Candidate Obfuscation of Null Quantum Circuits and Witness Encryption for QMA

James Bartusek[1] and Giulio Malavolta[2]

[1]University of California, Berkeley
[2]Max Planck Institute for Security and Privacy

Abstract

We present a construction of indistinguishability obfuscation for null quantum circuits (null-iO) with respect to a classical oracle, assuming the quantum hardness of the learning with errors (LWE) problem. Heuristically instantiating the classical oracle with quantum-secure indistinguishability obfuscation for classical circuits gives us the first candidate construction of null-iO for quantum circuits.

This scheme establishes the feasibility of a series of new cryptographic primitives that, prior to our work, were unknown to exist even making heuristic assumptions. Specifically, we obtain (in some cases additionally assuming indistinguishability obfuscation for classical circuits):

- A witness encryption (WE) scheme for QMA.
- A publicly-verifiable non-interactive zero-knowledge (NIZK) argument for QMA.
- A two-message publicly-verifiable witness-indistinguishable (ZAPR) argument for QMA.
- An attribute-based encryption (ABE) scheme for BQP.
- A secret sharing scheme for monotone QMA.

1 Introduction

The goal of program obfuscation [Had00, BGI+01] is to convert an arbitrary circuit C into an unintelligible but functionally equivalent circuit $\widetilde{C}$. Recent work has shown that program obfuscation enables a series of new remarkable applications (e.g. [GGH+13, SW14, GGG+14, BPR15]), establishing obfuscation as a central object in cryptography.

Yet, the scope of obfuscation has so far been concerned almost exclusively with classical cryptography. The advent of quantum computing has motivated researchers [Aar05, AF16] to ask whether program obfuscation is a meaningful notion also in a quantum world:

Can we obfuscate quantum circuits? Is this notion useful at all?

Unfortunately, results on the matter are largely negative [AF16, AP20, ABDS20], barring a few schemes for restricted function classes of questionable usefulness [BK20, AJJ14]. At present, it is unclear whether obfuscation of quantum circuits in its most general form can exist at all. The goal of this work is to make progress on this question.

1.1 Our Results

In this work, we study the notion of obfuscation for quantum circuits. Our contributions are twofold.

(1) Quantum Null-iO and Witness Encryption for QMA. We show that, assuming the quantum hardness of the learning with errors (LWE) problem, there exists an obfuscation scheme for null quantum circuits (i.e. quantum circuits that almost always reject), relative to a *classical oracle*.

Theorem 1.1 (Informal)**.** *Assuming the quantum hardness of the LWE problem, there exists a classical oracle relative to which there exists a null-iO scheme for quantum circuits.*

The oracle that we consider is entirely classical, but queriable in superposition. Heuristically instantiating such an oracle with post-quantum indistinguishability obfuscation (e.g. [BDGM20b, GP20, WW20]) gives us the first plausible construction of null-iO for quantum circuits.

Applying a well-known transformation, we obtain witness encryption for QMA as a corollary. Somewhat surprisingly, our scheme has an entirely classical encryption algorithm: In our witness encryption scheme [GGSW13], any classical user can encrypt a message m with respect to the membership of some statement x in a language $\mathcal{L} \in \mathsf{QMA}$. The message m can be (quantumly) decrypted by anyone possessing (multiple copies of) a valid witness $|\psi\rangle \in \mathsf{R}_{\mathcal{L}}(x)$.

(2) New Applications via Classical Synthesis of Quantum Programs. We show that witness encryption for QMA with classical encryption enables a series of new cryptographic primitives, thereby positioning witness encryption for QMA as a central catalyst in quantum cryptography. Most of our results are obtained via *classical synthesis of quantum programs*: We compress an exponential number of quantum programs into a small classical circuit via the use of classical indistinguishability obfuscation (iO).

We give an overview of the implications of our results below. We remark that, prior to our work, we did not even have a heuristic candidate (e.g. a scheme in the random oracle model) for any of the primitives that we obtain.

- **NIZK for QMA:** We present the first construction of NIZK [BFM88] for QMA. A (quantum) prover can efficiently produce a zero-knowledge certificate π that a certain statement $x \in \mathcal{L}$, where $\mathcal{L}$ is any language in QMA. This certificate is publicly verifiable with respect to a publicly-known common reference string. Prior to our work, all non-interactive proof systems for QMA [BG19, CVZ20, ACGH20, CCY20] were in the *secret parameters model* or the *designated verifier* setting, i.e. the verifier (or prover) needed some secret information not accessible to the other party.

  This gives a candidate solution for a long standing open problem in the area (see [Shm20] for a discussion on the barriers to achieving public verifiability). In addition, our NIZK scheme satisfies several properties of interest, namely, (i) it is statistically zero-knowledge, (ii) the verification algorithm (and CRS) is fully classical, and (iii) the verification algorithm is *succinct*, i.e. its runtime is independent of the size of the witness. In fact, we obtain the first succinct non-interactive argument (zk-SNARG) for QMA.

- **ZAPR for QMA:** We show how to transform our NIZK for QMA scheme into a (publicly verifiable) two-round statistically witness-indistinguishable argument (ZAPR) for QMA. The transformation additionally assumes NIWIs [BOV03], statistical ZAPs for NP, and a special notion of statistically hiding commitment that can be constructed from the sub-exponential hardness of LWE. Our approach is generic and can be thought of as a quantum analogue of the Dwork-Naor compiler [DN00], in the setting of computational soundness.

- **ABE for BQP:** Assuming witness encryption for QMA[1] and sub-exponentially secure classical iO (along with other standard cryptographic primitives) we obtain a ciphertext-policy ABE [SW05, GPSW06] scheme for BQP (bounded-error quantum polynomial-time) computation. In ciphertext-policy ABE for BQP, anyone can encrypt a message m with respect to some BQP language, represented by a quantum circuit Q. The key authority can generate decryption keys associated with any attribute x. The

[1] Actually, for this application we only need WE for BQP (with classical encryption).

ciphertext can then be decrypted if and only if evaluating Q on x produces 1 (which, by QMA amplification, can happen with either overwhelming probability or negligible probability). Interestingly, all algorithms except for the decryption circuit are fully classical.

The scheme satisfies the standard notion of payload-hiding. That is, the message m is hidden, though the policy Q is revealed by the ciphertext. We then show that we can upgrade the security of the scheme via a generic transformation to predicate-encryption security [GVW15] (i.e. the policy Q is hidden from the evaluator if they are only in possession of keys for rejecting attributes). We achieve this via a construction of lockable obfuscation [GKW17, WZ17] for quantum circuits from LWE.

This is the first example of an ABE scheme for functionalities beyond classical computations.

- **Secret Sharing for Monotone QMA:** Finally, as a direct application of our witness encryption scheme, we show how to construct a secret sharing scheme for access structures in monotone QMA.

1.2 Technical Overview

We give a cursory overview of the techniques introduced by our work and we provide some informal intuition on how we achieve our results.

How to Obfuscate Quantum Circuits? Before delving into the specifics of our approach, we highlight a few reasons why known techniques for obfuscating classical circuits do not seem to be directly portable to the quantum setting. For obvious reasons, we restrict this discussion to schemes that plausibly retain security in the presence of quantum adversaries. Recent proposals [BDGM20b, GP20, WW20] follow the *split fully homomorphic encryption* (split-FHE) approach [BDGM20a], which we loosely recall here. The obfuscator computes

$$\mathsf{FHE}(C) \xrightarrow{\mathsf{Eval}(x,\cdot)} \mathsf{FHE}(C(x)) \xrightarrow{\mathsf{Hint}(\mathsf{sk},\cdot)} h$$

where the decryption hint h is specific to the ciphertext encoding $C(x)$. The obfuscated circuit consists of $(\mathsf{FHE}(C), h)$ and the evaluator can recompute the homomorphic evaluation and use the decryption hint h to recover $C(x)$. It turns out that, as long as $|h| \ll |C(x)|$, this primitive alone is enough to build full-fledged obfuscation. One crucial aspect of this paradigm is that the split-FHE evaluation algorithm is deterministic, which allows the obfuscator and the evaluator to converge to the exact same ciphertext.

Translating this approach to the quantum setting seems to get stuck at a very fundamental level: known FHE schemes for quantum circuits [Mah18a] have an inherently randomized homomorphic evaluation algorithm. This means that the evaluator and the obfuscator would most likely end up with a different ciphertext even though they are computing the same function. This makes the hint h (which is ciphertext-specific) completely useless to recover the output.

Reducing to Classical Obfuscation. As a direct approach seems to be out of reach of current techniques, in this work we take a different route. Following the template of [GGH$^+$13], our high-level idea is to outsource the quantum computation to some *untrusted* component of the scheme and instead obfuscate only the circuit that *verifies* that the computation was carried out correctly. Let us for the moment forego the privacy of the circuit: Then we could simply run the quantum computation in plain and provide the output to the obfuscated verifier, together with a proof that the output is indeed the result of the quantum computation. The obfuscated verifier would then verify the proof and return the given output.

The important point here is that verifying the correctness of quantum computation is a much easier task than performing the computation itself. If this verification can be done by an entirely classical algorithm, then classical obfuscation could suffice for our goals.

Blind Classical Verification of Quantum Computation (CVQC). Our attention immediately turns to recent advancements in classical verification of quantum computation (CVQC) protocols. In a recent breakthrough result [Mah18b], it was shown that the validity of any BQP computation can be verified by a completely classical algorithm, assuming the quantum hardness of the LWE problem. Furthermore, recent works [ACGH20, CCY20] have shown that the protocol can be collapsed to two rounds (in the random oracle model): On input a quantum circuit Q, the verifier produces some public parameters pp, which can be used by the prover (holding a quantum state $|\psi\rangle$) to compute a classical proof π. The verifier can then locally verify π using some secret information which was sampled together with pp.

Using these protocols without any additional modification would allow us to implement the scheme as outlined above. However, we would not get any meaningful notion of privacy for the obfuscated circuit, since the prover needs to evaluate the circuit Q in plain. Our first step is then to turn these protocol *blind*: The prover is able to prove that $Q(|\psi\rangle) = y$ obliviously, without knowing Q. This can be done in a canonical way, using fully homomorphic encryption for quantum circuits with classical keys [Mah18a, Bra18].

Null-iO and Witness Encryption for QMA. Although everything seems to fall in place, there is a subtle aspect that makes our attempt not sound: We implicitly assumed that the CVQC protocol is *resettably secure*. If the prover is given access to a circuit implementing a (obfuscated) verifier, nothing prevents it from rewinding it in an attempt to extract the verifier's secret. Once this is leaked, the prover can fool the verifier into accepting false statements, ultimately learning some information about the obfuscated circuit. This is not only a theoretical concern, instead one can show concrete attacks against all known CVQC protocols (more discussion on this later).

While this class of attacks seems to be hard to prevent in general, we observe that, if we restrict our attention to null (i.e. always rejecting) circuits, then this concern disappears. This is not a coincidence: Any two-round CVQC protocol that is one-time sound is automatically many-time sound for reject-only circuits, since it is easy to simulate the responses of the verifier (always reject). We now have all the ingredients needed to construct our scheme: Combining a blind CVQC with classical obfuscation, we obtain a candidate construction of null-iO for quantum circuits. In our analysis we model the obfuscation of the classical verifier as an oracle, and we show that the protocol is still secure even if the prover is allowed to query such an oracle in superposition. We remark that one could also show security assuming post-quantum virtual black-box (VBB) obfuscation of a specific classical circuit, but we present the result with respect to a classical oracle (equivalently, assuming ideal obfuscation of classical circuits) for simplicity. Finally, applying a known transformation, we obtain a witness encryption scheme for QMA as an immediate corollary.

NIZK for QMA. Next, we explore some applications of the resulting witness encryption scheme. Our first result is a construction of NIZK arguments for QMA with public verifiability. To build up some intuition about the protocol, consider the simplified setting where we have a single fixed statement x. We can then define the common reference string to be

$$(\mathsf{vk}, \mathsf{WE.Enc}(x, \sigma)) \text{ such that } \mathsf{Verify}(\mathsf{vk}, \sigma, x) = 1$$

where vk is a verification key of a signature scheme. Anyone with a valid witness $|\psi\rangle$ for x can recover the signature by decrypting the ciphertext. This signature can then be publicly verified. To extend this approach to an exponential number of statements, we exploit the fact that our witness encryption scheme has a *completely classical* encryption procedure. Our idea is to place in the common reference string a classically obfuscated circuit that synthesizes exponentially-many ciphertexts (one for each statement x) encrypting the signature σ_x. This high-level approach can be translated into a provably secure construction using techniques from the obfuscation literature [SW14].

ZAPR for QMA. The next question that we ask is whether we can reduce the trust on the setup and obtain some meaningful guarantees also in the presence of a maliciously generated common reference string. Known generic transformations [DN00] do not apply to our case, since the NIZK that we obtain

is an argument, i.e. it has computational soundness. Our saving grace is again the fact that our NIZK has completely classical setup and verification procedures. This allows us to leverage powerful tools from the literature of (classical) zero-knowledge. We adopt a dual-track approach (reminiscent of the Naor-Yung [NY90] paradigm) where we define the setup to sample two copies $(\mathsf{crs}_0, \mathsf{crs}_1)$, the image of a one-way function y, and a *classical* non-interactive witness indistinguishable (NIWI) proof that either crs_0 or crs_1 is correctly generated.

At this point, it is still unclear whether we have any privacy guarantee, since one of the two strings can be maliciously generated and can therefore leak some information about the witness, if naively used by the prover. Thus, instead of having the prover directly compute the proof, we have it compute another NIWI for the statement

$$\left\{ \exists\, (\pi_0, \pi_1, z) \text{ such that: } \begin{array}{l} \mathsf{Verify}(\mathsf{crs}_0, \pi_0, x) = 1 \text{ OR} \\ \mathsf{Verify}(\mathsf{crs}_1, \pi_1, x) = 1 \text{ OR} \\ \mathsf{OWF}(z) = y. \end{array} \right\}$$

By the witness indistinguishbility of the NIWI, the verifier cannot distinguish whether the prover inverted the one-way function or possesses a valid proof. On the other hand, a valid proof is always guaranteed to exist for either crs_0 or crs_1, by the soundness of the NIWI. In the actual scheme, we work a little harder to ensure statistical witness indistinguishability, which can be achieved using recent results on statistical ZAPs [GJJM20, BFJ+20].

ABE for BQP. We next show how our witness encryption scheme for QMA yields the first ABE scheme for quantum functionalities. Consider the simplified setting where the key authority issues a single key for a fixed attribute x. The witness encryption suggests a natural ABE encryption procedure: $\mathsf{WE.Enc}((Q, x), m)$, where the statement (Q, x) returns 1 if and only if $Q(x) = 1$.[2] Now the challenge is to define an encryption algorithm that does not need to take the attribute x as an input.

Our previous idea is useful again: Instead of publishing the WE directly, the encrypter obfuscates the circuit that, on input x, returns a WE with respect to the statement (Q, x). Recall that the witness encryption scheme has a completely classical encryption algorithm, which allows us to use classical obfuscation for this task. The only remaining issue is that the decrypter can query the obfuscated circuit on any attribute x, whereas we would like to constrain the queries to authorized attributes only. This can be routinely done by letting the key authority issue a signature σ_x and modifying the obfuscated circuit to verify this signature before outputting the corresponding ciphertext. Some additional work is needed in order to obtain a provably secure scheme, but the main ideas are already present in this outline.

The scheme as described so far does not hide the circuit Q, i.e. it only satisfies the notion of payload hiding. We show a generic compiler that transforms any quantum ABE with payload-hiding security into one with predicate encryption security, i.e. where the circuit Q is hidden to the holders of keys for rejecting attributes. This result is obtained by introducing the notion of *quantum lockable obfuscation* and presenting a construction under LWE.

Secret Sharing for Monotone QMA. It is well-known that witness encryption for NP implies the existence of a secret sharing scheme for monotone NP [KNY14]. The high-level idea is to assign to each party P_i the opening of a perfectly binding commitment $c_i = \mathsf{Com}(i; r_i)$ encoding the index corresponding to the party. Then one can publish a witness encryption for the statement

$$\left\{ \exists (I \subseteq P, r_1, \ldots, r_{|I|}) \text{ such that: } I \in \mathcal{L} \text{ AND } \forall i \in I : c_i = \mathsf{Com}(i; r_i) \right\}$$

where I, parsed as a binary string, forms a statement in a NP-complete language $\mathcal{L}$ with witness w. It is not hard to show that decrypting the witness encryption (i.e. reconstructing the secret) can only be done by an authorized set of parties holding the witness w. We show that this construction naturally generalizes to the QMA setting, when given a witness encryption scheme for QMA.

[2] Technically, this is a BQP statement, but a witness encryption for QMA is also a witness encryption for BQP.

2 Discussion and Open Problems

In this section, we discuss two clear open problems that are suggested by this work. We identify barriers to making progress on each problem with our current approach.

2.1 Obfuscation Beyond Null Circuits

In this work, we only consider obfuscating the CVQC verification circuit in the setting where each instance the prover can query will be rejecting (with high probability). One could also consider obfuscating the verification circuit in the setting where the prover can query on an accepting instance, which would help in constructing full-fledged iO for quantum circuits. However, we argue here that such an approach will not be secure, given current constructions of CVQC.

Consider the two-message CVQC of [Mah18b, CCY20, ACGH20]. The first message $\mathsf{pk}_1, \ldots, \mathsf{pk}_k$ essentially commits to a string $x \in \{0,1\}^k$ where each bit x_i determines whether the i'th qubit of the prover's committed state will be measured in the computational basis or Hadamard basis. It is crucial that x be hidden from the prover, since a prover that can predict how its state will be measured can easily break soundness of the underlying information-theoretic protocol of [FHM18]. Now, the prover's proof includes a series of values $(b_1, d_1), \ldots, (b_k, d_k)$, where each b_i is a bit and each d_i is a string.[3] For each position where $x_i = 0$ (indicating a computational basis measurement), the verifier will completely ignore (b_i, d_i), and for each position where $x_i = 1$ (indicating a Hadamard basis measurement), the verifier will compute some bit $e_i := b_i \oplus (d_i \cdot s_i)$, where s_i is some secret known to the verifier. The e_i's are then computed on as part of the verifier's verdict function.

Consider an adversary, with access to an obfuscation of the verifier, that first honestly computes an accepting proof that includes values $(b_1, d_1), \ldots, (b_k, d_k)$. Then, it generates random $b'_1, \ldots, b'_k$ until the same proof except with $(b'_1, d_1), \ldots, (b'_k, d_k)$ rejects. Now the adversary can flip b'_i to b_i one at a time until the proof accepts again. If the proof flipped back to accepting at index i, then it must be the case that $x_i = 1$, since (b_i, d_i) was not ignored by the verifier. An adversary can repeat this process until it learns enough of x to break soundness. Once the adversary can make the verifier accept even on rejecting instances, it is able to learn any other secrets hidden in the obfuscation of the verifier (concretely, information about the obfuscated quantum circuit).

The above describes an attack against the basic scheme, but there may be ways of altering the protocol to avoid such an attack. Below, we argue that three natural variants of the protocol will still be susceptible to attacks.

Sampling Fresh Hamiltonian Terms. The above attack does assume that the same verification function is applied to the e_i's across all of the different proofs that the prover submits. This will be the case if the verifier has sampled and fixed the same set of Hamiltonian terms to measure. However, one could consider re-sampling the Hamiltonian terms each time, by using randomness derived from applying a PRF to the proof. Then, when the adversary observes that flipping b'_i to b_i results in the verification flipping from rejecting to accepting, it may not be clear whether this flip occurred as a result of changing e_i, or as a result of sampling a new subset of indices to compute on (corresponding to a different set of Hamiltonian terms). Unfortunately, this does not solve the issue that the keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_k$ commit to a fixed string x of measurement bases. Indeed, an adversary could still compute accepting/rejecting statistics for b_i vs. b'_i, by altering other parts of the proof. If the acceptance probability remains roughly the same, it is likely that $x_i = 0$, and otherwise it is likely that $x_i = 1$.

Removing the Commitment. One could try to design a CVQC protocol where $\mathsf{pk}_1, \ldots, \mathsf{pk}_k$ does not commit to x, i.e. each pk_i is the public key for a trapdoor claw-free function family (as opposed to Mahadev's protocol [Mah18b], where some of the public keys are for injective functions). However, the ability to flip

[3] Actually some positions will be designated for a test round, in which case the proof at those positions will have a different structure, but this can be ignored for the purpose of this discussion.

the verifier's verdict based on just flipping (b_i, d_i) to (b_i', d_i) will still lead to an attack. Indeed, the adversary can now alter the d_i in order to learn enough linear equations of the verifier's secret s_i to eventually recover s_i. The value s_i is actually related to the LWE secret underlying the pk_i, allowing the adversary to break the claw-freeness of the function, and thus soundness of the protocol.

Quantum CVQC. Morimae [Mor20] recently showed how to construct an information-theoretically secure two-message verification protocol where the first message from verifier to prover is quantum, but the proof and subsequent verification function are entirely classical. Thus, one could still use classical obfuscation to obfuscate the verification circuit (though now the obfuscation of the quantum circuit would be a quantum state). However, this scheme would succumb to the same class of attacks outlined above. Indeed, [Mor20]'s protocol (roughly) works by having the verifier prepare several EPR pairs and release half of each to the prover. The prover then prepares a (quantum) proof for [FHM18]'s protocol and *teleports* this proof into the verifier's halves of the EPR pairs. That is, the prover performs Bell measurements between its proof and the quantum state sent by the verifier, and the final proof just consists of the resulting teleportation errors. Now, the verification can be made completely classical by *pre-measuring* (in either the computational or Hadamard basis) each of the verifier halves of the EPR pairs and hard-coding these measurement results into the verification circuit. Unfortunately, this also implies that qubit i sent by the verifier is again essentially a commitment to measuring qubit i of the prover's proof in either the computational or Hadamard basis, and the attack sketched above will still apply.

2.2 Using Indistinguishability Obfuscation

Another natural question is whether one can replace the use of ideal obfuscation with indistinguishability obfuscation (iO) in the construction of null-iO for quantum circuits. This appears to be challenging, in particular because, even for rejecting instances, an exponential number of accepting transcripts still exist. Indeed, the CVQC protocol of [CCY20, ACGH20] is computationally sound, and even the two message protocol of [Mor20] (with quantum first message) is only statistically sound. The verification circuit in [Mor20] will accept exponentially many proofs even for no instances, as the underlying delegation of quantum computation protocol has a probabilistic verifier (that chooses which Hamiltonian terms to measure on each copy of the history state). Even though soundness can be driven to negligible by parallel repetition, this also rapidly increases the proof size. Thus, attempting to hybrid over each proof will fail, since the number of hybrids will be much larger than inverse of the soundness error. Given this barrier to constructing null-iO for quantum circuits using our approach, we leave constructing any of the primitives discussed in this work from iO for classical circuits as an intriguing open problem.

3 Preliminaries

We denote by λ the security parameter. A function $f : \mathbb{N} \to [0, 1]$ is negligible if for every constant $c \in \mathbb{N}$ there exists $N \in \mathbb{N}$ such that for all $n > N$, $f(n) < n^{-c}$. We recall some standard notation for classical Turing machines and Boolean circuits:

- We say that a Turing machine (or algorithm) is PPT if it is probabilistic and runs in polynomial time in λ.
- We sometimes think about PPT Turing machines as polynomial-size uniform families of circuits. A polynomial-size circuit family C is a sequence of circuits $C = \{C_\lambda\}_{\lambda\in\mathbb{N}}$, such that each circuit C_λ is of polynomial size $\lambda^{O(1)}$ and has $\lambda^{O(1)}$ input and output bits. We say that the family is uniform if there exists a polynomial-time deterministic Turing machine M that on input 1^λ outputs C_λ.
- For a PPT Turing machine (algorithm) M, we denote by $M(x; r)$ the output of M on input x and random coins r. For such an algorithm, and any input x, we write $m \in M(x)$ to denote that m is in

the support of $M(x;\cdot)$. Finally we write $y \leftarrow\!\$\, M(x)$ to denote the computation of M on input x with some uniformly sampled random coins.

3.1 Quantum Computation

We recall some notation for quantum computation and we define the notions of computational and statistical indistinguishability for quantum adversaries. Various parts of what follows are taken almost verbatim from [BS20].

We say that a Turing machine (or algorithm) is QPT if it is quantum and runs in polynomial time. We sometimes think about QPT Turing machines as polynomial-size uniform families of quantum circuits (as these are equivalent models). We call a polynomial-size quantum circuit family $C = \{C_\lambda\}_{\lambda\in\mathbb{N}}$ uniform if there exists a polynomial-time deterministic Turing machine M that on input 1^λ outputs C_λ.

Throughout this work, we model efficient adversaries as quantum circuits with non-uniform quantum advices. This is denoted by $\mathcal{A}^* = \{\mathcal{A}^*_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$, where $\{\mathcal{A}^*_\lambda\}_{\lambda\in\mathbb{N}}$ is a polynomial-size non-uniform sequence of quantum circuits, and $\{\rho_\lambda\}_{\lambda\in\mathbb{N}}$ is some polynomial-size sequence of mixed quantum states. We now define the formal notion of computational indistinguishability in the quantum setting, where random variables X, Y are represented as mixed quantum states.

Definition 3.1 (Computational Indistinguishability)**.** *Two ensembles of quantum random variables $\mathcal{X} = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ are said to be computationally indistinguishable (denoted by $\mathcal{X} \approx_c \mathcal{Y}$) if there exists a negligible function ν such that for all $\lambda \in \mathbb{N}$ and all non-uniform QPT distinguishers with quantum advice $\mathcal{A} = \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$, it holds that*

$$|\Pr[\mathcal{A}_\lambda(X_\lambda;\rho_\lambda) = 1] - \Pr[\mathcal{A}(Y_\lambda;\rho_\lambda) = 1]| \leq \nu(\lambda).$$

The trace distance between two quantum distributions (X_λ, Y_λ), denoted by $\mathsf{TD}(X_\lambda, Y_\lambda)$, is a generalization of statistical distance to the quantum setting and represents the maximal distinguishing advantage between two quantum distributions by an unbounded quantum algorithm. We define below the notion of statistical indistinguishability.

Definition 3.2 (Statistical Indistinguishability)**.** *Two ensembles of quantum random variables $\mathcal{X} = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ are said to be statistically indistinguishable (denoted by $\mathcal{X} \approx_s \mathcal{Y}$) if there exists a negligible function ν such that for all $\lambda \in \mathbb{N}$, it holds that*

$$\mathsf{TD}(X_\lambda, Y_\lambda) \leq \nu(\lambda).$$

Throughout this work, we will often consider quantum circuits Q that output a single classical bit. Any such circuit can be written as a unitary followed by a computational basis measurement of the first qubit. When we compute Q on some classical (resp. quantum) input x (resp. $|\psi\rangle$), we write $Q(x)$ (resp. $Q(|\psi\rangle)$) to denote the output that results from padding the input with sufficiently many ancillary $|0\rangle$ states (as determined by the description of Q), computing a unitary, and then measuring the first qubit. We will sometimes consider the following restricted families of "psuedo-deterministic" quantum circuits.

Definition 3.3 (Pseudo-Deterministic Quantum Circuit)**.** *A family of psuedo-deterministic quantum circuits is defined by a family of circuits $\{Q_\lambda\}_{\lambda\in\mathbb{N}}$. The circuit defined by Q_λ takes as input a bit string $x \in \{0,1\}^{n(\lambda)}$ (along with ancillary 0 states) and outputs a single classical bit $b \leftarrow U(x)$. The circuit is pseudo-deterministic if there exists a negligible function ν such that for every sequence of classical inputs $\{x_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a sequence of outputs $\{b_\lambda\}_{\lambda\in\mathbb{N}}$ such that*

$$\Pr[Q_\lambda(x_\lambda) = b_\lambda] = 1 - \nu(\lambda).$$

3.2 Learning with Errors

We recall the definition of the learning with errors (LWE) problem [Reg05].

Definition 3.4 (Learning with Errors)**.** *The LWE problem is parametrized by a modulus $q = q(\lambda)$, polynomials $n = n(\lambda)$ and $m = m(\lambda)$, and an error distribution χ. The LWE problem is hard if it holds that*

$$(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{u})$$

where $\mathbf{A} \leftarrow\$ \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow\$ \mathbb{Z}_q^n$, $\mathbf{u} \leftarrow\$ \mathbb{Z}_q^m$, and $\mathbf{e} \leftarrow\$ \chi^m$.

As shown in [Reg05, PRS17], for any sufficiently large modulus q the LWE problem where χ is a discrete Gaussian distribution with parameter $\sigma = \xi q \geq 2\sqrt{n}$ (i.e. the distribution over $\mathbb{Z}$ where the probability of x is proportional to $e^{-\pi(|x|/\sigma)^2}$), is at least as hard as approximating the shortest independent vector problem (SIVP) to within a factor of $\gamma = \tilde{O}(n/\xi)$ in *worst case* dimension n lattices.

3.3 Quantum Fully-Homomorphic Encryption

We recall the notion of quantum fully homomorphic encryption (QFHE) [BJ15]. In this work we are interested in QFHE schemes with classical keys, classical encryption of classical messages, and classical decryption and therefore we only define QFHE for this restricted case.

Definition 3.5 (Quantum Homomorphic Encryption)**.** *A quantum homomorphic encryption scheme* $(\mathsf{QFHE.Gen}, \mathsf{QFHE.Enc}, \mathsf{QFHE.Eval}, \mathsf{QFHE.Dec})$ *consists of the following efficient algorithms.*

- $\mathsf{QFHE.Gen}(1^\lambda)$*: On input the security parameter, the key generation algorithm returns secret/public key pair* $(\mathsf{sk}, \mathsf{pk})$.
- $\mathsf{QFHE.Enc}(\mathsf{pk}, m)$*: On input the public key* pk *and a message m, the encryption algorithm returns a ciphertext c.*
- $\mathsf{QFHE.Eval}(\mathsf{pk}, C, c)$*: On input the public key* pk*, a quantum circuit C, and a ciphertext c, the evaluation algorithm returns an evaluated ciphertext $\tilde{c}$.*
- $\mathsf{QFHE.Dec}(\mathsf{sk}, c)$*: On input the secret key* sk *and a ciphertext c, the decryption algorithm returns a message m.*

Analogously to the classical case [Gen09], we say that the scheme is fully homomorphic if the evaluation algorithm supports all polynomial-size quantum circuits. We say that the scheme is leveled if the maximum depth of the homomorphically evaluated circuits is bounded by the key generation algorithm (formally, the key generation algorithm takes as input an additional 1^d depth parameter). In this work, we are only interested in levelled schemes, so we drop the adjective wherever it is clear from the context. Next we define the notion of (single-hop) evaluation correctness for QFHE.

Definition 3.6 (Evaluation Correctness)**.** *A quantum homomorphic encryption scheme* $(\mathsf{QFHE.Gen}, \mathsf{QFHE.Enc}, \mathsf{QFHE.Eval}, \mathsf{QFHE.Dec})$ *is correct if for all $\lambda \in \mathbb{N}$, all $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{QFHE.Gen}(1^\lambda)$, all messages m, and all polynomial-size quantum circuits C, it holds that*

$$\mathsf{QFHE.Dec}(\mathsf{sk}, \mathsf{QFHE.Eval}(\mathsf{pk}, C, \mathsf{QFHE.Enc}(\mathsf{pk}, m))) \approx_s C(m).$$

The notion of semantic security is defined analogously to the classical case, and we refer the reader to [BJ15] for a formal definition. The works of Mahadev [Mah18a] and Brakerski [Bra18] show that QFHE with classical keys can be constructed from the quantum hardness of the LWE problem. We recall their results below.

Lemma 3.7 ([Mah18a, Bra18])**.** *Assuming the quantum hardness of the LWE problem, there exists a (leveled) QFHE scheme* $(\mathsf{QFHE.Gen}, \mathsf{QFHE.Enc}, \mathsf{QFHE.Eval}, \mathsf{QFHE.Dec})$ *with classical keys and classical decryption.*

3.4 Indistinguishability Obfuscation

We recall the definition of indistinguishability obuscation [GGH+13] for classical circuits. An obfuscator Obf is a PPT algorithm that must satisfy correctness and security (in an indistinguishability sense).

Definition 3.8 (Correctness)**.** *An obfuscator* Obf *is correct if for all* $\lambda \in \mathbb{N}$ *and all circuits* C *it holds that*

$$\Pr\left[\forall x \in \{0,1\}^n : C(x) = \mathsf{Obf}(1^\lambda, C)(x)\right] = 1.$$

Security is defined below.

Definition 3.9 (Security)**.** *An obfuscator* Obf *is secure if for all* $\lambda \in \mathbb{N}$ *and all pairs of circuits* (C_0, C_1) *such that for all inputs* x *it holds that* $C_0(x) = C_1(x)$ *and* $|C_0| = |C_1|$*, it holds that*

$$\mathsf{Obf}^*(1^\lambda, C_0) \approx_c \mathsf{Obf}^*(1^\lambda, C_1).$$

3.5 Puncturable Pseudorandom Functions

We recall the definition of a puncturable pseudorandom function (PRF) from [SW14].

Definition 3.10 (Puncturable Pseudorandom Function)**.** *A puncturable PRF* $(\mathsf{PRF.Gen}, \mathsf{PRF.Punct}, \mathsf{PRF.Eval})$ *consists of the following efficient algorithms.*

- $\mathsf{PRF.Gen}(1^\lambda)$*: On input the security parameter, the key generation algorithm returns a key* k*.*
- $\mathsf{PRF.Punct}(k, z)$*: On input a key* k *and a point* z*, the puncturing algorithm returns the punctured key* k_z*.*
- $\mathsf{PRF.Eval}(k, x)$*: On input a key* k *and a string* $x \in \{0,1\}^\lambda$*, the evaluation algorithm returns a string* $y \in \{0,1\}^\lambda$*.*

Correctness requires that the evaluation over the punctured key agrees with the evaluation over the non-punctured key, except for the punctured point.

Definition 3.11 (Correctness)**.** *A puncturable PRF* $(\mathsf{PRF.Gen}, \mathsf{PRF.Punct}, \mathsf{PRF.Eval})$ *is correct if for all* $\lambda \in \mathbb{N}$*, all* $z \in \{0,1\}^\lambda$*, and all strings* $x \neq z$ *it holds that*

$$\Pr\left[\mathsf{PRF.Eval}(k, x) = \mathsf{PRF.Eval}(k_z, x)\right] = 1$$

where $k \leftarrow\!\!\$\, \mathsf{PRF.Gen}(1^\lambda)$ *and* $k_z \leftarrow\!\!\$\, \mathsf{PRF.Punct}(k, z)$*.*

Pseudorandomness requires that the evaluation of the PRF at any point z is computationally indistinguishable from random, even given the punctured key k_z.

Definition 3.12 (Pseudorandomness)**.** *A puncturable PRF* $(\mathsf{PRF.Gen}, \mathsf{PRF.Punct}, \mathsf{PRF.Eval})$ *is pseudorandom if for all* $\lambda \in \mathbb{N}$ *and all* $z \in \{0,1\}^\lambda$ *it holds that*

$$(\mathsf{PRF.Eval}(k, z), k_z) \approx_c (u, k_z)$$

where $k \leftarrow\!\!\$\, \mathsf{PRF.Gen}(1^\lambda)$*,* $k_z \leftarrow\!\!\$\, \mathsf{PRF.Punct}(k, z)$*, and* $u \leftarrow\!\!\$\, \{0,1\}^\lambda$*.*

4 Obfuscation of Null Quantum Circuits

In this section, we present a scheme to obfuscate null quantum circuits. Then we show that such a scheme implies the existence of a witness encryption scheme for QMA.

4.1 Classical Verification of Quantum Computation

Mahadev [Mah18b] gave the first protocol for classical verification of quantum computation. The protocol was recently improved to only require two messages in the quantum random oracle model [CCY20, ACGH20].

Definition 4.1 (Classical Verification of Quantum Computation (CVQC))**.** *A two-message classical verification of quantum computation protocol consists of algorithms* $(\mathsf{KeyGen}, \mathsf{Prove}, \mathsf{Verify})$*. The following is defined for quantum circuits Q with one classical bit of output.*

- $\mathsf{KeyGen}(1^\lambda, Q; r)$*: On input the security parameter and a quantum circuit Q, the PPT algorithm* KeyGen *samples uniformly random coins r and outputs public parameters* pp*.*
- $\mathsf{Prove}^{\mathcal{H}}(\mathsf{pp}, |\psi\rangle)$*: On input the public parameters* pp *and a quantum state $|\psi\rangle$, the QPT* Prove *algorithm, with query access to random oracle $\mathcal{H}$, outputs a proof π.*
- $\mathsf{Verify}^{\mathcal{H}}(Q, \pi, r)$*: On input a quantum circuit Q, a proof π, and random coins r, the PPT algorithm* Verify*, with query access to a random oracle $\mathcal{H}$, outputs a bit b indicating acceptance or rejection.*

We require the following notion of correctness.

Definition 4.2 (Correctness)**.** *A CVQC protocol* $(\mathsf{KeyGen}, \mathsf{Prove}, \mathsf{Verify})$ *is correct if for any negligible function ν, there exists a polynomial k and negligible function μ such that for all polynomial-size families of quantum circuits $\{Q_\lambda\}_{\lambda\in\mathbb{N}}$ and inputs $\{|\psi_\lambda\rangle\}_{\lambda\in\mathbb{N}}$ such that there exists $b \in \{0,1\}$ such that $\Pr[Q_\lambda(|\psi_\lambda\rangle) = b] \geq 1 - \nu(\lambda)$, it holds that*

$$\Pr\left[\mathsf{Verify}^{\mathcal{H}}(Q_\lambda, \pi, r) = b : \mathsf{pp} = \mathsf{Gen}(1^\lambda, Q_\lambda; r), \pi \leftarrow\!\!\$\, \mathsf{Prove}^{\mathcal{H}}(\mathsf{pp}, |\psi_\lambda\rangle^{\otimes k(\lambda)})\right] = 1 - \mu(\lambda).$$

Remark 4.3. *We state correctness above with respect to (quantum circuit, input) pairs that either accept or reject with overwhelming probability. By standard QMA amplification, a protocol that satisfies this correctness guarantee can also be used to verify (quantum circuit, input) pairs that either accept with probability α or reject with probability β, where α and β are separated by an inverse polynomial.*

We define the notion of soundness below.

Definition 4.4 (Soundness)**.** *A CVQC protocol* $(\mathsf{KeyGen}, \mathsf{Prove}, \mathsf{Verify})$ *is sound if for any negligible function ν, any polynomial-size family of quantum circuits $\{Q_\lambda\}_{\lambda\in\mathbb{N}}$ such that for all inputs $\{|\psi_\lambda\rangle\}_{\lambda\in\mathbb{N}}$, $\Pr[Q_\lambda(|\psi_\lambda\rangle) = 1] \leq \nu(\lambda)$, and all QPT adversaries $\mathcal{A}$, there exists a negligible function μ such that*

$$\Pr\left[\begin{array}{c} \mathsf{Verify}^{\mathcal{H}}(Q_\lambda, \pi, r) = 1 \\ \vdots \end{array} \begin{array}{l} \mathsf{pp} = \mathsf{Gen}(1^\lambda, Q_\lambda; r), \\ \pi \leftarrow\!\!\$\, \mathcal{A}^{|\mathcal{H}\rangle}(\mathsf{pp}) \end{array}\right] = \mu(\lambda).$$

The notation $\mathcal{A}^{|\mathcal{H}\rangle}$ indicates that $\mathcal{A}$ has quantum *query access to the random oracle $\mathcal{H}$.*

Blindness. In this work we also require the notion of blindness for a CVQC protocol, which we define below.

Definition 4.5 (Blindness)**.** *A CVQC protocol* $(\mathsf{KeyGen}, \mathsf{Prove}, \mathsf{Verify})$ *is blind if for any QPT adversary $\mathcal{A}$ there exists a negligible function $\nu(\lambda)$ such that for any polynomial-size sequences of circuits $\{Q_{0,\lambda}\}_{\lambda\in\mathbb{N}}, \{Q_{1,\lambda}\}_{\lambda\in\mathbb{N}}$, it holds that*

$$\left|\Pr\left[\mathcal{A}^{|\mathcal{H}\rangle}(\mathsf{pp}) = 1 : \mathsf{pp} \leftarrow\!\!\$\, \mathsf{Gen}(1^\lambda, Q_{0,\lambda})\right] - \Pr\left[\mathcal{A}^{|\mathcal{H}\rangle}(\mathsf{pp}) = 1 : \mathsf{pp} \leftarrow\!\!\$\, \mathsf{Gen}(1^\lambda, Q_{1,\lambda})\right]\right| = \nu(\lambda).$$

While the works of [CCY20, ACGH20] give CVQC protocols that satisfy correctness and soundness, their protocols are not blind. However, using quantum fully-homomorphic encryption (Definition 3.5), it is straightforward to obtain a two-message blind protocol in the QROM, by applying Fiat-Shamir to a blind variant of the four-message CVQC protocol from [CCY20, ACGH20].

Lemma 4.6. *Assuming the quantum hardness of learning with errors, there exists a two-message blind classical verification of quantum computation protocol in the QROM.*

Proof. Consider the four-message CVQC protocol from [CCY20, ACGH20], which is correct and sound assuming the quantum hardness of learning with errors. The protocol includes two prover algorithms Prove_1 and Prove_2, and operates as follows. First, $\mathsf{KeyGen}(1^\lambda, Q; r)$ is run to produce parameters pp. Next, $\mathsf{Prove}_1(\mathsf{pp}, |\psi\rangle)$ outputs a classical string y and quantum state $|\psi_1\rangle$. Next, a uniformly random classical challenge c is sampled, and $\mathsf{Prove}_2(\mathsf{pp}, y, c, |\psi_1\rangle)$ is run to produce a classical proof π. Finally, $\mathsf{Verify}(Q, y, c, \pi, r)$ outputs a bit indicating acceptance of rejection.

This protocol consisting of algorithms $(\mathsf{KeyGen}, \mathsf{Prove}_1, \mathsf{Prove}_2, \mathsf{Verify})$ can be turned into a blind protocol $(\mathsf{KeyGen}', \mathsf{Prove}_1', \mathsf{Prove}_2', \mathsf{Verify}')$ as follows. KeyGen' will run KeyGen, sample a random QFHE key pair $(\mathsf{pk}, \mathsf{sk})$, and then output $\mathsf{Enc}(\mathsf{pk}, \mathsf{pp})$. Prove_1' will now run Prove_1 under the QFHE to produce $(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, y))$. The challenge c will still be sampled and given to the prover in the clear. Prove_2' will now run Prove_2 under the QFHE to produce $\mathsf{Enc}(\mathsf{pk}, \pi)$. Finally, the verification procedure $\mathsf{Verify}'(Q, \mathsf{Enc}(\mathsf{pk}, y), c, \mathsf{Enc}(\mathsf{pk}, \pi), r)$ will compute sk from r, decrypt the ciphertexts to obtain y and π, and then run $\mathsf{Verify}(Q, y, c, \pi, r)$.

Correctness of $(\mathsf{KeyGen}', \mathsf{Prove}_1', \mathsf{Prove}_2', \mathsf{Verify}')$ follows immediately from correctness of QFHE. Soundness follows by a reduction to the soundness of $(\mathsf{KeyGen}, \mathsf{Prove}_1, \mathsf{Prove}_2, \mathsf{Verify})$, in which the reduction samples the $(\mathsf{pk}, \mathsf{sk})$ key pair, encrypts pp received from its challenger, and decrypts each of $\mathsf{Enc}(\mathsf{pk}, y)$ and $\mathsf{Enc}(\mathsf{pk}, \pi)$ before forwarding them to its challenger. Blindness follows immediately from the semantic security of QFHE.

Finally, we can compile $(\mathsf{KeyGen}', \mathsf{Prove}_1', \mathsf{Prove}_2', \mathsf{Verify}')$ into a two-message protocol in the QROM, by appealing to the "Fiat-Shamir for generalized Σ-protocols" Lemma [ACGH20, Lemma 6.2]. □

4.2 Obfuscation of Null Quantum Circuits

Below we define obfuscation for a class of null quantum circuits. We again consider the set of quantum circuits that have one bit of classical output.

Definition 4.7 (Null-iO for Quantum Circuits)**.** *A null obfuscator* $(\mathsf{Obf}, \mathsf{Eval})$ *for quantum circuits is defined as the following QPT algorithms.*

- $\mathsf{Obf}(1^\lambda, Q)$*: The obfuscation algorithm takes as input a security parameter* 1^λ *and a quantum circuit* Q *with one classical bit of output, and outputs an obfuscation* $\widetilde{Q}$*.*
- $\mathsf{Eval}(\widetilde{Q}, |\psi\rangle)$*: The evaluation algorithm takes as input a unitary* $\widetilde{Q}$ *and an input* $|\psi\rangle$ *and outputs a bit* b*.*

We define correctness below.

Definition 4.8 (Correctness)**.** *A null obfuscator* $(\mathsf{Obf}, \mathsf{Eval})$ *is correct if for any negligible function* ν*, there exists a polynomial* k *and negligible function* μ *such that for all polynomial-size families of quantum circuits* $\{Q_\lambda\}_{\lambda\in\mathbb{N}}$ *and inputs* $\{|\psi_\lambda\rangle\}_{\lambda\in\mathbb{N}}$ *such that there exists* $b \in \{0,1\}$ *such that* $\Pr[Q_\lambda(|\psi_\lambda\rangle) = b] \geq 1 - \nu(\lambda)$*, it holds that*

$$\Pr[\mathsf{Eval}(\widetilde{Q}, |\psi_\lambda\rangle^{\otimes k(\lambda)}) = b : \widetilde{Q} \leftarrow\$ \mathsf{Obf}(1^\lambda, Q_\lambda)] \geq 1 - \mu(\lambda).$$

Remark 4.9. *The correctness guarantee stated above is a weakening of standard obfuscation correctness in two ways. First, it only guarantees correctness for inputs that either accept or reject with high probability. Second, it requires the evaluator to possess multiple copies of the quantum input. However, note that for the class of pseudo-deterministic quantum circuits on classical inputs (Definition 3.3), where each input* x *is mapped to a particular classical output* $y = Q(x)$ *with overwhelming probability, the above correctness guarantee is standard. Thus, this null-iO definition can be thought of as standard null-iO for psuedo-deterministic quantum circuits (strictly generalizing null-iO for classical circuits) with an additional correctness guarantee that holds when considering certain quantum inputs.*

We define the notion of security below.

Definition 4.10 (Security)**.** *A null obfuscator* $(\mathsf{Obf}, \mathsf{Eval})$ *is secure if for any negligible function* ν *and polynomial-size sequences of quantum circuits* $\{U_{0,\lambda}\}_{\lambda\in\mathbb{N}}, \{U_{1,\lambda}\}_{\lambda\in\mathbb{N}}$ *such that for all inputs* $\{|\psi_\lambda\rangle\}_{\lambda\in\mathbb{N}}$, $\Pr[Q_{0,\lambda}(|\psi_\lambda\rangle) = 0] \geq 1 - \nu(\lambda)$ *and* $\Pr[Q_{1,\lambda}(|\psi_\lambda\rangle) = 0] \geq 1 - \nu(\lambda)$*, it holds that*

$$\mathsf{Obf}(1^\lambda, Q_{0,\lambda}) \approx_c \mathsf{Obf}(1^\lambda, Q_{1,\lambda}).$$

Construction. Let $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ be a two-message blind CVQC scheme in the QROM (Definition 4.1). Let $\mathcal{O}$ be an ideal obfuscator for classical circuits. Let $F : \{0,1\}^\lambda \times \{0,1\}^* \to \{0,1\}^{m(\lambda)}$ be a quantum-secure PRF, that is, a function that remains indistinguishable from a truly random function even when the adversary can issue superposition queries. Quantum-secure PRFs are known from quantum-secure one-way functions [Zha12].

Null-iO for Quantum Circuits

- $\mathsf{Obf}(1^\lambda, Q)$:
  - Sample $r \leftarrow\$ \{0,1\}^\lambda$ and compute $\mathsf{pp} = \mathsf{Gen}(1^\lambda, Q; r)$.
  - Sample a PRF key $k \leftarrow\$ \{0,1\}^\lambda$ and compute $o_1 = \mathcal{O}(F(k, \cdot))$.
  - Let $V[Q, k, r](\cdot)$ be the circuit that takes as input π and computes and outputs $b = \mathsf{Verify}^{F(k,\cdot)}(Q, \pi, r)$. Compute $o_2 = \mathcal{O}(V[Q, k, r](\cdot))$.
  - Output $\widetilde{Q} = (\mathsf{pp}, o_1, o_2)$.
- $\mathsf{Eval}(\widetilde{Q}, |\psi\rangle)$: Compute $\pi \leftarrow\$ \mathsf{Prove}^{o_1}(\mathsf{pp}, |\psi\rangle)$ and output $o_2(\pi)$.

Figure 1: A null obfuscation scheme for quantum circuits.

Theorem 4.11. *Assuming ideal obfuscation of classical circuits and the quantum hardness of learning with errors, the scheme in Figure 1 is a secure null-iO for quantum circuits.*

Proof. Correctness follows immediately from correctness of CVQC and the ideal obfuscator, and security of the PRF.

To show security, fix any two null circuits Q_0, Q_1 (these are technically families of circuits, but we drop the indexing by λ to avoid clutter), an adversary $\mathcal{A}$ that makes at most q queries to o_2, and consider the following sequence of hybrids.

- $\mathcal{H}_0$: Adversary $\mathcal{A}$ receives $\mathsf{Obf}(1^\lambda, Q_0)$, which consists of pp, and access to the two oracles o_1 and o_2.
- $\mathcal{H}_1$: Queries to o_1 are forwarded to a quantum-accessible random oracle $\mathcal{H}$, and PRF calls during queries to o_2 are also forwarded to $\mathcal{H}$. Since F is a quantum-secure PRF, this is indistinguishable from $\mathcal{H}_0$.
- $\mathcal{H}_{2,i}$ for $i \in [q]$: The adversary's first i queries to o_2 are answered by $\sum_\pi \alpha_\pi |\pi\rangle \to \sum_\pi \alpha_\pi |\pi\rangle |0\rangle$. $\mathcal{H}_{2,i} \approx_c \mathcal{H}_{2,i-1}$ follows from the soundness of CVQC. Indeed, an adversary can only distinguish if its i'th query has some inverse polynomial amplitude on π such that $\mathsf{Verify}^{\mathcal{H}}(Q_0, \pi, r) = 1$. Otherwise, the hybrids would be statistically close. However, in this case, a reduction can produce an accepting proof for CVQC with inverse polynomial probability by answering each of the first $i-1$ queries with 0, and then measuring the i'th query. This violates the soundness of CVQC, since Q_0 rejects on all inputs with overwhelming probability.

- $\mathcal{H}_3$: Switch the obfuscation to Q_1. Indistinguishability follows from the blindness of CVQC. Indeed, the verifier's secret key is no longer needed to simulate this distribution, since all queries to o_2 are answered with 0.
- $\mathcal{H}_{4,i}$ for $i \in [q]$: reverse the changes from $\mathcal{H}_{2,i}$ for $i \in [q]$.
- $\mathcal{H}_5$: reverse the change from $\mathcal{H}_1$. This is $\mathsf{Obf}(1^\lambda, Q_1)$.

□

Remark 4.12. *We note that the above argument would also hold if we implemented the oracles via virtual black-box (VBB) obfuscation, instead of ideal obfuscation. We chose to present it using ideal obfuscation for conceptual simplicity.*

4.3 Witness Encryption for QMA

Recall that a language $\mathcal{L} = (\mathcal{L}_{\text{yes}}, \mathcal{L}_{\text{no}})$ in QMA is defined by a tuple $(\mathcal{V}, p, \alpha, \beta)$, where p is a polynomial, $\mathcal{V} = \{V_\lambda\}_{\lambda\in\mathbb{N}}$ is a uniformly generated family of circuits such that for every λ, V_λ takes as input a string $x \in \{0,1\}^\lambda$ and a quantum state $|\psi\rangle$ on $p(\lambda)$ qubits and returns a single bit, and $\alpha, \beta : \mathbb{N} \to [0,1]$ are such that $\alpha(\lambda) - \beta(\lambda) \geq 1/p(\lambda)$. The language is then defined as follows.

- For all $x \in \mathcal{L}_{\text{yes}}$ of length λ, there exists a quantum state $|\psi\rangle$ of size at most $p(\lambda)$ such that the probability that V_λ accepts $(x, |\psi\rangle)$ is at least $\alpha(\lambda)$. We denote the (possibly infinite) set of quantum witnesses that make V_λ accept x by $\mathsf{R}_{\mathcal{L}}(x)$.
- For all $x \in \mathcal{L}_{\text{no}}$ of length λ, and all quantum states $|\psi\rangle$ of size at most $p(\lambda)$, it holds that V_λ accepts on input $(x, |\psi\rangle)$ with probability at most $\beta(\lambda)$.

We now recall the definition of witness encryption [GGSW13], and adapt it to the quantum setting. Note that we define encryption only with respect to classical messages. This is without loss of generality, since one can encode a quantum state with the quantum one-time pad [AMTDW00] and use the witness encryption to encrypt the corresponding (classical) one-time pad keys.

Definition 4.13 (Witness Encryption for QMA)**.** *A witness encryption* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *for a language* $\mathcal{L} \in \mathsf{QMA}$ *with relation* $\mathsf{R}_{\mathcal{L}}$ *consists of the following efficient algorithms.*

- $\mathsf{WE.Enc}(1^\lambda, x, m)$*: On input the security parameter* 1^λ*, a statement* x*, and a message* $m \in \{0,1\}$*, the encryption algorithm returns a ciphertext* c*.*
- $\mathsf{WE.Dec}(x, c, |\psi\rangle)$*: On input a statement* x*, a ciphertext* c*, and a quantum state* $|\psi\rangle$*, the decryption algorithm returns a message* m *or* $\perp$*.*

We define correctness below.

Definition 4.14 (Correctness)**.** *A witness encryption* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *for a language* $\mathcal{L} \in \mathsf{QMA}$ *is correct if there exists a negligible function* $\nu(\lambda)$ *and a polynomial* $k(\lambda)$ *such that for all* $m \in \{0,1\}$*, all polynomial-length sequences of instances* $\{x_\lambda\}_{\lambda\in\mathbb{N}}$ *and witnesses* $\{|\psi_\lambda\rangle\}_{\lambda\in\mathbb{N}}$ *where each* $x_\lambda \in \mathcal{L}_{\mathsf{yes}}$ *and* $|\psi_\lambda\rangle \in \mathsf{R}_{\mathcal{L}}(x_\lambda)$*, it holds that*

$$\Pr\left[\mathsf{WE.Dec}(x_\lambda, \mathsf{WE.Enc}(1^\lambda, x_\lambda, m), |\psi_\lambda\rangle^{\otimes k(\lambda)}) = m\right] = 1 - \nu(\lambda).$$

Finally we recall the definition of security against quantum algorithms.

Definition 4.15 (Security)**.** *A witness encryption* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *for a language* $\mathcal{L} \in \mathsf{QMA}$ *is secure if for all polynomial-length sequences of instances* $\{x_\lambda\}_{\lambda\in\mathbb{N}}$ *where each* $x_\lambda \in \mathcal{L}_{\mathsf{no}}$*, it holds that*

$$\mathsf{WE.Enc}(1^\lambda, x_\lambda, 0) \approx_c \mathsf{WE.Enc}(1^\lambda, x_\lambda, 1).$$

Lemma 4.16. *Assuming null-iO for quantum circuits satisfying Definition 4.7, there exists witness encryption for QMA.*

Proof. $\mathsf{WE.Enc}(1^\lambda, x, m)$ will simply output $c \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, Q[x,m])$, where $Q[x,m]$ takes as input $|\psi\rangle$, runs the QMA verification procedure for instance x and witness $|\psi\rangle$, and then outputs m if verification accepts, and otherwise outputs $\bot$. $\mathsf{WE.Dec}(x, c, |\psi\rangle^{\otimes k(\lambda)})$ will take polynomially many copies of the witness $|\psi\rangle$ and run $\mathsf{Eval}(c, |\psi\rangle^{\otimes k(\lambda)})$ to produce either m or $\bot$. Assuming that the QMA language $\mathcal{L}$ is such that $\alpha(\lambda) = 1 - \mathsf{negl}(\lambda)$ and $\beta(\lambda) = \mathsf{negl}(\lambda)$ (which is without loss of generality by applying standard QMA amplification), correctness and security of the witness encryption scheme follow immediately from correctness and security of the null-iO. In particular, for $x \in \mathcal{L}_{\mathsf{no}}$, $Q[x,m]$ is a null circuit, implying that $\mathsf{Obf}(1^\lambda, Q[x,m]) \approx_c \mathsf{Obf}(1^\lambda, Q[x,0])$. □

5 Non-Interactive Zero-Knowledge for QMA

In the following we show how a witness encryption scheme for QMA with classical encryption allows us to obtain a non-interactive zero-knowledge (NIZK) argument for QMA. Before describing our scheme, we introduce the necessary building blocks that we are going to use.

5.1 Definition

We recall the definition of NIZK for QMA.

Definition 5.1 (NIZK Argument)**.** *A NIZK argument* $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ *for a language* $\mathcal{L} \in \mathsf{QMA}$ *with relation* $\mathsf{R}_\mathcal{L}$ *consists of the following efficient algorithms.*

- $\mathsf{NIZK.Setup}(1^\lambda)$*: On input the security parameter* 1^λ*, the setup returns a common reference string* crs*.*
- $\mathsf{NIZK.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes k(\lambda)}, x)$*: On input a common reference string* crs*,* $k(\lambda)$ *copies of the witness* $|\psi\rangle$*, and a statement* x*, the proving algorithm returns a proof* π*.*
- $\mathsf{NIZK.Verify}(\mathsf{crs}, \pi, x)$*: On input a common reference string* crs*, a proof* π*, and a statement* x*, the verification algorithm returns a bit* $\{0,1\}$*.*

We defined correctness below.

Definition 5.2 (Correctness)**.** *A NIZK argument* $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ *is correct if there exists a negligible function* ν *such that for all* $\lambda \in \mathbb{N}$*, all* $x \in \mathcal{L}_{\mathsf{yes}}$*, and all* $|\psi\rangle \in \mathsf{R}_\mathcal{L}(x)$ *it holds that*

$$\Pr\left[\mathsf{NIZK.Verify}(\mathsf{crs}, \mathsf{NIZK.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes k(\lambda)}, x), x) = 1\right] = 1 - \nu(\lambda)$$

where $\mathsf{crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$.

Next, we define (non-adaptive) computational soundness.

Definition 5.3 (Computational Soundness)**.** *A NIZK argument* $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ *is computationally sound if there exist a negligible function* ν *such that for all non-uniform QPT adversaries with quantum advice* $\mathcal{A} = \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$ *and all* $x^* \in \mathcal{L}_{\mathsf{no}}$*, it holds that*

$$\Pr\left[\mathsf{NIZK.Verify}(\mathsf{crs}, \mathcal{A}_\lambda(\mathsf{crs}, x^*; \rho_\lambda), x^*) = 1\right] = \nu(\lambda)$$

where $\mathsf{crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$.

In the following we present the notion of (statistical) zero-knowledge.

Definition 5.4 (Statistical Zero-Knowledge)**.** *A NIZK argument* $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ *is statistically zero-knowledge if here exists a simulator* Sim *such that for* $\lambda \in \mathbb{N}$*, all* $r \in \{0,1\}^\lambda$*, all statements* $x \in \mathcal{L}_{\mathsf{yes}}$*, and all witnesses* $|\psi\rangle \in \mathsf{R}_{\mathcal{L}}(x)$*, it holds that*

$$\mathsf{Sim}(1^\lambda, x, r) \approx_s \mathsf{NIZK.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes k(\lambda)}, x)$$

where $\mathsf{crs} = \mathsf{NIZK.Setup}(1^\lambda; r)$.

5.2 Construction

We describe in the following our NIZK argument system for any language $\mathcal{L} \in \mathsf{QMA}$ with relation $\mathsf{R}_{\mathcal{L}}$. We assume the existence of a witness encryption $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ with classical encryption for the same language $\mathcal{L}$, a puncturable PRF $(\mathsf{PRF.Gen}, \mathsf{PRF.Punct}, \mathsf{PRF.Eval})$, a one-way function OWF, and an indistinguishability obfuscator Obf for classical polynomial-size circuits. Our NIZK argument system $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ is presented in Figure 2.

NIZK for QMA

- $\mathsf{NIZK.Setup}(1^\lambda)$:
  - Sample two keys $k_0 \leftarrow\!\$\, \mathsf{PRF.Gen}(1^\lambda)$ and $k_1 \leftarrow\!\$\, \mathsf{PRF.Gen}(1^\lambda)$.
  - Compute the obfuscation $\widetilde{\mathbf{P}} \leftarrow\!\$\, \mathsf{Obf}(1^\lambda, \mathbf{P})$ where $\mathbf{P}$ is the circuit that, on input some statement x, returns $\mathsf{WE.Enc}(1^\lambda, x, \mathsf{PRF.Eval}(k_0, x); \mathsf{PRF.Eval}(k_1, x))$. The circuit $\mathbf{P}$ is padded to the maximum size of $\mathbf{P}^*$ (defined in the proof of Theorem 5.5).
  - Compute the obfuscation $\widetilde{\mathbf{V}} \leftarrow\!\$\, \mathsf{Obf}(1^\lambda, \mathbf{V})$ where $\mathbf{V}$ is the circuit that, on input some statement x and a string y, returns 1 if and only if $\mathsf{OWF}(\mathsf{PRF.Eval}(k_0, x)) = \mathsf{OWF}(y)$. The circuit $\mathbf{V}$ is padded to the maximum size of $\mathbf{V}^*$ (defined in the proof of Theorem 5.5).
  - Return $\mathsf{crs} = (\widetilde{\mathbf{P}}, \widetilde{\mathbf{V}})$.
- $\mathsf{NIZK.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes k(\lambda)}, x)$:
  - Compute $c = \widetilde{\mathbf{P}}(x)$.
  - Return $\pi = \mathsf{WE.Dec}(x, c, |\psi\rangle^{\otimes k(\lambda)})$.
- $\mathsf{NIZK.Verify}(\mathsf{crs}, \pi, x)$:
  - Return $\widetilde{\mathbf{V}}(x, \pi)$.

Figure 2: A publicly-verifiable NIZK argument for QMA

Correctness. It is easy to see that the scheme is correct, i.e. true statements correctly verify, except with negligible probability over the randomness imposed by the evaluation of the $\mathsf{WE.Dec}$ algorithm.

Soundness. Next we show that the scheme satisfies (non-adaptive) computational soundness.

Theorem 5.5 (Soundness)**.** *Let* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *be a witness encryption, let* $(\mathsf{PRF.Gen}, \mathsf{PRF.Punct}, \mathsf{PRF.Eval})$ *be a puncturable PRF, let* OWF *be a one-way function, and let* Obf *be an indistinguishability obfuscator. Then the scheme in Figure 2 is computationally sound.*

Proof. The proof proceeds by defining a series of hybrid distributions for the computation of the crs that we argue to be computationally indistinguishable from each other. In the last hybrid, the probability that any prover can cause the verifier to accept some $x^* \in \mathcal{L}_{\mathsf{no}}$ will be negligible.

- Hybrid $\mathcal{H}_0$: This is the original distribution where the crs is sampled from $\mathsf{crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}$.

- Hybrid $\mathcal{H}_1$: In this hybrid we compute $\widetilde{\mathbf{P}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{P}_1)$ where $\mathbf{P}_1$ is the circuit that on input some statement x, checks whether $x = x^*$. If that is the case, then it returns the ciphertext

$$c = \mathsf{WE.Enc}(1^\lambda, x^*, \mathsf{PRF.Eval}(k_0, x^*); \mathsf{PRF.Eval}(k_1, x^*)).$$

  Otherwise compute $c = \mathsf{WE.Enc}(1^\lambda, x, \mathsf{PRF.Eval}(k_0, x); \mathsf{PRF.Eval}(k_{1,x^*}, x))$, where $k_{1,x^*} \leftarrow\!\!\$\ \mathsf{PRF.Punct}(k_1, x^*)$.

  Note that the circuits $\mathbf{P}$ and $\mathbf{P}_1$ have different representations but are functionally equivalent. Thus, $\mathcal{H}_0$ and $\mathcal{H}_1$ are computationally indistinguishable by the security of the obfuscator Obf.

- Hybrid $\mathcal{H}_2$: In this hybrid we compute $\widetilde{\mathbf{P}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{P}_2)$ where $\mathbf{P}_2$ is defined as $\mathbf{P}_1$ except that if $x = x^*$, then it returns the ciphertext

$$c = \mathsf{WE.Enc}(1^\lambda, x^*, \mathsf{PRF.Eval}(k_0, x^*); u)$$

  where $u \leftarrow\!\!\$\ \{0,1\}^\lambda$.

  The indistinguishability $\mathcal{H}_1 \approx_c \mathcal{H}_2$ follows by the pseudorandomness of the puncturable PRF.

- Hybrid $\mathcal{H}_3$: Here we compute $\widetilde{\mathbf{P}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{P}_3)$ where $\mathbf{P}_3$ is defined as $\mathbf{P}_2$ except that if $x \neq x^*$, then it returns the ciphertext

$$c = \mathsf{WE.Enc}(1^\lambda, x, \mathsf{PRF.Eval}(k_{0,x^*}, x); \mathsf{PRF.Eval}(k_{1,x^*}, x))$$

  where $k_{0,x^*} \leftarrow\!\!\$\ \mathsf{PRF.Punct}(k_0, x^*)$.

  By the correctness of the puncturable PRF, the two circuits are functionally identical and therefore the computational indistinguishability follows from the security of Obf.

- Hybrid $\mathcal{H}_4$: In this hybrid we compute $\widetilde{\mathbf{P}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{P}^*)$ where $\mathbf{P}^*$ is defined as $\mathbf{P}_3$ except that if $x = x^*$, then it returns the ciphertext

$$c = \mathsf{WE.Enc}(1^\lambda, x^*, 0^\lambda; u)$$

  where $u \leftarrow\!\!\$\ \{0,1\}^\lambda$.

  Recall that $x^* \in \mathcal{L}_{\mathsf{no}}$ and thus indistinguishability between $\mathcal{H}_3$ and $\mathcal{H}_4$ follows from the security of the witness encryption scheme.

- Hybrid $\mathcal{H}_5$: We now compute $\widetilde{\mathbf{V}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{V}_1)$ where $\mathbf{V}_1$ is the circuit that, on input a pair of strings (x, y) checks whether $x = x^*$. If this is the case, then it returns 1 if $\mathsf{OWF}(\mathsf{PRF.Eval}(k_0, x^*)) = \mathsf{OWF}(y)$ and 0 otherwise. If $x \neq x^*$ it returns 1 if and only if $\mathsf{OWF}(\mathsf{PRF.Eval}(k_{0,x^*}, x)) = \mathsf{OWF}(y)$ where k_{0,x^*} is the punctured key.

  Observe that the circuits $\mathbf{V}$ and $\mathbf{V}_1$ are functionally equivalent and thus we can invoke the security of Obf to show that $\mathcal{H}_4 \approx_c \mathcal{H}_5$.

- Hybrid $\mathcal{H}_6$: In this hybrid we compute $\widetilde{\mathbf{V}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{V}_2)$ where $\mathbf{V}_2$ is defined as $\mathbf{V}_1$ except for the case where $x = x^*$. In this case the circuit returns 1 if and only if $\mathsf{OWF}(r) = \mathsf{OWF}(y)$, where $r \leftarrow\!\!\$\ \{0,1\}^\lambda$.

  The computational indistinguishability $\mathcal{H}_5 \approx_c \mathcal{H}_6$ follows from a reduction to the pseudorandomness of the puncturable PRF.

- Hybrid $\mathcal{H}_7$: In the final hybrid we compute $\widetilde{\mathbf{V}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{V}^*)$ where $\mathbf{V}^*$ is defined as $\mathbf{V}_2$ except for the case where $x = x^*$. In this case the circuit returns 1 if and only if $R = \mathsf{OWF}(y)$, where $R = \mathsf{OWF}(r)$, i.e. the image of the one-way function is hardwired in the circuit.

  Since the two circuits are functionally equivalent, we obatain that $\mathcal{H}_6 \approx_c \mathcal{H}_7$ by another invocation of the security of Obf.

Observe that causing the verifier to accept a proof π for $x^* \in \mathcal{L}_{\mathsf{no}}$ requires one to output a valid preimage of $R = \mathsf{OWF}(r)$, where r is uniformly sampled. This is a contradiction to the one-wayness of OWF and concludes our proof. □

Zero Knowledge. We now show that the scheme satisfies a strong variant of statistical zero-knowledge. Namely, we show the existence of an efficient simulator whose output is statistically close to the output of the prover, for all valid choices of the common reference string.

Theorem 5.6 (Zero Knowledge)**.** *The scheme in Figure 2 is statistically zero-knowledge.*

Proof. The simulator computes crs as in the $\mathsf{NIZK.Setup}$ algorithm and sets $\pi = \mathsf{PRF.Eval}(k_0, x)$. This distribution is identical to the one induced by the honest algorithms, except when the $\mathsf{WE.Dec}$ fails which happens only with negligible probability. □

6 ZAPR Arguments for QMA

In the following we show a transformation to lift our NIZK argument for QMA to the setting where the common reference string can be sampled maliciously. Specifically we construct a two-message witness indistinguishable argument for QMA with public verifiability. Such an argument has been referred to in the literature as a ZAPR argument (i.e. a ZAP [DN00] where the first message may be sampled with private random coins that are not needed for verification). Before presenting our scheme, we introduce the necessary cryptographic machinery.

6.1 Definition

In the following we define the notion of statistical ZAPR arguments for QMA, although a similar definition applies (with minor modifications) to the case of NP.

Definition 6.1 (ZAPR Argument)**.** *A ZAPR argument* $(\mathsf{ZAPR.Setup}, \mathsf{ZAPR.Prove}, \mathsf{ZAPR.Verify})$ *for a language* $\mathcal{L} \in \mathsf{QMA}$ *with relation* $\mathsf{R}_{\mathcal{L}}$ *consists of the following efficient algorithms.*

- $\mathsf{ZAPR.Setup}(1^\lambda)$*: On input the security parameter* 1^λ*, the setup returns a common reference string* crs*.*
- $\mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes k(\lambda)}, x)$*: On input a common reference string* crs*,* $k(\lambda)$ *copies of the witness* $|\psi\rangle$*, and a statement* x*, the proving algorithm returns a proof* π*.*
- $\mathsf{ZAPR.Verify}(\mathsf{crs}, \pi, x)$*: On input a common reference string* crs*, a proof* π*, and a statement* x*, the verification algorithm returns a bit* $\{0, 1\}$*.*

We define correctness below.

Definition 6.2 (Correctness)**.** *A ZAPR argument* $(\mathsf{ZAPR.Setup}, \mathsf{ZAPR.Prove}, \mathsf{ZAPR.Verify})$ *is correct if there exists a negligible function* ν *such that for all* $\lambda \in \mathbb{N}$*, all* $x \in \mathcal{L}_{\mathsf{yes}}$*, and all* $|\psi\rangle \in \mathsf{R}_{\mathcal{L}}(x)$ *it holds that*

$$\Pr\left[\mathsf{ZAPR.Verify}(\mathsf{crs}, \mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes k(\lambda)}, x), x) = 1\right] = 1 - \nu(\lambda)$$

where $\mathsf{crs} \leftarrow\!\!\$\ \mathsf{ZAPR.Setup}(1^\lambda)$.

Next, we define computational soundness.

Definition 6.3 (Computational Soundness)**.** *A ZAPR argument* $(\mathsf{ZAPR.Setup}, \mathsf{ZAPR.Prove}, \mathsf{ZAPR.Verify})$ *is computationally sound if there exist a negligible function* ν *such that for all non-uniform QPT adversaries with quantum advice* $\mathcal{A} = \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$ *and all* $x^* \in \mathcal{L}_{\mathsf{no}}$*, it holds that*

$$\Pr\left[\mathsf{ZAPR.Verify}(\mathsf{crs}, \mathcal{A}_\lambda(\mathsf{crs}, x^*; \rho_\lambda), x^*) = 1\right] = \nu(\lambda)$$

where $\mathsf{crs} \leftarrow\!\!\$\ \mathsf{ZAPR.Setup}(1^\lambda)$.

In the following we present the notion of (statistical) witness indistinguishability.

Definition 6.4 (Statistical Witness Indistinguishability)**.** *A ZAPR argument* $(\mathsf{ZAPR.Setup}, \mathsf{ZAPR.Prove}, \mathsf{ZAPR.Verify})$ *is witness indistinguishable if for all* $\lambda \in \mathbb{N}$*, all* $x \in \mathcal{L}_{\mathsf{yes}}$*, all pairs of witnesses* $|\psi_0\rangle, |\psi_1\rangle \in \mathsf{R}_{\mathcal{L}}(x)$*, and all common reference strings* crs*, it holds that*

$$\mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi_0\rangle^{k(\lambda)}, x) \approx_s \mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi_1\rangle^{k(\lambda)}, x).$$

Statistical ZAPs for NP. It was recently show in [BFJ+20, GJJM20] that statistical ZAPs for NP exist assuming the quasi-polynomial (quantum) hardness of the LWE problem.

Lemma 6.5 ([BFJ+20, GJJM20])**.** *Assuming the quantum quasi-polynomial hardness of the LWE problem, there exists a public coin ZAP for* NP $(\mathsf{ZAP.Setup}, \mathsf{ZAP.Prove}, \mathsf{ZAP.Verify})$*.*

6.2 Non-Interactive Witness-Indistinguishable Proofs for NP

We recall the notion of non-interactive witness-indistinguishable (NIWI) proof for NP [BOV03].

Definition 6.6 (NIWI Proof for NP)**.** *A NIWI proof* $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$ *for a language* $\mathcal{L} \in$ NP *with relation* $\mathsf{R}_{\mathcal{L}}$ *consists of the following efficient algorithms.*

- $\mathsf{NIWI.Prove}(1^\lambda, w, x)$*: On input the security parameter* 1^λ*, a witness* w*, and a statement* x*, the proving algorithm returns a proof* π*.*
- $\mathsf{NIWI.Verify}(\pi, x)$*: On input a proof* π*, and a statement* x*, the verification algorithm returns a bit* $\{0, 1\}$*.*

We defined the properties of interest below.

Definition 6.7 (Correctness)**.** *A NIWI proof* $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$ *is correct if for all* $\lambda \in \mathbb{N}$*, all* $x \in \mathcal{L}$*, and all* $w \in \mathsf{R}_{\mathcal{L}}(x)$ *it holds that*

$$\Pr\left[\mathsf{NIWI.Verify}(\mathsf{NIWI.Prove}(1^\lambda, w, x), x) = 1\right] = 1.$$

We define statistical soundness.

Definition 6.8 (Statistical Soundness)**.** *A NIWI proof* $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$ *is statistically sound if there exist a negligible function* ν *such that for all* $x^* \notin \mathcal{L}$ *and all proofs* π^* *it holds that*

$$\Pr\left[\mathsf{NIWI.Verify}(\pi^*, x^*) = 1\right] = \nu(\lambda).$$

Finally we define computational witness indistinguishability.

Definition 6.9 (Computational Witness Indistinguishability)**.** *A NIWI proof* $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$ *is witness indistinguishable if there exist a negligible function* ν *such that for all* $\lambda \in \mathbb{N}$*, all* $x \in \mathcal{L}$*, and all pairs of witnesses* $w_0, w_1 \in \mathsf{R}_{\mathcal{L}}(x)$ *it holds that*

$$\mathsf{NIWI.Prove}(1^\lambda, w_0, x) \approx_c \mathsf{NIWI.Prove}(1^\lambda, w_1, x).$$

NIWI proofs are known to exist under a variety of assumptions, but for the purpose of our work we only consider constructions that (plausibly) satisfy post-quantum security.

Lemma 6.10 ([BP15],[BPW16])**.** *Assuming the existence of post-quantum one-way functions and post-quantum sub-exponential indistinguishability obfuscation for classical circuits, there exists a post-quantum NIWI for* NP $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$*.*

6.3 Sometimes-Binding Statistically Hiding Commitments

We introduce the notion of sometimes-binding statistically hiding (SBSH) commitments, a notion formally introduced in [LVW20].

Definition 6.11 (SBSH Commitment)**.** *An SBSH commitment scheme* $(\mathsf{SBSH.Gen}, \mathsf{SBSH.Key}, \mathsf{SBSH.Com})$ *consists of the following efficient algorithms.*

- $\mathsf{SBSH.Gen}(1^\lambda)$*: On input the security parameter* 1^λ*, the generation algorithm returns a partial commitment key* ck_0*.*
- $\mathsf{SBSH.Key}(\mathsf{ck}_0)$*: On input a partial key* ck_0*, the key agreement algorithm returns the complement of the key* ck_1*.*
- $\mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), m)$*: On input a commitment key* $(\mathsf{ck}_0, \mathsf{ck}_1)$ *and a message* m*, the commitment algorithm returns a partial commitment key* ck_1 *and a commitment* c*.*

The commitment must satisfy the notion of statistical hiding.

Definition 6.12 (Statistical Hiding)**.** *An SBSH commitment scheme* $(\mathsf{SBSH.Gen}, \mathsf{SBSH.Key}, \mathsf{SBSH.Com})$ *is statistically hiding if for all* $\lambda \in \mathbb{N}$*, all partial keys* ck_0*, and all pairs of messages* (m_0, m_1)*, it holds that*

$$(\mathsf{ck}_0, \mathsf{ck}_1, \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), m_0)) \approx_s (\mathsf{ck}_0, \mathsf{ck}_1, \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), m_1))$$

where $\mathsf{ck}_1 \leftarrow\!\!\$\, \mathsf{SBSH.Key}(\mathsf{ck}_0)$*.*

Next we define the notion of sometimes-binding for an SBSH commitment scheme. We define the set $\mathsf{Binding}$ as the set of all commitment keys $(\mathsf{ck}_0, \mathsf{ck}_1)$ such that any resulting commitment is perfectly binding. We present the definition of the property in the following.

Definition 6.13 (Sometimes Binding)**.** *An SBSH commitment scheme* $(\mathsf{SBSH.Gen}, \mathsf{SBSH.Key}, \mathsf{SBSH.Com})$ *is* (ε, δ)*-sometimes binding if there exists a negligible function* ν *such that for all* $\lambda \in \mathbb{N}$ *and all (stateful) QPT distinguishers* $\mathcal{A} = \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda \in \mathbb{N}}$*, it holds that*

$$\Pr\left[\mathcal{A}_\lambda(\tau) = 1 \wedge (\mathsf{ck}_0, \mathsf{ck}_1) \in \mathsf{Binding}\right] = \varepsilon(\lambda) \cdot \Pr\left[\mathcal{A}_\lambda(\tau) = 1\right] - \delta(\lambda) \cdot \nu(\lambda)$$

where $\mathsf{ck}_0 \leftarrow\!\!\$\, \mathsf{SBSH.Gen}(1^\lambda)$ *and* $(\tau, \mathsf{ck}_1) \leftarrow \mathcal{A}_\lambda(\mathsf{ck}_0, \rho_\lambda)$*.*

We also require the existence of a polynomial-time extractor $\mathsf{SBSH.Ext}$ that, on input the random coins r used in the $\mathsf{SBSH.Gen}$ algorithm, extracts the committed message m from the protocol transcript if $(\mathsf{ck}_0, \mathsf{ck}_1) \in \mathsf{Binding}$. The works of [KKS18, BFJ+20, GJJM20] present constructions of SBSH commitment schemes (albeit with a slightly different syntax) for quasi-polynomial (ε, δ) assuming the quasi-polynomial hardness of LWE. Note that the extractor does not need to access the code of the adversary (not even as an oracle) and therefore it is well defined regardless on whether the adversary is classical or quantum.

Lemma 6.14 ([KKS18, BFJ+20, GJJM20])**.** *Assuming the quantum quasi-polynomial hardness of the LWE problem, there exists an* (ε, δ)*-sometimes binding SBSH commitment scheme* $(\mathsf{SBSH.Gen}, \mathsf{SBSH.Key}, \mathsf{SBSH.Com})$*.*

6.4 Construction

We are now in the position to present a formal description of our ZAPR argument system for any language $\mathcal{L} \in \mathsf{QMA}$ with relation $\mathsf{R}_\mathcal{L}$. Let ε be some fixed negligible function. We assume the existence of a NIZK for $\mathcal{L}$ $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ with classical setup and classical verification, a one-way function OWF, a statistical ZAP for NP $(\mathsf{ZAP.Setup}, \mathsf{ZAP.Prove}, \mathsf{ZAP.Verify})$, and a NIWI for NP $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$, all with quasi-polynomial security $\varepsilon(\lambda)^2 \cdot \nu(\lambda)$, for some negligible function ν. Finally, we assume the existence of an SBSH commitment scheme $(\mathsf{SBSH.Gen}, \mathsf{SBSH.Key}, \mathsf{SBSH.Com})$ with $(\varepsilon(\lambda), \varepsilon(\lambda)^2)$-sometimes binding. Our protocol is presented in Figure 3.

ZAPR for QMA

- $\mathsf{ZAPR.Setup}(1^\lambda)$:
  - Sample two common reference strings for the NIZK system $\mathsf{crs}_0 \leftarrow\$ \mathsf{NIZK.Setup}(1^\lambda)$ and $\mathsf{crs}_1 \leftarrow\$ \mathsf{NIZK.Setup}(1^\lambda)$.
  - Sample two strings $(x_0, x_1) \leftarrow\$ \{0,1\}^{2\lambda}$ and compute the corresponding images $y_0 = \mathsf{OWF}(x_0)$ and $y_1 = \mathsf{OWF}(x_1)$.
  - Compute the partial key of the SBSH commitment $\mathsf{ck}_0 \leftarrow\$ \mathsf{SBSH.Gen}(1^\lambda)$ and the first message of the ZAP $\mathsf{crs}'' \leftarrow\$ \mathsf{ZAP.Setup}(1^\lambda)$.
  - Compute a NIWI proof π' for the statement

$$\{\, (\mathsf{crs}_0 \in \mathsf{NIZK.Setup} \text{ AND } y_0 \in \mathsf{OWF}) \text{ OR } (\mathsf{crs}_1 \in \mathsf{NIZK.Setup} \text{ AND } y_1 \in \mathsf{OWF})\}.$$

  - Return $\mathsf{crs} = (\mathsf{crs}_0, \mathsf{crs}_1, y_0, y_1, \mathsf{ck}_0, \mathsf{crs}'', \pi')$.
- $\mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi\rangle^{\otimes 2k(\lambda)}, x)$:
  - Verify π' and abort if the verification does not succeed.
  - Compute two NIZK proofs for the statement x, $\pi_0 \leftarrow\$ \mathsf{NIZK.Prove}(\mathsf{crs}_0, |\psi\rangle^{\otimes k(\lambda)}, x)$ and $\pi_1 \leftarrow\$ \mathsf{NIZK.Prove}(\mathsf{crs}_1, |\psi\rangle^{\otimes k(\lambda)}, x)$. If π_0 is a valid proof, then set $b = 0$, else if π_1 is valid then set $b = 1$. If neither of the two proofs is valid, then abort.
  - Sample a partial key for the SBSH commitment $\mathsf{ck}_1 \leftarrow\$ \mathsf{SBSH.Key}(\mathsf{ck}_0)$.
  - Compute two SBSH commitments $c_{\mathsf{NIZK}} \leftarrow\$ \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), (b, \pi_b))$ and $c_{\mathsf{OWF}} \leftarrow\$ \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), 0)$.
  - Compute a ZAP proof π'' for the statement

$$\left\{ \begin{array}{l} c_{\mathsf{NIZK}} \in \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), (b, \pi_b)) \text{ s.t. } \mathsf{NIZK.Verify}(\mathsf{crs}_b, \pi_b, x) = 1 \\ \text{OR } c_{\mathsf{OWF}} \in \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), x_b) \text{ s.t. } \mathsf{OWF}(x_b) = y_b \end{array} \right\}.$$

  - Return $\pi = (\mathsf{ck}_1, c_{\mathsf{NIZK}}, c_{\mathsf{OWF}}, \pi'')$.
- $\mathsf{ZAPR.Verify}(\mathsf{crs}, \pi, x)$:
  - Accept if and only if π'' verifies.

Figure 3: A publicly-verifiable ZAPR argument for QMA

Correctness. The correctness of the scheme follows immediately from the correctness of the underlying primitives.

Soundness. We show how to reduce the (non-adaptive) computational soundness of our protocol to the security of the corresponding cryptographic primitives.

Theorem 6.15 (Soundness)**.** *Let* $(\mathsf{SBSH.Gen}, \mathsf{SBSH.Key}, \mathsf{SBSH.Com})$ *be an SBSH commitment with* $(\varepsilon(\lambda), \varepsilon(\lambda)^2)$*-sometimes binding. Let* $(\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ *be a NIZK for* $\mathcal{L}$*, let* OWF *be a one-way function, let* $(\mathsf{ZAP.Setup}, \mathsf{ZAP.Prove}, \mathsf{ZAP.Verify})$ *be a ZAP for NP, and let* $(\mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$*, all with negligible security in* $\varepsilon(\lambda)^2$*. Then the scheme in Figure 3 is computationally sound.*

Proof. The proof proceeds by showing a bound on the success probability of the prover of $\varepsilon(\lambda)$. Assume towards contradiction that

$$\Pr\left[\mathsf{ZAPR.Verify}(\mathsf{crs}, \mathcal{A}_\lambda(\mathsf{crs}, x^*; \rho_\lambda), x^*) = 1\right] \geq \varepsilon(\lambda).$$

By the $(\varepsilon(\lambda), \varepsilon(\lambda)^2)$-sometimes binding property of the SBSH commitment scheme, we have that

$$\Pr\left[\mathsf{ZAPR.Verify}(\mathsf{crs}, \mathcal{A}_\lambda(\mathsf{crs}, x^*; \rho_\lambda), x^*) = 1 \wedge (\mathsf{ck}_0, \mathsf{ck}_1) \in \mathsf{Binding}\right] \geq \varepsilon(\lambda)^2 \cdot (1 - \nu(\lambda))$$

for some negligible function $\nu(\lambda)$. We denote the outputs of the SBSH extractor by $r^*_{\mathsf{NIZK}} = \mathsf{SBSH.Ext}(r, \mathsf{ck}_0, \mathsf{ck}_1, c_{\mathsf{NIZK}})$ and $r^*_{\mathsf{OWF}} = \mathsf{SBSH.Ext}(r, \mathsf{ck}_0, \mathsf{ck}_1, c_{\mathsf{OWF}})$, where r denotes the random coins used in the $\mathsf{SBSH.Gen}$ algorithm. We now proceed by modifying the verification procedure to derive a contradiction.

- The verifier additionally checks whether r^*_{NIZK} or r^*_{OWF} define a bit $b \in \{0, 1\}$, either by containing a valid proof π_b with respect to crs_b or by containing a pre-image of y_b. If no such bit is defined, then the verifier aborts.

  We claim that the probability that the adversary cheats and that the bit b is well defined is at least $\varepsilon(\lambda)^2 \cdot 1/\mathsf{poly}(\lambda)$. Assume towards contradiction that this is not the case. Then we know that with probability at least $\varepsilon(\lambda)^2 \cdot (1 - \mathsf{negl}(\lambda))$ the adversary successfully cheats and the ZAP proof π'' proves a false statement. This contradicts the quasi-polynomial soundness of the ZAP for NP. It follows that a well-defined b is extracted by the proof with probability inverse polynomial in $\varepsilon(\lambda)^2$.

- During the computation of the common reference string, the verifier samples a bit b' and computes the NIWI proof using $x_{b' \oplus 1}$ and the random coins of the generation of $\mathsf{crs}_{b' \oplus 1}$ as a witness. In the verification algorithm, the verifier aborts if $b' \neq b$.

  We claim that, conditioned on the extraction being successful, the probabiliy that the verifier aborts is negligibly close to $1/2$. Assume the contrary, and consider the following reduction against the quasi-polynomial witness indistinguishability of the NIWI proof: The reduction performs the same operations of the verifier and, if the extraction is not successful, then it outputs a uniformly sampled bit, otherwise it outputs the extracted bit b. As argued above, the extraction succeeds with probability at least inverse polynomial in $\varepsilon(\lambda)^2$. Thus, the advantage of the reduction is also at least inverse polynomial in $\varepsilon(\lambda)^2$, which contradicts the security of the NIWI proof.

  The bias on the output of the reduction is identical to the probability that $b' \neq b$, conditioned on the fact that the extraction is successful. As argued above, the extraction succeeds with probability at least inverse polynomial in $\varepsilon(\lambda)^2$. This is a contradiction to the quasi-polynomial soundness of the NIWI.

By the above analysis, it follows that, with probability inverse polynomial in $\varepsilon(\lambda)^2$, the variables $(r^*_{\mathsf{NIZK}}, r^*_{\mathsf{OWF}})$ encode either (1) a valid NIZK proof π_b against crs_b or (2) the pre-image of y_b. Note that the random coins used to sample crs_b and y_b are not used by the verifier to compute the NIWI proof. Thus, case (1) contradicts the quasi-polynomial soundness of the NIZK argument, whereas (2) contradicts the quasi-polynomial security of the one-way function. □

Witness Indistinguishability. In the following we show that our scheme satisfies statistical witness indistinguishability.

Theorem 6.16 (Witness Indistinguishability)**.** *The scheme in Figure 3 is statistically witness indistinguishable.*

Proof. The proof proceeds by defining a sequence of hybrid distributions that we show to be statistically indistinguishable.

- Hybrid $\mathcal{H}_0$: This is the distribution with the proof being computed using $|\psi_0\rangle$, i.e. this is the distribution $\mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi_0\rangle^{\otimes 2k(\lambda)}, x)$.

- Hybrid $\mathcal{H}_1$: In this hybrid the algorithm checks inefficiently whether one of the two $(\mathsf{crs}_0, \mathsf{crs}_1)$ is correctly computed and whether the corresponding (y_0, y_1) is in the range of the one-way function, and aborts if this is not the case. Otherwise proceed as in $\mathcal{H}_0$.

  Note that the only difference between this hybrid and the previous hybrid is if the algorithm in $\mathcal{H}_1$ aborts and the algorithm in $\mathcal{H}_0$ does not. This however implies that both (crs_0, y_0) and (crs_1, y_1) are invalid and thus contradicts the statistical soundness of the NIWI.

- Hybrid $\mathcal{H}_2$: In this hybrid we compute $c_{\mathsf{OWF}} \leftarrow\$ \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), x')$ where x' is the (inefficiently computed) pre-image of either y_0 or y_1. Note at least one among y_0 and y_1 is guaranteed to have a pre-image.

  By the statistical hiding of the SBSH commitment, we have that $\mathcal{H}_1 \approx_s \mathcal{H}_2$.

- Hybrid $\mathcal{H}_3$: Here we compute the ZAP proof π'' using the alternative branch, i.e. using x' and the randomness of c_{OWF} as the witness.

  This change is statistically indistinguishable by the statistical indistinguishability of the ZAP argument.

- Hybrid $\mathcal{H}_4$: Here we compute $c_{\mathsf{NIZK}} \leftarrow\$ \mathsf{SBSH.Com}((\mathsf{ck}_0, \mathsf{ck}_1), 0)$.

  By the statistical indistinguishability of the SBSH commitment we have that $\mathcal{H}_3 \approx_s \mathcal{H}_4$.

- Hybrid $\mathcal{H}_5$: Here we use $|\psi_1\rangle^{\otimes k(\lambda)}$ to compute the NIZK, instead of $|\psi_0\rangle^{\otimes k(\lambda)}$.

  Note that at this point the ouput of the distribution is independent of the NIZK proofs π_0 and π_1 and therefore $\mathcal{H}_4 \equiv \mathcal{H}_5$.

- Hybrid $\mathcal{H}_6$: We revert the change done in $\mathcal{H}_4$.

- Hybrid $\mathcal{H}_7$: We revert the change done in $\mathcal{H}_3$.

- Hybrid $\mathcal{H}_7$: We revert the change done in $\mathcal{H}_2$.

- Hybrid $\mathcal{H}_8$: We revert the change done in $\mathcal{H}_1$.

Note that the last hybrid corresponds to the distribution $\mathsf{ZAPR.Prove}(\mathsf{crs}, |\psi_1\rangle^{\otimes 2k(\lambda)}, x)$, which concludes our proof. □

7 Attribute-Based Encryption for BQP

In the following we present our construction of attribute-based encryption (ABE) for quantum functionalities.

To help motivate the primitive of ABE for BQP, we remark that the verifiable computation protocol of [PRV12] readily ports to the quantum setting, giving succinct, reusable, and publicly-verifiable CVQC from attribute-based encryption for quantum circuits (with classical key generation and encryption).[4] By reusable, we mean that a potentially expensive pre-processing (that grows with the size of the circuit) can be followed by a succinct verification (that only grows with input/output size) on any unbounded number of inputs of the verifier's choice. Prior to our work, no heuristic constructions of reusable *or* publicly-verifiable CVQC were known. Indeed, [CCY20] gave a construction of two-message CVQC with succinct verifier, though it does not satisfy reusability (of the long crs), or public verifiability.

[4]This primitive can also be obtained from our SNARG for QMA.

7.1 Definition

We recall the definition of ABE. For convenience we consider the notion of *ciphertext-policy* ABE where messages are encrypted with respect to circuits and keys are issued for attribute strings. If the class of circuits supported by the scheme is large enough, then one can switch to the complementary notion (i.e. *key-policy* ABE) by encoding universal (quantum) circuits. We also consider without loss of generality an ABE that encrypts a single (classical) bit of information.

Definition 7.1 (Attribute-Based Encryption for BQP)**.** *An ABE scheme for BQP* $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ *consists of the following efficient algorithms.*

- $\mathsf{ABE.Gen}(1^\lambda, 1^\ell)$*: On input the security parameter* 1^λ *and the length* ℓ *of attributes, the parameters generation algorithm outputs a master public key* mpk*, and master secret key* msk*.*
- $\mathsf{ABE.Enc}(\mathsf{mpk}, Q, m)$*: On input the master public key* mpk*, a quantum circuit* Q *(implementing a BQP language), and a message* m*, the encryption algorithm outputs a ciphertext* ct_Q*.*
- $\mathsf{ABE.KeyGen}(\mathsf{msk}, x)$*: On input the master secret key* msk *and an attribute* x*, the key generation algorithm outputs a secret key* sk_x*.*
- $\mathsf{ABE.Dec}(\mathsf{sk}_x, \mathsf{ct}_Q)$*: On input a secret key* sk_x *and a ciphertext* ct_Q*, the decryption algorithms either outputs a message* m *or* $\perp$*.*

Throughout the rest of this work, we always assume that the ciphertexts also contain a description of the corresponding unitary U and that the keys also contain a description of the corresponding attribute x. We defined correctness below.

Definition 7.2 (Correctness)**.** *An ABE scheme* $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ *is correct if for all negligible functions* ν*, there exists a negligible function* μ *such that for any* $\lambda \in \mathbb{N}, \ell \in \mathbb{N}, m \in \{0,1\}, x \in \{0,1\}^\ell$*, and any quantum circuit* Q *on* ℓ *input bits such that* $\Pr[Q(x) = 1] = 1 - \nu(\lambda)$*, it holds that*

$$\Pr\left[\mathsf{ABE.Dec}\left(\mathsf{ABE.KeyGen}(\mathsf{msk}, x), \mathsf{ABE.Enc}(\mathsf{mpk}, Q, m)\right) = m\right] = 1 - \mu(\lambda)$$

where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow\!\!\$\, \mathsf{ABE.Gen}(1^\lambda, 1^\ell)$*.*

Finally we define the notion of security for ABE. We consider the selective notion of security, where the quantum circuit associated with the challenge ciphertext is known ahead of time. It is well known that this can be generically upgraded to the stronger notion of adaptive security (via complexity leveraging), although at the cost of an exponential decrease in the quality of the reduction.

Definition 7.3 (Security)**.** *An ABE scheme* $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ *is secure if there exists a pair of negligible functions* ν *and* μ *such that for all* $\lambda \in \mathbb{N}$*, all quantum circuits* U^**, and all admissible non-uniform QPT distinguishers with quantum advice* $\mathcal{A} = \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda \in \mathbb{N}}$*, it holds that*

$$\Pr\left[b = \mathcal{A}_\lambda(\mathsf{ct}_{Q^*}, \mathsf{mpk}; \rho_\lambda)^{\mathsf{ABE.KeyGen}(\mathsf{msk}, \cdot)} : \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow\!\!\$\, \mathsf{ABE.Gen}(1^\lambda, 1^\ell) \\ b \leftarrow\!\!\$\, \{0,1\} \\ \mathsf{ct}_{Q^*} \leftarrow\!\!\$\, \mathsf{ABE.Enc}(\mathsf{mpk}, Q^*, b)\end{array}\right] = 1/2 + \mu(\lambda)$$

where $\mathcal{A}$ *is admissible if each query* x *to* $\mathsf{ABE.KeyGen}(\mathsf{msk}, \cdot)$ *is such that* $\Pr[Q^*(x) = 1] \leq \nu(\lambda)$*.*

7.2 Construction

We are now ready the present our construction of ABE for quantum circuits. We assume the existence of a pseudorandom generator $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{\ell \cdot \lambda}$, a puncturable PRF $(\mathsf{PRF.Gen}, \mathsf{PRF.Punct}, \mathsf{PRF.Eval})$, and an indistinguishability obfuscator Obf for classical circuits, all with sub-exponential security. We additionally assume the existence of a sub-exponentially secure witness encryption scheme $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ for BQP with a classical encryption algorithm, which is implied by the scheme shown in Section 4.3. Our scheme $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ is described in Figure 4.

ABE for BQP

- $\mathsf{ABE.Gen}(1^\lambda, 1^\ell)$:
  - Sample a key $k \leftarrow\$ \mathsf{PRF.Gen}(1^\lambda)$.
  - Compute the obfuscation $\widetilde{\mathbf{P}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{P})$ where $\mathbf{P}$ is the circuit that, on input some attribute $x \in \{0,1\}^\ell$ and a string $s \in \{0,1\}^\lambda$, returns 1 if and only if $\mathsf{PRG}(s) = \mathsf{PRG}(\mathsf{PRF.Eval}(k, x))$. The circuit $\mathbf{P}[k]$ is padded to the maximum size of $\mathbf{P}^*$ (defined in the proof of Theorem 7.4).
  - Return $\mathsf{msk} = k$ and $\mathsf{mpk} = \widetilde{\mathbf{P}}$.
- $\mathsf{ABE.Enc}(\mathsf{mpk}, Q, m)$:
  - Sample a key $r \leftarrow\$ \mathsf{PRF.Gen}(1^\lambda)$.
  - Compute the obfuscation $\widetilde{\mathbf{E}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{E})$ where $\mathbf{E}$ is the circuit that, on input some attribute $x \in \{0,1\}^\ell$ and a string $s \in \{0,1\}^\lambda$, checks whether $\widetilde{\mathbf{P}}(x, s) = 1$ and returns $\mathsf{WE.Enc}(1^\lambda, (Q, x), m; \mathsf{PRF.Eval}(r, x))$ if this is the case. The circuit $\mathbf{E}[m, r]$ is padded to the maximum size of $\mathbf{E}^*$ (defined in the proof of Theorem 7.4).
  - Return $\widetilde{\mathbf{E}}$.
- $\mathsf{ABE.KeyGen}(\mathsf{msk}, x)$:
  - Return $\mathsf{PRF.Eval}(k, x)$.
- $\mathsf{ABE.Dec}(\mathsf{sk}_x, \mathsf{ct}_Q)$:
  - Compute $c = \widetilde{\mathbf{E}}(Q, x)$.
  - Return $\mathsf{WE.Dec}((Q, x), c)$.[a]

[a] Note that WE.Dec does not need to take a third input (the witness) since the statement is in BQP.

Figure 4: An attribute-based encryption scheme for BQP

Correctness. The statistical correctness of the scheme follows immediately from the correctness of the underlying building blocks. In particular, note that all the circuits that we obfuscate are entirely classical and thus obfuscation for classical circuits suffices.

Security. We analyze the (selective) security of our construction in the following. We remark that the proof can be upgraded to the stronger notion of adaptive security with additional complexity leveraging.

Theorem 7.4 (Security)**.** *Let* PRG *be a pseudorandom generator, let* (PRF.Gen, PRF.Punct, PRF.Eval) *be a puncturabl PRF, let* Obf *be an indistinguishability obfuscator for classical circuits, and let* (WE.Enc, WE.Dec) *be a witness encryption scheme, all with sub-exponential security (in the attribute size ℓ). Then the scheme in Figure 4 is secure.*

Proof. The proof proceeds by defining an exponentially long series of hybrids, iterating over all possible attributes $x \in \{0,1\}^\ell$. More specifically, starting from hybrid $\mathcal{H}_0$ (the original experiment with the bit b fixed to $b = 0$) we define, for each $i \in \{0,1\}^\ell$, a different sequence of hybrids and we argue about the indistinguishability of neighbouring distributions. As mentioned earlier, we assume all primitives we use are sub-exponentially secure, that is, there exists an $\epsilon > 0$ such that no efficient adversary can break the primitive with probability better than $2^{-\lambda^\epsilon}$. Thus, we can set the security parameter for each to be at least ℓ^c for some $c > 1/\epsilon$, ensuring that efficient adversaries have advantage $\mathsf{negl}(\lambda)/2^\ell$.

- Hybrid $\mathcal{H}_{i,0}$: Defined the previous hybrid, except that we change the way we compute the challenge ciphertext. We begin by computing a punctured key $r_i \leftarrow\$ \mathsf{PRF.Punct}(r, i)$. Then we compute $\widetilde{\mathbf{E}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{E}_1)$, where $\mathbf{E}_1$ takes as input a pair (x, s) and does the following.
  - If $x < i$: Check whether $\widetilde{\mathbf{P}}(x, s) = 1$ and return
    $$\mathsf{WE.Enc}(1^\lambda, (Q, x), 1; \mathsf{PRF.Eval}(r_i, x))$$
    if this is the case.
  - If $x = i$: Check whether $\widetilde{\mathbf{P}}(x, s) = 1$ and return
    $$\mathsf{WE.Enc}(1^\lambda, (Q, x), 0; \mathsf{PRF.Eval}(r, x))$$
    if this is the case.
  - If $x > i$: Check whether $\widetilde{\mathbf{P}}(x, s) = 1$ and return
    $$\mathsf{WE.Enc}(1^\lambda, (Q, x), 0; \mathsf{PRF.Eval}(r_i, x))$$
    if this is the case.

  Note that, by the correctness of the puncturable PRF, the circuits $\mathbf{E}$ and $\mathbf{E}_1$ are functionally equivalent and therefore indistinguishability follows from the security of the classical obfuscator Obf.

- Hybrid $\mathcal{H}_{i,1}$: Defined the previous hybrid, except that we compute $\widetilde{\mathbf{E}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{E}_2)$, where $\mathbf{E}_2$ takes as input a pair (x, s) and does the following.
  - If $x < i$: Same as $\mathbf{E}_1$.
  - If $x = i$: Check whether $\widetilde{\mathbf{P}}(x, s) = 1$ and return $\mathsf{WE.Enc}(1^\lambda, (Q, x), 0; \tilde{r})$ if this is the case, where $\tilde{r} \leftarrow\$ \{0, 1\}^\lambda$.
  - If $x > i$: Same as $\mathbf{E}_1$.

  Note that the two hybrids differ only in the definition of $\tilde{r}$, which is uniformly sampled in $\mathcal{H}_{i,1}$ and computed according to the puncturable PRF in $\mathcal{H}_{i,0}$. By the indistinguishability of the puncturable PRF, we have that the two distributions are computationally close.

- Hybrid $\mathcal{H}_{i,2}$: In this hybrid we check whether $Q^*(i) = 0$. If this is not the case, then we proceed as before. Otherwise, we compute $\widetilde{\mathbf{E}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{E}_3)$, where $\mathbf{E}_3$ takes as input a pair (x, s) and does the following.
  - If $x < i$: Same as $\mathbf{E}_2$.
  - If $x = i$: Check whether $\widetilde{\mathbf{P}}(x, s) = 1$ and return $\mathsf{WE.Enc}(1^\lambda, (Q, x), 1; \tilde{r})$ if this is the case, where $\tilde{r} \leftarrow\$ \{0, 1\}^\lambda$.
  - If $x > i$: Same as $\mathbf{E}_2$.

  Note that we change the view of the adversary only if $Q^*(i) = 0$, which implies that the statement (Q, i) is false. Thus indistinguishability follows from the security of the witness encryption scheme.

- Hybrid $\mathcal{H}_{i,3}$: This is defined as the previous one, except that we compute a punctured key $k_i \leftarrow\$ \mathsf{PRF.Punct}(k, i)$ and we modify the public parameters as follows. We obfuscate $\widetilde{\mathbf{P}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{P}_1)$ where $\mathbf{P}_1$ is the circuit that, on input some attribute $x \in \{0, 1\}^\ell$ and a string $s \in \{0, 1\}^\lambda$, does the following.
  - If $x \neq i$: Return 1 if and only if $\mathsf{PRG}(s) = \mathsf{PRG}(\mathsf{PRF.Eval}(k_i, x))$.
  - If $x = i$: Return 1 if and only if $\mathsf{PRG}(s) = \mathsf{PRG}(\mathsf{PRF.Eval}(k, i))$.

By the perfect correctness of the puncturable PRF, the two circuits are functionally equivalent and therefore the indistinguishability follows from the security of the obfuscator Obf.

- Hybrid $\mathcal{H}_{i,4}$: In this hybrid we compute $\widetilde{\mathbf{P}} \leftarrow\!\!\$\, \mathsf{Obf}(1^\lambda, \mathbf{P}_2)$ where $\mathbf{P}_2$ is the circuit that, on input some attribute $x \in \{0,1\}^\ell$ and a string $s \in \{0,1\}^\lambda$, does the following.
  - If $x \neq i$: Same as $\mathbf{P}_1$.
  - If $x = i$: Return 1 if and only if $\mathsf{PRG}(s) = \mathsf{PRG}(\tilde{k})$, where $\tilde{k} \leftarrow\!\!\$\, \{0,1\}^\lambda$.

  Additionally, we answer the query of the adversary to the key generation oracle with $\tilde{k}$, if queried on attribute i.

  Note that this hybrid is identical to the previous one, except that $\tilde{k}$ is sampled uniformly. By the security of the puncturable PRF, the two hybrids are computationally indistinguishable.

- Hybrid $\mathcal{H}_{i,5}$: Before sampling the public parameters, we check whether $Q^*(i) = 1$. If this is not the case, then we proceed as before. Otherwise we obfuscate $\widetilde{\mathbf{P}} \leftarrow\!\!\$\, \mathsf{Obf}(1^\lambda, \mathbf{P}_3)$ where $\mathbf{P}_3$ is the circuit that, on input some attribute $x \in \{0,1\}^\ell$ and a string $s \in \{0,1\}^\lambda$, does the following.
  - If $x \neq i$: Same as $\mathbf{P}_2$.
  - If $x = i$: Return 1 if and only if $\mathsf{PRG}(s) = K$, where $K \leftarrow\!\!\$\, \{0,1\}^{\lambda \cdot \ell}$.

  Note that if $Q^*(i) \neq 1$, then the distribution induced by this hybrid is identical to the previous one, so we only consider the case where $Q^*(i) = 1$. Observe that an admissible adversary never queries the key generation oracle on i. Thus, the key $\tilde{k}$ is not present in the view of the distinguisher. Indistinguishability follows from the pseudorandomness of PRG.

- Hybrid $\mathcal{H}_{i,6}$: Here we again check whether $Q^*(i) = 1$. If this is not the case, then we proceed as before. Otherwise we compute $\widetilde{\mathbf{P}} \leftarrow\!\!\$\, \mathsf{Obf}(1^\lambda, \mathbf{P}^*)$ where $\mathbf{P}^*$ is the circuit that, on input some attribute $x \in \{0,1\}^\ell$ and a string $s \in \{0,1\}^\lambda$, does the following.
  - If $x \neq i$: Same as $\mathbf{P}_3$.
  - If $x = i$: Return 0.

  Note that the programs $\mathbf{P}_3$ and $\mathbf{P}^*$ are identical except if K falls within the range of PRG. Since this happens only with negligible probability, then the two hybrids are computationally indistinguishable by the security of the obfuscator Obf.

- Hybrid $\mathcal{H}_{i,7}$: In this hybrid we check whether $Q^*(i) = 1$. If this is not the case, then we proceed as before. Otherwise, we compute the challenge ciphertext as $\widetilde{\mathbf{E}} \leftarrow\!\!\$\, \mathsf{Obf}(1^\lambda, \mathbf{E}^*)$, where $\mathbf{E}^*$ takes as input a pair (x, s) and does the following.
  - If $x < i$: Same as $\mathbf{E}_3$.
  - If $x = i$: Check whether $\widetilde{\mathbf{P}}(x, s) = 1$ and return $\mathsf{WE.Enc}(1^\lambda, (Q, x), 1; \tilde{r})$ if this is the case, where $\tilde{r} \leftarrow\!\!\$\, \{0,1\}^\lambda$.
  - If $x > i$: Same as $\mathbf{E}_3$.

  Observe that at this point $\widetilde{\mathbf{P}}$ always returns 0 whenever queried on i, and thus the programs $\mathbf{E}_3$ and $\mathbf{E}^*$ are functionally equivalent. Indistinguishability follows from the security of Obf.

- Hybrid $\mathcal{H}_{i,8}$: We revert the change done in $\mathcal{H}_{i,6}$.

- Hybrid $\mathcal{H}_{i,9}$: We revert the change done in $\mathcal{H}_{i,5}$.

- Hybrid $\mathcal{H}_{i,10}$: We revert the change done in $\mathcal{H}_{i,4}$.

- Hybrid $\mathcal{H}_{i,11}$: We revert the change done in $\mathcal{H}_{i,3}$.
- Hybrid $\mathcal{H}_{i,12}$: We revert the change done in $\mathcal{H}_{i,1}$.
- Hybrid $\mathcal{H}_{i,13}$: We revert the change done in $\mathcal{H}_{i,0}$.

We denote by $\mathcal{H}_1$ the last hybrid of the sequence $\mathcal{H}_{2^\ell,13}$. Observe that such an hybrid is identical to the original experiment with the bit b fixed to $b = 1$. This concludes our proof. □

Remark 7.5. *We remark that using null-iO instead of witness encryption in the above scheme would allow us to achieve a stronger security definition where the attributes are also hiding to the eyes of parties that have keys for non-accepting predicates (one-sided attribute hiding). In Appendix A we show an alternative transformation from any ABE scheme to one-sided attribute-hiding ABE, additionally assuming the post-quantum hardness of the LWE problem.*

8 Secret Sharing for Monotone QMA

In the following we show how a witness encryption scheme for QMA directly implies a secret sharing scheme for monotone QMA (mQMA). A language $\mathcal{L} = (\mathcal{L}_{\text{yes}}, \mathcal{L}_{\text{no}})$ is in monotone QMA if $\mathcal{L} \in \mathsf{QMA}$ and i) for all statements $x \in \mathcal{L}_{\text{yes}}$ and y such that $x \subseteq y$ it holds that $y \in \mathcal{L}_{\text{yes}}$, and ii) for all statements $x \in \mathcal{L}_{\text{no}}$ and y such that $y \subseteq x$ it holds that $y \in \mathcal{L}_{\text{no}}$. Here, by $x \subseteq y$ for binary strings x, y, we mean that for each index i, if $x_i = 1$ then $y_i = 1$.

8.1 Definition

We begin by defining the notion of secret sharing for languages monotone QMA. For convenience, we only define the scheme for binary secrets, although it is easy to extend it to arbitrary length strings.

Definition 8.1 (Secret Sharing)**.** *A secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *for* N *parties and a language* $\mathcal{L} \in \mathsf{mQMA}$ *consists of the following efficient algorithms.*

- $\mathsf{Share}(1^\lambda, s)$*: On input the security parameter* 1^λ *and a secret* $s \in \{0,1\}$*, the sharing algorithm returns a set of* N *shares* $(p_1, \ldots, p_N)$*.*
- $\mathsf{Rec}(p_1, \ldots, p_{|I|}, |\psi\rangle^{\otimes k(\lambda)})$*: On input a set* I *of shares* $(p_1, \ldots, p_{|I|})$ *and a* $k(\lambda)$ *copies of a quantum state* $|\psi\rangle$*, the reconstruction algorithm returns a secret* s*.*

Next we define the notion of correctness. We say that a set of parties I is qualified if its binary representation defines a statement $x \in \mathcal{L}_{\mathsf{yes}}$.

Definition 8.2 (Correctness)**.** *A secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *is correct if there exists a negligible function* ν *such that for all* $\lambda \in \mathbb{N}$*, all secrets* $s \in \{0,1\}$*, all qualified sets of parties* I *that define a statement* $x \in \mathcal{L}_{\mathsf{yes}}$*, and all* $|\psi\rangle \in \mathsf{R}_{\mathcal{L}}(x)$*, it holds that*

$$\Pr\left[s = \mathsf{Rec}(p_1, \ldots, p_{|I|}, |\psi\rangle^{\otimes k(\lambda)})\right] = 1 - \nu(\lambda)$$

where $(p_1, \ldots, p_N) \leftarrow\!\!\$\ \mathsf{Share}(1^\lambda, s)$*.*

Finally we define the notion of (non-uniform) computational security of a secret sharing scheme. In the following we say that a set S is unauthorized if it defines a statement $x \in \mathcal{L}_{\mathsf{no}}$.

Definition 8.3 (Security)**.** *A secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *is secure if for all* $\lambda \in \mathbb{N}$ *and all unauthorized sets* $S \subseteq \{1, \ldots, N\}$*, it holds that*

$$\{p_{i,0}\}_{i \in S} \approx_c \{p_{i,1}\}_{i \in S}$$

where $(p_{1,0}, \ldots, p_{N,0}) \leftarrow\!\!\$\ \mathsf{Share}(1^\lambda, 0)$ *and* $(p_{1,1}, \ldots, p_{N,1}) \leftarrow\!\!\$\ \mathsf{Share}(1^\lambda, 1)$*.*

8.2 Perfectly Binding Commitments

A (non-interactive) commitment scheme Com is a PPT algorithm that takes as input a message m and returns a commitment c. We require that it satisfies the notions of perfect binding and computational hiding, which we define below.

Definition 8.4 (Perfect Binding)**.** *A commitment scheme* Com *is perfectly binding if for all* (m_0, m_1) *such that* $m_0 \neq m_1$ *and all* $(r_0, r_1) \in \{0,1\}^{2\lambda}$ *it holds that* $\mathsf{Com}(m_0; r_0) \neq \mathsf{Com}(m_1; r_1)$*.*

Definition 8.5 (Computational Hiding)**.** *A commitment scheme* Com *is computationally hiding if for all* $\lambda \in \mathbb{N}$ *and all* (m_0, m_1) *it holds that*

$$\mathsf{Com}(m_0; r_0) \approx_c \mathsf{Com}(m_1; r_1)$$

where $(r_0, r_1) \leftarrow\!\!\$\, \{0,1\}^{2\lambda}$*.*

8.3 Construction

We describe our secret sharing scheme $(\mathsf{Share}, \mathsf{Rec})$ for any language $\mathcal{L} = (\mathcal{L}_{\text{yes}}, \mathcal{L}_{\text{no}}) \in \mathsf{mQMA}$ in the following. Let Com be a perfectly binding commitment scheme and let $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ be a witness encryption for the language $\tilde{\mathcal{L}} = (\tilde{\mathcal{L}}_{\text{yes}}, \tilde{\mathcal{L}}_{\text{no}})$ defined as

$$\tilde{\mathcal{L}}_{\text{yes}} = \left\{ (c_1, \ldots, c_N) : \begin{array}{c} \exists \text{ a vector } (r_1, \ldots, r_N) \in \{0,1\}^{N\lambda} \text{ such that } x \in \mathcal{L}_{\text{yes}} \\ \text{where } x_i = 1 \text{ if } c_i = \mathsf{Com}(i; r_i) \text{ and } x_i = 0 \text{ otherwise.} \end{array} \right\}$$

$$\tilde{\mathcal{L}}_{\text{no}} = \left\{ (c_1, \ldots, c_N) : \begin{array}{c} \forall \text{ vectors } (r_1, \ldots, r_N) \in \{0,1\}^{N\lambda} \text{ it holds that } x \in \mathcal{L}_{\text{no}} \\ \text{where } x_i = 1 \text{ if } c_i = \mathsf{Com}(i; r_i) \text{ and } x_i = 0 \text{ otherwise.} \end{array} \right\}$$

Our scheme $(\mathsf{Share}, \mathsf{Rec})$ is shown in Figure 5.

Secret Sharing for mQMA

- $\mathsf{Share}(1^\lambda, s)$:
  - For all $i \in [1, \ldots, N]$ sample $r_i \leftarrow\!\!\$\, \{0,1\}^\lambda$ and compute $c_i = \mathsf{Com}(i; r_i)$.
  - Compute $c \leftarrow\!\!\$\, \mathsf{WE.Enc}(1^\lambda, (c_1, \ldots, c_N), s)$.
  - Set the share of the i-th party to $p_i = (r_i, c)$.
- $\mathsf{Rec}(p_1, \ldots, p_{|I|}, |\psi\rangle^{\otimes k(\lambda)})$:
  - A subset $I \subseteq \{1, \ldots, N\}$ of parties parses their shares as $\{p_i = (r_i, c)\}_{i \in I}$.
  - For all $i \notin I$ set $r_i = \perp$.
  - Return $\mathsf{WE.Dec}(c, (r_1, \ldots, r_n, |\psi\rangle^{\otimes k(\lambda)}), (c_1, \ldots, c_N))$.

Figure 5: A secret sharing scheme for (monotone) QMA

Correctness. For the correctness of the scheme, observe that any authorized set of users always has a valid witness for the statement $(c_1, \ldots, c_N)$ and therefore, by the correctness of the witness encryption scheme, the reconstruction procedure returns the secret s, except with negligible probability.

Security. In the following we argue about the (non-uniform) security of the scheme.

Theorem 8.6 (Security)**.** *Let* Com *be a computationally hiding commitment scheme and let* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *be a witness encryption scheme for* $\tilde{\mathcal{L}}$*. Then the scheme in Figure 5 is secure.*

Proof. To prove this claim, we assume towards contradiction that there exists an efficient quantum algorithm that is able to distinguish between the shares of 0 and the shares of 1 given the shares of some non-authorized set of users S with probability $1/2 + \delta(\lambda)$, for some non-negligible function δ. We derive a reduction against the computational hiding property of the commitment Com. Specifically, we show that such an algorithm implies a distinguisher between the following distributions

$$(\mathsf{Com}(1), \ldots, \mathsf{Com}(N)) \text{ and } (\mathsf{Com}(\bot), \ldots, \mathsf{Com}(\bot))$$

where $\bot$ is some distinguished string that does not correspond to any index. This implies a contradiction to the hiding property of the commitment scheme, by a standard hybrid argument.

On input the challenge set of commitments $(c_1, \ldots, c_N)$, the reduction defines a new set of commitments $(c'_1, \ldots, c'_N)$ by setting $c'_i = c_i$ if $i \notin S$ and $c_i = \mathsf{Com}(i; r_i)$ if $i \in S$, where $r_i \leftarrow\!\!\$\, \{0,1\}^\lambda$. Finally, it samples a bit $b \leftarrow\!\!\$\, \{0,1\}$ and computes $c \leftarrow\!\!\$\, \mathsf{WE.Enc}(1^\lambda, (c'_1, \ldots, c'_N), b)$. The distinguisher is given the shares $\{r_i, c\}_{i \in S}$ and returns a bit b'. The reduction returns 1 if $b = b'$ and 0 otherwise.

In the first case, i.e. $(c_1, \ldots, c_N) = (\mathsf{Com}(1), \ldots, \mathsf{Com}(N))$ then the distribution of the shares given to the ditinguisher is identical to the one output by the algorithm Share and therefore the probability that the reduction outputs 1 is identical to $1/2 + \delta(\lambda)$. In the second case, where $(c_1, \ldots, c_N) = (\mathsf{Com}(\bot), \ldots, \mathsf{Com}(\bot))$, observe that the resulting instance $(c'_1, \ldots, c'_N) \in \tilde{\mathcal{L}}_{\mathsf{no}}$ since the commitment scheme is perfectly binding and the set S is non-authorized, i.e. it holds that $x \in \mathcal{L}_{\mathsf{no}}$ (and consequently that $z \in \mathcal{L}_{\mathsf{no}}$, for all $z \subseteq x$), where x is the statement defined by S. By the semantic security of the witness encryption scheme it holds that the probability the distinguisher correctly guesses the bit b (and consequently the reduction outputs 1) is negligibly close to $1/2$. It follows that the reduction successfully distinguishes between the two distributions of commitments with non-negligible probability, which is a contradiction to the hiding property of the commitment scheme. □

References

[Aar05] Scott Aaronson. Ten semi-grand challenges for quantum computing theory, 2005.

[ABDS20] Gorjan Alagic, Zvika Brakerski, Yfke Dulek, and Christian Schaffner. Impossibility of quantum virtual black-box obfuscation of classical circuits, 2020.

[ACGH20] Gorjan Alagic, Andrew M. Childs, Alex B. Grilo, and Shih-Han Hung. Non-interactive classical verification of quantum computation. LNCS, pages 153–180. Springer, Heidelberg, March 2020.

[AF16] Gorjan Alagic and Bill Fefferman. On quantum obfuscation. *CoRR*, abs/1602.01771, 2016.

[AJJ14] Gorjan Alagic, Stacey Jeffery, and Stephen Jordan. Circuit Obfuscation Using Braids. In Steven T. Flammia and Aram W. Harrow, editors, *9th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2014)*, volume 27 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 141–160, Dagstuhl, Germany, 2014. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[AMTDW00] Andris Ambainis, Michele Mosca, Alain Tapp, and Ronald De Wolf. Private quantum channels. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 547–553. IEEE, 2000.

[AP20] Prabhanjan Ananth and Rolando L. La Placa. Secure software leasing, 2020.

[BDGM20a] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate iO from homomorphic encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 79–109. Springer, Heidelberg, May 2020.

[BDGM20b] 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. https://eprint.iacr.org/2020/1024.

[BFJ+20] Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Statistical ZAP arguments. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 642–667. Springer, Heidelberg, May 2020.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.

[BG19] Anne Broadbent and Alex B. Grilo. Zero-knowledge for qma from locally simulatable proofs, 2019.

[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 Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Heidelberg, August 2001.

[BJ15] Anne Broadbent and Stacey Jeffery. Quantum homomorphic encryption for circuits of low T-gate complexity. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 609–629. Springer, Heidelberg, August 2015.

[BK20] Anne Broadbent and Raza Ali Kazmi. Indistinguishability obfuscation for quantum circuits of low t-count. Cryptology ePrint Archive, Report 2020/639, 2020. https://eprint.iacr.org/2020/639.

[BOV03] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 299–315. Springer, Heidelberg, August 2003.

[BP15] Nir Bitansky and Omer Paneth. ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 401–427. Springer, Heidelberg, March 2015.

[BPR15] Nir Bitansky, Omer Paneth, and Alon Rosen. On the cryptographic hardness of finding a Nash equilibrium. In Venkatesan Guruswami, editor, *56th FOCS*, pages 1480–1498. IEEE Computer Society Press, October 2015.

[BPW16] Nir Bitansky, Omer Paneth, and Daniel Wichs. Perfect structure on the edge of chaos - trapdoor permutations from indistinguishability obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 474–502. Springer, Heidelberg, January 2016.

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

[BS20] Nir Bitansky and Omri Shmueli. Post-quantum zero knowledge in constant rounds. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *52nd ACM STOC*, pages 269–279. ACM Press, June 2020.

[CCY20] Nai-Hui Chia, Kai-Min Chung, and Takashi Yamakawa. Classical verification of quantum computations with efficient verifier. LNCS, pages 181–206. Springer, Heidelberg, March 2020.

[CVZ20] Andrea Coladangelo, Thomas Vidick, and Tina Zhang. Non-interactive zero-knowledge arguments for QMA, with preprocessing. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 799–828. Springer, Heidelberg, August 2020.

[DN00] Cynthia Dwork and Moni Naor. Zaps and their applications. In *41st FOCS*, pages 283–293. IEEE Computer Society Press, November 2000.

[FHM18] Joseph F. Fitzsimons, Michal Hajdušek, and Tomoyuki Morimae. Post hoc verification of quantum computation. *Physical Review Letters*, 120(4), Jan 2018.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[GGG+14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 578–602. Springer, Heidelberg, May 2014.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013.

[GJJM20] Vipul Goyal, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Statistical zaps and new oblivious transfer protocols. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 668–699. Springer, Heidelberg, May 2020.

[GKVW20] Rishab Goyal, Venkata Koppula, Satyanarayana Vusirikala, and Brent Waters. On perfect correctness in (lockable) obfuscation. In *TCC 2020, Part I*, LNCS, pages 229–259. Springer, Heidelberg, March 2020.

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

[GP20] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. Cryptology ePrint Archive, Report 2020/1010, 2020. `https://eprint.iacr.org/2020/1010`.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 503–523. Springer, Heidelberg, August 2015.

[Had00] Satoshi Hada. Zero-knowledge and code obfuscation. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 443–457. Springer, Heidelberg, December 2000.

[KKS18] Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Statistical witness indistinguishability (and more) in two messages. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 34–65. Springer, Heidelberg, April / May 2018.

[KNY14] Ilan Komargodski, Moni Naor, and Eylon Yogev. Secret-sharing for NP. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 254–273. Springer, Heidelberg, December 2014.

[LVW20] Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Statistical ZAPR arguments from bilinear maps. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 620–641. Springer, Heidelberg, May 2020.

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

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

[Mor20] Tomoyuki Morimae. Information-theoretically-sound non-interactive classical verification of quantum computing with trusted center, 2020.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990.

[PRS17] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-LWE for any ring and modulus. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th ACM STOC*, pages 461–473. ACM Press, June 2017.

[PRV12] Bryan Parno, Mariana Raykova, and Vinod Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 422–439. Springer, Heidelberg, March 2012.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[Shm20] Omri Shmueli. Multi-theorem (malicious) designated-verifier nizk for qma. Cryptology ePrint Archive, Report 2020/928, 2020.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

[WW20] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious lwe sampling. Cryptology ePrint Archive, Report 2020/1042, 2020. `https://eprint.iacr.org/2020/1042`.

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

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

A One-Sided Attribute-Hiding

In the following we show how to convert generically an ABE for BQP into a predicate encryption scheme with one-sided security (i.e. where the quantum circuit associated with a ciphertext is hidden to the eyes of an adversary that cannot decrypt), additionally assuming the quantum hardness of the LWE problem. Towards this goal, we introduce the notion of quantum lockable obfuscation and we present a scheme from quantum-hard LWE.

A.1 Definition

We define the notion of one-sided attribute-hiding for attribute based encryption. In the literature, this primitive is also referred to as predicate encryption [GVW15]. Since the syntax is unchanged, we only present the upgraded security definition. Similarly as before, we consider the case of selective security, where the quantum circuit associated with the challenge ciphertext is fixed ahead of time, and in particular is chosen before seeing the public parameters of the scheme.

Definition A.1 (One-Sided Attribute-Hiding)**.** *An ABE scheme* $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ *is one-sided attribute-hiding if there exists a triple of negligible functions* ν_0*,* ν_1*, and* μ *such that for all* $\lambda \in \mathbb{N}$*, all pairs of quantum circuits* (Q_0, Q_1)*, and all admissible non-uniform QPT distinguishers with quantum advice* $\mathcal{A} = \{\mathcal{A}_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$*, it holds that*

$$\Pr\left[b = \mathcal{A}_\lambda(\mathsf{ct}_{Q_b}, \mathsf{mpk}; \rho_\lambda)^{\mathsf{ABE.KeyGen}(\mathsf{msk},\cdot)} : \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow\!\!\$\ \mathsf{ABE.Gen}(1^\lambda, 1^\ell)\\ b \leftarrow\!\!\$\ \{0,1\}\\ \mathsf{ct}_{Q_b} \leftarrow\!\!\$\ \mathsf{ABE.Enc}(\mathsf{mpk}, Q_b, b)\end{array}\right] = 1/2 + \mu(\lambda),$$

where $\mathcal{A}$ *is admissible if each query* x *to* $\mathsf{ABE.KeyGen}(\mathsf{msk},\cdot)$ *is such that* $\Pr[Q_0(x) = 1] = \nu_0(\lambda)$ *and* $\Pr[Q_1(x) = 1] = \nu_1(\lambda)$*.*

A.2 Quantum Lockable Obfuscation

In the following we give a construction of quantum lockable obfuscation assuming the quantum hardness of the LWE problem.

Compute-and-Compare Programs. We define the class of compute-and-compare circuits. The definition below applies both to classical and pseudo-deterministic quantum circuits (with classical input and output).

Definition A.2 (Compute-and-Compare)**.** *Let* $C : \{0,1\}^n \to \{0,1\}^\lambda$ *be a circuit, and let* $u \in \{0,1\}^\lambda$ *and* $z \in \{0,1\}^*$ *be two classical strings. Then* $\mathbf{CC}[C, u, z](x)$ *is a circuit that returns* z *if* $C(x) = u$*, and* 0 *otherwise.*

Definition. We are now ready to define the notion of lockable obfuscation for compute-and-compare programs. In what follows we only define the classical version of lockable obfuscation. The extension to quantum circuits follows along the same lines. A lockable obfuscator Obf is a PPT algorithm that takes as input a compute-and-compare program $\mathbf{CC}[C, u, z]$ and outputs a new circuit $\widetilde{\mathbf{CC}}$. We assume that the circuit $\mathbf{CC}[C, u, z]$ is given in some canonical description from which C, u, and z can be read. Correctness is defined as follows.

Definition A.3 (Correctness)**.** *A lockable obfuscator* Obf *is correct if there exists a negligible function* ν *such that for all* $\lambda \in \mathbb{N}$*, all circuits* $C : \{0,1\}^n \to \{0,1\}^\lambda$*, all* $u \in \{0,1\}^\lambda$*, and all* $z \in \{0,1\}^*$*, it holds that*

$$\Pr\left[\forall x \in \{0,1\}^n : \widetilde{\mathbf{CC}}(x) = \mathbf{CC}[C, u, z](x)\right] = 1 - \nu(\lambda)$$

where $\widetilde{\mathbf{CC}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{CC}[C, u, z])$*.*

We require a strong notion of simulation security for lockable obfuscation.

Definition A.4 (Simulation Security)**.** *A lockable obfuscator* Obf *is secure if there exists a simulator* Sim *such that for all* $\lambda \in \mathbb{N}$*, all pseudo-deterministic polynomial-size quantum circuits* $C : \{0,1\}^n \to \{0,1\}^\lambda$*, and all polynomial-length output strings* $z \in \{0,1\}^*$ *it holds that*

$$\mathsf{Obf}(1^\lambda, \mathbf{CC}[C, u, z]) \approx_c \mathsf{Sim}(1^\lambda, 1^{|C|}, 1^{|z|})$$

where $u \leftarrow\!\!\$\ \{0,1\}^\lambda$.

Classical Lockable Obfuscation. Lockable obfuscation for classical circuits and with almost perfect correctness were constructed in [GKW17, WZ17], assuming the quantum hardness of LWE. Recently, a construction with perfect correctness has been shown in [GKVW20].

Lemma A.5 ([GKW17, WZ17])**.** *Assuming the quantum hardness of the LWE problem, there exists a lockable obfuscation* Obf *for compute-and-compare programs.*

Quantum Lockable Obfuscation. We now propose a lockable obfuscation scheme for pseudo-deterministic compute-and-compare *quantum programs*. The scheme combines a QFHE scheme $(\mathsf{QFHE.Gen}, \mathsf{QFHE.Enc}, \mathsf{QFHE.Eval}, \mathsf{QFHE.Dec})$ with classical keys and classical decryption with a lockable obfuscator Obf for classical circuits. The scheme Obf^* is described in Figure 6.

Quantum Lockable Obfuscation

- $\mathsf{Obf}^*(1^\lambda, \mathbf{CC}[Q, u, z])$:
  - Sample $(\mathsf{sk}, \mathsf{pk}) \leftarrow\!\!\$\ \mathsf{QFHE.Gen}(1^\lambda)$.
  - Compute the classical encryption $c \leftarrow\!\!\$\ \mathsf{QFHE.Enc}(\mathsf{pk}, Q)$ of the circuit Q.
  - Compute the obfuscation $\widetilde{\mathbf{CC}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathbf{CC}[\mathsf{QFHE.Dec}(\mathsf{sk}, \cdot), u, z])$ of the classical compute-and-compare program that takes as input a ciphertext c^* and returns z if and only if $\mathsf{QFHE.Dec}(\mathsf{sk}, c^*) = u$.
  - Return $(c, \widetilde{\mathbf{CC}})$.

To evaluate the obfuscated circuit $(c, \widetilde{\mathbf{CC}})$ on some input x, the (quantum) algorithm computes $\tilde{c} \leftarrow\!\!\$\ \mathsf{QFHE.Eval}(\mathsf{pk}, \mathcal{U}_x, c)$ as the homomorphic evaluation of the universal quantum circuit $\mathcal{U}_x$ that has hard-wired input x and evaluates the encrypted circuit Q homomorphically. Then return the evaluation of $\widetilde{\mathbf{CC}}$ on input the resulting $\tilde{c}$.

Figure 6: A lockable obfuscator for (pseudo-deterministic) compute-and-compare quantum circuits

Correctness follows straightforwardly from the correctness of the underlying building blocks. One caveat is that the QFHE evaluation procedure is inherently probabilistic, so correctness only holds with probability negligibly close to one (over the randomness imposed by the QFHE evaluation). We now proceed to establish the simulation security of the scheme.

Theorem A.6 (Simulation Security)**.** *Let* $(\mathsf{QFHE.Gen}, \mathsf{QFHE.Enc}, \mathsf{QFHE.Eval}, \mathsf{QFHE.Dec})$ *be a secure QFHE scheme and let* Obf *be a simulation secure lockable obfuscator for classical compute-and-compare programs. Then the scheme in Figure 6 is simulation secure.*

Proof. To prove the security of our construction, we define a series of hybrid distributions that we argue to be computationally close. The real distribution for an obfuscated quantum circuit Q consists of

$$(\mathsf{QFHE.Enc}(\mathsf{pk}, Q), \mathsf{Obf}(1^\lambda, \mathbf{CC}[\mathsf{QFHE.Dec}(\mathsf{sk}, \cdot), u, z])).$$

In the first hybrid distribution we simulate the obfuscated classical circuit. Since u is uniformly sampled (and in particular is independent of all other variables of our distribution), by an invocation of the simulation security of Obf, we obtain that

$$\begin{aligned}&(\mathsf{QFHE.Enc}(\mathsf{pk}, Q), \mathsf{Obf}(1^\lambda, \mathbf{CC}[\mathsf{QFHE.Dec}(\mathsf{sk}, \cdot), u, z]))\\ &\approx_c (\mathsf{QFHE.Enc}(\mathsf{pk}, Q), \mathsf{Sim}(1^\lambda, 1^{|\mathsf{QFHE.Dec}|}, 1^{|z|})).\end{aligned}$$

Next, we switch the QFHE to be an encryption of $0^{|U|}$. By the semantic security of the QFHE scheme we have that

$$\begin{aligned}&(\mathsf{QFHE.Enc}(\mathsf{pk}, Q), \mathsf{Sim}(1^\lambda, 1^{|\mathsf{QFHE.Dec}|}, 1^{|z|}))\\ &\approx_c (\mathsf{QFHE.Enc}(\mathsf{pk}, 0^{|Q|}), \mathsf{Sim}(1^\lambda, 1^{|\mathsf{QFHE.Dec}|}, 1^{|z|})).\end{aligned}$$

Note that the latter distribution does not contain any information about the circuit Q, besides its size. This concludes our proof. □

A.3 Construction

We now present our transformation to upgrade the security of an ABE scheme for quantum circuit to one-sided attribute-hiding. The transformation is identical to [WZ17, GKW17] and we describe it here for completeness. We require the existence of an ABE scheme $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ which (for convenience) we assume that it can encrypt multiple bits, and a lockable obfuscator Obf for compute-and-compare quantum circuits. The scheme Obf^* is described in Figure 7.

One-Sided Attribute-Hiding ABE

- $\mathsf{ABE.Gen}^*(1^\lambda, 1^\ell)$:
  - Return $\mathsf{ABE.Gen}(1^\lambda, 1^\ell)$.
- $\mathsf{ABE.Enc}^*(\mathsf{mpk}, Q, m)$:
  - Sample a uniform $u \leftarrow\$ \{0,1\}^\lambda$.
  - Compute $c \leftarrow\$ \mathsf{ABE.Enc}(\mathsf{mpk}, Q, u)$.
  - Compute the obfuscation $\widetilde{\mathbf{CC}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{CC}[\mathsf{ABE.Dec}(\cdot, c), u, m])$ of the quantum compute-and-compare program that takes as input a key sk_x and returns m if and only if $\mathsf{ABE.Dec}(\mathsf{sk}_x, c) = u$.
  - Return $c_Q = \widetilde{\mathbf{CC}}$.
- $\mathsf{ABE.KeyGen}^*(\mathsf{msk}, x)$:
  - Return $\mathsf{ABE.KeyGen}(\mathsf{msk}, x)$.
- $\mathsf{ABE.Dec}^*(\mathsf{sk}_x, \mathsf{ct}_Q)$:
  - Return the output of the evaluation of $\widetilde{\mathbf{CC}}$ on sk_x.

Figure 7: A one-sided attribute-hiding ABE scheme

Correctness. The ABE decryption circuit is by definition pseudo-deterministiic and therefore the correctness of the scheme is routinely established by invoking the correctness of the ABE and of the obfuscator for compute-and-compare quantum programs.

Security. We now analyze the security of the scheme, which we establish with the following theorem.

Theorem A.7 (One-Sided Attribute-Hiding)**.** *Let* $(\mathsf{ABE.Gen}, \mathsf{ABE.Enc}, \mathsf{ABE.KeyGen}, \mathsf{ABE.Dec})$ *be a secure ABE scheme and let* Obf *be a simulation secure obfuscator for compute-and-compare quantum programs. Then the scheme in Figure 7 is one-sided attribute-hiding.*

Proof. The proof proceeds by defining the following series of hybrids.

- Hybrid $\mathcal{H}_0$: This is the original experiment with the bit fixed to $b = 0$.
- Hybrid $\mathcal{H}_1$: This is identical to the previous hybrid except that in the challenge ciphertext we compute $c \leftarrow\$ \mathsf{ABE.Enc}(\mathsf{pk}, Q_0, 0^\lambda)$.

  Indisinguishability follows from the (standard) security of the ABE scheme.
- Hybrid $\mathcal{H}_2$: In this hybrid we compute the challenge ciphertext via the simulator $\mathsf{Sim}(1^\lambda, 1^{|\mathsf{ABE.Dec}|}, 1^{|m|}))$.

  Note that in $\mathcal{H}_1$ the target value u is uniform to the eyes of the distinguisher and therefore indistinguishbility follows from the simulation security of the compute-and-compare obfuscator.
- Hybrid $\mathcal{H}_3$: In this hybrid we modify the challenge ciphertext by sampling $c \leftarrow\$ \mathsf{ABE.Enc}(\mathsf{pk}, Q_1, 0^\lambda)$.

  Since c does not affect the view of the adversary this modification is only syntactical.
- Hybrid $\mathcal{H}_4$: We revert the change done in $\mathcal{H}_2$, except that we compute the obfuscated circuit as $\widetilde{\mathbf{CC}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{CC}[\mathsf{ABE.Dec}(\cdot, c), u, m_1])$.

  Indistinguishability follows along the same lines as what argued above.
- Hybrid $\mathcal{H}_4$: We revert the change done in $\mathcal{H}_1$, except that we compute $c \leftarrow\$ \mathsf{ABE.Enc}(\mathsf{pk}, Q_1, u)$.

  Indistinguishability follows from another invocation of the security of the ABE scheme.

The proof is concluded by observing that the distribution induced by $\mathcal{H}_4$ is identical to the original experiment with the bit fixed to $b = 1$. □