Classical Obfuscation of Quantum Circuits via Publicly-Verifiable QFHE

James Bartusek
Columbia

Aparna Gupte*
MIT

Saachi Mutreja
Columbia

Omri Shmueli
NTT Research

October 10, 2025

Abstract

A classical obfuscator for quantum circuits is a classical program that, given the classical description of a quantum circuit Q, outputs the classical description of a functionally equivalent quantum circuit $\widetilde{Q}$ that hides as much as possible about Q. Previously, the *only* known feasibility result for classical obfuscation of quantum circuits (Bartusek and Malavolta, ITCS 2022) was limited to "null" security, which is only meaningful for circuits that always reject. On the other hand, if the obfuscator is allowed to compile the quantum circuit Q into a *quantum state* $|\widetilde{Q}\rangle$, there exist feasibility results for obfuscating much more expressive classes of circuits: All pseudo-deterministic quantum circuits (Bartusek, Kitagawa, Nishimaki and Yamakawa, STOC 2023, Bartusek, Brakerski and Vaikuntanathan, STOC 2024), and even all unitaries (Huang and Tang, FOCS 2025).

We show that (relative to a classical oracle) there exists a *classical* obfuscator for all pseudo-deterministic quantum circuits. As our main technical step, we give the first construction of a compact quantum fully-homomorphic encryption (QFHE) scheme that supports public verification of (pseudo-deterministic) quantum evaluation, relative to a classical oracle.

To construct our QFHE scheme, we improve on an approach introduced by Bartusek, Kitagawa, Nishimaki and Yamakawa (STOC 2023), which previously required ciphertexts that are both *quantum* and *non-compact* due to a heavy use of quantum coset states and their publicly-verifiable properties. As part of our core technical contribution, we introduce new techniques for analyzing coset states that can be generated "on the fly", by proving new cryptographic properties of the one-shot signature scheme of Shmueli and Zhandry (CRYPTO 2025). Our techniques allow us to produce QFHE ciphertexts that are purely classical, compact, and publicly-verifiable. This additionally yields the first classical verification of quantum computation protocol for BQP that simultaneously satisfies blindness and public-verifiability.

*This work was done in part while the author was at the Simons Institute and participating in the Challenge Institute for Quantum Computation at UC Berkeley.

Contents

1 Introduction

Program obfuscation disguises the inner workings of an algorithm while leaving its behavior unchanged. This concept traces back to the roots of modern cryptography [DH22], and ever since the formalization of *indistinguishability* obfuscation (iO) [BGI+01], constructing iO has been one of cryptography's overarching goals. It took no less than a decade to come up with the first candidates for secure iO [GGH+16], and almost an additional decade to base classically-secure iO on well-studied assumptions [JLS21]. Current efforts continue along this trajectory, with an active line of work devoted to developing quantum-secure iO, eventually grounded in similarly well-understood assumptions (e.g. [GGH15, BGMZ18, CVW18, APM20, BDGM23, WW21],...).

The original notion of program obfuscation implicitly considers computation as classical, and accordingly, seeks to obfuscate classical algorithms. Now, as quantum computation increasingly emerges as a standard model of computation, it is natural to revisit the problem in this broader context: Can quantum algorithms be obfuscated as well?

Ideal Program Obfuscation: From Classical to Quantum. Our understanding of obfuscation in the quantum setting is still at a very early stage: In particular, we still don't know how to base indistinguishability obfuscation for expressive classes of quantum circuits on (post-quantum) indistinguishability obfuscation for classical circuits.[1] Instead, recent work has turned to exploring the relationship between *idealized* notions of obfuscation in the classical and quantum worlds [BM22, BKNY23, BBV24, HT25]. In such an idealized model, also referred to as *virtual black-box* (VBB) obfuscation, an obfuscation of a circuit C is considered secure if everything that can be learned from the obfuscation $\widetilde{C}$ can be simulated given only *oracle access* to C.

Of course, ideal obfuscation cannot be achieved in full generality, as there are well-known counterexamples demonstrating its impossibility [BGI+01]. Nevertheless, studying the connection between classical and quantum ideal obfuscation remains valuable for at least two reasons. First, it isolates a fundamental question: Does the existence of an ideal obfuscator for classical computation imply one for quantum? Second, indistinguishability obfuscation is widely regarded as a strong heuristic for ideal obfuscation (e.g., it is the best-possible obfuscation [GR07]). Thus, establishing that ideal classical obfuscation gives rise to ideal quantum obfuscation results in the first real-world candidates for quantum obfuscation by instantiating the black-box with (post-quantum) indistinguishability obfuscation.

Prior work [BKNY23, BBV24, HT25] has established that, assuming a classical ideal obfuscator for classical circuits, one can construct a *quantum* ideal obfuscator for quantum circuits. In this setting, the obfuscator itself is a quantum algorithm, and on input a quantum circuit Q it outputs a quantum state $|\widetilde{Q}\rangle$ serving as the obfuscated program.[2] A distinctive feature of these constructions is that the obfuscated object consists of *unclonable* states: Once $|\widetilde{Q}\rangle$ is produced, it cannot be duplicated. As a consequence, it is impossible to create a single obfuscation of a quantum circuit and distribute it broadly. Moreover, these obfuscations cannot be transmitted over the classical channels that comprise the internet today.

However, the goal of constructing a classical obfuscator for quantum circuits (with security for non-null circuits), even in the ideal setting, has so far remained out of reach. This raises our central motivating question:

Is a quantum obfuscator fundamentally necessary to obfuscate quantum circuits?
Or can a quantum program—classically described—be obfuscated by a classical *obfuscator?*

We address this question by constructing a classical obfuscator for an expressive class of quantum circuits—namely, pseudo-deterministic circuits.

[1] The only existing result from indistinguishability obfuscation is limited to quantum circuits with logarithmically-many non-Clifford gates [BK21].

[2] We note that [BBV24, HT25] are able to do something slightly stronger—they obfuscate quantum circuits that also have an auxiliary quantum state. This is particularly relevant in some cryptographic applications, e.g. copy-protection.

1.1 Classical Obfuscation via Publicly-Verifiable QFHE

Previous work [BKNY23] identified *pseudo-deterministic* quantum circuits as a particularly useful class for obfuscation. A pseudo-deterministic quantum circuit Q from n to m qubits is one that, up to negligible error, computes a classical function: there exists a function $F : \{0,1\}^n \to \{0,1\}^m$ such that for every input $x \in \{0,1\}^n$, the output of $Q(x)$ equals $F(x)$ with probability $1 - \mathsf{negl}(n)$, where the probability is taken over the measurement of its output. This class is both expressive—capturing, for example, all quantum algorithms that solve problems in **NP**, including Shor's factoring algorithm [Sho99]—and structurally well-suited for cryptographic applications like obfuscation. Indeed, pseudo-deterministic circuits have already served as the foundation for obfuscating increasingly general classes of quantum computation [BBV24, HT25]. Motivated by these properties, our work focuses on the obfuscation of pseudo-deterministic circuits. Crucially, we consider obfuscation through only classical means.

As also observed in previous work [BKNY23], a natural route toward obfuscating pseudo-deterministic circuits relies on constructing quantum fully-homomorphic encryption (QFHE) with public verification of homomorphic evaluation for such circuits.

- A QFHE scheme is defined by algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$, where all but the evaluation algorithm Eval can be regarded as classical. On input a ciphertext ct and a quantum circuit Q, the evaluation algorithm Eval produces an evaluated ciphertext $\widetilde{\mathsf{ct}}$, which encrypts a classical value whenever $\mathsf{ct} = \mathsf{Enc}(x)$ is a valid ciphertext.

- In the publicly-verifiable variant of QFHE, Eval additionally outputs a succinct classical proof π, and correctness can be certified by an efficient classical verification algorithm Ver, which checks that the homomorphic evaluation has been honestly computed, and outputs $\mathsf{Ver}(\mathsf{ct}, Q, \widetilde{\mathsf{ct}}, \pi) \in \{\top, \bot\}$.

It is straightforward to see that, given such a powerful QFHE, obfuscation of pseudo-deterministic quantum circuits follows naturally. This can be accomplished by obfuscating the verifier and applying the standard "verify-then-decrypt" paradigm. Concretely, the obfuscation of a circuit Q consists of encrypting its classical description, i.e., $\mathsf{ct}_Q \leftarrow \mathsf{Enc}(Q)$. On input $x \in \{0,1\}^n$ and the obfuscated circuit ct_Q, the evaluator computes $(\widetilde{\mathsf{ct}}, \pi) \leftarrow \mathsf{Eval}(\mathsf{ct}_Q, U_x)$, where U_x denotes the universal quantum circuit with input x hard-wired. To recover the (classical) output $Q(x)$, the obfuscation makes use of a classical oracle implementing the circuit

$$C(x, \widetilde{\mathsf{ct}}, \pi) := \begin{cases} \mathsf{Dec}(\widetilde{\mathsf{ct}}) & \text{if } \mathsf{Ver}(\mathsf{ct}_Q, U_x, \widetilde{\mathsf{ct}}, \pi) = \top \\ \bot & \text{otherwise} \end{cases}$$

In this way, only correctly evaluated ciphertexts are ever decrypted, and the obfuscation securely reproduces the functionality of Q.

The central role of Pauli Functional Commitments. However, it turns out that verification of QFHE evaluations is a particularly delicate problem in quantum cryptography.

- On the one hand, by definition, evaluating QFHE ciphertexts is a quantum *sampling* task. The quantum QFHE evaluation gets the classical ct and outputs a quantumly computed classical sample $\widetilde{\mathsf{ct}}$, which is going to have high entropy even if the underlying evaluated circuit Q is deterministic (and this was shown to be inherent in [Shm22a, Shm22b]). This seemingly presents a barrier, since only results achieving inverse polynomial soundness error are known [CLLW22], and it remains open how to classically verify quantum sampling problems with negligible soundness error.

- On the other hand, the special structure of certain QFHE schemes, most notably Mahadev's construction [Mah20], enables a partial solution. As shown in [Bar21], the use of Mahadev's measurement protocol [Mah18] on Mahadev's QFHE scheme allows for verification of QFHE evaluations with negligible soundness error. However, since the measurement protocol of Mahadev is privately verifiable, then so is the resulting QFHE scheme.

The technical centerpiece behind Mahadev's measurement protocol was later abstracted by [BKNY23] as a Pauli functional commitment (PFC). Roughly speaking, a PFC allows a classical verifier to issue a public classical commitment key CK. A quantum prover can commit to a quantum state in such a way that later measurements of Pauli observables (i.e., plain classical or Hadamard measurements) on the state can be verified through classical communication. Viewed through this lens, the state of the art can be summarized as follows: Mahadev [Mah18] constructs a privately verifiable PFC; Bartusek [Bar21] demonstrates that applying Mahadev's PFCs to QFHE homomorphic evaluation yields privately verifiable evaluations; and finally, [BKNY23] show how to construct, relative to a classical oracle, PFCs with public verification, albeit with a *quantum* commitment key $|\mathsf{CK}\rangle$.

Measurement protocols, or in their formal abstraction as PFCs, were originally devised to allow a classical client to force a quantum server to perform some prescribed computation. Existing results on public verification, however, require the commitment keys to remain quantum, which prevents them from being generated by a classical verifier. This leads us to what we view as a fascinating technical question in its own right: can one design classical measurement protocols that are publicly verifiable? As we have explained, resolving this question would not only advance the theory of quantum cryptography but would also translate directly to our applications, enabling public verification of QFHE evaluations and, ultimately, classical obfuscation of quantum circuits.

1.2 Results

We summarize our main results in this section.

Theorem 1 (Informal)**.** *Assuming the quantum hardness of the Learning With Errors problem, there exists classical obfuscation for (classically-described) pseudo-deterministic quantum circuits in the classical oracle model.*

Along the way, as a tool for our obfuscation scheme, we build the first publicly-verifiable QFHE scheme for pseudo-deterministic circuits.

Theorem 2 (Informal)**.** *Assuming the quantum hardness of the Learning With Errors problem (resp. plus an appropriate circular-security assumption), there exists a leveled (resp. unleveled) publicly-verifiable quantum fully-homomorphic encryption scheme with classical encryption of classical messages that supports homomorphic evaluation of pseudo-deterministic circuits in the classical oracle model.*

Our main technical workhorse is the first publicly-verifiable Pauli functional commitment scheme with *classical keys*, which we describe in some detail in Section 1.4.

As an additional contribution that may be of independent interest, we show that post-quantum *succinct* ideal obfuscation for classical Turing machines exists in the classical oracle model (Section 4). We do this via a natural "FHE + SNARK" approach, and combine techniques from [CMS19, Zha19] in order to establish post-quantum security.

Finally, we note that our quantum obfuscation scheme is in fact *succinct*, meaning that the size of the obfuscated program grows only with the *description size* of the plaintext quantum program, as opposed to the number of physical gates in the circuit. As an immediate corollary, we obtain the first *succinct randomized encoding* scheme for pseudo-deterministic quantum computation. A succinct randomized encoding allows for compiling a circuit Q and an input x into a succinct representation $\widehat{Q(x)}$ in time much less than evaluating Q, such that $\widehat{Q(x)}$ reveals only $Q(x)$ and nothing else about x. Succinct randomized encodings have had several important applications in classical cryptography [BGL+15], and we expect them to be widely useful in the quantum setting as well.

1.3 Comparison with prior work on CVQC

Our publicly-verifiable QFHE protocol naturally yields a new type of classical verification of quantum computation (CVQC) protocol for BQP. Indeed, given some quantum computation Q, the verifier simply encrypts their input x as $\mathsf{Enc}(x)$, and the prover responds with $\mathsf{Enc}(Q(x))$ along with a proof π of correct evaluation.

	Blind	Verification	Succinct	Interactive	Heuristic
[Mah18]	Yes*	Private	No	Yes	No
[BKL+22]	Yes*	Private	Yes	Yes	No
[BM22]	No	Public	Yes	No	Yes
This work	Yes	Public	Yes	No	Yes

Table 1: A comparison among classical verification of quantum computation protocols for BQP. The asterisk in the Blind column refer to the fact that these protocols aren't necessarily presented as blind, but can be made blind by running under QFHE. This cannot be done for [BM22] without sacrificing public verification. The Heuristic column refers to provable security (i.e. reduction to an assumption) vs. heuristic security (concretely, argued in the classical oracle model). Finally, we note that while the protocols listed above are the first to achieve the listed guarantees, there have been several other protocols proposed that improve along other metrics such as the assumption [NZ23, MNZ24, GKNV25] and circuit class [CLLW22].

This protocol has several desirable properties: The prover remains blind to the verifier's input x, the proof π is publicly-verifiable (implying for example that the prover cannot break security by repeatedly querying the verifier and learning whether they accept or reject), and the client's work does not grow with the size of Q. In particular, we obtain the first CVQC protocol that is both *blind* and *publicly-verifiable*, and on top of that it is both non-interactive and succinct. We summarize a comparison with prior work in Table 1.

1.4 Approach

As mentioned above, the main technical workhorse behind our results can be seen as a *publicly-verifiable* measurement protocol with a classical verifier. To this end, we construct relative to a classical oracle *publicly-verifiable* Pauli functional commitments with classical commitment keys. We next recall what are PFCs, discuss previous work, and then present our new techniques.

A publicly-verifiable PFC (Definition 5) is given by algorithms $(\mathsf{Gen}, \mathsf{Com}, \mathsf{OpenZ}, \mathsf{OpenX}, \mathsf{DecZ}, \mathsf{DecX})$, where the classical algorithms here are Gen, $\mathsf{DecZ}, \mathsf{DecX}$. Our standard scenario is a classical verifier Ver that wants to securely make a Pauli measurement on some unknown quantum n-qubit state $|\psi\rangle$, held by a quantum prover P. In this instance, the flow of a PFC goes as follows.

1. Ver samples classical keys $(\mathsf{CK}, \mathsf{dk}) \leftarrow \mathsf{Gen}\left(1^\lambda\right)$. We think of CK as a public commitment key in the form of an ideally-obfuscated classical circuit, and dk is a secret decoding key. Ver sends CK to the prover.

2. P commits to its state by computing $(c, \mathcal{B}, \mathcal{U}) \leftarrow \mathsf{Com}^{\mathsf{CK}}\left(|\psi\rangle_{\mathcal{B}}\right)$, where c is a classical commitment string, $\mathcal{B} := (\mathcal{B}_i)_{i\in[n]}$ is the original n-qubit register that held $|\psi\rangle$ and $\mathcal{U} := (\mathcal{U}_i)_{i\in[n]}$ is an additional quantum register with some $\mathrm{poly}(n)$-qubits, now entangled with $\mathcal{B}$. P sends c to the verifier.

3. Ver chooses some measurement basis $h \in \{0,1\}^n$, where $h_i = 0$ denotes measuring qubit i of $|\psi\rangle$ in the standard basis and $h_i = 1$ denotes Hadamard basis. Ver sends h to the prover.

4. P decommits to the Pauli observables of its state by applying $u_i \leftarrow \mathsf{OpenZ}\left(\mathcal{B}_i, \mathcal{U}_i\right)$ for every $h_i = 0$ and $u_i \leftarrow \mathsf{OpenX}\left(\mathcal{B}_i, \mathcal{U}_i\right)$ otherwise. P sends $u := (u_i)_{i\in[n]}$ to the verifier.

5. Ver can now privately decode the measurement outcome by applying the corresponding decoding procedures $\mathsf{DecZ}\left(\mathsf{dk}, \cdot, \cdot\right)$ and $\mathsf{DecX}\left(\mathsf{dk}, \cdot, \cdot\right)$. For each $i \in [n]$, the corresponding algorithm takes as input the commitment c and the value u_i, and outputs a measurement result $b_i \in \{0,1\}$.

In words, what happens above is that at Step 2, the prover generates an *encoding* of the state $|\psi\rangle_{\mathcal{B}}$, by embedding it into a larger space $(\mathcal{B}, \mathcal{U})$. For each qubit, at the decommitment Step 4, the prover can generate an encoding of the standard basis measurement $u_i \leftarrow \mathsf{OpenZ}\left(\mathcal{B}_i, \mathcal{U}_i\right)$ or Hadamard basis measurement $u_i \leftarrow \mathsf{OpenX}\left(\mathcal{B}_i, \mathcal{U}_i\right)$. The values of these encodings are later extracted by Ver, using dk.

PFC Security: Private vs Public Verification. Intuitively, security of a PFC assures that the values of standard-basis measurements cannot be swapped. A bit more formally, for each part $i \in [n]$ of the committed state on register $(\mathcal{B}_i, \mathcal{U}_i)$ at the end of Step 2, if the state collapses to hold only encodings of standard basis measurements for value $b \in \{0,1\}$ (i.e., strings u_i such that $\mathsf{DecZ}(\mathsf{dk}, c, u_i) = b$), then it cannot be mapped (except with negligible probability) to any encodings of standard basis measurements for value $1-b$ (i.e., strings u_i such that $\mathsf{DecZ}(\mathsf{dk}, c, u_i) = 1-b$). In fact, this is just one way to describe the traditional notion of post-quantum binding for classical messages (equivalent to collapse-binding and sum-binding).

In a privately-verifiable PFC, the verifier's decoding key dk must remain completely inaccessible to the prover throughout the protocol, and such a PFC was (implicitly) constructed in [Mah18]. In a publicly-verifiable scheme, as considered by [BKNY23], we aim to give the prover access to the verifier's decoding functionalities without compromising the binding security stated above. A natural attempt at such a definition would require the scheme to remain secure against a prover given quantum oracle access to the classical functions $\mathsf{DecZ}(\mathsf{dk}, \cdot, \cdot)$, $\mathsf{DecX}(\mathsf{dk}, \cdot, \cdot)$.

It turns out that the security of publicly-verifiable PFC need not hold[3] if the prover is given *uninterrupted* access to both $\mathsf{DecZ}(\mathsf{dk}, \cdot, \cdot)$, $\mathsf{DecX}(\mathsf{dk}, \cdot, \cdot)$. Instead, we consider the relaxation where the prover P is given quantum access to $\mathsf{DecZ}(\mathsf{dk}, \cdot, \cdot)$, $\mathsf{DecX}(\mathsf{dk}, \cdot, \cdot)$ only up to the collapse of the encoding register $(\mathcal{B}, \mathcal{U})$. That is, we consider a game where a malicious prover P^* prepares the commitment $(c, \mathcal{B}, \mathcal{U}) \leftarrow \mathsf{Com}^{\mathsf{CK}}(|\psi\rangle_{\mathcal{B}})$ as in Step 2, with additional access to both $\mathsf{DecZ}(\mathsf{dk}, \cdot, \cdot)$, $\mathsf{DecX}(\mathsf{dk}, \cdot, \cdot)$. Then, the PFC challenger "removes" access to the Hadamard decoding functionality $\mathsf{DecX}(\mathsf{dk}, \cdot, \cdot)$, and collapses registers $(\mathcal{B}_i, \mathcal{U}_i)_{i\in[n]}$ to encodings of classical measurement results $\{b_i\}_{i\in[n]}$. Then, given access to only $\mathsf{DecZ}(\mathsf{dk}, \cdot, \cdot)$, the prover P^* is required to map these encodings to encodings of classical measurement results for the opposing values $\{1-b_i\}_{i\in[n]}$. That is, the prover has full access to the verification functionalities while preparing it's *commtiment*, but *loses* access to the Hadamard decoding functionality while attempting to map between standard basis openings. This is made formal in Definition 7.

From Quantum to Classical Keys (and, from long to short keys). The only previous result on publicly-verifiable PFCs is due to the work of Bartusek, Kitagawa, Nishimaki and Yamakawa [BKNY23], which achieves the above (in a classical oracle model), though where the commitment key $|\mathsf{CK}\rangle$ is *quantum*, and has size proportional to the number of qubits n in the state $|\psi\rangle$ to be committed.

Without going too deep into their construction, we remark that it makes use of a common tool in quantum cryptography: quantum coset states [CLLZ21]. A quantum coset state is defined by a random subspace $S \subset \mathbb{Z}_2^\lambda$ and shift $v \in \mathbb{Z}_2^\lambda$, and is defined as

$$|S+v\rangle := \frac{1}{\sqrt{|S|}} \sum_{x\in S} |x+v\rangle\,.$$

As part of their security proof, [BKNY23] prove a new, and highly-delicate quantum information-theoretic statement regarding coset states and their verification. In a nutshell, they prove that the ability to (1) *learn to detect collapsing* cannot be achieved from (2) the ability to verify coset states. The ability to verify cosets is given by quantum oracle access to the classical membership check for the cosets $S+v$ and $S^\perp$, where $S^\perp \subseteq \mathbb{Z}_2^\lambda$ is the dual of S (i.e., vectors that have an inner product of 0 with all vectors inside S). It is a seminal result [AC12] and common knowledge by now that the state $|S+v\rangle$ can be verified given quantum access to $S+v$ and $S^\perp$ while remaining *unclonable*. However, unclonability is a weaker property than what is required by [BKNY23].

Instead, the following information-theoretic property is proved in [BKNY23, Claim 4.10]:[4] A quantum algorithm A, given $|S+v\rangle$ and quantum oracle access to both $S+v$, $S^\perp$ outputs some superposition $|\phi\rangle$. The state $|\phi\rangle$ holds in its first part elements in the coset $S+v$, and possibly an entangled state in the rest of the registers (intuitively, the rest of $|\phi\rangle$ can hold information learned from the oracles $S+v$ and $S^\perp$ about

[3] In fact, there is good reason to believe that this definition is not satisfiable at all, since a Hadamard basis decoding functionality intuitively gives the committer the ability to perform a projection in the Hadamard basis and thus change their value in the standard basis.

[4] The statement in [BKNY23, Claim 4.10] is slightly different, but implies the statement we discuss here.

the next step of the experiment). Then, the challenger "takes away" oracle access to the membership check $S^\perp$ and either collapses or not (both with probability 1/2) the registers holding the strings of $S + v$. The task is for the adversary to now distinguish whether the part of the register that held elements in $S + v$ collapsed or not. Note that this would have been easy given access to $S^\perp$. Thus, what's really shown is that even if one has access to $S^\perp$ at some point in time, this access cannot be used to detect collapsing *later*.

Circling back to our publicly-verifiable PFCs, the above property of coset states is reminiscent of our desired security notion (again, formalized in Definition 7). Observe that our security notion of PFCs can be interpreted as the inability to (1) *learn to switch measurement encodings* from the (2) ability to commit and decode in both bases.

Our Work: the OSS Collapsing Game. Our work begings with the natural idea to employ *one-shot signatures* (OSS) in order to generate the coset states required by $|\mathsf{CK}\rangle$. OSS is a powerful primitive in quantum cryptography, first defined in [AGKZ20] and recently constructed for the first time in [SZ25]. The OSS of [SZ25] is defined with respect to public parameters that enable anyone to sample their own *unclonable* coset states. That is, the sampled coset states cannot be cloned even by the sampler that produced them. Moreover, the public parameters can be used to sample an unbounded polynomial number of such unclonable states.

In principle, the use of OSS may eliminate two unwanted properties of the [BKNY23] PFC in a single swipe: OSS can enable commitment keys that are both classical and short. However, it turns out that the previous results on OSS (in particular those of [SZ25]) are not sufficient for our needs. Similarly to the situation with subspace states, there is a need for more delicate and stronger statements.

The OSS of [SZ25] is given by three classical oracles $\left(P, P^{-1}, D\right)$, where P, P^{-1} can be thought of as the oracles that allow to generate the quantum tokens, and D is the oracle that allows for verification of the quantum tokens. Previous work proves an unclonable property of the [SZ25] OSS, which implies that given access to $\left(P, P^{-1}, D\right)$, it is impossible to produce two copies of a token that verify with respect to D.[5]

However, as discussed above, unclonability is not sufficient for the security of PFCs. Instead, we require a similar "collapsing" property for OSS. That is, we consider a game in which an adversary, given full access to (P, P^{-1}, D), first samples a quantum token (and potentially some side information). Then, a challenger either measures the token or not in the standard basis. Given D, the adversary could easily win this "collapsing" game, as the state returned would no longer verify in the case of measurement. However, we require that no adversary can distinguish whether or not the challenger measured if we *revoke* their access to the D oracle after the challenger's operation. Proving this requires a careful reexamination of the [SZ25] proof strategy, which is formalized in Section 3.4. As part of our approach, we also give a new and simpler proof of the "coset-collapsing" property of coset states that implicitly formed the heart of the [BKNY23] proof strategy, which is formalized in Section 3.4.2.

From PCFs to QFHE and Obfuscation. Equipped with our central building block of Pauli functional commitments with classical keys, our construction of publicly-verifiable QFHE and classical obfuscation of (pseudo-deterministic) quantum circuits closely follows the outline from [BKNY23]. In particular, we use PFCs to *upgrade* a privately-verifiable QFHE scheme satisfying certain properties to a publicly-verifiable scheme. This compiler is presented in Section 5. Then, we use a straightforward "verify-then-decrypt" construction to obtain obfuscation for pseudo-deterministic quantum circuits from publicly-verifiable QFHE in the classical oracle model (Section 6).

One important departure from the work of [BKNY23] is our requirement of *succinctness*. While neither the privately-verifiable QFHE scheme we start with nor the PFC-based compiler are themselves succinct, we show how to restore succinctness with a generic approach based on (post-quantum) succinct ideal obfuscation of *classical* circuits, in the classical oracle model. We provide the first construction of this object by combining (classical) FHE and post-quantum non-interactive succinct arguments of knowledge in the quan-

[5] In fact, an even stronger statement is proven by [SZ25] showing the one cannot produce two classical signatures derived from the same token.

tum random oracle model [CMS19]. The proof establishing ideal obfuscation of our construction requires a careful application of techniques from [CMS19], which we formalize in Section 4.

2 Preliminaries

2.1 Notation

Definition 1 (Pseudo-deterministic quantum circuit)**.** *A family of psuedo-deterministic quantum circuits $\{Q_\lambda\}_{\lambda\in\mathbb{N}}$ is defined as follows. The circuit Q_λ takes as input a classical string $x \in \{0,1\}^{n(\lambda)}$ and outputs a bit $b \leftarrow Q_\lambda(x)$. The circuit is pseudo-deterministic if 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*

$$\Pr[Q_\lambda(x_\lambda) \to b_\lambda] = 1 - \mathsf{negl}(\lambda).$$

We will often leave the dependence on λ implicit, and just refer to pseudo-deterministic circuits Q with input x. In a slight abuse of notation, we will denote by $Q(x)$ the bit b such that $\Pr[Q(x) \to b] = 1 - \mathsf{negl}(\lambda)$.

Sometimes, we will consider (potentially succinct) descriptions *of pseudo-deterministic circuits, which we write as P_Q, and refer to as (pseudo-deterministic)* programs*. That is, while Q is understood to consist of the list of gates in the circuit, P_Q is understood to be an arbitrary (potentially short) description from which the entire list of gates Q can be derived. Thus, in general it may be the case that $|P_Q| \ll |Q|$.*

2.2 Fully-homomorphic encryption

We will use fully-homomorphic encryption for classical circuits, defined as follows.

Definition 2 (FHE)**.** *A fully-homomorphic encryption scheme consists of the following algorithms.*

- $\mathsf{Gen}(1^\lambda, D) \to (\mathsf{pk}, \mathsf{sk})$*: The p.p.t. key generation algorithm takes as input the security parameter 1^λ and a circuit depth D, and returns a public key pk and secret key sk.*
- $\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$*: The p.p.t. encryption algorithm takes as input the public key pk and a plaintext x, and outputs a ciphertext ct.*
- $\mathsf{Eval}(\mathsf{ct}, C) \to \widetilde{\mathsf{ct}}$*: The classical deterministic evaluation algorithm takes as input a ciphertext ct and the description of a classical deterministic computation C, and outputs an evaluated ciphertext $\widetilde{\mathsf{ct}}$.*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to x$*: The classical deterministic decryption algorithm takes as input the secret key sk and a ciphertext ct and outputs a plaintext x.*

It should satisfy the following properties.

- ***Correctness.*** *For any $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Gen}(1^\lambda, D)$, $\mathsf{ct} \in \mathsf{Enc}(\mathsf{pk}, x)$ and circuits C of depth at most D*

$$\mathsf{P}\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}') = C(x) : \begin{array}{c}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x) \\ \mathsf{ct}' \leftarrow \mathsf{Eval}(\mathsf{ct}, C)\end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- ***Privacy.*** *For any QPT adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, polynomial D and messages x_0, x_1,*

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

- ***Compactness.*** *There exists a fixed polynomial p such that any classical circuit C of depth at most D and it holds that*
  - *$|\mathsf{pk}|, |\mathsf{ct}|, |\mathsf{ct}'| \le p(\lambda, D, |x|)$ and*
  - *the runtimes $\mathsf{Gen}(1^\lambda, D, |x|)$, $\mathsf{Enc}(\mathsf{pk}, x)$, $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}')$ are bounded by $p(\lambda, D, |x|)$.*

2.3 One-shot signatures (OSS)

Definition 3 (One-Shot Signature Scheme)**.** *A* one-shot signature (OSS) *scheme is a tuple of algorithms* $\mathsf{OSS} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *satisfying the following:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{O}$*:* A classical probabilistic polynomial-time algorithm that given the security parameter 1^λ, outputs a classical deterministic circuit O.
- $\mathsf{Gen}^{\mathsf{O}}(\ell) \to (\mathsf{vk}, |\mathsf{sk}\rangle)$*:* A quantum polynomial-time algorithm that takes as input a message length ℓ, has oracle access to O, and samples a classical public key vk and quantum secret key $|\mathsf{sk}\rangle$.
- $\mathsf{Sign}^{\mathsf{O}}(|\mathsf{sk}\rangle, m) \to \sigma$*:* A quantum polynomial-time algorithm that has oracle access to O, and, given the quantum key $|\mathsf{sk}\rangle$ and message $m \in \{0,1\}^\ell$ produces a classical signature σ.
- $\mathsf{Ver}^{\mathsf{O}}(\mathsf{vk}, m, \sigma) \to \{\bot, \top\}$*:* A classical deterministic polynomial-time algorithm that has oracle access to O and verifies a message m and signature σ relative to a public key vk.

Correctness: *For any ℓ and $m \in \{0,1\}^\ell$,*

$$\Pr_{\substack{\mathsf{O}\leftarrow\mathsf{Setup}(1^\lambda,1^\ell)\\(\mathsf{vk},|\mathsf{sk}\rangle)\leftarrow\mathsf{Gen}^{\mathsf{O}}\\\sigma\leftarrow\mathsf{Sign}^{\mathsf{O}}(|\mathsf{sk}\rangle,m)}}[\mathsf{Ver}^{\mathsf{O}}(\mathsf{vk}, m, \sigma) = \top] \geq 1 - \mathsf{negl}(\lambda).$$

Security (Strong unforgeability): *For any ℓ and QPT algorithm $\mathcal{A}$,*

$$\Pr_{\substack{\mathsf{O}\leftarrow\mathsf{Setup}(1^\lambda,1^\ell)\\(\mathsf{vk},m_0,m_1,\sigma_0,\sigma_1)\leftarrow\mathcal{A}^{\mathsf{O}}}}[\mathsf{Ver}^{\mathsf{O}}(\mathsf{vk}, m_0, \sigma_0) = \top \wedge \mathsf{Ver}^{\mathsf{O}}(\mathsf{vk}, m_1, \sigma_1) = \top \wedge \sigma_0 \neq \sigma_1] \leq \mathsf{negl}(\lambda).$$

Note that an adversary breaks the strong unforgeability security guarantee even if it produces two different signatures $\sigma_0 \neq \sigma_1$ of the same message $m_0 = m_1$.

We now describe the construction of one-shot signatures in [SZ25]. We will later use this construction in a non-black-box way in Section 3.2.

Parameters: Let $\lambda \in \mathbb{N}$ be the security parameter. Let $s = 16\lambda$ and let $n, r, k \in \mathbb{N}$ be such that $r = s \cdot (\lambda - 1)$, $n = r + \frac{3}{2} \cdot s$ and $k = n$.

- $\mathsf{OSS.Setup}(1^\lambda)$
  - Sample a random permutation $\Pi : \{0,1\}^n \to \{0,1\}^n$. Let $H(x)$ be the first r output bits of $\Pi(x)$ and let $J(x)$ denote the remaining $n - r$ bits. Interpret $J(x) \in \mathbb{Z}_2^{n-r}$.
  $$\Pi(x) = H(x) \| J(x).$$
  - Sample a random function $F : \{0,1\}^r \to \{0,1\}^{k\cdot(n-r+1)}$. Interpret $F(y) = (\mathbf{A}_y, \mathbf{b}_y)$ where $\mathbf{A}_y \in \mathbb{Z}_2^{k\times(n-r)}$ is a full-rank matrix and $\mathbf{b} \in \mathbb{Z}_2^n$.
  Let $S_y = \{\mathbf{A}_y\mathbf{x} + \mathbf{b}_y \mid \mathbf{x} \in \mathbb{Z}_2^{n-r}\}$ be the coset defined by $(\mathbf{A}_y, \mathbf{b}_y)$. Let $S_{y,b} := \{\mathbf{u} \in S_y \mid u_1 = b\}$ be the affine subspace of S_y whose first coordinate entry is b.
  - Define the functions P, P^{-1}, D as follows
  $$P(x) = (y, \mathbf{A}_y J(x) + \mathbf{b}_y) \in \{0,1\}^n \to \{0,1\}^r \times \mathbb{Z}_2^k \text{ where } y = H(x)$$
  $$P^{-1}(y, \mathbf{u}) = \begin{cases} \Pi^{-1}(y, \mathbf{z}) & \text{if } \exists \mathbf{z} \in \mathbb{Z}_2^{n-r} \text{ such that } \mathbf{A}_y\mathbf{z} + \mathbf{b}_y = \mathbf{u} \\ \bot & \text{otherwise.} \end{cases}$$
  $$D(y, \mathbf{v}) = \begin{cases} 1 & \text{if } \mathbf{v}^\top \mathbf{A}_y = \mathbf{0}^{n-r} \\ 0 & \text{otherwise.} \end{cases}$$
  Let $S_y^\perp := \{\mathbf{v} \mid D(y, \mathbf{v}) = 1\}$ be the subspace dual to S_y.

– Output $\mathsf{O} := (P, P^{-1}, D)$.

- $\mathsf{OSS.Gen}^{\mathsf{O}}(1^\lambda)$
  - Parse $\mathsf{O} = (P, P^{-1}, D)$.[6]
  - Prepare a uniform superposition over all strings $x \in \{0,1\}^n$ on register $\mathcal{X}$, and using the P oracle, compute $P(x) := (y, \mathbf{u})$ onto registers $\mathcal{Y}$,$\mathcal{U}$. Measure register register $\mathcal{Y}$ to get y.
  - Use the P^{-1} oracle on registers $(\mathcal{Y},\mathcal{U})$ to uncompute the $\mathcal{X}$ register.
  - Output $\mathsf{vk} := y$ and the state on $\mathcal{U}$ as $|\mathsf{sk}\rangle$.
- $\mathsf{OSS.Sign}^{\mathsf{O}}(|\mathsf{sk}\rangle, m \in \{0,1\})$
  - Parse $\mathsf{O} = (P, P^{-1}, D)$.
  - Let $\mathcal{U}$ be the register that $|\mathsf{sk}\rangle$ lives on. Measure the first qubit $\widehat{m}$ of the $\mathcal{U}$ register in the standard basis. If the result is $\widehat{m} = m$, then measure the entire $\mathcal{U}$ register in the standard basis to get $\mathbf{u}$ and output $\mathbf{u}$.
  - If $\widehat{m} \neq m$, apply the rotation $H^{\otimes n} \circ \mathsf{Phase}^D \circ H^{\otimes n}$ to the $\mathcal{U}$ register, where Phase^D is the map $|\mathbf{v}\rangle \mapsto (-1)^{D(\mathbf{v})} |\mathbf{v}\rangle$. Measure the register $\mathcal{U}$ to get $\mathbf{u}$. Abort if the first bit of $\mathbf{u}$ is 0, and otherwise output $\mathbf{u}$.
- $\mathsf{OSS.Ver}^{\mathsf{O}}(\mathsf{vk}, m, \sigma)$
  - Parse $\mathsf{vk} = y$ and $\sigma = \mathbf{u}$.
  - If the first bit of $\mathbf{u}$ is not m and reject.
  - Compute $P^{-1}(y, \mathbf{u})$ and reject if the output is $\perp$. Accept otherwise.

2.4 Measure and re-program

Theorem 3 (Measure and re-program [DFMS19, DFM20])**.** [7] *Let A, B be finite non-empty sets, and let $q \in \mathbb{N}$. Let $\mathcal{A}$ be an oracle-aided quantum circuit that makes q queries to a uniformly random function $H : A \to B$ and then outputs classical strings (a, z) where $a \in A$. There exists a two-stage quantum circuit $\mathsf{Sim}[\mathcal{A}]$ such that for any predicate V, it holds that*

$$\Pr\left[V(a,b,z)=1 : \begin{array}{r}(a,\mathsf{state}) \leftarrow \mathsf{Sim}[\mathcal{A}] \\ b \leftarrow B \\ z \leftarrow \mathsf{Sim}[\mathcal{A}](b,\mathsf{state})\end{array}\right] \geq \frac{\Pr\left[V(a,H(a),z)=1 : (a,z) \leftarrow \mathcal{A}^H\right]}{(2q+1)^2}.$$

Moreover, $\mathsf{Sim}[\mathcal{A}]$ operates as follows.

- *Sample $H : A \to B$ as a $2q$-wise independent function and $(i, d) \leftarrow (\{0, \ldots, q-1\} \times \{0,1\}) \cup \{(q, 0)\}$.*
- *Run $\mathcal{A}$ until it has made i oracle queries, answering each query using H.*
- *When $\mathcal{A}$ is about to make its $(i+1)$'th oracle query, measure its query registers in the standard basis to obtain a. In the special case that $(i, d) = (q, 0)$, the simulator measures (part of) the final output register of $\mathcal{A}$ to obtain a.*
- *The simulator receives $b \leftarrow B$.*

[6] In the construction of Pauli functional commitments in Section 3.2, it will be relevant that $\mathsf{OSS.Gen}$ does not actually need access to the D oracle, the oracles P and P^{-1} suffice.

[7] This theorem was stated more generally in [**?**, **?**] to consider the drop in expectation for each specific $a^* \in A$, and also to consider a more general class of quantum predicates.

- *If $d = 0$, answer $\mathcal{A}$'s $(i+1)$'th query using H, and if $d = 1$, answer $\mathcal{A}$'s $(i+1)$'th query using $H[a \to b]$, which is the function H except that $H(a)$ is re-programmed to b.*
- *Run $\mathcal{A}$ until it has made all q oracle queries. For queries $i+2$ through q, answer using $H[a \to b]$.*
- *Measure $\mathcal{A}$'s output z.*

Note that the running time of $\mathsf{Sim}[\mathcal{A}]$ is at most $\mathrm{poly}(q, \log|A|, \log|B|)$ times the running time of $\mathcal{A}$.

2.5 The Compressed Oracle Technique

The compressed random oracle [Zha19] acts on query and database registers $(\mathcal{Q}, \mathcal{D})$ as follows:

$$\mathsf{CStO} = \mathsf{Decomp} \circ \mathsf{CStO}' \circ \mathsf{Decomp} \circ \mathsf{Increase},$$

where $\mathsf{Increase}\,|x,y\rangle \otimes |D\rangle = |x,y\rangle \otimes |D\rangle\,|(\bot, 0^n)\rangle$ essentially adds another point in the database, and where Decomp and CStO' are defined as follows:

- Decomp is defined by $|x,u\rangle_{\mathcal{Q}} \otimes |D\rangle_{\mathcal{D}} \mapsto |x,u\rangle_{\mathcal{Q}} \otimes \mathsf{Decomp}_x\,|D\rangle_{\mathcal{D}}$, where Decomp_x is defined by its action on basis states as follows. Let D be such that $D(x) = \bot$:

$$|D\rangle \mapsto \frac{1}{\sqrt{|\mathcal{Y}|}} \sum_{y \in \mathcal{Y}} |D \cup \{(x,y)\}\rangle$$

$$\frac{1}{\sqrt{|\mathcal{Y}|}} \sum_{y \in \mathcal{Y}} |D \cup \{(x,y)\}\rangle \mapsto |D\rangle$$

$$\frac{1}{\sqrt{|\mathcal{Y}|}} \sum_{y \in \mathcal{Y}} (-1)^{y \cdot u} |D \cup \{(x,y)\}\rangle \mapsto \frac{1}{\sqrt{|\mathcal{Y}|}} \sum_{y \in \mathcal{Y}} (-1)^{y \cdot u} |D \cup \{(x,y)\}\rangle \qquad \text{for } u \neq 0$$

  In words, Decomp_x swaps the states $|D\rangle$ and $\frac{1}{\sqrt{|\mathcal{Y}|}} \sum_{y \in \mathcal{Y}} |D \cup \{(x,y)\}\rangle$, and acts as the identity on the space orthogonal to these two states.
- CStO' maps $|x,u\rangle_{\mathcal{Q}} \otimes |D\rangle_{\mathcal{D}} \mapsto |x, u \oplus D(x)\rangle \otimes |D\rangle$.

2.6 Useful Lemmas and Definitions

Definition 4 (Coset Partition Functions)**.** *[SZ25] For $n, \ell \in \mathbb{N}$, such that $\ell \leq n$, we say a function $Q : \{0,1\}^n \to \{0,1\}^m$ is a (n, m, ℓ) coset partition function if, for each y in the image of Q, the preimage set $Q^{-1}(y)$ has size 2^ℓ and is a coset of a linear space of dimension ℓ. Different preimage sets are allowed to be cosets of different linear spaces.*

Lemma 1 (Lemma 3.6 of [BKNY23])**.** *For each $\lambda \in \mathbb{N}$, let $\mathcal{K}_\lambda$ be a set of keys, and let $\{|\psi_k\rangle, O_k^0, O_k^1, B_k\}_{k \in \mathcal{K}_\lambda}$ be a set of state $|\psi_k\rangle$, classical functions O_k^0, O_k^1, and sets of inputs B_k. Suppose that the following properties holds.*

1. *The oracles O_k^0, O_k^1 are identical on inputs outside of S_k.*
2. *For any oracle-aided unitary U, with $q = q(\lambda)$ queries, there is some $\varepsilon = \varepsilon(\lambda)$ such that*[8]

$$\mathbb{E}_{k \leftarrow \mathcal{K}} \left[\| \Pi[B_k] U^{O_k^0} |\psi_k\rangle \|^2 \right] \leq \varepsilon$$

Then, for any oracle-aided unitary U with $q = q(\lambda)$ queries and for every distinguisher D,

$$\left| \Pr_{k \leftarrow \mathcal{K}} \left[D\left(k, U^{O_k^0} |\psi_k\rangle\right) = 0 \right] - \Pr_{k \leftarrow \mathcal{K}} \left[D\left(k, U^{O_k^1} |\psi_k\rangle\right) = 0 \right] \right| \leq 4q\sqrt{\varepsilon}.$$

[8] $\Pi[B_k]$ is the projector onto subspace spanned by B_k. Informally what this means is that the probability of that U given query access to $\mathcal{O}_k^0$ ouptuts $b \in B_k$ is at most ε.

Small-range distribution. [Zha21] Given a distribution D on $\mathcal{Y}$, define $\mathsf{SR}_r^D(\mathcal{X})$ as the following distribution on functions from $\mathcal{X}$ to $\mathcal{Y}$:

- For each $i \in [r]$, choose a random value $y_i \in \mathcal{Y}$ according to the distribution D.
- For each $x \in \mathcal{X}$, pick a random $i \in [r]$ and set $O(x) = y_i$.

Lemma 2 (Corollary 7.5 of [Zha21])**.** *The output distributions of a quantum algorithm making q quantum queries to an oracle either drawn from $\mathsf{SR}_r^D(\mathcal{X})$ or $D^{\mathcal{X}}$ are $\ell(q)/r$-close, where $\ell(q) = \pi^2(2q)^3/3 < 27q^3$.*

3 Pauli Functional Commitments with Classical Keys

3.1 Definition

The difference between the following definition and that of [BKNY23] is that while they allow the commitment keys to quantum states but we require classical commitment keys.

Another difference is that the commitment keys are reusable, and can be used to commit to many (qu)bits.

Definition 5 (Syntax)**.** *A Pauli functional commitment consists of six algorithms* $(\mathsf{Gen}, \mathsf{Com}, \mathsf{OpenZ}, \mathsf{OpenX}, \mathsf{DecZ}, \mathsf{DecX})$ *with the following syntax.*

- ***Key Generation:*** $\mathsf{Gen}(1^\lambda) \to (\mathsf{CK}, \mathsf{dk})$ *is a p.p.t. algorithm that takes as input the security parameter 1^λ and outputs a classical deterministic circuit CK and a classical decoding key dk.*
- ***Commitment:*** $\mathsf{Com}^{\mathsf{CK}}(b) \to (\mathcal{U}, c)$ *is a QPT algorithm that takes as input a bit b and has oracle access to CK. It outputs registers $(\mathcal{U}, \mathcal{C})$ and then measures $\mathcal{C}$ in the standard basis to obtain a classical string $c \in \{0,1\}^*$ and a left-over state on register $\mathcal{U}$. We then write*

$$\mathsf{Com}^{\mathsf{CK}} := |0\rangle\langle 0| \otimes \mathsf{Com}^{\mathsf{CK}}(0) + |1\rangle\langle 1| \otimes \mathsf{Com}^{\mathsf{CK}}(1)$$

  to refer to the map that applies $\mathsf{Com}^{\mathsf{CK}}(\cdot)$ classically controlled on a single-qubit register $\mathcal{B}$ to produce a state on registers $(\mathcal{B}, \mathcal{U}, \mathcal{C})$, and then measures $\mathcal{C}$ in the standard basis to obtain a classical string c along with a left-over quantum state on registers $(\mathcal{B}, \mathcal{U})$.
- ***Opening in Z, X bases:*** *q.p.t. measurements on registers $(\mathcal{B}, \mathcal{U})$ that outputs a classical string u:*

$$\mathsf{OpenZ}(\mathcal{B}, \mathcal{U}) \to u \quad \text{and} \quad \mathsf{OpenX}(\mathcal{B}, \mathcal{U}) \to u.$$

- ***Decoding in Z, X bases:*** *classical deterministic polynomial-time algorithm that takes as input the decoding key dk, a string c, and a string u, and outputs either a bit b or a $\perp$ symbol:*

$$\mathsf{DecZ}(\mathsf{dk}, c, u) \to \{0,1\} \cup \{\perp\} \quad \text{and} \quad \mathsf{DecX}(\mathsf{dk}, c, u) \to \{0,1\} \cup \{\perp\}.$$

Definition 6 (Correctness)**.** *A Pauli functional commitment scheme* $(\mathsf{Gen}, \mathsf{Com}, \mathsf{OpenZ}, \mathsf{OpenX}, \mathsf{DecZ}, \mathsf{DecX})$ *is correct if for any single-qubit (potentially mixed) state on register $\mathcal{B}$, it holds that*

$$\mathsf{TV}\left(\mathsf{Z}(\mathcal{B}), \mathsf{PFCZ}(1^\lambda, \mathcal{B})\right) = \mathsf{negl}(\lambda), \quad \text{and} \quad \mathsf{TV}\left(\mathsf{X}(\mathcal{B}), \mathsf{PFCX}(1^\lambda, \mathcal{B})\right) = \mathsf{negl}(\lambda),$$

where the distributions are defined as follows.

- $\mathsf{Z}(\mathcal{B})$ *measures $\mathcal{B}$ in the standard basis.*
- $\mathsf{X}(\mathcal{B})$ *measures $\mathcal{B}$ in the Hadamard basis.*
- $\mathsf{PFCZ}(1^\lambda, \mathcal{B})$ *samples* $(\mathsf{CK}, \mathsf{dk}) \leftarrow \mathsf{Gen}(1^\lambda)$, $(\mathcal{B}, \mathcal{U}, c) \leftarrow \mathsf{Com}^{\mathsf{CK}}(\mathcal{B})$, $u \leftarrow \mathsf{OpenZ}(\mathcal{B}, \mathcal{U})$, *and outputs* $\mathsf{DecZ}(\mathsf{dk}, c, u)$.

- $\mathsf{PFCX}(1^\lambda, \mathcal{B})$ *samples* $(\mathsf{CK}, \mathsf{dk}) \leftarrow \mathsf{Gen}(1^\lambda)$, $(\mathcal{B}, \mathcal{U}, c) \leftarrow \mathsf{Com}^{\mathsf{CK}}(\mathcal{B})$, $u \leftarrow \mathsf{OpenX}(\mathcal{B}, \mathcal{U})$, *and outputs* $\mathsf{DecX}(\mathsf{dk}, c, u)$.

Definition 7 (Single-bit Binding with public decodability)**.** *A Pauli functional commitment* $(\mathsf{Gen}, \mathsf{Com}, \mathsf{OpenZ}, \mathsf{OpenX}, \mathsf{DecZ}, \mathsf{DecX})$ *satisfies single-bit binding with public decodability if the following holds. Given* dk, c, *and* $b \in \{0,1\}$, *define the following projective measurement*

$$\Pi_{\mathsf{dk},c,b} := \sum_{d:\mathsf{DecZ}(\mathsf{dk},c,d)=b} |d\rangle\langle d|.$$

Consider any adversary $\mathcal{A} = \{(\mathcal{A}_1, \mathcal{A}_2)_\lambda\}_{\lambda\in\mathbb{N}}$, *where each* $\mathcal{A}_1$ *is an oracle-aided quantum operation, each* $\mathcal{A}_2$ *is an oracle-aided unitary, and each* $(\mathcal{A}_1, \mathcal{A}_2)_\lambda$ *make at most* $\mathrm{poly}(\lambda)$ *oracle queries. Then for any* $b \in \{0,1\}$,

$$\mathbb{E}\left[\left\|\Pi_{\mathsf{dk},c,1-b}\,\mathcal{A}_2^{\mathsf{CK},\mathsf{DecZ}[\mathsf{dk}]}\,\Pi_{\mathsf{dk},c,b}\,|\psi\rangle\right\| : (|\psi\rangle, c) \leftarrow \mathcal{A}_1^{\mathsf{CK},\mathsf{DecZ}[\mathsf{dk}],\mathsf{DecX}[\mathsf{dk}]}(1^\lambda)\right] = \mathsf{negl}(\lambda),$$

where the expectation is taken over $(\mathsf{CK}, \mathsf{dk}) \leftarrow \mathsf{Gen}(1^\lambda)$. *Here,* $\mathsf{DecZ}[\mathsf{dk}]$ *is the oracle implementing the classical functionality* $\mathsf{DecZ}(\mathsf{dk}, \cdot, \cdot)$, *and* $\mathsf{DecX}[\mathsf{dk}]$ *is the oracle implementing the classical functionality* $\mathsf{DecX}(\mathsf{dk}, \cdot, \cdot)$.

In the construction of the publicly verifiable QFHE scheme to consider the following equivalent notion of binding.

Definition 8 (String binding with public decodability)**.** *A Pauli functional commitment* $(\mathsf{Gen}, \mathsf{Com}, \mathsf{OpenZ}, \mathsf{OpenX}, \mathsf{DecZ}, \mathsf{DecX})$ *satisfies* string binding with public decodability *if the following holds for any polynomial* $m = m(\lambda)$ *and two disjoint sets* $W_0, W_1 \subset \{0,1\}^m$ *of* m*-bit strings. Given a set of* m *decoding keys* $\mathbf{dk} = (\mathsf{dk}_1, \ldots, \mathsf{dk}_m)$, m *strings* $\mathbf{c} = (c_1, \ldots, c_m)$, *and* $b \in \{0,1\}$, *define*

$$\Pi_{\mathbf{dk},\mathbf{c},W_b} := \sum_{w\in W_b} \left(\bigotimes_{i\in[m]} \Pi_{\mathsf{dk}_i,c_i,w_i}\right).$$

Consider any adversary $\{(\mathcal{A}_1, \mathcal{A}_2)_\lambda\}_{\lambda\in\mathbb{N}}$, *where each* $\mathcal{A}_1$ *is an oracle-aided quantum operation, each* $\mathcal{A}_2$ *is an oracle-aided unitary, and each* $(\mathcal{A}_1, \mathcal{A}_2)_\lambda$ *make at most* $\mathrm{poly}(\lambda)$ *oracle queries. Then,*

$$\mathbb{E}\left[\left\|\Pi_{\mathbf{dk},\mathbf{c},W_1}\mathcal{A}_2^{\mathbf{CK},\mathsf{DecZ}[\mathbf{dk}]}\Pi_{\mathbf{dk},\mathbf{c},W_0}\,|\psi\rangle\right\| : (|\psi\rangle, \mathbf{c}) \leftarrow \mathcal{A}_1^{\mathbf{CK},\mathsf{DecZ}[\mathbf{dk}],\mathsf{DecX}[\mathbf{dk}]}(1^\lambda)\right] = \mathsf{negl}(\lambda),$$

where the expectation is over $\{\mathsf{CK}_i, \mathsf{dk}_i \leftarrow \mathsf{Gen}(1^\lambda)\}_{i\in[m]}$. *Here,* $\mathbf{CK}$ *is the collection of oracles* $\mathsf{CK}_1, \ldots, \mathsf{CK}_m$, $\mathsf{DecZ}[\mathbf{dk}]$ *is the collection of oracles* $\mathsf{DecZ}[\mathsf{dk}_1], \ldots, \mathsf{DecZ}[\mathsf{dk}_m]$, *and* $\mathsf{DecX}[\mathbf{dk}]$ *is the collection of oracles* $\mathsf{DecX}[\mathsf{dk}_1], \ldots, \mathsf{DecX}[\mathsf{dk}_m]$.

Lemma 3 (Lemma 4.5 of [BKNY23])**.** *Any Pauli functional commitment that satisfies single-bit binding with public decodability (Definition 7) also satisfies string binding with public decodability (Definition 8).*

Since string-binding when $m = 1$ is exactly the definition of single-bit binding, we have the following corollary.

Corollary 1. *The single-bit binding definition (Definition 7) and string binding definition (Definition 8) are equivalent.*

3.2 Construction

In this section, we will present our construction of a Pauli Functional Commitment scheme. Let $n = n(\lambda) \geq \lambda$. Our construction makes use of the following ingredients:

1. A general one-shot signature scheme $\mathsf{OSS} = (\mathsf{OSS.Setup}, \mathsf{OSS.Gen}, \mathsf{OSS.Sign}, \mathsf{OSS.Ver})$.
2. The particular construction of one-shot signatures from [SZ25], which we will denote as $\overline{\mathsf{OSS}}$.
3. PRF F.

Now, we are ready to present the construction.

- $\mathsf{Gen}(1^\lambda; R)$:
  - Parse randomness $R = (R_1, R_2)$.
  - Run $\mathsf{O} \leftarrow \mathsf{OSS.Setup}(1^\lambda; R_1)$.
  - Sample a random PRF key k using randomness R_2.
  - Define oracle $\mathsf{CK}_P(\mathsf{vk}, x)$ as follows:
    - Run $\overline{\mathsf{O}} \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda; F_k(\mathsf{vk}))$ and parse $\overline{\mathsf{O}} = (P, P^{-1}, D)$.
    - Output $P(x)$.
  - Define oracle $\mathsf{CK}_{P^{-1}}(\mathsf{vk}, \overline{\mathsf{vk}}, u)$ as follows:
    - Run $\overline{\mathsf{O}} \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda; F_k(\mathsf{vk}))$ and parse $\overline{\mathsf{O}} = (P, P^{-1}, D)$.
    - Output $P^{-1}(\overline{\mathsf{vk}}, u)$.
  - Define the oracle $\mathsf{CK}_D(\mathsf{vk}, \sigma, \overline{\mathsf{vk}}, u)$ as follows:
    - Run $\overline{\mathsf{O}} \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda; F_k(\mathsf{vk}))$ and parse $\overline{\mathsf{O}} = (P, P^{-1}, D)$.
    - If $\mathsf{OSS.Ver}^{\mathsf{O}}(\mathsf{vk}, 0, \sigma) = \bot$, output $\bot$ and abort.
    - Otherwise, output 0 if $D(\overline{\mathsf{vk}}, u) = 1$ and 1 if $D(\overline{\mathsf{vk}}, u) = 0$.
  - Output $\mathsf{CK} = (\mathsf{O}, \mathsf{CK}_P, \mathsf{CK}_{P^{-1}}, \mathsf{CK}_D)$ and $\mathsf{dk} = (R_1, k)$.
- $\mathsf{Com}^{\mathsf{CK}}(b)$:
  - Parse $\mathsf{CK} = (\mathsf{O}, \mathsf{CK}_P, \mathsf{CK}_{P^{-1}}, \mathsf{CK}_D)$.
  - Run $(\mathsf{vk}, |\mathsf{sk}\rangle) \leftarrow \mathsf{OSS.Gen}^{\mathsf{O}}(1^\lambda)$. Coherently apply $\mathsf{OSS.Sign}^{\mathsf{O}}(|\mathsf{sk}\rangle, 0)$ to get a superposition over signatures σ on bit 0 on register $\mathcal{S}$. This signature register $\mathcal{S}$ will allow us to (coherently) access the oracle CK_D.
  - Run $(\overline{\mathsf{vk}}, |\overline{\mathsf{sk}}\rangle) \leftarrow \overline{\mathsf{OSS}}.\mathsf{Gen}^{\mathsf{CK}_P(\mathsf{vk},\cdot),\mathsf{CK}_{P^{-1}}(\mathsf{vk},\cdot,\cdot)}(1^\lambda)$,[9] where $\overline{\mathsf{vk}} = y$ and $|\overline{\mathsf{sk}}\rangle = |S_y\rangle$ on register $\mathcal{S}'$ for some y as determined by P. Measure the first qubit of $|S_y\rangle$ in the standard basis to get result b'. If $b = b'$ then the state has collapsed to $|S_{y,b}\rangle$ and we continue.

    Otherwise, perform a rotation from $|S_{y,b'}\rangle$ to $|S_{y,b}\rangle$ by applying the operation

    $$(\mathsf{H}^{\otimes n} \otimes \mathsf{I}) \circ \mathsf{Phase}^{\mathsf{CK}_D(\mathsf{vk},\cdot,\cdot)} \circ (\mathsf{H}^{\otimes n} \otimes \mathsf{I})$$

    to register $(\mathcal{S}', \mathcal{S})$, where $\mathsf{Phase}^{\mathsf{CK}_D(\mathsf{vk},\cdot,\cdot)}$ is the map

    $$|u\rangle_{\mathcal{S}'} |\sigma\rangle_{\mathcal{S}} \mapsto (-1)^{\mathsf{CK}_D(\mathsf{vk},\sigma,u)} |u\rangle_{\mathcal{S}'} |\sigma\rangle_{\mathcal{S}} .$$

  - Reverse the $\mathsf{OSS.Sign}^{\mathsf{O}}(\cdot, 0)$ operation on $\mathcal{S}$ to recover $|\mathsf{sk}\rangle$.

[9] Note that $\overline{\mathsf{OSS}}.\mathsf{Gen}$ does not need the entire O oracle in the OSS construction of SZ25, the oracles P and P^{-1} suffice.

– Sample $\sigma \leftarrow \mathsf{OSS.Sign}^{\mathsf{O}}(|\mathsf{sk}\rangle, 1)$, and output $c = (\mathsf{vk}, \sigma, \overline{\mathsf{vk}})$ along with the final state on register $\mathcal{U} := \mathcal{S}'$.

- $\mathsf{OpenZ}(\mathcal{B}, \mathcal{U})$: Measure all registers in the standard basis.
- $\mathsf{OpenX}(\mathcal{B}, \mathcal{U})$: Measure all registers in the Hadamard basis.
- $\mathsf{DecZ}(\mathsf{dk}, c, d)$:
  - Parse $\mathsf{dk} = (R, k)$, $c = (\mathsf{vk}, \sigma, \overline{\mathsf{vk}})$ and $d = (b, u)$, where $\mathsf{vk} = y$, $b \in \{0,1\}$ and $u \in \{0,1\}^n$.
  - Run $\mathsf{O} \leftarrow \mathsf{OSS.Setup}(1^\lambda; R)$.
  - Run $\overline{\mathsf{O}} \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda; F_k(\mathsf{vk}))$ and parse $\overline{\mathsf{O}} = (P, P^{-1}, D)$.
  - Check that $\mathsf{OSS.Ver}^{\mathsf{O}}(\mathsf{vk}, 1, \sigma) = \top$. Otherwise output $\bot$.
  - If $u \in S_{y,b}$, output b, and otherwise output $\bot$. This can be checked given oracle access to $\overline{\mathsf{O}}$, by checking that $P^{-1}(y, u) \neq \bot$.
- $\mathsf{DecX}(\mathsf{dk}, c, d)$:
  - Parse $\mathsf{dk} = (R, k)$, $c = (\mathsf{vk}, \sigma, \overline{\mathsf{vk}})$ and $d = (b', u)$, where $\overline{\mathsf{vk}} = y$, $b' \in \{0,1\}$ and $u \in \{0,1\}^n$.
  - Run $\mathsf{O} \leftarrow \mathsf{OSS.Setup}(1^\lambda; R)$.
  - Run $\overline{\mathsf{O}} \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda; F_k(\mathsf{vk}))$ and parse $\overline{\mathsf{O}} = (P, P^{-1}, D)$.
  - Check that $\mathsf{OSS.Ver}^{\mathsf{O}}(vk, 1, \sigma) = \top$, and if not output $\bot$.
  - Compute the bit r as follows and abort if $r = \bot$

$$r := \begin{cases} 0 & \text{if } u \in S_y^\perp \\ 1 & \text{if } u \oplus (1, 0, \ldots, 0) \in S_y^\perp \\ \bot & \text{otherwise.} \end{cases}$$

    This sets $r = 0$ if $u \in S_y^\perp$ and $r = 1$ if $u \in (S_{y,0})^\perp \setminus S_y^\perp$. Output $b' \oplus r$. This can be computed using $\overline{\mathsf{O}}$.

3.3 Correctness

Theorem 4. *The Pauli Functional commitment scheme described in Section 3.2 satisfies correctness as in Definition 6.*

The proof of this statement is almost identical to that of Theorem 4.6 of [BKNY23], except that we are using one-shot signatures instead of signature tokens. We defer the proof to Section A.

3.4 Binding

Theorem 5. *The construction in Section 3.2 satisfies single-bit binding with public decodability as in Definition 7.*

Proof. It will be convenient for us to work with a different but sufficient (in fact, equivalent) definition of binding. By Lemma 7, it suffices to show that the scheme satisfies the following definition of binding.

Definition 9 (Collapse-binding)**.** *For any adversary $\mathcal{A} := \{(\mathcal{A}_1, \mathcal{A}_2)_\lambda\}_{\lambda \in \mathbb{N}}$, where each of $\mathcal{A}_1$ and $\mathcal{A}_2$ are oracle-aided quantum operations that make at most $\mathrm{poly}(\lambda)$ oracle queries, define the collapse-binding experiment $\mathsf{CollapseBindingExpt}^{\mathcal{A}}(1^\lambda)$ as follows.*

- *Sample* $(\mathsf{CK}, \mathsf{dk}) \leftarrow \mathsf{Gen}(1^\lambda)$.

- *Run* $\mathcal{A}_1^{\mathsf{CK},\mathsf{DecZ}[\mathsf{dk}],\mathsf{DecX}[\mathsf{dk}]}(1^\lambda)$ *until it outputs a commitment* c *and a state on registers* $(\mathcal{B},\mathcal{U},\mathcal{R})$*. Here* $\mathcal{B}$ *is a single-qubit register,* $\mathcal{U}$ *is the opening register and* $\mathcal{R}$ *holds the internal state of* $\mathcal{A}$*.*
- *Sample* $b \leftarrow \{0,1\}$*. If* $b = 0$*, do nothing, and if* $b = 1$ *measure* $(\mathcal{B},\mathcal{U})$ *in the computational basis.*
- *Run* $\mathcal{A}_2^{\mathsf{CK},\mathsf{DecZ}[\mathsf{dk}]}(\mathcal{B},\mathcal{U},\mathcal{R})$ *until it outputs a bit* b'*.*
- *The experiment outputs 1 if* $b = b'$*.*

We say that adversary $\mathcal{A}$ *is* valid *if the state on* $(\mathcal{B},\mathcal{U})$ *output by* $\mathcal{A}_1$ *is in the image of* $|0\rangle\langle 0| \otimes \Pi_{\mathsf{dk},c,0} + |1\rangle\langle 1| \otimes \Pi_{\mathsf{dk},c,1}$*. Then, we say that a Pauli functional commitment* $(\mathsf{Gen},\mathsf{Com},\mathsf{OpenZ},\mathsf{OpenX},\mathsf{DecZ},\mathsf{DecX})$ *satisfies publicly-decodable collapse-binding if it holds that for all valid adversaries* $\mathcal{A}$*,*

$$\left|\Pr\left[\mathsf{CollapseBindingExpt}^{\mathcal{A}}(1^\lambda) = 1\right] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

We will now show, through a sequence of hybrids, that our construction satisfies the collapse-binding with public decodability property. For any valid q-query[10] adversary $\mathcal{A}$, let $\mathsf{Hyb}_i^{\mathcal{A}}$ be the advantage achieved by $\mathcal{A}$ in Hybrid i.

Hybrid 0. This hybrid is the collapse-binding game defined above.

Hybrid 1. We now move to a hybrid where we replace the PRF F_k in the scheme in Section 3.2 with a random oracle $H : \mathcal{X} \to \mathcal{R}$, so that we are effectively choosing a random instance of the one-shot signature oracles $\overline{\mathsf{O}}$ independently for every choice of verification key vk.

Hybrid 0 and Hybrid 1 are indistinguishable because of the post-quantum security of the PRF, so for every valid $\mathcal{A}$, $|\mathsf{Hyb}_0^{\mathcal{A}} - \mathsf{Hyb}_1^{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Hybrid 2. Observe that once $\mathcal{A}_1$ has committed to $c^* = (\mathsf{vk}^*, \sigma_1^*, \overline{vk})$, where σ_1^* is a valid signature of 1 under vk^*, we can replace the CK_D oracle given to the second-stage adversary $\mathcal{A}_2$ with $\mathsf{CK}_D\{\mathsf{vk}^*\}$, which works as follows.

$\mathsf{CK}_D\{\mathsf{vk}^*\}(\mathsf{vk},\sigma,\overline{\mathsf{vk}},u)$:

- If $\mathsf{vk} = \mathsf{vk}^*$, output $\perp$.
- Otherwise, output $\mathsf{CK}_D(\mathsf{vk},\sigma,\overline{\mathsf{vk}},u)$.

Formally, the experiment in this hybrid is as follows:

- Sample $(\mathsf{CK},\mathsf{dk}) \leftarrow \mathsf{Gen}(1^\lambda)$.
- Run $\mathcal{A}_1^{\mathsf{CK},\mathsf{DecZ}[\mathsf{dk}],\mathsf{DecX}[\mathsf{dk}]}(1^\lambda)$ until it outputs a commitment c and a state on registers $(\mathcal{B},\mathcal{U},\mathcal{R})$. Here $\mathcal{B}$ is a single-qubit register, $\mathcal{U}$ is the opening register and $\mathcal{R}$ holds the internal state of $\mathcal{A}$.
- Sample $b \leftarrow \{0,1\}$. If $b = 0$, do nothing, and if $b = 1$ measure $(\mathcal{B},\mathcal{U})$ in the computational basis.
- Run $\mathcal{A}_2^{\mathsf{CK}\{\mathsf{vk}^*\},\mathsf{DecZ}[\mathsf{dk}]}(\mathcal{B},\mathcal{U},\mathcal{R})$ until it outputs a bit b', where $\mathsf{CK}\{\mathsf{vk}^*\} = (\mathsf{O},\mathsf{CK}_P,\mathsf{CK}_{P^{-1}},\mathsf{CK}_D\{\mathsf{vk}^*\})$.
- The experiment outputs 1 if $b = b'$.

Claim 1. *For every valid adversary* $\mathcal{A}$*,* $|\mathsf{Hyb}_1^{\mathcal{A}} - \mathsf{Hyb}_2^{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

[10] The bound q on the number of queries made by $\mathcal{A}$ will be important in Hybrid 3, when we apply the small-range distribution argument of [Zha21].

Proof. By Lemma 1, if an adversary is able to distinguish between these two hybrids, there must be an adversary $\mathcal{A}'$ that produces an input on which CK and $\mathsf{CK}\{\mathsf{vk}^*\}$ differ. The only points on which CK_D and $\mathsf{CK}_D\{\mathsf{vk}^*\}$ differ are of the form $(\mathsf{vk}^*, \sigma_0, \cdot, \cdot)$, where σ_0 is a valid signature of 0 under vk^*. By the strong unforgeability of the one-shot signatures scheme OSS, once the adversary has committed to a valid signature σ_1^* of 1 under vk^*, it cannot produce a valid σ_0, and therefore cannot notice the difference between the oracles CK_D and $\mathsf{CK}_D\{\mathsf{vk}^*\}$. □

Hybrid 3. This hybrid is the same as Hybrid 2, except that use a small-range distribution argument to move to a hybrid where we replace the random oracle H introduced in Hybrid 1 with a random function H_{SR} of bounded range. This will allow us to rephrase the collapse-binding game in terms of the oracles of the inner one-shot signature scheme $\overline{\mathsf{OSS}}$, which we then later analyze.

Let $R = 300q^3/\varepsilon$, where $\varepsilon(\lambda) = \mathsf{Hyb}_2^{\mathcal{A}}$. We sample R independent and uniformly random values $r_i \leftarrow \mathcal{R}$, for $i \in [R]$, and for each $x \in \mathcal{X}$, we sample a random $i_x \leftarrow [R]$ and set $H_{\mathsf{SR}}(x) := r_{i_x}$.

Claim 2. *For every valid q-query adversary $\mathcal{A}$, $\mathsf{Hyb}_3^{\mathcal{A}} \geq \frac{4}{5}\mathsf{Hyb}_2^{\mathcal{A}}$.*

Proof. By Lemma 2, we know that the output distributions of $\mathcal{A}$ in Hybrids 2 and 3 are at most $27q^3/R$-far in statistical distance, and by our choice of R, this means that $|\mathsf{Hyb}_2^{\mathcal{A}} - \mathsf{Hyb}_3^{\mathcal{A}}| \leq \varepsilon/5$, which in turn implies that $\mathsf{Hyb}_3^{\mathcal{A}} \geq 4\varepsilon/5$. □

We now show that no query-efficient adversary can win the game in Hybrid 3 with non-negligible advantage. Towards that end, consider the following game, which is similar in spirit to the collapse-binding game in Definition 9, but rephrased in terms of the underlying oracles of $\overline{\mathsf{OSS}}$ in the [SZ25] construction.

Definition 10 (OSS collapse-binding experiment)**.** *Let $\overline{\mathsf{OSS}}$ be the one-shot signature scheme in Section 2.3. For any adversary $\mathcal{A} = \{(\mathcal{A}_1, \mathcal{A}_2)_\lambda\}_{\lambda\in\mathbb{N}}$, where each of $\mathcal{A}_1$ and $\mathcal{A}_2$ are quantum operations that make at most $\mathrm{poly}(\lambda)$ many oracle queries, define the experiment $\mathsf{OSSCollapseBindingExpt}^{\mathcal{A}}(1^\lambda)$ as follows.*

- *Sample $P, P^{-1}, D \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda)$. Let $\{S_y\}_{y\in\{0,1\}^r}$ be the cosets defined by P, P^{-1}.*
- *Run $\mathcal{A}_1^{P,P^{-1},D}(1^\lambda)$ until it outputs a string $y \in \{0,1\}^r$ and a state on registers $(\mathcal{U}, \mathcal{R})$. Here $\mathcal{U}$ is k-qubit register and $\mathcal{R}$ holds the internal state of $\mathcal{A}$.*
- *Sample $b \leftarrow \{0,1\}$. If $b = 0$, do nothing, and if $b = 1$ measure $\mathcal{U}$ in the computational basis.*
- *Run $\mathcal{A}_2^{P,P^{-1}}(\mathcal{U}, \mathcal{R})$ until it outputs a bit b'. The experiment outputs 1 if $b = b'$ and 0 otherwise.*

The adversary is valid if the state on $\mathcal{U}$ output by $\mathcal{A}_1$ is in the image of $\Pi_y = \sum_{u\in S_y} |u\rangle\langle u|$. We say that the adversary $\mathcal{A}$ wins the experiment with advantage

$$\left|\Pr[\mathsf{OSSCollapseBindingExpt}^{\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right|.$$

Given an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that wins the game in Hybrid 3 with non-negligible advantage, we give an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ that wins the OSS collapse-binding game with non-negligible advantage.
The first-stage adversary $\mathcal{B}_1$ works as follows:

- $\mathcal{B}_1$ receives oracles $P^*, (P^{-1})^*, D^*$ from the challenger.
- Let $\mathsf{CK}_3 = (\mathsf{O}_3, \mathsf{CK}_{P,3}, \mathsf{CK}_{P^{-1},3}, \mathsf{CK}_{D,3}), \mathsf{DecZ}_3, \mathsf{DecX}_3$ be oracles as defined in Hybrid 3 (using H_{SR} in place of the PRF), and let $\mathcal{R}_{\mathsf{SR}} \subseteq \mathcal{R}$ be the range of H_{SR}, such that $|\mathcal{R}_{\mathsf{SR}}| = R$.
- Sample a random $r^* \leftarrow \mathcal{R}_{\mathsf{SR}}$.
- Define oracles $\mathsf{CK}_P^*, \mathsf{CK}_{P^{-1}}^*, \mathsf{CK}_D^*$ and $\mathsf{DecZ}^*, \mathsf{DecX}^*$ to be identical to $\mathsf{CK}_{P,3}, \mathsf{CK}_{P^{-1},3}, \mathsf{CK}_{D,3}$ and DecZ_3, DecX_3, except that on inputs vk such that $H_{\mathsf{SR}}(\mathsf{vk}) = r^*$, we use oracles $P^*, (P^{-1})^*, D^*$ instead of the oracles you get by running $\overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda; H_{\mathsf{SR}}(\mathsf{vk}))$. For example, $\mathsf{CK}_P^*(\mathsf{vk}, x)$ is defined as follows:

- If $H_{\mathsf{SR}}(\mathsf{vk}) = r^*$, output $P^*(x)$.
- Otherwise, output $\mathsf{CK}_P(\mathsf{vk}, x)$.

The oracles $\mathsf{CK}^*_{P^{-1}}, \mathsf{CK}^*_D$ and $\mathsf{DecZ}^*, \mathsf{DecX}^*$ are similarly defined.

- Run $\mathcal{A}_1$ with $\mathsf{CK}^* = (\mathsf{O}_3, \mathsf{CK}^*_P, \mathsf{CK}^*_{P^{-1}}, \mathsf{CK}^*_D)$.
- Receive $c = (\widetilde{\mathsf{vk}}, \sigma, \overline{\mathsf{vk}})$ and registers $\mathcal{B}, \mathcal{U}$ from $\mathcal{A}_1$.
- If $H_{\mathsf{SR}}(\widetilde{\mathsf{vk}}) = r^*$, send $y = \overline{\mathsf{vk}}$ and $\mathcal{U}$ to the challenger.[11]
- Otherwise output a random y, where $(y, \mathcal{U}) \leftarrow \mathsf{OSS.Gen}^{P^*, (P^{-1})^*, D^*}(1^\lambda)$.

The second stage $\mathcal{B}_2$ adversary works as follows.

- $\mathcal{B}_2$ receives the oracles $P^*, (P^{-1})^*$ from the challenger.
- If $H_{\mathsf{SR}}(\widetilde{\mathsf{vk}}) \neq r^*$ then output $b' \leftarrow \{0,1\}$ uniformly at random.
- If $H_{\mathsf{SR}}(\widetilde{\mathsf{vk}}) = r^*$, run $\mathcal{A}_2$ by with oracles $\mathsf{CK}^*_D\{\widetilde{\mathsf{vk}}\}$ and DecZ^*, where $\mathsf{CK}^*_D\{\widetilde{\mathsf{vk}}\}(\mathsf{vk}, \sigma, \overline{\mathsf{vk}}, u)$ is defined like $\mathsf{CK}_D\{\widetilde{\mathsf{vk}}\}$, except that when $H(\widetilde{\mathsf{vk}}) = r^*$, we use the D^* oracle instead, to be consistent.
  - If $\mathsf{vk} = \widetilde{\mathsf{vk}}$, then output $\perp$ and abort.
  - If $H_{\mathsf{SR}}(\mathsf{vk}) = r^*$,
    - If $\mathsf{OSS.Ver}^{\mathsf{O}_3}(\mathsf{vk}, 0, \sigma) = \perp$, output $\perp$ and abort.
    - Otherwise, output 0 if $D^*(\overline{\mathsf{vk}}, u) = 1$ and 1 if $D^*(\overline{\mathsf{vk}}, u) = 0$.
  - Otherwise, output $\mathsf{CK}_{D,3}(\mathsf{vk}, \sigma, \overline{\mathsf{vk}}, u)$.
- Output the output of $\mathcal{A}_2$.

Observe that the oracles $\mathsf{CK}^*, \mathsf{DecZ}^*, \mathsf{DecX}^*$ and $\mathsf{CK}^*\{\cdot\}$ given to $\mathcal{A}_1, \mathcal{A}_2$ will have the same distribution as in Hybrid 3. Since $|\mathcal{R}_{\mathsf{SR}}| \leq R$, $\mathcal{B}_1$ correctly guesses $r^* = H_{\mathsf{SR}}(\widetilde{\mathsf{vk}})$ with probability at least $1/R$. So $\mathcal{B}$ wins the OSS collapse-binding game in Definition 10 with advantage at least

$$\frac{1}{R} \cdot \mathsf{Hyb}_3^{\mathcal{A}} \geq \Omega\left(\frac{\varepsilon^2}{q^3}\right),$$

where $\varepsilon = \mathsf{Hyb}_2^{\mathcal{A}}$, and the inequality follows from Claim 2. Finally, Theorem 6 shows that no query-efficient adversary can win the OSS collapse-binding game with non-negligible advantage, which completes the proof. □

3.4.1 Hardness of the OSS collapse-binding game

Theorem 6. *Let $\lambda \in \mathbb{N}$ denote the security parameter. Let q, s, n, r be polynomials in λ, and let $s \leq n - r$ such that $\frac{k^9 \cdot q^7 \cdot \frac{1}{\varepsilon^4}}{\sqrt{2^{n-r-s}}} \leq o(1)$. Then, for every q query adversary $\mathcal{A}$,*

$$\left|\Pr[\mathsf{OSSCollapseBindingExpt}^{\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

Proof. Suppose for contradiction that there exists an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with non negligible advantage in the OSS collapse-binding game. Then,

$$\left|\Pr[\mathsf{OSSCollapseBindingExpt}^{\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \geq \varepsilon(\lambda).$$

for some non-negligible function ε. Consider the following sequence of hybrids.

[11] For valid commitments the state on $(\mathcal{B}, \mathcal{U})$ is supported on basis states $|b, u\rangle$ where $b = u[1]$, so measuring $\mathcal{U}$ (or not) is the same as measuring both $\mathcal{B}, \mathcal{U}$.

$\mathbf{H}_1$ This corresponds to $\mathsf{OSSCollapseBindingExpt}^{\mathcal{A}}(1^\lambda)$

$\mathbf{H}_2$: In this hybrid, we use the standard small-range distribution technique to simulate the oracles P, P^{-1}, D using a polynomially bounded number of underlying cosets (A_y, b_y) sampled using a function $F' : \{0,1\}^r \to \{0,1\}^{k\cdot(n-r+1)}$ as follows:

- Let $R := (300 \cdot q^3) \cdot \frac{2^7 \cdot k^2}{\varepsilon}$
- for every $y \in \mathbb{Z}_2^r$, sample $i_y \leftarrow [R]$.
- Sample a random function $F' : \{0,1\}^r \to \{0,1\}^{k\cdot(n-r+1)}$. For every $i \in [R]$, interpret $F(i) = (\mathbf{A}_i \in \mathbb{Z}_2^{k\times(n-r)}, \mathbf{b}_i \in \mathbb{Z}_2^k)$.
- For $y \in \mathbb{Z}_2^r$, define $F'(y) := (\mathbf{A}_{i_y}, \mathbf{b}_{i_y})$.

Claim 3. *For any q query quantum algorithm $\mathcal{A}$,*

$$|\Pr[\boldsymbol{H}_2^{\mathcal{A}} = 1] - \Pr[\boldsymbol{H}_1^{\mathcal{A}} = 1]| \leq \frac{\varepsilon}{8}$$

Proof. From Theorem A.6 in [AGQY22], it follows that for every quantum algorithm making at most q queries, the distinguishing advantage between F and F' is bounded by $\frac{300\cdot q^3}{R} < \frac{\varepsilon}{8}$. □

$\mathbf{H}_3$: Bloating the dual Same as $\mathbf{H}_2$, except the oracle D is replaced with D', which on input y, accepts an evasive superspace of the dual subspace corresponding to the coset of y,

$$D'(y, \mathbf{v}) = \begin{cases} 1 & \text{if } \mathbf{z}^\top \mathbf{A}_y^{(1)} = \mathbf{0}^{n-r-s} \\ 0 & \text{otherwise.} \end{cases}$$

, where $\mathbf{A}_y^{(1)} \in \mathbb{Z}_2^{n-r-s}$ denotes the last $n-r-s$ columns of $\mathbf{A}_y \in \mathbb{Z}_2^{k\times(n-r)}$.

Claim 4. *For any q query quantum algorithm $\mathcal{A}$,*

$$|\Pr[\boldsymbol{H}_2^{\mathcal{A}}(1^\lambda) = 1] - \Pr[\boldsymbol{H}_3^{\mathcal{A}}(1^\lambda) = 1] \leq \frac{\varepsilon}{8}.$$

Proof. By Lemma 19 in [SZ25], due to $k-(k-(n-r))-s = n-r-s$ for every $i \in [R]$, changing D to check for membership in $\mathbf{A}_y^{(1)\perp}$ instead of $\mathbf{A}_y^{\perp}$ is $O\left(\frac{q\cdot s}{\sqrt{2^{n-r-s}}}\right)$ indistinguishable, for any q query algorithm. Since we use the above indistinguishability R times, it follows that the distinguishing advantage in the current hybrid is $:= \Pr[\mathbf{H}_2^{\mathcal{A}}(1^\lambda) = 1] - R \cdot O\left(\frac{q\cdot s}{\sqrt{2^{n-r-s}}}\right) \geq \frac{7\varepsilon}{8} - O\left(\frac{k^2\cdot q^4\cdot s\cdot\frac{1}{\varepsilon}}{\sqrt{2^{n-r-s}}}\right) \geq \frac{3\varepsilon}{4}$, when $\frac{k^9\cdot q^7\cdot\frac{1}{\varepsilon^4}}{\sqrt{2^{n-r-s}}} \leq o(1)$. This implies the distinguishing advantage $\leq \frac{\varepsilon}{8}$. □

$\mathbf{H}_4$: In this hybrid, we simulate the oracles P, P^{-1}, D' from $\mathbf{H}_3$ using a coset partition function (Definition 4) that operates on a smaller input domain. Formally, given oracle access to any $(r+s, r, s)$ coset partition function $Q : \{0,1\}^{r+s} \to \{0,1\}^r$, the oracles P, P^{-1}, D' are simulated as follows:

1. Sample a random permutation $\Gamma : \{0,1\}^n \to \{0,1\}^n$ and for every y, choose a random full rank matrix $C_y \in \mathbb{Z}_2^{k\times n}$, and a random vector $d_y \in \mathbb{Z}_2^k$.
2. Simulating $P(x \in \{0,1\}^n)$.
   - $(x_0 \in \{0,1\}^{r+s}, x_1 \in \{0,1\}^{n-r-s}) \leftarrow \Gamma(x)$.
   - $y \leftarrow Q(x_0)$.
   - $u \leftarrow (C_y \cdot \Gamma(x) + d_y)$.

- Output (y, u).

3. Simulating $P^{-1}(y \in \{0,1\}^r, u \in \{0,1\}^k)$.
   - $z \leftarrow C_y^{-1} \cdot (u - d_y)$.
   - $(z_0 \in \{0,1\}^{r+s}, z_1 \in \{0,1\}^{n-r-s}) := z$.
   - If $Q(z_0) = y$, output $\Gamma^{-1}(z)$. Otherwise, output $\perp$.
4. Simulating $D'(\mathbf{v}, y)$
   - $\mathbf{A}^{(1)\perp} :=$ last $n - r - s$ columns of C_y.
   - Output 1 iff $\mathbf{v}^T \cdot \mathbf{A}_y^{(1)} = 0^{n-r-s}$.

Claim 5.

$$\Pr[\boldsymbol{H}_4^{\mathcal{A}}(1^\lambda) = 1] = \Pr[\boldsymbol{H}_3^{\mathcal{A}}(1^\lambda) = 1]$$

Proof. We will argue that $\Gamma, \{C_y, d_y\}_y$ and Q perfectly simulate P, P^{-1}, D'. Observe that simulating P, P^{-1}, D' using Q is implicitly setting the following parameters:

- $\mathbf{A}_y$: The preimage set $Q^{-1}(y)$ is a coset, which can be described as the set $\{\mathbf{B}_y \cdot r + r_y\}$ as r ranges over $\mathbb{Z}_2^s$. Thus $\mathbf{A}_y := \mathbf{C}_y \cdot K_y$, where $K_y = \begin{pmatrix} \mathbf{B}_y \in \mathbb{Z}^{(r+s)\times s} & \\ & \ddots \quad \mathbf{I}_{n-r-s} \end{pmatrix}$. Since K_y is full rank and $\mathbf{C}_y$ is random, this implies that $\mathbf{A}_y$ is random.
- $b_y := d_y + \mathbf{C}_y \cdot (r_y || 0^{n-r-s})$. This follows by construction, now since d_y is random, this implies that b_y is also random.
- Π: Define an augmented function $Q' : \{0,1\}^n \to \{0,1\}^n$. On input $\mathbf{z}$, the first r bits of $Q'(\mathbf{z})$ are set to $y = Q(\mathbf{z})$. Define the function $\bar{J}(z)$ that outputs the unique vector in $\mathbb{Z}_2^{n-r}$ such that $\mathbf{z} = K_y \cdot \bar{J}(z) + r_y$. Then define $Q'(\mathbf{z}) = (Q(\mathbf{z}, \bar{J}(\mathbf{z}))$. This implies that $\Pi = Q' \circ \Gamma$.

□

$\mathbf{H}_5$: This hybrid describes the following experiment

- Simulate P, P^{-1}, D' given oracle access to $Q : \{0,1\}^{r+s} \to \{0,1\}^r$ as in $\mathbf{H}_3$.
- Run $\mathcal{A}_1^Q(1^\lambda)$ until it outputs a string $y \in \{0,1\}^r$ and a state on registers $(\mathcal{U}, \mathcal{R})$. Here $\mathcal{U}$ is k-qubit register and $\mathcal{R}$ holds the internal state of $\mathcal{A}$.
- Initialize a register T in the state $|0\rangle^n$ and apply the map $(u, 0) \to (u, C_y^{-1}(u - d_y))$ on the joint system $\mathcal{U} \otimes T$.
- Measure the first $r + s$ bits of register T in the computational basis.
- Apply the map $(x, 0) \to (x, C_y \cdot x + d_y)$ on the resulting state in register $T \otimes \mathcal{U}$.
- Uncompute the register T.
- Sample $b \leftarrow \{0,1\}$. If $b = 0$, do nothing, and if $b = 1$ measure $(\mathcal{B}, \mathcal{U})$ in the computational basis.
- Run $\mathcal{A}_2^Q(\mathcal{U}, \mathcal{R})$ until it outputs a bit b'. The experiment outputs 1 if $b = b'$ and 0 otherwise.

Claim 6. *For all QPT algorithms $\mathcal{A}$,*

$$|\Pr[\boldsymbol{H}_5^{\mathcal{A}}(1^\lambda) = 1] - \Pr[\boldsymbol{H}_4^{\mathcal{A}}(1^\lambda) = 1] \leq \mathit{negl}(\lambda).$$

Proof. Observe that $\mathcal{P}(x) = \left(y = Q\left(\overline{\Gamma(x)}\right), \mathbf{u} = \mathbf{C}_y \begin{bmatrix}\overline{\Gamma(x)} \\ \widetilde{\Gamma(x)}\end{bmatrix} + \mathbf{d}_y\right)$ where $\overline{\Gamma(x)}$ picks out the first $r+s$ bits and $\widetilde{\Gamma(x)}$ picks out the last $(n-r-s)$ bits. Now, for a fixed y, the state on register $\mathcal{U}$ has the following form:

$$\sum_{\overline{\Gamma(x)}:Q(\overline{\Gamma(x)})=y,\widetilde{\Gamma(x)}} \mathbf{C}_y \begin{bmatrix}\overline{\Gamma(x)} \\ \widetilde{\Gamma(x)}\end{bmatrix} + \mathbf{d}_y$$

Measuring the first $r+s$ bits of register T in the computational basis and then applying the map $(x,0) \to (x, C_y(x) + d_y)$ on the resulting state in register $T \otimes \mathcal{U}$ yields the following state:

$$\sum_{\widetilde{\Gamma(x)}} \mathbf{C}_y \begin{bmatrix}\overline{\Gamma(x)} \\ \widetilde{\Gamma(x)}\end{bmatrix} + \mathbf{d}_y$$

, where $\overline{\Gamma(x)}$ is such that $Q(\overline{\Gamma(x)}) = y$.
Now, [SZ25] construct collision resistant coset partition functions by composing 2 to 1 functions together. More concretely, let $H : \{0,1\}^n \to \{0,1\}^{n-1}$ be a random 2 to 1 function. Given this, the coset partition function $H^\ell : \{0,1\}^{n\ell} \to \{0,1\}^{(n-1)\ell}$ is the following:

$$H^\ell(x_1, \dots x_\ell) := H(x_1), \dots H(x_\ell)$$

In particular, constructing Q in this manner, we can conclude that, since it is a composition of 2 to 1 functions which are known to be collapsing from [Zha22] and composition preserves the collapsing property, for any computationally bounded adversary $\mathcal{A}$, the above state is indistinguishable from

$$\sum_{\overline{\Gamma(x)}:Q(\overline{\Gamma(x)})=y,\widetilde{\Gamma(x)}} \mathbf{C}_y \begin{bmatrix}\overline{\Gamma(x)} \\ \widetilde{\Gamma(x)}\end{bmatrix} + \mathbf{d}_y$$

and the claim then follows immediately. □

$\mathbf{H}_6$: In this hybrid, we "unsimulate" the oracles, and revert to defining the oracles P, P^{-1}, D' as in $\mathbf{H}_3$. This hybrid describes the following experiment:

- Sample $P, P^{-1}, D' \leftarrow \overline{\mathsf{OSS}}.\mathsf{Setup}(1^\lambda)$.
- Run $\mathcal{A}_1^{P,P^{-1},D'}(1^\lambda)$ until it outputs a string $y \in \{0,1\}^r$ and a state on registers $(\mathcal{U}, \mathcal{R})$. Here $\mathcal{U}$ is k-qubit register and $\mathcal{R}$ holds the internal state of $\mathcal{A}$.
- Initialize a register T in the state $|0\rangle^n$ and apply the map $(u,0) \to (u, (A_y^{-1}(u - b_y))$ on the joint system $\mathcal{U} \otimes T$.
- Uncompute $\mathcal{U}$ and measure the first s bits of register T in the computational basis, and then apply the map $(x,0) \to (x, A_y \cdot x + b_y)$ on the resulting state in register $T \otimes \mathcal{U}$.
- Uncompute the register T.
- Sample $b \leftarrow \{0,1\}$. If $b = 0$, do nothing, and if $b = 1$ measure $(\mathcal{B}, \mathcal{U})$ in the computational basis.
- Run $\mathcal{A}_2^{P,P^{-1}}(\mathcal{U}, \mathcal{R})$ until it outputs a bit b'. The experiment outputs 1 if $b = b'$ and 0 otherwise.

Claim 7.

$$\Pr[\boldsymbol{H}_6^{\mathcal{A}}(1^\lambda) = 1] = \Pr[\boldsymbol{H}_5^{\mathcal{A}}(1^\lambda) = 1]$$

Proof. This hybrid distributes identically to the previous hybrid by the simulation of P, P^{-1}, D' by Q in $\mathbf{H}_4$. □

$\mathbf{H}_7$: Same as $\mathbf{H}_6$ except, $\forall y \in [R], \mathbf{A}_y$ is sampled as follows: sample a random function $F : \{0,1\}^r \to \{0,1\}^{k\cdot(n-r+1)}$ and a random permutation $M : \{0,1\}^{n-r} \to \{0,1\}^{n-r}$. Interpret $F(y) = (\mathbf{A}'_y, \mathbf{b}_y)$ where $\mathbf{A}'_y \in \mathbb{Z}_2^{k\times(n-r)}$ is a full-rank matrix and $\mathbf{b} \in \mathbb{Z}_2^n$, and interpret $M(y) = M_y$, where $M_y \in \mathbb{Z}_2^{(n-r)\times(n-r)}$ is a full-rank matrix. Set $\mathbf{A}_y = \mathbf{A}'_y \circ M_y$.

Claim 8.

$$\Pr[\boldsymbol{H}_7^{\mathcal{A}}(1^\lambda) = 1] = \Pr[\boldsymbol{H}_6^{\mathcal{A}}(1^\lambda) = 1]$$

Proof. Ths distribution of $\mathbf{A}_y$ is identical in both hybrids, $\mathbf{H}_7$ and $\mathbf{H}_8$. □

$\mathbf{H}_8$: Same as $\mathbf{H}_7$ except we change how $\Pi : \{0,1\}^n \to \{0,1\}^n$ is sampled in $\overline{\mathsf{OSS}}$:

1. Sample a random permutation $\Pi' : \{0,1\}^n \to \{0,1\}^n$.
   - Let $\sigma : \{0,1\}^n \to \{0,1\}^n$ be the following permutation: $\sigma(x)$: Parse input as $(y \in \{0,1\}^r, z \in \{0,1\}^{n-r})$, and output $(y, M_y^{-1}(z))$.
2. $\Pi(x) = \Pi' \circ \sigma(x)$.

Claim 9.

$$\Pr[\boldsymbol{H}_8^{\mathcal{A}}(1^\lambda) = 1] = \Pr[\boldsymbol{H}_7^{\mathcal{A}}(1^\lambda) = 1]$$

Proof. The 2 hybrids are distributed identically. □

Claim 10. *If there exists an adversary $(\mathcal{A}_1, \mathcal{A}_2)$ that wins the experiment in $\boldsymbol{H}_8$ with non negligible advantage, then there exists an adversary which wins the coset-collapsing game with non negligible advantage.*

Proof. First, observe that the oracles P, P^{-1}, D' in $\mathbf{H}_8$ are the following:

- Sample a random permutation $\Pi' : \{0,1\}^n \to \{0,1\}^n$.
  - Let $\sigma : \{0,1\}^n \to \{0,1\}^n$ be the following permutation: $\sigma(x)$: Parse input as $(y \in \{0,1\}^r, z \in \{0,1\}^{n-r})$, and output $(y, M_y^{-1}(z))$.
- $\Pi(x) = \Pi' \circ \sigma(x)$. Let $H(x)$ be the first r output bits of $\Pi(x)$, and let $J'(x)$ denote the remaining $n-r$ bits.
- Sample a random function $F : \{0,1\}^r \to \{0,1\}^{k\cdot(n-r+1)}$ and a random permutation $M : \{0,1\}^{n-r} \to \{0,1\}^{n-r}$. For every $y \in [R]$, Interpret $F(y) = (\mathbf{A}'_y, \mathbf{b}_y)$ where $\mathbf{A}'_y \in \mathbb{Z}_2^{k\times(n-r)}$ is a full-rank matrix and $\mathbf{b} \in \mathbb{Z}_2^n$, and interpret $M(y) = M_y$, where $M_y \in \mathbb{Z}_2^{(n-r)\times(n-r)}$ is a full-rank matrix. Set $\mathbf{A}_y = \mathbf{A}'_y \circ M_y$. Let $S_y = \{\mathbf{A}'_y\mathbf{x} + \mathbf{b}_y \mid \mathbf{x} \in \mathbb{Z}_2^{n-r}\}$ be the coset defined by $(\mathbf{A}'_y, \mathbf{b}_y)$.
- Now, the functions P, P^{-1}, D' are defined as follows:

$$P(x) = (y, \mathbf{A}'_y \cdot J(x) + \mathbf{b}_y) \in \{0,1\}^n \to \{0,1\}^r \times \mathbb{Z}_2^k \text{ where } y = H(x)$$

$$P^{-1}(y, \mathbf{u}) = \begin{cases} \Pi'(y, \mathbf{z}) & \text{if } \exists \mathbf{z} \in \mathbb{Z}_2^{n-r} \text{ such that } \mathbf{A}'_y\mathbf{z} + \mathbf{b}_y = \mathbf{u} \\ \perp & \text{otherwise.} \end{cases}$$

$$D'(y, \mathbf{v}) = \begin{cases} 1 & \text{if } \mathbf{z}^\top \mathbf{A}_y^{(1)} = \mathbf{0}^{n-r-s} \\ 0 & \text{otherwise.} \end{cases}$$

The experiment in $\mathbf{H}_8$ is the following:

- Run $\mathcal{A}_1^{P,P^{-1},D'}(1^\lambda)$ until it outputs a string $y \in \{0,1\}^r$ and a state on registers $(\mathcal{U}, \mathcal{R})$. Here $\mathcal{U}$ is k-qubit register and $\mathcal{R}$ holds the internal state of $\mathcal{A}$.

- Initialize a register T in the state $|0\rangle^n$ and apply the map $(u,0) \to (u, (\mathbf{A}_y^{-1}(u - b_y))$ on the joint system $\mathcal{U} \otimes T$.
- Measure the first s bits of register T in the computational basis, and then apply the map $(x,0) \to (x, \mathbf{A}_y \cdot x + b_y)$ on the resulting state in register $T \otimes \mathcal{U}$.
- Uncompute the register T.
- Sample $b \leftarrow \{0,1\}$. If $b = 0$, do nothing, and if $b = 1$ measure $(\mathcal{B},\mathcal{U})$ in the computational basis.
- Run $\mathcal{A}_2^{P,P^{-1}}(\mathcal{U},\mathcal{R})$ until it outputs a bit b'. The experiment outputs 1 if $b = b'$ and 0 otherwise.

Now, consider the following experiment.

Definition 11 (Coset-collapsing game with Dual Access)**.** *This is a game between a challenger and a two-stage adversary $(\mathcal{A}_1, \mathcal{A}_2)$. There is a public subspace $S \subset \mathbb{F}_2^n$ that is known to all parties.*

1. *The challenger samples a random subspace $T \subset S$, and sends to A_1 the oracle $O_{T^\perp}$. Note that $T^\perp$ is a random superspace of $S^\perp$.*
2. *The adversary $\mathcal{A}_1$ outputs a quantum state on two registers $\mathcal{R},\mathcal{U}$, and sends $\mathcal{U}$ to the challenger and $\mathcal{R}$ to the adversary $\mathcal{A}_2$.*
3. *The challenger does the following:*
   (a) *Sample a random subspace $T \subset S$ of $\dim(T) = k$.*
   (b) *Challenger checks in superposition $O_S(\mathcal{U}) = \mathsf{Accept}$; if the check fails, the adversary automatically loses the game.*
   (c) *Perform a measurement that projects $\mathcal{U}$ to a coset of T in S. Concretely, let $\mathsf{co}(T)$ be the set of coset representatives of T. Then, the measurement is described by the projectors $\{\Pi_u\}_{\mathsf{co}(T)}$, where $\Pi_u := \sum_{t \in T} |t+u\rangle\langle t+u|$ for $u \in \mathsf{co}(T)$.*
   (d) *The challenger samples random $b \leftarrow \{0,1\}$. If $b = 0$, do nothing and send the register $\mathcal{U}$ to $\mathcal{A}_2$. If $b = 1$, measure the $\mathcal{U}$ register in the computational basis and then send it to $\mathcal{A}_2$.*
4. *Adversary $\mathcal{A}_2$ outputs a bit b' and wins if $b' = b$.*

Now by construction, $\mathbf{H}_8$ is exactly the coset-collapsing game with dual access when $T = \text{Colspan}(\mathbf{A}_y^{(1)})$, $S = \text{Colspan}(\mathbf{A}'_y)$. Assume for contradiction that there exists an adversary $(\mathcal{B}_1, \mathcal{B}_2)$ which has distinguishing advantage ε in $\mathbf{H}_8$, where ε is a non negligible function. Now consider an adversary $(\mathcal{A}_1, \mathcal{A}_2)$ for the coset collapsing game with dual access, with knowledge of the public subspace $S = \text{ColSpan}(\mathbf{A}'_y)$ for $y \in [R]$ and access to a membership oracle for $T^\perp = \text{Colspan}(\mathbf{A}_y^{(1)\perp})$. With probability $1/R$, $\mathcal{A}$ guesses correctly the value of y that $\mathcal{B}$ commits to in the game. Conditioned on guessing correctly, $(\mathcal{A}_1, \mathcal{A}_2)$ can run $(\mathcal{B}_1, \mathcal{B}_2)$, so the distinguishing advantage of $(\mathcal{A}_1, \mathcal{A}_2)$ is $\frac{1}{R} \cdot \varepsilon$, which is still non negligible. We will establish next that the coset collapsing game with dual access is indistinguishable from the coset-collapsing game (Definition 12), and prove that the coset-collapsing game is information theoretically hard (Theorem 7).Assuming this, we can complete the proof by invoking Theorem 7. □

□

3.4.2 Coset-collapsing game

We will first consider the following hybrids.
$\mathbf{H}_0$: This is the Coset collapsing game with dual access (Definition 11).
$\mathbf{H}_1$: Same as $\mathbf{H}_0$, but $\mathcal{A}_1$ is given access to $O_{S^\perp}$ instead of $O_{T^\perp}$.

Claim 11. *For any q query algorithm $(\mathcal{A}_1, \mathcal{A}_2)$,*

$$\Pr[1 \leftarrow \boldsymbol{H}_0^{(\mathcal{A}_1,\mathcal{A}_2)}] - \Pr[1 \leftarrow \boldsymbol{H}_1^{(\mathcal{A}_1,\mathcal{A}_2)}] \leq negl(\lambda)$$

Proof. We unbloated the dual in $\mathbf{H}_1$, and the proof is identical to the proof of Claim 4. □

$\mathbf{H}_2$: Same as $\mathbf{H}_1$, but $(\mathcal{A}_1, \mathcal{A}_2)$ are not given access to $O_{S^\perp}$.

Claim 12.

$$\Pr[1 \leftarrow \boldsymbol{H}_1^{(\mathcal{A}_1,\mathcal{A}_2)}] = \Pr[1 \leftarrow \boldsymbol{H}_2^{(\mathcal{A}_1,\mathcal{A}_2)}]$$

Proof. This claim follows because $(\mathcal{A}_1, \mathcal{A}_2)$ know the description of S, since its a public subspace. □

Now, $\mathbf{H}_2$ is the coset collapsing game, which we define more concretely below.

Definition 12 (Coset-collapsing game)**.** *This is a game between a challenger and a two-stage adversary $(\mathcal{A}_1, \mathcal{A}_2)$. There is a public subspace $S \subset \mathbb{F}_2^n$ that is known to all parties. Let $n \geq k$.*

1. *The adversary $\mathcal{A}_1$ outputs a quantum state on two registers $\mathcal{R}$,$\mathcal{U}$, and sends $\mathcal{U}$ to the challenger and $\mathcal{R}$ to the adversary $\mathcal{A}_2$.*
2. *The challenger does the following:*
   (a) *Sample a random subspace $T \subset S$ of $\dim(T) = k$.*
   (b) *Challenger checks in superposition $O_S(\mathcal{U}) = \mathsf{Accept}$; if the check fails, the adversary automatically loses the game.*
   (c) *Perform a measurement that projects $\mathcal{U}$ to a coset of T in S. Concretely, let $\mathsf{co}(T)$ be the set of coset representatives of T. Then, the measurement is described by the projectors $\{\Pi_u\}_{\mathsf{co}(T)}$, where $\Pi_u := \sum_{t\in T} |t+u\rangle\langle t+u|$ for $u \in \mathsf{co}(T)$.*
   (d) *The challenger samples random $b \leftarrow \{0,1\}$. If $b = 0$, do nothing and send the register $\mathcal{U}$ to $\mathcal{A}_2$. If $b = 1$, measure the $\mathcal{U}$ register in the computational basis and then send it to $\mathcal{A}_2$.*
3. *Adversary $\mathcal{A}_2$ outputs a bit b' and wins if $b' = b$.*

Theorem 7. *Every (potentially unbounded) quantum adversary wins the coset-collapsing game in Definition 12 with advantage at most $2^k/2^n$.*

Proof. Since the subspace S is public and known to the adversary, we can assume that the subspace-check in Item 2b of Definition 12 always passes, and can perform a change-of-basis so that $S = \mathbb{F}_2^{n'}$ where $n' = \dim(S)$.

After this change of basis, if the challenger samples $b = 1$, the channel it applies is simply Φ_1 as defined in Lemma 4, and if $b = 0$, the challenger applies the Φ_2 channel. Then, the theorem follows from Lemma 4. □

Lemma 4. *The diamond distance between the following two channels Φ_1, Φ_2 is*

$$\|\Phi_1 - \Phi_2\|_\diamond \leq \frac{2^k}{2^n}.$$

- Φ_1 *is defined as follows: Sample a random subspace* $T \subset \mathbb{F}_2^n$ *of* $\dim(T) = k$*, and project onto the cosets of* T*. Concretely, let* $\mathsf{co}(T)$ *be the set of coset representatives of* T*, and let* $\#T$ *be the number of subspaces* T *such that* $\dim(T) = k$*.*

$$\Phi_1(\rho) := \frac{1}{\#T} \sum_T \sum_{u \in \mathsf{co}(T)} \Pi_u \rho \Pi_u,$$

where $\Pi_u := \sum_{t \in T} |t+u\rangle\langle t+u|$ *for any* $u \in \mathsf{co}(T)$*.*

- Φ_2 *is defined as simply measuring in the computational basis*

$$\Phi_2(\rho) := \sum_{x \in \mathbb{F}_2^n} \Pi_x \rho \Pi_x,$$

where $\Pi_x := |x\rangle\langle x|$*.*

Proof. First let us define the following pairs of channels:

- Let Φ_1' (resp. Φ_2') be channels defined as follows: append an EPR pair $|\mathsf{EPR}\rangle_{\mathcal{C},\mathcal{D}} \propto \sum_x |x\rangle_{\mathcal{C}} \otimes |x\rangle_{\mathcal{D}}$ on registers $\mathcal{C}, \mathcal{D}$ and teleport the state on input register $\mathcal{A}$ to $\mathcal{D}$. Then, apply Φ_1 (resp. Φ_2) on register $\mathcal{D}$ and output the resulting state. Trace out registers $\mathcal{A}, \mathcal{C}$.
- Let Φ_1'' (resp. Φ_2'') be channels defined as follows: append an EPR pair on registers $\mathcal{C}, \mathcal{D}$. Apply the channel Φ_1 (resp. Φ_2) to register $\mathcal{D}$. Perform the "teleportation step": apply a Bell basis measurement on registers $\mathcal{A}, \mathcal{C}$ and then apply the Z-gate teleportation corrections on register $\mathcal{D}$. Trace out registers $\mathcal{A}, \mathcal{C}$.

Since Φ_1', Φ_2' simply teleports the input state and then applies Φ_1, Φ_2, it is clear that

$$\|\Phi_1 - \Phi_2\|_\diamond = \|\Phi_1' - \Phi_2'\|_\diamond.$$

The teleportation step consists of a Bell basis measurement on registers $\mathcal{A}, \mathcal{C}$ followed by a Pauli correction on register $\mathcal{D}$. Claim 13 shows that Φ_1 and Φ_2 absorb the Pauli X corrections and commute with the Pauli Z corrections, which implies that

$$\|\Phi_1 - \Phi_2\|_\diamond = \|\Phi_1'' - \Phi_2''\|_\diamond.$$

Claim 13. *Let* Φ_1', Φ_2' *and* Φ_1'', Φ_2'' *be channels as defined above. Then,* $\Phi_1' = \Phi_1''$ *and* $\Phi_2' = \Phi_2''$*.*

Proof. Observe that Φ_1, Φ_2 are both channels of the form $\Phi(\rho) = \sum_i p_i \Pi_i \rho \Pi_i$ for some $\{(p_i, \Pi_i, S_i)\}_{i \in \mathcal{I}}$ such that

$$\Pi_i = \sum_{x \in S_i} |x\rangle\langle x|,$$

where $\mathcal{I}$ is a finite index set, $\sum_i p_i = 1$ are probability masses, Π_i are projections[12] and $S_i \in \{0,1\}^n$ are (not necessarily disjoint) sets that cover all of $\{0,1\}^n$. When convenient, we will alternate between the following two notations: $\Pi_i = \Pi_{S_i}$.

- For Φ_1, the index set is $\mathcal{I} = \{(T, u)\}$ where T is a k-dimensional subspace of $\mathbb{F}_2^n$ and $u \in \mathsf{co}(T)$ is a representative of one of the cosets of T. The probabilities $p_i = \frac{1}{\#T}$ and $S_{T,u} = T + u$.
- For Φ_2, the index set is particular simple $\mathcal{I} = \{x \mid x \in \mathbb{F}_2^n\}$ and $S_x = \{x\}$.

[12] Note that it is not true in the case of Φ_1 that $\sum_i \Pi_i = I$.

The channels Φ_1, Φ_2 have the further property that the probability distribution $\{p_i\}_{i\in\mathcal{I}}$ is uniform, and for every $\overline{x} \in \{0,1\}^n$, shifting by $\overline{z}$ simply rotates the sets S_i. That is, $\{S_i\}_{i\in\mathcal{I}} = \{S_i + \overline{x}\}_{i\in\mathcal{I}}$.

For any Pauli corrections $\overline{x}, \overline{z}$, and for any input state ρ, we have that

$$\begin{aligned}\Phi(\mathsf{X}^{\overline{x}}\mathsf{Z}^{\overline{z}}\rho\mathsf{X}^{\overline{x}}\mathsf{Z}^{\overline{z}}) &= \sum_i p_i\, \Pi_i\, \mathsf{X}^{\overline{x}}\mathsf{Z}^{\overline{z}}\, \rho\, \mathsf{X}^{\overline{x}}\mathsf{Z}^{\overline{z}}\Pi_i = \sum_i p_i \Pi_{S_i+\overline{x}}\mathsf{Z}^{\overline{z}}\rho\mathsf{Z}^{\overline{z}}\Pi_{S_i+\overline{x}} \\ &= \mathsf{Z}^{\overline{z}}\sum_i p_i\Pi_i\rho\Pi_i\mathsf{Z}^{\overline{z}} = \mathsf{Z}^{\overline{z}}\Phi(\rho)\mathsf{Z}^{\overline{z}}.\end{aligned}$$

□

Since these channels are first applying Φ_1 (resp. Φ_2) to half of the EPR pair and the performing some additional measurements as part of the teleportation step (which can only decrease the distance), we only need to worry about how Φ_1, Φ_2 act on halves of EPR pairs. That is, for every bipartite state $\rho_{\mathcal{A},\mathcal{B}}$,

$$\|(\Phi_1'' \otimes I)\rho_{\mathcal{A},\mathcal{B}} - (\Phi_2'' \otimes I)\rho_{\mathcal{A},\mathcal{B}}\|_1 \leq \|(I\otimes\Phi_1)\,|\mathsf{EPR}\rangle\langle\mathsf{EPR}| - (I\otimes\Phi_2)\,|\mathsf{EPR}\rangle\langle\mathsf{EPR}|\,\|_1.$$

Therefore,

$$\begin{aligned}\|\Phi_1 - \Phi_2\|_\diamond &= \|\Phi_1'' - \Phi_2''\|_\diamond \\ &= \max_{\rho_{\mathcal{A},\mathcal{B}}} \|(\Phi_1''\otimes I)(\rho_{\mathcal{A},\mathcal{B}}) - (\Phi_2''\otimes I)(\rho_{\mathcal{A},\mathcal{B}})\|_1 \\ &\leq \|(I\otimes\Phi_1)\,|\mathsf{EPR}\rangle\langle\mathsf{EPR}| - (I\otimes\Phi_2)\,|\mathsf{EPR}\rangle\langle\mathsf{EPR}|\,\|_1.\end{aligned}$$

For channels Φ of the form in the proof of Claim 13,

$$\begin{aligned}I\otimes\Phi(|\mathsf{EPR}\rangle\langle\mathsf{EPR}|) &= \sum_i\sum_{x,x'} p_i (I\otimes\Pi_i)\,|x,x\rangle\langle x',x'|\,(I\otimes\Pi_i) \\ &= \sum_i \sum_{x,x'\in S_i} p_i\,|x,x\rangle\langle x',x'| \\ &= \mathsf{CNOT}(\Phi(|+\rangle\langle+|^{\otimes})\otimes|0^n\rangle\langle 0^n|).\end{aligned}$$

Since the map $\mathsf{CNOT}(\cdot\otimes|0^n\rangle\langle 0^n|)$ preserves the trace distance, we have that

$$\|(I\otimes\Phi_1)\,|\mathsf{EPR}\rangle\langle\mathsf{EPR}| - (I\otimes\Phi_2)\,|\mathsf{EPR}\rangle\langle\mathsf{EPR}|\,\|_1 = \|\Phi_1(|+\rangle\langle+|^{\otimes n}) - \Phi_2(|+\rangle\langle+|^{\otimes n})\|_1,$$

and Lemma 5 completes the proof. □

Lemma 5.

$$\left\|\Phi_1\left(|+\rangle\langle+|^{\otimes n}\right) - \Phi_2\left(|+\rangle\langle+|^{\otimes n}\right)\right\|_1 \leq \frac{2^k}{2^n}.$$

Proof. For a fixed subspace T of $\dim(T) = k$,

$$\begin{aligned}\sum_{u\in\mathsf{co}(T)}\Pi_u\,|+\rangle\langle+|^{\otimes n}\,\Pi_u &= \sum_{u\in\mathsf{co}(T)}\left(\sum_{t\in T}|t+u\rangle\langle t+u|\right)\left(\frac{1}{2^n}\sum_{x,x'\in\mathbb{F}_2^n}|x\rangle\langle x'|\right)\left(\sum_{t'\in T}|t'+u\rangle\langle t'+u|\right) \\ &= \frac{1}{2^n}\sum_{u\in\mathsf{co}(T)}\sum_{t,t'\in T}\sum_{x,x'}|t+u\rangle\langle t+u|x\rangle\langle x'|t'+u\rangle\langle t'+u| \\ &= \frac{1}{2^n}\sum_{u\in\mathsf{co}(T)}\sum_{t,t'\in T}|t+u\rangle\langle t'+u|\,.\end{aligned}$$

Grouping the cosets of T together, this is a block diagonal matrix containing 2^{n-k} blocks, where each block is $\frac{1}{2^n}\mathbf{1}\mathbf{1}^\top 2^k \times 2^k$, where $\mathbf{1}\mathbf{1}^\top$ is the all ones matrix.

Average over all possible subspaces T, we can argue that by symmetry, every diagonal entry must be the same value, say α, and every off-diagonal entry must also be the same value β

$$\Phi_1\left(|+\rangle\langle+|^{\otimes n}\right) = \sum_x \alpha\,|x\rangle\langle x| + \sum_{x\neq x'} \beta\,|x\rangle\langle x'|\,.$$

By an averaging argument, we get that $\alpha = \frac{1}{2^n}$ and $\beta = \frac{2^k-1}{2^{2n}-2^n}$.

Channel Φ_2 is simply the full standard basis measurement, so

$$\Phi_2\left(|+\rangle\langle+|^{\otimes n}\right) = \frac{1}{2^n}\sum_{x\in\mathbb{F}_2^n} |x\rangle\langle x|\,.$$

Since $\mathbf{1}_d\mathbf{1}_d^\top - I_d$ has eigenvalues $(0,-1,-1,\ldots,-1)$, we know that $\|\mathbf{1}_d\mathbf{1}_d^\top - I_d\|_1 = d-1$, and

$$\left\|\Phi_1\left(|+\rangle\langle+|^{\otimes n}\right) - \Phi_2\left(|+\rangle\langle+|^{\otimes n}\right)\right\|_1 = \beta\cdot(2^n-1) = \frac{2^k-1}{2^n}.$$

□

4 Post-quantum Succinct Ideal Obfuscation

Definition 13 (Post-quantum succinct ideal obfuscation)**.** *A (post-quantum) succinct ideal obfuscator in the oracle model consists of the following algorithms.*

- $\mathsf{Obf}(1^\lambda, P) \to (O[P], \mathsf{pp})$*: The obfuscator takes as input the security parameter* 1^λ *and the description of a classical Turing machine* P *with single-bit output, and outputs the description of a* circuit $O[P]$ *and public parameters* pp*.*
- $\mathsf{Eval}^{O[P]}(\mathsf{pp}, x) \to y$*: The evaluation algorithm has oracle access to* $O[P]$*, takes an input* x*, along with public parameters* pp*, and produces an output* y*.*

It should satisfy the following properties.

- ***Correctness.*** *For any* $1^\lambda, P, x$*, and* $(O[P], \mathsf{pp}) \in \mathsf{Obf}(1^\lambda, P)$*, it holds that* $\mathsf{Eval}^{O[P]}(\mathsf{pp}, x) = P(x)$*.*
- ***Succinctness.*** *There exists a fixed polynomial* p *such that for any program* P *and* $(O[P], \mathsf{pp}) \leftarrow \mathsf{Obf}(1^\lambda, P)$*, it holds that* $|(O[P], \mathsf{pp})| = p(\lambda, |P|)$*.*
- ***Ideal obfuscation.*** *There exists a QPT simulator* $\mathcal{S}$ *such that for any QPT adversary* $\mathcal{A}$*, program* P*, and auxiliary information* z*,*

$$\left\{\mathcal{A}^{O[P]}(\mathsf{pp}, z) : O[P], \mathsf{pp} \leftarrow \mathsf{Obf}(1^\lambda, P)\right\} \approx \left\{\mathcal{S}^P(z)\right\}.$$

Classically, the above follows from FHE and SNARKs in the oracle model in a straightforward manner. In this section, we prove that the same construction is also post-quantum secure. Before describing the construction, we will introduce some relevant prelimnaries.

4.1 SNARKs

Definition 14. *A succinct non-interactive argument of knowledge (SNARK) for NP in the quantum random oracle model (QROM) consists of the following algorithms. Let $H : \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$ be a fixed-size random oracle.*

- $\mathsf{Prove}^H(1^\lambda, C, x) \to (z, \pi)$*: The p.p.t. prove algorithm takes as input the security parameter 1^λ, the description of a classical deterministic computation C, and an input x, and produces an output z and a proof π.*
- $\mathsf{Ver}^H(1^\lambda, C, z, \pi) \to \{\top, \bot\}$*: The deterministic classical verification algorithm takes as input the security parameter 1^λ, the description of a classical deterministic computation C, an output z, and a proof π, and either accepts or rejects.*

It should satisfy the following properties.

- ***Completeness.*** *For any λ, C, x, z such that $C(x) = z$,*

$$\Pr_H\left[\mathsf{Ver}^H(1^\lambda, C, z, \pi) = \top : (z, \pi) \leftarrow \mathsf{Prove}^H(1^\lambda, C, x)\right] = 1.$$

- ***Succinctness.*** *There exists a fixed polynomial p such that for any time T computation C, the running time of $\mathsf{Ver}^H(1^\lambda, C, z, \pi)$ is $p(\lambda, \log T, |C|, |z|)$. Note that $|x|$ and the time T might be much larger than $p(\lambda, \log T, |C|, |z|)$.*
- ***Argument of Knowledge.*** *The argument $(\mathsf{Prove}, \mathsf{Ver})$ is an argument of knowledge with extraction probability κ if there exists a polynomial time quantum extractor $\mathcal{E}$ such that, for every input x and t query quantum oracle algorithm $\tilde{P}$, if, over a random oracle H, for $\pi := \tilde{P}^H$, it holds that $\mathsf{Ver}^H(z, \pi) = 1$ with probability μ, the probability that $\mathcal{E}^{\tilde{P}}(1^t, 1^\lambda)$ outputs a valid witness for π is at least $\kappa(t, \mu, \lambda)$. Here the notaion $\mathcal{E}^{\tilde{P}}$ denotes that $\mathcal{E}$ has black box access to $\tilde{P}$.*

4.2 Probabilistically Checkable Proofs (PCPs)

Definition 15. *A probabilistically checkable proof (PCP) for a relation $\mathcal{R}$ with soundness error ε, proof length ℓ and alphabet Σ is a pair of poly-time algorithms $(\mathbf{P}, \mathbf{V})$ for which the following holds:*

- ***Completeness:*** *For every instance-witness pair $(x, w) \in \mathcal{R}$, $\mathbf{P}(x, w)$ outputs a proof string $\Pi : [\ell] \to \Sigma$ such that $\Pr[\mathbf{V}^\Pi(x) = 1] = 1$.*
- ***Soundness:*** *For every instance $x \notin \mathcal{L}(\mathcal{R})$ and proof string $\Pi : [\ell] \to \Sigma$, $\Pr[\mathbf{V}^\Pi(x) = 1] \leq \varepsilon$.*
- ***Proof of Knowledge:*** *The PCP $(\mathbf{P}, \mathbf{V})$ has knowledge error k if there exists a polynomial time extractor $\mathbf{E}$ such that for every witness x and proof string $\Pi : [\ell] \to \Sigma$, if $\Pr[\mathbf{V}^\Pi(x) = 1] > k$, then $\mathbf{E}(x, \Pi)$ outputs a valid witness w.*

4.3 The construction of Micali

Let (**P**,**V**) be the corresponding PCP with prover **P** and verifier **V**. Before describing the construction formally, we will define some notation relevant to Merkle trees. Given a domain Z and $\ell = 2^d$, a *Merkle tree* on a list $\mathbf{v} = (v_i)_{i \in \ell} \in Z^\ell$ with respect to the function $H : Z \times Z \to Z$ is a list of values $(z_{k,i})_{k \in \{0,\ldots d\}, i \in [2^k]} \in Z^{2\ell}$, where

$$\forall i \in \ell, z_{d,i} = v_i$$

$$\forall k \in \{0, \ldots d-1\}, \forall i \in [2^k], z_{k,i} = H(z_{k+1,2i-1}, z_{k+1,2i})$$

The root of the merkle tree, denoted rt is $z_{0,1}$. Given $i \in [\ell]$, where $[\ell] = 2^d$, and d is the depth of the tree, the authentication path for the i^{th} leaf is

$$ap = (a_k)_{k \in d},$$

where a_k is the sibling of the k^{th} node on the path from v_i (the i^{th} node) to rt.
To check whether an authentication path is valid, we define

$$\text{CheckPath}^H(\mathsf{rt}, i, v, ap) = 1 \text{ iff } (\mathbf{x}, \mathbf{y}) \leftarrow \text{ Expand}^{\mathbf{H}}(\mathsf{rt}, \mathbf{i}, \mathbf{v}, \mathbf{ap}) \text{ has } \mathbf{y_0} = \mathsf{rt},$$

where $\text{Expand}^H(\mathsf{rt}, i, v, ap)$ outputs all the input-output pairs, $(\mathbf{x}, \mathbf{y})$, that arise when validating an authentication path.

Extracting Trees: We will recall the algorithm $\mathsf{Extract}$ that is used to obtain a Merkle tree rooted at a chosen entry in the database corresponding to the compresssed oracle for H. This algorithm is used in [CMS19]. In more detail, $\mathsf{Extract}$ receives a database $D : Z \times Z \to Z$, a root value $\mathsf{rt} \in Z$, and a height bound d, and outputs a rooteed binary tree $T = (V, E)$ of depth at most d where $V \subseteq Z \times \{0,1\}^{\leq d}$.

$\text{Extract}(D, \mathsf{rt}, d)$:

1. Initialize the vertex set with the root $V := \{(\mathsf{rt}, \emptyset)\}$ and the edge set to be empty $E := \emptyset$.
2. While there exists an unmarked vertex $(u, s) \in V$ such that $u \in \text{Im}(D)$ and $s \in \{0,1\}^{<d}$:
   - Mark the vertex (u, s).
   - If $\exists x \neq x' : D(x) = D(x') = u$, return $\perp$.
   - Let x_0, x_1 be the unique values such that $D(x_0, x_1) = u$.
   - Update the vertex set $V := V \cup \{(x_0, s||0), (x_1, s||1)\}$.
   - Update the edge set $E := E \cup \{((u, s), (x_0, s||0)), ((u, s), (x_1, s||1))\}$.

   Given a tree $T := \mathsf{Extract}(D, \mathsf{rt}, d)$we can define a string $\mathsf{leaves}(T)$ where for each $i \in \{0,1\}^d$, if there exists a vertex $(x, i) \in T$ then $\mathsf{leaves}(T)_i := x$, else $\mathsf{leaves}(T)_i := \perp$.

The formal construction of the Micali SNARK is now as follows:

- $\mathsf{SNARK.Prove}(x, \text{w}) \to \pi$:
  - Prover P uses H to commit to the proof string $\Pi \leftarrow \mathbf{P}(x, \text{w})$ via a Merkle tree, obtaining a corresponding root rt.
  - Randomness r is derived by applying H to rt.
  - P simulates $\mathbf{V}(\Pi, x, r)$ to deduce the queried locations $(v_i)_{i \in [q]}$ in Π.
  - Output $\pi = \{\mathsf{rt}, (v_i)_{i \in [q]}, (ap_i)_{i \in [q]}\}$.
- $\mathsf{SNARK.Verify}(\pi, x) \to 0/1$:
  - Using H (by making queries to it), check that, for $i \in [q]$,

    $$\text{CheckPath}^H(\mathsf{rt}, i, v, ap) = 1$$

  - Let $r = H(\mathsf{rt})$.
  - Run $\mathbf{V}(x, r)$:
    - For every query $j_i \in [\ell]$, where j_i is the is the location of the i^{th} query made by $\mathbf{V}$, V runs $\text{CheckPath}^H(\mathsf{rt}, ap_i, v, j)$, and returns v_i if $\text{CheckPath}^H(\mathsf{rt}, ap_i, v, j) = 1$.
  - Output the same answer as $\mathbf{V}(x, r)$.

Theorem 8. *[CMS19] Let $(\mathbf{P}, \mathbf{V})$ be a PCP for relation $\mathcal{R}$ that has soundness error ε, proof length ℓ and query complexity q. The Micali construction when based on $(\mathbf{P}, \mathbf{V})$ is a non-interactive argument for $\mathcal{R}$ that has soundness error $O(t^2\varepsilon + t^3/2^\lambda)$ against quantum attackers $\tilde{P}$ that make at most $t - O(q \log \ell)$ queries to H.*

Now we will describe the knowledge extractor for the Micali SNARK in the QROM for any t' query algorithm $\tilde{P}$.

$\mathcal{E}*^{\tilde{P}}(1^{t'}, 1^\lambda)$:

1. Set $t = t' + O(q \log \ell)$.
2. Compute the quantum state $|\mathrm{Sim}^*(\tilde{P})\rangle$ by simulating $\tilde{P}$, and recording its queries to the random oracle H with compressed oracles.
3. Measure the database register to get a database D.
4. For each $\mathsf{rt} \in D$:
   - Run $T \leftarrow \mathrm{Extract}(D, \mathsf{rt}, d)$.
   - Set Π_T equal to $\mathrm{leaves}(T)$ on all points where $\mathrm{leaves}(T) \neq \perp$, and 0 otherwise.
   - Run $x \leftarrow \mathbf{E}(\Pi_T)$.
   - Output x.

Lemma 6. *[CMS19] The Micali construction, when based on a PCP of knowledge with knowledge error k, has the following guarantee:*

- *For any quantum attacker $\tilde{P}$ that makes at most t' queries to H, the following holds with probability $\Omega(\mu - t^2 k - t^3/2^\lambda)$:*
  - *For all inputs (ct', π) such that $\mathsf{SNARK.Verify}(\pi) = 1$ with probability $\geq \mu$, $\mathcal{E}*^{\tilde{P}}(1^{t'}, 1^\lambda)$ extracts a valid witness x.*

4.4 Construction

Let $U(\cdot, \cdot)$ be the universal circuit that takes as input a Turing machine description P and an input x and outputs $P(x)$. Let $U_x := U(\cdot, x)$ be the same universal circuit except that the input is hardcoded and fixed to x, so it only takes as input a Turing machine description P.

Let H be a random oracle. We describe the construction of succinct ideal obfuscation in Figure 1.

4.5 Correctness

Theorem 9. *The construction in Figure 1 satisfies correctness as defined in Definition 13.*

Proof. Correctness of the succinct obfuscation scheme in Figure 1 follows from the completeness of the SNARK and the correctness of the FHE scheme. □

4.6 Succinctness

Theorem 10. *The construction in Figure 1 satisfies succinctness as defined in Definition 13.*

Proof. For a Turing machine with description P, its obfuscation consists of a circuit $O[P]$ that's accessible as an oracle and a FHE ciphertext $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, P)$. The ciphertext ct encrypts the description of P, so its size is bounded $|\mathsf{ct}| \leq \mathrm{poly}(\lambda, |P|)$, which is succinct. The circuit $O[P](\mathsf{ct}', \pi)$ consists of two circuits: (1) the SNARK verification circuit and (2) the FHE decryption circuit. By the succinctness of the SNARK scheme and compactness of the FHE scheme, both these circuits have size bounded by $\mathrm{poly}(\lambda, |P|)$. Therefore the construction in Figure 1 satisfies succinctness. □

Succinct Ideal Obfuscation

- $\mathsf{Obf}^H(1^\lambda, P)$:
  - $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$
  - $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, P)$
  - Define the circuit $O^H_{\mathsf{ct},\mathsf{sk}}(\mathsf{ct}', \pi)$ to take as input a ciphertext ct' and proof π. It runs $\mathsf{SNARK.Verify}^H((\mathsf{ct}, \mathsf{ct}'), \pi)$ to check that π is a valid proof of the following NP statement:

    $$\exists x : \mathsf{ct}' = \mathsf{FHE.Eval}(\mathsf{pk}, U_x, \mathsf{ct}).$$

    If $\mathsf{SNARK.Verify}^H((\mathsf{ct}, \mathsf{ct}'), \pi)$ accepts, output $y \leftarrow \mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct}')$ and otherwise output $\perp$.
  - Output the obfuscation $(O[P], \mathsf{pp})$ where $O[P] := O^H_{\mathsf{ct},\mathsf{sk}}$ and $\mathsf{pp} := \mathsf{ct}$.
- $\mathsf{Eval}^{O[P],H}(\mathsf{pp}, x)$:
  - Parse $\mathsf{pp} = \mathsf{ct}$ and evaluate $\mathsf{ct}' \leftarrow \mathsf{FHE.Eval}(\mathsf{pk}, U_x, \mathsf{ct})$.
  - Prove that ct' was honestly generated: run $\pi \leftarrow \mathsf{SNARK.Prove}^H((\mathsf{ct}, \mathsf{ct}'), x)$ to get a SNARK π of the statement

    $$\exists x : \mathsf{ct}' = \mathsf{FHE.Eval}(\mathsf{pk}, U_x, \mathsf{ct}).$$

  - Output $O[P](\mathsf{ct}', \pi)$.

Figure 1: Succinct Ideal Obfuscation using Classical Oracles and FHE

4.7 Security

Theorem 11. *The construction in Figure 1 satisfies ideal obfuscation security as defined in Definition 13.*

Proof. We will start by formally describing the simulator $\mathcal{S}$.

Before describing our simulator, we need some terminology. We define a procedure $\mathsf{FindWitness}$. Very roughly, $\mathsf{FindWitness}$ is the witness extractor from Lemma 7.4 in [CMS19], with the crucial difference being that the database register is not measured by $\mathsf{FindWitness}$.

Now we are ready to define our simulator. On (superposition) query (ct', π) to $O[P]$, simulator $\mathcal{S}$ does the following:

- $\mathcal{S}$ outputs $\mathsf{pp} = \mathsf{FHE.Enc}(pk, 0)$.
- The joint state of the distinguisher $\mathcal{D}$ and H is given by a superposition over $|ct', \pi\rangle |y\rangle \otimes |D\rangle |\text{aux}\rangle$, where $|ct', \pi\rangle, |y\rangle$ are the query input and query output registers respectively, and $|D\rangle, |\text{aux}\rangle$ are the database D (recording the queries to H) and $\mathcal{D}$'s auxiliary registers respectively. $|D\rangle$ will update according to the Ctso procedure.
- On query (ct', π) to $O[P]$, $\mathcal{S}$ performs the following operation on the state $|ct'\rangle |\pi\rangle \otimes |D\rangle$:
  - Initilialize ancilla register $|0\rangle_b$. Run $\mathsf{SNARK.Verify}^H(\pi)$ and XOR the output in register b. If output is 0, output $\perp$.
  - Perform the map $|ct'\rangle |\pi\rangle \otimes |D\rangle \rightarrow |ct'\rangle |\pi\rangle \otimes |D\rangle |\mathsf{FindWitness}(D)\rangle_{\mathcal{W}}$.
- Evaluate P on the input from register $\mathcal{W}$, and XOR in the query output register.

FindWitness

- **Input**:
  - Database D.
- **Procedure:**
  - Run $x \leftarrow \mathbf{E}(0^\ell)$, where 0^ℓ denotes the all 0s proof, and $\ell = 2^d$ for some $d \in \mathbb{N}$ is the proof length. If x is a valid witness, output x, otherwise continue.
  - For each $\mathsf{rt} \in D$:
    - Run $T \leftarrow \text{Extract}(D, \mathsf{rt}, d)$.
    - Set Π_T equal to leaves(T) on all points where leaves$(T) \neq \perp$, and 0 otherwise.
    - Run $x \leftarrow \mathbf{E}(\Pi_T)$.
    - Output x.

Figure 2: FindWitness

- Uncompute the $\mathcal{W}$ register by evaluating $\mathsf{FindWitness}(D)$ and XOR-ing in the $\mathcal{W}$ register.

Hybrid 0: Sample $(O[P], \text{pp}) \leftarrow \mathsf{Obf}^H(1^\lambda, P)$.

Hybrid 1: Same as **Hybrid 0**, but the random oracle H is simulated as a compressed standard oracle for a database D.

Claim 14. $\Pr[\textbf{\textit{Hybrid 0}} = 1] = \Pr[\textbf{\textit{Hybrid 1}} = 1]$.

Proof. This follows directly from Lemma 4 in [Zha19]. □

Hybrid 2: Same as **Hybrid 1**, but we add an abort condition for $\mathsf{FindWitness}(D)$ outputting invalid witnesses. More precisely,

- On query (ct', π) to $O[P]$, the joint state of the distinguisher $\mathcal{D}$ and H is $|ct', \pi\rangle\,|y\rangle \otimes |D\rangle\,|\text{aux}\rangle$
  - Initilialize ancilla register $|0\rangle_b$. Run $\mathsf{SNARK.Verify}^H(\pi)$ and XOR the output in register b. If output is 0, output $\perp$.
  - Measure the database register D, and run $\mathsf{FindWitness}(D)$.
  - If $x \leftarrow \mathsf{FindWitness}(D)$ is not a valid witness, output $\perp$.
  - Otherwise, compute $y \leftarrow \mathsf{FHE.Dec}(sk, ct')$ and XOR into the query output register of $\mathcal{D}$.

Claim 15. $\Pr[\textbf{\textit{Hybrid 2}} = 1] - \Pr[\textbf{\textit{Hybrid 1}} = 1] \leq \textit{negl}(\lambda)$

Proof. Assume otherwise, then there must exist some query made by $\mathcal{D}$ with noticeable amplitude on $|ct, \pi\rangle \otimes |D\rangle$ such that $\mathsf{FindWitness}(D)$ does not output a valid witness, but $\mathsf{SNARK.Verify}^H(\pi) = 1$. Formally, define the following set B of inputs of the form

$$B = \{(ct', \pi, D, b) : x \leftarrow \mathsf{FindWitness}(D) \text{ is not valid but } b = 1\}$$

Let $|\psi\rangle$ be the joint state of $\mathcal{D}$ and H right before measuring the database D. Then our assumption implies that

$$||\Pi[B]\,|\psi\rangle\,||^2 = \varepsilon(\lambda)$$

for some non negligible function ε. For a database D, define $B[D] := \{(., D) : (., D) \in B\}$. We can conclude that there exists a database D' such that

$$||\Pi[B[D']] \, |\psi\rangle \, ||^2 = \varepsilon'(\lambda),$$

where $\varepsilon'(.)$ is also a non negligible function. Now define the state

$$|\psi'\rangle = \langle x| \, \mathsf{FindWitness}(D') \, |\psi\rangle \, , |\psi^*\rangle := \frac{|\psi'\rangle}{||\, |\psi'\rangle \, ||}$$

So $|\psi^*\rangle$ is the result of running $\mathsf{FindWitness}(D')$ coherently, and post selecting on the witness being invalid. From the gentle measurement lemma, we can conclude that, the probability that $\mathsf{FindWitness}(D')$ finds an invalid witness on $\Pi[B[D']] \, |\psi^*\rangle$ is non negligible as well, but this is a contradiction to Lemma 6 . □

Hybrid 3: Same as **Hybrid 2**, but queries to $O[P]$ are answered as follows:

- On query (ct', π) to $O[P]$, $\mathcal{S}$ performs the following operation on the state $|ct'\rangle \, |\pi\rangle \otimes |D\rangle$:
  - Initilialize ancilla register $|0\rangle_b$. Run $\mathsf{SNARK.Verify}^H(\pi)$ and XOR the output in register b. If output is 0, output $\perp$.
  - Perform the map $|ct'\rangle \, |\pi\rangle \otimes |D\rangle \to |ct'\rangle \, |\pi\rangle \otimes |D\rangle \, |\mathsf{FindWitness}(D)\rangle_{\mathcal{W}}$.
  - If $x \leftarrow \mathsf{FindWitness}(D)$ is a not a valid witness, that is, if $P(x) \neq \mathsf{FHE.Dec}(sk, ct')$, output $\perp$.
  - Otherwise, evaluate P on the input from register $\mathcal{W}$, and XOR in the query output register.
- Uncompute the $\mathcal{W}$ register by evaluating $\mathsf{FindWitness}(D)$ and XOR-ing in the $\mathcal{W}$ register.

Claim 16. $\Pr[\textbf{\textit{Hybrid 2}} = 1] = \Pr[\textbf{\textit{Hybrid 3}} = 1]$

Proof. First observe that $\mathsf{FindWitness}$ is essentially the extractor $\mathcal{E}*$ for the Micali SNARK, with the only (but crucial) difference being that the database D is not measured. Since $\mathsf{Extract}$ aborts if there exists a $u : D(x) = D(x') = u, x \neq x'$, by construction, this implies that, for a fixed database D, $\mathsf{FindWitness}(D)$ has a unique solution. Thus, when all invalid solutions are aborted, it follows from the correctness of $\mathcal{E}*$ that queries are answered identically in **Hybrid 2** and **Hybrid 3**. □

Hybrid 4: Same as **Hybrid 3**, but the abort condition for $\mathsf{FindWitness}$ outputting invalid witnesses is removed, and queries to $O[P]$ are answered by $\mathcal{S}$ as follows:

- On query (ct', π) to $O[P]$, $\mathcal{S}$ performs the following operation on the state $|ct'\rangle \, |\pi\rangle \otimes |D\rangle$:
  - Initilialize ancilla register $|0\rangle_b$. Run $\mathsf{SNARK.Verify}^H(\pi)$ and XOR the output in register b. If output is 0, output $\perp$.
  - Perform the map $|ct'\rangle \, |\pi\rangle \otimes |D\rangle \to |ct'\rangle \, |\pi\rangle \otimes |D\rangle \, |\mathsf{FindWitness}(D)\rangle_{\mathcal{W}}$.
  - Evaluate P on the input from register $\mathcal{W}$, and XOR in the query output register.
- Uncompute the $\mathcal{W}$ register by evaluating $\mathsf{FindWitness}(D)$ and XOR-ing in the $\mathcal{W}$ register.

Claim 17. $\left| \Pr[\textbf{\textit{Hybrid 3}} = 1] - \Pr[\textbf{\textit{Hybrid 4}} = 1] \right| \leq \mathit{negl}(\lambda)$

Proof. The proof of this claim is identical to Claim 15. □

Hybrid 5: This is the ideal world.

- Queries to $O[P]$ are answered as in the description of $\mathcal{S}$ as follows:

- Initilialize ancilla register $|0\rangle_b$. Run $\mathsf{SNARK.Verify}^H(\pi)$ and XOR the output in register b. If output is 0, output $\perp$.
- Perform the map $|ct'\rangle|\pi\rangle \otimes |D\rangle \to |ct'\rangle|\pi\rangle \otimes |D\rangle |\mathsf{FindWitness}(D)\rangle_{\mathcal{W}}$.
- Evaluate P on the input from register $\mathcal{W}$, and XOR in the query output register.

- Uncompute the $\mathcal{W}$ register by evaluating $\mathsf{FindWitness}(D)$ and XOR-ing in the $\mathcal{W}$ register.
- Output $\mathrm{pp} \leftarrow \mathsf{FHE.Enc}(pk, 0^{|P|})$.

Claim 18. $\left|\Pr[\textbf{\textit{Hybrid 5}} = 1] - \Pr[\textbf{\textit{Hybrid 4}} = 1]\right| \leq negl(\lambda)$

Proof. This follows directly from the security of the QFHE scheme. □

□

5 Publicly-verifiable QFHE

We define publicly-verifiable QFHE in the oracle model. We consider both leveled and unleveled QFHE.

5.1 Definition

Definition 16 (Publicly-verifiable QFHE)**.** *A leveled publicly-verifiable quantum fully-homomorphic encryption scheme for pseudo-deterministic circuits consists of the following algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Ver}, \mathsf{Dec})$*.*

- $\mathsf{Gen}(1^\lambda, D, N, S) \to (\mathsf{pk}, \mathsf{sk}, \mathsf{PP})$*: The key generation algorithm is a PPT algorithm that takes as input the security parameter* 1^λ*, an upper bound on circuit depth* D*, an upper bound on input length* N*, and an upper bound on circuit description size* S*, and returns a public key* pk*, a secret key* sk*, and the description of a classical deterministic circuit* PP*.*
- $\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$*: The encryption algorithm is a PPT algorithm that takes as input the public key* pk *and a classical plaintext* x*, and outputs a ciphertext* ct*.*
- $\mathsf{Eval}^{\mathsf{PP}}(\mathsf{ct}, Q) \to (\widetilde{\mathsf{ct}}, \pi)$*: The evaluation algorithm is a QPT algorithm that has oracle access to* PP*, takes as input a ciphertext* ct*, a quantum circuit* Q*, and outputs an evaluated ciphertext* $\widetilde{\mathsf{ct}}$ *and proof* π*.*
- $\mathsf{Ver}^{\mathsf{PP}}(\mathsf{ct}, P_Q, \widetilde{\mathsf{ct}}, \pi) \to \{\top, \perp\}$*: The verification algorithm is a PPT algorithm that has oracle access to* PP*, takes an input ciphertext* ct*, a quantum program* P_Q *(i.e. a potentially succinct description of the quantum circuit* Q*), an output ciphertext* $\widetilde{\mathsf{ct}}$ *and a proof* π*, and accepts* $(\top)$ *or rejects* $(\perp)$*.*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to x$*: The decryption algorithm is a PPT algorithm that takes as input the secret key* sk *and a classical ciphertext* ct*, and outputs plaintext* x*.*

These algorithms should satisfy the following properties. Recall that pseudo-deterministic circuits Q *are defined to have a single bit of output, without loss of generality.*

- ***Correctness.*** *For any polynomials* $D = D(\lambda), N = N(\lambda), S = S(\lambda)$*, any pseudo-deterministic program* P_Q *where* Q *has circuit depth at most* D *and* $|P_Q| \leq S$*, and any input* x *with* $|x| \leq N$*,*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}^{\mathsf{PP}}(\mathsf{ct}, P_Q, \widetilde{\mathsf{ct}}, \pi) = \top \ \wedge & (\mathsf{pk}, \mathsf{sk}, \mathsf{PP}) \leftarrow \mathsf{Gen}(1^\lambda, D, N, S) \\ \mathsf{Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}}) = Q(x) & \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x) \\ & (\widetilde{\mathsf{ct}}, \pi) \leftarrow \mathsf{Eval}^{\mathsf{PP}}(\mathsf{ct}, Q) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- ***Privacy.*** *For any QPT adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, polynomials $D = D(\lambda), N = N(\lambda), S = S(\lambda)$, and messages x_0, x_1,*

$$\left|\Pr\left[\mathcal{A}^{\mathsf{PP}}(\mathsf{pk},\mathsf{ct}) = 1 : \begin{array}{r}(\mathsf{pk},\mathsf{sk},\mathsf{PP}) \leftarrow \mathsf{Gen}(1^\lambda, D, N, S)\\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x_0)\end{array}\right]\right.$$
$$\left. - \Pr\left[\mathcal{A}^{\mathsf{PP}}(\mathsf{pk},\mathsf{ct}) = 1 : \begin{array}{r}(\mathsf{pk},\mathsf{sk},\mathsf{PP}) \leftarrow \mathsf{Gen}(1^\lambda, D, S)\\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x_1)\end{array}\right]\right| = \mathsf{negl}(\lambda).$$

- ***Soundness.*** *For any QPT adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, polynomials $D = D(\lambda), N = N(\lambda), S = S(\lambda)$, pseudo-deterministic program P_Q where Q has circuit depth at most D and $|P_Q| \leq S$, and input x with $|x| \leq N$,*

$$\Pr\left[\begin{array}{l}\mathsf{Ver}^{\mathsf{PP}}(\mathsf{ct}, P_Q, \widetilde{\mathsf{ct}}, \pi) = \top \ \wedge\\ \mathsf{Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}}) \neq Q(x)\end{array} : \begin{array}{r}(\mathsf{pk},\mathsf{sk},\mathsf{PP}) \leftarrow \mathsf{Gen}(1^\lambda, D, N, S)\\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x)\\ (\widetilde{\mathsf{ct}}, \pi) \leftarrow \mathcal{A}^{\mathsf{PP}}(\mathsf{ct})\end{array}\right] = \mathsf{negl}(\lambda).$$

- ***Compactness.*** *There exists a fixed polynomial p such that the following holds. Fix any polynomials $D = D(\lambda), N = N(\lambda), S = S(\lambda)$, any pseudo-deterministic progam P_Q where Q has circuit depth at most D and $|P_Q| \leq S$, and any x with $|x| \leq N$, and let $(\mathsf{pk},\mathsf{sk},\mathsf{PP}) \leftarrow \mathsf{Gen}(1^\lambda, D, N, S), \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x)$, and $(\widetilde{\mathsf{ct}}, \pi) \leftarrow \mathsf{Eval}^{\mathsf{PP}}(\mathsf{ct}, Q)$. Then it holds that*
  - *$|\mathsf{pk}|, |\mathsf{sk}|, |\mathsf{PP}|, |\mathsf{ct}|, |\widetilde{\mathsf{ct}}|, |\pi| \leq p(\lambda, D, N, S)$, and*
  - *the run-times of $\mathsf{Gen}(1^\lambda, D, N, S), \mathsf{Enc}(\mathsf{pk}, x)$, $\mathsf{Ver}^{\mathsf{PP}}(\mathsf{ct}, P_Q, \widetilde{\mathsf{ct}}, \pi)$, and $\mathsf{Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}})$ are bounded by $p(\lambda, D, N, S)$.*

An *unleveled* publicly-verifiable QFHE scheme is defined exactly as above, except that Gen no longer takes a depth parameter D and all properties are defined without the parameter D.

5.2 Upgradable privately-verifiable QFHE

We will construct publicly-verifiable QFHE by *upgrading* a privately-verifiable scheme that satisfies certain properties. These properties are encapsulated in the following definition.

Definition 17 (Upgradable privately-verifiable QFHE)**.** *A leveled privately-verifiable QFHE scheme that is amenable to being upgraded to public-verifiability via our compiler has the following syntax. Let $H : \{0,1\}^* \to \{0,1\}$ be a random oracle. Let $\ell(\lambda, |Q|)$ be the "state size" parameter and $p(\lambda, N, D)$ be the "compactness" parameter.*

- $\mathsf{Gen}(1^\lambda, D) \to (\mathsf{pk}, \mathsf{sk})$*: The PPT key generation algorithm takes as input the security parameter 1^λ and an upper bound on circuit depth D and outputs a public key pk and secret key sk.*
- $\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$*: The PPT encryption algorithm takes as input a public key sk and an input x and outputs a ciphertext ct.*
- $\mathsf{VerGen}(1^\lambda, \mathsf{ct}, Q) \to \mathsf{pp}, \mathsf{sp}$*: The PPT verification parameter generation algorithm takes as input the security parameter 1^λ, a ciphertext ct, and quantum circuit Q, and outputs public parameters pp and secret parameters sp.*
- $\mathsf{Eval}^H(\mathsf{pp}, \mathsf{ct}, Q) \to \pi$*: The QPT evaluation algorithm operates as follows.*
  - *Let $\ell = \ell(\lambda, |Q|)$, initialize an ℓ-qubit state $|\psi_{\mathsf{ct},Q}\rangle$ on register $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_\ell)$, and initialize two fresh registers $\mathcal{Y}, \mathcal{Z}$.*
  - *Apply a measurement M_{pp} on $(\mathcal{M}, \mathcal{Z}, \mathcal{Y})$ that is classically-controlled on $\mathcal{M}$ to obtain a string y.*

- *Compute $T = H(y)$, where T is parsed as a subset $T \subseteq [\ell]$.*[13]
- *Apply a measurement $M_{\mathsf{pp},y,T}$ on $(\mathcal{M}, \mathcal{Z})$ that is classically-controlled on $\mathcal{M}$ to obtain a string z.*
- *For all $i \in [\ell]$, measure $\mathcal{M}_i$ in the standard basis if $i = 0$ or in the Hadamard basis if $i = 1$ to obtain a string $m \in \{0,1\}^\ell$.*
- *Return $\pi = (m, y, z)$.*

- $\mathsf{Ver}^H(\mathsf{sp}, \mathsf{ct}, Q, \pi) \to \{\bot, \widetilde{\mathsf{ct}}\}$*: The PPT verification algorithm takes as input secret parameters* sp*, a ciphertext* ct*, a quantum circuit Q, and a proof π, and outputs either reject ($\bot$) or a ciphertext* $\widetilde{\mathsf{ct}}$*.*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to x$*: The PPT decryption algorithm takes as input the secret key* sk *and a ciphertext* ct *and outputs a plaintext x.*

It should satisfy the following properties. We leave probabilities taken over the sampling of the random oracle H implicit.

- ***Privacy.*** *For any QPT adversary $\{\mathcal{A}_\lambda\}_\lambda$, depth $D = D(\lambda)$, and messages x_0, x_1,*

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

- ***Completeness.*** *For any $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Gen}(1^\lambda, D)$, $\mathsf{ct} \in \mathsf{Enc}(\mathsf{pk}, x)$, and Q,*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{Ver}^H(\mathsf{sp}, \mathsf{ct}, Q, \pi)) = Q(x) : \begin{array}{r}\mathsf{pp}, \mathsf{sp} \leftarrow \mathsf{VerGen}(\mathsf{ct}, Q)\\ \pi \leftarrow \mathsf{Eval}^H(\mathsf{pp}, \mathsf{ct}, Q)\end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- ***Soundness.*** *For any $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Gen}(1^\lambda, D)$, $\mathsf{ct} \in \mathsf{Enc}(\mathsf{pk}, x)$, Q, and QPT adversary $\{\mathcal{A}_\lambda\}_\lambda$,*

$$\Pr\left[\widetilde{\mathsf{ct}} \neq \bot \wedge \mathsf{Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}}) \neq Q(x) : \begin{array}{r}\mathsf{pp}, \mathsf{sp} \leftarrow \mathsf{VerGen}(\mathsf{ct}, Q)\\ \pi \leftarrow \mathcal{A}^H(\mathsf{pp})\\ \widetilde{\mathsf{ct}} \leftarrow \mathsf{Ver}^H(\mathsf{sp}, \mathsf{ct}, Q, \pi)\end{array}\right] = \mathsf{negl}(\lambda).$$

- ***Incompatible standard-basis measurements.*** *First, we introduce some notation.*
  - *Given a subset $I \subset [\ell]$, define $\overline{I} := [\ell] \setminus I$.*
  - *Given any $m \in \{0,1\}^\ell$ and a subset $I \subseteq [\ell]$, we define $m_{I \to *} \in \{0,1,*\}^\ell$ to be equal to m_i at indices $i \in \overline{I}$ and equal to $*$ at indices $i \in I$.*
  - *Each choice of $(\cdot, \mathsf{sp}) \in \mathsf{VerGen}(\mathsf{ct}, Q)$ defines a set of "standard basis positions" $S \subseteq [\ell]$, which we denote by $S = S[\mathsf{sp}]$ (not to be confused with the positions $T \subseteq [\ell]$ that are* measured *in the standard basis).*
  - *We say that two sets of strings $M_0, M_1 \in \{0,1,*\}^\ell$ are* incompatible *if for any $m_0 \in M_0, m_1 \in M_1$, there exists an $i \in [\ell]$ such that $m_{0,i}, m_{1,i} \neq *$ and $m_{0,i} \neq m_{1,i}$. Each choice of $(\cdot, \mathsf{sp}) \in \mathsf{VerGen}(\mathsf{ct}, Q)$ defines two incompatible sets of strings $M_0, M_1 \in \{0,1,*\}^\ell$.*

Now, we need the following properties.

[13] In a slight abuse of notation, we define an ℓ-bit output $H(y) := H(1, y) \| \ldots \| H(\ell, y)$.

– *For any* $(\mathsf{pk},\mathsf{sk}) \in \mathsf{Gen}(1^\lambda, D)$, $\mathsf{ct} \in \mathsf{Enc}(\mathsf{pk}, x)$, Q, $(\mathsf{pp},\mathsf{sp}) \in \mathsf{VerGen}(\mathsf{ct}, Q)$, *and proof* $\pi = (m, y, z)$, *it holds that*

$$\mathsf{Ver}^H(\mathsf{sp},\mathsf{ct},Q,(m,y,z)) = \mathsf{Ver}^H(\mathsf{sp},\mathsf{ct},Q,(m_{S\cap\overline{T}\to *},y,z)).$$

That is, Ver *ignores all bits of* m *that are measured in the Hadamard basis, but correspond to "standard basis positions".*

– *For any* $(\mathsf{pk},\mathsf{sk}) \in \mathsf{Gen}(1^\lambda, D)$, $\mathsf{ct} \in \mathsf{Enc}(\mathsf{pk}, x)$, Q, *and QPT adversary* $\{\mathcal{A}_\lambda\}_\lambda$,

$$\Pr\left[\widetilde{\mathsf{ct}} \neq \bot \wedge m_{\overline{S}\cup\overline{T}\to *} \notin M_0 : \begin{array}{r} \mathsf{pp},\mathsf{sp} \leftarrow \mathsf{VerGen}(\mathsf{ct}, Q) \\ \pi = (m,y,z) \leftarrow \mathcal{A}^H(\mathsf{pp}) \\ T := H(y) \\ \widetilde{\mathsf{ct}} \leftarrow \mathsf{Ver}^H(\mathsf{sp},\mathsf{ct},Q,\pi) \end{array}\right] = \mathsf{negl}(\lambda).$$

That is, it is hard to find an accepting proof π *such that* $m_{\overline{S}\cap\overline{T}\to *}$ *(where only the standard basis positions that are measured in the standard basis remain) is in* M_0.

– *For any* $(\mathsf{pk},\mathsf{sk}) \in \mathsf{Gen}(1^\lambda, D)$, $\mathsf{ct} \in \mathsf{Enc}(\mathsf{pk}, x)$, Q, *and QPT adversary* $\{\mathcal{A}_\lambda\}_\lambda$,

$$\Pr\left[\widetilde{\mathsf{ct}} \neq \bot \wedge \mathsf{Dec}(\mathsf{sk},\widetilde{\mathsf{ct}}) \neq Q(x) \wedge m_{\overline{S}\cup\overline{T}\to *} \notin M_1 : \begin{array}{r} \mathsf{pp},\mathsf{sp} \leftarrow \mathsf{VerGen}(\mathsf{ct}, Q) \\ \pi = (m,y,z) \leftarrow \mathcal{A}^H(\mathsf{pp},\mathsf{sp}) \\ T := H(y) \\ \widetilde{\mathsf{ct}} \leftarrow \mathsf{Ver}^H(\mathsf{sp},\mathsf{ct},Q,\pi) \end{array}\right] = \mathsf{negl}(\lambda).$$

That is, it is hard to find an accepting but faulty proof π *such that* $m_{\overline{S}\cap\overline{T}\to *}$ *(where only the standard basis positions that are measured in the standard basis remain) is in* M_1, even given the secret parameters sp.

- ***Parallelizable*** VerGen**.** *There exists a fixed polynomial* $k(\lambda)$ *and a deterministic classical algorithm* $\mathsf{ParVerGen}(\mathsf{ct}, Q, \mathsf{sp}, i) \to \mathsf{pp}_i$, *where* $|\mathsf{pp}_i| = k(\lambda)$, *such that for any* ct, Q, *the operation* $\mathsf{VerGen}(1^\lambda, \mathsf{ct}, Q)$ *is equivalent to:*
  – *Sample* $\mathsf{sp} \leftarrow \{0,1\}^\lambda$
  – *Run* $\mathsf{pp}_i := \mathsf{ParVerGen}(\mathsf{ct}, Q, \mathsf{sp}, i)$ *for* $i \in [\ell]$, *and set* $\mathsf{pp} := (\mathsf{pp}_1, \ldots, \mathsf{pp}_\ell)$.
- ***Compactness.*** *Fix any polynomials* $N = N(\lambda), D = D(\lambda)$, *any input* x *such that* $|x| \leq N$, *and any pseudo-deterministic quantum circuit* Q *with depth at most* D, *and let* $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda, D), \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x), (\mathsf{pp},\mathsf{sp}) \leftarrow \mathsf{VerGen}(\mathsf{ct}, Q), \pi \leftarrow \mathsf{Eval}^H(\mathsf{pp},\mathsf{ct},Q), \widetilde{\mathsf{ct}} \leftarrow \mathsf{Ver}^H(\mathsf{sp},\mathsf{ct},Q,\pi)$, *and* $Q(x) \leftarrow \mathsf{Dec}(\mathsf{sk},\mathsf{ct})$. *Then it holds that*
  – $|\mathsf{pk}|, |\mathsf{sk}|, |\mathsf{ct}|, |\widetilde{\mathsf{ct}}| \leq p(\lambda, N, D)$, *and*
  – *The run-times of* $\mathsf{Gen}(1^\lambda, D), \mathsf{Enc}(\mathsf{pk}, x), \mathsf{Dec}(\mathsf{sk},\mathsf{ct})$ *are bounded by* $p(\lambda, N, D)$.

An *unleveled* upgradable privately-verifiable QFHE scheme is defined exactly as above, except that Gen no longer takes the depth parameter D and all properties are defined without the parameter D.

Theorem 12 ([BKNY23], Section 5.3)**.** *Assuming LWE, there exists a leveled upgradable privately-verifiable QFHE in the QROM. Assuming LWE plus an appropriate circular-security assumption [Bra18], there exists unleveled upgradable privately-verifiable QFHE in the QROM.*

5.3 Construction

Ingredients:

- PRF $F_k : \{0,1\}^* \to \{0,1\}^\lambda$.

- OSS ($\mathsf{OSS.Setup}, \mathsf{OSS.Gen}, \mathsf{OSS.Sign}, \mathsf{OSS.Ver}$) in the classical oracle model (Definition 3).
- PFC ($\mathsf{PFC.Gen}, \mathsf{PFC.Com}, \mathsf{PFC.OpenZ}, \mathsf{PFC.OpenZ}, \mathsf{PFC.DecZ}, \mathsf{PFC.DecX}$) in the classical oracle model (see Definition 5), where single-qubit commitments are size $t(\lambda)$.
- Upgradable privately-verifiable QFHE ($\mathsf{Priv.Gen}, \mathsf{Priv.Enc}, \mathsf{Priv.VerGen}, \mathsf{Priv.Eval}^H, \mathsf{Priv.Ver}^H, \mathsf{Priv.Dec}$), in the QROM with state size parameter $\ell(\lambda, |Q|)$ and compactness parameter $p(\lambda, N, D)$ (see Definition 17).
- Post-quantum succinct ideal obfuscation ($\mathsf{SObf.Obf}, \mathsf{SObf.Eval}$) in the classical oracle model (see Definition 13).

Publicly-verifiable QFHE

- $\mathsf{Gen}(1^\lambda, N, D, S)$:
  - Sample $\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{Priv.Gen}(1^\lambda)$.
  - Sample PRF keys $k_{\mathsf{PFC}}, k_{\mathsf{Priv}}, k_H \leftarrow \{0,1\}^\lambda$.
  - Sample $\mathsf{O} \leftarrow \mathsf{OSS.Setup}(1^\lambda)$.
  - Define $\mathsf{CK}(\mathsf{pk}_{\mathsf{OSS}}, i, v)$ as follows.
    - Compute $(\mathsf{CK}_{\mathsf{pk},i}, \mathsf{dk}_{\mathsf{pk},i}) \coloneqq \mathsf{PFC.Gen}(1^\lambda, F_{k_{\mathsf{PFC}}}(\mathsf{pk}_{\mathsf{OSS}}, i))$.
    - Output $\mathsf{CK}_{\mathsf{pk},i}(v)$.
  - Define $\mathsf{PrivGen}(\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, i)$ as follows.
    - If $\mathsf{OSS.Ver}^{\mathsf{O}}(\mathsf{pk}_{\mathsf{OSS}}, (\mathsf{ct}, Q, c), \sigma) = \top$, then continue, and otherwise return $\bot$.
    - Define $\mathsf{sp} \coloneqq F_{k_{\mathsf{Priv}}}(\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma)$.
    - Output $\mathsf{pp}_i \coloneqq \mathsf{Priv.ParVerGen}(\mathsf{ct}, Q, \mathsf{sp}, i)$.
  - Let $\mathsf{H} \coloneqq F_{k_H}$.
  - Define $\mathsf{PrivVer}(\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$ as follows.
    - If $\mathsf{OSS.Ver}^{\mathsf{O}}(\mathsf{pk}_{\mathsf{OSS}}, (\mathsf{ct}, Q, c), \sigma) = \top$, then set $\mathsf{sp} \coloneqq F_{k_{\mathsf{Priv}}}(\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma)$, $S \coloneqq S[\mathsf{sp}]$, and otherwise return $\bot$.
    - Parse $u = (u_1, \ldots, u_\ell)$, $c = (c_1, \ldots, c_\ell)$, and define $T \coloneqq \mathsf{H}(y)$.
    - For each $i \in [\ell]$, compute $(\mathsf{CK}_{\mathsf{pk},i}, \mathsf{dk}_{\mathsf{pk},i}) \coloneqq \mathsf{PFC.Gen}(1^\lambda, F_{k_{\mathsf{PFC}}}(\mathsf{pk}_{\mathsf{OSS}}, i))$.
    - For each $i \in T$, compute $m_i \coloneqq \mathsf{PFC.DecZ}(\mathsf{dk}_{\mathsf{pk},i}, c_i, u_i)$, and return $\bot$ if $m_i = \bot$.
    - For each $i \in \overline{T} \setminus S$, compute $m_i \coloneqq \mathsf{PFC.DecX}(\mathsf{dk}_{\mathsf{pk},i}, c_i, u_i)$, and return $\bot$ if $m_i = \bot$.
    - For each $i \in \overline{T} \cap S$, set $m_i = *$.
    - Let $m = (m_1, \ldots, m_\ell)$, and output $\mathsf{Priv.Ver}^{\mathsf{H}}(\mathsf{sp}, \mathsf{ct}, Q, (m, y, z))$.
  - Output $\mathsf{pk}, \mathsf{sk}, \mathsf{PP} \coloneqq \mathsf{SObf.Obf}(\mathsf{O}, \mathsf{CK}, \mathsf{PrivGen}, \mathsf{H}, \mathsf{PrivVer})$
- $\mathsf{Enc}(\mathsf{pk}, x)$: Output $\mathsf{ct} \leftarrow \mathsf{Priv.Enc}(\mathsf{pk}, x)$.
- $\mathsf{Eval}^{\mathsf{PP}}(\mathsf{ct}, Q)$
  - Let $\ell = \ell(\lambda, |Q|)$ and sample $(\mathsf{pk}_{\mathsf{OSS}}, |\mathsf{sk}_{\mathsf{OSS}}\rangle) \leftarrow \mathsf{OSS.Gen}^{\mathsf{O}}(|\mathsf{ct}| + |Q| + \ell \cdot t(\lambda))$.
  - Initialize $|\psi_{\mathsf{ct},Q}\rangle$ on register $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_\ell)$.
  - For each $i \in [\ell]$ apply $\mathsf{PFC.Com}^{\mathsf{CK}(\mathsf{pk}_{\mathsf{OSS}}, i, \cdot)} : \mathcal{M}_i \to \mathcal{M}_i, \mathcal{U}_i, c_i$, and set $c \coloneqq (c_1, \ldots, c_\ell)$.
  - Compute $\sigma \leftarrow \mathsf{OSS.Sign}^{\mathsf{O}}(|\mathsf{sk}_{\mathsf{OSS}}\rangle, (\mathsf{ct}, Q, c))$ and, for $i \in [\ell]$, compute $\mathsf{pp}_i \leftarrow \mathsf{PrivGen}(\mathsf{pk}_{\mathsf{OSS}}, \mathsf{ct}, Q, c, \sigma, i)$. Set $\mathsf{pp} \coloneqq (\mathsf{pp}_1, \ldots, \mathsf{pp}_\ell)$.
  - Initialize fresh registers $\mathcal{Y}, \mathcal{Z}$, apply M_{pp} on $(\mathcal{M}, \mathcal{Y}, \mathcal{Z})$ to obtain y, compute $T = \mathsf{H}(y)$, and apply $M_{\mathsf{pp},y,T}$ on $(\mathcal{M}, \mathcal{Z})$ to obtain z.
  - For each $i \in T$, compute $u_i \leftarrow \mathsf{PFC.OpenZ}(\mathcal{M}_i, \mathcal{U}_i)$ and for each $i \in \overline{T}$, compute $u_i \leftarrow \mathsf{PFC.OpenX}(\mathcal{M}_i, \mathcal{U}_i)$. Set $u \coloneqq (u_1, \ldots, u_\ell)$.
  - Output $\widetilde{\mathsf{ct}} \coloneqq \mathsf{PrivVer}(\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$ and $\pi \coloneqq (\mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$.
- $\mathsf{Ver}^{\mathsf{PP}}(\mathsf{ct}, Q, \widetilde{\mathsf{ct}}, \pi)$: Output $\top$ iff $\mathsf{PrivVer}(\mathsf{ct}, Q, \pi) = \widetilde{\mathsf{ct}}$.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Output $\mathsf{Priv.Dec}(\mathsf{sk}, \mathsf{ct})$.

Figure 3: Our construction of publicly-verifiable QFHE in the classical oracle model.

5.4 Proofs

Theorem 13. *The QFHE protocol given in Figure 3 satisfies correctness (see Definition 16).*

Proof. This is a straightforward adaptation of the completeness part of [BKNY23, Theorem 5.12], where we use the correctness of OSS (along with the other primitives) rather than the signature token. □

Theorem 14. *The QFHE protocol given in Figure 3 satisfies privacy (see Definition 16).*

Proof. This follows directly from the privacy of the upgradable privately-verifiable QFHE. □

Functionalities used in the proof of Theorem 15

- $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s](\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$
  - Same as $\mathsf{PrivVer}$ except that it sets $\mathsf{sp} = s$ if $\mathsf{pk}_{\mathsf{OSS}} = \mathsf{pk}^*_{\mathsf{OSS}}$.
- $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)](\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$
  - Same as $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s]$ except output $\bot$ if $\mathsf{pk}_{\mathsf{OSS}} = \mathsf{pk}^*_{\mathsf{OSS}} \wedge (\mathsf{ct}, Q, c, \sigma) \neq (\mathsf{ct}^*, Q^*, c^*, \sigma^*)$.
- $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*), 0](\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$
  - Same as $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)]$ except output $\bot$ if $\mathsf{pk}_{\mathsf{OSS}} = \mathsf{pk}^*_{\mathsf{OSS}} \wedge m_{\overline{S \cup T}} \notin M_0$, where S, M_0 are defined based on $\mathsf{sp} = s$ as in Definition 17.
- $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*), 1](\mathsf{ct}, Q, \mathsf{pk}_{\mathsf{OSS}}, c, \sigma, u, y, z)$
  - Same as $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)]$ except output $\bot$ if $\mathsf{pk}_{\mathsf{OSS}} = \mathsf{pk}^*_{\mathsf{OSS}} \wedge m_{\overline{S \cup T}} \notin M_1$, where S, M_1 are defined based on $\mathsf{sp} = s$ as in Definition 17.
- $V(a, s, \mathsf{aux})$:
  - Parse $a := (\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*)$ and $\mathsf{aux} := (u^*, y^*, z^*)$.
  - Compute $\widetilde{\mathsf{ct}} := \mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s](\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*, u^*, y^*, z^*)$.
  - Output 1 iff $\widetilde{\mathsf{ct}} \neq \bot$ and $\mathsf{Priv.Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}}) \neq Q^*(x)$.
- $V[1](a, s, \mathsf{aux})$:
  - Parse $a := (\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*)$ and $\mathsf{aux} := (u^*, y^*, z^*)$.
  - Compute $\widetilde{\mathsf{ct}} := \mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*), 1](\mathsf{ct}^*, Q^*, (\mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*, u^*, y^*, z^*))$.
  - Output 1 iff $\widetilde{\mathsf{ct}} \neq \bot$ and $\mathsf{Priv.Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}}) \neq Q^*(x)$.

Figure 4

Theorem 15. *The QFHE protocol given in Figure 3 satisfies soundness (see Definition 16).*

Proof. We follow the soundness part of the proof of [BKNY23, Theorem 5.12], but with a couple of tweaks.

Suppose there exists a program P_{Q^*}, input x, and adversary $\mathcal{A}_1^{\mathsf{PP}}$ that violates the soundness of Definition 16, where we have dropped the indexing by λ for convenience. Our first step will be to appeal to the security of the succinct obfuscation (Definition 13) to replace $\mathcal{A}_1$ with an adversary $\mathcal{A}_2^{\mathsf{O},\mathsf{CK},\mathsf{PrivGen},\mathsf{H},\mathsf{PrivVer}}$ that has direct access to the large-input functionalities. This only has a negligble affect on the output of $\mathcal{A}_1$.

Next, we replace the PRFs $F_{k_{\mathsf{PFC}}}, F_{k_{\mathsf{Priv}}}, F_{k_H}$ with random oracles $\mathsf{H}_{\mathsf{PFC}}, \mathsf{H}_{\mathsf{Priv}}, \mathsf{H}$. Since $\mathcal{A}_2$ only has polynomially-bounded oracle access to these functionalities, this has a negligible affect on the output of $\mathcal{A}_2$ [Zha12]. $\mathcal{A}_2$ can then be used to define an oracle algorithm $\mathcal{A}_3^{\mathsf{H}_{\mathsf{Priv}}}$ that operates as follows.

- Sample $(\mathsf{pk}, \mathsf{sk}, \mathsf{O}, \mathsf{CK}, \mathsf{PrivGen}, \mathsf{H}, \mathsf{PrivVer})$ as in $\mathsf{Gen}(1^\lambda, N, D, S)$, except that $\mathsf{H}_{\mathsf{PFC}}$ and H are sampled as random oracles.
- Sample $\mathsf{ct}^* \leftarrow \mathsf{Priv.Enc}(\mathsf{pk}, x)$.
- Run $\mathcal{A}_2^{\mathsf{O},\mathsf{CK},\mathsf{PrivGen},\mathsf{H},\mathsf{PrivVer}}(\mathsf{ct}^*)$, forwarding calls to $\mathsf{H}_{\mathsf{Priv}}$ (which occur as part of calls to $\mathsf{PrivGen}$ and $\mathsf{PrivVer}$) to an external random oracle $\mathsf{H}_{\mathsf{Priv}}$.
- Measure $\mathcal{A}_2$'s output proof π^*, parse π^* as $(\mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*, m^*, y^*, z^*)$ and output $a := (\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*)$ and $\mathsf{aux} := (m^*, y^*, z^*)$.

Note that $\mathcal{A}_3$ makes $p = \mathrm{poly}(\lambda)$ total queries to $\mathsf{H}_{\mathsf{Priv}}$. Now, define V as in Figure 4. Then since $\mathcal{A}_1$ breaks soundness,

$$\Pr\left[V(a, \mathsf{H}_{\mathsf{Priv}}(a), \mathsf{aux}) = 1 : (a, \mathsf{aux}) \leftarrow \mathcal{A}_3^{\mathsf{H}_{\mathsf{Priv}}}\right] = \mathsf{non\text{-}negl}(\lambda).$$

Next, since $p = \mathrm{poly}(\lambda)$, by Theorem 3 there exists an algorithm $\mathcal{A}_4 := \mathsf{Sim}[\mathcal{A}_3]$ such that

$$\Pr\left[V((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), s, (m^*, y^*, z^*)) = 1 : \begin{array}{r}((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), \mathsf{state}) \leftarrow \mathcal{A}_4 \\ s \leftarrow \{0,1\}^\lambda \\ (m^*, y^*, z^*) \leftarrow \mathcal{A}_4(s, \mathsf{state})\end{array}\right] = \mathsf{non\text{-}negl}(\lambda).$$

Moreover, $\mathcal{A}_4$ operates as follows.

- Sample $\mathsf{H}_{\mathsf{Priv}}$ as a $2p$-wise independent function and $(i, d) \leftarrow (\{0, \ldots, p-1\} \times \{0,1\}) \cup \{(p, 0)\}$.
- Run $\mathcal{A}_3$ for i oracle queries, answering each query using the function $\mathsf{H}_{\mathsf{Priv}}$.
- When $\mathcal{A}_3$ is about to make its $(i+1)$'th oracle query, measure its query register in the standard basis to obtain $a := (\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*)$. In the special case that $(i, d) = (p, 0)$, just measure (part of) the final output register of $\mathcal{A}_3$ to obtain a.
- Receive s externally.
- If $d = 0$, answer $\mathcal{A}_3$'s $(i+1)$'th query with $\mathsf{H}_{\mathsf{Priv}}$. If $d = 1$, answer $\mathcal{A}_3$'s $(i+1)$'th query instead with the re-programmed oracle $\mathsf{H}_{\mathsf{Priv}}[(\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*) \to s]$.
- Run $\mathcal{A}_3$ until it has made all p queries to $\mathsf{H}_{\mathsf{Priv}}$. For queries $i+2$ through p, answer with the re-programmed oracle $\mathsf{H}_{\mathsf{Priv}}[(\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*) \to s]$.
- Measure $\mathcal{A}_3$'s output $\mathsf{aux} := (u^*, y^*, z^*)$.

Recall that $\mathcal{A}_3$ is internally running $\mathcal{A}_2$, who expects oracle access to $\mathsf{O}, \mathsf{CK}, \mathsf{PrivGen}, \mathsf{H}, \mathsf{PrivVer}$. These oracle queries will be answered by $\mathcal{A}_4$. Next, we define $\mathcal{A}_5$ to be the same as $\mathcal{A}_4$, except that after $(\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*)$ is measured by $\mathcal{A}_4$, $\mathcal{A}_2$'s queries to $\mathsf{PrivVer}$ are answered instead with the functionality $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*)]$ from Figure 4.

Claim 19.

$$\Pr\left[V((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), s, (m^*, y^*, z^*)) = 1 : \begin{array}{r}((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), \mathsf{state}) \leftarrow \mathcal{A}_5 \\ s \leftarrow \{0,1\}^\lambda \\ (m^*, y^*, z^*) \leftarrow \mathcal{A}_5(s, \mathsf{state})\end{array}\right] = \mathsf{non\text{-}negl}(\lambda).$$

Proof. We can condition on $\mathsf{OSS.Ver}^{\mathsf{O}}(\mathsf{pk}^*_{\mathsf{OSS}}, (\mathsf{ct}^*, Q^*, c^*), \sigma^*) = \top$, since otherwise V would output 0. Then, by the security of OSS (Definition 3), once $(\mathsf{pk}^*_{\mathsf{OSS}}, \mathsf{ct}^*, Q^*, c^*, \sigma^*)$ is measured, $\mathcal{A}_2$ cannot produce any query that has noticeable amplitude on any $(\mathsf{ct}, Q, c, \sigma)$ such that

$$(\mathsf{ct}, Q, c, \sigma) \neq (\mathsf{ct}^*, Q^*, c^*, \sigma^*) \quad \text{and} \quad \mathsf{OSS.Ver}(\mathsf{pk}^*_{\mathsf{OSS}}, (\mathsf{ct}, Q, c), \sigma) = \top.$$

But after $(\mathsf{pk}^*_{\mathsf{OSS}}, \mathsf{ct}^*, Q^*, c^*, \sigma^*)$ is measured and s is sampled, $\mathsf{PrivVer}$ and $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)]$ can only differ on $(\mathsf{ct}, Q, c, \sigma)$ such that

$$(\mathsf{ct}, Q, c, \sigma) \neq (\mathsf{ct}^*, Q^*, c^*, \sigma^*) \quad \text{and} \quad \mathsf{OSS.Ver}(\mathsf{pk}^*_{\mathsf{OSS}}, (\mathsf{ct}, Q, c), \sigma) = \top.$$

Thus, since $\mathcal{A}_2$ only has polynomially-many queries, changing the oracle in this way can only have a negligible affect on the final probability, which completes the proof. □

Next, we claim the following, where $V[1]$ is defined in Figure 4.

Claim 20.

$$\Pr\left[V[1]((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), s, (m^*, y^*, z^*)) = 1 : \begin{array}{r}((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), \mathsf{state}) \leftarrow \mathcal{A}_5 \\ s \leftarrow \{0,1\}^\lambda \\ (m^*, y^*, z^*) \leftarrow \mathcal{A}_5(s, \mathsf{state})\end{array}\right] = \mathsf{non\text{-}negl}(\lambda).$$

Proof. This is a straightforward adaptation of [BKNY23, Claim 5.14], reducing to the "Incompatible standard-basis measurements" property of Priv (Definition 17).

□

Finally, we will define a sequence of hybrids $\{\mathcal{H}_\iota\}_{\iota \in [0,p]}$ based on $\mathcal{A}_5$. Hybrid $\mathcal{H}_\iota$ is defined as follows.

- Run $((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), \mathsf{state}) \leftarrow \mathcal{A}_5$.
- Sample $s \leftarrow \{0,1\}^\lambda$.
- Run $(m^*, y^*, z^*) \leftarrow \mathcal{A}_5(s, \mathsf{state})$ with the following difference. Recall that at some point, $\mathcal{A}_5$ begins using the oracle $\mathsf{H}_{\mathsf{Priv}}[(\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*) \to s]$ while answering $\mathcal{A}_2$'s queries. For the first ι times that $\mathcal{A}_2$ queries $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)]$ after this point, respond using the oracle $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}} \to \bot]$ that outputs $\bot$ on every input that includes $\mathsf{pk}^*_{\mathsf{OSS}}$ (and behaves normally otherwise).
- Output $V[1]((\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*), s, (m^*, y^*, z^*))$.

Note that Claim 20 is stating exactly that $\Pr[\mathcal{H}_0 = 1] = \mathsf{non\text{-}negl}(\lambda)$. Next, we have the following claim.

Claim 21. $\Pr[\mathcal{H}_p = 1] = \mathsf{negl}(\lambda)$.

Proof. This is a straightforward adaptation of [BKNY23, Claim 5.15], reducing to the soundness of Priv (Definition 17).

□

Finally, we prove the following Claim 22. Since $p = \mathrm{poly}(\lambda)$, this contradicts Claim 20 and Claim 21, which completes the proof.

□

Claim 22. *For any* $\iota \in [p]$, $\Pr[\mathcal{H}_\iota = 1] \geq \Pr[\mathcal{H}_{\iota-1} = 1] - \mathsf{negl}(\lambda)$.

Proof. Throughout this proof, when we refer to "query ι" in some hybrid, we mean the ι'th query that $\mathcal{A}_2$ makes to $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)]$ after $\mathcal{A}_5$ has begun using the oracle $\mathsf{H}_{\mathsf{Priv}}[(\mathsf{ct}^*, Q^*, \mathsf{pk}^*_{\mathsf{OSS}}, c^*, \sigma^*) \to s]$ (if such a query exists).

Now, we introduce an intermediate hybrid $\mathcal{H}'_{\iota-1}$ which is the same as $\mathcal{H}_{\iota-1}$ except that query ι is answered with the functionality $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*), M_0]$ defined in Figure 4.

So, it suffices to show that

- $\Pr[\mathcal{H}'_{\iota-1} = 1] \geq \Pr[\mathcal{H}_{\iota-1} = 1] - \mathsf{negl}(\lambda)$, and
- $\Pr[\mathcal{H}_\iota = 1] \geq \Pr[\mathcal{H}'_{\iota-1} = 1] - \mathsf{negl}(\lambda)$.

We note that the only difference between the three hybrids is how query ι is answered:

- In $\mathcal{H}_{\iota-1}$, query ι is answered with $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*)]$.
- In $\mathcal{H}'_{\iota-1}$, query ι is answered with $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}}, s, (\mathsf{ct}^*, Q^*, c^*, \sigma^*), M_0]$.
- In $\mathcal{H}_{\iota}$, query ι is answered with $\mathsf{PrivVer}[\mathsf{pk}^*_{\mathsf{OSS}} \to \bot]$.

Now, the proof is completed by appealing to the following two claims. □

Claim 23. $\Pr[\mathcal{H}'_{\iota-1} = 1] \geq \Pr[\mathcal{H}_{\iota-1} = 1] - \mathsf{negl}(\lambda)$.

Proof. This is a straightforward adaptation of [BKNY23, Claim 5.17], reducing to the "Incompatible standard-basis measurements" property of Priv (Definition 17).

□

Claim 24. $\Pr[\mathcal{H}_{\iota} = 1] \geq \Pr[\mathcal{H}'_{\iota-1} = 1] - \mathsf{negl}(\lambda)$

Proof. This is a straightforward adaptation of [BKNY23, Claim 5.18], reducing to the string binding with public decodability of PFC (Definition 8).

□

6 (Succinct) Classical Obfuscation of Pseudo-deterministic Quantum Circuits

Definition 18 (Classical Virtual black-box obfuscation for Pseudodeterministic Quantum Circuits)**.** *A classical virtual black-box (VBB) obfuscator for a family of pseudo-deterministic quantum circuits is a pair of QPT algorithms* $(\mathsf{Obf}, \mathsf{Eval})$ *with the following syntax.*

- $\mathsf{Obf}(1^\lambda, P_Q) \to P_{\widetilde{Q}}$*:* Obf *is a classical p.p.t. algorithm that takes as input the security parameter* 1^λ *and the classical description* P_Q *of a quantum circuit* Q*, and outputs the classical description of an obfuscated circuit* $P_{\widetilde{Q}}$*.*
- $\mathsf{Eval}(P_{\widetilde{Q}}, x) \to b$*:* Eval *is a (quantum) polynomial-time algorithm that takes as input the description* $P_{\widetilde{Q}}$ *of an obfuscated circuit* $\widetilde{Q}$ *and an input* x*, and outputs a bit* $b \in \{0,1\}$*.*

A VBB obfuscator should satisfy the following properties for any pseudo-deterministic family of circuits $Q = \{Q_\lambda\}_{\lambda\in\mathbb{N}}$ *with input length* $n = n(\lambda)$*.*

- ***Correctness****: It holds with probability* $1 - \mathsf{negl}(\lambda)$ *over* $\widetilde{Q} \leftarrow \mathsf{Obf}(1^\lambda, P_Q)$ *that for all* $x \in \{0,1\}^n$*,* $\Pr[\mathsf{Eval}(P_{\widetilde{Q}}, x) \to Q(x)] = 1 - \mathsf{negl}(\lambda)$*.*
- ***Security****: For any QPT adversary* $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$*, there exists a QPT simulator* $\{\mathcal{S}_\lambda\}_{\lambda\in\mathbb{N}}$ *such that*

$$\left| \Pr\left[1 \leftarrow \mathcal{A}_\lambda\left(\mathsf{Obf}(1^\lambda, P_Q)\right)\right] - \Pr\left[1 \leftarrow \mathcal{S}_\lambda^{O[Q]}\right] \right| = \mathsf{negl}(\lambda),$$

where $O[Q]$ *is the oracle that computes the map* $x \to Q(x)$*.*

In addition, the VBB obfuscator could optionally satisfy a succinctness criterion.

- ***Succinctness****: The size of the obfuscation* $P_{\widetilde{Q}} \leftarrow \mathsf{Obf}(1^\lambda, P_Q)$ *is polynomially bounded by the size of the description* P_Q *of the input circuit, rather than the size of the circuit (in terms of number of gates) itself. That is,* $|P_{\widetilde{Q}}| \leq \mathrm{poly}(\lambda, |P_Q|)$*, rather than* $|P_{\widetilde{Q}}| \leq \mathrm{poly}(\lambda, |Q|)$*.*

6.1 Construction

- A publicly-verifiable QFHE for pseudo-deterministic circuits in the oracle model $\mathsf{pvQFHE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Ver}, \mathsf{Dec})$ (Definition 16).
- A post-quantum input-succinct obfuscator $(\mathsf{SObf}, \mathsf{SEval})$, which we constructed in Section 4.

Given the above building blocks, we give the following construction of classical obfuscation for pseudo-deterministic quantum circuits.

Classical obfuscation of pseudo-deterministic quantum circuits

- $\mathsf{Obf}(1^\lambda, P_Q)$:
  - Let U be the universal quantum circuit that takes as input the description P_Q of a circuit of input of size N, where N is the length of an input to Q. Let D be the depth of U, and let S be a bound on the size of U.
  - Sample $(\mathsf{pk}, \mathsf{sk}, \mathsf{PP}) \leftarrow \mathsf{pvQFHE.Gen}(1^\lambda, D, |P_Q|, S)$, and $\mathsf{ct} \leftarrow \mathsf{pvQFHE.Enc}(\mathsf{pk}, P_Q)$.
  - Let $\mathsf{DK}(x, \mathsf{ct}', \pi)$ be the following functionality. First, run $\mathsf{Ver}^{\mathsf{PP}}(x, \mathsf{ct}', \pi)$. If the output was $\perp$, then output $\perp$, and otherwise output $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}')$.
  - Sample $\widetilde{\mathsf{DK}} \leftarrow \mathsf{SObf}(1^\lambda, \mathsf{DK})$.
  - Output $\widetilde{Q} := \left(\mathsf{ct}, \mathsf{PP}, \widetilde{\mathsf{DK}}\right)$.
- $\mathsf{Eval}(P_{\widetilde{Q}}, x)$:
  - Let U_x be the quantum circuit that takes as input the description of a circuit P_Q and runs the universal quantum circuit U on $P_Q \| x$.
  - Parse $P_{\widetilde{Q}}$ as $(\mathsf{ct}, \mathsf{PP}, \widetilde{\mathsf{DK}})$.
  - Compute $(\mathsf{ct}', \pi) \leftarrow \mathsf{Eval}^{\mathsf{PP}}(\mathsf{ct}, U_x)$.
  - Output $b := \widetilde{\mathsf{DK}}(x, \mathsf{ct}', \pi)$.

Figure 5: Succinct classical obfuscation of pseudo-deterministic quantum circuits

Theorem 16. *The construction in Figure 5 satisfies correctness, security and succinctness definitions in Definition 18.*

The proofs of correctness and security are almost identical to the proof of Theorem 6.1 of [BKNY23].

Proof. First, correctness follows immediately from the correctness of the VBB obfuscator and the correctness of the publicly-verifiable QFHE scheme (Definition 16). Note that even though the evaluation procedure may include measurements, an evaluator could run coherently, measure just the output bit b, and reverse. By the Gentle Measurement lemma, this implies the ability to run the obfuscated program on any $\mathrm{poly}(\lambda)$ number of inputs.

Next, we show security. For any QPT adversary $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, we define a simulator $\{\mathcal{S}_\lambda\}_{\lambda\in\mathbb{N}}$ as follows, where $\{\widetilde{\mathcal{A}}_\lambda\}_{\lambda\in\mathbb{N}}$ is the simulator for the classical obfuscation scheme $(\mathsf{CObf}, \mathsf{CEval})$, defined based on $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$.

- Sample $(\mathsf{pk}, \mathsf{sk}, \mathsf{PP}) \leftarrow \mathsf{pvQFHE.Gen}(1^\lambda, D, |Q|, S)$, $\mathsf{ct} \leftarrow \mathsf{pvQFHE.Enc}(\mathsf{pk}, 0^{|Q|})$.
- Run $\widetilde{\mathcal{A}}_\lambda^{\mathsf{PP},\mathsf{DK}}(\mathsf{ct})$, answering PP calls honestly, and DK calls as follows.
  - Take (x, ct', π) as input.

- Run $\mathsf{Ver}^{\mathsf{PP}}(x, \mathsf{ct}', \pi)$. If the output was $\perp$ then output $\perp$.
- Otherwise, forward x to the external oracle $O[Q]$, and return the result $b = O[Q](x)$.

- Output $\widetilde{\mathcal{A}}_\lambda$'s output.

Now, for any circuit Q, we define a sequence of hybrids.

- Hybrid 0: Sample $\widetilde{Q} \leftarrow \mathsf{Obf}(1^\lambda, Q)$ and run $\mathcal{A}_\lambda(1^\lambda, \widetilde{Q})$.
- Hybrid 1: Sample $(\mathsf{ct}, \mathsf{PP}, \mathsf{DK})$ as in $\mathsf{Obf}(1^\lambda, Q)$, and run $\widetilde{\mathcal{A}}_\lambda^{\mathsf{PP},\mathsf{DK}}(\mathsf{ct})$.
- Hybrid 2: Same as hybrid 1, except that calls to DK are answered as in the description of $\mathcal{S}_\lambda$.
- Hybrid 3: Same as hybrid 2, except that we sample $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, 0^{|Q|})$. This is $\mathcal{S}_\lambda$.

We complete the proof by showing the following.

- $|\Pr[1 \leftarrow \text{Hybrid 0}] - \Pr[1 \leftarrow \text{Hybrid 1}]| = \mathsf{negl}(\lambda)$. This follows from the security of the classical obfuscation scheme $(\mathsf{CObf}, \mathsf{CEval})$.
- $|\Pr[1 \leftarrow \text{Hybrid 1}] - \Pr[1 \leftarrow \text{Hybrid 2}]| = \mathsf{negl}(\lambda)$. Suppose otherwise. Then there must exist some query made by $\widetilde{\mathcal{A}}_\lambda$ to DK with noticeable amplitude on $(x, \widetilde{\mathsf{ct}}, \pi)$ such that DK does not return $\perp$ but $\mathsf{Dec}(\mathsf{sk}, \widetilde{\mathsf{ct}}) \neq Q(x)$. Thus, we can measure a random one of the $\mathrm{poly}(\lambda)$ many queries made by $\widetilde{\mathcal{A}}_\lambda$ to obtain such an $(x, \widetilde{\mathsf{ct}}, \pi)$, which violates the soundness of the publicly-verifiable QFHE scheme (Definition 16).
- $|\Pr[1 \leftarrow \text{Hybrid 2}] - \Pr[1 \leftarrow \text{Hybrid 3}]| = \mathsf{negl}(\lambda)$. Since sk is no longer used in hybrid 2 to respond to DK queries, this follows directly from the security of the publicly-verifiable QFHE scheme (Definition 16).

Finally, the scheme is succinct. This follows from the compactness of the pvQFHE scheme and the input-succinct obfuscation $(\mathsf{SObf}, \mathsf{SEval})$ scheme. □

References

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

[AGKZ20] Ryan Amos, Marios Georgiou, Aggelos Kiayias, and Mark Zhandry. One-shot signatures and applications to hybrid quantum/classical authentication. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 255–268, 2020.

[AGQY22] Prabhanjan Ananth, Aditya Gulati, Luowen Qian, and Henry Yuen. Pseudorandom (function-like) quantum state generators: New definitions and applications. In *Theory of Cryptography Conference*, pages 237–265. Springer, 2022.

[APM20] Shweta Agrawal and Alice Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear fe. In *Advances in Cryptology – EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I*, page 110–140, Berlin, Heidelberg, 2020. Springer-Verlag.

[Bar21] James Bartusek. Secure quantum computation with classical communication. In *Theory of Cryptography Conference*, pages 1–30. Springer, 2021.

[BBV24] James Bartusek, Zvika Brakerski, and Vinod Vaikuntanathan. Quantum state obfuscation from classical oracles. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1009–1017, 2024.

[BDGM23] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate io from homomorphic encryption schemes. *Journal of Cryptology*, 36(3):27, 2023.

[BGI+01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im) possibility of obfuscating programs. In *Annual international cryptology conference*, pages 1–18. Springer, 2001.

[BGL+15] Nir Bitansky, Sanjam Garg, Huijia Lin, Rafael Pass, and Sidharth Telang. Succinct randomized encodings and their applications. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '15, page 439–448, New York, NY, USA, 2015. Association for Computing Machinery.

[BGMZ18] James Bartusek, Jiaxin Guan, Fermi Ma, and Mark Zhandry. Return of ggh15: provable security against zeroizing attacks. In *Theory of Cryptography Conference*, pages 544–574. Springer, 2018.

[BK21] Anne Broadbent and Raza Ali Kazmi. Constructions for quantum indistinguishability obfuscation. In *Progress in Cryptology – LATINCRYPT 2021: 7th International Conference on Cryptology and Information Security in Latin America, Bogotá, Colombia, October 6–8, 2021, Proceedings*, page 24–43, Berlin, Heidelberg, 2021. Springer-Verlag.

[BKL+22] James Bartusek, Yael Tauman Kalai, Alex Lombardi, Fermi Ma, Giulio Malavolta, Vinod Vaikuntanathan, Thomas Vidick, and Lisa Yang. Succinct classical verification of quantum computation. In *Annual International Cryptology Conference*, pages 195–211. Springer, 2022.

[BKNY23] James Bartusek, Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. Obfuscation of pseudo-deterministic quantum circuits. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1567–1578, 2023.

[BM22] James Bartusek and Giulio Malavolta. Indistinguishability Obfuscation of Null Quantum Circuits and Applications. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:13, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Bra18] Zvika Brakerski. Quantum fhe (almost) as secure as classical. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, pages 67–95, Cham, 2018. Springer International Publishing.

[CLLW22] Kai-Min Chung, Yi Lee, Han-Hsuan Lin, and Xiaodi Wu. Constant-round blind classical verification of quantum sampling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 707–736. Springer, 2022.

[CLLZ21] Andrea Coladangelo, Jiahui Liu, Qipeng Liu, and Mark Zhandry. Hidden cosets and applications to unclonable cryptography. In *Annual International Cryptology Conference*, pages 556–584. Springer, 2021.

[CMS19] Alessandro Chiesa, Peter Manohar, and Nicholas Spooner. Succinct arguments in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography*, pages 1–29, Cham, 2019. Springer International Publishing.

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. Ggh15 beyond permutation branching programs: proofs, attacks, and candidates. In *Annual International Cryptology Conference*, pages 577–607. Springer, 2018.

[DFM20] Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In *Advances in Cryptology – CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part III*, page 602–631, Berlin, Heidelberg, 2020. Springer-Verlag.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the fiat-shamir transformation in the quantum random-oracle model. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II*, page 356–383, Berlin, Heidelberg, 2019. Springer-Verlag.

[DH22] Whitfield Diffie and Martin E Hellman. New directions in cryptography. In *Democratizing cryptography: the work of Whitfield Diffie and Martin Hellman*, pages 365–390. 2022.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography Conference*, pages 498–527. Springer, 2015.

[GGH+16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.

[GKNV25] Sam Gunn, Yael Kalai, Anand Natarajan, and Ági Villányi. Classical commitments to quantum states. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, page 234–244, New York, NY, USA, 2025. Association for Computing Machinery.

[GR07] Shafi Goldwasser and Guy N Rothblum. On best-possible obfuscation. In *Theory of Cryptography Conference*, pages 194–213. Springer, 2007.

[HT25] Mi-Ying Huang and Er-Cheng Tang. Obfuscation of unitary quantum programs. *arXiv preprint arXiv:2507.11970*, 2025.

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd annual ACM SIGACT symposium on theory of computing*, pages 60–73, 2021.

[Mah18] Urmila Mahadev. Classical verification of quantum computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267, 2018.

[Mah20] Urmila Mahadev. Classical homomorphic encryption for quantum circuits. *SIAM Journal on Computing*, 52(6):FOCS18–189, 2020.

[MNZ24] Tony Metger, Anand Natarajan, and Tina Zhang. Succinct Arguments for QMA from Standard Assumptions via Compiled Nonlocal Games . In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1193–1201, Los Alamitos, CA, USA, October 2024. IEEE Computer Society.

[NZ23] Anand Natarajan and Tina Zhang. Bounding the Quantum Value of Compiled Nonlocal Games: From CHSH to BQP Verification . In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1342–1348, Los Alamitos, CA, USA, November 2023. IEEE Computer Society.

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

[Shm22b] Omri Shmueli. Semi-quantum tokenized signatures. In *Annual International Cryptology Conference*, pages 296–319. Springer, 2022.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[SZ25] Omri Shmueli and Mark Zhandry. On one-shot signatures, quantum vs classical binding, and obfuscating permutations. *Cryptology ePrint Archive*, 2025.

[WW21] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious lwe sampling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 127–156. Springer, 2021.

[Zha12] Mark Zhandry. How to construct quantum random functions. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 679–687, 2012.

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In *Annual International Cryptology Conference*, pages 239–268. Springer, 2019.

[Zha21] Mark Zhandry. How to construct quantum random functions. *Journal of the ACM (JACM)*, 68(5):1–43, 2021.

[Zha22] Mark Zhandry. New constructions of collapsing hashes. In *Annual international cryptology conference*. Springer, 2022.

A Remaining proofs from Section 3

A.1 Correctness

Proof of Theorem 4. This proof is almost identical to the proof of Theorem 4.6 of [BKNY23], but we repeat it here for completeness. The only difference is that we are using one-shot signatures instead of signature tokens; this is what allows us to make the commitment keys entirely classical.

We will assume that the one-shot signature schemes $\mathsf{OSS}, \overline{\mathsf{OSS}}$ are perfectly correct. If they are statistically correct, then the Pauli functional commitment scheme we obtain is still correct, because we allow for a negligible statistical distance.

We will first show that the map applied by $\mathsf{Com}^{\mathsf{CK}}(b)$ in the case that the measurement of the first qubit of $\mathcal{S}'$ is $1-b$ successfully takes $|S_{y,1-b}\rangle \to |S_{y,b}\rangle$. Since we are assuming perfect correctness from $\overline{\mathsf{OSS}}$, it suffices to show that for any balanced affine subspace $|S_y\rangle$,

$$H^{\otimes n}\mathsf{Phase}^{O[S_y^\perp]}H^{\otimes n}\,|S_{y,1-b}\rangle \to |S_{y,b}\rangle\,,$$

where $\mathsf{Phase}^{O[S_y^\perp]}$ is the map $|s\rangle \to (-1)^{O[S_y^\perp](s)}\,|s\rangle$, and $O[S_y^\perp]$ is the oracle that outputs 0 if $s \in S^\perp$ and 1 if $s \notin S^\perp$. This was also shown in [AGKZ20].

Let S_y^* be the subspace parallel to affine subspace S_y, and let $S_{y,b}^*$ be the coset of S_y^* whose first coordinate is b. We will use the facts that $S_{y,1}^* = S_{y,0}^* + w$ for some w, and that $S_{y,0} = S_{y,0}^* + v_0$ and $S_{y,1} = S_{y,1}^* + v_1$ for some v_0, v_1 such that $v_0 + v_1 = w$. Also note that for any $s \in S_y^\perp$, $s \cdot w = 0$, and for any $s \in (S_{y,0}^*)^\perp \setminus (S_y^*)^\perp$, $s \cdot w = 1$.

$$
\begin{aligned}
&H^{\otimes n}\mathsf{Ph}^{O[S_y^\perp]}H^{\otimes n}\left|S_{y,1-b}\right\rangle \\
&\quad= H^{\otimes n}\mathsf{Ph}^{O[S_y^\perp]}H^{\otimes n}\frac{1}{\sqrt{2^{n/2-1}}}\left(\sum_{s\in S^*_{y,0}}\left|s+v_{1-b}\right\rangle\right) \\
&\quad= H^{\otimes n}\mathsf{Ph}^{O[S_y^\perp]}\frac{1}{\sqrt{2^{n/2+1}}}\left(\sum_{s\in (S^*_{y,0})^\perp}(-1)^{s\cdot v_{1-b}}\left|s\right\rangle\right) \\
&\quad= H^{\otimes n}\mathsf{Ph}^{O[S_y^\perp]}\frac{1}{\sqrt{2^{n/2+1}}}\left(\sum_{s\in (S^*_{y})^\perp}(-1)^{s\cdot w+s\cdot v_b}\left|s\right\rangle+\sum_{s\in (S^*_{y,0})^\perp\setminus(S^*_{y})^\perp}(-1)^{s\cdot w+s\cdot v_b}\left|s\right\rangle\right) \\
&\quad= H^{\otimes n}\mathsf{Ph}^{O[S_y^\perp]}\frac{1}{\sqrt{2^{n/2+1}}}\left(\sum_{s\in (S^*_{y})^\perp}(-1)^{s\cdot v_b}\left|s\right\rangle+\sum_{s\in (S^*_{y,0})^\perp\setminus(S^*_{y})^\perp}(-1)^{1+s\cdot v_b}\left|s\right\rangle\right) \\
&\quad= H^{\otimes n}\frac{1}{\sqrt{2^{n/2+1}}}\left(\sum_{s\in S^\perp}(-1)^{s\cdot v_b}\left|s\right\rangle+\sum_{s\in (S^*_{y,0})^\perp\setminus(S^*_{y})^\perp}(-1)^{s\cdot v_b}\left|s\right\rangle\right) \\
&\quad= H^{\otimes n}\frac{1}{\sqrt{2^{n/2+1}}}\left(\sum_{s\in (S^*_{y,0})^\perp}(-1)^{s\cdot v_b}\left|s\right\rangle\right) \\
&\quad= \left|S_{y,b}\right\rangle.
\end{aligned}
$$

Now consider applying $\mathsf{Com}^{\mathsf{CK}}$ to a pure state. By the correctness of OSS, Com can produce a valid signature σ_0 of 0, and is able to access the $\overline{\mathsf{O}} = (P, P^{-1}, D)$ oracles of OSS through $\mathsf{CK}_P(\mathsf{vk},\cdot)$, $\mathsf{CK}_{P^{-1}}(\mathsf{vk},\cdot,\cdot)$ and $\mathsf{CK}_D(\mathsf{vk},\sigma_0)$. These oracles $\overline{O} = (P, P^{-1}, D)$ are an instance of $\overline{\mathsf{OSS}}$ generated using randomness determined by $\mathsf{RO}(\mathsf{vk})$. By the correctness of $\overline{\mathsf{OSS}}.\mathsf{Gen}^{P,P^{-1},D}$, Com is able to generate a $(\overline{vk}, |\overline{\mathsf{sk}}\rangle)$ pair $(y, |S_y\rangle)$.

Applying $\mathsf{Com}^{\mathsf{CK}}$ to pure state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ produces (up to negligible trace distance) the state

$$|\psi_{\mathsf{Com}}\rangle = \alpha_0|0\rangle|S_{y,0}\rangle + \alpha_1|1\rangle|S_{y,1}\rangle,$$

and a signature c on the bit 1.

We claim that measuring and decoding $|\psi_{\mathsf{Com}}\rangle$ in the standard (resp. Hadamard) basis produces the same distribution as directly measuring $|\psi\rangle$ in the standard (resp. Hadamard) basis. As a mixed state is a probability distribution over pure states, this will complete the proof of correctness.

First, it is immediate that measuring $|\psi_{\mathsf{Com}}\rangle$ in the standard basis produces a bit b with probability $|\alpha_b|^2$ along with a vector s such that $s \in (S+v)_b$.

Next, note that applying Hadamard to each qubit of $|\psi_{\mathsf{Com}}\rangle$ except the first results in the state

$$\alpha_0|0\rangle\left(\sum_{s\in(S^*_{y,0})^\perp}(-1)^{s\cdot v_0}|s\rangle\right) + \alpha_1|1\rangle\left(\sum_{s\in(S^*_{y,0})^\perp}(-1)^{s\cdot v_1}|s\rangle\right),$$

and thus, measuring each of these qubits (except the first) in the Hadamard basis produces a vector s and a single-qubit state

$$(-1)^{s\cdot v_0}\alpha_0|0\rangle + (-1)^{s\cdot v_1}\alpha_1|1\rangle = \alpha_0|0\rangle + (-1)^{s\cdot w}\alpha_1|1\rangle.$$

So, measuring this qubit in the Hadamard basis is equivalent to measuring $|\psi\rangle$ in the Hadamard basis and masking the result with $s\cdot w$. Recalling that $s\cdot w = 0$ if $s \in (S^*_y)^\perp$ and $s\cdot w = 1$ if $s \in (S^*_{y,0})^\perp \setminus (S^*_y)^\perp$ completes the proof of correctness, since $S^\perp_y = (S^*_y)^\perp$ and $S^\perp_{y,0} = (S^*_{y,0})^\perp$ □

A.2 Binding

Lemma 7. *The collapse-binding definition of Definition 9 implies the single-bit binding definition of Definition 7.*

Proof. This statement is implicitly proved in Lemma A.1 of [BKNY23], which shows that collapse-binding implies a notion of binding called "unique-message binding", and also shows that this notion implies string-binding. String-binding implies single-bit binding, and therefore we are done.

□