Obfuscation of Unitary Quantum Programs

Mi-Ying (Miryam) Huang*[1] and Er-Cheng Tang†[2]

[1]University of Southern California
[2]University of Washington

Abstract

Program obfuscation aims to hide the inner workings of a program while preserving its functionality. In the quantum setting, recent works have obtained obfuscation schemes for specialized classes of quantum circuits. For instance, Bartusek, Brakerski, and Vaikuntanathan (STOC 2024) constructed a quantum state obfuscation scheme, which supports the obfuscation of quantum programs represented as quantum states for pseudo-deterministic quantum programs with classical inputs and outputs in the classical oracle model.

In this work, we improve upon existing results by constructing the first quantum state obfuscation scheme for *unitary* (or *approximately unitary*) quantum programs supporting *quantum* inputs and outputs in the classical oracle model. At the core of our obfuscation scheme are two novel ingredients: a *functional* quantum authentication scheme that allows key holders to learn specific functions of the authenticated quantum state with *simulation-based* security, and a compiler that represents an arbitrary quantum circuit as a *projective* linear-plus-measurement quantum program described by a sequence of non-adaptive Clifford gates interleaved with adaptive and *compatible* measurements.

*Email: miying.huang@usc.edu
†Email: erchtang@uw.edu

This is the full version of a paper that appears in FOCS 2025.

Contents

1 Introduction **3**
- 1.1 Our Results . . . 3

2 Technical Overview **6**
- 2.1 Hurdles in Quantum Functionalities . . . 7
- 2.2 Our Solutions . . . 8
- 2.3 Obfuscation of Approximately Unitary Quantum Programs . . . 10

3 Preliminary **13**
- 3.1 Quantum Circuits and Programs . . . 14
- 3.2 Quantum Teleportation . . . 14
- 3.3 Purified Random Oracle . . . 15
- 3.4 Tokenized Signature . . . 16
- 3.5 Post-quantum PRF . . . 17
- 3.6 Useful Lemmas . . . 17

4 Functional Authentication **19**
- 4.1 Definition . . . 19
- 4.2 Construction and Security Proof . . . 21

5 Projective LM Quantum Program **27**

6 Approximately Unitary Quantum Program **37**

7 Quantum State Obfuscation **40**
- 7.1 Construction . . . 41
- 7.2 Security . . . 44

1 Introduction

Program obfuscation is an important problem in cryptography and complexity, where the goal is to obscure the internal workings of a program while preserving its functionality. The study of obfuscation has a long tradition in classical cryptography. It is well-known that the notion of virtual black-box (VBB) obfuscation, which guarantees that the obfuscated program discloses no additional information beyond what can be learned from its input-output behavior, is impossible to achieve for general circuits [BGI+01]. On the one hand, the impossibility has led to the study of weaker notions, such as indistinguishability obfuscation (iO) [BGI+01, GGH+13, AJL+19, JLS21, JJ22], which only requires the obfuscations of functionally equivalent circuits to be computationally indistinguishable. On the other hand, it has led to the consideration of stronger models, such as the pseudorandom oracle model [JLLW23], where VBB obfuscation, and even the stronger notion of ideal obfuscation, is feasible.

Motivated by the importance of obfuscation in classical cryptography, a more recent line of work has emerged that studies the feasibility of obfuscating quantum programs. Similarly to the classical setting, it is known that VBB obfuscation of general quantum circuits is impossible in the plain model under standard cryptographic assumptions [ABDS21], so efforts have naturally turned to weaker notions of obfuscation or stronger models. For example, [BK21] achieves iO for quantum circuits with logarithmically many non-Clifford gates supporting quantum inputs and outputs. [BKNY23] achieves obfuscation for pseudo-deterministic quantum circuits with classical inputs and outputs in the classical oracle model. More recently, [CG24] introduces the concept of "quantum state obfuscation," which additionally supports obfuscation of programs equipped with a quantum state. The same work demonstrates its applicability for constructing other quantum cryptographic primitives, e.g., copy-protection. Informally, a quantum state obfuscation scheme takes a quantum program (ψ, Q), comprising a quantum circuit Q and an auxiliary quantum state ψ, and outputs an obfuscated quantum program with the same functionality. Later on, [BBV24] constructs ideal quantum state obfuscation in the classical oracle model, again for pseudo-deterministic quantum programs with classical inputs and outputs.

While these works represent significant progress, they fall short of supporting obfuscation of general quantum input-output functionalities: they are either restricted to pseudo-deterministic quantum programs with classical inputs and outputs, or to quantum circuits with a logarithmic number of non-Clifford gates. These limitations lead to the following compelling question:

Is it possible to obfuscate a broad class of quantum programs that support quantum inputs and outputs?

1.1 Our Results

We affirmatively address the aforementioned question by presenting the first quantum state obfuscation scheme for *unitary* (or negligibly close to unitary) quantum programs supporting *quantum* inputs and outputs in the classical oracle model. Our scheme resolves an open question raised by Bartusek, Brakerski, and Vaikuntanathan [BBV24], who presented a

scheme that only obfuscates pseudo-deterministic[1] quantum programs supporting classical inputs and outputs. To capture the security of our scheme, we propose a definition of ideal obfuscation for quantum programs that implement unitary transformations. Moreover, when restricted to programs with classical functionality, our definition reduces to the standard notion of ideal obfuscation. The obfuscated program of our scheme can be used to evaluate multiple quantum inputs.

Theorem 1.1 (Informal)**.** *There exists a quantum state obfuscation scheme for the class of all approximately*[2] *unitary quantum programs supporting quantum inputs and outputs, achieving the notion of ideal obfuscation (Definition 7.1) in the classical oracle model.*

Our quantum state obfuscation is proven secure in the classical oracle model, where the obfuscator can output a classical oracle that is accessible to all parties. This can be viewed as an idealized model in which ideal obfuscation is achievable for all classical functions. Our result achieves unconditional security when the constructed oracle internally uses a truly random function, and additionally relies on the existence of a post-quantum one-way function when the classical oracle is required to be efficiently computable, as in [BBV24].

Historically, such oracle-assisted results have often foreshadowed oracle-free constructions, as demonstrated by public-key quantum money [Zha21b] and copy-protection [CLLZ21]. The classical oracle model can be heuristically instantiated with post-quantum iO in the plain model. In light of a recent work [JLLW23], it also seems plausible to provably instantiate the classical oracle model with post-quantum security from the pseudorandom oracle model, a setting that only requires modeling idealized hash functions. Below, we summarize the comparison between our result and prior quantum obfuscation constructions in Table 1.

	Obfuscator input	Obfuscator output	Program input	Program output	Program class	Assumption/ Model	Result
[BK21]	Classical	Quantum	Quantum	Quantum	Unitaries with log-many non-Clifford gates	iO for classical circuits	iO*
[BM22]	Classical	Classical	Quantum*	Classical	Null circuits	Classical oracle model + LWE	iO
[BKNY23]	Classical	Quantum	Classical	Classical	(Pseudo-)deterministic circuits	Classical oracle model + LWE	VBB
[CG24]	Quantum	Quantum	Classical	Classical	Deterministic circuits	Quantum oracle model	iO
[BBV24]	Quantum	Quantum	Classical	Classical	(Pseudo-)deterministic circuits	Classical oracle model	Ideal
This paper	Quantum	Quantum	Quantum	Quantum	(Approximately-)unitary circuits	Classical oracle model	Ideal

Table 1: Construction of quantum obfuscation schemes. Note the caveats: (1) The obfuscator in [BK21] produces a quantum state that enables evaluation of the program on a single input, after which the state may become destroyed. While suitable for certain applications, this behavior deviates from the standard notion of program obfuscation, which typically requires support for multiple evaluations on different inputs. (2) The construction in [BM22] supports quantum inputs but requires multiple identical copies of the quantum input.

[1] Every pseudo-deterministic quantum program can be directly converted into an approximately unitary quantum program by coherent execution of the program. Therefore, our scheme also obfuscates pseudo-deterministic quantum programs.

[2] As an improvement on the allowed approximation factor, our result can obfuscate programs with up to $\mathrm{negl}(\lambda)$ approximation error, whereas [BBV24] only obfuscates programs that have $2^{-2n}\,\mathrm{negl}(\lambda)$ error.

Related Work. Very recently, [Zha25] showed that every *classically described* quantum circuit implementing a unitary operator U, potentially using ancilla qubits (i.e., $|0\rangle$), can be used to implement its conjugate U^*, transpose $U^\top$, and controlled operator $\mathsf{ctrl}\text{-}U$. Based on this observation, [Zha25] proposed that having access to a unitary U should be modeled as having oracle access to the full collection $\mathsf{ctrl}\text{-}U$, $\mathsf{ctrl}\text{-}U^\dagger$, $\mathsf{ctrl}\text{-}U^*$, $\mathsf{ctrl}\text{-}U^\top$. In this work, we show how to obfuscate every program that implements a unitary transformation $\mathcal{U} : \rho \mapsto U\rho U^\dagger$ into a *quantumly described* quantum program with an auxiliary quantum state[3]. Moreover, the latter allows *only* the computation of $\mathcal{U}$ and $\mathcal{U}^\dagger$, or equivalently, the computation of $\mathsf{ctrl}\text{-}(U^\dagger \mathsf{SWAP}^{\otimes n} U)$ where U acts on n qubits (see Remark 2). This highlights a key distinction between two flavors of program access to a unitary transformation: The computational power of program access to a quantumly described quantum program can be encapsulated using a single oracle, whereas the power of program access to a classically described quantum circuit needs to be captured using multiple oracles.

Applications. [CG24] demonstrated that quantum state obfuscation implies a best-possible copy-protection scheme. Accordingly, our construction realizes such a scheme and possibly extends to copy-protecting unitary functionalities. [BKNY23] employed quantum obfuscation to construct a functional encryption scheme. By instantiating their construction with our obfuscator, we enable a functional encryption scheme supporting unitary functionalities.

Open Problems. Here, we discuss some open problems motivated by this work.

1. One interesting problem is to obfuscate isometries, which map a quantum input ρ_{in} to a quantum output of the form $V(\rho_{\mathsf{in}} \otimes |0^\kappa\rangle\langle 0^\kappa|)V^\dagger$ for some efficient unitary operator V, while ensuring reusability of the obfuscated program. Moreover, a related question would be the possibility of obfuscating classical randomized functions.

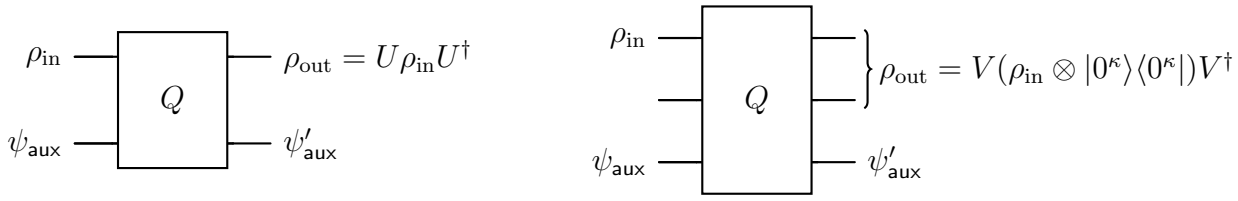


Figure 1: The quantum program (Q, ψ_{aux}) on the left implements a unitary on its input, whereas the quantum program (Q, ψ_{aux}) on the right implements an isometry on its input. Our scheme obfuscates unitaries, and it remains open to obfuscate isometries.

2. Can one achieve quantum state indistinguishability obfuscation in the plain model assuming post-quantum indistinguishability obfuscation?

[3] Looking ahead, our obfuscated program is designed to carry several authenticated T magic states to assist the implementation of each individual T gate. These states can only be used to compute T but not its conjugate T^*. Consequently, the obfuscated program cannot be used to implement $\mathcal{U}^*$. Furthermore, since each magic state is protected under a distinct Pauli key and the classical oracle that holds the keys rejects out-of-order executions, the obfuscated program cannot be used to implement $\mathcal{U}^\top$. It is worth noting that $\mathcal{U}^\dagger$ is achievable (unlike $\mathcal{U}^*$ and $\mathcal{U}^\top$) by performing a rewinding execution after a normal forward execution.

3. The use of classical obfuscation in conjunction with pseudorandom functions has found great applications in classical cryptography. Can the combination of our obfuscation scheme for unitary programs and recent studies on pseudorandom unitaries [MH25, SHH25] open new avenues in quantum cryptography?
4. In quantum complexity, several oracle separation results have been established relative to quantum oracles but remain unresolved relative to classical oracles. Can obfuscation of unitaries in the classical oracle model help bridge the gap by instantiating the quantum oracles using classical oracles and auxiliary quantum states?

2 Technical Overview

In this section, we start by revisiting the work of [BBV24], which served as a key inspiration for our work. We then explore the challenges of obfuscating general unitary programs with quantum inputs and outputs. Afterward, we present our new techniques, which overcome these challenges and substantially strengthen the results of [BBV24].

We begin by discussing an approach for obfuscating quantum circuits. A common method to perform quantum computation without revealing the computation's internal details is to apply homomorphic operations on authenticated quantum states.[4] However, no known quantum authentication scheme can fully support universal quantum computation through homomorphic operations. A common workaround is to implement the unsupported homomorphic gates through "magic state injection," which appends auxiliary magic states to the input state, performs measurements on them in the middle of the computation, and applies adaptive correction gates based on the measurement outcomes. Nevertheless, this method faces a key challenge in the context of obfuscation: revealing the adaptive correction gates may leak the intermediate measurement outcomes. For instance, one approach is to compile a quantum circuit as an adaptive sequence of Clifford gates, applied to the input along with some magic states. However, this approach runs precisely into the issue that subsequent choices of Clifford gates reveal the previous measurement outcomes.

To overcome this issue, [BBV24] develops a new representation of quantum programs (including those with auxiliary quantum states) with classical inputs and outputs as what they call a "linear-plus-measurement" (LM) quantum program. The latter consists of a quantum state ψ_+, a predetermined sequence of Clifford gates $U_1, \ldots, U_t$, and some classical functions $f_1, \ldots, f_t, g$ for performing *adaptive measurements*. The LM quantum program takes a classical input $\mathbb{i}$, and produces a classical output as follows.

- Initialize a quantum register V with the quantum state ψ_+.
- For $j = 1, \ldots, t$, do:
  - Apply U_j on register V.
  - Take the classical function $f_j^{\mathbb{i},r_1,\ldots,r_{j-1}}(\cdot)$ that depends on the classical input $\mathbb{i}$ and all previous measurement outcomes $r_1, \ldots, r_{j-1}$. Coherently compute $f_j^{\mathbb{i},r_1,\ldots,r_{j-1}}(V)$

[4] It is well-known that quantum authentication implies quantum encryption. Such a method also appeared in works that aimed for quantum one-time programs [BGS13].

and measure the function output in the computational basis. Denote the measurement result by r_j.

– Apply $U_j^\dagger$ on register V.

- Output $g(\mathbb{i}, r_1, \ldots, r_t)$.

Representing a quantum computation as an LM quantum program brings a key benefit: the adaptive component of the quantum computation is now reduced to the coherent evaluation of classical functions, which can be protected using classical obfuscation. [BBV24] shows that any quantum program can be compiled into an LM quantum program, where the state ψ_+ consists of the original program along with ancillary zero states and magic states.

With the compiler of LM quantum program in hand, [BBV24] realizes the aforementioned obfuscation approach as follows. The obfuscator starts by running the compiler to produce an LM quantum program, where all information about the original program is encoded in the quantum state ψ_+. Then the obfuscator authenticates ψ_+, prepares classical oracles F_j, G that internally computes f_j, g, and outputs the authenticated state and the classical oracles as the obfuscated program. To evaluate the obfuscated program on classical input $\mathbb{i}$, one can homomorphically apply each Clifford gate U_j on the authenticated states on their own. Meanwhile, each homomorphic adaptive measurement f_j can be performed by querying the classical oracle F_j in superposition on the tuple[5] consisting of the input $\mathbb{i}$, the authenticated state, and the outputs of $\mathsf{F}_1, \ldots, \mathsf{F}_{j-1}$. Crucially, each oracle F_j is designed to output only the encoding of previous intermediate measurement results $r_1, \ldots, r_j$, which allows future measurements to adaptively depend on them without having the measurement results revealed in plaintext. At the end, one queries the oracle G on the encodings produced by $\mathsf{F}_1, \ldots, \mathsf{F}_t$ to obtain the final output $g(r_1, \ldots, r_t)$ in the clear. [BBV24] shows that these steps could be done assuming ideal classical obfuscation and obtains a quantum state obfuscation scheme for a restricted class of quantum circuits, whose inputs and outputs are required to be classical and pseudo-deterministic.

2.1 Hurdles in Quantum Functionalities

We now examine in more detail why [BBV24] confines its scope to classical pseudo-deterministic functionalities. Then, we highlight the challenges that arise when extending obfuscation to quantum programs with quantum inputs and outputs, as well as to more general quantum circuits beyond pseudo-deterministic ones. Here, we explain the difficulties from the perspectives of construction, functionality, and security analysis.

In the construction of [BBV24], the classical oracle plays a central role in interpreting the execution of the obfuscated quantum program. In particular, the classical oracle learns the classical input $\mathbb{i}$, checks whether the same classical input is being used throughout the execution, and returns the computed classical output. However, when the inputs and outputs are quantum states, it is unclear whether a classical oracle can interpret the quantum input, ensure the consistency of quantum input, and return the correct quantum output to

[5] The tuple also contains a tokenized signature for ensuring input consistency across oracle calls, but we omit it here for presentation simplicity.

the evaluator. Therefore, restricting to programs with classical inputs and outputs seems inherent in their approach.

From the functionality standpoint, the obfuscator has to produce a program that can be evaluated on as many inputs as needed. Such a reusablity requirement can be difficult to achieve for general quantum programs that are equipped with quantum states due to the no-cloning theorem. To ensure that the obfuscated program remains reusable, prior works [BKNY23, BBV24] focus on quantum programs where the output is classical and pseudo-deterministic. In this case, measuring the output of the obfuscated program in the standard basis would barely collapse the quantum state, so the obfuscated program can be restored after each evaluation through rewinding. Nevertheless, such an argument does not work beyond classical pseudo-deterministic quantum programs, and existing techniques fail to obfuscate more general quantum programs (with quantum states) in a way that ensures reusability of the obfuscated program.

Regarding security analysis, [BBV24] heavily relies on the functionality being classical and pseudo-deterministic, which allows classifying all potential intermediate measurement results into possible and impossible ones, according to whether they may occur in honest executions. Following this idea, [BBV24] reduces the security proof to the simpler task of showing the infeasibility of producing impossible measurement results. However, such a classification is no longer effective when going beyond classical and pseudo-deterministic functionalities. Instead of tackling only the support of the distribution, one has to analyze the entire distribution of homomorphic measurement results that the adversary produces dynamically, a technique that is not present in [BBV24]. Even if the distribution can be analyzed dynamically, one must still demonstrate that the quantum state always corresponds to some honestly computed state. However, correctly interpreting a quantum state as an intermediate state in the LM quantum program of [BBV24] requires identifying the exact stage of execution to which the state belongs. This is particularly challenging when different stages of the LM quantum program are only loosely related or not directly comparable. Moreover, it would be difficult to characterize quantum states that correspond to superpositions of intermediate states across multiple stages of the LM quantum program execution.

2.2 Our Solutions

Improved representation of quantum computation. We offer a new perspective on quantum computation by introducing the notion of *projective* LM (PLM) quantum programs, a variant of LM quantum programs where the adaptive measurements involved are all compatible[6] with each other and effectively projects the quantum state onto a known orthonormal basis. Formally, an LM quantum program is called *projective*, if for every classical input $\mathbb{i}$, there exists an orthonormal basis $\{|\Phi_{\mathbb{i},(r_1,\cdots,r_t)}\rangle\}$ such that the cumulative effect of the adaptive measurements is to apply the projective measurement $\{|\Phi_{\mathbb{i},(r_1,\cdots,r_t)}\rangle\langle\Phi_{\mathbb{i},(r_1,\cdots,r_t)}|\}_{r_1,\ldots,r_t}$, where r_j stands for the j-th adaptive measurement outcome. Following this definition, on classical input $\mathbb{i}$, executing the PLM quantum program whose quantum state is changed from

[6] Recall that two measurements are considered compatible if their corresponding projectors commute. In our case, each adaptive measurement can be specified by a pair $(U_j, f_j^{\mathbb{i},r_1,\ldots,r_{j-1}})$. We require compatibility only between any two pairs $(U_i, f_i^{\mathbb{i},r_1,\ldots,r_{i-1}})$ and $(U_j, f_j^{\mathbb{i},r_1,\ldots,r_{j-1}})$ that share the same prefix $(\mathbb{i}, r_1, \ldots)$.

ψ_+ to $|\Phi_{\mathtt{i},(r_1,\cdots,r_t)}\rangle$ always yields deterministic adaptive measurement outcomes $(r_1,\ldots,r_t)$.

The projective property provides a unified basis for analyzing the quantum state across different stages of program execution, thereby eliminating the need to identify the stage to which the state corresponds. In addition, PLM quantum programs offer a natural way to interpret adaptive measurements as a process of revealing how ψ_+ decomposes into orthonormal basis vectors $\{|\Phi_{\mathtt{i},(r_1,\cdots,r_t)}\rangle\}_{r_1,\ldots,r_t}$ without altering the amplitudes. Moreover, this basis is a convenient basis for analysis. For example, performing the measurement $\left\{\sum_{r:g(\mathtt{i},r)=y}|\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|\right\}_y$ on ψ_+ is equivalent to measuring the classical output y of the original quantum program on input $\mathtt{i}$. To explain, LM quantum program on input $\mathtt{i}$ performs the adaptive measurements to obtain $r_1,\ldots,r_t$ and outputs $y=g(r_1,\ldots,r_t)$. In PLM quantum program, $r_1,\ldots,r_t$ can be obtained alternatively by performing the projective measurement $\{|\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|\}_r$ on ψ_+. If one only wants to obtain the final output and not the intermediate measurement results, then they can directly apply the projective measurement $\left\{\sum_{r:g(\mathtt{i},r)=y}|\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|\right\}_y$ on ψ_+ to obtain the output y. Furthermore, such an argument continues to hold even if ψ_+ is entangled with the adversary's states. Thus, decomposing in this basis enables a clean analysis of the adversary's behavior on each component while preserving the entanglement structure.

Handling quantum inputs and outputs. To obfuscate programs with quantum inputs and outputs, we integrate quantum teleportation into the obfuscation construction. In more detail, the obfuscator additionally prepares EPR pairs $(\mathsf{epr}_{\mathrm{in},\mathsf{pub}},\mathsf{epr}_{\mathrm{in},\mathsf{priv}})$, $(\mathsf{epr}_{\mathrm{out},\mathsf{priv}},\mathsf{epr}_{\mathrm{out},\mathsf{pub}})$, provides $(\mathsf{epr}_{\mathrm{in},\mathsf{pub}},\mathsf{epr}_{\mathrm{out},\mathsf{pub}})$ in the clear, and provides the authentication of $(\mathsf{epr}_{\mathrm{in},\mathsf{priv}},\mathsf{epr}_{\mathrm{out},\mathsf{priv}})$ as part of the obfuscated quantum program.[7] To evaluate the obfuscated program, the evaluator teleports a quantum input using $\mathsf{epr}_{\mathrm{in},\mathsf{pub}}$ and obtains a classical teleportation result $\mathtt{i}$. Next, the evaluator performs the following computation homomorphically over authenticated states, where only $\mathtt{i}$ and y are represented as plaintext:

- Interpret $\mathtt{i}$ as a Pauli correction $P_{\mathtt{i}}$ and apply it on $\mathsf{epr}_{\mathrm{in},\mathsf{priv}}$.
- Use the resulting state as the quantum input.
- Perform the quantum computation of interest.
- Teleport the quantum output using $\mathsf{epr}_{\mathrm{out},\mathsf{priv}}$ and output the teleportation result y.

The evaluator can apply the Pauli correction P_y on $\mathsf{epr}_{\mathrm{out},\mathsf{pub}}$ to recover the quantum output.

Enabling program reusability. The idea above runs into a natural challenge. Equipped only with a limited number of EPR pairs, how can the obfuscated program be reused for multiple evaluations? Fortunately, we show that reusability can be guaranteed whenever the quantum program implements a unitary or approximately-unitary functionality, i.e., the quantum output is close to the result of applying a unitary to the quantum input. Intuitively, such programs can be reused because the quantum output would be unentangled with the auxiliary state, so one can retrieve the output without damaging the auxiliary state. We refer the reader to Section 6 for further details.

[7] A similar idea was considered in [BGS13] in the context of quantum one-time programs.

Providing functional security. Although the quantum authentication scheme of [BBV24] supports homomorphic measurements, its security guarantees only capture static distributional properties of each measurement result, but do not account for the joint distribution with quantum states or any dynamic behavior. We remedy this shortcoming by upgrading the authentication into a functional authentication scheme with simulation-based security. Syntactically, the functional authentication scheme consists of the following components:

- $k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n)$ generates a classical key k with security parameter λ.
- $\widetilde{\rho} \leftarrow \mathsf{Enc}_k(\rho)$ authenticates an n-qubit state ρ into a ciphertext state $\widetilde{\rho}$.
- $\widetilde{\rho}_{\theta,G} \leftarrow \mathsf{Eval}_{\theta,G}(\widetilde{\rho})$ homomorphically applies the unitary $\mathsf{H}^\theta G$ over the ciphertext $\widetilde{\rho}$, where H is a Hadamard gate, $\theta \in \{0,1\}^n$, and G is a unitary consisting of CNOT gates.
- $\top/\bot \leftarrow \mathsf{Ver}_{k,\theta,G}(\widetilde{m})$ classically verifies whether $\widetilde{m}$ is a valid measurement result of the ciphertext state in the column basis of $\mathsf{H}^\theta G$. It outputs either $\top$ (valid) or $\bot$ (invalid).
- $m/\bot \leftarrow \mathsf{Dec}_{k,\theta,G}(\widetilde{m})$ classically decodes $\widetilde{m}$ into the measurement result m of the plaintext state in the column basis of $\mathsf{H}^\theta G$. It outputs $\bot$ if $\widetilde{m}$ is an invalid string.

We can view the classical oracle $\mathsf{Dec}_{k,\theta,G}$ as a functional key for learning the plaintext measurement result of the ciphertext state in the column basis of $\mathsf{H}^\theta G$. Our notion of security ensures that an adversary with the ciphertext state and multiple functional keys learns nothing beyond what is permitted by the combined functionality of the keys. More precisely, our simulation-based definition guarantees that access to the ciphertext state and multiple functional keys is indistinguishable from access to a dummy ciphertext state and multiple simulated oracles, each conditionally revealing the corresponding measurement information of the plaintext state. That is, the security is guaranteed even if the adversary has partial decryption ability.

2.3 Obfuscation of Approximately Unitary Quantum Programs

How should ideal obfuscation be defined for quantum programs? We begin by exploring the implications of white-box access to a quantum program that implements a unitary transformation $\mathcal{U} : \rho \mapsto U\rho U^\dagger$. While a circuit that implements a unitary operator U naturally yields a circuit that implements $\mathsf{ctrl}\text{-}U$, a quantum program with auxiliary state that implements $\mathcal{U}$ does not necessarily admit implementing $\mathsf{ctrl}\text{-}U$ for any choice of U's global phase.[8] The latter situation could happen if the auxiliary state of the program becomes a different state after program execution. In such a case, attempting to use the program to implement $\mathsf{ctrl}\text{-}U$ would entangle the control qubit with the auxiliary register, thereby failing to implement $\mathsf{ctrl}\text{-}U$. Fortunately, if one considers the combined process of executing the program, applying an efficient unitary A, and then rewinding the program, the resulting auxiliary state will always be restored to its original state. Therefore, we can show that white-box access to a quantum program that implements $\mathcal{U}$ inherently allows one to perform the transformations associated[9] with U, $U^\dagger$, and $\mathsf{ctrl}\text{-}(U^\dagger A U)$ for any number of times, provided that A itself

[8] U is determined by $\mathcal{U}$, but only up to a global phase.

[9] Associated with the operator U is the transformation $\mathcal{U} : \rho \mapsto U\rho U^\dagger$.

can be efficiently implemented.[10] This motivates a tentative definition of ideal obfuscation for approximately unitary quantum programs: the obfuscated program should only grant black-box access to the unitaries U, $U^\dagger$, and $\mathsf{ctrl}\text{-}(U^\dagger AU)$. In this work, we show that such a notion of ideal obfuscation can, in fact, be realized in the classical oracle model. Hence, we define ideal obfuscation as follows.

- An ideal obfuscation scheme consists of an obfuscator $\mathsf{QObf}(1^\lambda, (\psi, Q)) \to (\widetilde{\psi}, \mathsf{F})$ transforming a quantum program (ψ, Q) into an obfuscated program specified by state $\widetilde{\psi}$ and classical oracle F, and an algorithm $\mathsf{QEval}^{\mathsf{F}}(\cdot, \widetilde{\psi})$ for executing the obfuscated program.
- $\mathsf{QEval}^{\mathsf{F}}(\cdot, \widetilde{\psi})$ approximately implements the same functionality as (ψ, Q).
- $(\widetilde{\psi}, \mathsf{F})$ can be efficiently simulated with oracle access to some unitary $\mathsf{ctrl}\text{-}(U^\dagger AU)$.[11]

Such a notion of ideal obfuscation says that any efficiently implementable unitary can be implemented in a way that reveals nothing about the program other than its ability to perform $\mathsf{ctrl}\text{-}(U^\dagger AU)$.

Our definition generalizes the classical notion of ideal obfuscation to the setting of unitary functionalities. In particular, when restricted to programs that implement (the unitary of) classical deterministic functions, our definition recovers the standard notion of classical ideal obfuscation.[12] Furthermore, the definition captures quantum features such as the global phase. Although a global phase introduced by a quantum program has no computational significance when the program is executed in isolation, this phase can become detectable when the program is used under quantum-controlled operations. Our definition implicitly requires the global phase to be obfuscated because the simulator is given oracle access only to $\mathsf{ctrl}\text{-}(U^\dagger AU)$, which is oblivious to the choice of U's global phase. As a result, circuits that implement distinct unitary operators but realize the same unitary transformation (e.g., $\mathsf{X} \circ \mathsf{Z}$ versus Y, which differ by a global phase i) would become indistinguishable under obfuscation schemes that satisfy our definition. Below, we demonstrate that every quantum program that approximately implements a unitary transformation can be ideally obfuscated in the classical oracle model when assuming a random oracle.

Our Construction. When designing obfuscation schemes, it is crucial to ensure that the same input is used consistently throughout the entire execution of the program. For classical inputs, prior works [BKNY23, BBV24] address the issue with a tokenized signature scheme, which includes a classical verification key vk and a quantum token state τ_{token} . This token can be used to produce a classical signature on a single *classical* message, but prevents signing two different messages simultaneously, thereby enforcing input consistency. We extend this approach by showing that signature tokens can also be adapted to ensure consistency of *quantum* inputs. Together with the tools introduced in the previous subsection, we can obfuscate any quantum program that approximately implements a unitary as follows.

[10] $\mathsf{ctrl}\text{-}(U^\dagger AU)$ is invariant under the choice of U's global phase.

[11] In fact, it turns out that $U, U^\dagger, \mathsf{ctrl}\text{-}(U^\dagger AU)$ can all be simulated from oracle access to $\mathsf{ctrl}\text{-}(U^\dagger AU)$ for a particular choice of A. One such choice is $A = \mathsf{SWAP}^{\otimes n}$ if U acts on n qubits (see Lemma 6.7).

[12] For unitary of the form $U : |x, y\rangle \mapsto |x, y \oplus F(x)\rangle$ for some classical function F, the unitary $\mathsf{ctrl}\text{-}(U^\dagger AU)$ can be implemented from black-box access to $\mathsf{ctrl}\text{-}U$ (because $U^\dagger = U$), which can in turn be implemented from black-box access to $\mathcal{U}$ using controlled swap gates and the efficiently preparable 1-eigenvectors of U.

- Prepare EPR pairs $(\mathsf{epr}_{\mathrm{in,pub}}, \mathsf{epr}_{\mathrm{in,priv}})$ and $(\mathsf{epr}_{\mathrm{out,priv}}, \mathsf{epr}_{\mathrm{out,pub}})$.
- Consider the extended computation that takes a classical string $\mathbb{i}$ as input, applies the Pauli correction $P_{\mathbb{i}}$ on $\mathsf{epr}_{\mathrm{in,priv}}$, computes the quantum program of interest, teleports the output through $\mathsf{epr}_{\mathrm{out,priv}}$, and outputs the teleportation result y.
- Transform the extended computation into a PLM quantum program consisting of a quantum state ψ_+ and instructions $\{(\theta_j, G_j, f_j)\}$ of the adaptive measurements. Specifically, ψ_+ contains the description ψ of the original program, the states $\mathsf{epr}_{\mathrm{in,priv}}, \mathsf{epr}_{\mathrm{out,priv}}$, and some magic states. The j-th instruction (θ_j, G_j, f_j) suggests an adaptive measurement specified by the classical function f_j and the column basis of $\mathsf{H}^{\theta_j} G_j$.
- Encode the state ψ_+ into a state $\psi_{\mathsf{auth}} = \mathsf{Auth.Enc}_k(\psi_+)$ using functional authentication.
- Generate a one-time signature token $(\mathsf{vk}, \tau_{\mathsf{token}})$ and take a random function H.
- Construct the following classical function $\mathsf{F}\,(j, \widetilde{v}, \mathbb{i}, s, \ell_1, \ldots, \ell_t)$, which enables homomorphic adaptive measurements without revealing intermediate measurement results.

  1. Compute $v \leftarrow \mathsf{Auth.Dec}_{k,\theta_j,G_j}(\widetilde{v})$. If it rejects, output $\bot$.
  2. Compute $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$. If it rejects, output $\bot$.
  3. For $i = 1, \ldots, j-1$, reconstruct r_i as follows:
     - Set $r_i \in \{0,1\}$ so that $\ell_i = H(i, r_i, \mathbb{i}, s)$.
     - If there are two or no solutions for r_i, output $\bot$.
  4. Compute $r_j = f_j^{\mathbb{i}, r_1, \ldots, r_{j-1}}(v)$.
  5. Output $\begin{cases} \text{the label } \ell_j = H(j, r_j, \mathbb{i}, s) & \text{if } j < t. \\ \text{the outcome } g(\mathbb{i}, r_1, \ldots, r_t) & \text{if } j = t. \end{cases}$

The obfuscated program consists of the authenticated quantum state ψ_{auth}, the half EPR states $\mathsf{epr}_{\mathrm{in,pub}}, \mathsf{epr}_{\mathrm{out,pub}}$, the signature token τ_{token}, and the classical oracle F. The oracle F is designed to output random labels ℓ_j instead of directly revealing the measurement outcomes r_j. To explain, the oracle F at step j would first verify and decode the authenticated quantum state in the corresponding basis to ensure that the state is not tampered with. After verifying the tokenized signature, F then tries to reconstruct prior measurement results r_i from ℓ_i. Next, F computes the outcome r_j of the function $f_j^{r_1, \ldots, r_{j-1}}$, and outputs a random label ℓ_j derived from r_j when $j < t$ is an intermediate step. When $j = t$ is the last step, F outputs $g(\mathbb{i}, r_1, \ldots, r_t)$ which represents the teleportation result of the quantum output.

Security Proof Sketch. We can first apply the simulation-based security of the functional authentication scheme, which shows that it is indistinguishable if the adversary gets a dummy authenticated state instead and the oracle is now simulated by a simulator who holds the state ψ_+ on some register V. Recall that ψ_+ contains half EPR states $\mathsf{epr}_{\mathrm{in,priv}}, \mathsf{epr}_{\mathrm{out,priv}}$ and the encoding ψ of the original quantum program. We would like to show that the simulator never reveals unnecessary information about ψ, even under malicious queries.

When $j < t$, the simulated oracle only outputs either $H(j, r_j, \mathbb{i}, s)$ or $\bot$. The simulator can apply query-recording techniques (e.g., [Zha19]) of random oracles to observe whether any information of the form $H(\cdot, \cdot, \mathbb{i}, \cdot)$ has been learned by the adversary. At any time,

the tokenized signature scheme ensures that the adversary can only learn information of the form $H(\cdot,\cdot,\mathbb{i},\cdot)$ for at most one such $\mathbb{i}$ simultaneously. This mechanism forces the adversary to use a consistent classical string $\mathbb{i}$ when it performs each evaluation. Since the simulator can extract the adversary's quantum input from $\mathbb{i}$ and $\mathsf{epr}_{\mathsf{in},\mathsf{priv}}$, the mechanism also ensures the consistency of the adversary's quantum input throughout the evaluation.

Controlled that some information of the form $H(\cdot,\cdot,\mathbb{i},\cdot)$ has been learned by the adversary, the simulator can expand the current state on V under the $\mathbb{i}$-th orthonormal basis $\{|\Phi_{\mathbb{i},r}\rangle\}_r$ of the PLM quantum program. By the design of the oracle and that $|\Phi_{\mathbb{i},r}\rangle$ yields deterministic adaptive measurement results, querying the oracle cannot affect the amplitudes of V on the basis vectors, except with negligibly small probability. Hence, the power of the adversary is essentially limited to passively gathering the outputs of the oracle.

There are three kinds of output that the oracle may produce: $\perp$, $H(\cdot,\cdot,\mathbb{i},\cdot)$, or $g(\mathbb{i},\cdot)$. The values of H do not provide any useful information because they are random, so the adversary only acquires useful information through $g(\mathbb{i},r)$. By the guarantee of the PLM quantum program and that the program approximately implements a unitary U, the simulated oracle that conditionally reveals the final output $g(\mathbb{i},r)$ can be simulated from $\mathsf{ctrl}\text{-}(U^\dagger A U)$ for some efficient $\mathsf{ctrl}\text{-}A$, showing that the construction is an ideal obfuscation. We refer the readers to Section 7.2 for a detailed security proof.

3 Preliminary

Let $[n]$ be $\{1,\cdots,n\}$. Sampling uniformly from a set S is denoted by $s \leftarrow S$. A function $f\colon \mathbb{N}\to[0,1]$ is called negligible, if for every polynomial $\mathrm{poly}(\cdot)$ and all sufficiently large n, it holds that $f(n) < |1/\mathrm{poly}(n)|$. We use $\mathrm{negl}(\cdot)$ to denote an unspecified negligible function. QPT stands for quantum polynomial time.

An n-qubit pure state $|\phi\rangle$ is a unit vector in the Hilbert space $\mathbb{C}^{2^n}$, and is identified with the density matrix $\mathsf{Mixed}\left[|\phi\rangle\right] = \phi = |\phi\rangle\langle\phi| \in \mathbb{C}^{2^n\times 2^n}$. The set of n-qubit (mixed) states, denoted $\mathcal{D}^n$, consists of positive semi-definite matrices in $\mathbb{C}^{2^n\times 2^n}$ with trace 1. Sometimes we identify a mixed state ρ with its purification (denoted $|\rho\rangle$ when we abuse notation), which is a pure state $|\phi\rangle$ such that $\mathsf{Mixed}\left[|\phi\rangle\right]$ has partial trace ρ. A quantum operation $\mathcal{F}$ is a completely-positive trace-preserving (CPTP) map from $\mathcal{D}^n$ to $\mathcal{D}^{n'}$. The quantum operation of applying an operator V refers to the map $\rho \mapsto V\rho V^\dagger$. We denote a state ρ on register R as ρ^R and a quantum operation $\mathcal{F}$ on register R as $\mathcal{F}^R$. The notation (ρ,σ) denotes a possibly entangled state (i.e., need not be a product state) on two registers. Let Tr denote the trace. The partial trace Tr_R is the unique linear map from registers R,S to S such that $\mathrm{Tr}_R(\rho^R,\sigma^S) = \mathrm{Tr}(\rho)\sigma$ for every state (ρ,σ). The trace norm of a linear operator A is defined as $\|A\|_1 := \mathrm{Tr}(\sqrt{A^\dagger A})$ and the operator norm is defined as $\|A\|_{\mathsf{op}} := \max_{|v\rangle} \frac{\|Av\|}{\|v\|}$. The operator norm of a linear map $\mathcal{F}:\mathcal{D}^n\to\mathcal{D}^{n'}$ is defined as $\|\mathcal{F}\|_{\mathsf{op}} := \max_{\rho\in\mathcal{D}^n} \|\mathcal{F}(\rho)\|_1$. For quantum operations $\mathcal{F}$ and $\mathcal{G}$ from $\mathcal{D}^n$ to $\mathcal{D}^{n'}$, their diamond distance is defined as $\|\mathcal{F}-\mathcal{G}\|_\diamond := \max_{\rho\in\mathcal{D}^{2n}} \|(\mathcal{F}\otimes\mathbb{1}_n)(\rho)-(\mathcal{G}\otimes\mathbb{1}_n)(\rho)\|_1$. Given two states ρ,σ, we write $\rho\approx_\epsilon\sigma$ if they have trace distance $\frac{1}{2}\|\rho-\sigma\|_1 \le \epsilon$.

For every classical string, we append an additional status bit that defaults to 0. When we say that a function outputs $\perp$, we mean that it returns the all-zero string (of the appropriate length) with the status bit set to 1. All functions will be defined under the convention that

if the input contains a status bit set to 1, then the function outputs $\perp$. As a result, when viewed as a unitary operator, every function has a known 1-eigenvector by taking the input part as $|\perp\rangle$ and the output part as $\frac{1}{\sqrt{2}}(|0\rangle + |\perp\rangle)$. It is well known that given a 1-eigenvector $|v\rangle$ of a unitary operator U, the controlled unitary $\mathsf{ctrl}\text{-}U := |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U$ can be efficiently computed using controlled swap gates and a single oracle query to U.

3.1 Quantum Circuits and Programs

A quantum circuit Q is described by a sequence of unitary gates and standard-basis measurements followed by selecting the output bits or qubits. The width of Q is the number of qubits it acts on, and the size of Q is the number of gates and measurements it has. It is well-known that any quantum computation can be implemented by a quantum circuit whose unitary gates are chosen from the gate set $\mathscr{C}_2 \cup \{T\}$ defined as follows.

- **Pauli Group.** The single-qubit Pauli group $\mathscr{P}_1$ consists of the group generated by Pauli gates $\mathsf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $\mathsf{Y} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, $\mathsf{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. The n-qubit Pauli group $\mathscr{P}_n$ is the n-fold tensor product of $\mathscr{P}_1$.
- **Clifford Group.** The n-qubit Clifford group $\mathscr{C}_n$ consists of unitaries C such that $C\mathscr{P}_n C^\dagger = \mathscr{P}_n$. The group is generated by Hadamard $\mathsf{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$, Phase $\mathsf{S} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$, and $\mathsf{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ gates, and contains $\mathsf{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ gate.
- T **gate.** The T gate is defined as $\mathsf{T} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$.

A quantum program (ψ, Q) is specified by a quantum circuit Q and an auxiliary state ψ. The size of the quantum program is the size of Q plus the width of Q. The input length of the quantum program is the width of Q minus the length of ψ. Running the quantum program on an input state ρ is to run the quantum circuit Q on the state (ρ, ψ). It is well-known that every quantum program (ψ, Q) can be efficiently transformed into an equivalent quantum program $(\psi', Q^{\mathsf{univ}})$ where Q^{univ} is a universal quantum circuit that depends only on the size of the original quantum program and the output length, whereas the length of ψ' depends only on the size of the original quantum program. Without loss of generality, we will consider quantum programs of the form $(\psi', Q^{\mathsf{univ}})$ throughout this paper.

3.2 Quantum Teleportation

Quantum teleportation facilitates the transmission of quantum information with classical communication and pre-shared quantum states. Define the state EPR as $\frac{1}{\sqrt{2}}\sum_{b\in\{0,1\}} |b_j, b_j\rangle$. An EPR pair of length n is a state $(\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n}$ where their i-th qubits forms an EPR state for every $i \in [n]$. For $(z, x) \in \{0,1\}^{2n}$, we define $P_{(z,x)} := \mathsf{X}^x\mathsf{Z}^z \in \mathscr{P}_n$.

Definition 3.1 (Quantum Teleportation)**.** Let $(\mathsf{epr}_L, \mathsf{epr}_R)$ be an EPR pair of length n pre-shared between a sender holding input $\phi \in \mathcal{D}^n$ and a receiver. Quantum teleportation consists of two algorithms.

- $\mathsf{TP.Send}(\phi^M, \mathsf{epr}_L{}^L)$: applies $\mathsf{H}^M\mathsf{CNOT}^{M,L}$ on $(\phi^M, \mathsf{epr}_L{}^L)$, performs a standard-basis measurement on registers (M, L), and outputs the measurement outcome (z, x).
- $\mathsf{TP.Recv}((z, x), \mathsf{epr}_R{}^R)$: applies $P_{(z,x)}{}^\dagger$ to $\mathsf{epr}_R{}^R$ and outputs register R.

We also define the teleportation unitary operator $\mathsf{TP}^{M,L} := \mathsf{H}^M\mathsf{CNOT}^{M,L}$. The following lemma characterizes the correctness of quantum teleportation.

Lemma 3.2. *Let $|\phi, \tau\rangle$ be a purified state separable from $(\mathsf{epr}_L, \mathsf{epr}_R)$. Then*

$$\mathsf{TP}^{M,L}\,|\phi^M, \mathsf{epr}_L{}^L, \mathsf{epr}_R{}^R, \tau^N\rangle = \frac{1}{2^n} \sum_{z,x\in\{0,1\}^n} |z^M, x^L\rangle \otimes P_{(z,x)}{}^R\,|\phi^R, \tau^N\rangle$$

3.3 Purified Random Oracle

A random oracle is the classical oracle of a random function. One can perfectly simulate a random oracle mapping $\mathcal{X}$ to $\{0,1\}^\kappa$ by purifying the randomness of the random function. To do so, we introduce a database register $D := \{D_x\}_{x\in\mathcal{X}}$ that stores the truth table of the random function and initialized it with the uniform superposition state

$$|\mathcal{L}_\emptyset\rangle^D := \bigotimes_{x\in\mathcal{X}} |+^\kappa\rangle^{D_x}$$

Querying the random oracle with classical input x and output register Y can be carried out by applying the unitary

$$\mathsf{WriteH}(x)^{Y,D} := \mathsf{CNOT}^{D_x,Y}.$$

For comparison, we also define

$$\mathsf{Write}(x)^Y := \mathsf{X}^{xY}$$

which writes the value x to register Y in the standard basis. For a family of unitary operators $U(x)$ parameterized by a classical string x, we denote

$$U(A)^B := \sum_x |x\rangle\langle x|^A \otimes U(x)^B.$$

With these notations, the random oracle can be simulated as follows:

- Initialize a database register $D \leftarrow |\mathcal{L}_\emptyset\rangle$.
- Apply $\mathsf{WriteH}(X)^{Y,D}$ for each query, given input register X and output register Y.

Define the projector $\mathsf{EqualH}(x)^{Y,D} := \mathsf{WriteH}(x)^{Y,D}|0^\kappa\rangle\langle 0^\kappa|^Y\mathsf{WriteH}(x)^{Y,D}$ that indicates if the value on Y matches the x-th entry of the database in the standard basis. A straightforward calculation shows $\mathsf{EqualH}(x)^{Y,D} = \sum_{y\in\{0,1\}^\kappa} |y\rangle\langle y|^Y \otimes |y\rangle\langle y|^{D_x}$ and that $\mathsf{EqualH}(x)$ commutes with $\mathsf{EqualH}(x')$ for every $x \neq x'$. Given a non-empty subset $S \subseteq \mathcal{X}$, we define the following projectors on D that informs if certain function values have been provided to other parties.

- $\mathsf{Has}_S{}^D := I - \bigotimes_{x\in S} |+^\kappa\rangle\langle+^\kappa|^{D_x}$ indicates that some entries in S have been given out.
- $\mathsf{Only}_S{}^D := \bigotimes_{x\notin S} |+^\kappa\rangle\langle+^\kappa|^{D_x} - |\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset|^D$ indicates that some entries in S, and only entries in S, have been given out.

Lemma 3.3. $\left\|\mathsf{EqualH}(x)^{L,D}(I-\mathsf{Has}_{\{x\}})^D\right\|_{\mathsf{op}} = 2^{-\kappa/2}$ *for every* $x\in\mathcal{X}$.

Proof. This follows from

$$\begin{aligned}
&\left\|\mathsf{EqualH}(x)^{L,D}(I-\mathsf{Has}_{\{x\}})^D\right\|_{\mathsf{op}}^2\\
=&\left\|(I-\mathsf{Has}_{\{x\}})^D\mathsf{EqualH}(x)^{L,D}(I-\mathsf{Has}_{\{x\}})^D\right\|_{\mathsf{op}}\\
=&\left\||+^\kappa\rangle\langle+^\kappa|^{D_x}\left(\sum_{y\in\{0,1\}^\kappa}|y\rangle\langle y|\otimes|y\rangle\langle y|\right)^{L,D_x}|+^\kappa\rangle\langle+^\kappa|^{D_x}\right\|_{\mathsf{op}}\\
=&\left\|\frac{1}{2^\kappa}\left(\sum_{y\in\{0,1\}^\kappa}|y\rangle\langle y|\otimes|+^\kappa\rangle\langle+^\kappa|\right)\right\|_{\mathsf{op}} = \left\|\frac{1}{2^\kappa}\left(I\otimes|+^\kappa\rangle\langle+^\kappa|\right)\right\|_{\mathsf{op}} = \frac{1}{2^\kappa}.
\end{aligned}$$

□

3.4 Tokenized Signature

Definition 3.4. A *tokenized signature scheme* Token consists of algorithms $(\mathsf{Gen},\mathsf{Sign},\mathsf{Ver})$ with the following syntax:

- $\mathsf{Gen}(1^\lambda,1^n)\to(\mathsf{vk},\tau_{\mathsf{token}})$ takes as input the security parameter λ and the message length n, and outputs a classical verification key vk and a quantum signing key τ_{token}.
- $\mathsf{Sign}(m,\tau_{\mathsf{token}})\to s$ takes as input a message $m\in\{0,1\}^n$ and the quantum signing key τ_{token}, and outputs a classical signature s.
- $\mathsf{Ver}_{\mathsf{vk}}(m,s)\to\top/\bot$ takes as input the verification key vk, a message m, and a signature s, and outputs $\top$ (accept) or $\bot$ (reject).

A tokenized signature scheme should satisfy correctness and unforgeability as follows.

- Correctness: For any message $r\in\{0,1\}^n$,

$$\Pr\left[\mathsf{Ver}_{\mathsf{vk}}(m,s)=\top \;\middle|\; \begin{array}{l}(\mathsf{vk},\tau_{\mathsf{token}})\leftarrow\mathsf{Gen}(1^\lambda,1^n)\\ s\leftarrow\mathsf{Sign}(m,\tau_{\mathsf{token}})\end{array}\right]=1.$$

- Unforgeability: For any $n=\mathrm{poly}(\lambda)$ and polynomial-query quantum adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{c}m_0\neq m_1\\ \wedge\,\mathsf{Ver}_{\mathsf{vk}}(m_0,s_0)=\top\\ \wedge\,\mathsf{Ver}_{\mathsf{vk}}(m_1,s_1)=\top\end{array}\;\middle|\;\begin{array}{l}(\mathsf{vk},\tau_{\mathsf{token}})\leftarrow\mathsf{Gen}(1^\lambda,1^n)\\ (m_0,m_1,s_0,s_1)\leftarrow\mathcal{A}^{\mathsf{Ver}_{\mathsf{vk}}(\cdot,\cdot)}(\tau_{\mathsf{token}})\end{array}\right]=2^{-\Omega(\lambda)}$$

The definition requires that honestly generated signatures will always be accepted by the verification algorithm. Moreover, it asserts that any adversary with polynomially-many quantum queries to the verification oracle cannot produce valid signatures for two distinct messages simultaneously, except with exponentially small probability.

Theorem 3.5 ([BDS23])**.** *There exists a tokenized signature scheme* Token *unconditionally with signature length* $O(\lambda n)$*.*

3.5 Post-quantum PRF

A pseudorandom function is a family of efficiently computable functions that are computationally indistinguishable to truly random functions under query access. Here, we consider post-quantum pseudorandom functions, where indistinguishability holds even for adversaries who can query the function in superposition.

Definition 3.6 (Post-quantum PRF)**.** A post-quantum PRF is a family $\{\mathrm{PRF}_k : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ of efficiently computable functions, where the key space $\mathcal{K}$, the domain $\mathcal{X}$, and the range $\mathcal{Y}$ implicitly depend on the security parameter λ. We require computational indistinguishability between PRF and the truly random function, in the sense that the following should hold for every QPT adversary $\mathcal{A}$ with quantum queries to the function.

$$\left|\Pr_{k\leftarrow\mathcal{K}}[\mathcal{A}^{\mathrm{PRF}_k}(\cdot)=1] - \Pr_{F\leftarrow\mathcal{Y}^{\mathcal{X}}}[\mathcal{A}^{F}(\cdot)=1]\right| \leq \mathrm{negl}(\lambda)$$

Theorem 3.7 ([Zha21a])**.** *Post-quantum PRF can be constructed from post-quantum one-way functions.*

3.6 Useful Lemmas

Lemma 3.8 ([Wat18])**.** *Let* $\Phi_0, \Phi_1 : \mathcal{D}^n \to \mathcal{D}^m$ *be quantum operations.*

- *If* Φ_1 *is a unitary transformation, then* $\|\Phi_0 - \Phi_1\|_\diamond \leq \sqrt{2\max_{\rho\in\mathcal{D}^n}\|\Phi_0(\rho)-\Phi_1(\rho)\|_1}$*.*
- *If* Φ_0, Φ_1 *are both isometries, then* $\|\Phi_0 - \Phi_1\|_\diamond = \|\Phi_0 - \Phi_1\|_{\mathsf{op}}$*.*

Theorem 3.9 ([KSW08])**.** *Let* $\Phi_0, \Phi_1 : \mathcal{D}^n \to \mathcal{D}^m$ *be quantum operations and* $U_0, U_1 : \mathbb{C}^n \to \mathbb{C}^m \otimes \mathbb{C}^k$ *be isometries such that* $\Phi_i(\rho) \equiv \mathrm{Tr}_2(U_i\rho U_i^\dagger)$ *for* $i = 0, 1$*. Then we have*

$$\inf_{\text{unitary } V} \|U_0 - (I\otimes V)U_1\|_{\mathsf{op}} \leq \sqrt{\|\Phi_0 - \Phi_1\|_\diamond}$$

Lemma 3.10. *Let* $U, W : \mathbb{C}^n \to \mathbb{C}^m$ *be isometries. Then* $\left\|U(\cdot)U^\dagger - W(\cdot)W^\dagger\right\|_\diamond \leq 2\|U-W\|_{\mathsf{op}}$*.*

Proof. By Lemma 3.8, the triangle inequality, and that operator norm is sub-multiplicative,

$$\left\|U(\cdot)U^\dagger - W(\cdot)W^\dagger\right\|_\diamond = \left\|U(\cdot)U^\dagger - W(\cdot)W^\dagger\right\|_{\mathsf{op}} = \max_{\text{state }\rho}\left\|U\rho U^\dagger - W\rho W^\dagger\right\|_1$$
$$\leq \max_{\text{state }\rho}\left(\left\|(U-W)\rho U^\dagger\right\|_1 + \left\|(W\rho(U-W)^\dagger\right\|_1\right) \leq 2\|U-W\|_{\mathsf{op}}$$

□

Lemma 3.11 (Gentle Measurement, [Win99]). *Let ρ be a quantum state, and $\{\Pi, I-\Pi\}$ be a projective measurement such that $\mathrm{Tr}(\Pi\rho) \leq \epsilon$. Let $\rho' = \frac{(I-\Pi)\rho(I-\Pi)}{\mathrm{Tr}((I-\Pi)\rho)}$ be the state after applying $\{\Pi, I-\Pi\}$ on ρ and post-selecting on obtaining the second outcome. Then the states ρ and ρ' are within trace distance $2\sqrt{\epsilon}$.*

The following lemma allows us to switch between oracles that differ on hard-to-find inputs without perturbing the state too much.

Lemma 3.12 (Oracle Hybrid Lemma). *Let $\mathcal{K}$ be a distribution. For all possible $k \leftarrow \mathcal{K}$, let $|\psi_k\rangle$ be a quantum state, Π_k be a projector, and O_k and O'_k be unitary operators such that $O_k(I-\Pi_k) = O'_k(I-\Pi_k)$. Suppose for every q-query algorithm $\mathcal{A}$, it holds that $\mathbb{E}_{k\leftarrow\mathcal{K}}\left[\left\|\Pi_k \mathcal{A}^{O_k}|\psi_k\rangle\right\|^2\right] \leq \epsilon$. Then for every q-query algorithm $\mathcal{A}$, we have*

$$\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\mathsf{Mixed}\left[\mathcal{A}^{O_k}|\psi_k\rangle\right]\right] \mathop{\approx}_{O(q\sqrt{\epsilon})} \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\mathsf{Mixed}\left[\mathcal{A}^{O'_k}|\psi_k\rangle\right]\right]$$

Proof. For each possible $k \leftarrow \mathcal{K}$ and $i = 0, 1, \ldots, q$, we define the following states

- $|\psi_{k,i}\rangle$ is obtained by applying $\mathcal{A}$ to $|\psi_k\rangle$ where the first i queries are made to O_k and the rest of the queries are made to O'_k.
- $|\psi'_{k,i}\rangle$ is obtained similarly to $|\psi_{k,i}\rangle$, except that we apply the measurement $\{\Pi_k, I-\Pi_k\}$ right before the i-th oracle query and post-selecting on obtaining the second outcome.
- $|\varphi_{k,i}\rangle$ is obtained by applying $\mathcal{A}^{O_k}$ to $|\psi_k\rangle$ up to and not including the i-th query.

Set $\epsilon_{k,i} := \|\Pi_k|\varphi_{k,i}\rangle\|^2$. By assumption, we have $\mathbb{E}_{k\leftarrow\mathcal{K}}[\epsilon_{k,i}] \leq \epsilon$. Applying the triangle inequality, the gentle measurement lemma, and the Cauchy-Schwartz inequality, we obtain

$$\begin{aligned}
&\frac{1}{2}\mathrm{Tr}\left|\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[|\psi_{k,i}\rangle\right] - \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[|\psi'_{k,i}\rangle\right]\right| \\
\leq &\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\frac{1}{2}\mathrm{Tr}\left|\mathsf{Mixed}\left[|\psi_{k,i}\rangle\right] - \mathsf{Mixed}\left[|\psi'_{k,i}\rangle\right]\right|\right] \\
\leq &\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[2\sqrt{\epsilon_{k,i}}\right] \leq 2\sqrt{\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}[\epsilon_{k,i}]} \leq 2\sqrt{\epsilon}
\end{aligned}$$

By assumption, right after post-selecting on the measurement result of $I-\Pi_k$, applying O_k has the same effect as applying O'_k. Hence, the state $|\psi'_{k,i}\rangle$ can also be obtained similarly to $|\psi_{k,i-1}\rangle$, except that we apply the measurement $\{\Pi_k, I-\Pi_k\}$ right before the i-th oracle query and post-selecting on obtaining the second outcome. By a similar argument, we have

$$\frac{1}{2}\mathrm{Tr}\left|\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[|\psi_{k,i-1}\rangle\right] - \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[|\psi'_{k,i}\rangle\right]\right| \leq 2\sqrt{\epsilon}$$

Putting together, we have

$$\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[\mathcal{A}^{O_k}|\psi_k\rangle\right] = \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[|\psi_{k,q}\rangle\right] \mathop{\approx}_{O(q\sqrt{\epsilon})} \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[|\psi_{k,0}\rangle\right] = \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\mathsf{Mixed}\left[\mathcal{A}^{O'_k}|\psi_k\rangle\right]$$

□

4 Functional Authentication

In this section, we introduce functional authentication, which enhances standard quantum authentication by enabling fine-grained access to measurement results of the state. In particular, the functional authentication scheme supports classical decoding oracles, which are interpreted as functional keys. The security of functional authentication will ensure that the adversaries gain no additional advantage beyond the combined functionality allowed by the given functional keys. Below, we call a unitary gate G linear if it is composed of CNOT gates. We sometimes use the expression $\mathsf{Write}(A)^B := \sum_a |a\rangle\langle a|^A \otimes \mathsf{X}^{aB}$ for CNOT gates.

4.1 Definition

Definition 4.1 (Functional Authentication)**.** A functional authentication scheme consists of algorithms with the following syntax:

- $\mathsf{KeyGen}(1^\lambda, 1^n) \to k$ takes as input the security parameter λ and the qubit count $n = \mathrm{poly}(\lambda)$, and outputs a classical authentication key k.
- $\mathsf{Enc}_k(\rho) \to \sigma$ takes as input an n-qubit state ρ and the key k, and outputs a state $\sigma = (\sigma_1, \ldots, \sigma_n)$ where each σ_i has $p(\lambda)$ qubits. This algorithm is an isometry parameterized by the authentication key k.
- $\mathsf{Eval}_\theta \to \widetilde{\theta}$ and $\mathsf{Eval}_G \to \widetilde{G}$ are classical algorithms that output the descriptions of homomorphic gates. Here, $\theta \in \{0,1\}^n$, $\widetilde{\theta} \in \{0,1\}^{np(\lambda)}$ stand for measurement bases, and $G, \widetilde{G}$ are linear gates on $n, np(\lambda)$ qubits respectively.
- $\mathsf{Dec}_{k,\theta,G}(c) \to m/\bot$ is a classical algorithm that takes a string c, an authentication key k, measurement bases $\theta \in \{0,1\}^n$ and a linear gate G on n qubits, and outputs either a decoded outcome m or $\bot$ (failure).
- $\mathsf{Ver}_{k,\theta,G}(c) \to \top/\bot$ is a classical algorithm that verifies the string c in a manner similar to the decoding algorithm, and outputs either $\top$ if $m \neq \bot$, or $\bot$ if $m = \bot$.

For a classical function $f : \{0,1\}^n \to \{0,1\}^*$, a string $\theta \in \{0,1\}^n$, and a linear gate G on n-qubits, we define the measurement

$$\mathcal{M}[f, \theta, G] := \left\{ G^\dagger \mathsf{H}^\theta \left(\sum_{x : f(x) = y} |x\rangle\langle x| \right) \mathsf{H}^\theta G \right\}_y$$

Functional authentication scheme needs to satisfy the following properties:

- Correctness: For any classical function $f : \{0,1\}^n \to \{0,1\}^*$, bases $\theta \in \{0,1\}^n$, linear gate G on n qubits, and all possible $k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n)$, we have

  $$\mathcal{M}\left[(f \circ \mathsf{Dec}_{k,\theta,G}), \widetilde{\theta}, \widetilde{G} \right] \circ \mathsf{Enc}_k = \mathsf{Enc}_k \circ \mathcal{M}[f, \theta, G].$$

  under the convention $f(\bot) := \bot$. Moreover, for every n-qubit state ρ, we have

  $$\Pr\left[\top \leftarrow \mathcal{M}\left[\mathsf{Ver}_{k,\theta,G}, \widetilde{\theta}, \widetilde{G} \right] (\mathsf{Enc}_k(\rho)) \right] = 1.$$

- Functional Security: Let $f_1, \ldots, f_t : \{0,1\}^n \to \{0,1\}^{n'}$ be any functions, $\theta_1, \ldots, \theta_t \in \{0,1\}^n$ be any measurement bases, and $G_1, \ldots, G_t$ be any linear gates on n qubits. Define corresponding unitary operators $\mathsf{fDec}_k(j, \widetilde{v}, Y)$ and $\mathsf{fVer}_k(j, \widetilde{v}, Y, V)$ as follows. The "If-Then" expressions should be interpreted as controlled operators, whereas the measurements are to be applied coherently.

  – $\mathsf{fDec}_k(j, \widetilde{v}, Y)$:

  1. $v \leftarrow \mathsf{Dec}_{k,\theta_j,G_j}(\widetilde{v})$
  2. If $v = \perp$, apply $\mathsf{Write}(\perp)^Y$
  3. Otherwise, apply $\mathsf{Write}(f_j(v))^Y$

  – $\mathsf{fVer}_k(j, \widetilde{v}, Y, V)$:

  1. $b \leftarrow \mathsf{Ver}_{k,\theta_j,G_j}(\widetilde{v})$
  2. If $b = \perp$, apply $\mathsf{Write}(\perp)^Y$
  3. Otherwise, apply $\mathsf{Write}\left(\mathcal{M}\left[f_j, \theta_j, G_j\right](V)\right)^Y$

  Let $t, n' = \mathrm{poly}(\lambda)$. Then for every $\mathrm{poly}(\lambda)$-query adversary $\mathcal{A}$ and state $|\psi\rangle^{V,W}$,

$$\left\{ (\widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \mathcal{A}^{\mathsf{fDec}_k(\cdot,\cdot,\cdot)}(\widetilde{V}, W) \end{array} \right\} \underset{2^{-\Omega(\lambda)}}{\approx} \left\{ (\widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(0^n) \\ \widetilde{V}, W \leftarrow \mathcal{A}^{\mathsf{fVer}_k(\cdot,\cdot,\cdot,V)}(\widetilde{V}, W) \end{array} \right\}$$

Our notion of functional security provides strong guarantees by ensuring that the authenticated quantum state remains secure under a simulation-based definition, even in the presence of both decryption and verification oracles. While our scheme shares syntactic similarities with the quantum authentication scheme of [BBV24], their security analysis is shown secure only under a property-based definition in the presence of verification oracles. As a result, their scheme falls short of achieving the goals of functional authentication. In contrast, our definition of functional security captures dynamic adversarial behaviors beyond the static settings considered in their work, thereby enabling a wider range of applications.

Theorem 4.2 ([BBV24])**.** *There exists a quantum authentication scheme with the following properties:*

- *Syntax: Same as functional authentication.*
- *Correctness: Same as functional authentication.*
- *Soundness: For every classical function $f : \{0,1\}^n \to \{0,1\}^*$, bases $\theta \in \{0,1\}^n$, linear gate G, and polynomial-query adversary $\mathcal{A}$, there exists an $\epsilon(\lambda) \in [0,1]$ such that for every n-qubit state ρ,*

$$\left\{ y : \begin{array}{c} k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ \sigma \leftarrow \mathcal{A}^{\mathsf{Ver}_{k,\cdot,\cdot}(\cdot)}\left(\mathsf{Enc}_k(\rho)\right) \\ y \leftarrow \mathcal{M}\left[\left(f \circ \mathsf{Dec}_{k,\theta,G}\right), \widetilde{\theta}, \widetilde{G}\right](\sigma) \end{array} \right\} \underset{2^{-\Omega(\lambda)}}{\approx} \begin{array}{c} (1-\epsilon(\lambda))\left\{y : y \leftarrow \mathcal{M}\left[f, \theta, G\right](\rho)\right\} \\ +\epsilon(\lambda)\{\perp\} \end{array}.$$

- *Privacy: For every quantum state ρ with $n = \mathrm{poly}(\lambda)$ qubits and every $\mathrm{poly}(\lambda)$-query quantum adversary $\mathcal{A}$, we have*

$$\mathop{\mathbb{E}}_{k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n)} \left[\mathcal{A}^{\mathsf{Ver}_{k,\cdot,\cdot}(\cdot)}(\mathsf{Enc}_k(\rho))\right] \underset{2^{-\Omega(\lambda)}}{\approx} \mathop{\mathbb{E}}_{k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n)} \left[\mathcal{A}^{\mathsf{Ver}_{k,\cdot,\cdot}(\cdot)}(\mathsf{Enc}_k(0^n))\right].$$

4.2 Construction and Security Proof

Interestingly, we show that the construction of the quantum authentication scheme proposed in [BBV24] can, in fact, be an instance of our functional authentication. Although the authors of [BBV24] established security under a weaker notion, our analysis in this section demonstrates that their construction indeed satisfies the stronger definition of functional authentication (Definition 4.1).

For clarity, before presenting our security analysis, we briefly recall their construction, which we refer to as the coset authentication code. For a linear subspace $S \subset \mathbb{F}_2^d$ and a vector $v \in \mathbb{F}_2^d$, the coset state $|S+v\rangle$ is defined as

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

Construction 1 ([BBV24]) Coset Authentication Code

- $\mathsf{KeyGen}(1^\lambda, 1^n)$: Sample a uniformly random subspace $S \subset \mathbb{F}_2^{2\lambda+1}$ of dimension λ. Sample a uniformly random vector $\Delta \in \mathbb{F}_2^{2\lambda+1} \setminus S$, and for each $i \in [n]$, sample uniformly random vectors $x_i, z_i \in \mathbb{F}_2^{2\lambda+1}$. The output is the authentication key $k := (S, \Delta, x, z)$.
- $\mathsf{Enc}_k(\rho)$: First applies the isometry $\sum_{b\in\{0,1\}} \alpha_b |b\rangle \mapsto \sum_{b\in\{0,1\}} \alpha_b |S + b\Delta\rangle$ to each qubit of ρ, and then applies the quantum one-time pad $\mathsf{X}^x\mathsf{Z}^z$.
- Eval_θ and Eval_G: For $\theta = (\theta_1, \cdots, \theta_n)$, set $\widetilde{\theta} = (\theta_1^{p(\lambda)} || \cdots || \theta_n^{p(\lambda)})$. Set $\widetilde{G} = G^{\otimes n}$.
- $\mathsf{Dec}_{k,\theta,G}(c)$: Parse $k = (S, \Delta, x, z)$. Let $S_\Delta = S \cup (S+\Delta)$, $\widehat{S} = (S_\Delta)^\perp$, $\widehat{\Delta} \in S^\perp \setminus (S_\Delta)^\perp$, and $\widehat{S}_{\widehat{\Delta}} = \widehat{S} \cup (\widehat{S} + \widehat{\Delta})$. Let (z_G, x_G) be the result of starting with (z, x) and applying the following update rule sequentially for each CNOT gate (say acting on qubits i, j) in G: $((z_i, x_i), (z_j, x_j)) \mapsto ((z_i \oplus z_j, x_i), (z_j, x_i \oplus x_j))$. For $i \in [n]$, compute

$$m_i = \begin{cases} 0 & \text{if } (\theta_i = 0 \text{ and } c_i \in S + x_{G,i}) \text{ or } (\theta_i = 1 \text{ and } c_i \in \widehat{S} + z_{G,i}) \\ 1 & \text{if } (\theta_i = 0 \text{ and } c_i \in S + \Delta + x_{G,i}) \text{ or } (\theta_i = 1 \text{ and } c_i \in \widehat{S} + \widehat{\Delta} + z_{G,i}) \\ \perp & \text{otherwise} \end{cases}$$

  If any $m_i = \perp$, then output $\perp$. Otherwise output $m = (m_1, \ldots, m_n)$.
- $\mathsf{Ver}_{k,\theta,G}(c)$: For $i \in [n]$, check whether $c_i \in S_\Delta + x_{G,i}$ whenever $\theta_i = 0$ and $c_i \in \widehat{S}_{\widehat{\Delta}} + z_{G,i}$ whenever $\theta_i = 1$. If all checks pass, output $\top$. Otherwise, output $\perp$.

Theorem 4.3. *Construction 1 is a functional authentication scheme for a regime $\lambda \geq \Omega(n)$.*

By Theorem 4.2, we are left with proving that construction 1 satisfies functional security (Definition 4.1). Consider an alternate key generation algorithm KeyGen' that, in addition to generating the key $k = (S, \Delta, x, z)$ as in KeyGen, also samples larger subspaces R and $\widehat{R}$, and a modified verification algorithm Ver' that checks memberships in cosets of R and $\widehat{R}$.

- $\mathsf{KeyGen}'(1^\lambda, 1^n)$:
  - Sample $k = (S, \Delta, x, z) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n)$.
  - Sample a random subspace $R \supset S_\Delta = S \cup (S + \Delta)$ such that $\dim(R) = \frac{3\lambda}{2} + 1$.
  - Sample a random subspace $\widehat{R} \supset \widehat{S}_{\widehat{\Delta}} = S^\perp$ such that $\dim(\widehat{R}) = \frac{3\lambda}{2} + 1$.
  - Sample a random $x^+ \leftarrow x + R^n$ and a random $z^+ \leftarrow z + \widehat{R}^n$.
  - Output (k, k^+) where we define the extended key $k^+ := (R, \widehat{R}, x^+, z^+)$.
- $\mathsf{Ver}'_{k^+,\theta,G}(c)$: For $i \in [n]$, check whether $c_i - x^+_{G,i} \in R$ whenever $\theta_i = 0$ and $c_i - z^+_{G,i} \in \widehat{R}$ whenever $\theta_i = 1$. If all checks pass, output $\top$. Otherwise, output $\bot$.

We show that oracle access to Ver_k is indistinguishable from oracle access to Ver'_{k^+}.

Lemma 4.4. *For every* $\mathrm{poly}(\lambda)$*-query adversary* $\mathcal{A}$ *and quantum state* $|\psi\rangle^{V,W}$,

$$\left\{ (k, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \mathcal{A}^{\mathsf{Ver}_k}(\widetilde{V}, W) \end{array} \right\} \approx_{2^{-\Omega(\lambda)}} \left\{ (k, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \mathcal{A}^{\mathsf{Ver}'_{k^+}}(\widetilde{V}, W) \end{array} \right\}$$

Proof. To highlight the difference, Ver_k checks memberships in cosets $x_{G,i} + S_\Delta$ and $z_{G,i} + \widehat{S}_{\widehat{\Delta}}$ whereas Ver'_{k^+} checks memberships in larger cosets $x_{G,i} + R$ and $z_{G,i} + \widehat{R}$. For every $(\theta, G, \widetilde{v})$ output by a polynomial-query adversary on the left-hand side, the two verification oracles behave differently when either $\widetilde{v}_i - x_{G,i} \in R \setminus S_\Delta$ or $\widetilde{v}_i - z^+_{G,i} \in \widehat{R} \setminus \widehat{S}_{\widehat{\Delta}}$ for some $i \in [n]$. Since the adversary on the left-hand side has no information about $(R, \widehat{R})$, the first type of event occurs with probability at most

$$\Pr_R \left[\widetilde{v}_i - x^+_{G,i} \in R \setminus S_\Delta \right] = \frac{|R \setminus S_\Delta|}{|\mathbb{F}_2^{2\lambda+1} \setminus S_\Delta|} = \frac{2^{(3\lambda/2)+1} - 2^{\lambda+1}}{2^{2\lambda+1} - 2^{\lambda+1}} = 2^{-\Omega(\lambda)}.$$

Similarly, the second type of event occurs with probability at most $2^{-\Omega(\lambda)}$. By a union bound, the two verification oracles behave differently on $(\theta, G, \widetilde{v})$ with probability at most $2^{-\Omega(\lambda)}$. The lemma now follows by applying the oracle hybrid argument (Lemma 3.12). □

An additional feature of Construction 1 is that homomorphic evaluation of Pauli gates can be performed solely via key updates.

Lemma 4.5. *For every* $k = (S\Delta, x, z) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n)$ *and every Pauli* $P = \mathsf{X}^u\mathsf{Z}^v \in \mathscr{P}_n$*, we define the key* $k_P := (S, \Delta, x', z')$ *where* $x'_i = x_i \oplus u_i\Delta$ *and* $z'_i = z_i \oplus v_i\widehat{\Delta}$ *for every* $i \in [n]$*. Then* $\mathsf{Ver}_{k_P,\cdot,\cdot}(\cdot) = \mathsf{Ver}_{k,\cdot,\cdot}(\cdot)$ *and* $\mathsf{Enc}_{k_P} = \mathsf{Enc}_k \circ P$ *up to a global phase.*

Proof. Since $x'_i - x_i = u_i\Delta \in S_\Delta$ for every $i \in [n]$, we have $x'_{G,i} - x_{G,i} \in S_\Delta$ and similarly we have $z'_{G,i} - z_{G,i} \in \widehat{S}_{\widehat{\Delta}}$ for every $i \in [n]$. Hence, $\mathsf{Ver}_{k_P,\cdot,\cdot}(\cdot)$ and $\mathsf{Ver}_{k,\cdot,\cdot}(\cdot)$ computes the same

function because they verify memberships of the same cosets. Next, for every $w \in \{0,1\}^n$,

$$\begin{aligned}
\mathsf{Enc}_{k_P} |w\rangle &= \bigotimes_{i=1}^{n} \mathsf{X}^{x_i \oplus u_i \Delta} \mathsf{Z}^{z_i \oplus v_i \widehat{\Delta}} |S + w_i \Delta\rangle = \bigotimes_{i=1}^{n} \mathsf{X}^{x_i \oplus u_i \Delta} \mathsf{Z}^{z_i \oplus v_i \widehat{\Delta}} \mathsf{X}^{w_i \Delta} |S\rangle \\
&= \bigotimes_{i=1}^{n} (-1)^{u_i \langle \Delta, z_i \rangle + v_i w_i \langle \widehat{\Delta}, \Delta \rangle} \mathsf{X}^{x_i} \mathsf{Z}^{z_i} \mathsf{X}^{(u_i \oplus w_i)\Delta} \mathsf{Z}^{v_i \widehat{\Delta}} |S\rangle \\
&= (-1)^{\sum_i u_i \langle \Delta, z_i \rangle} (-1)^{\langle v, w \rangle} \bigotimes_{i=1}^{n} \mathsf{X}^{x_i} \mathsf{Z}^{z_i} |S + (u_i \oplus w_i)\Delta\rangle \\
&= (-1)^{\sum_i u_i \langle \Delta, z_i \rangle} (-1)^{\langle v, w \rangle} \mathsf{Enc}_k |u \oplus w\rangle \\
&= (-1)^{\sum_i u_i \langle \Delta, z_i \rangle} \mathsf{Enc}_k \, \mathsf{X}^u \mathsf{Z}^v |w\rangle = (-1)^{\sum_i u_i \langle \Delta, z_i \rangle} \mathsf{Enc}_k \circ P \, |w\rangle
\end{aligned}$$

where the second line uses the anti-commutativity of X and Z, and the third line follows because we have chosen $\Delta, \widehat{\Delta}$ such that $\langle \widehat{\Delta}, \Delta \rangle = 1$ and $\widehat{\Delta} \in S^\perp$. Hence, we observe that Enc_{k_P} and $\mathsf{Enc}_k \circ P$ differs only by a global phase $(-1)^{\sum_i u_i \langle \Delta, z_i \rangle}$ independent of w. □

In the next lemma, we focus on the case $t = 1$ and $n' = 1$ (i.e., f outputs 1 bit), therefore omitting reference to the index j in fDec and fVer.

Lemma 4.6. *The following holds for a regime $\lambda \geq \Omega(n)$. For every quantum adversary $\mathcal{A}$ and quantum state $|\psi\rangle^{V,W}$ where we parse $W = (Y, Z)$,*

$$\left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ \widetilde{V}, W, V \leftarrow \mathcal{A}^{\mathsf{Ver}_k}(\widetilde{V}, W, V) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \\ \widetilde{V}, Y \leftarrow \mathsf{fDec}_{k_P}(\widetilde{V}, Y) \end{array} \right\} \approx_{2^{-\Omega(\lambda)}} \left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ \widetilde{V}, W, V \leftarrow \mathcal{A}^{\mathsf{Ver}_k}(\widetilde{V}, W, V) \\ \widetilde{V}, Y, V \leftarrow \mathsf{fVer}_k(\widetilde{V}, Y, V) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \end{array} \right\}$$

Proof. By Lemma 4.4, it suffices to prove the following where we switched Ver_k to Ver'_{k^+}:

$$\left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ \widetilde{V}, W, V \leftarrow \mathcal{A}^{\mathsf{Ver}'_{k^+}}(\widetilde{V}, W, V) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \\ \widetilde{V}, Y \leftarrow \mathsf{fDec}_{k_P}(\widetilde{V}, Y) \end{array} \right\} \approx_{2^{-\Omega(\lambda)}} \left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ \widetilde{V}, W, V \leftarrow \mathcal{A}^{\mathsf{Ver}'_{k^+}}(\widetilde{V}, W, V) \\ \widetilde{V}, Y, V \leftarrow \mathsf{fVer}_k(\widetilde{V}, Y, V) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \end{array} \right\}$$

We will prove the lemma even when k^+ is known to the adversary. Hence, it suffices to consider adversaries that can depend on k^+, and prove indistinguishability when k is generated from the output distribution of $\mathsf{KeyGen}'(1^\lambda, 1^n)$ conditioned on k^+, denoted as $\mathcal{K}$. More concretely, the distribution $\mathcal{K}$ can be sampled as follows:

- Parse $k^+ = (R, \widehat{R}, x^+, z^+)$.

- Sample a uniformly random subspace S conditioned on $\widehat{R}^{\perp} \subset S \subset R$.
- Sample a uniformly random $\Delta \leftarrow R \setminus S$.
- Sample uniformly random $x_i^R \leftarrow R$ and $z_i^{\widehat{R}} \leftarrow \widehat{R}$ for each $i \in [n]$.
- Output $k = (S, \Delta, x, z)$ where $x = x^+ + x^R$ and $z = z^+ + z^R$.

Without loss of generality, $\mathcal{A}$ performs a unitary operator $A^{\widetilde{V},W,V}$, which we decompose into $A^{\widetilde{V},W,V} = \sum_{Q\in\mathscr{P}_n} {A_Q}^{\widetilde{V},W} \otimes Q^V$. We define the following projectors.

- $\Pi_{k,b} := \sum_{\widetilde{v}:\mathsf{Ver}_{k,\theta,G}(\widetilde{v})=b}$ for $b \in \{\top, \bot\}$.
- $\Pi_{k,y} := \sum_{\widetilde{v}:f(\mathsf{Dec}_{k,\theta,G}(\widetilde{v}))=y}$ for $j \in \{0,1\}$.
- $\hat{\Pi}_y := \mathcal{M}_{=y}[f,\theta,G] = G^\dagger \mathsf{H}^\theta \left(\sum_{x:f(x)=y} |x\rangle\langle x|\right) \mathsf{H}^\theta G$

The state on the left hand side of the lemma is

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\sum_{P\in\mathscr{P}_n} k_P \otimes \mathsf{Mixed}\left[\left(\mathsf{X}^{\bot^Y}\Pi_{k_P,\bot}{}^{\widetilde{V}} + \sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\Pi_{k_P,y}{}^{\widetilde{V}}\right)(\langle P|\,\mathsf{TP})^{V,M_0} A^{\widetilde{V},W,V}\,\mathsf{Enc}_k{}^{M_1\to\widetilde{V}}|\mathsf{epr}\rangle^{M_0,M_1}|\psi\rangle^{V,W}\right]\right]\\
=&\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\sum_{P\in\mathscr{P}_n} k_P \otimes \mathsf{Mixed}\left[\left(\mathsf{X}^{\bot^Y}\Pi_{k_P,\bot}{}^{\widetilde{V}} + \sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\Pi_{k_P,y}{}^{\widetilde{V}}\right)\sum_{Q\in\mathscr{P}_n}{A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_k{}^{M_1\to\widetilde{V}}(\langle P|\,\mathsf{TP})^{V,M_0}|\mathsf{epr}\rangle^{M_0,M_1}Q^V|\psi\rangle^{V,W}\right]\right]\\
=&\mathop{\mathbb{E}}_{\substack{k\leftarrow\mathcal{K}\\P\leftarrow\mathscr{P}_n}}\left[k_P \otimes \mathsf{Mixed}\left[\sum_{Q\in\mathscr{P}_n}\left(\mathsf{X}^{\bot^Y}\Pi_{k_P,\bot}{}^{\widetilde{V}} + \sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\Pi_{k_P,y}{}^{\widetilde{V}}\right){A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_k{}^{V\to\widetilde{V}}P^VQ^V|\psi\rangle^{V,W}\right]\right]\\
=&\mathop{\mathbb{E}}_{\substack{k\leftarrow\mathcal{K}\\P\leftarrow\mathscr{P}_n}}\left[k_P \otimes \mathsf{Mixed}\left[\sum_{Q\in\mathscr{P}_n}\left(\mathsf{X}^{\bot^Y}\Pi_{k,\bot}{}^{\widetilde{V}} + \sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\Pi_{k_P,y}{}^{\widetilde{V}}\right){A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}}Q^V|\psi\rangle^{V,W}\right]\right]
\end{aligned}
$$

where the first equality follows from the commutativity between several operators, the second equality follows from quantum teleportation, and the last equality follows from Lemma 4.5. The state on the right-hand side of the lemma is

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\sum_{P\in\mathscr{P}_n} k_P \otimes \mathsf{Mixed}\left[(\langle P|\,\mathsf{TP})^{V,M_0}\left(\mathsf{X}^{\bot^Y}\Pi_{k,\bot}{}^{\widetilde{V}} + \sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\hat{\Pi}_y^V\Pi_{k,\top}{}^{\widetilde{V}}\right)A^{\widetilde{V},W,V}\,\mathsf{Enc}_k{}^{M_1\to\widetilde{V}}|\mathsf{epr}\rangle^{M_0,M_1}|\psi\rangle^{V,W}\right]\right]\\
=&\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}}\left[\sum_{P\in\mathscr{P}_n} k_P \otimes \mathsf{Mixed}\left[\begin{array}{l}\sum_{Q\in\mathscr{P}_n}\mathsf{X}^{\bot^Y}\Pi_{k,\bot}{}^{\widetilde{V}}{A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_k{}^{M_1\to\widetilde{V}}(\langle P|\,\mathsf{TP})^{V,M_0}|\mathsf{epr}\rangle^{M_0,M_1}Q^V|\psi\rangle^{V,W}\\ +\sum_{Q\in\mathscr{P}_n}\sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\Pi_{k,\top}{}^{\widetilde{V}}{A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_k{}^{M_1\to\widetilde{V}}(\langle P|\,\mathsf{TP})^{V,M_0}|\mathsf{epr}\rangle^{M_0,M_1}\hat{\Pi}_y^VQ^V|\psi\rangle^{V,W}\end{array}\right]\right]\\
=&\mathop{\mathbb{E}}_{\substack{k\leftarrow\mathcal{K}\\P\leftarrow\mathscr{P}_n}}\left[k_P \otimes \mathsf{Mixed}\left[\begin{array}{l}\sum_{Q\in\mathscr{P}_n}\mathsf{X}^{\bot^Y}\Pi_{k,\bot}{}^{\widetilde{V}}{A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}}Q^V|\psi\rangle^{V,W}\\ +\sum_{Q\in\mathscr{P}_n}\sum_{y\in\{0,1\}}\mathsf{X}^{y^Y}\Pi_{k,\top}{}^{\widetilde{V}}{A_Q}^{\widetilde{V},W}\,\mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}}\hat{\Pi}_y^VQ^V|\psi\rangle^{V,W}\end{array}\right]\right]
\end{aligned}
$$

where the first equality follows from the commutativity between several operators, and the second equality follows from quantum teleportation and Lemma 4.5. Since we always have $\mathrm{Tr}\left|\mathsf{Mixed}\left[|\phi_0\rangle\right] - \mathsf{Mixed}\left[|\phi_1\rangle\right]\right| \le 2\,\||\phi_0\rangle - |\phi_1\rangle\|$, the two states we considered above have trace distance at most

$$
\begin{aligned}
&2 \mathop{\mathbb{E}}_{\substack{k\leftarrow\mathcal{K}\\ P\leftarrow\mathscr{P}_n}} \left\| \sum_{Q\in\mathscr{P}_n}\sum_{y\in\{0,1\}} \mathsf{X}^{y^Y} \left(\Pi_{k_P,y}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} - \Pi_{k,\top}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} \hat{\Pi}_y^V\right) Q^V |\psi\rangle^{V,W} \right\| \\
&= 2 \mathop{\mathbb{E}}_{\substack{k\leftarrow\mathcal{K}\\ P\leftarrow\mathscr{P}_n}} \left\| \sum_{Q\in\mathscr{P}_n}\sum_{y\in\{0,1\}} \mathsf{X}^{y^Y} \begin{pmatrix} \Pi_{k_P,y}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} \hat{\Pi}_y^V + \Pi_{k_P,y}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} \hat{\Pi}_{y\oplus 1}^V \\ -\Pi_{k_P,y}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} \hat{\Pi}_y^V - \Pi_{k_P,y\oplus 1}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} \hat{\Pi}_y^V \end{pmatrix} Q^V |\psi\rangle^{V,W} \right\| \\
&\le 4 \sum_{y\in\{0,1\}}\sum_{Q\in\mathscr{P}_n} \mathop{\mathbb{E}}_{\substack{k\leftarrow\mathcal{K}\\ P\leftarrow\mathscr{P}_n}} \left\| \Pi_{k_P,y\oplus 1}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_{k_P}{}^{V\to\widetilde{V}} \hat{\Pi}_y^V Q^V |\psi\rangle^{V,W} \right\| \\
&\le 4 \sum_{y\in\{0,1\}}\sum_{Q\in\mathscr{P}_n} \left(\mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}} \left\| \Pi_{k,y\oplus 1}{}^{\widetilde{V}} A_Q{}^{\widetilde{V},W} \mathsf{Enc}_k{}^{V\to\widetilde{V}} \hat{\Pi}_y^V Q^V |\psi\rangle^{V,W} \right\|^2 \right)^{\frac{1}{2}} \\
&\le 4 \sum_{y\in\{0,1\}}\sum_{Q\in\mathscr{P}_n} \left(\mathop{\mathbb{E}}_{P'\leftarrow\mathscr{P}_n} \mathop{\mathbb{E}}_{k\leftarrow\mathcal{K}} \left\| \Pi_{k,y\oplus 1}{}^{\widetilde{V}} \left(P'^{\dagger V'} A^{\widetilde{V},W,V'} P'^{V'}\right) \mathsf{Enc}_k{}^{V\to\widetilde{V}} \hat{\Pi}_y^V \left(Q^V |\psi\rangle^{V,W} |0^n\rangle^{V'}\right) \right\|^2 \right)^{\frac{1}{2}}
\end{aligned}
$$

where the second line uses $I = \hat{\Pi}_y + \hat{\Pi}_{y\oplus 1}$ and $\Pi_{k,\top} = \Pi_{k_P,\top} = \Pi_{k_P,y} + \Pi_{k_P,y\oplus 1}$, the third line follows from cancellation and the triangle inequality, the fourth line makes the observation that the distribution of k_P is $\mathcal{K}$ and applies the Cauchy-Schwartz inequality, and the last line follows from a standard Pauli twirling on $A = \sum_{Q\in\mathscr{P}_n} A_Q \otimes Q$. At this point, recall that by definition $\Pi_{k,y\oplus 1} = \mathcal{M}_{=y\oplus 1}\left[(f\circ \mathsf{Dec}_{k,\theta,G}), \widetilde{\theta}, \widetilde{G}\right]$ whereas $\hat{\Pi}_y = \mathcal{M}_{=y}[f,\theta,G]$. To conclude the proof, we incorporate the following lemma from [BBV24] stating that the coset authentication code satisfies soundness even given k^+, which implies that the trace distance is upper bounded by $2^{O(n)}\cdot 2^{-\Omega(\lambda)} \le 2^{-\Omega(\lambda)}$ for a regime where $\lambda = \Omega(n)$. □

Lemma 4.7 (Soundness given k^+, [BBV24])**.** *Let $\mathcal{K}$ be the conditional distribution of k from $\mathsf{KeyGen}'(1^\lambda, 1^n)$ given k^+. For every k^+, f, θ, G, and quantum adversary $\mathcal{A}$, there exists an $\epsilon = \epsilon(\lambda) \in [0,1]$ such that for any state $|\phi\rangle$,*

$$
\left\{ y \leftarrow \mathcal{M}\left[(f\circ \mathsf{Dec}_{k,\theta,G}), \widetilde{\theta}, \widetilde{G}\right](\widetilde{V}) \;\middle|\; \begin{array}{l} V, W \leftarrow |\phi\rangle \\ k \leftarrow \mathcal{K} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \mathcal{A}(\widetilde{V}, W) \end{array} \right\} \mathop{\approx}_{2^{-\Omega(\lambda)}} (1-\epsilon)\{y \leftarrow \mathcal{M}[f,\theta,G](\phi)\} + \epsilon\{\perp\}.
$$

In particular, if $\Pr[y_1 \leftarrow \mathcal{M}[f,\theta,G](\phi)] = 0$ for some y_1, then for every adversary $\mathcal{A}$,

$$
\Pr\left[y_1 \leftarrow \mathcal{M}\left[(f\circ \mathsf{Dec}_{k,\theta,G}), \widetilde{\theta}, \widetilde{G}\right](\widetilde{V}) \;\middle|\; \begin{array}{l} V, W \leftarrow |\phi\rangle \\ k \leftarrow \mathcal{K} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \mathcal{A}(\widetilde{V}, W) \end{array} \right] \le 2^{-\Omega(\lambda)}.
$$

We are ready to prove that the coset authentication code satisfies functional security.

Proof of Theorem 4.3. Consider the function $f_{j,i}(x)$ that outputs the i-th bit of $f_j(x)$. We reserve the notations $\mathsf{fDec}, \mathsf{fVer}$ for the unitary operators defined with respect to the tuple $(f_1,\ldots,f_t,\theta_1,\ldots,\theta_t,G_1,\ldots,G_t)$ and use $\mathsf{fDec}^{(j,i)}, \mathsf{fVer}^{(j,i)}$ to denote the unitary operators defined with respect to $(f_{j,i},\theta_j,G_j)$. By our convention in Section 3, the classical oracles $\{\mathsf{fDec}^{(j,i)}\}_{j,i}, \{\mathsf{fVer}^{(j,i)}\}_{j,i}$ can be used to construct their controlled versions. It is clear that $\mathsf{fDec}, \mathsf{fVer}$ can be efficiently simulated using the same oracle algorithm from the controlled

versions of $\{\mathsf{fDec}^{(j,i)}\}_{j,i}, \{\mathsf{fVer}^{(j,i)}\}_{j,i}$ respectively. Hence, it suffices to show the following indistinguishability for every adversary $\mathcal{A}$ making $\mathrm{poly}(t, n', \lambda) = \mathrm{poly}(\lambda)$ queries.

$$\left\{ (\widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \mathcal{A}^{\{\mathsf{fDec}^{(j,i)}_{k,\cdot,\cdot}(\cdot,V)\}_{j,i}}(\widetilde{V}, W) \end{array} \right\} \underset{2^{-\Omega(\lambda)}}{\approx} \left\{ (\widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(0^n) \\ \widetilde{V}, W \leftarrow \mathcal{A}^{\{\mathsf{fVer}^{(j,i)}_{k,\cdot,\cdot}(\cdot,\cdot,V)\}_{j,i}}(\widetilde{V}, W) \end{array} \right\}$$

We can express the adversary as $(U_q\ \mathsf{fDec}^{(q)} \cdots U_1\ \mathsf{fDec}^{(1)} U_0)$ on the left hand side and as $(U_q\ \mathsf{fVer}^{(q)} \cdots U_1\ \mathsf{fVer}^{(1)} U_0)$ on the right-hand side for some unitaries $U_0, \ldots, U_q$, where $q = \mathrm{poly}(\lambda)$ is the number of queries and each $\mathsf{fDec}^{(a)}, \mathsf{fVer}^{(a)}$ refers to some $\mathsf{fDec}^{(j,i)}, \mathsf{fVer}^{(j,i)}$. Let us start with the state on the left where we also keep track of the key k in the output.

$$\left\{ (k, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(V) \\ \widetilde{V}, W \leftarrow \left(U_q\ \mathsf{fDec}^{(q)}_k(\cdot, V) \cdots U_1\ \mathsf{fDec}^{(1)}_k(\cdot, V)\ U_0\right)(\widetilde{V}, W) \end{array} \right\}$$

By quantum teleportation (Lemma 3.2) and Pauli key update (Lemma 4.5), the state equals to

$$\left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \\ \widetilde{V}, W \leftarrow \left(U_q\ \mathsf{fDec}^{(q)}_{k_P}(\cdot, V) \cdots U_1\ \mathsf{fDec}^{(1)}_{k_P}(\cdot, V)\ U_0\right)(\widetilde{V}, W) \end{array} \right\}$$

By Lemma 4.6, the state is within $2^{-\Omega(\lambda)}$ trace distance to

$$\left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ (k, k^+) \leftarrow \mathsf{KeyGen}'(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ \widetilde{V}, W \leftarrow \left(\mathsf{fVer}^{(1)}_k(\cdot, \cdot, V)\ U_0\right)(\widetilde{V}, W) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \\ \widetilde{V}, W \leftarrow \left(U_q\ \mathsf{fDec}^{(q)}_{k_P}(\cdot, V) \cdots U_2\ \mathsf{fDec}^{(2)}_{k_P}(\cdot, V)\ U_1\right)(\widetilde{V}, W) \end{array} \right\}$$

By repeatedly applying Lemma 4.6, an induction proof would show that the starting state is within $O(q \cdot 2^{-\Omega(\lambda)}) = 2^{-\Omega(\lambda)}$ trace distance to the following state.

$$\left\{ (k_P, \widetilde{V}, W) \;\middle|\; \begin{array}{l} V, W \leftarrow |\psi\rangle \\ k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^n) \\ (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes n} \\ \widetilde{V} \leftarrow \mathsf{Enc}_k(\mathsf{epr}_R) \\ \widetilde{V}, W \leftarrow \left(U_q\ \mathsf{fVer}^{(q)}_k(\cdot, \cdot, V) \cdots U_1\ \mathsf{fVer}^{(1)}_k(\cdot, \cdot, V)\ U_0\right)(\widetilde{V}, W) \\ P \leftarrow \mathsf{TP.Send}(V, \mathsf{epr}_L) \end{array} \right\}$$

By tracing out the register that holds the key, we obtain

$$\left\{(\widetilde{V},W)\;\middle|\;\begin{array}{l}V,W\leftarrow|\psi\rangle\\ k\leftarrow\mathsf{KeyGen}(1^\lambda,1^n)\\ \widetilde{V}\leftarrow\mathsf{Enc}_k(V)\\ \widetilde{V},W\leftarrow\mathcal{A}^{\{\mathsf{fDec}^{(j,i)}_{k,\cdot,\cdot}(\cdot,V)\}_{j,i}}(\widetilde{V},W)\end{array}\right\}\approx_{2^{-\Omega(\lambda)}}\left\{(\widetilde{V},W)\;\middle|\;\begin{array}{l}V,W\leftarrow|\psi\rangle\\ k\leftarrow\mathsf{KeyGen}(1^\lambda,1^n)\\ (\mathsf{epr}_L,\mathsf{epr}_R)\leftarrow\mathsf{EPR}^{\otimes n}\\ \widetilde{V}\leftarrow\mathsf{Enc}_k(\mathsf{epr}_R)\\ \widetilde{V},W\leftarrow\mathcal{A}^{\{\mathsf{fVer}^{(j,i)}_{k,\cdot,\cdot}(\cdot,\cdot,V)\}_{j,i}}(\widetilde{V},W)\\ P\leftarrow\mathsf{TP.Send}(V,\mathsf{epr}_L)\end{array}\right\}$$

On the right-hand side, note that the information of (P,V,epr_L) is traced out in the end, so removing the teleportation does not change the final state. Moreover, since epr_L is never used after initialization, we can apply the teleportation $P\leftarrow\mathsf{TP.Send}(0^n,\mathsf{epr}_L)$ right after the initialization without changing the final state. By teleportation and Lemma 4.5, we then have $\widetilde{V}\leftarrow\mathsf{Enc}_k(\mathsf{epr}_R)=\mathsf{Enc}_k\circ P(0^n)=\mathsf{Enc}_{k_P}(0^n)$ and that replacing $\mathsf{fVer}^{(j,i)}_k$ with $\mathsf{fVer}^{(j,i)}_{k_P}$ does not change the final state. Now the key being used has been changed to k_P, and since k_P has the same distribution as k, the state on the right equals to

$$\left\{(\widetilde{V},W)\;\middle|\;\begin{array}{l}V,W\leftarrow|\psi\rangle\\ k\leftarrow\mathsf{KeyGen}(1^\lambda,1^n)\\ \widetilde{V}\leftarrow\mathsf{Enc}_k(0^n)\\ \widetilde{V},W\leftarrow\mathcal{A}^{\{\mathsf{fVer}^{(j,i)}_{k,\cdot,\cdot}(\cdot,\cdot,V)\}_{j,i}}(\widetilde{V},W)\end{array}\right\}$$

□

5 Projective LM Quantum Program

Finding a good representation of quantum circuits is crucial to program obfuscation. Recently, [BBV24] proposed LM quantum programs and showed how to obfuscate these programs as long as they have classical inputs and pseudo-deterministic classical outputs. In this section, we transform any quantum circuit into a "projective" LM (PLM) quantum program whose structure will facilitate the obfuscation of general quantum circuits.

First, we refine the definition of LM quantum program of [BBV24]. A unitary gate G is said to be linear if it is composed of CNOT gates. For a classical function $f:\{0,1\}^n\to\{0,1\}^*$, a string $\theta\in\{0,1\}^n$, and a linear gate G on n-qubits, we define the measurement

$$\mathcal{M}\left[f,\theta,G\right]:=\left\{G^\dagger\mathsf{H}^\theta\left(\sum_{x:f(x)=y}|x\rangle\langle x|\right)\mathsf{H}^\theta G\right\}_y$$

Moreover, for $S\subseteq\{0,1\}^*$, we define the projector

$$\mathcal{M}_{\in S}\left[f,\theta,G\right]:=G^\dagger\mathsf{H}^\theta\left(\sum_{x:f(x)\in S}|x\rangle\langle x|\right)\mathsf{H}^\theta G$$

We sometimes write $\mathcal{M}_{=y}\left[f,\theta,G\right]$ as a shorthand for $\mathcal{M}_{\in\{y\}}\left[f,\theta,G\right]$.

Definition 5.1 (LM quantum program)**.** An LM quantum program P is described by a quantum state ψ, a sequence of instructions $((f_j, \theta_j, G_j)_{j\in[t]}, g)$, and parameters (n_q, n_c, n'). Here, $f_j = \left\{ f_j^{\mathbb{i},r_1,\cdots,r_{j-1}} \right\}_{\mathbb{i}\in\{0,1\}^{n_c},\, r_1,\cdots,r_{j-1}\in\{0,1\}}$ is a set of classical functions, θ_j is a bit string, G_j is a linear gate, and g is a classical function. The program $P(\mathbb{i}, \rho)$ is executed as follows:

1. Input a classical string $\mathbb{i} \in \{0,1\}^{n_c}$ and a quantum state $\rho \in \mathcal{D}^{n_q}$.
2. Initialize a register V with the state (ρ, ψ).
3. Iteratively apply $r_j \leftarrow \mathcal{M}\left[f_j^{\mathbb{i},r_1,\cdots,r_{j-1}}, \theta_j, G_j\right]^V$ for $j = 1, \cdots, t$.
4. Output the classical outcome $g(\mathbb{i}, r_1, \ldots, r_t) \in \{0,1\}^{n'}$.

Now, we introduce our notion of PLM quantum programs, which will be a useful representation of quantum circuits.

Definition 5.2 (PLM quantum program)**.** An LM quantum program with instructions $((f_j, \theta_j, G_j)_{j\in[t]}, g)$ is said to be projective if for every $\mathbb{i}$ there exists an orthonormal basis $\left\{\Phi_{\mathbb{i},(r_1,\ldots,r_t)}\right\}$ on V such that the product

$$\mathcal{M}_{=r_t}\left[f_t^{\mathbb{i},r_1,\cdots,r_{t-1}}, \theta_t, G_t\right] \cdots \mathcal{M}_{=r_2}\left[f_2^{\mathbb{i},r_1}, \theta_2, G_2\right] \mathcal{M}_{=r_1}\left[f_1^{\mathbb{i}}, \theta_1, G_1\right]$$

is the projector $|\Phi_{\mathbb{i},(r_1,\ldots,r_t)}\rangle\langle\Phi_{\mathbb{i},(r_1,\ldots,r_t)}|$ for every $r_1, \ldots, r_t$.

In other words, applying adaptive measurements on these basis elements always yield deterministic results. These bases will become useful in the security proof of our obfuscation scheme in Section 7. The following theorem states that every quantum circuit with classical outputs can be efficiently compiled into a PLM quantum program.

Theorem 5.3 (Compilation)**.** *There is a polynomial time algorithm that converts any quantum circuit Q with classical output into a PLM quantum program $P = (\psi_{\mathsf{PLM}}, \{f_j, \theta_j, G_j\}_{j\in[t]})$ with the following properties.*

- $|\psi_{\mathsf{PLM}}|, t = O(m)$ *where m is the size plus the width of Q.*
- *For every classical input $\mathbb{i}$, quantum state ρ, and possibly entangled state τ, running the circuit Q and running the PLM quantum program P yield identical distributions*

$$\left\{(y,\tau) \;\middle|\; y \leftarrow Q(\mathbb{i},\rho)\right\} \equiv \left\{(y,\tau) \;\middle|\; y \leftarrow P(\mathbb{i},\rho)\right\}$$

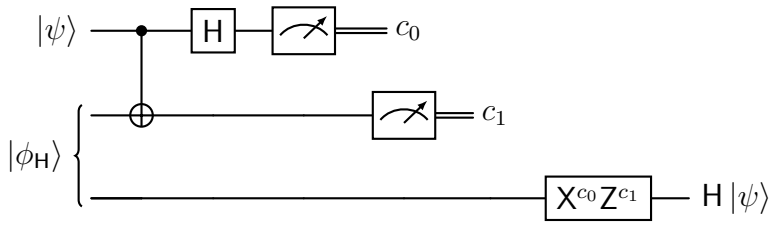
- *By deferring measurements, let $U_{Q,\mathbb{i}}$ be a unitary operator where $Q(\mathbb{i}, \rho)$ is equivalent to first applying $U_{Q,\mathbb{i}}$ to ρ then measuring n' qubits in the standard basis as the output. Let $\{|\Phi_{\mathbb{i},r}\rangle\}_{r\in\{0,1\}^t}$ be the orthonormal basis associated with P and $\mathbb{i}$ as in Definition 5.2. Then for every $\mathbb{i} \in \{0,1\}^{n_c}$ and $y \in \{0,1\}^{n'}$,*

$$\sum_y X^y \otimes \sum_{r:g(\mathbb{i},r)=y} |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}| \left(I^{\otimes n_q} \otimes |\psi_{\mathsf{PLM}}\rangle\right) = \sum_y X^y \otimes \left(U_{Q,\mathbb{i}}^\dagger \left(|y\rangle\langle y| \otimes I^{\otimes(n_q-n')}\right) U_{Q,\mathbb{i}}\right) \otimes |\psi_{\mathsf{PLM}}\rangle$$

We start by describing how to implement each gate in the universal gate set $\{\mathsf{CNOT}, \mathsf{H}, \mathsf{T}\}$ using magic states, Clifford gates, adaptive measurements, and Pauli corrections. For each implementation, we also provide an orthonormal basis on which the adaptive measurements yield deterministic outcomes. These orthonormal bases will ultimately be tensored into an orthonormal basis over the entire circuit, which facilitates the construction of projective LM quantum programs. The most distinctive aspect of our construction lies in our implementation of T gates.

Implementing the H gate. We prepare the magic state $|\phi_{\mathsf{H}}\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$ and perform the following implementation of H as in [BGS13].

Circuit 1.



The correctness of the computation has been shown in both [BGS13, BBV24]. We remark that the two measurements would completely collapse the state in the top two wires, and the measurement results would decide the Pauli correction on the last wire. Also, it is immediate to see that on the first two wires, every state in the Bell basis will produce deterministic measurement results. More precisely, it is the basis

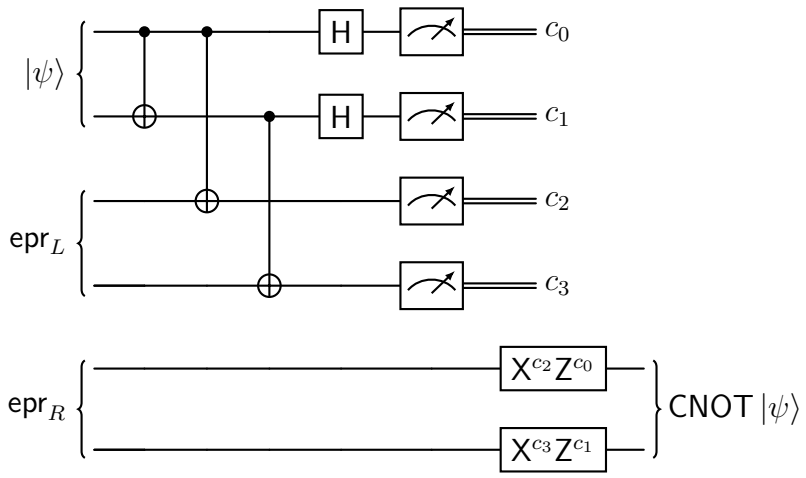
$$\beta^{\mathsf{H}} := \left\{ |\widetilde{\Phi}^{\mathsf{H}}_{c_0,c_1}\rangle \;\middle|\; c_0, c_1 \in \{0,1\} \right\} := \left\{ (I \otimes \mathsf{X}^{c_1}\mathsf{Z}^{c_0}) \frac{|00\rangle + |11\rangle}{\sqrt{2}} \;\middle|\; c_0, c_1 \in \{0,1\} \right\}$$

where c_0, c_1 would be the deterministic measurement results. The correctness of the circuit shows that for some coefficients $\{\alpha^{\mathsf{H}}_{c_0,c_1}\}$, we have

$$|\psi\rangle \otimes |\phi_{\mathsf{H}}\rangle = \sum_{c_0,c_1} \alpha^{\mathsf{H}}_{c_0,c_1} |\widetilde{\Phi}^{\mathsf{H}}_{c_0,c_1}\rangle \otimes \mathsf{X}^{c_0}\mathsf{Z}^{c_1}\mathsf{H}\,|\psi\rangle$$

Implementing the CNOT gate. We prepare the state $\phi_{\mathsf{CNOT}} = (\mathsf{epr}_L, \mathsf{epr}_R) \leftarrow \mathsf{EPR}^{\otimes 2}$.

Circuit 2.



The correctness of the computation follows from quantum teleportation (Lemma 3.2). We remark that the four measurements would completely collapse the state in the top four wires, and the measurement results would determine the Pauli corrections on the last two wires. Also, it is immediate to see that on the first four wires, every state in the two-fold tensor of the Bell basis will produce deterministic measurement results. We denote the basis as

$$\beta^{\mathsf{CNOT}} := \left\{ |\widetilde{\Phi}^{\mathsf{CNOT}}_{c_0,c_1,c_2,c_3}\rangle \;\middle|\; c_0,c_1,c_2,c_3 \in \{0,1\} \right\}$$

where c_0, c_1, c_2, c_3 would be the deterministic measurement results. The correctness of the circuit shows that for some coefficients $\{\alpha^{\mathsf{CNOT}}_{c_0,c_1,c_2,c_3}\}$, we have

$$|\psi\rangle \otimes |\phi_{\mathsf{CNOT}}\rangle = \sum_{c_0,c_1,c_2,c_3} \alpha^{\mathsf{CNOT}}_{c_0,c_1,c_2,c_3} |\widetilde{\Phi}^{\mathsf{CNOT}}_{c_0,c_1,c_2,c_3}\rangle \otimes (\mathsf{X}^{c_2}\mathsf{Z}^{c_0} \otimes \mathsf{X}^{c_3}\mathsf{Z}^{c_1})\mathsf{CNOT}\,|\psi\rangle$$

Implementing the T gate. We prepare the magic state $|\phi'_{\mathsf{T}}\rangle = |\phi_{\mathsf{T}}\rangle \otimes |\phi_{\mathsf{S}^\dagger}\rangle \otimes |\mathsf{epr}\rangle$, where

$$|\phi_{\mathsf{T}}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}\,|1\rangle), \quad |\phi_{\mathsf{S}^\dagger}\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle), \quad |\mathsf{epr}\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

We then define the measurements $\Gamma_0, \Gamma'_0, \Gamma_1, \Gamma'_1$ so that

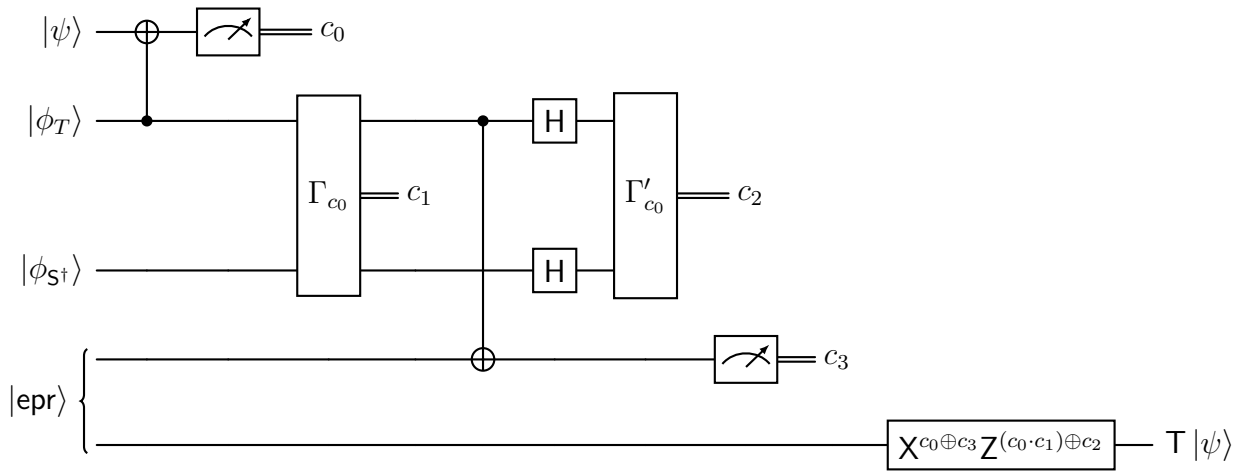
$$\begin{aligned} \Gamma_0 &\text{ measures the output bit of the classical function } (a,b) \mapsto b \\ \Gamma'_0 &\text{ measures the output bit of the classical function } (a,b) \mapsto a \\ \Gamma_1, \Gamma'_1 &\text{ measure the output bit of the classical function } (a,b) \mapsto a \oplus b \end{aligned}$$

In other words, these measurements are

$$\begin{aligned} \Gamma_0 &= \{|00\rangle\langle 00| + |10\rangle\langle 10|\,,\ |01\rangle\langle 01| + |11\rangle\langle 11|\} \\ \Gamma'_0 &= \{|00\rangle\langle 00| + |01\rangle\langle 01|\,,\ |10\rangle\langle 10| + |11\rangle\langle 11|\} \\ \Gamma_1 = \Gamma'_1 &= \{|00\rangle\langle 00| + |11\rangle\langle 11|\,,\ |01\rangle\langle 01| + |10\rangle\langle 10|\} \end{aligned}$$

Lemma 5.4. *The following circuit implements the* T *gate.*

Circuit 3.



We remark that the four measurements would completely collapse the state in the top four wires, and the measurement results would decide the Pauli correction on the last wire. We now prove that the computation in the diagram is indeed correct.

Proof. We write $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$. After applying the first CNOT, the state on the top two wires becomes

$$\begin{aligned}&\frac{1}{\sqrt{2}}\left(\alpha\,|00\rangle + \beta\,|10\rangle + e^{i\pi/4}\beta\,|01\rangle + e^{i\pi/4}\alpha\,|11\rangle\right)\\ =&\frac{1}{\sqrt{2}}\,|0\rangle \otimes \left(\alpha\,|0\rangle + e^{i\pi/4}\beta\,|1\rangle\right) + \frac{1}{\sqrt{2}}\,|1\rangle \otimes \left(\beta\,|0\rangle + e^{i\pi/4}\alpha\,|1\rangle\right)\end{aligned}$$

When the measurement result of the first wire is $c_0 = 0$, the state on the second wire collapses to $\alpha\,|0\rangle + e^{i\pi/4}\beta\,|1\rangle = \mathsf{T}\,|\psi\rangle$. In this case, the next measurement Γ_0 only measures the third wire, so the state on the second wire stays the same. Then, observe that Γ'_0 is a standard basis measurement on the second wire, so the operations on the second and fourth wires after Γ_0 is exactly the teleportation operation. Hence, the fifth wire would hold the state $\mathsf{X}^{c_3}\mathsf{Z}^{c_2}\mathsf{T}\,|\psi\rangle$, and after applying $\mathsf{X}^{c_0\oplus c_3}\mathsf{Z}^{(c_0\cdot c_1)\oplus c_2} = \mathsf{X}^{c_3}\mathsf{Z}^{c_2}$, the state becomes $\mathsf{T}\,|\psi\rangle$ up to a global phase.

When the measurement result of the first wire is $c_0 = 1$, the state on the second wire collapses to $\left(\beta\,|0\rangle + e^{i\pi/4}\alpha\,|1\rangle\right)$. The joint state on wires 2 and 3 is

$$\begin{aligned}&\left(\beta\,|0\rangle + e^{i\pi/4}\alpha\,|1\rangle\right) \otimes \frac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle)\\ =&\frac{1}{\sqrt{2}}\left(\beta\,|00\rangle - ie^{i\pi/4}\alpha\,|11\rangle\right) + \frac{1}{\sqrt{2}}\left(e^{i\pi/4}\alpha\,|10\rangle - i\beta\,|01\rangle\right) \qquad (1)\end{aligned}$$

In this case, Γ_1 measures the parity of wires 2 and 3 in the standard basis. When the measurement result is $c_1 = 0$, the state becomes $\beta\,|00\rangle - ie^{i\pi/4}\alpha\,|11\rangle = e^{-i\pi/4}\left(e^{i\pi/4}\beta\,|00\rangle + \alpha\,|11\rangle\right)$.

After the following CNOT, the state on wires $2,3,4,5$, up to a global phase, becomes

$$\begin{aligned}
&\frac{1}{\sqrt{2}}\left(e^{i\pi/4}\beta\,|0000\rangle + e^{i\pi/4}\beta\,|0011\rangle + \alpha\,|1110\rangle + \alpha\,|1101\rangle\right)\\
=&\frac{1}{2\sqrt{2}}\begin{pmatrix} e^{i\pi/4}\beta\,|++00\rangle + e^{i\pi/4}\beta\,|--00\rangle + e^{i\pi/4}\beta\,|+-00\rangle + e^{i\pi/4}\beta\,|-+00\rangle\\ +e^{i\pi/4}\beta\,|++11\rangle + e^{i\pi/4}\beta\,|--11\rangle + e^{i\pi/4}\beta\,|+-11\rangle + e^{i\pi/4}\beta\,|-+11\rangle\\ +\alpha\,|++10\rangle + \alpha\,|--10\rangle - \alpha\,|+-10\rangle - \alpha\,|-+10\rangle\\ +\alpha\,|++01\rangle + \alpha\,|--01\rangle - \alpha\,|+-01\rangle - \alpha\,|-+01\rangle \end{pmatrix}\\
=&\ \frac{1}{2}\left(\frac{|++\rangle + |--\rangle}{\sqrt{2}}\right)\otimes\left(\,|0\rangle\otimes\left(e^{i\pi/4}\beta\,|0\rangle + \alpha\,|1\rangle\right) + |1\rangle\otimes\left(e^{i\pi/4}\beta\,|1\rangle + \alpha\,|0\rangle\right)\right)\\
&+\frac{1}{2}\left(\frac{|+-\rangle + |-+\rangle}{\sqrt{2}}\right)\otimes\left(\,|0\rangle\otimes\left(e^{i\pi/4}\beta\,|0\rangle - \alpha\,|1\rangle\right) + |1\rangle\otimes\left(e^{i\pi/4}\beta\,|1\rangle - \alpha\,|0\rangle\right)\right)
\end{aligned}$$

Next, the Hadamard gates and Γ_1' is to measure the parity on wires 2 and 3 in the Hadamard basis. One also measures the fourth wire in the standard basis. When these two measurement results are c_2 and c_3 respectively, from the above expansion we see that the state on the fifth wire becomes $\mathsf{X}^{c_3}\mathsf{Z}^{c_2}(e^{i\pi/4}\beta\,|0\rangle + \alpha\,|1\rangle) = \mathsf{X}^{c_3}\mathsf{Z}^{c_2}\mathsf{T}\,|\psi\rangle$. After applying $\mathsf{X}^{c_0\oplus c_3}\mathsf{Z}^{(c_0\cdot c_1)\oplus c_2} = \mathsf{X}^{c_3\oplus 1}\mathsf{Z}^{c_2}$, the state on the fifth wire becomes $\mathsf{T}\,|\psi\rangle$ up to a global phase.

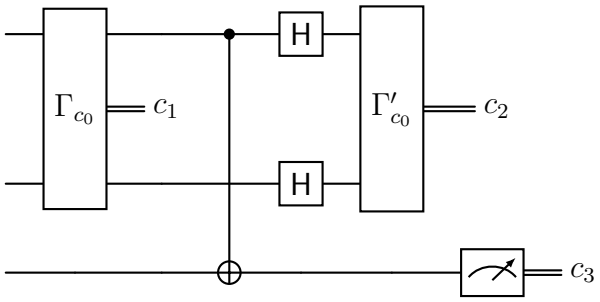
When the measurement results are $c_0 = 1$ and $c_1 = 1$, we see from expression (1) that the state on wires 2 and 3 becomes $e^{i\pi/4}\alpha\,|10\rangle - i\beta\,|01\rangle = e^{i\pi/4}\left(\alpha\,|10\rangle - e^{i\pi/4}\beta\,|01\rangle\right)$. After the following CNOT, the state on wires $2,3,4,5$, up to a global phase, becomes

$$\begin{aligned}
&\frac{1}{\sqrt{2}}\left(\alpha\,|1010\rangle + \alpha\,|1001\rangle - e^{i\pi/4}\beta\,|0100\rangle - e^{i\pi/4}\beta\,|0111\rangle\right)\\
=&\frac{1}{2\sqrt{2}}\begin{pmatrix} \alpha\,|++10\rangle - \alpha\,|--10\rangle + \alpha\,|+-10\rangle - \alpha\,|-+10\rangle\\ +\alpha\,|++01\rangle - \alpha\,|--01\rangle + \alpha\,|+-01\rangle - \alpha\,|-+01\rangle\\ -e^{i\pi/4}\beta\,|++00\rangle + e^{i\pi/4}\beta\,|--00\rangle + e^{i\pi/4}\beta\,|+-00\rangle - e^{i\pi/4}\beta\,|-+00\rangle\\ -e^{i\pi/4}\beta\,|++11\rangle + e^{i\pi/4}\beta\,|--11\rangle + e^{i\pi/4}\beta\,|+-11\rangle - e^{i\pi/4}\beta\,|-+11\rangle \end{pmatrix}\\
=&\ \frac{1}{2}\left(\frac{|++\rangle - |--\rangle}{\sqrt{2}}\right)\otimes\left(\,|0\rangle\otimes\left(\alpha\,|1\rangle - e^{i\pi/4}\beta\,|0\rangle\right) + |1\rangle\otimes\left(\alpha\,|0\rangle - e^{i\pi/4}\beta\,|1\rangle\right)\right)\\
&+\frac{1}{2}\left(\frac{|+-\rangle - |-+\rangle}{\sqrt{2}}\right)\otimes\left(\,|0\rangle\otimes\left(\alpha\,|1\rangle + e^{i\pi/4}\beta\,|0\rangle\right) + |1\rangle\otimes\left(\alpha\,|0\rangle + e^{i\pi/4}\beta\,|1\rangle\right)\right)
\end{aligned}$$

Next, the Hadamard gates and Γ_1' is to measure the parity on wires 2 and 3 in the Hadamard basis. One also measures the fourth wire in the standard basis. When these two measurement results are c_2 and c_3 respectively, we see that the state on the fifth wire becomes $\mathsf{X}^{c_3\oplus 1}\mathsf{Z}^{c_2\oplus 1}(\alpha\,|0\rangle + e^{i\pi/4}\beta\,|1\rangle) = \mathsf{X}^{c_3\oplus 1}\mathsf{Z}^{c_2\oplus 1}\mathsf{T}\,|\psi\rangle$. After applying $\mathsf{X}^{c_0\oplus c_3}\mathsf{Z}^{(c_0\cdot c_1)\oplus c_2} = \mathsf{X}^{c_3\oplus 1}\mathsf{Z}^{c_2\oplus 1}$, the state on the fifth wire becomes $\mathsf{T}\,|\psi\rangle$ up to a global phase. □

Our implementation of T gate is designed to admit an orthonormal basis on which the adaptive measurement results are deterministic. We first prove it for a sub-circuit.

Lemma 5.5. *For every $c_0 \in \{0,1\}$, there exists an orthonormal basis on three qubits such that every state in the basis, when applied the following circuit, produces deterministic measurement outcome (c_1, c_2, c_3).*



Proof. For $c_0 = 0$, the measurement Γ_0 measures the second wire in the standard basis and Γ'_0 measures the first wire in the standard basis. Thus, it directly follows that the standard basis on the second wire tensor with the Bell basis on the first and third wires constitute an orthonormal basis with deterministic measurement outcome (c_1, c_2, c_3).

For $c_0 = 1$, both Γ_1 and Γ'_1 are measuring the parity of first and second wires in the standard basis. Consider the following orthonormal basis

$$\left\{ \frac{|000\rangle \pm |111\rangle}{\sqrt{2}}, \frac{|001\rangle \pm |110\rangle}{\sqrt{2}}, \frac{|101\rangle \pm |010\rangle}{\sqrt{2}}, \frac{|100\rangle \pm |011\rangle}{\sqrt{2}} \right\}$$

Each basis element can be expressed as $\frac{1}{\sqrt{2}}\left(|b_1 b_2 b_3\rangle + (-1)^z |\bar{b}_1 \bar{b}_2 \bar{b}_3\rangle\right)$ for some $b_1, b_2, b_3, z \in \{0,1\}$, where $\bar{b} = 1 - b$. We now show that this state always produces deterministic measurement results c_1, c_2, c_3 in the circuit. The first measurement Γ_1 measures the parity of the first two wires in the standard basis. Since $b_1 \oplus b_2 = \bar{b}_1 \oplus \bar{b}_2$, the measurement Γ_1 always produces $c_1 = b_1 \oplus b_2$. After applying the CNOT gate, the state becomes $\frac{1}{\sqrt{2}}\left(|b_1 b_2\rangle |b_1 \oplus b_3\rangle + (-1)^z |\bar{b}_1 \bar{b}_2\rangle |\bar{b}_1 \oplus \bar{b}_3\rangle\right) = \frac{1}{\sqrt{2}}\left(|b_1 b_2\rangle + (-1)^z |\bar{b}_1 \bar{b}_2\rangle\right) \otimes |b_1 \oplus b_3\rangle$. So the measurement on the third wire always produces $c_3 = b_1 \oplus b_3$. Lastly, we apply Hadamard gates on $\frac{1}{\sqrt{2}}\left(|b_1 b_2\rangle + (-1)^z |\bar{b}_1 \bar{b}_2\rangle\right)$. The resulting state will be one of the following four cases.

$$\frac{|{+}{+}\rangle + |{-}{-}\rangle}{\sqrt{2}} = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \qquad \frac{|{+}{-}\rangle + |{-}{+}\rangle}{\sqrt{2}} = \frac{|00\rangle - |11\rangle}{\sqrt{2}},$$
$$\frac{|{+}{+}\rangle - |{-}{-}\rangle}{\sqrt{2}} = \frac{|10\rangle + |01\rangle}{\sqrt{2}}, \qquad \frac{|{+}{-}\rangle - |{-}{+}\rangle}{\sqrt{2}} = \frac{|10\rangle - |01\rangle}{\sqrt{2}}.$$

The measurement Γ'_1 then measures the parity in the standard basis. In the top two cases, it always produces $c_2 = 0$. In the bottom two cases, it always produces $c_2 = 1$. □

Lemma 5.6. *There exists an orthonormal basis* $\beta^{\mathsf{T}} := \left\{ |\widetilde{\Phi}^{\mathsf{T}}_{c_0,c_1,c_2,c_3}\rangle \;\middle|\; c_0, c_1, c_2, c_3 \in \{0,1\} \right\}$ *on four qubits such that applying Circuit 3 with the top four wires being* $|\widetilde{\Phi}^{\mathsf{T}}_{c_0,c_1,c_2,c_3}\rangle$ *would produce deterministic measurement outcome* (c_0, c_1, c_2, c_3).

Proof. By Lemma 5.5, there exist two orthonormal bases $\left\{ |\Phi^{(0)}_{c_1,c_2,c_3}\rangle \right\}$ and $\left\{ |\Phi^{(1)}_{c_1,c_2,c_3}\rangle \right\}$ such that applying the sub-circuit of Lemma 5.5 on $|\Phi^{(c_0)}_{c_1,c_2,c_3}\rangle$ yields deterministic outcome (c_1, c_2, c_3). Define $|\widetilde{\Phi}^{\mathsf{T}}_{c_0,c_1,c_2,c_3}\rangle := \mathsf{CNOT}^{2,1}\left(|c_0\rangle \otimes |\Phi^{(c_0)}_{c_1,c_2,c_3}\rangle\right)$. It is clear that such states form an orthonormal basis on four qubits, and directly applying Circuit 3 shows that $|\widetilde{\Phi}^{\mathsf{T}}_{c_0,c_1,c_2,c_3}\rangle$ yields deterministic measurement outcome (c_0, c_1, c_2, c_3). □

It follows from Lemmas 5.4 and 5.6 that there exist some coefficients $\{\alpha^{\mathsf{T}}_{c_0,c_1,c_2,c_3}\}$ with

$$|\psi\rangle \otimes |\phi'_{\mathsf{T}}\rangle = \sum_{c_0,c_1,c_2,c_3} \alpha^{\mathsf{T}}_{c_0,c_1,c_2,c_3} |\widetilde{\Phi}^{\mathsf{T}}_{c_0,c_1,c_2,c_3}\rangle \otimes \mathsf{X}^{c_0 \oplus c_3} \mathsf{Z}^{(c_0 \cdot c_1) \oplus c_2} \mathsf{T} |\psi\rangle$$

We are now ready to prove the compilation theorem.

Proof of Theorem 5.3. Given a quantum circuit Q that also takes a classical input $\mathbb{i}$, we will introduce ancilla qubits $|\mathbb{i}\rangle$ and replace the gates of Q with controlled gates whose descriptions do not depend on $\mathbb{i}$. By deferring all measurements and adding extra measurements, we further ensure that the quantum circuit first applies a sequence of gates and then measures all wires in the standard basis. We collect the sequence of gates as another circuit $\widehat{Q}$ and express it using the universal gate set $\{\mathsf{CNOT}, \mathsf{H}, \mathsf{T}\}$.

We now construct our compilation algorithm, which iteratively reads the gates of $\widehat{Q}$ and maintains a function h, a linear gate G^*, and a string θ^*. Here, the function h computes the Pauli corrections (z_i, x_i) of each wire i using the classical input $\mathbb{i}$ and previous measurement results. At the end of each iteration, the algorithm appends additional states and instructions to the LM quantum program. In addition to the algorithm and throughout the iterations, we will maintain an orthonormal basis $\widetilde{\gamma}$ on the wires that have encountered measurements, where the basis $\widetilde{\gamma}$ depends on the classical input $\mathbb{i}$.

First, the algorithm appends n_c additional wires initialized with 0^{n_c}. On these wires, we set the value of h to be $(0^{n_c}, \mathbb{i})$ to remember that there is a Pauli correction $\mathsf{X}^{\mathbb{i}}$, and we set the value of h as zero on all other wires. We initialize $G^* = I$ and $\theta^* = \vec{0}$. There is nothing to be included in the orthonormal basis in the beginning. Iteratively, for each gate in the quantum circuit, the algorithm uses the corresponding circuit from 2, 1, 3 to instantiate the gate. Note that circuits 2, 1, 3 only involve Clifford gates and measurements in the standard or Hadamard basis, so Pauli corrections on the input state can be delayed. The compilation of each gate, including the computation of delayed Pauli corrections through the function h, is described as follows.

- H on wire i: composing Circuit 1.
  - Append two new wires j, k to the current set of wires.
  - Append the state $|\phi_{\mathsf{H}}\rangle$ to the quantum state of the LM quantum program.
  - Append a CNOT gate from wire i to wire j to the end of G^*, and set $\theta^*_i = 1$.
  - Append $(d_i : v \mapsto v_i,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Suppose this corresponds to the e-th measurement.
  - Append $(d_j : v \mapsto v_j,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Suppose this corresponds to the $(e+1)$-th measurement.
  - Update h as follows on top of the original computation result (z, x): Set the values $(z_k, x_k) \leftarrow (z_i \oplus r_e, x_i \oplus r_{e+1})$ and delete the values (z_i, x_i).
  - Extend $\widetilde{\gamma}$ by tensoring with the basis β^{H} on wires i, j.
  - Use wire k whenever future operations refer to wire i.
- CNOT from wire i to j: composing Circuit 2.

- Append four new wires k, ℓ, m, s to the current set of wires.
- Append the state $|\phi_{\mathsf{CNOT}}\rangle$ to the quantum state of the LM quantum program.
- Append a CNOT gate from wire i to wire j to the end of G^*.
- Append a CNOT gate from wire i to wire k to the end of G^*, and set $\theta^*_i = 1$.
- Append a CNOT gate from wire j to wire ℓ to the end of G^*, and set $\theta^*_j = 1$.
- Append $(d_i : v \mapsto v_i,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Append $(d_j : v \mapsto v_j,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Append $(d_k : v \mapsto v_k,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Append $(d_\ell : v \mapsto v_\ell,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Suppose these correspond to the e-th to $(e+3)$-th measurements.
- Update h as follows on top of the original computation result (z, x): Set values $(z_m, x_m) \leftarrow (z_i \oplus z_j, x_i)$ and $(z_s, x_s) \leftarrow (z_j, x_i \oplus x_j)$ and then delete $(z_i, x_i), (z_j, x_j)$.
- Extend $\widetilde{\gamma}$ by tensoring with the basis β^{CNOT} on wires i, j, k, ℓ.
- Use wires m, s whenever future operations refer to wires i, j.

- T on wire i: composing Circuit 3.
  - Append four new wires j, k, ℓ, m to the current set of wires.
  - Append the state $|\phi'_{\mathsf{T}}\rangle$ to the quantum state of the LM quantum program.
  - Append a CNOT gate from wire j to wire i to the end of G^*.
  - Append $(d_i : v \mapsto v_i,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Suppose this corresponds to the e-th measurement.
  - Compute the output (z_i, x_i) of h on wire i.
  - Define the following functions d', d'' that depend on the e-th measurement result r_e and the correction bit x_i derived from $\mathbb{i}, r_1, \ldots, r_e$.
    $$d' : v \mapsto \begin{cases} v_k & \text{if } r_e \oplus x_i = 0 \\ v_j \oplus v_k & \text{if } r_e \oplus x_i = 1 \end{cases} \qquad d'' : v \mapsto \begin{cases} v_j & \text{if } r_e \oplus x_i = 0 \\ v_j \oplus v_k & \text{if } r_e \oplus x_i = 1 \end{cases}$$
  - Append (d', θ^*, G^*) to the instruction of the LM quantum program. Suppose this corresponds to the $(e+1)$-th measurement.
  - Append a CNOT gate from wire j to wire ℓ to the end of G^*, and set $\theta^*_j = \theta^*_k = 1$.
  - Append (d'', θ^*, G^*) to the instruction of the LM quantum program. Suppose this corresponds to the $(e+2)$-th measurement.
  - Append $(d_\ell : v \mapsto v_\ell,\ \theta^*, G^*)$ to the instruction of the LM quantum program. Suppose this corresponds to the $(e+3)$-th measurement.
  - Update h as follows on top of the original computation result (z, x): Set the values $(z_m, x_m) \leftarrow (z_i \oplus (r_e \cdot r_{e+1}) \oplus r_{e+2},\ x_i \oplus r_e \oplus r_{e+3})$ and delete the values (z_i, x_i).
  - Extend $\widetilde{\gamma}$ by tensoring with the basis β^{T} on wires i, j, k, ℓ.
  - Use wire m whenever future operations refer to wire i.

Denote the quantum auxiliary state constructed so far as $|\psi_{\mathsf{PLM}}\rangle$. Since $\widetilde{\gamma}$ is a tensor of $\beta^{\mathsf{CNOT}}, \beta^{\mathsf{H}}, \beta^{\mathsf{T}}$ on locations corresponding to our implementations of $\mathsf{CNOT}, \mathsf{H}, \mathsf{T}$, we can write $\widetilde{\gamma} = \{|\widetilde{\Phi}_{\mathbb{i},c}\rangle\}$ for some $|\widetilde{\Phi}_{\mathbb{i},c}\rangle$ that yields deterministic measurement outcome c. By an

induction proof, together with the correctness of circuits 1, 2, 3, the basis $\{|\widetilde{\Phi}_{\mathbb{i},c}\rangle\}$, and the correct computation of delayed Pauli corrections, we have the identity

$$|\phi\rangle \otimes |\psi_{\mathsf{PLM}}\rangle = \sum_c \alpha_{\mathbb{i},c} |\widetilde{\Phi}_{\mathbb{i},c}\rangle \otimes P_{h(\mathbb{i},c)} \widehat{Q} |\phi, \mathbb{i}\rangle \tag{2}$$

for some coefficients $\{\alpha_{\mathbb{i},c}\}$ that do not depend on $|\phi\rangle$. Recall that $P_{h(\mathbb{i},c)}$ is the correction Pauli computed from the classical input $\mathbb{i}$ and the measurement result c. Note also that $\widehat{Q} |\phi, \mathbb{i}\rangle = U_{Q,\mathbb{i}} |\phi\rangle \otimes |\mathbb{i}\rangle$ where $U_{Q,\mathbb{i}}$ is defined in the theorem statement.

The compilation algorithm ends with providing measurement instructions on remaining wires and defining the function $g(\mathbb{i}, r)$ that computes the classical output.

- Standard-basis measurement on wire i:
  - Append $(d_i : v \mapsto v_i,\ \theta^*, G^*)$ to the instruction of the LM quantum program.
- Suppose the original circuit uses the measurement results on wires $w_1, \ldots, w_{n'}$ as the classical output. Say they are the $e_1, \ldots, e_{n'}$-th measurements in the constructed LM quantum program. Then the algorithm defines the output of $g(\mathbb{i}, r)$ as $(r_{e_1} \oplus x_{w_1}, \ldots, r_{e_i} \oplus x_{w_i})$ where the values $(x_{w_1}, \ldots, x_{w_i})$ are computed using h.

Clearly, tensoring the orthonormal basis $\widetilde{\gamma}$ with the standard basis on the remaining wires yields an orthonormal basis $\{|\Phi_{\mathbb{i},r}\rangle\}_r$ on which the adaptive measurements are deterministic. Hence, the compiled LM quantum program is projective. It is clear by construction that $|\psi_{\mathsf{PLM}}|, t = O(m)$. Consider any pure state $\sum_j |\psi_j\rangle \otimes |\phi_j\rangle$ where $|\psi_j\rangle$ may not be normalized. Since (2) holds for every $|\phi\rangle$, we have

$$\begin{aligned}
&\sum_{y,j} \mathsf{X}^y |\psi_j\rangle \otimes \left(U_{Q,\mathbb{i}}^\dagger (|y\rangle\langle y| \otimes I) U_{Q,\mathbb{i}}\right) |\phi_j\rangle \otimes |\psi_{\mathsf{PLM}}\rangle \\
=&\sum_{y,j} \mathsf{X}^y |\psi_j\rangle \otimes \sum_c \alpha_{\mathbb{i},c} |\widetilde{\Phi}_{\mathbb{i},c}\rangle \otimes P_{h(\mathbb{i},c)} \left(U_{Q,\mathbb{i}} \left(U_{Q,\mathbb{i}}^\dagger (|y\rangle\langle y| \otimes I) U_{Q,\mathbb{i}}\right) |\phi_j\rangle \otimes |\mathbb{i}\rangle\right) \\
=&\sum_{y,j} \mathsf{X}^y |\psi_j\rangle \otimes \sum_c \alpha_{\mathbb{i},c} |\widetilde{\Phi}_{\mathbb{i},c}\rangle \otimes P_{h(\mathbb{i},c)} \left((|y\rangle\langle y| \otimes I) U_{Q,\mathbb{i}} |\phi_j\rangle \otimes |\mathbb{i}\rangle\right) \\
=&\sum_{y,z,j} \mathsf{X}^{g(\mathbb{i},(c,y,z))} |\psi_j\rangle \otimes \sum_c \alpha_{\mathbb{i},c} |\widetilde{\Phi}_{\mathbb{i},c}\rangle \otimes (|y\rangle\langle y| \otimes |z\rangle\langle z|) P_{h(\mathbb{i},c)} \left(U_{Q,\mathbb{i}} |\phi_j\rangle \otimes |\mathbb{i}\rangle\right) \\
=&\sum_{y',j} \mathsf{X}^{y'} |\psi_j\rangle \otimes \left(\sum_{r:g(\mathbb{i},r)=y'} |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|\right) \sum_c \alpha_{\mathbb{i},c} |\widetilde{\Phi}_{\mathbb{i},c}\rangle \otimes P_{h(\mathbb{i},c)} \left(U_{Q,\mathbb{i}} |\phi_j\rangle \otimes |\mathbb{i}\rangle\right) \\
=&\sum_{y,j} \mathsf{X}^{y} |\psi_j\rangle \otimes \left(\sum_{r:g(\mathbb{i},r)=y} |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|\right) \left(|\phi_j\rangle \otimes |\psi_{\mathsf{PLM}}\rangle\right)
\end{aligned}$$

where the third equality follows because $g(\mathbb{i}, (c, y, z))$ outputs the XOR of y with the x-correction specified by $h(\mathbb{i}, c)$ on wires that $|y\rangle\langle y|$ took place, the fourth equality follows from setting $y' = g(\mathbb{i}, r)$, $r = (c, y, z)$ and that $|\Phi_{\mathbb{i},r}\rangle = |\widetilde{\Phi}_{\mathbb{i},c}\rangle \otimes |y\rangle \otimes |z\rangle$. This implies that

$$\sum_y \mathsf{X}^y \otimes \left(U_{Q,\mathbb{i}}^\dagger (|y\rangle\langle y| \otimes I) U_{Q,\mathbb{i}}\right) \otimes |\psi_{\mathsf{PLM}}\rangle = \sum_{y'} \mathsf{X}^y \otimes \left(\sum_{r:g(\mathbb{i},r)=y} |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|\right) (I \otimes |\psi_{\mathsf{PLM}}\rangle)$$

Let (ρ, τ) be any state. Applying the above two operators with the first register being $0^{n'}$ and the second register being ρ, we get $\{(\tau, y) \mid y \leftarrow Q(\mathbb{i}, \rho)\} = \{(\tau, y) \mid y \leftarrow P(\mathbb{i}, \rho)\}$. □

6 Approximately Unitary Quantum Program

This section investigates the power of white-box access to quantum programs (ψ, Q) that approximately implements some unitary transformation $\mathcal{U} : \rho \mapsto U\rho U^\dagger$.

Definition 6.1. A quantum program (ψ, Q) is said to be an ϵ-approximation of a unitary transformation $\mathcal{U}$ if

$$\|Q(\cdot \otimes \psi) - \mathcal{U}(\cdot)\|_\diamond \leq \epsilon.$$

The following lemma shows a sufficient condition for ϵ-approximation of a unitary.

Lemma 6.2. *Let (ψ, Q) be a quantum program and $\mathcal{U}$ be a unitary transformation such that $\|Q(\rho \otimes \psi) - \mathcal{U}(\rho)\|_1 \leq \epsilon^2/2$ for every input state ρ. Then (ψ, Q) ϵ-approximates $\mathcal{U}$.*

Proof. This follows directly from the first item of Lemma 3.8. □

Definition 6.3 (Approximately Unitary Quantum Programs)**.** The class of approximately unitary quantum programs is defined as the set of all sequences of quantum programs $(\psi_\lambda, Q_\lambda)$ that are $\mathrm{negl}(\lambda)$-approximations of some unitary transformations $\mathcal{U}_\lambda$.

We often make the dependency on λ implicitly. By Lemma 6.2, (ψ, Q) is an approximately unitary quantum program if for every state ρ, it holds that $Q(\rho \otimes \psi) \approx_{\mathrm{negl}(\lambda)} \mathcal{U}(\rho)$.

We have been considering quantum programs (ψ, Q) whose output space has the same dimension as the input space. Without loss of generality, we will assume that the quantum circuit Q operates as first taking the input and auxiliary registers, applying some gates $G_1, \ldots, G_k$ in sequence, and then tracing out the auxiliary register. Let $U_Q = G_k \cdots G_1$ be the associated unitary operator and $\mathcal{U}_Q : \rho \mapsto U_Q\, \rho\, U_Q^\dagger$ be the associated unitary transformation. The following lemma shows that accessing the code of a quantum program that approximates a unitary enables other computation beyond the implemented unitary itself.

Lemma 6.4. *Let (ψ, Q) be an ϵ-approximation of a unitary transformation $\mathcal{U}(\cdot) = U(\cdot)U^\dagger$. Let U_Q act on registers B, E for placing the input state and the auxiliary state ψ respectively. Let A be a unitary operator on registers R, B. Let C be a register for placing a control qubit. Set ${W_Q}^{C,R,B,E} := \mathsf{ctrl}^C\text{-}\left({U_Q^\dagger}^{B,E} A^{R,B} {U_Q}^{B,E}\right)$ and $W^{C,R,B} := \mathsf{ctrl}^C\text{-}\left({U^\dagger}^B A^{R,B} U^B\right)$. Then*

$$\left\| W_Q(\cdot \otimes \psi^E) W_Q^\dagger - \left(W(\cdot)W^\dagger \otimes \psi^E\right) \right\|_\diamond \leq O(\sqrt{\epsilon}).$$

Proof. Suppose ψ is a pure state. The quantum program (ψ, Q) computes $\mathrm{Tr}_E\left(\mathcal{U}_Q(\cdot \otimes \psi^E)\right)$ and is an ϵ-approximation of $\mathcal{U}(\cdot) = \mathrm{Tr}_E\left(\mathcal{U}(\cdot) \otimes \psi^E\right)$, which means that

$$\left\| \mathrm{Tr}_E\left(\mathcal{U}_Q(\cdot \otimes \psi^E)\right) - \mathrm{Tr}_E\left(\mathcal{U}(\cdot) \otimes \psi^E\right) \right\|_\diamond \leq \epsilon$$

By Theorem 3.9, there exists a unitary operator V on E such that

$$\left\| {U_Q}^{B,E}\left(\cdot \otimes |\psi\rangle^E\right) - U^B \otimes V^E\left(\cdot \otimes |\psi\rangle^E\right) \right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$$

By setting $|\psi'\rangle = V|\psi\rangle$, we have

$$\left\| {U_Q}^{B,E}\left(\cdot \otimes |\psi\rangle^E\right) - \left(U^B(\cdot) \otimes |\psi'\rangle^E\right)\right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$$

Since composing with unitary operators preserves operator norm, we have

$$\left\| \left(\cdot \otimes |\psi\rangle^E\right) - {U_Q^\dagger}^{B,E}\left(U^B(\cdot) \otimes |\psi'\rangle^E\right)\right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$$

$$\left\| \left({U^\dagger}^B A^{R,B} U^B(\cdot) \otimes |\psi\rangle^E\right) - {U_Q^\dagger}^{B,E}\left(A^{R,B}U^B(\cdot) \otimes |\psi'\rangle^E\right)\right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$$

By the triangle inequality, we then obtain

$$\left\| \left({U^\dagger}^B A^{R,B} U^B(\cdot) \otimes |\psi\rangle^E\right) - {U_Q^\dagger}^{B,E} A^{R,B} {U_Q}^{B,E}\left(\cdot \otimes |\psi\rangle^E\right)\right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$$

Since both terms within the norm start with tensoring $|\psi\rangle$, we have

$$\left\| \left(\mathsf{ctrl}^C\text{-}\left({U^\dagger}^B A^{R,B} U^B\right)(\cdot) \otimes |\psi\rangle^E\right) - \left(\mathsf{ctrl}^C\text{-}\left({U_Q^\dagger}^{B,E} A^{R,B} {U_Q}^{B,E}\right)\right)\left(\cdot \otimes |\psi\rangle^E\right)\right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$$

By definition, this is equivalent to $\left\| {W}^{C,R,B}(\cdot) \otimes |\psi\rangle^E - {W_Q}^{C,R,B,E}\left(\cdot \otimes |\psi\rangle^E\right)\right\|_{\mathsf{op}} \leq O(\sqrt{\epsilon})$. Applying Lemma 3.10, we have

$$\left\| W_Q(\cdot \otimes \psi^E) W_Q^\dagger - \left(W(\cdot)W^\dagger \otimes \psi^E\right)\right\|_\diamond \leq O(\sqrt{\epsilon})$$

The last inequality holds even if ψ is not a pure state because we can first obtain the inequality for the purification of ψ and then use the property that taking partial trace does not increase the diamond norm to derive the inequality that refers only to ψ. □

The following lemma shows that a quantum program approximating a unitary allows repeated evaluation of the unitary and its inverse.

Lemma 6.5. *There is an efficient classical algorithm $\mathcal{T}$ that, given quantum circuits Q', Q of the quantum programs $(\psi', Q'), (\psi, Q)$ where*

- *Q' uses the gate set[13] $\mathcal{G}' \cup \{U, U^\dagger\}$ and applies the gates $U, U^\dagger$ at most q times,*
- *Q uses the gate set $\mathcal{G}$ and (ψ, Q) is an ϵ-approximation of $\mathcal{U} : \rho \mapsto U\rho U^\dagger$,*

$\mathcal{T}$ outputs a quantum circuit Q'' using the gate set $\mathcal{G}' \cup \mathcal{G} \cup \{\mathsf{SWAP}\}$ such that with the auxiliary state $\psi'' = \psi' \otimes \psi \otimes |0\rangle\langle 0|^{\otimes n}$ where n is the input length of U, we have

$$\|Q''(\cdot \otimes \psi'') - Q'(\cdot \otimes \psi')\|_\diamond \leq O\left(q\sqrt{\epsilon}\right)$$

Proof. Let $G_1, \ldots, G_k \in \mathcal{G}$ be the gates that Q applies in sequence and set $U_Q = G_k \cdots G_1$. The algorithm $\mathcal{T}$ is described as follows:

[13] We assume that every gate set under consideration is closed under inversion.

1. Take the circuit Q', which uses the gate set $\mathcal{G}' \cup \{U, U^\dagger\}$ and specifies a register for ψ'.
2. Specify an additional register E for storing ψ and a register B for storing $|0\rangle\langle 0|^{\otimes n}$.
3. Replace each U gate (say on R) with $(G_1^\dagger \cdots G_k^\dagger)^{B,E} \mathsf{SWAP}^{B,R} (G_k \cdots G_1)^{B,E} \mathsf{SWAP}^{B,R}$.
4. Replace each $U^\dagger$ gate (say on R) with $\mathsf{SWAP}^{B,R} (G_1^\dagger \cdots G_k^\dagger)^{B,E} \mathsf{SWAP}^{B,R} (G_k \cdots G_1)^{B,E}$.
5. Output the resulting circuit Q''.

The resulting circuit Q'' uses the gate set $\mathcal{G}' \cup \mathcal{G} \cup \{\mathsf{SWAP}\}$ by construction. Now we analyze the behavior of the quantum program (ψ'', Q''). Consider the circuit $\widetilde{Q}$ generated similarly as Q'' except that

- In step 3, replace each U gate (say on R) with ${U^\dagger}^B \mathsf{SWAP}^{B,R} U^B \mathsf{SWAP}^{B,R}$ instead.
- In step 4, replace each $U^\dagger$ gate (say on R) with $\mathsf{SWAP}^{B,R} {U^\dagger}^B \mathsf{SWAP}^{B,R} U^B$ instead.

Define $W_Q := (G_1^\dagger \cdots G_k^\dagger)^{B,E} \mathsf{SWAP}^{B,R} (G_k \cdots G_1)^{B,E}$ and $W := {U^\dagger}^B \mathsf{SWAP}^{R,B} U^B$. Applying Lemma 6.4 with the control qubit being $|1\rangle$ and $A = \mathsf{SWAP}$, we have

$$\left\| W_Q(\cdot \otimes \psi^E) W_Q^\dagger - \left(W(\cdot) W^\dagger \otimes \psi^E\right) \right\|_\diamond \leq O(\sqrt{\epsilon}).$$

By the triangle inequality and noting that ψ'' initializes ψ on register E, we have

$$\left\| Q''(\cdot \otimes \psi'') - \widetilde{Q}(\cdot \otimes \psi'') \right\|_\diamond \leq O(q\sqrt{\epsilon}).$$

Since ${U^\dagger}^B \mathsf{SWAP}^{B,R} U^B \mathsf{SWAP}^{B,R} = {U^\dagger}^B U^R$ and $\mathsf{SWAP}^{B,R} {U^\dagger}^B \mathsf{SWAP}^{B,R} U^B = {U^\dagger}^R U^B$, the program $\widetilde{Q}(\cdot \otimes \psi'')$ only differs from the original program $Q'(\cdot \otimes \psi')$ by having additional tensored auxiliary registers E, B and extra computation on B. These additional registers are not used for the output, so the two programs implement the same quantum mapping. Thus, we conclude that

$$\| Q''(\cdot \otimes \psi'') - Q'(\cdot \otimes \psi') \|_\diamond \leq O(q\sqrt{\epsilon})$$

□

We summarize the capability of approximately unitary quantum programs as follows.

Corollary 6.6. *Let quantum program (ψ, Q) be a $\mathrm{negl}(\lambda)$-approximation of $\mathcal{U} : \rho \mapsto U\rho U^\dagger$. Then the quantum program can be used to apply U and $U^\dagger$ for a polynomial number of times with negligible error in diamond distance. Moreover, suppose the gate set $\mathcal{G}$ used by Q satisfy that for every $G \in \mathcal{G}$, the operator ctrl-G can be efficiently implemented from $\mathcal{G}$. Then for every operator A that can be efficiently implemented from $\mathcal{G}$, one can efficiently construct a quantum program that $\mathrm{negl}(\lambda)$-approximates ctrl-$(U^\dagger A U)$ using the quantum program (ψ, Q).*

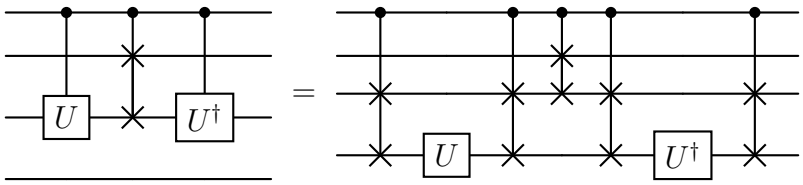
The following lemmas show how ctrl-$(U^\dagger A U)$ is connected to $U, U^\dagger$ through black-box reductions. Note that black-box access to a unitary operator U is equivalent to black-box access to its associated unitary transformation $\mathcal{U} : \rho \mapsto U\rho U^\dagger$.

Lemma 6.7. *Let U be a unitary operator on n qubits and A be a unitary operator. Then* $\mathsf{ctrl}\text{-}(U^\dagger A U)$ *can be computed from* $\mathsf{ctrl}\text{-}(U^\dagger \mathsf{SWAP}^{\otimes n} U)$ *and* $\mathsf{ctrl}\text{-}A$ *in a black-box manner.*

Proof. Observe that $\left(U^{\dagger B}\mathsf{SWAP}^{\otimes n B,C}U^B\right) A^{C,D} \left(U^{\dagger B}\mathsf{SWAP}^{\otimes n B,C}U^B\right) = U^{\dagger B}A^{B,D}U^B$ holds. The lemma follows from taking the controlled version of the identity. □

Lemma 6.8. $\mathsf{ctrl}\text{-}(U^\dagger \mathsf{SWAP}^{\otimes n} U)$ *can be computed from U and $U^\dagger$ in a black-box manner.*

Proof. $\mathsf{ctrl}\text{-}(U^\dagger \mathsf{SWAP}^{\otimes n} U)$ can be computed by the following circuit.



□

7 Quantum State Obfuscation

We now present an obfuscation scheme for quantum programs that approximately implement unitary transformations. This setting captures a broad and expressive class of quantum functionalities while supporting program reusability. We begin by formalizing the notion of quantum state ideal obfuscation in the classical oracle model, followed by the description of our construction. The security of our obfuscation scheme critically relies on the functional authentication scheme and PLM quantum programs developed in previous sections.

Definition 7.1 (Quantum State Obfuscation)**.** A *quantum state ideal obfuscation* for the class of approximately-unitary quantum programs in the classical oracle model is a pair of QPT algorithms $(\mathsf{QObf}, \mathsf{QEval})$ with the following syntax:

- $\mathsf{QObf}(1^\lambda, (\psi, Q)) \to (\widetilde{\psi}, \mathsf{F})$: The obfuscator takes as input the security parameter 1^λ and a quantum program (ψ, Q) in the plain model[14], and outputs an obfuscated program specified by a state $\widetilde{\psi}$ and a classical function F.
- $\mathsf{QEval}^{\mathsf{F}}(\rho_{\mathrm{in}}, \widetilde{\psi}) \to \rho_{\mathrm{out}}$: The evaluation algorithm executes the obfuscated program on quantum input ρ_{in} by making use of the state $\widetilde{\psi}$ and making superposition queries to the classical oracle F, and produces the quantum output ρ_{out}.

These algorithms have to satisfy the following properties.

- **Functionality-Preserving**: For every quantum program (ψ, Q) which is an $\mathrm{negl}(\lambda)$-approximation of some unitary transformation $\mathcal{U}$, the quantum program

$$\mathop{\mathbb{E}}_{(\widetilde{\psi},\mathsf{F})\leftarrow\mathsf{QObf}(1^\lambda,(\psi,Q))} \left(\widetilde{\psi},\ \mathsf{QEval}^{\mathsf{F}}\right)$$

  is also a $\mathrm{negl}(\lambda)$-approximation of $\mathcal{U}$. Note that our convention for classical oracles (Section 3) ensures that $\mathsf{ctrl}\text{-}\mathsf{F}$ can be efficiently computed by querying F. Therefore, the resulting quantum program would satisfy all properties described in Corollary 6.6.

[14] Q is described without reference to oracles, so as to circumvent the impossibility result of [BGI+01].

- **Ideal Obfuscation**: There exists a stateful[15] QPT simulator Sim such that for every QPT adversary $\mathcal{A}$ and quantum program (ψ, Q) that $\mathrm{negl}(\lambda)$-approximates some unitary transformation $\mathcal{U}$ associated with a unitary operator U,

$$\left| \Pr_{(\widetilde{\psi},\mathsf{F})\leftarrow\mathsf{QObf}(1^\lambda,(\psi,Q))}\left[1 \leftarrow \mathcal{A}^{\mathsf{F}}\left(\widetilde{\psi}\right)\right] - \Pr_{(\widetilde{\psi},\mathsf{F}_{\mathsf{Sim}})\leftarrow\mathsf{Sim}(1^\lambda,1^n,1^m)}\left[1 \leftarrow \mathcal{A}^{\mathsf{F}_{\mathsf{Sim}}^{\mathsf{ctrl}\text{-}(U^\dagger\cdot U)}}\left(\widetilde{\psi}\right)\right]\right| = \mathrm{negl}(\lambda)$$

  Here, n is the input length of the quantum program, m is the size of the quantum program, and $\mathsf{F}_{\mathsf{Sim}}$ makes black-box oracle access to $\mathsf{ctrl}\text{-}(U^\dagger A U)$ for some unitary A that can be efficiently implemented.

Remark 1. *While our definition allows A to be chosen freely, one can also set $A = \mathsf{SWAP}^{\otimes n}$ explicitly without loss of generality by Lemma 6.7, where n is the program's input length. In this case, the definition of ideal obfuscation admits an intuitive interpretation: the obfuscated program only enables the controlled computation $\mathsf{ctrl}\text{-}(U^\dagger\mathsf{SWAP}^{\otimes n}U)$ that corresponds to the combined operation of applying U, swapping the result with any state, and applying $U^\dagger$.*

Remark 2. *An alternative way of defining ideal obfuscation is to require that $\mathsf{F}_{\mathsf{Sim}}$ makes black-box oracle access to $\mathcal{U}$ and $\mathcal{U}^\dagger$ instead. This definition is automatically satisfied by our Definition 7.1, according to Lemmas 6.7 and 6.8. This definition also provides an intuitive interpretation: the obfuscated program only enables black-box computation of $\mathcal{U}$ and $\mathcal{U}^\dagger$.*

Theorem 7.2 (Main theorem)**.** *There exists a quantum state ideal obfuscation for the class of approximately-unitary quantum programs with quantum inputs and outputs in the classical oracle model, assuming post-quantum one-way functions.*

Remark 3. *Similar to [BBV24], the assumption of a post-quantum one-way function can be replaced with a truly random function (or a random oracle) if the classical oracle is not required to be efficiently computable.*

Remark 4. *We also improve the tolerated approximation error when obfuscating ε-pseudo-deterministic quantum programs. [BBV24] only handles the case $\varepsilon = 2^{-2n}\,\mathrm{negl}(\lambda)$, whereas our result applies whenever $\varepsilon = \mathrm{negl}(\lambda)$. This follows from our main theorem and Lemma 6.2, which shows that any $\mathrm{negl}(\lambda)$-pseudo-deterministic quantum program is also an $\mathrm{negl}(\lambda)$-approximation of the unitary associated with the classical function it implements.*

7.1 Construction

Our obfuscation scheme is constructed using the compilation algorithm of Theorem 5.3, the quantum authentication scheme introduced in Section 4, a tokenized signature scheme (Section 3.4), and a post-quantum PRF with $\kappa = O(\lambda n)$ bits of output (Section 3.5).

Without loss of generality, the input of our obfuscation scheme will be a quantum program $(\psi, Q^{\mathsf{univ}})$ where Q^{univ} is a universal circuit from the universal gate set $\mathscr{C}_2 \cup \{\mathsf{T}\}$ and the output qubits will be placed at the same position as the input qubits. The state ψ encodes the actual content of the program. From now on, we will omit writing Q^{univ} as part of the input arguments to QObf. Using Q^{univ}, we construct the following quantum circuit $\widetilde{Q}^{\mathsf{univ}}$ with classical outputs:

[15] The simulator can produce an oracle that carries a quantum state across queries as auxiliary input.

1. Take a quantum state on register $V = (V_{\mathrm{in}}, V_{\mathrm{out}}, V_{\mathsf{aux}})$, of length n, n, m respectively.
2. Take a classical input $\mathbb{i} \in \{0,1\}^{2n}$.
3. Apply the teleportation receiving operation $\mathsf{TP.Recv}(P_{\mathbb{i}}, V_{\mathrm{in}})$.
4. Apply Q^{univ} on $(V_{\mathrm{in}}, V_{\mathsf{aux}})$.
5. Perform quantum teleportation $\mathsf{TP.Send}(V_{\mathrm{in}}, V_{\mathrm{out}})$ and output its outcome.

Applying Theorem 5.3, we compile $\widetilde{Q}^{\mathsf{univ}}$ into a PLM quantum program P with auxiliary state ψ_{PLM} (on some register V_{PLM}) and instructions $\{f_j, \theta_j, G_j\}_{j\in[t]}$ with $|\psi_{\mathsf{PLM}}|, t = O(m)$. Our obfuscation scheme is described as follows.

Construction of Quantum State Obfuscation

$\mathsf{QObf}\left(1^\lambda, \psi\right)$:

- Generate input EPR pairs $(\mathsf{epr}_{(\mathrm{in},\mathsf{pub})}, \mathsf{epr}_{(\mathrm{in},\mathsf{priv})})$ of length n.
  Generate output EPR pairs $(\mathsf{epr}_{(\mathrm{out},\mathsf{priv})}, \mathsf{epr}_{(\mathrm{out},\mathsf{pub})})$ of length n.
- Prepare the auxiliary state ψ_{PLM} for the PLM quantum program.
- Sample an authentication key $k \leftarrow \mathsf{Auth.KeyGen}(1^{O(\lambda+m)}, 1^{O(m)})$.
  Compute the authenticated state $\psi_{\mathsf{auth}}{}^{\widetilde{V}} \leftarrow \mathsf{Auth.Enc}_k((\mathsf{epr}_{(\mathrm{in},\mathsf{priv})}, \mathsf{epr}_{(\mathrm{out},\mathsf{priv})}, \psi, \psi_{\mathsf{PLM}}))$.
- Sample a signature token $(\mathsf{vk}, \tau_{\mathsf{token}}) \leftarrow \mathsf{Token.Gen}(1^\lambda, 1^{2n})$.
- Sample a PRF key $k_0 \leftarrow \{0,1\}^\lambda$ and set $H \equiv \mathrm{PRF}_{k_0} : \{0,1\}^* \to \{0,1\}^\kappa$.
- Define the classical function $\mathsf{F}\left(j, \widetilde{v}, \mathbb{i}, s, \ell_1, \ldots, \ell_t\right)$ where $j \in [t]$:
  1. Compute $v \leftarrow \mathsf{Auth.Dec}_{k,\theta_j,G_j}(\widetilde{v})$. If it rejects, output $\perp$.
  2. Compute $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$. If it rejects, output $\perp$.
  3. For $i = 1, \ldots, j-1$, reconstruct r_i as follows:
     - Set $r_i \in \{0,1\}$ so that $\ell_i = H(i, r_i, \mathbb{i}, s)$.
     - If there are two or no solutions for r_i, output $\perp$.
  4. Compute $r_j = f_j^{\mathbb{i}, r_1, \ldots, r_{j-1}}(v)$.
  5. Output $\begin{cases} \text{the label } H(j, r_j, \mathbb{i}, s) & \text{if } j < t. \\ \text{the value } g(\mathbb{i}, r_1, \ldots, r_t) & \text{if } j = t. \end{cases}$
- Output $\widetilde{\psi} = (\psi_{\mathsf{auth}}{}^{\widetilde{V}}, \mathsf{epr}_{(\mathrm{in},\mathsf{pub})}, \mathsf{epr}_{(\mathrm{out},\mathsf{pub})}, \tau_{\mathsf{token}})$ and the classical function F.

$\mathsf{QEval}^{\mathsf{F}}\left(\rho_{\mathrm{in}}, \widetilde{\psi}\right)$:

- Parse $\widetilde{\psi} = (\psi_{\mathsf{auth}}, \mathsf{epr}_{(\mathrm{in},\mathsf{pub})}, \mathsf{epr}_{(\mathrm{out},\mathsf{pub})}, \tau_{\mathsf{token}})$.
- Perform quantum teleportation, obtaining the result $\mathbb{i} \leftarrow \mathsf{TP.Send}(\rho_{\mathrm{in}}, \mathsf{epr}_{(\mathrm{in},\mathsf{pub})})$.
- Sign the teleportation result, obtaining its signature $s \leftarrow \mathsf{Token.Sign}\left(\mathbb{i}, \tau_{\mathsf{token}}\right)$.
- Place $\widetilde{\psi}$ on register $\widetilde{V}$ and do the following iteratively for $j = 1, \ldots, t$:
  - Apply $\mathsf{H}^{\widetilde{\theta}_j}\widetilde{G_j}$ on register $\widetilde{V}$.
  - Measure the outcome $\ell_j \leftarrow \mathsf{F}\left(j, \widetilde{V}, \mathbb{i}, s, \ell_1, \ldots, \ell_{j-1}, 0^\kappa, \ldots, 0^\kappa\right)$.
  - Apply $\left(\mathsf{H}^{\widetilde{\theta}_j}\widetilde{G_j}\right)^\dagger$ on register $\widetilde{V}$.
- Output $\rho_{\mathrm{out}} \leftarrow \mathsf{TP.Recv}(\ell_t, \mathsf{epr}_{(\mathrm{out},\mathsf{pub})})$.

7.2 Security

We prove Theorem 7.2 by showing that Construction 7.1 is a quantum state ideal obfuscation for approximately-unitary quantum programs. We start by constructing the following hybrids, where we may change the way we describe quantumly accessible oracles, from a purely classical interface to a partially quantum interface. For example, outputting the classical value x to the output register Y will be written as $\mathsf{Write}(x)^Y$. Furthermore, all "If–Then" expressions are to be interpreted as describing quantum controlled operations, whereas all measurements are to be applied coherently.

We extensively use the notations $\mathsf{WriteH}(x)^{Y,D}$, $\mathsf{EqualH}(x)^{Y,D}$, $|\mathcal{L}_\emptyset\rangle$, Has_S, Only_S introduced in Section 3.3 to handle purified random oracles. We also make the following abbreviations: $\mathsf{Has}_{(i,b,\mathbb{i},s)} := \mathsf{Has}_{\{(i,b,\mathbb{i},s)\}}$, $\mathsf{Has}_{\neq\mathbb{i}} := \mathsf{Has}_{\{(i,b,\mathbb{i}',s)|i,b,s,\mathbb{i}'\neq\mathbb{i}\}}$, $\mathsf{Only}_{\mathbb{i}} := \mathsf{Only}_{\{(i,b,\mathbb{i},s)|i,b,s\}}$.

$\mathsf{QObf}(1^\lambda,\psi)$: Output $\widetilde{\psi}$ and the oracle F.

$\mathsf{Hyb}_1(1^\lambda,\psi)$:

- Generate input EPR pairs $(\mathsf{epr}_{(\mathrm{in},\mathsf{pub})},\mathsf{epr}_{(\mathrm{in},\mathsf{priv})})$ of length n.
  Generate output EPR pairs $(\mathsf{epr}_{(\mathrm{out},\mathsf{priv})},\mathsf{epr}_{(\mathrm{out},\mathsf{pub})})$ of length n.
- Sample keys $k,(\mathsf{vk},\tau_{\mathsf{token}}),k_0\leftarrow\{0,1\}^\lambda$ and set $H(\cdot)\equiv\mathrm{PRF}_{k_0}(\cdot)$ as in QObf.
- Set $\psi_{\mathsf{auth}}{}^{\widetilde{V}}\leftarrow\mathsf{Auth.Enc}_k(0^{O(m)})$ and $V\leftarrow(\mathsf{epr}_{(\mathrm{in},\mathsf{priv})},\mathsf{epr}_{(\mathrm{out},\mathsf{priv})},\psi,\psi_{\mathsf{PLM}})$.
- Define the oracle $\mathsf{F}_1\,(j,\widetilde{v},\mathbb{i},s,\ell_1,\ldots,\ell_t,Y)^V$ where Y denotes the output register:
  1. Compute $\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})$. If it rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  2. Compute $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i},s)$. If it rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  3. For $i=1,\ldots,j-1$, reconstruct r_i as follows:
     - Set $r_i\in\{0,1\}$ so that $\ell_i=H(i,r_i,\mathbb{i},s)$.
     - If there are two or no solutions for r_i, apply $\mathsf{Write}(\bot)^Y$ and return.
  4. Compute $r_j\leftarrow\mathcal{M}\left[f_j^{\mathbb{i},r_1,\ldots,r_{j-1}},\theta_j,G_j\right](V)$.
  5. Apply $\begin{cases}\mathsf{Write}(H(j,r_j,\mathbb{i},s))^Y & \text{if } j<t.\\ \mathsf{Write}(g(\mathbb{i},r_1,\ldots,r_t))^Y & \text{if } j=t.\end{cases}$
- Output $\widetilde{\psi}=(\psi_{\mathsf{auth}}{}^{\widetilde{V}},\mathsf{epr}_{(\mathrm{in},\mathsf{pub})},\mathsf{epr}_{(\mathrm{out},\mathsf{pub})},\tau_{\mathsf{token}})$ and the oracle F_1.

$\mathsf{Hyb}_2(1^\lambda,\psi)$:

- Generate $\widetilde{\psi},k,\mathsf{vk}$ and initialize $V\leftarrow(\mathsf{epr}_{(\mathrm{in},\mathsf{priv})},\mathsf{epr}_{(\mathrm{out},\mathsf{priv})},\psi,\psi_{\mathsf{PLM}})$ as in Hyb_1.
- Initialize a database register $D\leftarrow|\mathcal{L}_\emptyset\rangle$.
- Define the oracle $\mathsf{F}_2\,(j,\widetilde{v},\mathbb{i},s,L_1,\ldots,L_t,Y)^{V,D}$ where Y denotes the output register:
  1. Compute $\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})$. If it rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  2. Compute $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i},s)$. If it rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  3. For $i=1,\ldots,j-1$, reconstruct r_i as follows:[16]

- Set $r_i = 0$ if $\mathsf{EqualH}(i, 0, \mathbb{i}, s)^{L_i, D}$ holds.
- Set $r_i = 1$ if $\mathsf{EqualH}(i, 1, \mathbb{i}, s)^{L_i, D}$ holds.
- If none or both of the above hold, apply $\mathsf{Write}(\bot)^{Y}$ and return.

4. Compute $r_j \leftarrow \mathcal{M}\left[f_j^{\mathbb{i}, r_1, \ldots, r_{j-1}}, \theta_j, G_j\right](V)$.
5. Apply $\begin{cases} \mathsf{WriteH}(j, r_j, \mathbb{i}, s)^{Y, D} & \text{if } j < t. \\ \mathsf{Write}(g(\mathbb{i}, r_1, \ldots, r_t))^{Y} & \text{if } j = t. \end{cases}$

- Output $\widetilde{\psi}$ and the oracle F_2.

$\mathsf{Hyb}_3(1^\lambda, \psi)$:

- Generate $\widetilde{\psi}, k, \mathsf{vk}$ and initialize V, D as in Hyb_2.
- Define the oracle $\mathsf{F}_3\,(j, \widetilde{v}, \mathbb{i}, s, L_1, \ldots, L_t, Y)^{V, D}$ where Y denotes the output register:

1. If $\mathsf{Auth.Ver}_{k, \theta_j, G_j}(\widetilde{v})$ rejects, apply $\mathsf{Write}(\bot)^{Y}$ and return.
2. If $\mathsf{Has}_{\neq \mathbb{i}}{}^{D}$ holds or $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$ rejects, apply $\mathsf{Write}(\bot)^{Y}$ and return.
3. For $i = 1, \ldots, j-1$, reconstruct r_i as follows:[17]
   - Set $r_i = 0$ if $\mathsf{EqualH}(i, 0, \mathbb{i}, s)^{L_i, D}$ holds.
   - Set $r_i = 1$ if $\mathsf{EqualH}(i, 1, \mathbb{i}, s)^{L_i, D}$ holds.
   - If none or both of the above hold, apply $\mathsf{Write}(\bot)^{Y}$ and return.
4. Compute $r_j \leftarrow \mathcal{M}\left[f_j^{\mathbb{i}, r_1, \ldots, r_{j-1}}, \theta_j, G_j\right](V)$.
5. Apply $\begin{cases} \mathsf{WriteH}(j, r_j, \mathbb{i}, s)^{Y, D} & \text{if } j < t. \\ \mathsf{Write}(g(\mathbb{i}, r_1, \ldots, r_t))^{Y} & \text{if } j = t. \end{cases}$

- Output $\widetilde{\psi}$ and the oracle F_3.

$\mathsf{Hyb}_4(1^\lambda, \psi)$:

- Generate $\widetilde{\psi}, k, \mathsf{vk}$ and initialize V, D as in Hyb_2.
- Define the oracle $\mathsf{F}_4\,(j, \widetilde{v}, \mathbb{i}, s, L_1, \ldots, L_t, Y)^{V, D}$ where Y denotes the output register:

1. If $\mathsf{Auth.Ver}_{k, \theta_j, G_j}(\widetilde{v})$ rejects, apply $\mathsf{Write}(\bot)^{Y}$ and return.
2. If $\mathsf{Has}_{\neq \mathbb{i}}{}^{D}$ holds or $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$ rejects, apply $\mathsf{Write}(\bot)^{Y}$ and return.
3. For $i = 1, \ldots, j-1$,
   - If none or both of the conditions $\mathsf{EqualH}(i, 0, \mathbb{i}, s)^{L_i, D}$ and $\mathsf{EqualH}(i, 1, \mathbb{i}, s)^{L_i, D}$ hold, apply $\mathsf{Write}(\bot)^{Y}$ and return.
   - Compute $r_i \leftarrow \mathcal{M}\left[f_i^{\mathbb{i}, r_1, \ldots, r_{i-1}}, \theta_i, G_i\right](V)$.
4. Compute $r_j \leftarrow \mathcal{M}\left[f_j^{\mathbb{i}, r_1, \ldots, r_{j-1}}, \theta_j, G_j\right](V)$.
5. Apply $\begin{cases} \mathsf{WriteH}(j, r_j, \mathbb{i}, s)^{Y, D} & \text{if } j < t. \\ \mathsf{Write}(g(\mathbb{i}, r_1, \ldots, r_t))^{Y} & \text{if } j = t. \end{cases}$

- Output $\widetilde{\psi}$ and the oracle F_4.

$\mathsf{Hyb}_5(1^\lambda, \psi)$:

- Generate $\widetilde{\psi}, k, \mathsf{vk}$ and initialize V, D as in Hyb_2.
- Define the oracle $\mathsf{F}_5\left(j, \widetilde{v}, \mathbb{i}, s, L_1, \ldots, L_t, Y\right)^{V,D}$ where Y denotes the output register:
  1. If $\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  2. If $\mathsf{Has}_{\neq \mathbb{i}}{}^D$ holds or $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  3. For $i = 1, \ldots, j-1$,
     - If none or both of the conditions $\mathsf{EqualH}(i, 0, \mathbb{i}, s)^{L_i,D}$ and $\mathsf{EqualH}(i, 1, \mathbb{i}, s)^{L_i,D}$ hold, apply $\mathsf{Write}(\bot)^Y$ and return.
  4. For $i = 1, \ldots, j$, compute $r_i \leftarrow \mathcal{M}\left[f_i^{\mathbb{i}, r_1, \ldots, r_{i-1}}, \theta_i, G_i\right](V)$.
  5. Apply $\begin{cases} \mathsf{WriteH}(j, 0, \mathbb{i}, s)^{Y,D} & \text{if } j < t. \\ \mathsf{Write}(g(\mathbb{i}, r_1, \ldots, r_t))^Y & \text{if } j = t. \end{cases}$
- Output $\widetilde{\psi}$ and the oracle F_5.

$\mathsf{Hyb}_6(1^\lambda, \psi)$:

- Generate $\widetilde{\psi}, k, \mathsf{vk}$ and initialize V, D as in Hyb_2.
- Define the oracle $\mathsf{F}_6\left(j, \widetilde{v}, \mathbb{i}, s, L_1, \ldots, L_t, Y\right)^{V,D}$ where Y denotes the output register:
  1. If $\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  2. If ~~$\mathsf{Has}_{\neq \mathbb{i}}{}^D$ holds or~~ $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  3. Whenever $\mathsf{EqualH}(i, 0, \mathbb{i}, s)^{L_i,D}$ fails for some $i < j$, apply $\mathsf{Write}(\bot)^Y$ and return.
  4. For $i = 1, \ldots, j$, compute $r_i \leftarrow \mathcal{M}\left[f_i^{\mathbb{i}, r_1, \ldots, r_{i-1}}, \theta_i, G_i\right](V)$.
  5. Apply $\begin{cases} \mathsf{WriteH}(j, 0, \mathbb{i}, s)^{Y,D} & \text{if } j < t. \\ \mathsf{Write}(g(\mathbb{i}, r_1, \ldots, r_t))^Y & \text{if } j = t. \end{cases}$
- Output $\widetilde{\psi}$ and the oracle F_6.

$\mathsf{Hyb}_7(1^\lambda, \psi)$:

- Generate $\widetilde{\psi}, k, \mathsf{vk}$ and initialize V, D as in Hyb_2.
- Parse $V = (V_{\text{in}}, V_{\text{out}}, V_{\mathsf{aux}}, V_{\mathsf{PLM}})$ as four parts.
- Define the oracle $\mathsf{F}_7\left(j, \widetilde{v}, \mathbb{i}, s, L_1, \ldots, L_t, Y\right)^{V,D}$ where Y denotes the output register:
  1. If $\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  2. If $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  3. Whenever $\mathsf{EqualH}(i, 0, \mathbb{i}, s)^{L_i,D}$ fails for some $i < j$, apply $\mathsf{Write}(\bot)^Y$ and return.
  4. ~~For $i = 1, \ldots, j$, compute $r_i \leftarrow \mathcal{M}\left[f_i^{\mathbb{i}, r_1, \ldots, r_{i-1}}, \theta_i, G_i\right](V)$.~~
  5. Apply $\begin{cases} \mathsf{WriteH}(j, 0, \mathbb{i}, s)^{Y,D} & \text{if } j < t. \\ \left(P_{\mathbb{i}}{}^{V_{\text{in}}} U^{\dagger}{}^{V_{\text{in}}} \mathsf{TP}^{\dagger}{}^{V_{\text{in}}, V_{\text{out}}}\right) \mathsf{Write}(V_{\text{in}}, V_{\text{out}})^Y \left(\mathsf{TP}^{V_{\text{in}}, V_{\text{out}}} U^{V_{\text{in}}} P_{\mathbb{i}}{}^{\dagger}{}^{V_{\text{in}}}\right) & \text{if } j = t. \end{cases}$

- Output $\widetilde{\psi}$ and the oracle F_7.

$\mathsf{Sim}(1^\lambda, 1^n, 1^m)$:

- Generate input EPR pairs $(\mathsf{epr}_{(\mathsf{in},\mathsf{pub})}, \mathsf{epr}_{(\mathsf{in},\mathsf{priv})})$ of length n.
  Generate output EPR pairs $(\mathsf{epr}_{(\mathsf{out},\mathsf{priv})}, \mathsf{epr}_{(\mathsf{out},\mathsf{pub})})$ of length n.
- Sample an authentication key $k \leftarrow \mathsf{Auth.KeyGen}(1^{O(\lambda+m)}, 1^{O(m)})$.
  Compute the authenticated state $\psi_{\mathsf{auth}}{}^{\widetilde{V}} \leftarrow \mathsf{Auth.Enc}_k(0^{O(m)})$.
- Sample a signature token $(\mathsf{vk}, \tau_{\mathsf{token}}) \leftarrow \mathsf{Token.Gen}(1^\lambda, 1^{2n})$.
- Sample a PRF key $k_0 \leftarrow \{0,1\}^\lambda$ and set $H \equiv \mathrm{PRF}_{k_0} : \{0,1\}^* \to \{0,1\}^\kappa$.
- Initialize registers $S_{\mathrm{in}} \leftarrow \mathsf{epr}_{(\mathsf{in},\mathsf{priv})}$ and $S_{\mathrm{out}} \leftarrow \mathsf{epr}_{(\mathsf{out},\mathsf{priv})}$.
- Define the oracle $\mathsf{F}_{\mathsf{Sim}}\,(j, \widetilde{v}, \mathbb{i}, s, \ell_1, \ldots, \ell_t, Y)^{S_{\mathrm{in}}, S_{\mathrm{out}}}$ which makes black-box queries to $\mathsf{ctrl}\text{-}\left(U^\dagger \left(\mathsf{TP}^\dagger\ \mathsf{Write}\ \mathsf{TP}\right) U\right)$, where Y denotes the output register:

  1. If $\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  2. If $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}, s)$ rejects, apply $\mathsf{Write}(\bot)^Y$ and return.
  3. Whenever $\ell_i \neq H(i, 0, \mathbb{i}, s)$ for some $i < j$, apply $\mathsf{Write}(\bot)^Y$ and return.
  4. Apply $\begin{cases} \mathsf{Write}(H(j,0,\mathbb{i},s))^Y & \text{if } j < t. \\ \left(P_{\mathbb{i}}{}^{S_{\mathrm{in}}} U^{\dagger S_{\mathrm{in}}} \mathsf{TP}^{\dagger S_{\mathrm{in}}, S_{\mathrm{out}}}\right) \mathsf{Write}(S_{\mathrm{in}}, S_{\mathrm{out}})^Y \left(\mathsf{TP}^{S_{\mathrm{in}}, S_{\mathrm{out}}} U^{S_{\mathrm{in}}} P_{\mathbb{i}}{}^{\dagger S_{\mathrm{in}}}\right) & \text{if } j = t. \end{cases}$

Let $\mathcal{A}$ be a quantum polynomial-time adversary throughout the following lemmas. The output of the adversary will be denoted as $\mathcal{A}^{\mathcal{O}}(\widetilde{\psi})$. For simplicity, we sometimes write $|\widetilde{\psi}\rangle$ to denote the purified state that includes $\widetilde{\psi}$ and the state on registers D, V.

Lemma 7.3. $\left\{\mathcal{A}^{\mathsf{F}}(\widetilde{\psi}) \middle| (\widetilde{\psi}, \mathsf{F}) \leftarrow \mathsf{QObf}(1^\lambda, \psi)\right\} \underset{2^{-\Omega(\lambda)}}{\approx} \left\{\mathcal{A}^{\mathsf{F}_1}(\widetilde{\psi}) \middle| (\widetilde{\psi}, \mathsf{F}_1) \leftarrow \mathsf{Hyb}_1(1^\lambda, \psi)\right\}$

Proof. The lemma follows directly from the functional security (Definition 4.1) of the authentication scheme (Theorem 4.3) since we have set its security parameter to be $O(\lambda + m)$. □

Lemma 7.4. $\left|\Pr_{(\widetilde{\psi},\mathsf{F}_1)\leftarrow\mathsf{Hyb}_1(1^\lambda,\psi)}\left[1 \leftarrow \mathcal{A}^{\mathsf{F}_1}(\widetilde{\psi})\right] - \Pr_{(\widetilde{\psi},\mathsf{F}_2)\leftarrow\mathsf{Hyb}_2(1^\lambda,\psi)}\left[1 \leftarrow \mathcal{A}^{\mathsf{F}_2}(\widetilde{\psi})\right]\right| = \mathrm{negl}(\lambda)$

Proof. The lemma follows from the computational indistinguishability between post-quantum PRFs and random oracles under polynomially-many quantum queries (Definition 3.6) and that random oracles can be perfectly simulated in a purified form (Section 3.3). □

Lemma 7.5. $\left\{\mathcal{A}^{\mathsf{F}_2}(\widetilde{\psi}) \middle| (\widetilde{\psi}, \mathsf{F}_2) \leftarrow \mathsf{Hyb}_2(1^\lambda, \psi)\right\} \underset{2^{-\Omega(\lambda)}}{\approx} \left\{\mathcal{A}^{\mathsf{F}_3}(\widetilde{\psi}) \middle| (\widetilde{\psi}, \mathsf{F}_3) \leftarrow \mathsf{Hyb}_3(1^\lambda, \psi)\right\}$

Proof. From the definitions of the hybrids, the oracles $\mathsf{F}_2, \mathsf{F}_3$ only differ on inputs that have overlapping with the 1-eigenspace of the projector

$$\Pi_{\mathsf{token}} := \sum_{\mathbb{i},s:\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i},s)=\top} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes |s\rangle\langle s|^S \otimes \mathsf{Has}_{\neq\mathbb{i}}{}^D.$$

[17] The operator is well-defined because $\mathsf{EqualH}(x)$ and $\mathsf{EqualH}(x')$ commute for every x, x' (Section 3.3).

By the oracle hybrid lemma (Lemma 3.12), it suffices to show that

$$\mathop{\mathbb{E}}_{(\widetilde{\psi},\mathsf{F}_3)\leftarrow\mathsf{Hyb}_3}\left\|\Pi_{\mathsf{token}}\mathcal{A}^{\mathsf{F}_3}|\widetilde{\psi}\rangle\right\| \leq 2^{-\Omega(\lambda)}.$$

Given any $\mathrm{poly}(\lambda)$-query adversary $\mathcal{A}$, let us consider the following algorithm $\mathcal{B}$ that attempts to break the unforgeability of the tokenized signature scheme (Definition 3.4).

1. Receive a challenge $(\mathsf{vk},\tau_{\mathsf{token}})$ from the unforgeability game.
2. Run $(\widetilde{\psi},\mathsf{F}_3)\leftarrow\mathsf{Hyb}_3(1^\lambda,\psi)$ but using the challenge $(\mathsf{vk},\tau_{\mathsf{token}})$ instead.
3. Run $\mathcal{A}^{\mathsf{F}_3}$ and measure register (I,S) in the standard basis to get $(\mathbb{i},s)$.
4. Measure registers $D\setminus D_{(\cdot,\cdot,\mathbb{i},\cdot)}$ in the Hadamard basis.
5. Output $(\mathbb{i},s,\mathbb{i}',s')$ if there is a measured entry $D_{(i',b',\mathbb{i}',s')}$ that is not $|+^\kappa\rangle$.

From the definition of F_3, applying $\mathcal{A}^{\mathsf{F}_3}$ will keep entries $D_{(i',b',\mathbb{i}',s')}$ in the state $|+^\kappa\rangle$ unless $\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i}',s')=\top$. Also, measuring if there is an entry in $D\setminus D_{(\cdot,\cdot,\mathbb{i},\cdot)}$ that is orthogonal to $|+^\kappa\rangle$ is equivalent to applying the projector ${\mathsf{Has}_{\neq\mathbb{i}}}^D$. Therefore, $\mathcal{B}$ wins the unforgeability game with probability at least $\mathbb{E}_{(\widetilde{\psi},\mathsf{F}_3)\leftarrow\mathsf{Hyb}_3}\left\|\Pi_{\mathsf{token}}\mathcal{A}^{\mathsf{F}_3}|\widetilde{\psi}\rangle\right\|$. Note that $\mathcal{B}$ only makes $\mathrm{poly}(\lambda)$ queries to $\mathsf{Token.Ver}_{\mathsf{vk}}$. By the unforgeability of the tokenized signature scheme, the winning probability of $\mathcal{B}$ will be at most $2^{-\Omega(\lambda)}$, which implies the desired inequality.

□

Lemma 7.6. $\left\{\mathcal{A}^{\mathsf{F}_3}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_3)\leftarrow\mathsf{Hyb}_3(1^\lambda,\psi)\right\}\approx_{2^{-\Omega(\lambda)}}\left\{\mathcal{A}^{\mathsf{F}_4}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_4)\leftarrow\mathsf{Hyb}_4(1^\lambda,\psi)\right\}$

Proof. By Theorem 5.3, making adaptive measurements $r_i\leftarrow\mathcal{M}\left[f_i^{\mathbb{i},r_1,\ldots,r_{i-1}},\theta_i,G_i\right](V)$ is equivalent to making the projective measurement $\{|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V\}_r$. From the definitions of the hybrids, the oracles $\mathsf{F}_3,\mathsf{F}_4$ only differ on inputs that have overlapping with the 1-eigenspace of one of the following projectors for some $i,\mathbb{i},s$.

$$\Pi'_{i,\mathbb{i},s}{}^{L_i,D,V}:=\sum_r\mathsf{EqualH}(i,r_i\oplus 1,\mathbb{i},s)^{L_i,D}\otimes|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V$$

Consider another projector $\Pi_{i,\mathbb{i},s}{}^{D,V}:=\sum_r{\mathsf{Has}_{(i,r_i\oplus1,\mathbb{i},s)}}^D\otimes|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V$.

Claim 7.7. $\left\|\Pi_{i,\mathbb{i},s}{}^{D,V}\mathcal{A}^{\mathsf{F}_4}|\widetilde{\psi}\rangle\right\|\leq 2^{-\Omega(\kappa)}$ *for every* $i,\mathbb{i},s$*, every instantiation of* $(\widetilde{\psi},\mathsf{F}_4)\leftarrow\mathsf{Hyb}_4$*, and every* q*-query adversary* $\mathcal{A}$ *where* $q=\mathrm{poly}(\lambda)$.

The claim will be proved shortly after. Following the claim, we have

$$\begin{aligned}
\left\|\Pi'_{i,\mathbb{i},s}{}^{L_i,D,V}\mathcal{A}^{\mathsf{F}_4}|\widetilde{\psi}\rangle\right\| &\leq \left\|\Pi'_{i,\mathbb{i},s}{}^{L_i,D,V}(I-\Pi_{i,\mathbb{i},s}{}^{D,V})\mathcal{A}^{\mathsf{F}_4}|\widetilde{\psi}\rangle\right\|+\left\|\Pi'_{i,\mathbb{i},s}{}^{L_i,D,V}\Pi_{i,\mathbb{i},s}{}^{D,V}\mathcal{A}^{\mathsf{F}_4}|\widetilde{\psi}\rangle\right\|\\
&\leq \left\|\Pi'_{i,\mathbb{i},s}{}^{L_i,D,V}(I-\Pi_{i,\mathbb{i},s}{}^{D,V})\right\|_{\mathsf{op}}+2^{-\Omega(\kappa)}\\
&= \left\|\sum_r\mathsf{EqualH}(i,r_i\oplus1,\mathbb{i},s)^{L_i,D}(I-\mathsf{Has}_{i,r_i\oplus1,\mathbb{i},s})^D\otimes|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V\right\|_{\mathsf{op}}+2^{-\Omega(\kappa)}\\
&\leq \sum_{a\in\{0,1\}}\left\|\mathsf{EqualH}(i,a,\mathbb{i},s)^{L_i,D}(I-\mathsf{Has}_{i,a,\mathbb{i},s})^D\right\|_{\mathsf{op}}+2^{-\Omega(\kappa)}\leq 2^{-\Omega(\kappa)}
\end{aligned}$$

where the first inequality uses the triangle inequality, the second inequality follows from the claim, the third inequality follows from the triangle inequality and that the norm is multiplicative under tensor, and the last inequality follows from a random oracle analysis (Lemma 3.3). Now a union bound over $(i, \mathtt{i}, s) \in [t] \times \{0,1\}^{2n} \times \{0,1\}^{O(2n\cdot\lambda)}$ and the oracle hybrid argument (Lemma 3.12) shows that the output states of Hyb_3 and Hyb_4 are within trace distance

$$O\left(q \cdot t \cdot 2^{2n} \cdot 2^{O(2n\cdot\lambda)} \cdot 2^{-\Omega(\kappa)}\right) = 2^{-\Omega(\lambda)}$$

□

Proof of Claim 7.7. We will write $\Pi_{i',\mathtt{i}',s'}$ instead of $\Pi_{i,\mathtt{i},s}$. By Theorem 5.3, making adaptive measurements $r_i \leftarrow \mathcal{M}\left[f_i^{\mathtt{i},r_1,\ldots,r_{i-1}}, \theta_i, G_i\right](V)$ is equivalent to making the projective measurement $\{|\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|^V\}_r$. Looking at Hyb_4, we can express $\mathsf{F}_4 \equiv U_P P + \mathsf{Write}(\bot)^Y(I-P)$ with the following projector P and unitary U_P.

$$P := \sum_j |j\rangle\langle j|^J \otimes \sum_{\substack{\widetilde{v}:\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})=\top \\ \mathtt{i},s:\mathsf{Token.Ver}_{\mathsf{vk}}(\mathtt{i},s)=\top}} |\widetilde{v}\rangle\langle\widetilde{v}|^{\widetilde{V}} \otimes |\mathtt{i}\rangle\langle\mathtt{i}|^I \otimes |s\rangle\langle s|^S \otimes \prod_{i<j} \left(\begin{smallmatrix}\mathsf{EqualH}(i,0,\mathtt{i},s)(I-\mathsf{EqualH}(i,1,\mathtt{i},s)) \\ +\mathsf{EqualH}(i,1,\mathtt{i},s)(I-\mathsf{EqualH}(i,0,\mathtt{i},s))\end{smallmatrix}\right)^{L_i,D}$$

$$U_P := \left(\sum_{\mathtt{i},s,r} |\mathtt{i}\rangle\langle\mathtt{i}|^I \otimes |s\rangle\langle s|^S \otimes |\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|^V \otimes \left(\begin{smallmatrix}\sum_{j<t}|j\rangle\langle j|^J \otimes \mathsf{WriteH}(j,r_j,\mathtt{i},s)^{Y,D} \\ +|t\rangle\langle t|^J \otimes \mathsf{Write}(g(\mathtt{i},r))^Y\end{smallmatrix}\right)(I-\mathsf{Has}_{\neq\mathtt{i}})^D\right) + \mathsf{Write}(\bot)^Y \otimes \sum_{\mathtt{i}} |\mathtt{i}\rangle\langle\mathtt{i}|^I \otimes \mathsf{Has}_{\neq\mathtt{i}}{}^D$$

We will show that the operator norm of $\Pi_{i',\mathtt{i}',s'}\mathsf{F}_4(I-\Pi_{i',\mathtt{i}',s'})$ is exponentially small, which means that F_4 does not help much with landing into the 1-eigenspace of $\Pi_{i',\mathtt{i}',s'}$. Intuitively, this is because on input $\mathtt{i}=\mathtt{i}'$, F_4 does not query the random oracle at the positions favored by $\Pi_{i',\mathtt{i}',s'}$, whereas on input $\mathtt{i}\neq\mathtt{i}'$, F_4 will refuse to make changes to register V if the adversary has learned random oracle values on positions $(\cdot,\cdot,\mathtt{i}',\cdot)$. To make the argument precise, we first show that $\Pi_{i',\mathtt{i}',s'}U_P(I-\Pi_{i',\mathtt{i}',s'})=0$. We start with some observations.

- $\Pi_{i',\mathtt{i}',s'}$ only acts on registers $(D_{(i',\cdot,\mathtt{i}',s')}, V)$.
- $\mathsf{Has}_{\neq\mathtt{i}}$ only acts on registers $D \setminus D_{(\cdot,\cdot,\mathtt{i},\cdot)}$.
- For $\mathtt{i}'\neq\mathtt{i}$ and any i,b,s, we have $\mathsf{Has}_{(i,b,\mathtt{i}',s)}\mathsf{Has}_{\neq\mathtt{i}} = \mathsf{Has}_{(i,b,\mathtt{i}',s)} = \mathsf{Has}_{\neq\mathtt{i}}\mathsf{Has}_{(i,b,\mathtt{i}',s)}$.

Hence, $\Pi_{i',\mathtt{i}',s'} = \sum_r \mathsf{Has}_{(i',r_{i'}\oplus 1,\mathtt{i}',s')}{}^D \otimes |\Phi_{\mathtt{i}',r}\rangle\langle\Phi_{\mathtt{i}',r}|^V$ commutes with every $\mathsf{Has}_{\neq\mathtt{i}}$, and thus it commutes also with $\left(\sum_{\mathtt{i}} |\mathtt{i}\rangle\langle\mathtt{i}|^I \otimes \mathsf{Has}_{\neq\mathtt{i}}{}^D\right)$. This shows that

$$\Pi_{i',\mathtt{i}',s'}\left(\mathsf{Write}(\bot)^Y\left(\sum_{\mathtt{i}} |\mathtt{i}\rangle\langle\mathtt{i}|^I \otimes \mathsf{Has}_{\neq\mathtt{i}}{}^D\right)\right)(I-\Pi_{i',\mathtt{i}',s'}) = \mathsf{Write}(\bot)^Y\left(\sum_{\mathtt{i}} |\mathtt{i}\rangle\langle\mathtt{i}|^I \otimes \mathsf{Has}_{\neq\mathtt{i}}{}^D\right)\Pi_{i',\mathtt{i}',s'}(I-\Pi_{i',\mathtt{i}',s'}) = 0$$

For $\mathtt{i}'\neq\mathtt{i}$, we also derive that $\mathsf{Has}_{(i,b,\mathtt{i}',s)}(I-\mathsf{Has}_{\neq\mathtt{i}})=0$, and thus $\Pi_{i',\mathtt{i}',s'}(I-\mathsf{Has}_{\neq\mathtt{i}})=0$.

Furthermore, $\mathsf{WriteH}(j,b,\mathtt{i},s)$ only acts on register $(Y, D_{(j,b,\mathtt{i},s)})$, so we have

$$
\begin{aligned}
&\Pi_{i',\mathtt{i}',s'}\left(\sum_{\mathtt{i},s,r}|\mathtt{i}\rangle\langle\mathtt{i}|^I\otimes|s\rangle\langle s|^S\otimes|\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|^V\otimes\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathtt{i},s)^{Y,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathtt{i},r))^Y\end{pmatrix}(I-\mathsf{Has}_{\neq\mathtt{i}})^D\right)(I-\Pi_{i',\mathtt{i}',s'})\\
=&\Pi_{i',\mathtt{i}',s'}\left(\sum_{\mathtt{i},s,r}|\mathtt{i}\rangle\langle\mathtt{i}|^I\otimes|s\rangle\langle s|^S\otimes|\Phi_{\mathtt{i},r}\rangle\langle\Phi_{\mathtt{i},r}|^V\otimes(I-\mathsf{Has}_{\neq\mathtt{i}})^D\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathtt{i},s)^{Y,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathtt{i},r))^Y\end{pmatrix}\right)(I-\Pi_{i',\mathtt{i}',s'})\\
=&\Pi_{i',\mathtt{i}',s'}\left(\sum_{s,r}|\mathtt{i}',s,\Phi_{\mathtt{i}',r}\rangle\langle\mathtt{i}',s,\Phi_{\mathtt{i}',r}|^{I,S,V}\otimes(I-\mathsf{Has}_{\neq\mathtt{i}'})^D\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathtt{i}',s)^{Y,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathtt{i}',r))^Y\end{pmatrix}\right)(I-\Pi_{i',\mathtt{i}',s'})\\
=&\sum_{s,r}|\mathtt{i}',s,\Phi_{\mathtt{i}',r}\rangle\langle\mathtt{i}',s,\Phi_{\mathtt{i}',r}|^{I,S,V}\otimes(I-\mathsf{Has}_{\neq\mathtt{i}'})^D\left(\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')}{}^D\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathtt{i}',s)^{Y,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathtt{i}',r))^Y\end{pmatrix}(I-\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')})^D\right)\\
=&\sum_{s,r}|\mathtt{i}',s,\Phi_{\mathtt{i}',r}\rangle\langle\mathtt{i}',s,\Phi_{\mathtt{i}',r}|^{I,S,V}\otimes(I-\mathsf{Has}_{\neq\mathtt{i}'})^D\left(\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathtt{i}',s)^{Y,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathtt{i}',r))^Y\end{pmatrix}\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')}{}^D(I-\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')})^D\right)\\
=&0
\end{aligned}
$$

Combining the equalities gives $\Pi_{i',\mathtt{i}',s'}U_P(I-\Pi_{i',\mathtt{i}',s'})=0$. From commutativity, we also have $\Pi_{i',\mathtt{i}',s'}\mathsf{Write}(\perp)^Y(I-\Pi_{i',\mathtt{i}',s'})=0$. Next, we derive an upper bound for $\|\Pi_{i',\mathtt{i}',s'}P(I-\Pi_{i',\mathtt{i}',s'})\|_{\mathsf{op}}$. Since $\mathsf{EqualH}(i,\cdot,\mathtt{i},s)$ only acts on register $(L_i, D_{(i,\cdot,\mathtt{i},s)})$, several terms in P commutes with $\Pi_{i',\mathtt{i}',s'}$ and allows us to simplify $\Pi_{i',\mathtt{i}',s'}P(I-\Pi_{i',\mathtt{i}',s'})$ as

$$
\begin{aligned}
&\Pi_{i',\mathtt{i}',s'}\left(\sum_{j>i'}|j\rangle\langle j|^J\otimes\sum_{\widetilde{v}:\mathsf{Ver}(\widetilde{v})=\top}|\widetilde{v}\rangle\langle\widetilde{v}|^{\widetilde{V}}\otimes|\mathtt{i}'\rangle\langle\mathtt{i}'|^I\otimes|s'\rangle\langle s'|^S\otimes\begin{pmatrix}\mathsf{EqualH}(i',0,\mathtt{i}',s')(I-\mathsf{EqualH}(i',1,\mathtt{i}',s'))\\ +\mathsf{EqualH}(i',1,\mathtt{i}',s')(I-\mathsf{EqualH}(i',0,\mathtt{i}',s'))\end{pmatrix}^{L_{i'},D}\right)(I-\Pi_{i',\mathtt{i}',s'})\\
=&\sum_{j>i'}\sum_{\widetilde{v}:\mathsf{Ver}(\widetilde{v})=\top}\sum_r|j\rangle\langle j|^J\otimes|\widetilde{v}\rangle\langle\widetilde{v}|^{\widetilde{V}}\otimes|\mathtt{i}'\rangle\langle\mathtt{i}'|^I\otimes|s'\rangle\langle s'|^S\otimes|\Phi_{\mathtt{i}',r}\rangle\langle\Phi_{\mathtt{i}',r}|^V\otimes\\
&\qquad\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')}{}^D\begin{pmatrix}\mathsf{EqualH}(i',0,\mathtt{i}',s')(I-\mathsf{EqualH}(i',1,\mathtt{i}',s'))\\ +\mathsf{EqualH}(i',1,\mathtt{i}',s')(I-\mathsf{EqualH}(i',0,\mathtt{i}',s'))\end{pmatrix}^{L_{i'},D}(I-\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')})^D
\end{aligned}
$$

By the fact that the operator norm is multiplicative under tensor and the triangle inequality, the expression above has operator norm at most

$$
\begin{aligned}
&\sum_{r_{i'}\in\{0,1\}}\left\|\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')}{}^D\begin{pmatrix}\mathsf{EqualH}(i',0,\mathtt{i}',s')(I-\mathsf{EqualH}(i',1,\mathtt{i}',s'))\\ +\mathsf{EqualH}(i',1,\mathtt{i}',s')(I-\mathsf{EqualH}(i',0,\mathtt{i}',s'))\end{pmatrix}^{L_{i'},D}(I-\mathsf{Has}_{(i',r_{i'}\oplus1,\mathtt{i}',s')})^D\right\|_{\mathsf{op}}\\
=&\sum_{a\in\{0,1\}}\left\|\mathsf{Has}_{(i',a,\mathtt{i}',s')}{}^D\begin{pmatrix}\mathsf{EqualH}(i',a,\mathtt{i}',s')(I-\mathsf{EqualH}(i',a\oplus1,\mathtt{i}',s'))\\ +\mathsf{EqualH}(i',a\oplus1,\mathtt{i}',s')(I-\mathsf{EqualH}(i',a,\mathtt{i}',s'))\end{pmatrix}^{L_{i'},D}(I-\mathsf{Has}_{(i',a,\mathtt{i}',s')})^D\right\|_{\mathsf{op}}\\
\le&\sum_{a\in\{0,1\}}2\left\|\mathsf{EqualH}(i',a,\mathtt{i}',s')^{L_{i'},D}\left(I-\mathsf{Has}_{(i',a,\mathtt{i}',s')}\right)^D\right\|_{\mathsf{op}}\le2^{-\Omega(\kappa)}
\end{aligned}
$$

where we use the commutativity of $\mathsf{EqualH}(i',a,\mathtt{i}',s'),\mathsf{EqualH}(i',a\oplus1,\mathtt{i}',s')$, the triangle inequality, that norm is sub-multiplicative, followed by a random oracle analysis (Lemma 3.3). Hence, we have $\|\Pi_{i',\mathtt{i}',s'}P(I-\Pi_{i',\mathtt{i}',s'})\|_{\mathsf{op}}\le2^{-\Omega(\kappa)}$. Together with $\Pi_{i',\mathtt{i}',s'}U_P(I-\Pi_{i',\mathtt{i}',s'})=0$ and $\Pi_{i',\mathtt{i}',s'}\mathsf{Write}(\perp)^Y(I-\Pi_{i',\mathtt{i}',s'})=0$, we derive that

$$
\begin{aligned}
\left\|\Pi_{i',\mathtt{i}',s'}\mathsf{F}_4(I-\Pi_{i',\mathtt{i}',s'})\right\|_{\mathsf{op}}&=\left\|\Pi_{i',\mathtt{i}',s'}\left(U_PP+\mathsf{Write}(\perp)^Y(I-P)\right)(I-\Pi_{i',\mathtt{i}',s'})\right\|_{\mathsf{op}}\\
&=\left\|\Pi_{i',\mathtt{i}',s'}\left(U_P\Pi_{i',\mathtt{i}',s'}P+\mathsf{Write}(\perp)^Y\Pi_{i',\mathtt{i}',s'}(I-P)\right)(I-\Pi_{i',\mathtt{i}',s'})\right\|_{\mathsf{op}}\\
&\le\left\|\Pi_{i',\mathtt{i}',s'}P(I-\Pi_{i',\mathtt{i}',s'})\right\|_{\mathsf{op}}+\left\|\Pi_{i',\mathtt{i}',s'}(I-P)(I-\Pi_{i',\mathtt{i}',s'})\right\|_{\mathsf{op}}\le2^{-\Omega(\kappa)}
\end{aligned}
$$

If $\mathcal{A}$ makes $q > 0$ queries, we can write $\mathcal{A}^{\mathsf{F}_4} = U_q \mathsf{F}_4 \mathcal{A}^{\mathsf{F}_4}_{<q}$ where $\mathcal{A}_{<q}$ makes $q-1$ queries. Since U_q commutes with $\Pi_{i,\mathbb{i},s}$, we have

$$\begin{aligned}\left\|\Pi_{i,\mathbb{i},s}\mathcal{A}^{\mathsf{F}_4}|\widetilde{\psi}\rangle\right\| &= \left\|\Pi_{i,\mathbb{i},s}U_q\mathsf{F}_4\mathcal{A}^{\mathsf{F}_4}_{<q}|\widetilde{\psi}\rangle\right\| \\ &\le \left\|U_q\Pi_{i,\mathbb{i},s}\mathsf{F}_4(I-\Pi_{i,\mathbb{i},s})\mathcal{A}^{\mathsf{F}_4}_{<q}|\widetilde{\psi}\rangle\right\| + \left\|U_q\Pi_{i,\mathbb{i},s}\mathsf{F}_4\Pi_{i,\mathbb{i},s}\mathcal{A}^{\mathsf{F}_4}_{<q}|\widetilde{\psi}\rangle\right\| \\ &\le \left\|\Pi_{i,\mathbb{i},s}\mathsf{F}_4(I-\Pi_{i,\mathbb{i},s})\right\|_{\mathsf{op}} + \left\|\Pi_{i,\mathbb{i},s}\mathcal{A}^{\mathsf{F}_4}_{<q}|\widetilde{\psi}\rangle\right\|\end{aligned}$$

Before any queries have been made, the state on D is $|\mathcal{L}_\emptyset\rangle$ so we have

$$\Pi_{i,\mathbb{i},s}{}^{D,V}\mathcal{A}^{\mathsf{F}_4}_{<1}|\widetilde{\psi}\rangle = \Pi_{i,\mathbb{i},s}{}^{D,V}|\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset|^D\mathcal{A}^{\mathsf{F}_4}_{<1}|\widetilde{\psi}\rangle = 0$$

By induction on $q = \mathrm{poly}(\lambda)$, we get

$$\left\|\Pi_{i,\mathbb{i},s}\mathcal{A}^{\mathsf{F}_4}|\widetilde{\psi}\rangle\right\| \le q\left\|\Pi_{i,\mathbb{i},s}\mathsf{F}_4(I-\Pi_{i,\mathbb{i},s})\right\|_{\mathsf{op}} \le 2^{-\Omega(\kappa)}$$

□

Lemma 7.8. $\left\{\mathcal{A}^{\mathsf{F}_4}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_4)\leftarrow\mathsf{Hyb}_4(1^\lambda,\psi)\right\} = \left\{\mathcal{A}^{\mathsf{F}_5}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_5)\leftarrow\mathsf{Hyb}_5(1^\lambda,\psi)\right\}$

Proof. Define the projector

$$P := \sum_j |j\rangle\langle j|^J \otimes \sum_{\substack{\widetilde{v}:\mathsf{Auth.Ver}_{k,\theta_j,G_j}(\widetilde{v})=\top \\ \mathbb{i},s:\mathsf{Token.Ver}_{\mathsf{vk}}(\mathbb{i},s)=\top}} |\widetilde{v}\rangle\langle\widetilde{v}|^{\widetilde{V}} \otimes \mathbb{i}^I \otimes s^S \otimes \prod_{i<j}\begin{pmatrix}\mathsf{EqualH}(i,0,\mathbb{i},s)(I-\mathsf{EqualH}(i,1,\mathbb{i},s)) \\ +\mathsf{EqualH}(i,1,\mathbb{i},s)(I-\mathsf{EqualH}(i,0,\mathbb{i},s))\end{pmatrix}^{L_i,D}$$

By definition of $\mathsf{Hyb}_4, \mathsf{Hyb}_5$, we can express $\mathsf{F}_4, \mathsf{F}_5$ as

$$\begin{aligned}\mathsf{F}_4 = &\left(\sum_{\mathbb{i},s,r}|\mathbb{i}\rangle\langle\mathbb{i}|^I\otimes|s\rangle\langle s|^S\otimes|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V\otimes\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathbb{i},s)^{L_i,D} \\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{pmatrix}(I-\mathsf{Has}_{\neq\mathbb{i}})^D\right)P \\ &+\mathsf{Write}(\bot)^{L_i}\left(\sum_{\mathbb{i}}|\mathbb{i}\rangle\langle\mathbb{i}|^I\otimes\mathsf{Has}_{\neq\mathbb{i}}{}^D\right)P + \mathsf{Write}(\bot)^{L_i}(I-P)\end{aligned}$$

$$\begin{aligned}\mathsf{F}_5 = &\left(\sum_{\mathbb{i},s,r}|\mathbb{i}\rangle\langle\mathbb{i}|^I\otimes|s\rangle\langle s|^S\otimes|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V\otimes\begin{pmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,0,\mathbb{i},s)^{L_i,D} \\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{pmatrix}(I-\mathsf{Has}_{\neq\mathbb{i}})^D\right)P \\ &+\mathsf{Write}(\bot)^{L_i}\left(\sum_{\mathbb{i}}|\mathbb{i}\rangle\langle\mathbb{i}|^I\otimes\mathsf{Has}_{\neq\mathbb{i}}{}^D\right)P + \mathsf{Write}(\bot)^{L_i}(I-P)\end{aligned}$$

Consider the following unitary W that conditionally swaps the database entries of $(i, r_i, \mathbb{i}, s)$ and $(i, 0, \mathbb{i}, s)$. Here, we make the convention that $\mathsf{SWAP}^{D_x,D_{x'}} := I^{D_x}$ when $x = x'$.

$$W^{V,D} := I^V\otimes\left(I-\sum_{\mathbb{i}}\mathsf{Only}_{\mathbb{i}}\right)^D + \sum_{\mathbb{i},r}|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V\otimes\left(\prod_{i,s}\mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\right)\mathsf{Only}_{\mathbb{i}}{}^D$$

We will show that $W\mathsf{F}_4W = \mathsf{F}_5$. First, we have $W^2 = I$ because $\mathsf{SWAP}^2 = I$ and the projectors $\mathsf{Only}_{\mathbb{i}}$ are pairwise orthogonal. Also, since $I - \mathsf{Has}_{\neq \mathbb{i}} = |\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset| + \mathsf{Only}_{\mathbb{i}}$, we have

$$W\left(\sum_{\mathbb{i}} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes \mathsf{Has}_{\neq\mathbb{i}}{}^D\right) W = W\left(\sum_{\mathbb{i}} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes (I - |\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset| - \mathsf{Only}_{\mathbb{i}})^D\right) W$$
$$= \sum_{\mathbb{i}} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes (I - |\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset| - \mathsf{Only}_{\mathbb{i}})^D = \sum_{\mathbb{i}} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes \mathsf{Has}_{\neq\mathbb{i}}{}^D$$

Next, observe that for any $r_i \in \{0,1\}$, the term $\left(\begin{smallmatrix}\mathsf{EqualH}(i,0,\mathbb{i},s)(I-\mathsf{EqualH}(i,1,\mathbb{i},s))\\ +\mathsf{EqualH}(i,1,\mathbb{i},s)(I-\mathsf{EqualH}(i,0,\mathbb{i},s))\end{smallmatrix}\right)^{L_i,D}$ is equal to

$$\mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\left(\begin{smallmatrix}\mathsf{EqualH}(i,0,\mathbb{i},s)(I-\mathsf{EqualH}(i,1,\mathbb{i},s))\\ +\mathsf{EqualH}(i,1,\mathbb{i},s)(I-\mathsf{EqualH}(i,0,\mathbb{i},s))\end{smallmatrix}\right)^{L_i,D}\mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}$$

since it is symmetric in the second argument of $(i,\cdot,\mathbb{i},s)$. This implies that $WPW = P$. Next, observe that $W(I - \mathsf{Has}_{\neq\mathbb{i}})^D = W(|\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset| + \mathsf{Only}_{\mathbb{i}})^D$ is equal to

$$I^V \otimes |\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset|^D + \sum_r |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V \otimes \left(\prod_{i,s} \mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\right)\mathsf{Only}_{\mathbb{i}}{}^D$$
$$= \sum_r |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V \otimes \left(\prod_{i,s} \mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\right)(|\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset| + \mathsf{Only}_{\mathbb{i}})^D$$
$$= \sum_r |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V \otimes \left(\prod_{i,s} \mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\right)(I - \mathsf{Has}_{\neq\mathbb{i}})^D$$

where the first equality follows because $\mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}|\mathcal{L}_\emptyset\rangle = |\mathcal{L}_\emptyset\rangle$. Similarly, the expression is also equal to $(I - \mathsf{Has}_{\neq\mathbb{i}})^D W$. Also, simple calculations show that

$$\left(\begin{smallmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathbb{i},s)^{L_i,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{smallmatrix}\right)(I - \mathsf{Has}_{\neq\mathbb{i}})^D = (I - \mathsf{Has}_{\neq\mathbb{i}})^D\left(\begin{smallmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathbb{i},s)^{L_i,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{smallmatrix}\right)(I - \mathsf{Has}_{\neq\mathbb{i}})^D$$

and that $\left(\sum_{j<t}|j\rangle\langle j|^J \otimes \mathsf{WriteH}(j,0,\mathbb{i},s)^{L_i,D} + |t\rangle\langle t|^J \otimes \mathsf{Write}(g(\mathbb{i},r))^{L_i}\right)$ is equal to

$$\left(\prod_{i,s} \mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\right)\left(\begin{smallmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathbb{i},s)^{L_i,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{smallmatrix}\right)\left(\prod_{i,s} \mathsf{SWAP}^{D_{(i,r_i,\mathbb{i},s)},D_{(i,0,\mathbb{i},s)}}\right)$$

Combining these observations, we derive that

$$W\left(\sum_{\mathbb{i},s,r} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes |s\rangle\langle s|^S \otimes |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V \otimes \left(\begin{smallmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,r_j,\mathbb{i},s)^{L_i,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{smallmatrix}\right)(I - \mathsf{Has}_{\neq\mathbb{i}})^D\right) W$$
$$= \left(\sum_{\mathbb{i},s,r} |\mathbb{i}\rangle\langle\mathbb{i}|^I \otimes |s\rangle\langle s|^S \otimes |\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V \otimes \left(\begin{smallmatrix}\sum_{j<t}|j\rangle\langle j|^J\otimes\mathsf{WriteH}(j,0,\mathbb{i},s)^{L_i,D}\\ +|t\rangle\langle t|^J\otimes\mathsf{Write}(g(\mathbb{i},r))^{L_i}\end{smallmatrix}\right)(I - \mathsf{Has}_{\neq\mathbb{i}})^D\right)$$

From all of the above, we obtain that $W\mathsf{F}_4W = \mathsf{F}_5$. Hence, we have

$$\mathcal{A}^{\mathsf{F}_5}(\widetilde{\psi}) = W^{V,D}\mathcal{A}^{\mathsf{F}_4}W^{V,D}(|\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset|^D\widetilde{\psi}) = W^{V,D}\mathcal{A}^{\mathsf{F}_4}(\widetilde{\psi})$$

where we recall that register D was initialized as $|\mathcal{L}_\emptyset\rangle$ during the preparation of $\widetilde{\psi}$ and we use the relation $W^{V,D}|\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset|^D = |\mathcal{L}_\emptyset\rangle\langle\mathcal{L}_\emptyset|^D$. This proves the lemma because the adversary's output state does not contain registers V, D. □

Lemma 7.9. $\left\{\mathcal{A}^{\mathsf{F}_5}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_5)\leftarrow\mathsf{Hyb}_5(1^\lambda,\psi)\right\}\approx_{2^{-\Omega(\lambda)}}\left\{\mathcal{A}^{\mathsf{F}_6}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_6)\leftarrow\mathsf{Hyb}_6(1^\lambda,\psi)\right\}$

Proof. First, by a similar argument as the proof of Lemma 7.5, removing the condition "If $\mathsf{Has}_{\neq\mathbb{i}}$ holds" from F_5 would only affect the output state by at most $2^{-\Omega(\lambda)}$ in trace distance. The modified oracle and F_6 only differ on inputs that have overlapping with the 1-eigenspace of $\mathsf{EqualH}(i,1,\mathbb{i},s)^{L_i,D}$ for some $i,\mathbb{i},s$. Similar to the proof of Lemma 7.6, by a union bound and the oracle hybrid lemma (Lemma 3.12), it suffices to show that

$$\left\|\mathsf{EqualH}(i,1,\mathbb{i},s)^{L_i,D}\mathcal{A}^{\mathsf{F}_6}|\widetilde{\psi}\rangle\right\|\leq 2^{-\Omega(\kappa)}$$

for every $i,\mathbb{i},s$, every instantiation of $(\widetilde{\psi},\mathsf{F}_6)\leftarrow\mathsf{Hyb}_6$, and polynomial-query adversary $\mathcal{A}$. By the definition of Hyb_6, we have $\mathsf{Has}_{(i,1,\mathbb{i},s)}{}^D\mathcal{A}^{\mathsf{F}_6}(\widetilde{\psi})=0$. Hence, we have

$$\left\|\mathsf{EqualH}(i,1,\mathbb{i},s)^{L_i,D}\mathcal{A}^{\mathsf{F}_6}|\widetilde{\psi}\rangle\right\|=\left\|\mathsf{EqualH}(i,1,\mathbb{i},s)^{L_i,D}(I-\mathsf{Has}_{(i,1,\mathbb{i},s)})^D\mathcal{A}^{\mathsf{F}_6}|\widetilde{\psi}\rangle\right\|\leq 2^{-\Omega(\kappa)}$$

as desired by a random oracle analysis (Lemma 3.3). □

Lemma 7.10. $\left\{\mathcal{A}^{\mathsf{F}_6}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_6)\leftarrow\mathsf{Hyb}_6(1^\lambda,\psi)\right\}\approx_{\mathrm{negl}(\lambda)}\left\{\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})\middle|(\widetilde{\psi},\mathsf{F}_7)\leftarrow\mathsf{Hyb}_7(1^\lambda,\psi)\right\}$

Proof. Since $\mathcal{A}$ only makes $\mathrm{poly}(\lambda)$ queries, by a standard hybrid argument, it suffices to prove that replacing the last query to F_7 with F_6 one at a time only affects the output state by $\mathrm{negl}(\lambda)$ trace distance. Therefore, we just have to prove that for every instantiation of $(\widetilde{\psi},\mathsf{F}_7)\leftarrow\mathsf{Hyb}_7$ together with F_6 defined using the same key as F_7, and every adversary $\mathcal{A}$,

$$\mathsf{F}_7\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})\approx_{\mathrm{negl}(\lambda)}\mathsf{F}_6\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})$$

To compare the difference between F_6 and F_7, observe that when the query input $j<t$, the internal results r_i computed in line 4 of F_6 are never used, so removing this step does not affect oracle behavior in this case. When the query input $j=t$ and controlled on all verifications pass, F_6 computes $g(\mathbb{i},r)$ according to the PLM quantum program compiled from $\widetilde{Q}^{\mathsf{univ}}$ and applies $\mathsf{Write}(g(\mathbb{i},r))^Y$. It is equivalent to applying

$$\sum_y(\mathsf{X}^y)^Y\otimes\sum_{r:g(\mathbb{i},r)=y}|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|^V$$

Note that in $\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})$, registers $V_{\mathsf{aux}},V_{\mathsf{PLM}}$ still hold the states ψ,ψ_{PLM} because F_7 does not use these registers at all. By Theorem 5.3,

$$\sum_y\mathsf{X}^y\otimes\sum_{r:g(\mathbb{i},r)=y}|\Phi_{\mathbb{i},r}\rangle\langle\Phi_{\mathbb{i},r}|\left(I\otimes|\psi_{\mathsf{PLM}}\rangle\right)=\sum_y\mathsf{X}^y\otimes\left(U^\dagger_{\widetilde{Q}^{\mathsf{univ}},\mathbb{i}}\left(|y\rangle\langle y|\otimes I\right)U_{\widetilde{Q}^{\mathsf{univ}},\mathbb{i}}\right)\otimes|\psi_{\mathsf{PLM}}\rangle$$

where ${U_{\widetilde{Q}^{\mathsf{univ}},\mathbb{i}}}^{V} = \mathsf{TP}^{V_{\mathrm{in}},V_{\mathrm{out}}} {Q^{\mathsf{univ}}}^{V_{\mathrm{in}},V_{\mathsf{aux}}} {P_{\mathbb{i}}^{\dagger}}^{V_{\mathrm{in}}}$. Hence, it is equivalent if F_6 instead applies

$$\begin{aligned}
&{U^{\dagger}_{\widetilde{Q}^{\mathsf{univ}},\mathbb{i}}}^{V} \mathsf{Write}(V_{\mathrm{in}}, V_{\mathrm{out}})^{Y} {U_{\widetilde{Q}^{\mathsf{univ}},\mathbb{i}}}^{V} \\
&= \left({P_{\mathbb{i}}}^{V_{\mathrm{in}}} {Q^{\mathsf{univ}}}^{\dagger V_{\mathrm{in}},V_{\mathsf{aux}}} \mathsf{TP}^{\dagger V_{\mathrm{in}},V_{\mathrm{out}}}\right) \mathsf{Write}(V_{\mathrm{in}}, V_{\mathrm{out}})^{Y} \left(\mathsf{TP}^{V_{\mathrm{in}},V_{\mathrm{out}}} {Q^{\mathsf{univ}}}^{V_{\mathrm{in}},V_{\mathsf{aux}}} {P_{\mathbb{i}}^{\dagger}}^{V_{\mathrm{in}}}\right)
\end{aligned}$$

In comparison, when the query input $j = t$ and controlled on all verifications pass, F_7 applies

$$\left({P_{\mathbb{i}}}^{V_{\mathrm{in}}} {U^{\dagger}}^{V_{\mathrm{in}}} \mathsf{TP}^{\dagger V_{\mathrm{in}},V_{\mathrm{out}}}\right) \mathsf{Write}(V_{\mathrm{in}}, V_{\mathrm{out}})^{Y} \left(\mathsf{TP}^{V_{\mathrm{in}},V_{\mathrm{out}}} {U^{V_{\mathrm{in}}}} {P_{\mathbb{i}}^{\dagger}}^{V_{\mathrm{in}}}\right)$$

As noted above, the state $\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})$ still holds ψ on register V_{aux}. By assumption, $Q^{\mathsf{univ}}(\cdot,\psi)$ is an $\mathrm{negl}(\lambda)$-approximation of U, so it follows by Lemma 6.4 that $\mathsf{F}_6\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})$ is negligibly close to $\mathsf{F}_7\mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})$ in trace distance. □

Lemma 7.11. $$\left| \Pr_{(\widetilde{\psi},\mathsf{F}_7)\leftarrow\mathsf{Hyb}_7(1^\lambda,\psi)} \left[1 \leftarrow \mathcal{A}^{\mathsf{F}_7}(\widetilde{\psi})\right] - \Pr_{(\widetilde{\psi},\mathsf{F}_{\mathsf{Sim}})\leftarrow\mathsf{Sim}(1^\lambda,1^n,1^m)} \left[1 \leftarrow \mathcal{A}^{\mathsf{F}_{\mathsf{Sim}}^{\mathsf{ctrl}\text{-}(U^\dagger\cdot U)}}(\widetilde{\psi})\right] \right| = \mathrm{negl}(\lambda).$$

Proof. Observe that the registers $V_{\mathsf{aux}}, V_{\mathsf{PLM}}$ storing $\psi, \psi_{\mathsf{PLM}}$ are never used in Hyb_7, so they can be completely removed. The remaining difference is that the purified random oracle in Hyb_7 is changed to a post-quantum PRF in Sim, and they are computationally indistinguishable under polynomially-many quantum queries. □

Proof of Theorem 7.2. It is clear by definition that Sim runs in polynomial time. From the above lemmas which prove indistinguishability between consecutive hybrids, it follows that for every polynomial-time adversary $\mathcal{A}$,

$$\left| \Pr_{(\widetilde{\psi},\mathsf{F})\leftarrow\mathsf{QObf}(1^\lambda,\psi)} \left[1 \leftarrow \mathcal{A}^{\mathsf{F}}(\widetilde{\psi})\right] - \Pr_{(\widetilde{\psi},\mathsf{F}_{\mathsf{Sim}})\leftarrow\mathsf{Sim}(1^\lambda,1^n,1^m)} \left[1 \leftarrow \mathcal{A}^{\mathsf{F}_{\mathsf{Sim}}^{\mathsf{ctrl}\text{-}(U^\dagger\cdot U)}}(\widetilde{\psi})\right] \right| = \mathrm{negl}(\lambda).$$

This proves the ideal obfuscation property. To prove the functionality-preserving property, we apply Lemmas 6.2 and 7.3 with $\mathcal{A} = \mathsf{QEval}$, so it suffices to prove that for every state ρ_{in},

$$\mathop{\mathbb{E}}_{(\widetilde{\psi},\mathsf{F}_1)\leftarrow\mathsf{Hyb}_1(1^\lambda,(\psi,Q^{\mathsf{univ}}))} \left[\mathsf{QEval}^{\mathsf{F}_1}\left(\rho_{\mathrm{in}}, \widetilde{\psi}\right)\right] \mathop{\approx}_{\mathrm{negl}(\lambda)} Q^{\mathsf{univ}}(\rho_{\mathrm{in}}, \psi).$$

First, QEval performs teleportation $\mathbb{i} \leftarrow \mathsf{TP.Send}(\rho_{\mathrm{in}}, \mathsf{epr}_{(\mathrm{in},\mathsf{pub})})$, and the state $\mathsf{epr}_{(\mathrm{in},\mathsf{priv})}$ on register V_{in} collapses to $P_{\mathbb{i}}\rho_{\mathrm{in}}P_{\mathbb{i}}^{\dagger}$. Then QEval obtains a signature $s \leftarrow \mathsf{Token.Sign}(\mathbb{i}, \tau_{\mathsf{token}})$. When QEval calls F_1, the first two verification steps of F_1 always succeed by the correctness of the authentication code and the tokenized signature. Furthermore, for every $\mathbb{i}$ and s, the third verification step of F_1 encounters a collision only with negligible probability over the choice of PRF key. When no collision occurs, the value ℓ_t is computed as

$$\ell_t \leftarrow g(\mathbb{i}, r_1, \ldots, r_t) \text{ where } V \leftarrow \left(P_{\mathbb{i}}\rho_{\mathrm{in}}P_{\mathbb{i}}^{\dagger}, \mathsf{epr}_{(\mathrm{out},\mathsf{priv})}, \psi, \psi_{\mathsf{PLM}}\right) \text{ and } r_j \leftarrow \mathcal{M}\left[f_j^{\mathbb{i},r_j,\ldots,r_{j-1}}, \theta_j, G_j\right](V)$$

In this case, by Theorem 5.3, the value ℓ_t can also be computed as

$$\ell_t \leftarrow \widetilde{Q}^{\mathsf{univ}}\left(P_{\mathbb{i}}\rho_{\mathrm{in}}P_{\mathbb{i}}^{\dagger}, \mathsf{epr}_{(\mathrm{out},\mathsf{priv})}, \psi, \psi_{\mathsf{PLM}} \;;\; \mathbb{i}\right) \equiv \mathsf{TP.Send}\left(Q^{\mathsf{univ}}\left(\rho_{\mathrm{in}}, \psi\right), \mathsf{epr}_{(\mathrm{out},\mathsf{priv})}\right)$$

Hence, the output $\mathsf{TP.Recv}(\ell_t, \mathsf{epr}_{(\mathrm{out},\mathsf{pub})})$ of $\mathsf{QEval}^{\mathsf{F}_1}$ is negligibly close in trace distance to

$$\mathsf{TP.Recv}\left(\mathsf{TP.Send}\left(Q^{\mathsf{univ}}\left(\rho_{\mathrm{in}}, \psi\right), \mathsf{epr}_{(\mathrm{out},\mathsf{priv})}\right), \mathsf{epr}_{(\mathrm{out},\mathsf{pub})}\right) = Q^{\mathsf{univ}}\left(\rho_{\mathrm{in}}, \psi\right)$$

by the correctness of quantum teleportation since $(\mathsf{epr}_{(\mathrm{out},\mathsf{priv})}, \mathsf{epr}_{(\mathrm{out},\mathsf{pub})})$ is an EPR . □

Acknowledgement

The authors deeply appreciate Andrea Coladangelo for his selfless devotion in his line-by-line feedback in polishing the paper and discussing it with them. The authors are grateful to Rachel Lin for deepening their understanding of classical obfuscation. The authors also thank Jiapeng Zhang, Shanghua Teng, Kai-Min Chung, Eli Goldin, Aparna Gupte, and anonymous FOCS reviewers for useful discussions.

Miryam Huang is supported by NSF CAREER award 2141536. Part of the work was done when Miryam Huang was visiting to UW Quantum group and Cryptography group. Er-Cheng Tang is supported by NSF NRT-QL award 2021540. This work was initiated during Miryam Huang and Er-Cheng Tang's visit to the Simons Institute for the Theory of Computing.

References

[ABDS21] Gorjan Alagic, Zvika Brakerski, Yfke Dulek, and Christian Schaffner. Impossibility of quantum virtual black-box obfuscation of classical circuits. In *Annual International Cryptology Conference*, pages 497–525. Springer, 2021.

[AJL+19] Prabhanjan Ananth, Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. Indistinguishability obfuscation without multilinear maps: new paradigms via low degree weak pseudorandomness and security amplification. In *Annual International Cryptology Conference*, pages 284–332. Springer, 2019.

[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.

[BDS23] Shalev Ben-David and Or Sattath. Quantum tokens for digital signatures. *Quantum*, 7:901, 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.

[BGS13] Anne Broadbent, Gus Gutoski, and Douglas Stebila. Quantum one-time programs. In *Annual Cryptology Conference*, pages 344–360. Springer, 2013.

[BK21] Anne Broadbent and Raza Ali Kazmi. Constructions for quantum indistinguishability obfuscation. In *International Conference on Cryptology and Information Security in Latin America*, pages 24–43. Springer, 2021.

[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 *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, pages 15:1–15:13, 2022.

[CG24] Andrea Coladangelo and Sam Gunn. How to use quantum indistinguishability obfuscation. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1003–1008, 2024.

[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.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 40–49, 2013.

[JJ22] Abhishek Jain and Zhengzhong Jin. Indistinguishability obfuscation via mathematical proofs of equivalence. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1023–1034. IEEE, 2022.

[JLLW23] Aayush Jain, Huijia Lin, Ji Luo, and Daniel Wichs. The pseudorandom oracle model and ideal obfuscation. In *Annual International Cryptology Conference*, pages 233–262. Springer, 2023.

[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.

[KSW08] Dennis Kretschmann, Dirk Schlingemann, and Reinhard F Werner. The information-disturbance tradeoff and the continuity of stinespring's representation. *IEEE transactions on information theory*, 54(4):1708–1717, 2008.

[MH25] Fermi Ma and Hsin-Yuan Huang. How to construct random unitaries. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 806–809, 2025.

[SHH25] Thomas Schuster, Jonas Haferkamp, and Hsin-Yuan Huang. Random unitaries in extremely low depth. *Science*, 389(6755):92–96, 2025.

[Wat18] John Watrous. *The theory of quantum information*. Cambridge university press, 2018.

[Win99] Andreas Winter. Coding theorem and strong converse for quantum channels. *IEEE Transactions on Information Theory*, 45(7):2481–2485, 1999.

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

[Zha21a] Mark Zhandry. How to construct quantum random functions. *Journal of the ACM (JACM)*, 68(5):1–43, 2021.

[Zha21b] Mark Zhandry. Quantum lightning never strikes the same state twice. or: quantum money from cryptographic assumptions. *Journal of Cryptology*, 34:1–56, 2021.

[Zha25] Mark Zhandry. How to model unitary oracles. In *Annual International Cryptology Conference*, pages 237–268. Springer, 2025.