{accept}$ for all $i \in \overline{T}$, and aborts otherwise.

This is a natural quantum analogue of the following classical sampling experiment ([BF10, Example 1]) on a length-$2N$ bitstring X to test if X is close to the all-zero string:

1. randomly select a size-N subset $\overline{T} \subset [\![1, 2N]\!]$,
2. compute $\omega(X|_{\overline{T}})$, and accept if the estimate vanishes and else reject.

Noting that this sample-and-estimate strategy is exactly the Ψ_{uniform} strategy described at the end of Appendix A.3, we have by Corollary 2 that the quantum error probability of this strategy is bounded by $2\exp(\frac{-n\delta^2}{64})$, for $\delta = 1/2$. By the definition of quantum error probability (Definition 8), this means that, with overwhelming probability over $\overline{T}$, the state of the prover $\mathcal{P}$ in the remaining set T also satisfies Equation (64). Indeed, by changing of basis, this reduces to the question of testing if the state of the prover before running the self-testing protocol is close to the all-zero state. Then the quantum sample-and-estimate technique tells us that the state of the prover must be supported on vectors

[14] We thank Alexandru Gheorghiu for providing us this insightful comments.

with relative Hamming distance $< 1/2$, and it means there must be at least 1 bit in string which is 0. If this is the case, it corresponds (up to some inverse polynomial error) to the coset state $|A_{\alpha,\beta}\rangle$ in Equation (64). This completes the proof of the proposition. □

C Single-Decryptor

In this section, we present the definition of single-decryptors, as defined in [CLLZ21]. We also introduce a new security property for single-decryptors, namely anti-piracy security of single-decryptors in the real-or-random style. A variant of semi-quantum single-decryptors will be also introduced.

C.1 Definition

Definition 32 (Single-Decryptor Encryption Scheme [CLLZ21]). *A single-decryptor encryption scheme is a tuple of algorithms* $\mathcal{E} = \langle \mathsf{Setup}, \mathsf{QKeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}\rangle$ *with the following properties:*

- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$. *On input a security parameter* λ*, the classical setup algorithm* Setup *outputs a classical secret key* sk *and a public key* pk*.*
- $(\rho_{\mathsf{sk}}) \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$. *On input a classical secret key* sk*, the quantum key generation algorithm* QKeyGen *outputs a quantum secret key* ρ_{sk}*.*
- $y \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$. *On input a public key* pk*, a message* x *in the message space* $\mathcal{M}$*, the classical encryption algorithm* Encrypt *outputs a classical ciphertext* y*.*
- $x/\bot \leftarrow \mathsf{Decrypt}(\rho_{\mathsf{sk}}, y)$. *On input a quantum secret key* ρ_{sk}*, a classical ciphertext* y*, the quantum decryption algorithm* Decrypt *outputs a classical message* x *or a decryption failure symbol* $\bot$*.*

Correctness. There exists a negligible function $\mathsf{negl}(\cdot)$, such that for all $\lambda \in \mathbb{N}$, for all $x \in \mathcal{M}$, the following holds:

$$\Pr\left[\mathsf{Decrypt}(\rho_{\mathsf{sk}}, y) = x \;\middle|\; \begin{array}{c}(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)\\ \rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})\\ y \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

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 [Wil11].

C.2 Anti-Piracy Game of Single-Decryptor (Real-or-Random Style)

We present below an anti-piracy game of single-decryptors in the real-or-random CPA style, parameterized by a single-decryptor scheme $\mathcal{E} = \langle \mathsf{Setup}, \mathsf{QKeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}\rangle$, a security parameter λ. This game is between a challenger and an adversary represented by three QPT algorithms $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$.

- **Setup phase:**
  - The challenger samples $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$.
  - The challenger samples $\rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$.
  - The challenger sends $(\mathsf{pk}, \rho_{\mathsf{sk}})$ to $\mathcal{A}_0$.
- **Splitting phase:**
  - $\mathcal{A}_0$ prepares a bipartite quantum state σ_{12}.
  - $\mathcal{A}_0$ sends σ_1 to $\mathcal{A}_1$ and σ_2 to $\mathcal{A}_2$.
  - $\mathcal{A}_0$ sends a challenge message m_0 to the challenger.
- **Challenge phase:**
  - $\mathcal{A}_1$ sends a message m_1 to the challenger, and $\mathcal{A}_2$ sends a message m_2 to the challenger.
  - The challenger then generates ciphertexts c_1, c_2 as follows.
    * $c_1 = \mathsf{Encrypt}(\mathsf{pk}, m_0)$ and $c_2 = \mathsf{Encrypt}(\mathsf{pk}, m_2)$ with probability $1/3$. Set $b_1 = 0$ and $b_2 = 1$.
    * $c_1 = \mathsf{Encrypt}(\mathsf{pk}, m_1)$ and $c_2 = \mathsf{Encrypt}(\mathsf{pk}, m_0)$ with probability $1/3$. Set $b_1 = 1$ and $b_2 = 0$.
    * $c_1 = \mathsf{Encrypt}(\mathsf{pk}, m_1)$ and $c_2 = \mathsf{Encrypt}(\mathsf{pk}, m_2)$ with probability $1/3$. Set $b_1 = 1$ and $b_2 = 1$.
  - The challenger sends c_1 to $\mathcal{A}_1$ and c_2 to $\mathcal{A}_2$.
- **Answer phase:**
  - For $i \in \{1, 2\}$: $\mathcal{A}_i$ outputs a bit b'_i.

The adversary wins the game if $\mathcal{A}_1$ and $\mathcal{A}_2$ both make a correct guess, that is $b'_i = b_i$ for $i \in \{1, 2\}$.

We denote the random variable that indicates whether an adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ wins the game or not as $\mathsf{SD\text{-}AP\text{-}RoR}^{\mathcal{E}}_D(1^\lambda, (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2))$.

Definition 33 (Anti-Piracy Security, Real-or-Random style). *A single-decryptor scheme has* anti-piracy security (real-or-random style) *if no* QPT *adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ can win the anti-piracy game (real-or-random style) with a probability significantly greater than $2/3$. More precisely, for any* QPT *adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$*

$$\Pr\left[\mathsf{SD\text{-}AP\text{-}RoR}^{\mathcal{E}}_D(1^\lambda, (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)) = 1\right] \leq 2/3 + \mathsf{negl}(\lambda).$$

We observe that the construction of single-decryptor given in [CLLZ21] also satisfies our definition of anti-piracy in the real-or-random style. For completeness, we recall their construction below.

Construction 3 ([CLLZ21] Single-Decryptor Scheme). *Given a security parameter λ, let $n = \lambda$ and κ be polynomial in λ.*

- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$*:*

- *Sample coset spaces* $\{A_i, s_i, s'_i\}_{i\in[\![1,\kappa]\!]}$ *where each* A_i *is of dimension* $n/2$*;*
- *Construct the membership programs for each coset* $\{\mathsf{R}^0_i, \mathsf{R}^1_i\}_{i\in[\![1,\kappa]\!]}$*;*
- *Return* $\left(\mathsf{sk} \coloneqq \{A_i, s_i, s'_i\}_{i\in[\![1,\kappa]\!]}, \mathsf{pk} \coloneqq \{\mathsf{R}^0_i, \mathsf{R}^1_i\}_{i\in[\![1,\kappa]\!]}\right)$*.*

- $\rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$*:*
  - *Parse* $\mathsf{sk} \leftarrow \{A_i, s_i, s'_i\}_{i\in[\![1,\kappa]\!]}$*;*
  - *Return* $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\kappa]\!]}$*.*
- $c \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m)$*:*
  - *Parse* $\mathsf{pk} \leftarrow \{\mathsf{R}^0_i, \mathsf{R}^1_i\}_{i\in[\![1,\kappa]\!]}$*;*
  - *Sample* $r \xleftarrow{\$} \{0,1\}^\kappa$*;*
  - *Generate an obfuscated program* $i\mathcal{O}(\mathsf{Q}_{m,r})$ *of program* $\mathsf{Q}_{m,r}$ *described in Appendix C.2.*
  - *Return* $c \coloneqq (r, i\mathcal{O}(\mathsf{Q}_{m,r}))$*.*
- $m/\bot \leftarrow \mathsf{Decrypt}(\rho_{\mathsf{sk}}, c)$*:*
  - *Parse* $\rho_{\mathsf{sk}} \leftarrow \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\kappa]\!]}$ *and* $c \leftarrow (r, i\mathcal{O}(\mathsf{Q}_{m,r}))$*;*
  - *For all* $i \in [\![1,\kappa]\!]$*, if* $r_i = 1$*, apply* $\mathsf{H}^{\otimes^n}$ *to* $|A_{i,s_i,s'_i}\rangle$*;*
  - *Let* ρ'_{sk} *be the resulting state, run* $i\mathcal{O}(\mathsf{Q}_{m,r})$ *coherently on* ρ'_{k} *and measure the final register to get* m*;*
  - *Return* m*.*

Hardcoded: Keys $\mathsf{k}_1, \mathsf{k}_2, \mathsf{k}_3$, programs $\mathsf{R}^0_i, \mathsf{R}^1_i$ for all $i \in [\![1,\kappa]\!]$.
On input vectors $u_1, u_2, \ldots, u_\kappa$, do the following:

1. If for all $i \in [\![1,\kappa]\!]$, $\mathsf{R}^{r_i}_i(u_i) = 1$, then output m.
2. Otherwise, output $\bot$.

Fig. 2. Program $\mathsf{Q}_{m,r}$.

C.3 Semi-Quantum Single-Decryptor

Alternatively, in the definition of single-decryptors above, we can combine the Setup and QKeyGen algorithms to be a single interactive protocol with classical communication. The security definition is defined analogously, in which the setup phase is now an interactive setup phase where the challenger obtains the the secret key and the adversary obtains the quantum unclonable secret key. This defines a notion of semi-quantum single-decryptors.

Remark 7. Of course, now if the sender wants to generate a new quantum secret key, it needs to run the interactive protocol again, which effectively also generates a new classical secret key sk and a new classical public key pk. To recover the original setting where there are only one classical secret/public key pair and possibly many quantum secret keys, the sender can use any post-quantum semantic-secure public-key encryption scheme to encrypt the new classical secret key sk generated by the semi-quantum protocol, and send this encryption of sk to the receiver. This encryption of sk will also be included in the ciphertext, which the sender can decrypt using its "master" secret key and perform the original decryption algorithm. We note that for our construction of semi-quantum copy-protection, this is not necessary though.

A construction of semi-quantum single-decryptors is identical to Construction 3, except now we replace the Setup and $\mathsf{QKeyGen}$ algorithms by polynomially many runs of Protocol 5. Security proof of Construction 3 also carries over this semi-quantum setting directly, with only a small change as follows. In the reduction showing that an adversary $\mathcal{A}$ that breaks the anti-piracy game of single-decryptors can be used to construct an adversary $\mathcal{A}'$ breaking the monogamy of entanglement game (defined in Definition 2), $\mathcal{A}'$ simulates the security game for $\mathcal{A}$ (in which $\mathcal{A}'$ runs polynomially many executions of Protocol 5 with $\mathcal{A}$), $\mathcal{A}'$ then picks one execution uniformly at random and lets $\mathcal{A}$ runs the protocol with $\mathcal{A}'$'s challenger. The rest of the reduction is identical as the one given in [CLLZ21], we omit the full details here.

D Proof of Anti-Piracy Security of Construction 1

We present below a security proof for our Construction 1. We will proceed with the proof by doing a reduction between the anti-piracy security of single-decryptor (real-or-random style) of Construction 3 and the anti-piracy security of our copy-protection construction (Construction 1). The security proof of our semi-quantum copy-protection of point functions (Construction 2) is done identically by reducing to the security of the semi-quantum single-decryptor.

Notations. In the proof, we will sometimes parse $x \in \{0,1\}^n$ as (x_0, x_1, x_2) such that $x = x_0 \| x_1 \| x_2$ (where $\|$ is the concatenation operator) and the length of x_i is ℓ_i for $i \in \{0,1,2\}$.

Procedure. We define the $\mathsf{GenTrigger}$ procedure (Figure 3) which, given an input's prefix x_0 and a PRF image y returns a so-called *trigger input* x' that: passes the "Hidden Trigger" condition of the program P.

Trigger's Inputs Lemma. The following lemma follows from [CLLZ21, Lemma 7.17].

Lemma 19. *Assuming post-quantum $i\mathcal{O}$ and one-way functions, any efficient* QPT *algorithm $\mathcal{A}$ cannot win the following game with non-negligible advantage:*

- Given as input $x_0 \in \{0,1\}^{\ell_0}$, $z \in \{0,1\}^m$, $\mathsf{k}_2, \mathsf{k}_3 \in \mathcal{K}_2 \times \mathcal{K}_3$ and cosets $\{A_{i,s_i,s'_i}\}_{i\in[\![1,\ell_0]\!]}$:
- Let Q be the program which, given $v_0, \ldots, v_{\ell_0}$, returns y if $R_i^{x_{0,i}}(v_i) = 1$ for all i or $\perp$ otherwise.
- $x'_1 \leftarrow \mathsf{PRF}_2(\mathsf{k}_2, x_0 \| \mathsf{Q})$;
- $x'_2 \leftarrow \mathsf{PRF}_3(\mathsf{k}_3, x'_1) \oplus (x_0 \| \mathsf{Q})$;
- Return $x_0 \| x'_1 \| x'_2$.

Fig. 3. GenTrigger procedure.

- *A challenger samples* $\mathsf{k}_1 \leftarrow \mathsf{Setup}(1^\lambda)$ *and prepares a quantum key* $\rho_\mathsf{k} := (\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,l_0]\!]}, i\mathcal{O}(\mathsf{P}))$ *(recall that* P *has keys* $\mathsf{k}_1, \mathsf{k}_2, \mathsf{k}_3$ *hardcoded).*
- *The challenger then samples a random input* $x \leftarrow \{0,1\}^n$*. Let* $z \leftarrow \mathsf{PRF}_1(\mathsf{k}_1, x)$*.*
- *The challenger samples challenges* x_1, x_2 *according to the following distribution:*
  - $x_1 := x$ *and* $x_2 \xleftarrow{\$} \{0,1\}^n$ *with probability* $1/3$*;*
  - $x_1 \xleftarrow{\$} \{0,1\}^n$ *and* $x_2 := x$ *with probability* $1/3$*;*
  - $x_1, x_2 \xleftarrow{\$} \{0,1\}^n$ *with probability* $1/3$*;*
- *The challenger computes* $z_i \leftarrow \mathsf{PRF}_1(\mathsf{k}_1, x_i)$*, and parses the inputs* x_i *as* $x_i = x_{i,0}||x_{i,1}||x_{i,2}$ *for* $i \in \{1,2\}$*. Let* $x'_i \leftarrow \mathsf{GenTrigger}(x_{i,0}, z_i, \mathsf{k}_2, \mathsf{k}_3, \{A_j, s_j, s'_j\}_{j\in[\![1,\ell_0]\!]})$ *for* $i \in \{1,2\}$*.*
- *The challenger flips a coin* b*, and sends either* x_1*,* x_2 *or* x'_1*,* x'_2 *to respectively Bob and Charlie, depending on the value of the coin.* $\mathcal{A}$ *wins if it guesses* b *correctly.*

Proof of Anti-Piracy Security.

Proof. We proceed with the proof via a sequence of hybrids. We note the differences between the current hybrid and the previous one in red. For any pair of hybrids (G_i, G_j), we say that G_i is *negligibly close to* G_j if for every QPT adversary $\mathcal{A} := (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, the probability that $\mathcal{A}$ wins G_i is negligibly close to the probability that they win G_j. For the sake of simplicity, we denote the uniform distribution over $\{(0,1),(1,0),(1,1)\}$ as $D_{1/3}$.

Game G_0: This is the piracy game of our copy-protection protocol.

- **Setup phase:** The challenger does the following:
  - Sample ℓ_0 random cosets $\{A_i, s_i, s'_i\}_{i\in[\![1,\ell_0]\!]}$, and prepare the associated coset states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}$ and the obfuscated membership programs $\{(\mathsf{R}_i^0, \mathsf{R}_i^1)\}_{i\in[\![1,\ell_0]\!]}$.
  - Sample $k_i \leftarrow \mathsf{PRF}_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1,2,3\}$.

- Generate the obfuscated program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$.
- Sample $y \xleftarrow{\$} \{0,1\}^n$, compute $z := \mathsf{PRF}_1(\mathsf{k}_1, y)$.
- Send $\rho_y := \left(\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$ to $\mathcal{A}_0$.

- **Splitting phase:** $\mathcal{A}_0$ prepares a bipartite quantum state σ_{12}, then sends σ_1 to $\mathcal{A}_1$ and σ_2 to $\mathcal{A}_2$.
- **Challenge phase:**
  - The challenger samples $(b_1, b_2) \xleftarrow{\$} D_{1/3}$.
  - For $i \in \{1,2\}$:
    - If $b_i = 0$: the challenger sets $x_i := y$.
    - Otherwise, the challenger samples $x_i \xleftarrow{\$} \{0,1\}^n$.
  - The challenge sends x_1 to $\mathcal{A}_1$ and x_2 to $\mathcal{A}_2$.
- **Answer phase:**
  - $\mathcal{A}_1$ returns b'_1 and $\mathcal{A}_2$ returns b'_2.
  - The adversary wins if $b'_1 = b_1$ and $b'_2 = b_2$.

Game G_1: In this game, we replace x_1, x_2 by the trigger inputs. The trigger's inputs lemma (Lemma 19) implies that G_1 is negligibly close to G_0.

- **Setup phase:** The challenger does the following:
  - Sample ℓ_0 random cosets $\{A_i, s_i, s'_i\}_{i\in[\![1,\ell_0]\!]}$, and prepare the associated coset states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}$ and the obfuscated membership programs $\{(\mathsf{R}^0_i, \mathsf{R}^1_i)\}_{i\in[\![1,\ell_0]\!]}$.
  - Sample $k_i \leftarrow \mathsf{PRF}_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1,2,3\}$.
  - Generate the obfuscated program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$.
  - Sample $y \xleftarrow{\$} \{0,1\}^n$, compute $z := \mathsf{PRF}_1(\mathsf{k}_1, y)$.
  - Send $\rho_y := \left(\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$ to $\mathcal{A}_0$.
- **Splitting phase:** $\mathcal{A}_0$ prepares a bipartite quantum state σ_{12}, then sends σ_1 to $\mathcal{A}_1$ and σ_2 to $\mathcal{A}_2$.
- **Challenge phase:**
  - The challenger samples $(b_1, b_2) \xleftarrow{\$} D_{1/3}$.
  - For $i \in \{1,2\}$:
    - If $b_i = 0$: the challenger sets $x_i := y$ and $z_i := z$.
    - Otherwise, the challenger samples $x_i \xleftarrow{\$} \{0,1\}^n$ and computes $z_i \leftarrow \mathsf{PRF}_1(\mathsf{k}_1, x_i)$.
    - In both case, the challenger computes $x'_i \leftarrow \mathsf{GenTrigger}(x_{i,0}, z_i, \mathsf{k}_2, \mathsf{k}_3, \{A_{i,s_i,s'_i}\}_{i\in[\![1,\ell_0]\!]})$.
  - The challenge sends x'_1 to $\mathcal{A}_1$ and x'_2 to $\mathcal{A}_2$.

- **Answer phase:**
  - $\mathcal{A}_1$ returns b'_1 and $\mathcal{A}_2$ returns b'_2.
  - The adversary wins if $b'_1 = b_1$ and $b'_2 = b_2$.

Game G_2: In this game, we replace z, z_1, z_2 by uniformly random strings. Since all the inputs have enough min-entropy $\ell_1 + \ell_2 \geq m + 2\lambda + 4$ and PRF_1 is extracting, the outcomes z, z_1, z_2 are statistically close to independently random outcomes. Thus G_2 is negligibly close to G_1.

- **Setup phase:** The challenger does the following:
  - Sample ℓ_0 random cosets $\{A_i, s_i, s'_i\}_{i\in[\![1,\ell_0]\!]}$, and prepare the associated coset states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}$ and the obfuscated membership programs $\{(\mathsf{R}^0_i, \mathsf{R}^1_i)\}_{i\in[\![1,\ell_0]\!]}$.
  - Sample $k_i \leftarrow \mathsf{PRF}_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1, 2, 3\}$.
  - Generate the obfuscated program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$.
  - Sample $y \xleftarrow{\$} \{0,1\}^n$, sample $z \xleftarrow{\$} \{0,1\}^m$.
  - Send $\rho_y := \left(\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$ to $\mathcal{A}_0$.
- **Splitting phase:** $\mathcal{A}_0$ prepares a bipartite quantum state σ_{12}, then sends σ_1 to $\mathcal{A}_1$ and σ_2 to $\mathcal{A}_2$.
- **Challenge phase:**
  - The challenger samples $(b_1, b_2) \xleftarrow{\$} D_{1/3}$.
  - For $i \in \{1, 2\}$:
    - If $b_i = 0$: the challenger sets $x_i := y$ and $z_i := z$.
    - Otherwise, the challenger samples $x_i \xleftarrow{\$} \{0,1\}^n$ and samples $z_i \xleftarrow{\$} \{0,1\}^m$.
    - In both case, the challenger computes $x'_i \leftarrow \mathsf{GenTrigger}(x_{i,0}, z_i, \mathsf{k}_2, \mathsf{k}_3, \{A_{i,s_i,s'_i}\}_{i\in[\![1,\ell_0]\!]})$.
  - The challenge sends x'_1 to $\mathcal{A}_1$ and x'_2 to $\mathcal{A}_2$.
- **Answer phase:**
  - $\mathcal{A}_1$ returns b'_1 and $\mathcal{A}_2$ returns b'_2.
  - The adversary wins if $b'_1 = b_1$ and $b'_2 = b_2$.

Game G_3: This game has exactly the same distribution as that of G_2. We only change the order in which some values are sampled, and recognize that certain procedures become identical to encryptions in the single-decryptor encryption scheme $\langle \mathsf{SD.Setup}, \mathsf{SD.QKeyGen}, \mathsf{SD.Encrypt}, \mathsf{SD.Decrypt}\rangle$ from Appendix C. Thus, the probability of winning in G_3 is the same as in G_2.

- **Setup phase:** The challenger does the following:
  - Run $\mathsf{SD.Setup}(1^\lambda)$ to obtain ℓ_0 random cosets $\{A_i, s_i, s'_i\}_{i\in[\![1,\ell_0]\!]}$, the associated coset states $\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}$ and the obfuscated membership programs $\{(\mathsf{R}^0_i, \mathsf{R}^1_i)\}_{i\in[\![1,\ell_0]\!]}$. Let $\rho_{\mathsf{sk}} := \{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}$.

- Sample $k_i \leftarrow \mathsf{PRF}_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1,2,3\}$.
- Generate the obfuscated program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$.
- Sample $y \xleftarrow{\$} \{0,1\}^n$, sample $z \xleftarrow{\$} \{0,1\}^m$.
- Send $\rho_y := \left(\{|A_{i,s_i,s'_i}\rangle\}_{i\in[\![1,\ell_0]\!]}, \hat{\mathsf{P}}, z\right)$ to $\mathcal{A}_0$.

- **Splitting phase:** $\mathcal{A}_0$ prepares a bipartite quantum state σ_{12}, then sends σ_1 to $\mathcal{A}_1$ and σ_2 to $\mathcal{A}_2$.
- **Challenge phase**
  - The challenger samples $(b_1, b_2) \xleftarrow{\$} D_{1/3}$.
  - For $i \in \{1,2\}$:
    - If $b_i = 0$: the challenger sets $z_i := z$.
    - Otherwise, the challenger samples $z_i \xleftarrow{\$} \{0,1\}^m$.
    - In both case, the challenger computes $(x_i, Q) \leftarrow \mathsf{SD.Encrypt}(\mathsf{pk}, z_i)$.
    - Finally, the challenger computes x'_i as in $\mathsf{GenTrigger}$ using $x_{i,0}$ and Q.
  - The challenge sends x'_1 to $\mathcal{A}_1$ and x'_2 to $\mathcal{A}_2$.
- **Answer phase**
  - $\mathcal{A}_1$ returns b'_1 and $\mathcal{A}_2$ returns b'_2.
  - The adversary wins if $b'_1 = b_1$ and $b'_2 = b_2$.

Reduction to Single-Decryptor's Piracy Game. Assume that there exists a QPT adversary $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ who wins the last hybrid G_3 with advantage δ. We construct an adversary $(\mathcal{B}_0, \mathcal{B}_1, \mathcal{B}_2)$ who wins the piracy game of the Single-Decryptor scheme with the same advantage δ.

- **Setup phase:** The challenger samples $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda)$, then samples $\rho_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(\mathsf{sk})$ and finally sends $(\mathsf{pk}, \rho_{\mathsf{sk}})$ to $\mathcal{B}_0$.
- **Splitting phase:**
  - $\mathcal{B}_0$ samples $\mathsf{k}_i \leftarrow \mathsf{PRF}_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1,2,3\}$ and use these keys and pk to prepare the obfuscated program $\hat{\mathsf{P}} \leftarrow i\mathcal{O}(\mathsf{P})$;
  - Then $\mathcal{B}_0$ samples $z \xleftarrow{\$} \{0,1\}^m$ and runs $\mathcal{A}_0$ on $(\rho_{\mathsf{sk}}, \hat{\mathsf{P}}, z)$ to get σ_{12};
  - Finally, $\mathcal{B}_0$ sends $\sigma'_1 := (\sigma_1, \mathsf{k}_2, \mathsf{k}_3)$ to $\mathcal{B}_1$, $\sigma'_2 := (\sigma_2, \mathsf{k}_2, \mathsf{k}_3)$ to $\mathcal{B}_2$ and z as the challenge message to its challenger.
- **Challenge phase:** For $i \in \{1,2\}$:
  - $\mathcal{B}_i$ samples $z_i \xleftarrow{\$} \{0,1\}^m$ and sends z_i as the challenge message to its challenger. Upon receiving the challenge ciphertext c_i, $\mathcal{B}_i$ parses c_i as $(x_{i,0}, Q)$, then prepares $x'_i := (x_{i,0}||x'_{i,1}||x'_{i,2})$ as in $\mathsf{GenTrigger}$: $x'_{i,1} := \mathsf{PRF}_2(\mathsf{k}_2, x_{i,0}||Q)$ and $x'_{i,2} := \mathsf{PRF}_3(\mathsf{k}_3, x'_{i,1}) \oplus (x_{i,0}||Q)$.
- **Answer phase:** For $i \in \{1,2\}$:

– $\mathcal{B}_i$ runs $\mathcal{A}_i$ on (σ_i, x'_i) and returns the outcome b'_i;

The adversary $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1, \mathcal{B}_2)$ perfectly simulates $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, and thus $\mathcal{B}$ breaks the anti-piracy security of the single-decryptor scheme with the same probability δ, which completes the proof. □