On The Round Complexity of Secure Quantum Computation

James Bartusek*, Andrea Coladangelo**, Dakshita Khurana***, and Fermi Ma†

Abstract. We construct the first *constant-round* protocols for secure quantum computation in the two-party (2PQC) and multi-party (MPQC) settings with security against *malicious* adversaries. Our protocols are in the common random string (CRS) model.

- Assuming two-message oblivious transfer (OT), we obtain (*i*) three-message 2PQC, and (*ii*) five-round MPQC with only three rounds of *online* (input-dependent) communication; such OT is known from quantum-hard Learning with Errors (QLWE).
- Assuming sub-exponential hardness of QLWE, we obtain (*i*) three-round 2PQC with two online rounds and (*ii*) four-round MPQC with two online rounds.
- When only one (out of two) parties receives output, we achieve *minimal interaction* (two messages) from two-message OT; classically, such protocols are known as non-interactive secure computation (NISC), and our result constitutes the first maliciously-secure quantum NISC.

  Additionally assuming reusable malicious designated-verifier NIZK arguments for NP (MDV-NIZKs), we give the first MDV-NIZK for QMA that only requires one copy of the quantum witness.

Finally, we perform a preliminary investigation into *two-round* secure quantum computation where each party must obtain output. On the negative side, we identify a broad class of simulation strategies that suffice for *classical* two-round secure computation that are *unlikely* to work in the quantum setting. Next, as a proof-of-concept, we show that two-round secure quantum computation exists with respect to a quantum oracle.

1 Introduction

Secure computation allows mutually distrusting parties to compute arbitrary functions on their private inputs, revealing only the outputs of the computation while hiding all other private information [40,27,12,18]. With the emergence of quantum computers, it becomes important to understand the landscape of secure *quantum* computation over distributed, private quantum (or classical) states. In

* UC Berkeley. Email: bartusek.james@gmail.com
** UC Berkeley. Email: andrea.coladangelo@gmail.com
*** UIUC. Email: dakshita@illinois.edu
† Princeton University and NTT Research. Email: fermima@alum.mit.edu

the most general setting, n parties hold (possibly entangled) quantum inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n$, and would like to evaluate a quantum circuit $Q(\mathbf{x}_1, \ldots, \mathbf{x}_n)$. The output is of the form $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$, so at the end of the protocol party i holds state $\mathbf{y}_i$.

Secure computation with classical inputs and circuits forms a centerpiece of classical cryptography. Solutions to this problem in the classical setting were first obtained nearly 35 years ago, when [40] built garbled circuits to enable secure two-party computation, and [27,12,18] obtained the first secure multi-party computation protocols. Since then, there has been an extensive body of work in this area, of which a large chunk focuses on understanding the amount of back-and-forth interaction required to implement these protocols. Notably, the work of Beaver, Micali and Rogaway [9] obtained the first constant-round classical multi-party computation protocols in the dishonest majority setting. There have been several subsequent works including but not limited to [31,25,5,14,7,19] that have nearly completely characterized the *exact* round complexity of classical secure computation.

The problem of secure *quantum* computation on distributed quantum states is not nearly as well-understood as its classical counterpart. The quantum setting was first studied by [20,11], who obtained unconditional maliciously-secure multi-party quantum computation with honest majority. Just like the classical setting, when half (or more) of the players are malicious, secure quantum computation also requires computational assumptions due to the impossibility of unconditionally secure quantum bit commitment [35,32,21].

In the dishonest majority setting, [23] gave a two-party quantum computation (2PQC) protocol secure against the quantum analogue of semi-honest adversaries (specious adversaries); this was later extended to the malicious setting by [24]. A work of [22] constructed maliciously-secure *multi-party* quantum computation (MPQC) with dishonest majority from any maliciously-secure post-quantum classical MPC, where the round complexity grows with the size of the circuit *and* the number of participants. Very recently, [4] constructed MPQC with identifiable abort, and with round complexity that does not grow with the circuit size but grows with the number of participants.

However, the feasibility of maliciously-secure MPQC with *constant* rounds has remained open until this work. In addition to settling this question, we also make several headways in understanding the *exact* round complexity of secure quantum computation with up to all-but-one malicious corruptions.

1.1 Our Results

We assume that parties have access to a common random string (CRS), and obtain answers to a range of fundamental questions, as we discuss next[1].

[1] We point out that the post-quantum MPC protocol of [1] can be used to generate a CRS in constant rounds. This, combined with our results, yields the first constant round multi-party quantum computation protocols without trusted setup in the standard model.

Quantum Non-Interactive Secure Computation Our first result pertains to the most basic setting for secure (quantum) computation: a sender holds input $\mathbf{y}$, a receiver holds input $\mathbf{x}$, and the goal is for the receiver to obtain $Q(\mathbf{x}, \mathbf{y})$ for some quantum circuit Q. We construct a protocol achieving *minimal interaction* — commonly known as non-interactive secure computation (NISC) [30] — where the receiver publishes an encryption of $\mathbf{x}$, the sender replies with an encryption of $\mathbf{y}$, and the receiver subsequently obtains $Q(\mathbf{x}, \mathbf{y})$. Our result constitutes the first maliciously-secure NISC for quantum computation (Q-NISC).

Theorem 1. *(Informal) Maliciously-secure NISC for quantum computation exists assuming post-quantum maliciously-secure two-message oblivious transfer (OT) with straight-line simulation.*

Such OT protocols are known from the post-quantum hardness of Learning with Errors (LWE) [36]. We remark that our Q-NISC result also extends to the *reusable* setting where the receiver has a classical input that they would like to reuse across multiple quantum computations on different sender inputs.

Application: Malicious Designated-Verifier NIZK Arguments for QMA*.* As an application of our maliciously-secure Q-NISC, we construct (reusable) *malicious designated-verifier non-interactive zero-knowledge arguments* (MDV-NIZKs) for QMA in the common random string model. Specifically, our MDV-NIZK enables the following interaction for any QMA language: a verifier can publish a classical public key pk that enables a prover to send an instance x and quantum message $\mathbf{m}$, such that the verifier holding the corresponding secret key sk can determine if x is a valid instance.

Theorem 2. *(Informal) There exists a reusable MDV-NIZK for* QMA *with a classical CRS and classical proving key assuming the existence of post-quantum maliciously-secure two-message oblivious transfer with straight-line simulation in the CRS model, and post-quantum (adaptively sound) reusable MDV-NIZK for NP. All of the underlying primitives exist assuming the quantum hardness of learning with errors.*

We briefly elaborate on the security guarantees of our reusable MDV-NIZK. Reusability means that soundness holds for multiple proofs (of potentially different statements) computed with respect to the same setup (i.e., the common random string and the public key), even if the prover learns whether or not the verifier accepted each proof; we remark that reusable security is sometimes referred to as multi-theorem security. Malicious security means that the zero-knowledge property holds even against verifiers that generate the public key maliciously. Previously, such a reusable MDV-NIZK for QMA required the prover to have access to multiple copies of the quantum witness [37], while our MDV-NIZK only requires the prover to have a single copy.

Constant-round 2PQC and MPQC Our next set of results concerns the general setting for 2PQC and MPQC *where all parties obtain output.* We focus

on minimizing total round complexity as well as *online* round complexity, where the latter refers to the number of *input-dependent* rounds; if a protocol has round complexity d and online round complexity k, then the parties can perform the first $d - k$ rounds *before* they receive their inputs.[2]

We obtain various results, some from the generic assumption of quantum polynomially-secure two-message oblivious transfer, and others from the specific assumption of sub-exponential QLWE. Our results in this section are summarized in Table 1.[3]

Table 1. Maliciously-Secure Quantum Computation in the CRS Model

	From OT	From sub-exp QLWE
Two-party	3 rounds (3 online)	3 rounds (2 online)
Multi-party	5 rounds (3 online)	4 rounds (2 online)

In order to prove the security of these protocols, we develop a delayed simulation technique, which we call "simulation via teleportation", which may be of independent interest.

Is Two-Round Secure Quantum Computation Possible? A natural next question is whether it is possible to construct two-round secure quantum computation *without* pre-processing. This appears to be a challenging question to resolve, either positively or negatively. We provide some preliminary results on both fronts: we give a negative result indicating that common simulation strategies from the classical setting will not suffice in the quantum setting, but we also provide a proof-of-concept positive result, with a new simulation strategy, assuming virtual-black-box obfuscation of quantum circuits. We stress that the latter result is primarily to give intuition, as virtual-black-box obfuscation is known to be impossible even for classical circuits [8]. We limit the scope of this preliminary investigation to the *two-party* setting.

First, we give some intuition for why it seems hard to design a two-round two-party protocol by showing that, under a plausible quantum information-theoretic conjecture, a large class of common simulation techniques would *not* suffice. We consider any simulator that learns which player (between Alice and Bob) is corrupted only *after* it has generated the simulated CRS. We call such a simulator an *oblivious simulator*. To the best of our knowledge, all existing

[2] We remark that a k-online round protocol can also be viewed as a k-round protocol in a quantum pre-processing model, i.e. a model where parties receive some quantum correlations as setup.

[3] The results below are in the setting of security with abort, as opposed to security with *unanimous* abort (which is only a distinction in the multi-party setting). If one wants security with unanimous abort, the overall round complexity will not change, but one more round of online communication will be required.

classical and quantum two-party computation protocols in the CRS model either (1) already admit oblivious simulation, or (2) can generically be transformed to admit oblivious simulation via post-quantum NIZK proofs of knowledge for NP.

In the quantum setting, we show, roughly, that any two-round 2PQC protocol for general quantum functionalities *with an oblivious simulator* would yield an *instantaneous nonlocal quantum computation* protocol [39,10,38,28] for general quantum functionalities, with polynomial-size pre-processing.

Instantaneous nonlocal quantum computation is well-studied in the quantum information literature [39,10,38,28], and the best known protocols for general functionalities require exponential-size pre-processing [10]. Thus, a two-round 2PQC for general functionalities with oblivious simulation would immediately yield an exponential improvement in the size of the pre-processing for this task.

Theorem 3. *(Informal) Under the conjecture that there exists a quantum functionality that does not admit an instantaneous nonlocal quantum computation protocol with polynomial-size pre-processing, there exists a quantum functionality that cannot be securely computed in two rounds in the classical CRS model with an oblivious simulator.*

Towards getting around this potential barrier, we give a proof-of-concept construction of a protocol with non-oblivious simulation. Specifically, we assume a (strong) form of VBB obfuscation for quantum circuits that contain unitary and measurement gates, where the former may be classically controlled on the outcome of measurement gates. We point out, however, that VBB-obfuscation of circuits with measurement gates is potentially even more powerful than the VBB obfuscation for unitaries that was formalized in [3] (further discussion on this is available in the full version). Under this assumption, we obtain a two-round two-party secure quantum computation protocol in the CRS model.

Theorem 4. *(Informal) Two-round two-party secure quantum computation in the common reference string model exists assuming a strong form of VBB or ideal obfuscation for quantum circuits as discussed above.*

We remark that while there exist (contrived) examples of functionalities that cannot be VBB obfuscated [3,2,6], it is still plausible that many quantum functionalities can be obfuscated. However, without any candidate constructions of obfuscation for quantum circuits, we stress that our result should only be taken as a proof-of-concept.

1.2 Paper Organization

In Section 2, we provide technical intuition for all of our results. In Section 3, we give a full technical specification of our three-message 2PQC protocol. We prove that security holds against a malicious Alice, and we defer a security proof for malicious Bob to the full version (as will become clear in Section 2, handling malicious Alice is the more challenging case). We defer the remainder of our results to the full version, which includes the two-round 2PQC with preprocessing, the

MPQC results, the technical formalization of [15] C+M garbling, MDV-NIZKs for QMA, our oblivious simulation barrier, and our VBB-based proof-of-concept construction.

2 Technical Overview

2.1 Quantum Background

We briefly recap some relevant concepts from quantum computation.

Notation. We use bold letters to write the density matrix of a quantum state $\mathbf{x}$. We use the shorthand $U(\mathbf{x})$ to mean $U\mathbf{x}U^\dagger$, the result of applying unitary U to $\mathbf{x}$. The notation $(\mathbf{x}, \mathbf{y})$ denotes a state on two registers, where $\mathbf{x}$ and $\mathbf{y}$ are potentially entangled. The k-fold tensor product of a state $\mathbf{x} \otimes \mathbf{x} \otimes \cdots \otimes \mathbf{x}$ will be written as $\mathbf{x}^k$.

The Pauli Group. The Pauli group on a single qubit, denoted by $\mathscr{P}_1$, is generated by the unitary operations X (bit flip) and Z (phase flip), defined as $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. The Pauli group on n qubits, denoted by $\mathscr{P}_n$, is the n-fold tensor product of the single qubit Pauli group. Any unitary in the Pauli group $\mathscr{P}_n$ can be written (up to global phase) as $\bigotimes_{i\in[n]} X^{r_i} Z^{s_i}$ for $r, s \in \{0,1\}^n$.

The Clifford Group. The Clifford group on n qubits, denoted by $\mathscr{C}_n$, is the group of unitaries that normalize $\mathscr{P}_n$, i.e. $C \in \mathscr{C}_n$ if and only if for all $U \in \mathscr{P}_n$, we have $CUC^\dagger \in \mathscr{P}_n$. Alternatively, we can think of a Clifford unitary C as an operation where for any choice of $r, s \in \{0,1\}^n$, there exists a choice of $r', s' \in \{0,1\}^n$ such that

$$C\left(\bigotimes_{i\in[n]} X^{r_i} Z^{s_i}\right) = \left(\bigotimes_{i\in[n]} X^{r'_i} Z^{s'_i}\right) C.$$

Intuitively, this means that with a suitable update of the Pauli operation, one can swap the order in which a Clifford and a Pauli are applied.

Clifford Authentication Codes. We will make extensive use of Clifford authentication codes. Clifford authentication codes are an information-theoretic encoding scheme for quantum states that provides both secrecy and authentication. An n-qubit quantum state $\mathbf{x}$ can be encoded in a Clifford authentication code as follows: prepare a λ-qubit all 0's state which we denote as $\mathbf{0}^\lambda$ (where λ is a security parameter), sample a random Clifford unitary $C \leftarrow \mathscr{C}_{n+\lambda}$, and output $C(\mathbf{x}, \mathbf{0}^\lambda)$. The Clifford C serves as a secret key, while the $\mathbf{0}^\lambda$ qubits enable authentication, and are called "trap" qubits. A party without knowledge of C cannot modify the encoding without modifying the trap qubits (except with negligible probability). Therefore, decoding works by applying $C^\dagger$ and then measuring the λ trap qubits in the computational basis. If these measurements are all 0, this ensures that with all but negligible probability, the n remaining registers hold the originally encoded state $\mathbf{x}$.

Clifford + Measurement Circuits. We will rely heavily on the "Clifford + Measurement" representation of quantum circuits (henceforth "C+M circuits") due to [16]. In this representation, a quantum circuit can be decomposed into layers. Each layer consists of a Clifford unitary whose output wires are partitioned into wires that will be fed as inputs into the next layer, and wires that will be measured. The latter group of wires are measured in the computational basis, resulting in a classical bitstring that is used to select the Clifford unitary to be applied in the subsequent layer. The first layer takes in all of the inputs to the quantum circuit, ancilla $\mathbf{0}$ states, and "magic" $\mathbf{T}$ states defined as $\mathbf{T} := (|0\rangle + e^{i\pi/4}|1\rangle)/\sqrt{2}$. The final layer only produces output wires (i.e. its output registers have no wires to be measured), which are interpreted as the output of the circuit. [16] demonstrate that, with constant multiplicative factor overhead in size, any quantum circuit can be written as a "C + M circuit" or equivalently, in a magic state representation.

Therefore, for the purposes of this technical overview, we will assume that any quantum circuit F is written as a C+M circuit F_{CM}, and its evaluation on an input $\mathbf{x}$ is computed as $F(\mathbf{x}) = F_{\mathrm{CM}}(\mathbf{x}, \mathbf{T}^k, \mathbf{0}^k)$. For simplicity, we use the same k to denote the number of $\mathbf{T}$ states and the number of ancilla $\mathbf{0}$ states.

Magic State Distillation. In settings where malicious parties are tasked with providing the $\mathbf{T}$ states, we will use cryptographic techniques such as "cut-and-choose" to ensure that F_{CM} is evaluated on an input of the form $(\mathbf{x}, \widehat{\mathbf{T}^k}, \mathbf{0}^k)$ where $\widehat{\mathbf{T}^k}$ is a state guaranteed to be "somewhat" close to $\mathbf{T}^k$. However, correctness of F_{CM} will require states that are negligibly close to real magic states. To that end, we will make use of a magic state distillation C+M circuit D due to [22] which takes in somewhat-close magic states $\widehat{\mathbf{T}^k}$ and outputs states negligibly close to $\mathbf{T}^{k'}$, for $k' < k$. Therefore, the representation of any functionality F will in fact be a C+M circuit $F_{\mathrm{CM},D}$ that first applies D to $\widehat{\mathbf{T}^k}$, and then runs F_{CM}.

2.2 Why is Malicious Security Hard to Achieve?

We begin this technical overview by describing our results in the two-party setting. Before this, we briefly explain why malicious security does not follow readily from existing techniques. Indeed, a candidate two-message 2PQC (where one party receives output) with *specious* security (the quantum analogue of classical semi-honest security [23]) was recently proposed in [15]. Alternatively, any construction of quantum fully-homomorphic encryption (QFHE) naturally yields a two-message 2PQC protocol: (1) Alice QFHE-encodes her input and sends it to Bob, (2) Bob evaluates the functionality on his input and Alice's encoded input, and (3) Bob sends Alice the encryption of her output.

One might hope to compile this QFHE-based protocol or the [15] protocol into a maliciously secure protocol by having the parties include proofs that their messages are well-formed. Unfortunately, it is unclear how to implement this in the quantum setting. In both of these approaches, the parties would have to prove (in zero-knowledge) statements of the form "$\mathbf{y}$ is the result of evaluating

quantum circuit C on $\mathbf{x}$." Crucially, the *statement* the parties need to prove explicitly makes reference to a quantum state. This is beyond the reach of what one can prove with, say, NIZKs for QMA, in which witnesses are quantum but the statements are entirely classical.

Therefore, we design our malicious 2PQC so that parties do not have to prove general statements about quantum states. A core ingredient in our protocol is a quantum garbled circuit construction sketched in [15, §2.5], where the circuit garbling procedure is entirely classical.[4] Combining this with a post-quantum maliciously-secure *classical* 2PC, we will ensure valid circuit garbling against malicious quantum adversaries.

2.3 A Garbling Scheme for $\mathsf{C}+\mathsf{M}$ Circuits

Our first step is to formalize the proposal sketched in [15, §2.5] for garbling $\mathsf{C}+\mathsf{M}$ circuits. The starting point for the [15, §2.5] construction is a simple technique for garbling any quantum circuit that consists of a single Clifford unitary F.[5] The idea is to sample a random Clifford E and give out $FE^\dagger$ as the garbled circuit; note that the description of $FE^\dagger$ will be entirely classical. Since the Clifford unitaries form a group, $FE^\dagger$ is a uniformly random Clifford unitary independent of F. To garble the input quantum state $\mathbf{x}$, simply compute $E(\mathbf{x})$. The construction in [15, §2.5] extends this simple construction to any circuit.

To build intuition, we will consider a two-layer $\mathsf{C}+\mathsf{M}$ circuit $Q = (F_1, f)$, where F_1 is the first layer Clifford unitary, and f is a classical circuit that takes as input a single bit measurement result m, and outputs a classical description of F_2, the second layer Clifford unitary. On input $\mathbf{x}$, the circuit operates as follows:

1. Apply F_1 to $\mathbf{x}$.
2. Measure the last output wire in the computational basis to obtain $m \in \{0,1\}$, and feed the remaining wires to the next layer. Compute the second layer Clifford unitary $F_2 = f(m)$.
3. Apply F_2 to the non-measured output wires from the first layer. Return the result.

One could try to extend the simple idea for one-layer garbling to this circuit. We still sample a random input-garbling Clifford E_0 and compute $F_1E_0^\dagger$. To hide the second layer Clifford, a natural idea is to sample yet another random Clifford E_1 to be applied to the non-measured output wires of F_1. That is, we replace $F_1E_0^\dagger$ with $(E_1 \otimes \mathbb{I})F_1E_0^\dagger$, and release the description of a function g such that $g(m) = f(m)E_1^\dagger$.

[4] We remark that the 2PQC proposed in [15] is based on their "main" quantum garbled circuit construction, which crucially does *not* have a classical circuit garbling procedure. The advantage of their main construction is that garbling can be done in low depth, whereas the alternative construction requires an expensive but classical garbling procedure.

[5] [15] call this *group-randomizing quantum randomized encoding*.

However, this may in general be insecure. Let $F_2^{(0)}$ be the Clifford output by function f when $m = 0$, and $F_2^{(1)}$ the Clifford output by function f when $m = 1$. Suppose $F_2^{(0)} - F_2^{(1)} = A$ for some invertible matrix A. Then, an attacker with access to g could obtain $F_2^{(0)}E_1^\dagger - F_2^{(1)}E_1^\dagger$, and multiplying the result by A^{-1} yields $A^{-1}(F_2^{(0)}E_1^\dagger - F_2^{(1)}E_1^\dagger) = A^{-1}AE_1^\dagger = E_1^\dagger$.

Therefore, instead of giving out g, the construction of [15, §2.5] gives out a classical garbling of g. To accommodate this, the output wire from the first layer that is measured to produce $m \in \{0, 1\}$ must be replaced by a collection of wires that produces the corresponding label lab_m for the garbled circuit. This can be easily achieved by applying a suitable "label unitary" to the m wire (and ancilla wires) within the garbled gate for the first layer.

There is one last issue with this approach: an attacker that chooses not to measure the wires containing lab_m can obtain a superposition over two valid labels. Recall that the standard definition of security for classical garbled circuits only guarantees simulation of one label, not a quantum superposition of both labels. To ensure the attacker cannot get away with skipping the computational basis measurement, the [15, §2.5] construction applies a Z-twirl to m before the "label unitary" is applied. Recall that a Z-twirl is simply a random application of a Pauli Z gate, i.e. Z^b for a uniformly random bit b; applying Z^b to a wire is equivalent to performing a computational basis measurement (without recording the result).

To recap, a garbled 2-layer $\mathsf{C}+\mathsf{M}$ circuit Q consists of three components: an "input garbling" Clifford E_0, an initial Clifford unitary to be applied to the garbled input $D_0 \coloneqq (E_1 \otimes \mathbb{I})F_1E_0^\dagger$, and a classical garbled circuit $\widetilde{g}$. Extrapolating, we see that in general a garbled $\mathsf{C}+\mathsf{M}$ circuit takes the form

$$(E_0, D_0, \widetilde{g}_1, \ldots, \widetilde{g}_d) \coloneqq (E_0, \widetilde{Q}),$$

where the $\widetilde{g}_i$'s are garblings of classical circuits. Crucially, all of these components can be generated by an entirely classical circuit. The only quantum operation involved in the garbling process is the application of E_0 to the input $\mathbf{x}$ to garble the input. Next, we show how we can take advantage of this mostly classical garbling procedure to obtain maliciously-secure 2PQC.

2.4 A Three-Message Protocol with Malicious Security

In this section, we describe a three-message 2PQC protocol where both parties obtain output. This implies the two-message 2PQC result with one-sided output described in the first part of our results section, and fills in the upper left corner of Table 1.

We begin with a plausible but *insecure* construction of a three-message 2PQC based on the above quantum garbled circuit construction. We will then highlight the ways a malicious attacker might break this construction, and arrive at our final construction by implementing suitable modifications.

Our protocol relies only on a *classical* two-message 2PC with one-sided output that is (post-quantum) secure against malicious adversaries; this can be

realized by combining (post-quantum) classical garbled circuits [40] with (post-quantum) two-message oblivious transfer [36] following eg. [30].

We will consider two parties: Alice with input $\mathbf{x}_A$ and Bob with input $\mathbf{x}_B$. They wish to jointly compute a quantum circuit Q on their inputs whose output is delivered to both players. Q is represented as a Clifford+Measurement circuit that takes input $(\mathbf{x}_A, \mathbf{x}_B, \mathbf{T}^k, \mathbf{0}^k)$. We denote by $(\mathbf{y}_A, \mathbf{y}_B)$ the joint outputs of Alice and Bob. At a high level, the parties will use the first two messages (Bob $\to$ Alice, Alice $\to$ Bob) to jointly encode their quantum inputs, while in parallel computing a two-message classical 2PC that outputs the classical description of a quantum garbled circuit to Bob. By evaluating the garbled circuit, Bob can learn his own output, as well as Alice's encoded output, which he sends to Alice in the 3rd message.

In more detail, the classical functionality $\mathcal{F}[Q]$ to be computed by the classical 2PC is defined as follows. It takes as input (the classical description of) a Clifford unitary $C_{B,\text{in}}$ from Bob and Clifford unitaries $(C_{A,\text{in}}, C_{A,\text{out}})$ from Alice. Let Q_B be a modification of Q that outputs $(C_{A,\text{out}}(\mathbf{y}_A, \mathbf{0}^\lambda), \mathbf{y}_B)$ in place of $(\mathbf{y}_A, \mathbf{y}_B)$; looking ahead, this will enable Bob to evaluate (a garbling of) Q_B on (a garbling of) their joint inputs without learning Alice's output. The functionality computes a garbling $(E_0, \widetilde{Q_B})$ of Q_B. Finally, it computes $W \coloneqq E_0 \cdot (\mathbb{I} \otimes C_{B,\text{in}}^{-1} \otimes \mathbb{I}) \cdot C_{A,\text{in}}^{-1}$ (where the registers implied by the tensor product will become clear below), and outputs $(W, \widetilde{Q_B})$ to Bob.

The (insecure) protocol template is as follows:

- **First Message (Bob $\to$ Alice).** Bob picks a random Clifford $C_{B,\text{in}}$ and uses it to encrypt and authenticate his input $\mathbf{x}_B$ as $\mathbf{m}_1 \coloneqq C_{B,\text{in}}(\mathbf{x}_B, \mathbf{0}^\lambda)$. He also computes the first round message m_1 of the classical 2PC, using $C_{B,\text{in}}$ as his input. He sends $(\mathbf{m}_1, m_1)$ to Alice.
- **Second Message (Alice $\to$ Bob).** After receiving $(\mathbf{m}_1, m_1)$, Alice picks a random Clifford $C_{A,\text{in}}$ and uses it to encrypt her input $\mathbf{x}_A$ along with Bob's encoding $\mathbf{m}_1$, k copies of a $\mathbf{T}$ state, and $k+\lambda$ copies of a $\mathbf{0}$ state. The result of this is $\mathbf{m}_2 \coloneqq C_{A,\text{in}}(\mathbf{x}_A, \mathbf{m}_1, \mathbf{T}^k, \mathbf{0}^{k+\lambda})$. Alice also samples another random Clifford $C_{A,\text{out}}$ that will serve to encrypt and authenticate her output, and computes the second round message m_2 of the classical 2PC using input $(C_{A,\text{in}}, C_{A,\text{out}})$. She sends $(\mathbf{m}_2, m_2)$ to Bob.
- **Third Message (Bob $\to$ Alice).** After receiving $(\mathbf{m}_2, m_2)$, Bob can compute his output of the classical 2PC, which is $(W, \widetilde{Q_B})$. He computes

  $$W(\mathbf{m}_2) = E_0 \cdot (\mathbb{I} \otimes C_{B,\text{in}}^{-1} \otimes \mathbb{I}) \cdot C_{A,\text{in}}^{-1}\left(C_{A,\text{in}}(\mathbf{x}_A, \mathbf{m}_1, \mathbf{T}^k, \mathbf{0}^{k+\lambda})\right) = E_0(\mathbf{x}_A, \mathbf{x}_B, \mathbf{T}^k, \mathbf{0}^{k+\lambda}).$$

  Recall that $E_0(\mathbf{x}_A, \mathbf{x}_B, \mathbf{T}^k, \mathbf{0}^{k+\lambda})$ corresponds to a garbled input for $\widetilde{Q_B}$. He evaluates $\widetilde{Q_B}$ on this garbled input and obtains $(C_{A,\text{out}}(\mathbf{y}_A, \mathbf{0}^\lambda), \mathbf{y}_B)$.
  At this point, Bob has his output $\mathbf{y}_B$ in the clear. Next he sets $\mathbf{m}_3 = C_{A,\text{out}}(\mathbf{y}_A, \mathbf{0}^\lambda)$, and sends $\mathbf{m}_3$ to Alice. Upon receiving $\mathbf{m}_3$, Alice can recover her output by computing $C_{A,\text{out}}^{-1}(\mathbf{m}_3)$.

The above protocol can already be shown to be secure against malicious Bob by relying on security of the classical two-party computation protocol against

malicious adversaries. But malicious Alice can break security by generating ill-formed auxiliary states. We now describe this issue in some more detail and then present modifications to address the problem.

Malicious Generation of Auxiliary States. In the second message of the protocol, Alice is instructed to send a quantum state $C_{A,\text{in}}(\mathbf{x}_A, \mathbf{m}_1, \mathbf{T}^k, \mathbf{0}^{k+\lambda})$. A malicious Alice can deviate from honest behavior by submitting arbitrary states in place of the magic $\mathbf{T}$ states and the auxiliary $\mathbf{0}$ states, either of which may compromise security.

We therefore modify the classical 2PC to include randomized checks that will enable Bob to detect if Alice has deviated from honest behavior.

We check validity of $\mathbf{0}$ states using the "random linear map" technique of [22]. The classical 2PC will sample a uniformly random matrix $M \in \mathbb{F}_2^{k\times k}$, and apply a unitary U_M that maps the quantum state $\mathbf{v} = |v\rangle\langle v|$ for any $v \in \mathbb{F}_2^k$ to the state $\mathbf{Mv} = |Mv\rangle\langle Mv|$. For any $M \in \mathbb{F}_2^{k\times k}$, there exists an efficient Clifford unitary U_M implementing this map. This check takes advantage of the fact that $U_M(\mathbf{0}^k) = \mathbf{0}^k$ for any M, but on any other pure state $\mathbf{v} = |v\rangle\langle v|$ for non-zero $v \in \mathbb{F}_2^k$, we have $U_M(\mathbf{v}) \neq \mathbf{0}^k$ with overwhelming probability in k.

More precisely, our protocol will now ask Alice to prepare twice ($2k$) the required number of $\mathbf{0}$ states. The classical 2PC will generate a Clifford unitary U_M implementing a random linear map $M \in \mathbb{F}_2^{2k\times 2k}$, and incorporate U_M into its output Clifford W, which is now $W = (E_0 \otimes \mathbb{I}) \cdot (\mathbb{I} \otimes C_{B,\text{in}}^{-1} \otimes \mathbb{I}) \cdot (\mathbb{I} \otimes U_M) \cdot C_{A,\text{in}}^{-1}$. Now when Bob applies W to Alice's message $C_{A,\text{in}}(\mathbf{x}_A, C_{B,\text{in}}(\mathbf{x}_B, \mathbf{0}^\lambda), \mathbf{T}^k, \mathbf{0}^{2k})$, it has the effect of stripping off $C_{A,\text{in}}$ by applying $C_{A,\text{in}}^{-1}$, and then applying U_M to the last $2k$ registers. The rest of the application of W has the same effect as before the modification, so it undoes the application of $C_{B,\text{in}}$, and then re-encodes *all but the last k registers* under the input garbling Clifford E_0 to produce a garbled input. Crucially, the last k registers are designated "$\mathbf{0}$-state check registers", which Bob can simply measure in the computational basis to detect if Alice prepared the $\mathbf{0}$ states properly.

Unfortunately, this technique does not extend to checking validity of $\mathbf{T}$ states. To do so, we would have to map $\mathbf{T}$ states to $\mathbf{0}$ states, but there is no Clifford unitary that realizes this transformation.[6] The problem with using a non-Clifford unitary is that security of W relies on the fact that it is the product of a random Clifford $C_{A,\text{in}}$ and some other Clifford W'. Since the Clifford unitaries form a group, multiplication by a random $C_{A,\text{in}}$ perfectly masks the details of W', but only when W' is Clifford.

We will therefore employ the "cut-and-choose" technique from [22]. The protocol will now have Alice prepare $\lambda(k+1)$-many $\mathbf{T}$ states instead of just k. The classical 2PC will generate a random permutation π on $[\lambda(k+1)]$, which will move a random selection of λ of the $\mathbf{T}$ states into "$\mathbf{T}$-state check registers." The application of π will be implemented by a unitary U_π incorporated into W. Af-

[6] The existence of such a Clifford would imply that Clifford + Measurement circuits *without* magic states are universal for quantum computing, contradicting the Gottesman–Knill theorem (assuming $\mathsf{BPP} \neq \mathsf{BQP}$).

ter applying W, Bob will apply a projective measurement onto $\mathbf{T}$ to each of the $\mathbf{T}$-state check registers, and will abort if any of the λ measurements fails.

If all of the λ measurements pass, this means the remaining λk un-tested $\mathbf{T}$ states are "somewhat close" to being real $\mathbf{T}$ states. However, being "somewhat close" will not be sufficient; for instance, an attacker who prepares exactly one completely invalid $\mathbf{T}$ state will only be caught with $1/(k+1)$ probability.

We will therefore need to apply magic-state distillation to transform these into states which are negligibly close to real $\mathbf{T}$ states. For this, we use a magic-state distillation circuit of [22, §2.5] (which builds on [16]). This circuit consists solely of Clifford gates and computational basis measurements. To apply this circuit we modify our underlying functionality, so that we now give out a garbling of a circuit that first implements magic-state distillation and only then applies Q_B.

This completes an overview of our protocol, and a formal construction and analysis can be found in Section 3.

2.5 Application: Reusable MDV-NIZK for QMA

Now we briefly describe how the above techniques readily give a reusable malicious designated-verifier NIZK for QMA in the CRS model. Note that NIZK for QMA is a special case of two-party quantum computation, where the functionality being computed is the verification circuit $\mathcal{V}$ for some QMA language, the prover (previously Alice) has the quantum witness $\mathbf{w}$ as input, and the verifier (previously Bob) has no input and receives a binary output indicating whether $\mathcal{V}(x, \mathbf{w})$ accepts or rejects, where x is the (classical) description of the instance they are considering.

Since the prover does not receive output, there is no need for the third message in the protocol of Section 2.4. Furthermore, since the verifier has no input, there is no need for any quantum message from him in the first message. The verifier only needs to send a first-round classical 2PC message which then functions as a proving key. The (classical) left-over state is the verifier's secret verification key. After this, the prover just sends one quantum message (the Second Message in the above protocol), proving that $\mathcal{V}(x, \mathbf{w}) = 1$.

In order to make the above template reusable, we can first instantiate the underlying classical 2PC with a reusable 2PC. Once this is in place, the verifier's first-round message is necessarily instance-indepedent. Then, to ensure that a cheating prover cannot break soundness by observing whether the verifier accepts its proofs or not, we modify the classical functionality to take as input a PRF key from the verifier, and generate all required randomness (used for the $\mathbf{0}$ and $\mathbf{T}$ checks, and the quantum garbling procedure) by applying this PRF to the (classical) description of the instance x. By security of the reusable 2PC and the PRF, a verifier will never accept a maliciously sampled proof for any instance x not in the language.

2.6 Challenges in Achieving a Two-Round Protocol in the Quantum Setting

The previous sections show that we can achieve 2PQC in two messages if only one party receives output, which is optimal in terms of round complexity. Now we ask whether both parties can obtain output with just two rounds of simultaneous exchange. Indeed, in the classical setting, there is a natural approach to obtaining a two-round protocol, given a two-message protocol where one party receives output. The parties simply run two parallel executions of the two-message protocol on the same inputs - one in which Alice speaks first and the functionality only computes her part of the output, and another in which Bob speaks first and the functionality only computes his part of the output. Unfortunately, this natural approach completely fails in the quantum setting, for at least two reasons.

- Running two parallel executions of the same protocol on the same set of inputs seems to require *cloning* those inputs, which is in general impossible if the inputs may be arbitrary quantum states.
- Running two parallel executions of a randomized functionality requires the parties to fix the same random coins to be used in each execution, as otherwise their outputs may not be properly jointly distributed. This is not possible in the quantum setting, since randomness can come from measurement, and measurement results cannot be fixed and agreed upon beforehand.

These issues motivate the rest of our work. Since running two protocols in parallel on the same inputs is problematic, we take as our guiding principle that one party must be performing the actual computation at some point in the protocol, and then distributing the outputs.

Interestingly, while the first issue mentioned above is unique to the setting of quantum inputs, the second issue applies even if the parties wish to compute a quantum circuit over just *classical* inputs, which we regard as a very natural setting. Thus, while this paper focuses on the most general case of secure quantum computation over potentially quantum inputs, we stress that all the results we achieve are the best known even for the classical input setting. Furthermore, note that both issues also exist in the specious setting, so it doesn't appear to be straightforward to achieve two-round 2PQC even in this setting. While the focus of this paper is on the setting of malicious security, exploring these questions in the specious setting is also an interesting direction.

2.7 A Two-Round Protocol with Pre-Processing

Our next result is a three-round protocol for 2PQC which requires only two *online* rounds of communication, filling in the upper right corner of Table 1.

In fact, we construct a protocol in which the pre-input phase only consists of a *single* message from Bob to Alice (computed with respect to a CRS). We take our three sequential message protocol as a starting point, and introduce several modifications. The first modification will immediately achieve the goal of removing input-dependence from Bob's first message, and all the subsequent modifications will be necessary to restore correctness and security.

Modification 1: Removing Input-Dependence via Teleportation. Before sending his first message, Bob samples n EPR pairs, where n is the number of qubits of the input $\mathbf{x}_B$. We denote these EPR pairs by $(\mathbf{epr}_1, \mathbf{epr}_2)$, where $\mathbf{epr}_1$ denotes the left n qubits, and $\mathbf{epr}_2$ denotes the right n qubits. In place of sending $C_{B,\text{in}}(\mathbf{x}_B, \mathbf{0}^\lambda)$, Bob sends $\mathbf{m}_{B,1} \coloneqq C_{B,\text{in}}(\mathbf{epr}_1, \mathbf{0}^\lambda)$. Note that the classical 2PC only requires input $C_{B,\text{in}}$, which is a random Clifford that Bob samples for himself, so Bob's entire first round message $(\mathbf{m}_{B,1}, m_{B,1})$ can now be sent *before* Bob receives his input. The idea is that later on, when Bob learns his input $\mathbf{x}_B$, he will perform Bell measurements on $(\mathbf{x}_B, \mathbf{epr}_2)$ to teleport $\mathbf{x}_B$ into $\mathbf{epr}_1$.

Issue: Incorporating Bob's Teleportation Errors. Teleporting $\mathbf{x}_B$ into $\mathbf{epr}_1$ will require Bob to somehow correct $\mathbf{epr}_1$ later in the protocol using the results of his Bell measurements on $(\mathbf{x}_B, \mathbf{epr}_2)$. But enabling Bob to do this in a way that does not compromise security will be tricky, as we now explain.

After receiving the second round message from Alice in our original malicious 2PQC protocol, Bob learns the output of the classical 2PC, which includes (1) a (classical description of a) quantum garbled circuit $\widetilde{Q}$, and (2) a Clifford unitary W. Bob applies W to Alice's quantum message $\mathbf{m}_{A,2}$, performs the appropriate $\mathbf{0}$ and $\mathbf{T}$ state checks, and conditioned on the checks passing, is left with a state of the form $E_0(\mathbf{x}_A, \mathbf{x}_B, \widehat{\mathbf{T}}, \mathbf{0})$, where $\widehat{\mathbf{T}}$ is a state "somewhat close" to $\mathbf{T}^k$. But at this point in our newly modified protocol, Bob is holding the state $E_0(\mathbf{x}_A, \mathbf{epr}_1, \widehat{\mathbf{T}}, \mathbf{0})$. To restore correctness, we somehow need to modify the protocol so that Bob can apply $X^{x_{\text{inp}}} Z^{z_{\text{inp}}}$ to $\mathbf{epr}_1$ "inside" the E_0 mask, where $x_{\text{inp}}, z_{\text{inp}}$ are the result of Bell basis measurements on $(\mathbf{x}_B, \mathbf{epr}_2)$.

Recall that the structure of W is $W = E_0 \cdot U^\dagger_{\text{dec-check}}$, where E_0 is the input garbling Clifford for the quantum garbled circuit, and $U_{\text{dec-check}}$ is the matrix that undoes $C_{A,\text{in}}$, undoes $C_{B,\text{in}}$, and then applies a permutation π and a random linear map M, and rearranges all the to-be-checked registers to the last few (rightmost) register slots. The multiplication by E_0 is applied only to the non-checked registers.

Thus, it seems like correctness would have to be restored by inserting the unitary $(\mathbb{I} \otimes X^{x_{\text{inp}}} Z^{z_{\text{inp}}} \otimes \mathbb{I})$ in between E_0 and $U^\dagger_{\text{dec-check}}$. But if Bob can learn $E_0(\mathbb{I} \otimes X^{x_{\text{inp}}} Z^{z_{\text{inp}}} \otimes \mathbb{I}) U^\dagger_{\text{dec-check}}$ for even two different values of x_{inp} and z_{inp}, security of the input garbling Clifford E_0 may be lost entirely.

Modification 2: Classical Garbling + Quantum Multi-Key Fully Homomorphic Encryption In order to resolve this issue, we will split up the matrix $E_0(\mathbb{I} \otimes X^{x_{\text{inp}}} Z^{z_{\text{inp}}} \otimes \mathbb{I}) U^\dagger_{\text{dec-check}}$ into two matrices

$$U_{x_{\text{inp}}, z_{\text{inp}}} \coloneqq E_0(\mathbb{I} \otimes X^{x_{\text{inp}}} Z^{z_{\text{inp}}} \otimes \mathbb{I}) U^\dagger_{\text{rand}}$$
$$U_{\text{check}} \coloneqq U_{\text{rand}} U^\dagger_{\text{dec-check}}$$

where U_{rand} is a "re-randomizing" Clifford.

The matrix U_{check} is independent of Bob's teleportation errors, and will now be output to Bob by the classical 2PC. But to preserve security, we will

have Bob obtain $U_{x_{\text{inp}},z_{\text{inp}}}$ by evaluating a *classical* garbled circuit $\widetilde{f}_{\text{inp}}$ where $f_{\text{inp}}(x_{\text{inp}}, z_{\text{inp}}) \coloneqq U_{x_{\text{inp}},z_{\text{inp}}}$; the garbled circuit $\widetilde{f}_{\text{inp}}$ is included in the output of the classical 2PC.

But now we are faced with a new problem: how does Bob obtain the (classical) labels for $\widetilde{f}_{\text{inp}}$? Since we only have one round of interaction remaining, Bob won't be able to run an OT to learn the correct labels (Bob could learn the labels by the end of the two online rounds, but then we would still need another round for Bob to send Alice her encrypted output).

We resolve this problem with *quantum multi-key fully-homomorphic encryption* (QMFHE), which we will use in tandem with our classical garbled circuit $\widetilde{f}_{\text{inp}}$ to enable Bob to compute (a homomorphic encryption of) $U_{x_{\text{inp}},z_{\text{inp}}}$ without leaking anything else. Before we continue, we give a brief, intuition-level recap of QMFHE (we refer the reader to the full version for a formal description). Recall that a standard fully-homomorphic encryption (FHE) allows one to apply arbitrary efficient computation to encrypted data (without needing to first decrypt). *Multi-key* FHE (MFHE) extends FHE to enable computation over multiple ciphertexts encrypted under different keys; the output of such a homomorphic computation is a "multi-key" ciphertext which can only be decrypted given all the secret keys for all of the ciphertexts involved in the computation [33]. Finally, QMFHE extends MFHE a step further to allow arbitrary efficient *quantum* computation over encrypted (classical or quantum) data [29,13,34,1].

We will encrypt each of the garbled circuit labels for $\widetilde{f}_{\text{inp}}$ under an independent QMFHE key. All of these encrypted labels along with the corresponding QMFHE public keys (to enable quantum computations over these ciphertexts) will also be output to Bob as part of the classical 2PC. We remark that this requires a QMFHE scheme where encryptions of classical plaintexts are themselves classical; such schemes are known assuming the quantum hardness of the learning with errors (QLWE) assumption [1].[7]

To recap, Bob obtains from the classical 2PC a collection of QMFHE ciphertexts, one for each of the garbled circuit labels for $\widetilde{f}_{\text{inp}}$. Bob picks out the ciphertexts corresponding to $x_{\text{inp}}, z_{\text{inp}}$ and performs quantum multi-key evaluation of $\widetilde{f}_{\text{inp}}$ over these ciphertexts, obtaining a QMFHE encryption of the output of $\widetilde{f}_{\text{inp}}$, i.e. $\mathsf{QMFHE.Enc}(\mathsf{pk}_{x_{\text{inp}},z_{\text{inp}}}, U_{x_{\text{inp}},z_{\text{inp}}})$ where $\mathsf{pk}_{x_{\text{inp}},z_{\text{inp}}}$ denotes the collection of QMFHE public keys corresponding to $x_{\text{inp}}, z_{\text{inp}}$. The classical 2PC output also includes U_{check} in the clear, which Bob can apply to $\mathbf{m}_{A,2}$ to obtain $U_{\text{rand}}(\mathbf{x}_A, \mathbf{epr}_1, \widehat{\mathbf{T}}, \mathbf{0})$ (after performing appropriate measurement checks). Then Bob can homomorphically compute the ciphertext $\mathsf{QMFHE.Enc}(\mathsf{pk}_{x_{\text{inp}},z_{\text{inp}}}, E_0(\mathbf{x}_A, \mathbf{x}_B, \widehat{\mathbf{T}}, \mathbf{0}))$, and proceed to homomorphically evaluate his quantum garbled circuit to obtain $\mathsf{QMFHE.Enc}(\mathsf{pk}_{x_{\text{inp}},z_{\text{inp}}}, (C_{A,\text{out}}(\mathbf{y}_A, \mathbf{0}^\lambda), \mathbf{y}_B))$.

In order for Bob to obtain his final output in the clear, we will have Bob send Alice $x_{\text{inp}}, z_{\text{inp}}$ in the first online round. In response, in the second online round Alice will reply with $\mathsf{sk}_{x_{\text{inp}},z_{\text{inp}}}$; security of the QMFHE will guar-

[7] We only require *leveled* QMFHE, which can be based solely on the QLWE assumption. Unleveled QMFHE requires an additional circularity security assumption.

antee that Bob cannot decrypt ciphertexts corresponding to any other choice of the teleportation errors. In the second online round, Bob will send Alice $\mathsf{QMFHE.Enc}(\mathsf{pk}_{x_{\mathrm{inp}},z_{\mathrm{inp}}}, (C_{A,\mathrm{out}}(\mathbf{y}_A, \mathbf{0}^\lambda))$, which she can decrypt to obtain $\mathbf{y}_A$. Finally, Bob produces his output by performing QMFHE decryption with $\mathsf{sk}_{x_{\mathrm{inp}},z_{\mathrm{inp}}}$.

Issue: Simulating a Quantum Garbled Circuit with Unknown Output. At this point, we have a correct protocol whose first round is completely input-independent. However, we will run into issues when attempting to prove malicious security.

The problem arises in the security proof for a malicious Bob. In the original three-round maliciously secure protocol, the simulator is able to extract $\mathbf{x}_B$ from Bob's first round message to Alice; this is done by first extracting $C_{B,\mathrm{in}}$ from Bob's first round classical message for the classical 2PC, and then applying $C_{B,\mathrm{in}}^{-1}$ to Bob's first round quantum message. Extracting $\mathbf{x}_B$ from Bob's first round message to Alice is crucial for proving security, since it enables the simulator to query the ideal functionality on $\mathbf{x}_B$, learn the output $\mathbf{y}_B$, and finally simulate the quantum garbled circuit using Bob's output $\mathbf{y}_B$ before computing Alice's simulated second round message to be sent to Bob. This second round message reveals to Bob the quantum garbled circuit, so it is crucial that the quantum garbled circuit simulator has been executed at this point.

Not surprisingly, this simulation strategy runs into a major problem in our newly modified protocol. Bob's first message is independent of $\mathbf{x}_B$, so the simulator cannot query the ideal functionality, and therefore seemingly cannot simulate the quantum garbled circuit before computing Alice's message, which in particular reveals the quantum garbled circuit to Bob. In summary, the simulator must provide Bob with the quantum garbled circuit (part of Alice's first online round message), *before* it has enough information to extract Bob's input. This appears quite problematic since simulating a garbled circuit certainly requires knowing the output. However, since Bob can only obtain an *encryption* of the output of the garbled circuit after receiving Alice's first message, it is still reasonable to expect that the protocol is secure.

Modification 3: Simulation via Teleportation. We fix this problem through a new technique we call *simulation via teleportation*. The idea is as follows. Instead of running the quantum garbled circuit simulator on the output of the circuit (which the simulator does not yet know), the simulator will first prepare fresh EPR pairs $\mathbf{epr}'_1, \mathbf{epr}'_2$ and then run the quantum garbled circuit simulator on $(C_{A,\mathrm{out}}(\mathbf{0}, \mathbf{0}^\lambda), \mathbf{epr}'_1)$ (where $\mathbf{0}$ takes the place of Alice's input $\mathbf{x}_A$ and $\mathbf{epr}'_1$ takes the place of Bob's output $\mathbf{y}_B$). In the following round, after Bob has teleported over his input state $\mathbf{x}_B$, the simulator will query the ideal functionality, learn $\mathbf{y}_B$, and then *teleport* $\mathbf{y}_B$ *into* $\mathbf{epr}'_1$.

Implementing the final teleportation step requires some care. When the simulator learns $\mathbf{y}_B$, it performs Bell measurements on $(\mathbf{y}_B, \mathbf{epr}'_2)$, obtaining measurement outcomes $x_{\mathrm{out}}, z_{\mathrm{out}}$. It must then find some way to apply $x_{\mathrm{out}}, z_{\mathrm{out}}$ to the state $\mathbf{epr}'_1$ so that Bob can obtain his correct output.

So we further modify the protocol so that the garbled circuit Bob receives from the classical 2PC is modified to output $(C_{A,\mathrm{out}}(\mathbf{y}_A, \mathbf{0}^\lambda), X^{x_{\mathrm{out}}} Z^{z_{\mathrm{out}}} \mathbf{y}_B)$ in-

stead of $(C_{A,\text{out}}(\mathbf{y}_A, \mathbf{0}^\lambda), \mathbf{y}_B)$, as before. That is, in the real protocol, an honest Alice will sample random $x_{\text{out}}, z_{\text{out}}$, and then the 2PC will output the circuit implementing this functionality. Alice will send $x_{\text{out}}, z_{\text{out}}$ to Bob in the second online round, and Bob will first apply Pauli corrections $X^{x_{\text{out}}} Z^{z_{\text{out}}}$ to his output to obtain $\mathbf{y}_B$. In the simulated protocol, however, $x_{\text{out}}, z_{\text{out}}$ are not sampled by the simulator. Instead, they are the result of the simulator's Bell measurements on $(\mathbf{y}_B, \mathbf{epr}'_2)$. The simulator thus simulates a garbled circuit that outputs $(C_{A,\text{out}}(\mathbf{0}, \mathbf{0}^\lambda), \mathbf{epr}'_1)$, and then sends $x_{\text{out}}, z_{\text{out}}$ in the second online round. Note that this teleportation step occurs *exclusively within the simulation.*

Modification 4: Alice (Equivocally) Commits to Pauli Corrections. To arrive at a fully secure protocol, we need to address one last issue. As currently described, there is nothing that prevents a malicious Alice from misreporting her choice of $x_{\text{out}}, z_{\text{out}}$. This can introduce arbitrary Pauli errors into Bob's output that he has no way of detecting. However, this can easily be fixed using equivocal commitments. That is, Alice inputs $x_{\text{out}}, z_{\text{out}}$ to the classical 2PC, along with commitment randomness s. Bob obtains the commitment as part of the output of the classical 2PC, and later when Alice sends $x_{\text{out}}, z_{\text{out}}$ in the second online round, she must also send along s. The equivocality property enables the simulation strategy to work as before, as the simulator will have the power to send Bob a commitment to an arbitrary value, and after learning $x_{\text{out}}, z_{\text{out}}$ from its Bell measurements, use equivocation to produce a valid opening.

2.8 The Multi-Party Setting

In this section, we describe our results in the multi-party setting, filling in the bottom row of Table 1.

We begin by describing our approach to obtaining a five-round protocol from quantum-secure OT. Our approach follows the same high-level idea as the three-message 2PQC protocol described in Section 2.4, where one party (the "designated party", or P_1) will evaluate a quantum garbled circuit on encodings of each party's input, and then distribute the encoded outputs to each party. However, implementing this template in the multi-party setting requires resolving a host of new challenges.

Input Encoding. Recall that in our two-party protocol, Alice received an encoding of Bob's input, concatenated their own input, re-randomized the entire set of registers with a random Clifford C, and then sent the re-randomized state to Bob. This re-randomization ensures that the only meaningful computation Bob can perform is to apply the quantum garbled circuit, whose classical description is re-randomized with $C^\dagger$. A natural extension of this idea to the multi-party setting goes as follows. First, each party sends their encoded input to P_1. Then P_1 concatenates all inputs together and re-randomizes the resulting set of registers with their own random Clifford C_1. Then, these registers are passed around in a circle, each party P_i applying their own re-randomizing Clifford C_i. Finally, P_1 receives the fully re-randomized state, along with some classical description

of a quantum garbled circuit obtained via classical MPC, and re-randomized with $C_1^\dagger \dots C_n^\dagger$. The fact that each party applies their own re-randomizing Clifford is necessary, since we are in the dishonest majority setting. Indeed, if only one party P_i is honest, their security will crucially rely on the fact that the adversary does not know their re-randomizing Clifford C_i. This approach of encrypting and sending a state around the circle of parties for re-randomization is similar to [22]'s "input encoding" protocol, in which each individual party's input is sent around the circle of parties for re-randomization.

Unfortunately, the round complexity of this encoding step will grow linearly with the number of parties. To obtain a constant-round protocol, our idea is to round-collapse this input-encoding via the use of quantum teleportation. In the first round, parties will send EPR pairs to each other following the topology of the computation described above. That is, each party sets up EPR pairs with P_1 that will be used to teleport their encoded inputs to P_1, and each consecutive pair of parties will set up EPR pairs that will be used to teleport the encoded state around the circle. After this setup, the parties can *simultaneously* apply re-randomization Cliffords and teleport the encoded state around the circle. This will introduce teleportation errors, but since the re-randomization operations are Clifford, these can be later corrected. Indeed, this correction will be facilitated by a classical MPC protocol that takes as input each party's Clifford and set of teleportation errors.

0 and T State Checks. The next challenge is how to enforce 0 and T state checks in the multi-party setting. Recall that in the two-party setting, we had the non-evaluator party (Alice) prepare the 0 and T states, which were then checked by the garbled circuit evaluator (Bob). This approach works because we know that if Alice is malicious and tried to cheat during preparation of these states, then Bob must be honest and will then refuse to evaluate the garbled circuit. However, this does not carry over to the multi-party setting. If we try to fix some party P_i to prepare the 0 and T states and then have the evaluator P_1 check them, it may be the case that *both* P_i and P_1 are malicious, which would be problematic.

Thus, we take a different approach, instructing P_1 to prepare the 0 and T states, and designing a *distributed* checking protocol, similar to that of [22]. We now briefly describe the T state check, leaving a description of the 0 state check to the body. P_1 will be instructed to concatenate all parties' inputs with their own T states, and then send the resulting state around the circle for re-randomization. Later, they receive the re-randomized state, along with a unitary from the classical MPC that i) undoes the re-randomization, ii) samples a different subset of T states for each party, iii) Clifford-encodes each subset, and iv) garbles the inputs together with the remaining T states. Thus, P_1 obtains n encoded subsets of T states, and is supposed to send one to each party. Each party will then receive their encoded subset, decode (using information obtained from the classical MPC), and measure in the T-basis. Each party will then abort the protocol if their check failed. Only if *no* parties abort will the classical MPC send information to each party allowing them to decrypt their output from the

quantum garbled circuit. It is crucial that *no* party receives output until all honest parties indicate that their T state check passed, because using malformed T states in the quantum garbled circuit could result in outputs that leak information about honest party inputs.

The Five-Round Protocol. We give a high-level overview of the five rounds of the protocol.

- Round 1: Each party P_i generates EPR pairs and sends half of each pair to its neighbor P_{i+1}. Additionally, party P_1 generates enough EPR pairs so that it can send EPR pair halves to every other party P_i for $i \neq 1$.
- Round 2: Teleport inputs to P_1 and teleport the resulting state around the circle (with re-randomization Cliffords C_i applied along the way). Input teleportation errors and $\{C_i\}_{i\in[n]}$ to the classical MPC.
- Round 3: Classical MPC delivers unitary to P_1 that samples subsets of T states and garbles inputs, along with classical description of the quantum garbled circuit.
- Round 4: P_1 evaluates the unitary and garbled circuit, then delivers encoded subsets of T states and encrypted outputs to each party.
- Round 5: If no parties abort after their T state check, the classical MPC delivers key to each party allowing them to decrypt their output.

Note that the distributed T state check is the reason that the protocol requires five rounds. The first round is used for setting up EPR pairs. At this point the parties can perform quantum teleportation and obtain their Pauli errors. Now, these must be corrected by the classical MPC, which takes a minimum of two rounds. Thus, P_1 can only obtain output from the MPC, and thus from the quantum garbled circuit, after Round 3. Then, Round 4 must be used to distribute subsets of T states, and Round 5 must be used to deliver decryption keys conditioned on all parties being happy with their T states. As we describe in the body, the actual computation of the garbled circuit can be delayed one round (at the cost of settling for security with abort rather than unanimous abort), giving a five-round protocol with three online rounds.

Now we discuss how to instantiate the classical MPC. We are going to need an MPC that supports *reactive* functionalities, where inputs may depend on previous outputs obtained from the MPC. Moreover, we need the MPC to be *round-optimal*, in the sense that outputs delivered in round i may depend on inputs from round $i-1$. We observe that the round-collapsing compiler of [26] gives exactly this — an $\ell+1$ round MPC for a reactive functionality with ℓ rounds of output. Thus, we can rely solely on quantum-secure two-message OT to construct the above five-round quantum MPC.

The Four-Round Protocol. Finally, we observe that there is some slack in the aforementioned protocol. Indeed, P_1 does not obtain any output from the classical MPC until after round 3, when in principle the classical MPC can be used to compute some output in only two rounds. The reason we waited three rounds

is that we wanted to include the parties' teleportation errors in the computation performed by the MPC, and these are not known until the beginning of the second round.

However, we can use ideas similar to those in Section 2.7 in order to allow the MPC to compute something meaningful during the first two rounds without yet knowing the teleportation errors. In particular, we make use of classical garbled circuits and quantum multi-key FHE to provide a mechanism by which the classical MPC can output information allowing P_1 to (homomorphically) compute a function of the teleportation errors after Round 2. This allows us to collapse the total number of required rounds to 4. Moreover, a similar idea allows the parties to delay teleportation of their inputs another round, giving a four-round protocol with (optimal) *two* rounds of online interaction. Equivalently, our protocol can be seen as two-round MPQC in a quantum pre-processing model.

2.9 Two Round 2PQC Without Pre-Processing: Challenges and Possibilities

In this section, we explore the possibility of achieving a two-round 2PQC protocol in the CRS model *without pre-processing*. We stress that this model *does not permit pre-shared entanglement* between the two parties, as we consider sharing of entanglement to be a pre-processing step.

The Challenge of Oblivious Simulation. In the classical setting, all known two-round two-party computation protocols (in the CRS model) can be modified so that security is proven via (what we call) an *oblivious simulator*.[8] That is, the simulator (1) only makes black-box queries to the adversary, (2) is straight-line (meaning it only runs the adversary a single time without rewinding), and (3) it generates the simulated CRS *independently of the choice of corrupted party* (between Alice and Bob).

By focusing on protocols with oblivious simulation, we can highlight an apparent difficulty of building secure two-round protocols for quantum functionalities in the CRS model. Assume without loss of generality that Alice is adversarial (the identical argument applies to Bob). Observe that if the first message that Alice sends is not computationally binding to her input $\mathbf{x}_A$, she can potentially cheat by *equivocating*, i.e. acting as if she had received a different input, and subsequently learn multiple outputs of the functionality. If the simulation is oblivious, then this reasoning applies simultaneously to Alice and Bob — that is, both parties must, in the first round, send computationally-binding commitments to their respective inputs. This is immediately problematic for quantum inputs, since no-cloning implies that their leftover states will have no (computationally) useful information about their original inputs. Thus, it is unclear how a general computation can be performed on their *joint* inputs before the start

[8] Each party will use a NIZK proof of knowledge to prove that their first message is well-formed, using their input and randomness as witness. Then, a simulator programming the CRS may extract either party's input.

of the second round, as the parties have effectively swapped their initial states. And somehow, after just one more round of messaging, they must hold their correctly computed output states.

Our negative result formalizes this intuitive difficulty. If the simulator is oblivious, then by roughly following the above reasoning, at the end of the first round:

- Alice holds a computationally binding commitment to Bob's input $\mathbf{x}_B$,
- Bob holds a computationally binding commitment to Alice's input $\mathbf{x}_A$, and
- Neither party has information about their original inputs.

Moreover, the correctness of oblivious simulation implies that for a computationally indistinguishable CRS, there exists a "trapdoor" that would enable Alice to extract $\mathbf{x}_B$ and would enable Bob to extract $\mathbf{x}_A$. But now their states can be viewed as the states of two parties at the *beginning of a one-round protocol with polynomial-size pre-processing* in which the parties' inputs are *swapped*; the pre-processing step is necessary to give both parties the trapdoor information of the simulator. The resulting one-round protocol no longer satisfies any meaningful security guarantees, but crucially, it still satisfies correctness. Moreover, the one-round protocol falls into a model of "instantaneous non-local computation" that has been previously studied in the quantum information literature [10]. It is currently open whether this model enables general quantum computation with only polynomial-size preprocessing, and a positive result for two-round 2PQC with oblivious simulation would affirmatively answer this question.

A Proof-of-Concept Construction from Quantum VBB Obfuscation. Given the above barrier, one could attempt to construct a two-round protocol whose security relies crucially on a *non-oblivious* simulation strategy. In this work, we take an initial step in this direction by providing a proof-of-concept construction from a strong form of quantum VBB obfuscation that handles obfuscation of quantum circuits that include both unitary gates and measurement gates (further discussion is available in the full version).

In our construction, Alice will send an encryption of her input to Bob in round 1, who will then homomorphically compute the functionality over their joint inputs and respond with Alice's encrypted output in round 2. Alice will also send a message in round 2 that allows Bob to decrypt his output. However, the key is that this interaction will actually be indistinguishable from an interaction in which the *opposite* flow of computation is occuring. In particular, if the CRS if sampled differently (but in an indistinguishable way), it will be the case that Bob is actually sending his encrypted input to Alice in the first round, and then Alice homomorphically computes the functionality and sends Bob's encrypted output back in the second round.

To instantiate this template, we provide a number of quantum obfuscations in the CRS, three per party. First, there are the "input" obfuscations $\mathcal{O}_{A,\mathsf{inp}}$ and $\mathcal{O}_{B,\mathsf{inp}}$. $\mathcal{O}_{A,\mathsf{inp}}$ will take as input Alice's input $\mathbf{x}_A$ along with a "dummy" input $\mathbf{d}_A$, and output Clifford encodings of each. Alice is instructed to send the first output of this obfuscation as her first message, and keep the second output as

her state. In the real protocol, the obfuscated functionality will be such that the first output will be the Clifford encoding of the first input (Alice's real input $\mathbf{x}_A$), and the second output will be the Clifford encoding of the second input (Alice's dummy input $\mathbf{d}_A$). On the other hand, $\mathcal{O}_{B,\mathsf{inp}}$ will obfuscate the functionality that does the exact opposite, setting its first output to be a Clifford encoding of its second input, and its second output to be a Clifford encodings of its first input. Thus, in round 1, Alice sends a Clifford encoding of her real input and keeps a Clifford encoding of her dummy input in her state, while Bob sends a Clifford encoding of his dummy input and keeps a Clifford encoding of his real input in his state.

The next obfuscations $\mathcal{O}_{A,\mathsf{cmp}}$ and $\mathcal{O}_{B,\mathsf{cmp}}$ share secret randomness with the input obfuscations (in the form of PRF keys) and can thus decrypt Clifford encodings output by the input obfuscations. They each are defined to decrypt and check the authenticity of their inputs, apply the functionality Q that the parties wish to compute, and then encode the outputs with freshly sampled Cliffords. Each party will run their respective obfuscation on their state and the other party's first round message. Note that then Alice is just using $\mathcal{O}_{A,\mathsf{cmp}}$ to compute Q over dummy inputs, while Bob is using $\mathcal{O}_{B,\mathsf{cmp}}$ to compute Q over their real inputs. Alice will send an encrypted dummy output to Bob in round 2, while Bob will send an encrypted real output to Alice.

Finally, each party applies their respective output obfuscation $\mathcal{O}_{A,\mathsf{out}}$ and $\mathcal{O}_{B,\mathsf{out}}$ to their final state and other party's second round message. $\mathcal{O}_{A,\mathsf{out}}$ will ignore Alice's state (which contains Alice's dummy output) and decrypt and output Bob's second round message (which contains Alice's real output). On the other hand, $\mathcal{O}_{B,\mathsf{out}}$ will ignore Alice's second round message and decrypt and output Bob's state.

Now, it is possible to argue (under the assumption that the obfuscations in the CRS are in fact VBB obfuscations) that, because all intermediate states and messages are Clifford-encoded, "switching the direction" of the input and output obfuscations cannot be noticed by the parties. Note that if each of $\mathcal{O}_{A,\mathsf{inp}}$ and $\mathcal{O}_{B,\mathsf{inp}}$ are re-defined to permute the order of their outputs, then the flow of computation will be completely reversed. In particular, Alice will be computing the functionality over real inputs with $\mathcal{O}_{A,\mathsf{cmp}}$, and Bob will be computing the functionality over dummy inputs with $\mathcal{O}_{B,\mathsf{cmp}}$. Thus, depending on how the simulator programs the CRS, it can either extract directly from Alice's first round message OR it can extract directly from Bob's first round message, but it could never extract from both simultaneously.

Thus, this template represents a potential method for securely computing a quantum functionality in two rounds, where one of the two parties actually performs the computation between rounds 1 and 2 and then distributes the output in round 2. In other words, it is an instantiation of our guiding principle mentioned in Section 2.6 in a model without pre-processing.

3 Quantum Non-Interactive Secure Computation

3.1 Useful Lemmas

Lemma 1 (Magic State Distillation [16,22]). *Let $p(\cdot)$ be a polynomial. Then there exists a $\mathrm{poly}(\lambda)$ size $\mathsf{C}+\mathsf{M}$ circuit Q from $\lambda p(\lambda)$ input qubits to $p(\lambda)$ output qubits such that the following holds. Take any state $\mathbf{x}$ on $\lambda p(\lambda)+\lambda$ qubits. Apply a uniformly random permutation to the registers of $\mathbf{x}$ and then measure the final λ qubits in the T-basis to obtain a bitstring s. Let $\widetilde{\mathbf{x}}$ be the remaining $\lambda p(\lambda)$ registers. Then there exist negligible functions μ, ν such that*

$$\Pr\left[(s=0)\wedge\left(\left\|Q(\widetilde{\mathbf{x}})-\mathbf{T}^{p(\lambda)}\right\|_1>\mu(\lambda)\right)\right]\leq\nu(\lambda).$$

Proof. This follows from applying [22, Lemma I.1] with parameters $n=\lambda p(\lambda)$, $k=\lambda$, $\delta=1/2$ followed by [22, Lemma 2.7] with parameters $m=\lambda p(\lambda)$, $\ell=m/2, t=p(\lambda)$.

Lemma 2 ([22]). *For any $n\in\mathbb{N}$ and projector Π on $2n$ qubits, define the quantum channel $\mathcal{L}^{\Pi}$ by*

$$\mathcal{L}^{\Pi}(\mathbf{x}):=\Pi\mathbf{x}\Pi+|\bot\rangle\langle\bot|\operatorname{Tr}[(\mathbb{I}^{2n}-\Pi)\mathbf{x}],$$

where $|\bot\rangle$ is a distinguished state on $2n$ qubits with $\Pi|\bot\rangle=0$. For any $t\in\{0,1\}^n$, let $\Pi_{t,\mathsf{Full}}:=|0^{2n}\rangle\langle 0^{2n}|$ if $t=0^n$ and $\Pi_{t,\mathsf{Full}}:=0$ otherwise. Let $\Pi_{t,\mathsf{Half}}:=\mathbb{I}^n\otimes|t\rangle\langle t|$. Then for any QRV $\mathbf{x}$ on $2n$ registers and $t\in\{0,1\}^n$,

$$\left\|\mathcal{L}^{\Pi_{t,\mathsf{Full}}}(\mathbf{x})-\mathop{\mathbb{E}}_{U\leftarrow\mathsf{GL}(2n,\mathbb{F}_2)}\left[\mathcal{L}^{\Pi_{t,\mathsf{Half}}}(U(\mathbf{x}))\right]\right\|_1=\mathrm{negl}(n).$$

3.2 The Protocol

In what follows, we describe our protocol for two-party quantum computation in the setting of sequential messages. This protocol requires three messages of interaction when both players desire output, and two messages in a setting where only one party obtains an output, which can be seen as a Q-NISC (Quantum Non-interactive Secure Computation) protocol.

Ingredients. Our protocol will make use of the following cryptographic primitives: (1) Quantum-secure two-message two-party classical computation in the CRS model $(\mathsf{2PC.Gen}, \mathsf{2PC}_1, \mathsf{2PC}_2, \mathsf{2PC}_{\mathsf{out}})$ with a straight-line black-box simulator (see Section 3.4 of the full version), and (2) a garbling scheme for $\mathsf{C}+\mathsf{M}$ circuits $(\mathsf{QGarble}, \mathsf{QGEval}, \mathsf{QGSim})$. (see Section 4 of the full version)

Notation. The protocol below computes a two-party quantum functionality represented by a $\mathsf{C}+\mathsf{M}$ circuit Q that takes n_A+n_B input qubits, produces m_A+m_B output qubits, and requires n_Z auxiliary $\mathbf{0}$ states and n_T auxiliary $\mathbf{T}$ states. Let λ be the security parameter. The total number of quantum registers used will be $s = n_A + (n_B+\lambda) + (2n_Z+\lambda) + (n_T+1)\lambda$, and we'll give a name to different groups of these registers.

Given a $\mathsf{C}+\mathsf{M}$ circuit Q and a Clifford $C_{\mathsf{out}} \in \mathscr{C}_{m_A+\lambda}$, we define another $\mathsf{C}+\mathsf{M}$ circuit $Q_{\mathsf{dist}}[C_{\mathsf{out}}]$. This circuit takes as input $n_A + n_B + n_Z + \lambda + n_T\lambda$ qubits $(\mathbf{x}_A, \mathbf{x}_B, \mathbf{z}_{\mathsf{inp}}, \mathsf{trap}_A, \mathbf{t}_{\mathsf{inp}})$ on registers $(\mathsf{A}, \mathsf{B}, \mathsf{Z}_{\mathsf{inp}}, \mathsf{Trap}_{\mathsf{A}}, \mathsf{T}_{\mathsf{inp}})$. It will first apply the magic state distillation circuit from Lemma 1 with parameters $(n_T\lambda, \lambda)$ to $\mathbf{t}_{\mathsf{inp}}$ to produce QRV $\mathbf{t}$ of size n_T. It will then run Q on $(\mathbf{x}_A, \mathbf{x}_B, \mathbf{z}_{\mathsf{inp}}, \mathbf{t})$ to produce $(\mathbf{y}_A, \mathbf{y}_B)$. Finally, it will output $(C_{\mathsf{out}}(\mathbf{y}_A, \mathsf{trap}_A), \mathbf{y}_B)$.

Protocol 1: Classical Functionality $\mathcal{F}[Q]$

Common Information: Security parameter λ, and $\mathsf{C}+\mathsf{M}$ circuit Q to be computed with $n_A + n_B$ input qubits, $m_A + m_B$ output qubits, n_Z auxiliary $\mathbf{0}$ states, and n_T auxiliary $\mathbf{T}$ states. Let $s = n_A + (n_B+\lambda) + (2n_Z+\lambda) + (n_T+1)\lambda$.

Party A Input: Classical descriptions of $C_A \in \mathscr{C}_s$ and $C_{\mathsf{out}} \in \mathscr{C}_{m_A+\lambda}$.
Party B Input: Classical description of $C_B \in \mathscr{C}_{n_B+\lambda}$.

The Functionality:

1. Sample the unitary $U_{\mathsf{dec-check}}$ as follows:
   - Sample a random permutation π on $(n_T+1)\lambda$ elements.
   - Sample a random element $M \leftarrow \mathsf{GL}(2n_T, \mathbb{F}_2)$.
   - Compute a description of the Clifford U_{check} that operates as follows on registers $(\mathsf{A}, \mathsf{B}, \mathsf{Trap}_{\mathsf{B}}, \mathsf{Z}_{\mathsf{A}}, \mathsf{Trap}_{\mathsf{A}}, \mathsf{T}_{\mathsf{A}})$.
     (a) Rearrange the registers of T_{A} according to the permutation π and then partition the registers into $(\mathsf{T}_{\mathsf{inp}}, \mathsf{T}_{\mathsf{check}})$.
     (b) Apply the linear map M to the registers Z_{A} and then partition the registers into $(\mathsf{Z}_{\mathsf{inp}}, \mathsf{Z}_{\mathsf{check}})$.
     (c) Re-arrange the registers to $(\mathsf{A}, \mathsf{B}, \mathsf{Z}_{\mathsf{inp}}, \mathsf{Trap}_{\mathsf{A}}, \mathsf{T}_{\mathsf{inp}}, \mathsf{Z}_{\mathsf{check}}, \mathsf{Trap}_{\mathsf{B}}, \mathsf{T}_{\mathsf{check}})$.
   - Define $U_{\mathsf{dec-check}}$ as:
   $$U_{\mathsf{dec-check}} \coloneqq U_{\mathsf{check}} \left(\mathbb{I}^{n_A} \otimes C_B^\dagger \otimes \mathbb{I}^{(2n_Z+\lambda)+(n_T+1)\lambda}\right) C_A^\dagger.$$
2. Sample $(E_0, D_0, \widetilde{g}_1, \ldots, \widetilde{g}_d) \leftarrow \mathsf{QGarble}(1^\lambda, Q_{\mathsf{dist}}[C_{\mathsf{out}}])$.
3. Compute a description of $U_{\mathsf{dec-check-enc}} \coloneqq \left(E_0 \otimes \mathbb{I}^{(n_Z+\lambda)+\lambda}\right) U_{\mathsf{dec-check}}^\dagger$.

Party B Output: (1) A unitary $U_{\mathsf{dec-check-enc}}$ on s qubits (to be applied on registers $(\mathsf{A}, \mathsf{B}, \mathsf{Trap}_{\mathsf{B}}, \mathsf{Z}_{\mathsf{A}}, \mathsf{Trap}_{\mathsf{A}}, \mathsf{T}_{\mathsf{A}})$), and (2) A QGC $(D_0, \widetilde{g}_1, \ldots, \widetilde{g}_d)$ (to be applied to registers $(\mathsf{A}, \mathsf{B}, \mathsf{Z}_{\mathsf{inp}}, \mathsf{Trap}_{\mathsf{A}}, \mathsf{T}_{\mathsf{inp}})$).

Fig. 1. Classical functionality to be used in Protocol 2.

Protocol 2: Three-message two-party quantum computation

Common Information: (1) Security parameter λ, and (2) a $\mathsf{C}+\mathsf{M}$ circuit Q over $n_A + n_B$ input qubits, $m_A + m_B$ output qubits, n_Z auxiliary $\mathbf{0}$ states, and n_T auxiliary $\mathbf{T}$ states. Let $s = n_A + (n_B + \lambda) + (2n_Z + \lambda) + (n_T + 1)\lambda$.

Party A Input: $\mathbf{x}_A$
Party B Input: $\mathbf{x}_B$

The Protocol:
Setup. Run classical 2PC setup: $\mathsf{crs} \leftarrow \mathsf{2PC.Gen}(1^\lambda)$.

Round 1. *Party B:*

1. Sample $C_B \leftarrow \mathscr{C}_{n_B+\lambda}$ and compute $\mathbf{m}_{B,1} \coloneqq C_B(\mathbf{x}_B, \mathbf{0}^\lambda)$.
2. Compute $(m_{B,1}, \mathsf{st}) \leftarrow \mathsf{2PC}_1(1^\lambda, \mathcal{F}[Q], \mathsf{crs}, C_B)$.
3. Send to Party A: $(m_{B,1}, \mathbf{m}_{B,1})$.

Round 2. *Party A:*

1. Sample $C_A \leftarrow \mathscr{C}_s$ and $C_{\mathsf{out}} \leftarrow \mathscr{C}_{m_A+\lambda}$.
2. Compute $\mathbf{m}_{A,2} \coloneqq C_A(\mathbf{x}_A, \mathbf{m}_{B,1}, \mathbf{0}^{2n_Z}, \mathbf{0}^\lambda, \mathbf{T}^{(n_T+1)\lambda})$.
3. Compute $m_{A,2} \leftarrow \mathsf{2PC}_2(1^\lambda, \mathcal{F}[Q], \mathsf{crs}, m_{B,1}, (C_A, C_{\mathsf{out}}))$.
4. Send to Party B: $(m_{A,2}, \mathbf{m}_{A,2})$.

Round 3. *Party B:*

1. Compute $(U_{\mathsf{dec-check-enc}}, D_0, \tilde{g}_1, \ldots, \tilde{g}_d) \leftarrow \mathsf{2PC}_{\mathsf{out}}(1^\lambda, \mathsf{st}, m_{A,2})$.
2. Compute $(\mathbf{m}_{\mathsf{inp}}, \mathbf{z}_{\mathsf{check}}, \mathsf{trap}_B, \mathbf{t}_{\mathsf{check}}) \coloneqq U_{\mathsf{dec-check-enc}}(\mathbf{m}_2)$, where
   - $\mathbf{m}_{\mathsf{inp}}$ is on registers $(\mathsf{A}, \mathsf{B}, \mathsf{Z}_{\mathsf{inp}}, \mathsf{Trap}_{\mathsf{A}}, \mathsf{T}_{\mathsf{inp}})$,
   - $(\mathbf{z}_{\mathsf{check}}, \mathsf{trap}_B, \mathbf{t}_{\mathsf{check}})$ is on registers $(\mathsf{Z}_{\mathsf{check}}, \mathsf{Trap}_{\mathsf{B}}, \mathsf{T}_{\mathsf{check}})$.
3. Measure each qubit of $(\mathbf{z}_{\mathsf{check}}, \mathsf{trap}_B)$ in the standard basis and abort if any measurement is not zero.
4. Measure each qubit of $\mathbf{t}_{\mathsf{check}}$ in the T-basis and abort if any measurement is not zero.
5. Compute $(\widehat{\mathbf{y}}_A, \mathbf{y}_B) \leftarrow \mathsf{QGEval}((D_0, \tilde{g}_1, \ldots, \tilde{g}_d), \mathbf{m}_{\mathsf{inp}})$, where $\widehat{\mathbf{y}}_A$ consists of $m_A + \lambda$ qubits and $\mathbf{y}_B$ consists of m_B qubits.
6. Send to Party A: $\widehat{\mathbf{y}}_A$.

Output Reconstruction.

- *Party A:* Compute $(\mathbf{y}_A, \mathsf{trap}_A) \coloneqq C^\dagger_{\mathsf{out}}(\widehat{\mathbf{y}}_A)$, where $\mathbf{y}_A$ consists of m_A qubits and trap_A consists of λ qubits. Measure each qubit of trap_A in the standard basis and abort if any measurement is not zero. Otherwise, output $\mathbf{y}_A$.
- *Party B:* Output $\mathbf{y}_B$.

Fig. 2. Three-message two-party quantum computation.

3.3 Security

Theorem 5. *Assuming post-quantum maliciously-secure two-message oblivious transfer, there exists maliciously-secure NISC for quantum computation and maliciously-secure three-message two-party quantum computation.*

Proof. Let Π be the protocol described in Protocol 2 computing some quantum circuit Q. Here, we only show security against a malicious party A and defer the remainder of the proof to the full version.

The simulator. Consider any QPT adversary $\mathsf{Adv} = \{\mathsf{Adv}_\lambda\}_{\lambda\in\mathbb{N}}$ corrupting party A. The simulator Sim is defined as follows. Whenever we say that the simulator aborts, we mean that it sends $\bot$ to the ideal functionality and to the adversary.

$\mathsf{Sim}^{\mathcal{I}[\mathbf{x}_B](\cdot)}(\mathbf{x}_A, \mathbf{aux}_{\mathsf{Adv}})$*:*

- Compute $(\mathsf{crs}, \tau, m_{B,1}) \leftarrow \mathsf{2PC.Sim}_A^{(1)}(1^\lambda)$, sample $C_B \leftarrow \mathscr{C}_{n_B+\lambda}$, compute $\mathbf{m}_{B,1} \coloneqq C_B(\mathbf{0}^{n_B}, \mathbf{0}^\lambda)$, and send $(\mathsf{crs}, m_{B,1}, \mathbf{m}_{B,1})$ to $\mathsf{Adv}_\lambda(\mathbf{x}_A, \mathbf{aux}_{\mathsf{Adv}})$.
- Receive $(m_{A,2}, \mathbf{m}_{A,2})$ from Adv_λ and compute $\mathsf{out} \leftarrow \mathsf{2PC.Sim}_A^{(1)}(1^\lambda, \tau, m_{A,2})$. If $\mathsf{out} = \bot$ then abort. Otherwise, parse out as (C_A, C_{out}).
- Using (C_A, C_B), sample $U_{\mathsf{dec-check}}$ as in the description of $\mathcal{F}[Q]$. Compute

$$(\mathbf{x}'_A, \mathbf{x}'_B, \mathbf{z}_{\mathsf{inp}}, \mathsf{trap}_A, \mathbf{t}_{\mathsf{inp}}, \mathbf{z}_{\mathsf{check}}, \mathsf{trap}_B, \mathbf{t}_{\mathsf{check}}) \coloneqq U_{\mathsf{dec-check}}(\mathbf{m}_{A,2}).$$

  Measure each qubit of $\mathbf{z}_{\mathsf{check}}$ and trap_B in the standard basis and each qubit of $\mathbf{t}_{\mathsf{check}}$ in the T-basis. If any measurement is non-zero, then abort.
- Forward $\mathbf{x}'_A$ to $\mathcal{I}[\mathbf{x}_B](\cdot)$ and receive back $\mathbf{y}_A$. Compute $\widehat{\mathbf{y}}_A \coloneqq C_{\mathsf{out}}(\mathbf{y}_A, \mathsf{trap}_A)$, send $\widehat{\mathbf{y}}_A$ to Adv_λ, send ok to $\mathcal{I}[\mathbf{x}_B]$, and output the output of Adv_λ.

We consider a sequence of hybrid distributions, where the first hybrid $\mathcal{H}_0$ is $\mathsf{REAL}_{\Pi,\mathsf{Q}}(\mathsf{Adv}_\lambda, \mathbf{x}_A, \mathbf{x}_B, \mathbf{aux}_{\mathsf{Adv}})$, i.e. the real interaction between the adversary $\mathsf{Adv}_\lambda(\mathbf{x}_A, \mathbf{aux}_{\mathsf{Adv}})$ and an honest party $B(1^\lambda, \mathbf{x}_B)$. In each hybrid, we describe the differences from the previous hybrid.

- $\mathcal{H}_1$: Simulate $\mathsf{2PC}$ as described in Sim, using $\mathsf{2PC.Sim}_A^{(1)}$ to compute $m_{B,1}$ and $\mathsf{2PC.Sim}_A^{(2)}$ to extract an input (C_A, C_{out}) (or abort). Use (C_A, C_{out}) to sample an output $(U_{\mathsf{dec-check-enc}}, D_0, \widetilde{g}_1, \ldots, \widetilde{g}_d)$ of the classical functionality. Use this output to run party B's honest Message 3 algorithm.
- $\mathcal{H}_2$: In this hybrid, we change how B's third round message $\widehat{\mathbf{y}}_A$ is sampled. In particular, rather than evaluating the quantum garbled circuit on $\mathbf{m}_{\mathsf{inp}}$, we will directly evaluate $Q_{\mathsf{dist}}[C_{\mathsf{out}}]$ on the input. In more detail, given $\mathbf{m}_{A,2}$ returned by Adv_λ, (C_A, C_{out}) extracted from Adv_λ, and C_B sampled in Message 1, $\widehat{\mathbf{y}}_A$ is sampled as follows. Sample $U_{\mathsf{dec-check}}$ as in Step 1 of $\mathcal{F}[Q]$. Compute

$$(\mathbf{x}'_A, \mathbf{x}'_B, \mathbf{z}_{\mathsf{inp}}, \mathsf{trap}_A, \mathbf{t}_{\mathsf{inp}}, \mathbf{z}_{\mathsf{check}}, \mathsf{trap}_B, \mathbf{t}_{\mathsf{check}}) \coloneqq U_{\mathsf{dec-check}}(\mathbf{m}_{A,2})$$

and carry out the checks on $\mathbf{z}_{\mathsf{check}}, \mathsf{trap}_B, \mathbf{t}_{\mathsf{check}}$ as described in Steps 3.(c) and 3.(d) of Protocol 2, aborting if needed. Then, compute

$$(\widehat{\mathbf{y}}_A, \mathbf{y}_B) \leftarrow Q_{\mathsf{dist}}[C_{\mathsf{out}}](\mathbf{x}'_A, \mathbf{x}'_B, \mathbf{z}_{\mathsf{inp}}, \mathsf{trap}_A, \mathbf{t}_{\mathsf{inp}})$$

and return $\widehat{\mathbf{y}}_A$ to Adv_λ.

- $\mathcal{H}_3$: Compute $\mathbf{m}_{B,1}$ as $C_B(\mathbf{0}^{n_B}, \mathbf{0}^\lambda)$, and substitute $\mathbf{x}_B$ for $\mathbf{x}'_B$ before applying $Q_{\mathsf{dist}}[C_{\mathsf{out}}]$ to the registers described above in $\mathcal{H}_2$.
- $\mathcal{H}_4$: Rather than directly computing $Q_{\mathsf{dist}}[C_{\mathsf{out}}]$, query the ideal functionality with $\mathbf{x}'_A$, receive $\mathbf{y}_A$, and send $\widehat{\mathbf{y}}_A := C_{\mathsf{out}}(\mathbf{y}_A, \mathsf{trap}_A)$ to Adv_λ. This hybrid is $\mathsf{IDEAL}_{\Pi,\mathsf{Q},A}(\mathsf{Sim}, \boldsymbol{\rho}_\lambda, \mathbf{x}_A, \mathbf{x}_B, \mathbf{aux})$.

We show indistinguishability between each pair of hybrids.

- $\mathcal{H}_0 \approx_c \mathcal{H}_1$: This follows from the security against corrupted A of 2PC.
- $\mathcal{H}_1 \approx_s \mathcal{H}_2$: This follows from the statistical correctness of QGC.
- $\mathcal{H}_2 \approx_s \mathcal{H}_3$: First, by the security of the Clifford authentication code, conditioned on all measurements of qubits in trap_B returning 0, we have that $\mathbf{x}'_B \approx_s \mathbf{x}_B$. Next, switching $\mathbf{x}_B$ to $\mathbf{0}^{n_B}$ in B's first message is perfectly indistinguishable due to the perfect hiding of the Clifford authentication code.
- $\mathcal{H}_3 \approx_s \mathcal{H}_4$: First, by Lemma 2, conditioned on all measurements of qubits in $\mathbf{z}_{\mathsf{check}}$ returning 0, we have that $\mathbf{z}_{\mathsf{inp}} \approx_s \mathbf{0}^{n_Z}$.
  Next, the above observation, along with Lemma 1, implies that, conditioned on all T-basis measurements of qubits in $\mathbf{t}_{\mathsf{check}}$ returning 0, it holds that the output of $Q_{\mathsf{dist}}[C_{\mathsf{out}}](\mathbf{x}'_A, \mathbf{x}_B, \mathbf{z}_{\mathsf{inp}}, \mathsf{trap}_A, \mathbf{t}_{\mathsf{inp}})$ is statistically close to the result of computing $(\mathbf{y}_A, \mathbf{y}_B) \leftarrow Q(\mathbf{x}'_A, \mathbf{x}_B, \mathbf{0}^{n_Z}, \mathbf{T}^{n_T})$ and returning $(C_{\mathsf{out}}(\mathbf{y}_A, \mathsf{trap}_A), \mathbf{y}_B)$. This is precisely what is being computed in $\mathcal{H}_4$.

On Reusable Security against Malicious A. We remark that the two-message special case of the above protocol, that is, our Quantum NISC protocol, can be lightly modified to also achieve *reusable* security. A reusable classical NISC protocol (see, eg. [17]) retains security against malicious A in a setting where A and B execute many instances of secure computation that *reuse* the first message of B. A natural quantum analogue of this protocol enables computation of quantum circuits while guaranteeing security against malicious A, in a setting where A and B execute many instances of secure computation that *reuse* the first message of B. Here we assume that B's input is classical, and so functionality will hold over repeated executions. We note that our protocol can be lightly modified to achieve reusable security against malicious A, by replacing the underlying classical 2PC with a reusable classical 2PC. The proof of security remains identical, except that the indistinguishability between hybrids 0 and 1 relies on the reusable security of the underlying classical two-party computation protocol. In the full version, we discuss how to achieve reusable MDV-NIZKs for NP, which can be viewed as a special case of reusable Q-NISC.

References

1. Agarwal, A., Bartusek, J., Goyal, V., Khurana, D., Malavolta, G.: Post-quantum multi-party computation in constant rounds. To Appear in Eurocrypt (2020)
2. Alagic, G., Brakerski, Z., Dulek, Y., Schaffner, C.: Impossibility of quantum virtual black-box obfuscation of classical circuits. arXiv preprint arXiv:2005.06432 (2020)
3. Alagic, G., Fefferman, B.: On quantum obfuscation. ArXiv **abs/1602.01771** (2016)
4. Alon, B., Chung, H., Chung, K.M., Huang, M.Y., Lee, Y., Shen, Y.C.: Round efficient secure multiparty quantum computation with identifiable abort. Cryptology ePrint Archive, Report 2020/1464 (2020), https://eprint.iacr.org/2020/1464
5. Ananth, P., Choudhuri, A.R., Jain, A.: A new approach to round-optimal secure multiparty computation. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 468–499. Springer, Heidelberg (Aug 2017)
6. Ananth, P., La Placa, R.L.: Secure software leasing. arXiv preprint arXiv:2005.05289 (2020)
7. Badrinarayanan, S., Goyal, V., Jain, A., Kalai, Y.T., Khurana, D., Sahai, A.: Promise zero knowledge and its applications to round optimal MPC. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 459–487. Springer, Heidelberg (Aug 2018)
8. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (Aug 2001)
9. Beaver, D., Micali, S., Rogaway, P.: The round complexity of secure protocols (extended abstract). In: 22nd ACM STOC. pp. 503–513. ACM Press (May 1990)
10. Beigi, S., Koenig, R.: Simplified instantaneous non-local quantum computation with applications to position-based cryptography. New Journal of Physics **13**(9), 093036 (sep 2011)
11. Ben-Or, M., Crépeau, C., Gottesman, D., Hassidim, A., Smith, A.: Secure multiparty quantum computation with (only) a strict honest majority. In: 47th FOCS. pp. 249–260. IEEE Computer Society Press (Oct 2006)
12. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: 20th ACM STOC. pp. 1–10. ACM Press (May 1988)
13. Brakerski, Z.: Quantum FHE (almost) as secure as classical. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 67–95. Springer, Heidelberg (Aug 2018)
14. Brakerski, Z., Halevi, S., Polychroniadou, A.: Four round secure computation without setup. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 645–677. Springer, Heidelberg (Nov 2017)
15. Brakerski, Z., Yuen, H.: Quantum garbled circuits. arXiv preprint arXiv:2006.01085 (2020)
16. Bravyi, S., Kitaev, A.: Universal quantum computation with ideal clifford gates and noisy ancillas. Physical Review A **71**(2), 022316 (2005)
17. Chase, M., Dodis, Y., Ishai, Y., Kraschewski, D., Liu, T., Ostrovsky, R., Vaikuntanathan, V.: Reusable non-interactive secure computation. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 462–488. Springer, Heidelberg (Aug 2019)
18. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols (abstract) (informal contribution). In: Pomerance, C. (ed.) CRYPTO'87. LNCS, vol. 293, p. 462. Springer, Heidelberg (Aug 1988)

19. Choudhuri, A.R., Ciampi, M., Goyal, V., Jain, A., Ostrovsky, R.: Round optimal secure multiparty computation from minimal assumptions. In: Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II. pp. 291–319 (2020)
20. Crépeau, C., Gottesman, D., Smith, A.: Secure multi-party quantum computation. In: 34th ACM STOC. pp. 643–652. ACM Press (May 2002)
21. D'Ariano, G.M., Schlingemann, D., Werner, R., Kretschmann, D.: Quantum bit commitment revisited: the possible and the impossible. Tech. rep. (2006)
22. Dulek, Y., Grilo, A.B., Jeffery, S., Majenz, C., Schaffner, C.: Secure multi-party quantum computation with a dishonest majority. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 729–758. Springer, Heidelberg (May 2020)
23. Dupuis, F., Nielsen, J.B., Salvail, L.: Secure two-party quantum evaluation of unitaries against specious adversaries. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 685–706. Springer, Heidelberg (Aug 2010)
24. Dupuis, F., Nielsen, J.B., Salvail, L.: Actively secure two-party evaluation of any quantum operation. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology - CRYPTO 2012. Proceedings. LNCS, vol. 7417, pp. 794–811. Springer (2012)
25. Garg, S., Mukherjee, P., Pandey, O., Polychroniadou, A.: The exact round complexity of secure computation. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 448–476. Springer, Heidelberg (May 2016)
26. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 468–499. Springer, Heidelberg (Apr / May 2018)
27. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th ACM STOC. pp. 218–229. ACM Press (May 1987)
28. Gonzales, A., Chitambar, E.: Bounds on instantaneous nonlocal quantum computation. IEEE Trans. Inf. Theory **66**(5), 2951–2963 (2020)
29. Goyal, R.: Quantum multi-key homomorphic encryption for polynomial-sized circuits. Cryptology ePrint Archive, Report 2018/443 (2018), https://eprint.iacr.org/2018/443
30. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Prabhakaran, M., Sahai, A.: Efficient non-interactive secure computation. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 406–425. Springer, Heidelberg (May 2011)
31. Katz, J., Ostrovsky, R.: Round-optimal secure two-party computation. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 335–354. Springer, Heidelberg (Aug 2004)
32. Lo, H.K., Chau, H.F.: Why quantum bit commitment and ideal quantum coin tossing are impossible. Physica D: Nonlinear Phenomena **120**(1-2), 177–187 (1998)
33. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Karloff, H.J., Pitassi, T. (eds.) 44th ACM STOC. pp. 1219–1234. ACM Press (May 2012)
34. Mahadev, U.: Classical homomorphic encryption for quantum circuits. In: Thorup, M. (ed.) 59th FOCS. pp. 332–338. IEEE Computer Society Press (Oct 2018)
35. Mayers, D.: Unconditionally secure quantum bit commitment is impossible. Physical review letters **78**(17), 3414 (1997)
36. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (Aug 2008)

37. Shmueli, O.: Multi-theorem (malicious) designated-verifier nizk for qma (2020)
38. Speelman, F.: Instantaneous non-local computation of low t-depth quantum circuits. In: Broadbent, A. (ed.) 11th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2016, September 27-29, 2016, Berlin, Germany. LIPIcs, vol. 61, pp. 9:1–9:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2016)
39. Vaidman, L.: Instantaneous measurement of nonlocal variables. Phys. Rev. Lett. **90**, 010402 (Jan 2003)
40. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: 27th FOCS. pp. 162–167. IEEE Computer Society Press (Oct 1986)