Asymmetric Quantum Secure Multi-Party Computation With Weak Clients Against Dishonest Majority

Theodoros Kapourniotis[1], Elham Kashefi[2,3], Dominik Leichtle[3], Luka Music[4], and Harold Ollivier[5]

[1] Department of Physics, University of Warwick, Coventry CV4 7AL, United Kingdom
[2] School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB, United Kingdom
[3] Laboratoire d'Informatique de Paris 6, CNRS, Sorbonne Université, 4 Place Jussieu, 75005 Paris, France
[4] Quandela, 7 Rue Léonard de Vinci, 91300 Massy, France
[5] DI-ENS, Ecole Normale Supérieure, PSL University, CNRS, INRIA, 45 rue d'Ulm, 75005 Paris, France

Abstract. Secure multi-party computation (SMPC) protocols allow several parties that distrust each other to collectively compute a function on their inputs. In this paper, we introduce a protocol that lifts classical SMPC to quantum SMPC in a composably and statistically secure way, even for a single honest party. Unlike previous quantum SMPC protocols, our proposal only requires very limited quantum resources from all but one party; it suffices that the weak parties, i.e. the *clients*, are able to prepare single-qubit states in the $X - Y$ plane.
The novel quantum SMPC protocol is constructed in a naturally modular way, and relies on a new technique for quantum verification that is of independent interest. This verification technique requires the remote preparation of states only in a single plane of the Bloch sphere. In the course of proving the security of the new verification protocol, we also uncover a fundamental invariance that is inherent to measurement-based quantum computing.

Keywords: Quantum Verification, Delegated Computation, Secure Multi-Party Computation, Distributed Quantum Computing.

1 Introduction

1.1 Motivation

Secure Multi-Party Computation (SMPC) protocols allow several parties who do not trust one another to collectively compute a function on their inputs. This question was first considered by Yao [36] and has been developed extensively in various settings (see [6] and references therein). Several security guarantees can be provided by such protocols depending on the setting: all parties can be on an equal footing, doing each their share of the computation, or one can handle the brunt of the computation while all others provide the data. In the first case, the security goal is to maximise the privacy of the data, while in the latter it extends to the privacy of the computation which is delegated to the server.

Practical computationally-secure protocols have been developed and implemented in commercial solutions for protecting classical multi-party computations. In the quantum case, several concrete protocols have been proposed (see § 1.2). In the circuit model, the state-of-the-art protocol [9] provides an information theoretic upgrade of classical SMPC that can withstand a dishonest majority. In the measurement-based model, where weakly quantum clients delegate their computation to a powerful server, the best protocol [25] does not provide verification of the computation and settles instead for blindness (i.e. privacy) of the data when there is no client-server collusion.

In this work, we show that this difference is not due to the asymmetry of the clients-server setting. We introduce for this specific situation a statistically secure lift of a classical SMPC protocol to a quantum one that provides blindness and verification for BQP computations. It remains secure so long as a single client is honest, thus withstanding possible collusions between dishonest clients and the server. Building on the techniques introduced in [22], its security is proved in the Abstract Cryptography (AC) framework. The protocol is highly modular and can tolerate a fixed amount of global noise during the quantum computation

without aborting nor compromising statistical security. Additionally, it has no space overhead compared to an unprotected delegated computation, thereby allowing clients to use the server's full power to perform their desired computation, while security comes only at the price of a polynomial number of repetitions.

1.2 Related Work

Quantum SMPC is a long standing research topic in quantum cryptography, with several directions being explored in the past two decades.

The first one started with [7]. Along with the introduction of the concept itself, it provided a concrete protocol for performing such computations in the quantum circuit model. It guarantees the security of the computation as long as the fraction of malicious parties does not exceed 1/6. This work has been later extended in [4], lowering the minimum number of honest players required for security to a strict majority.

The second focuses on the interesting edge case of two-party quantum computations. Several constructive results have been proposed in the circuit model. In [12], a protocol was introduced and proven secure for quantum honest-but-curious adversaries. This restriction on the adversaries was removed in [13] which proved security in the fully malicious setting and with negligible security bounds. The measurement-based model of quantum computation has also been considered for constructing secure two-party quantum computations as it provides a different set of tools than the circuit model. Verifiable Blind Quantum Computation (VBQC) first was introduced in [16] in this model, followed by optimised protocols [27,21]. In [26] a protocol was proposed in this setting and proven secure against honest-but-curious adversaries. In [24] this result was extended to fully malicious adversaries with inverse-polynomial security using the Quantum Cut-and-Choose technique. More recently, a round-optimal protocol was given in [3] based on Oblivious Transfer and LWE, showing that two-party quantum computation tasks can be performed in as little as three rounds in the CRS model, and two if quantum pre-processing is allowed.

A third set of results focuses on the composability of such protocols, as earlier results didn't satisfy this property. Bit commitment was shown to be complete in the Quantum Universal Composability framework of [35], meaning that it is sufficient for constructing quantum or classical SMPC if parties have access to quantum channels and operations. This result was later extended in [14,11], which gives a full analysis of feasibility and completeness of cryptographic primitives in a composable setting.

More recently, building on these previous works, new concrete protocols have been proposed to decrease the restrictions on adversaries while also providing composable security. In the circuit model, a composably-secure protocol has been introduced in [9]. It is an extension of [13] that is able to cope with a dishonest majority, but which relies on a complete graph for quantum communication and on a large number of quantum communication rounds together with powerful quantum participants. In the MBQC model, [25] describes a protocol that is composable, can tolerate a dishonest majority and allows the clients to delegate the quantum computation to a powerful server. Its security is an information-theoretic upgrade of the classical SMPC primitive used for constructing the protocol. It is however limited by the absence of verifiability of outputs and the impossibility to tolerate client-server collusions. Other protocols have been proposed in alternative models or with different trust assumptions such as [20,29]. Finally, recent protocols for secure delegated quantum computations can be run even by purely classical clients. These have been lifted to a multi-client setting in [2] while at the same time optimising the number of classical rounds of communication. This is however at the cost of a larger computation space on the server's device, which needs to be able to perform QFHE computations of functions large enough to be computationally-secure.

A subset of the authors proposed an earlier protocol for QSMPC [23] which comprised a blind pre-computation step meant to produce a resource state that could then be used to perform VBQC. This pre-computation turned out to be vulnerable to an attack that can be applied blindly by the server while having an effect only on some specific types of qubits thereby compromising security of the whole protocol. While the present work is a complete redesign of the protocol that shows improved performance, we include in § C an in-depth analysis of the shortcommings of the previous construction. This might be a useful tool to revisit earlier work where a similar blind pre-computation step is used.

1.3 Overview of the Protocol and Results

In this paper, we consider the setting where several weakly quantum clients want to securely delegate their quantum computation to a powerful server. The proposed construction turns a single-client MBQC-based protocol into a multi-party one. More precisely, we use single-client Secure Delegated Quantum Computing (SDQC) protocols obtained through the techniques presented in [22]. Such protocols interleave several computation rounds and test rounds, the latter of which correspond to stabiliser measurements of the MBQC resource graph-state used to perform the computations. In such a protocol, the client must perform two different tasks. First, it has to prepare encrypted single-qubit states and send them to the server. This prevents the server from distinguishing computation and test rounds and also hides the client's data. Then, the client uses the classical encryption key as well as the measurement outcomes reported by the server to classically drive the computations and tests performed by the server on these encrypted qubits. Hence, turning this protocol into a multi-party one amounts to finding (i) an appropriate single-client SDQC protocol that will (ii) be composed with a secure collaborative remote state preparation for the single qubit encrypted states and that will (iii) be driven collaboratively to perform and verify the desired computation.

In § 2, we describe a single-client SDQC Protocol using only $|+_\theta\rangle = (|0\rangle + e^{i\theta}|1\rangle)/\sqrt{2}$ states, based on the generic single-client SDQC Protocol of [22]. This was an open question in the field as all previous SDQC protocols in the MBQC framework with a formal security analysis use computational basis states (called dummies) to isolate single qubits in the computation graph. These remain unchanged if the server is honest and can be used as traps to detect deviations. To overcome this restriction, we must ideally find a generating set of stabilisers of the graph state for the client's computation that can be written with I, X and Y Paulis only.

However, while it is possible to construct $N-1$ independent stabilisers of this form – where N denotes the number of vertices of the graph – it seems that the stabiliser which consists of Z operators on odd-degree vertices of the graph cannot be generated. This therefore corresponds to a server's deviation which cannot be caught by our tests on graphs containing odd degree nodes. If this attack would corrupt the client's computation, the whole protocol would be insecure. Fortunately, this is not the case for classical input/output computations. Indeed, we prove that this deviation corresponds to a server which has chosen a different orientation of the Z axis compared to the client. Because inputs are prepared in the $X-Y$ plane and outputs are projected onto it, we show that this has no effect on the outcome of the computation. As a consequence, it is not necessary to detect this specific deviation by the server to verify the computation. This proves that the generic single-client SDQC Protocol of [22] can be used to produce secure dummyless protocols.[6]

Theorem (Informal)**.** *For any graph G, there exists a single-client statistically secure SDQC protocol in the Abstract Cryptography framework that requires the client to only prepare states in the $X-Y$ plane.*

We then focus on turning this new single-client protocol into a multi-party one. In § 3, we introduce a Collaborative Remote State Preparation (CRSP) protocol. We show that this gadget (Protocol 2) securely implements Remote State Preparation (Resource 2), which allows a classical party request any $|+_\theta\rangle$ state to be prepared on the server's device with the help of clients preparing single qubit states in the $X-Y$ plane.

Theorem (Informal)**.** *The CRSP gadget is a statistically secure implementation of the Remote State Preparation Resource in the Abstract Cryptography framework.*

The second set of tasks in the single-client protocol, i.e. choosing the measurement angles of the various computation and test rounds according to the states prepared using CRSP, only involve classical computations. These can be performed using a composably secure classical SMPC.[7]

[6] Note that here has been a previous protocol for dummyless verification [15], whose security analysis didn't take into account the above deviation. Our proof of invariance of MBQC to this specific error shows that this deviation does not constitute a security threat to the protocol in [15].

[7] The Abstract Cryptography framework used in this work is equivalent to the Quantum Universal Composability (Q-UC) Model of [35] if a single Adversary controls all corrupted parties – which is the case here. Therefore any Classical SMPC protocol which is secure in the Q-UC model can be used to instantiate this functionality.

In § 4, we compose the CRSP gadget, classical SMPC, and the dummyless SDQC protocol into a complete quantum SMPC protocol (Protocol 3). Its outline is:

1. The clients use the CRSP gadget to prepare $|+_\theta\rangle$ states on the server's side.
2. They use the classical SMPC together to drive and verify the single-client SDQC protocol.
3. Upon acceptance, the results and decryption keys are sent by the classical SMPC to each client.

The security proof relies on the composable security of all three ingredients. Because the CRSP gadget and the dummyless protocol are statistically secure, this is a direct upgrade of classical to quantum SMPC.

Theorem (Informal)**.** *Composable classical SMPC can be lifted to perform robust quantum SMPC for* BQP *computations in a statistically secure way, such that all parties but one are restricted to singe-qubit preparations.*

We note that this protocol requires no additional resources in terms of hardware or quantum communication from the client's side compared to the single-client protocol. The server only needs to be able to perform the CRSP gadget in addition to the operations required by the single-client protocol.

1.4 Discussion

In the course of constructing our protocol, we have built two new ingredients that we believe are of independent interest.

The first one is the Collaborative Remote State Preparation gadget. Its main feature is to provide some privacy amplification for the classical-quantum correlations that clients share with the server. Interestingly, we give evidence that it is hard to construct a generic gadget that would have similar features for correlations outside of a single plane of the Bloch sphere, while retaining its usefulness for cryptographic purposes. We leave it as an open question to prove a full no-go theorem in the Abstract Cryptography framework to further explore what seems to be a deep difference between classical and quantum input-output computations. Note also that this work supersedes a previous effort to construct a quantum SMPC protocol in the clients-server setting with quantum input and outputs. The proposed construction was similar in spirit with a collaborative remote state preparation gadget that allowed to prepare encrypted $X - Y$ plane states but also dummies. However, we give an attack on multiple approaches which were explored to perform this task, further strengthening the belief that such cryptographic protocols are hard if not impossible to construct.

The second new ingredient of our proof is the first dummyless SDQC protocol. Outside of the specific purpose of quantum SMPC, it exemplifies the usefulness of the general tests that were introduced in [22]. By reducing the requirements on the client side, it also possibly decreases a source of errors in physical implementations as it is not uncommon that rotations around one specific axis of the Bloch sphere are notably easier to perform than others. We also strongly believe that similar approaches, where traps are tailored to specific settings, will find applications in the future. Additionally, we show that while dummyless tests were not enough to detect all deviations, it is possible to nonetheless verify computations thanks to an as of now unknown invariance in MBQC. This raises the question of whether it is possible to do this on purpose, and engineer an invariance in order to lighten the constraints on the error-detection scheme that the traps implement.

Finally, note that because all SDQC protocols constructed from the generic protocol of [22] are robust to a fixed amount of global noise, so is our new multi-party protocol. While not being enough to scale to large quantum computations, it opens the possibility to implement experimental proof-of-concepts without resorting to error correction on near term devices.

1.5 Organisation of the Paper

In § 2 we construct a single-client SDQC Protocol using only $|+_\theta\rangle = (|0\rangle + e^{i\theta}|1\rangle)/\sqrt{2}$ states. §§ 2.1-2.3 construct a family of such schemes and prove their security, while § 2.4 provides optimised protocols for various classes of MBQC resource graph-states. In § 3, we introduce a Collaborative Remote State Preparation (CRSP) protocol and prove its security in the AC framework. In § 4, we compose the CRSP

Protocol, the dummyless SDQC Protocol and a classical SMPC into a complete quantum SMPC Protocol (Protocol 3) for BQP computations. In § 5, we provide an in-depth comparison with other protocols, give arguments justifying the proposed construction – especially the need for a dummyless SDQC Protocol – and discuss some open questions.

Some preliminary notation and material can be found in the corresponding sections of the Auxiliary Supporting Material part: Abstract Cryptography in § A, MBQC computation in § B and the Universal Blind Quantum Computation (UBQC) Protocol in § B.1. A detailed analysis of a previous attempt at constructing quantum SMPC for weakly quantum clients is provided in § C.

2 Verification with States in a Single Plane

2.1 A Framework for Verification

The goal of the protocol presented in this section is to construct the Secure Delegated Quantum Computation Resource 1 (SDQC), introduced by [10]. It allows a single Client to run a quantum computation on a Server so that the Server cannot corrupt the computation and doesn't learn anything besides a controlled leakage l_ρ about the Client's computation and input. The value of l_ρ, as a function of inputs and computation, is specified by each protocol.

Resource 1 Secure Delegated Quantum Computation with Classical Inputs and Outputs

Inputs:
- The Client inputs a bit-string x and the classical description of a unitary U.
- The Server chooses whether or not to deviate. This interface is filtered by two control bits (e, c).

Computation by the Resource:
1. If $e = 1$, the Resource sends the leakage l_ρ to the Server's interface and awaits further input from the Server; if it receives $c = 1$, the Resource outputs Abort at the Client's output interface.
2. If $c = 0$, it outputs $O = \mathcal{M}_C \circ \mathsf{U}\,|x\rangle$ at the Client's output interface, where $\mathcal{M}_C$ is a computational basis measurement.

Several protocols implementing this resource have been constructed in the past [17]. Yet, none has the ability to provide negligible statistical security while having a client sending *states in a single plane*. To achieve this, we use the framework from [22] which neatly separates the various ingredients required to implement SDQC. We start by briefly summarising the ingredients which are relevant to the present paper.

Reduction to Pauli Deviations. Using the UBQC Protocol 4 (see appendix) to delegate computations from Client to Server hides the operations which the Client wishes to delegate. The encoding scheme of UBQC naturally imposes a Pauli twirl on any deviation and hence any attacks by the Server can always be decomposed as a convex combination of Pauli operators acting on the qubits of the graph just before performing the X-basis measurement. Because X Pauli operators applied in this fashion have no effect on the computation, as they are absorbed by the X-basis measurement, we can focus on convex combinations of deviations of the form $\bigotimes_{v\in V} \mathsf{Z}(v)^{e(v)}$ where the values of $e(v)$ are chosen by the Server and $\mathsf{Z}(v)$ applies the Pauli Z to qubit v. Such deviation are equivalent to flipping the measurement outcome for vertices where $e(v) = 1$.

General Strategy for Robust Verification. Once all operations delegated to the Server are blind a general strategy for robust and secure computation follows from the intuition that (i) correctness is obtained by accepting with overwhelming probability in the absence of deviation, (ii) security derives from the ability of the protocol to detect with overwhelming probability all deviations that potentially affect the computation, and (iii) robustness follows from accepting additional deviations which have, with overwhelming probability, no effect on the computation.

Generic Trappified Schemes for Classical I/O. With this strategy in mind, a whole class of protocols for verifying BQP computations can be easily described. Their flexible design is able to accommodate objectives that go beyond security, e.g. for instance the absence of dummy qubits. These protocols work by performing separate rounds which are indistinguishable from the Server's point of view, some implementing tests, and others computing C, the Client's target computation. More precisely, s test rounds and d computation rounds are delegated to the Server using the UBQC Protocol 4, with the requirement that they share the same graph G and the same order $\preceq_G$ for measuring the qubits.

Each test round is sampled uniformly at random from a set $\boldsymbol{P}$ of possible traps called a *trappified scheme*. They each consist of an input state σ which is a tensor product of single-qubit states, one for each vertex in the graph G, a measurement pattern T, and a binary decision function τ. The test round is accepted when the decision function outputs 0 when evaluated on the measurement results returned by the Server for this trap. It is rejected when the output is 1. The d computation rounds correspond to repeating d times the target computation C on the target input chosen by the Client using the graph G. The outputs of these computations are then combined through a majority vote. When all rounds have been executed, the Client accepts if less than a fixed fraction of test rounds reject. In this case, the output of the protocol is the result of the majority vote. The formal protocol is described in Protocol 1.

Protocol 1 Trappified Delegated Blind Computation

Public Information:
- $G = (V, E, I, O)$, a graph with input and output vertices I and O respectively;
- $\boldsymbol{P}$, a trappified scheme on graph G;
- $\preceq_G$, a partial order on the set V of vertices;
- N, d, w, parameters representing the number of runs, the number of computation runs, and the number of tolerated failed tests.

Client's Inputs: A set of angles $\{\phi_i\}_{i\in V}$ and a flow f which induces an ordering compatible with $\preceq_G$.

Protocol:
1. The Client samples uniformly at random a subset $C \subset [N]$ of size d representing the runs which will be its desired computation, henceforth called computation runs.
2. For $k \in [N]$, the Client and Server perform the following:
   (a) If $k \in C$, the Client sets the computation for the run to its desired computation $(\{\phi_i\}_{i\in V}, f)$. Otherwise, the Client samples a test (T, σ, τ) from the trappified scheme $\boldsymbol{P}$.
   (b) The Client and Server blindly execute the run using the UBQC Protocol 4.
   (c) If it is a test, it uses τ on the measurement results to decide whether the test passed or not.
3. At the end of all runs, let x be the number of failed tests. If $x \geq w$, the Client rejects and outputs $(\perp, \mathsf{Rej})$.
4. Otherwise, the Client accepts the computation. It performs a majority vote on the output results of the computation runs and sets the result as its output.

Security Conditions for Trappified Schemes with Classical I/O. The analysis of the security and robustness properties in the Abstract Cryptography framework for the resulting protocol depends on two sets of Pauli operators defined relatively to $\boldsymbol{P}$: the set of detectable deviations and the set of deviations to which $\boldsymbol{P}$ is insensitive. These rely on the following definitions, where we use $\mathcal{T}$ to denote the probability of the measurement outcomes for a trap T in $\boldsymbol{P}$ and $\mathsf{E} \circ \mathcal{T}$ to denote the probability distribution of measurement outcomes when the deviation E is applied to T.

Definition 1 (Pauli Insensitivity). *We say that the trappified scheme $\boldsymbol{P}$ is* δ-insensitive to $\mathcal{E} \subset \mathcal{G}_V$ *if:*

$$\forall \mathsf{E} \in \mathcal{E}, \sum_{T\in\boldsymbol{P}} \Pr_{t\sim \mathsf{E}\circ\mathcal{T}}^{T\sim\mathcal{P}} [\tau(t) = 0, T] \geq 1 - \delta. \tag{1}$$

Definition 2 (Pauli Detection). *We say that a trappified scheme* $\boldsymbol{P}$ ϵ-detects $\mathcal{E} \subset \mathcal{G}_V$ *if:*

$$\forall \mathsf{E} \in \mathcal{E}, \sum_{T \in \boldsymbol{P}} \Pr_{\substack{T \sim \mathcal{P} \\ t \sim \mathsf{E} \circ \mathcal{T}}} [\tau(t) = 1, T] \geq 1 - \epsilon. \tag{2}$$

Definition 3 (Pauli Correctness (Informal)). *We say that a computation is correct on deviation* E *if the output distribution is the same whether the deviation is applied or not.*

The virtue of defining these properties is that the sets of deviations above can be characterised efficiently and yield correctness and security with negligible errors for the overall protocol:

Theorem 1 (Security of Protocol 1, Combining Theorems 8 and 13 from [22]). *Let* $\mathfrak{C}$ *be a set of classical* BQP *computations on graph* G*. Let* $\boldsymbol{P}$ *be a trappified scheme on graph* G *that* ϵ*-detects a set of Pauli deviations* $\mathcal{E}_1$ *and is* δ*-insensitive to* $\mathcal{E}_2$ *and perfectly insensitive to* I*. Assume that all computations in a set* $\mathfrak{C}$ *are correct on* $\mathcal{G}_V \setminus \mathcal{E}_1$*. Let* $n = s + d$ *for* d *and* s *proportional to* n*, and* c *the bounded error of* BQP*. Let* w *be the maximum number of test rounds allowed to fail, chosen such that* $w < \frac{2c-1}{2c-2} s(1-\epsilon)$*.*

Then Protocol 1 $\eta(n)$*-constructs the Secure Delegated Quantum Computation Resource 1 for computations in set* $\mathfrak{C}$ *in the Abstract Cryptography framework, where the leak is defined as* $l_\rho = (\mathfrak{C}, G, \boldsymbol{P}, \preceq_G)$*, for* $\eta(n)$ *negligible in* n*.*

Note that the value of $\eta(n)$ heavily depends on the value of δ and ϵ, in particular via the coefficient in the exponential. This means that it is crucial to minimise these detection and insensitivity errors.

Notice also that w is also reliant on ϵ, and minimising this error also allows the protocol to tolerate more honest errors before aborting. This noise-robustness of Protocol 1 can be characterised as follows.

Theorem 2 (Noise-Robustness of Protocol 1, Combining Theorems 9 and 13 from [22]). *For the same parameter choices as in Theorem 1, assume an execution of Protocol 1 with an honest-but-noisy Server such that* p *is the probability that less than* $\frac{w}{s\delta} n$ *rounds are affected by a Pauli error. Then the Client accepts the outcome with probability* $(1-p)(1-\delta')$*, for* δ' *negligible in* n*.*

Since the protocol is secure, we can then guarantee that, if the client accepts, the outcome is also correct up to negligible total variational distance. This means that for machines with a constant amount of global noise below a certain bound, our protocol accepts and yields the correct result with overwhelming probability.

Traps from Stabiliser Tests. As a result, the performance of Protocol 1 is governed by the choice of s, d, w defined above, together with the error detection and insensitivity capabilities of traps in $\boldsymbol{P}$. Ref. [22] § 6.1 shows how to construct general traps from subset stabiliser testing. Indeed, let S be the stabiliser group for $|G\rangle\langle G|$ the graph state associated to G, and consider $\{\mathsf{S}_v = \mathsf{X}(v) \bigotimes_{(v,w)\in E} \mathsf{Z}(w), v \in V\}$ the set of canonical generators of S. One can then associate a trap to each $\mathsf{R} \in \mathrm{S}$ by (i) having the Client prepare a $+1$ eigenspace of R as input, and (ii) delegating to the Server the computation consisting of measuring R using the UBQC Protocol 4. An accepted trap then corresponds to the measurement of R returning the $+1$ eigenvalue.

For the preparation, the client sets each qubit $v \in V$ in the $+1$ eigenstate of $\mathsf{R}(v)$ with $\mathsf{R}(v)$ being uniquely defined by:

$$\begin{aligned} \mathsf{R} &= \bigotimes_{v \in V} \mathsf{R}(v) \\ \mathsf{R}(v_0) &\in \{\pm 1\} \times \{\mathsf{I}, \mathsf{X}, \mathsf{Y}, \mathsf{Z}\}, \text{, for } v_0 = \mathrm{argmin}_{v \in V} \mathsf{R}(v) \neq \mathsf{I} \\ \mathsf{R}(v) &\in \{\mathsf{I}, \mathsf{X}, \mathsf{Y}, \mathsf{Z}\}, \text{ for } v \in V \setminus v_0. \end{aligned}$$

This corresponds to preparing a $+1$ eigenstate of the group generated by $\{\mathsf{R}(v), v \in V\}$ which contains R hence satisfying (i) above.

For the delegated computation consisting of measuring R, the Client simply instructs the Server to measure each qubit in the X-basis, getting outcome $t(v)$. The motivation for these measurements is better understood by examining to which observable they correspond on the inputs provided by the Client. To this

end, one can conjugate each $\mathsf{X}(v)$ by $\prod_{(v,w)\in E}\mathsf{CZ}_{(v,w)}$, the entangling operation that the Server performs prior to the measurement. A simple stabiliser computation shows that $\mathsf{X}(v)$ is mapped to S_v. That is, measuring $\mathsf{X}(v)$ after the entangling operation corresponds to measuring S_v on the inputs provided by the client. As R is uniquely defined as $\prod_{v\in\mathbb{1}_\mathsf{R}}\mathsf{S}_v$ for some set $\mathbb{1}_\mathsf{R}\subset V$, and because S is abelian, the outcome of R on the input state provided by the client is the binary sum of the outcomes of S_v. Using the above correspondence for measurements of S_v on the inputs, one concludes that $\bigoplus_{v\in\mathbb{1}_\mathsf{R}}t(v)$ determines the outcome of the measurement of R on the inputs provided by the Client. Combining the preparation and the measurement, the Client therefore expects that for an honest Server, $\bigoplus_{v\in\mathbb{1}_\mathsf{R}}t(v)=1$, thereby fulfilling (ii) above.

The freedom in choosing which R's to include in the trappified scheme $\boldsymbol{P}$ will be at the core of constructing dummyless verification protocols.

2.2 A Natural Invariance of MBQC with Classical Input and Output

In MBQC, computation qubits, i.e. $v\in O^c$, are measured in the $\left|\pm_{\phi'(v)}\right\rangle$ basis, where $\phi'(v)\in\Theta=\left\{\frac{k\pi}{4}\right\}_{k\in\{0,\dots,7\}}$ is defined by the pattern used for the computation. As a result, the computation is invariant under rotations around the $\phi'(v)$ axis in the $X-Y$ plane just before the measurement. The reason is that such rotations leave the projectors $\left|+_{\phi'(v)}\right\rangle\!\left\langle+_{\phi'(v)}\right|$ and $\left|-_{\phi'(v)}\right\rangle\!\left\langle-_{\phi'(v)}\right|$ untouched so that it does not affect the probabilities of the outcomes of a measurement in the $\left|\pm_{\phi'(v)}\right\rangle$ basis. This property is well known and is actively used in the proof of security of the UBQC protocol as it allows to fully twirl the deviation of the server on computation qubits.

If one not only considers local unitary transformations but more generally local invertible transformations, then MBQC is also invariant under reflections through the $X-Y$ plane for $v\in O^c$. The reason is similar to the one given above: such transformations do not change the projectors onto the $\left|\pm_{\phi'(v)}\right\rangle$ basis and hence do not affect probability distributions of measurements in the $\left|\pm_{\phi'(v)}\right\rangle$ basis.

We will now explore the latter invariance in the special case of classical input classical output computations where it naturally extends to the result of the computation itself, as in such case all qubits are measured in the $X-Y$ plane.

Lemma 1. *For matrices $\rho=\sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y},\mathsf{Z}\}^{\otimes n}}\alpha_\mathsf{P}\mathsf{P}$ decomposed in the Pauli basis, let F_A be the linear map that applies the reflection through the $X-Y$ plane for all vertices in $A\subset V$, defined as*

$$F_A(\rho)=\sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y},\mathsf{Z}\}^{\otimes n}}(-1)^{\mathrm{zwt}_A(\mathsf{P})}\alpha_\mathsf{P}\mathsf{P},$$

where $\mathrm{zwt}_A(\mathsf{P})=|\{v\in A|\mathsf{P}_v=\mathsf{Z}\}|$ counts the number of vertices in A on which P equals the Pauli Z. Then, MBQC is invariant under F_A when applied right before the $\left|\pm_{\phi'(v)}\right\rangle$ measurements.

Proof. The probability to obtain the all-zero outcome when measuring all qubits $v\in O^c$ of a state ρ in the $\left|+_{\phi'(v)}\right\rangle\!\left\langle+_{\phi'(v)}\right|$-bases is given by

$$\mathrm{Tr}\left(\left(\mathbb{1}_O\otimes\bigotimes_{v\in O^c}\left|+_{\phi'(v)}\right\rangle\!\left\langle+_{\phi'(v)}\right|\right)\rho\right).$$

Decomposing the above expression in the Pauli basis yields

$$\begin{aligned}&\mathrm{Tr}\left(\left(\sum_{\mathsf{P}'\in\{\mathsf{I},\mathsf{X},\mathsf{Y}\}^{\otimes n}}\beta_{\mathsf{P}'}\mathsf{P}'\right)\left(\sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y},\mathsf{Z}\}^{\otimes n}}\alpha_\mathsf{P}\mathsf{P}\right)\right)\\&=\sum_{\mathsf{P}'\in\{\mathsf{I},\mathsf{X},\mathsf{Y}\}^{\otimes n}}\sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y},\mathsf{Z}\}^{\otimes n}}\beta_{\mathsf{P}'}\alpha_\mathsf{P}\,\mathrm{Tr}\left(\mathsf{P}'\mathsf{P}\right)=\sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y}\}^{\otimes n}}\beta_\mathsf{P}\alpha_\mathsf{P}2^{|V|}.\end{aligned}$$

Calculating the same probability for the all-zero outcome when measuring after applying F_A yields

$$\begin{aligned}
&\mathrm{Tr}\left(\left(\mathbb{1}_O \otimes \bigotimes_{v\in O^c} |+_{\phi'(v)}\rangle\langle+_{\phi'(v)}|\right) F_A(\rho)\right)\\
&\mathrm{Tr}\left(\left(\sum_{\mathsf{P}'\in\{\mathsf{I},\mathsf{X},\mathsf{Y}\}^{\otimes n}} \beta_{\mathsf{P}'}\mathsf{P}'\right)\left(\sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y},\mathsf{Z}\}^{\otimes n}} (-1)^{\mathrm{zwt}_A(\mathsf{P})}\alpha_{\mathsf{P}}\mathsf{P}\right)\right)\\
&= \sum_{\mathsf{P}\in\{\mathsf{I},\mathsf{X},\mathsf{Y}\}^{\otimes n}} \beta_{\mathsf{P}}\alpha_{\mathsf{P}}2^{|V|},
\end{aligned}$$

and therefore the same value. By an analogous argument, the probabilities for any other outcome coincide as well. ■

Note that $F_A(\rho)$ might not always be a physical state. As a result, if $|G\rangle\langle G|$ denotes the graph state used to implement classical input classical output MBQC on G, one has:

$$|G\rangle\langle G| = \frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathcal{S}}\mathsf{S}, \tag{3}$$

for $\mathcal{S}$ the stabiliser group of the graph state, so that for any $\mathsf{S}'\in\mathrm{S}$ we have:

$$\mathrm{Tr}(\mathsf{S}'\,|G\rangle\langle G|) = \frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathrm{S}}\mathrm{tr}(\mathsf{S}'\mathsf{S}) = 1. \tag{4}$$

In turn, this implies that

$$\mathrm{Tr}(\mathsf{S}'F_A(|G\rangle\langle G|)) = \frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathrm{S}}(-1)^{\mathrm{zwt}_A(\mathsf{S})}\,\mathrm{Tr}(\mathsf{S}'\mathsf{S}) = (-1)^{\mathrm{zwt}_A(\mathsf{S}')}. \tag{5}$$

If $F_A(|G\rangle\langle G|)$ was a physical state, Equation (5) would imply that it would be stabilised by $(-1)^{\mathrm{zwt}_A(\mathsf{S})}\mathsf{S}$ for all $\mathsf{S}\in\mathrm{S}$. The group structure of stabilisers would then imply that it is also stabilised by the operator $(-1)^{\mathrm{zwt}_A(\mathsf{S})+\mathrm{zwt}_A(\mathsf{S}')}\mathsf{S}\mathsf{S}'$ for all $\mathsf{S},\mathsf{S}'\in\mathcal{S}$, and hence $\mathrm{zwt}_A(\mathsf{S}\mathsf{S}')\equiv\mathrm{zwt}_A(\mathsf{S})+\mathrm{zwt}_A(\mathsf{S}')\pmod 2$.

However, for $A\subsetneq V$, $\mathrm{zwt}_A(\cdot)$ does not in general satisfy the above equation. More precisely, take $(v,w)\in E$, the stabiliser $\mathsf{S}_v\mathsf{S}_w$ will then satisfy $\mathrm{zwt}(\mathsf{S}_v\mathsf{S}_w)\equiv\mathrm{zwt}(\mathsf{S}_v)+\mathrm{zwt}(\mathsf{S}_w)-1\pmod 2$. This is because the overlap of S_v and S_w at v will always remove a single Z coming from S_w, while if the two stabilisers overlap at some other Z location in A this will remove 2 from the weight.

Conversely[8], setting $A=V$, then indeed $\mathrm{zwt}_V(\mathsf{S}\mathsf{S}')\equiv\mathrm{zwt}_V(\mathsf{S})+\mathrm{zwt}_V(\mathsf{S}')\pmod 2$ for all $\mathsf{S},\mathsf{S}'\in\mathcal{S}$. Moreover, it is possible to find a unitary transformation that has the same effect as F_A on $|G\rangle\langle G|$, implying that $F_A(|G\rangle\langle G|)$ is then a physical state, as witnessed by the following lemma.

Lemma 2. *For any graph $G=(V,E)$ it holds that $F_A(|G\rangle\langle G|) = \mathsf{U}\,|G\rangle\langle G|\,\mathsf{U}^\dagger$, where*

$$\mathsf{U} = \prod_{\substack{v\in V,\\ \deg v\equiv 1\,(\mathrm{mod}\ 2)}} \mathsf{Z}_v$$

describes the application of Z's to all odd-degree vertices of G.

Proof. It will be useful to rewrite the stabilisers of $|G\rangle\langle G|$ as follows. For every $\mathsf{S}\in\mathrm{S}$, there exists exactly one subset of vertices $V_{\mathsf{S}}\subset V$ such that

$$\mathsf{S} = \prod_{v\in V_{\mathsf{S}}}\mathsf{S}_v.$$

[8] More generally, for disconnected graphs this holds if and only if A is a connected component or a union of connected components.

We start with the right side of the equation:

$$\mathsf{U}\,|G\rangle\langle G|\,\mathsf{U}^\dagger = \mathsf{U}\left(\frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathrm{S}}\mathsf{S}\right)\mathsf{U}^\dagger = \frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathrm{S}}\mathsf{U}\left(\prod_{v\in V_\mathsf{S}}\mathsf{S}_v\right)\mathsf{U}^\dagger.$$

Complementing $\mathsf{U}^\dagger\mathsf{U}$ terms, this expression gives:

$$\frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathrm{S}}\prod_{v\in V_\mathsf{S}}\mathsf{U}\mathsf{S}_v\mathsf{U}^\dagger.$$

It is easy to verify that $\mathsf{U}\mathsf{S}_v\mathsf{U}^\dagger = (-1)^{\mathrm{zwt}_V(\mathsf{S}_v)}\mathsf{S}_v$ because of the particular structure of U, and hence the above expression equals

$$\frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathcal{S}}\prod_{v\in V_\mathsf{S}}(-1)^{\mathrm{zwt}_V(\mathsf{S}_v)}\mathsf{S}_v.$$

Exploiting the additivity of $\mathrm{zwt}_V(\cdot)$, we arrive at

$$\frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathcal{S}}(-1)^{\sum_{v\in V_\mathsf{S}}\mathrm{zwt}_V(\mathsf{S}_v)}\mathsf{S} = \frac{1}{2^{|V|}}\sum_{\mathsf{S}\in\mathcal{S}}(-1)^{\mathrm{zwt}_V(\mathsf{S})}\mathsf{S} = F_V(|G\rangle\langle G|),$$

which concludes the proof. ■

Combining the statements of Lemma 1 and Lemma 2, we finally arrive at the following result, capturing the inherent invariance of classical I/O MBQC to one specific nontrivial error.

Lemma 3. *Let $G=(V,E)$ be a graph and U be the unitary operation given by*

$$\mathsf{U} = \prod_{\substack{v\in V,\\ \deg v\equiv 1 \,(\mathrm{mod}\ 2)}}\mathsf{Z}_v,$$

describing the application of Z's to all odd-degree vertices of G. For MBQC on G with classical input and output, the application of U before the measurements has no effect on the results of the computation.

Summarising the results of this section, for any classical-input classical-output MBQC there exists a non-trivial and non-stabiliser deviation that has no influence on the results of the computation. It is important to bear in mind the harmlessness of this error when constructing a verification scheme, as dummyless stabiliser tests will – by construction – not be able to detect it.

2.3 Dummyless Verification

We now arrive at the core of this section: designing single-round traps restricted to preparing states in the $X-Y$ plane. Using the construction of traps from Section 2.1, it amounts to finding a set of stabilisers of $|G\rangle\langle G|$ that are only made out of I, X, Y tensor products.

More precisely, we show that

Lemma 4. *For any $G=(V,E)$, consider the graph state $|G\rangle$ and its stabiliser group S. Then, it is always possible to find $|V|-1$ generators of S that are tensor products of I, X and Y only.*

Proof. We proceed constructively and exhibit a set of $|V|-1$ generators of R, subgroup of S, and show that $|\mathrm{R}| = 2^{|V|-1}$.

We start with one such stabiliser, $\mathsf{R}_{\mathrm{full}} = \prod_v \mathsf{S}_v$. This follows simply from

$$\mathsf{R}_{\mathrm{full}}(v) = \mathsf{X}\mathsf{Z}^{\deg(v)}, \tag{6}$$

as for qubit v, S_v contributes to the X and all neighbours contribute a Z each. Additionally, this shows that for vertices v of even degree $\mathsf{R}_{\setminus v} = \prod_{w\in V\setminus v}\mathsf{S}_v = \mathsf{R}_{\text{full}}\mathsf{S}_v$ is also a tensor product of I, X, Y. This is because removing S_v from R_{full} leaves an I at v and changes by one the number of Zs on the neighbours of v. Unfortunately, removing S_v for v of odd degree leaves a Z at v. To further remove this unwanted Z, one can also remove one stabiliser S_w from a neighbouring node w of v from the product. If, in addition, w is of odd degree, then the obtained stabiliser will be a tensor product of I, X, Y only. The reason is that at w, one Z has been removed when S_v was removed from R_{full} thereby leaving an X at w, so that removing S_w leaves an I. In the general case, one can always remove from R_{full} the stabilisers S_v along a chain between u and w consisting of even degree nodes except for u and w that are odd degree. We denote by $\mathsf{R}_{\setminus(u,w)}$ such generator. Note that a given odd-degree node will always be in at least one such stabiliser as there are always an even number of odd degree nodes in a connected component of a graph.

Now define the group R generated by R_{full}, $\mathsf{R}_{\setminus v}$ and $\mathsf{R}_{\setminus(u,w)}$ above. Notice that multiplying R_{full} with $\mathsf{R}_{\setminus v}$ gives S_v, so that S_v is in R for even $\deg(v)$. Similarly, multiplying R_{full} with $\mathsf{R}_{\setminus(u,w)}$ and S_v for v an even-degree node linking u to w shows that any $\mathsf{S}_u\mathsf{S}_w$ with u and w odd-degree nodes are also in R. Therefore, R contains all stabilisers that have an arbitrary number of even-degree node and an even number of odd-degree ones. Counting the number of such stabilisers gives $2^{|V|-1}$ while we know that the size of S is $2^{|V|}$, which concludes the proof. ■

We now consider the trappified scheme $\boldsymbol{P}$ that can be obtained by sampling uniformly at random from all these traps rounds. We can characterise the errors that can be detected by $\boldsymbol{P}$ and those to which it is insensitive using properties of stabilisers. To this end, recall that if a Pauli error E is applied right before the measurement of a 2-outcome observable M, then (i) the measurement outcome probabilities are unchanged if $[\mathsf{E},\mathsf{M}] = 0$, and (ii) are swapped for $\{\mathsf{E},\mathsf{M}\} = 0$. Hence, whenever E commutes with $\bigotimes_{v\in\mathbb{1}_\mathsf{R}}\mathsf{X}(v)$ the trap never detects E, whereas it always detects it whenever it anticommutes. As a consequence, the set of detectable errors is the set of errors that anticommute with at least one of the $\bigotimes_{v\in\mathbb{1}_\mathsf{R}}\mathsf{X}(v)$ for R a dummyless trap measurement.

Hence, for an error $E = \mathsf{E}_Z\mathsf{E}_X$ we need to assess whether there exists at least one R in $\boldsymbol{P}$ such that $|\mathbb{1}_\mathsf{R}\cap\mathbb{1}_{\mathsf{E}_Z}| \equiv 1 \pmod 2$ – where we have implicitely defined E_Z (resp. E_X) as the operators made of Zs at location of Y or Z qubits in E (resp. X or Y qubits). To this end, consider F such that $\mathsf{U}_G\mathsf{F} = \mathsf{E}\mathsf{U}_G$ where U_G is the entangling operation for creating the graph state. Because a trap amounts to measuring the corresponding stabiliser before the entangling operation, the above question amounts to knowing whether F commutes with the stabilisers used to define the dummyless traps of $\boldsymbol{P}$. Alternatively, we can answer this question by finding out which Pauli operations commute with all stabilisers defining the dummyless traps while not being a product of them.

Using Lemma 4, there is one generator S_0 of S that is not in R and such that all errors that commute with R and are not in R are of the form $\mathsf{S}_0\mathsf{R}$. From the above description of R, S_0 can be taken as being equal to Z on all odd-degree nodes. S_0 commutes with all elements of R since they have an even number of S_v for v odd-degree, and it is not in R as R has no element with Zs only. Yet, Lemma 3 shows that while S_0 cannot be detected, it is indeed harmless for the computation.

Hence, we are led to conclude that all possibly harmful errors are detected by the trappified scheme $\boldsymbol{P}$. Using § 2.1, we conclude that

Theorem 3. *Let $G = (V, E)$ be a graph, and $\boldsymbol{P}$ the trappified scheme on G defined by sampling at random from a generating set of* R *containing only stabilisers with no* Z*s. Then, $\boldsymbol{P}$ constructs the SDQC Resource 1 for* BQP *computations that can be embedded on the graph G with negligible correctness and security errors.*

This follows from the fact that Theorem 1 states that a secure verification scheme can be built from a trappified scheme that 1) detects a specific set $\mathcal{E}$ of Z-Pauli errors, and 2) correctly evaluates the target computation in the presence of any other Z-Pauli error in $\mathcal{G}_V^\mathsf{Z}\setminus\mathcal{E}$. Lemma 3 then shows that there is a specific error E^* which never affects the output distribution of the target computation and which therefore does not need to be detected. It hence suffices to find a dummyless trappified scheme detecting $\mathcal{E} = \mathcal{G}_V^\mathsf{Z}\setminus\{\mathsf{I},\mathsf{E}^*\}$. As shown with Lemma 4, it is indeed possible to find such a trappified scheme. Therefore, this settles the question whether dummyless verification for BQP is possible by the affirmative.

2.4 Concrete Dummyless Tests

The previous subsection left open how to concretely construct the trappified scheme $\boldsymbol{P}$. More precisely, since the efficiency of the resulting SDQC protocol is tightly linked to the detection rate of the trappified scheme, it is important to minimise its detection, insensitivity and correctness errors. In this section, we discuss the question of optimising the detection rate. In particular, we construct concrete dummyless trappified schemes for universal BQP computations with constant detection rates, independent of the size of the computation.

[22] shows that the general optimisation problem of maximising the detection rate can be expressed in the language of linear programming. Adapted to the case of dummyless trappified schemes, we recall it in the following, as Problem 1.

Problem 1 Optimisation of the Distribution of Tests

Given

- the set of errors $\mathcal{E} = \mathcal{G}_V^Z \setminus \{\mathsf{I}, \mathsf{E}^*\}$ to be detected,
- the set of dummyless tests $\mathcal{T}_{\text{dummyless}}$,
- the relation between tests and errors describing whether a test detects an error, $R : \mathcal{T}_{\text{dummyless}} \times \mathcal{E} \to \{0, 1\}$,

find an optimal distribution $p : \mathcal{T}_{\text{dummyless}} \to [0, 1]$ **maximising** the detection rate $\epsilon \in [0, 1]$ **subject to** the following conditions:

- p describes a probability distribution, i.e. $\sum_{T \in \mathcal{T}_{\text{dummyless}}} p(T) \leq 1$,
- errors are detected at least with the target detection rate, i.e.

$$\forall E \in \mathcal{E} : \sum_{\substack{T \in \mathcal{T}_{\text{dummyless}} \\ R(T,E)=1}} p(T) \geq \epsilon.$$

For any feasible solution to Problem 1, the trappified scheme induced by the given distribution of tests gives rise to a secure dummyless SDQC protocol if and only if the detection rate satisfies $\epsilon > 0$.

Recall from Section 2.2 the structure of the harmless error:

$$\mathsf{E}^* = \prod_{\substack{v \in V(G), \\ \deg(v) \equiv 1 \pmod 2}} \mathsf{Z}_v.$$

Further, as described in Section 2.3, the set of dummyless tests can be expressed as:

$$\mathcal{T}_{\text{dummyless}} = \left\{ \prod_{v \in V_{\text{trap}}} \mathsf{X}_v \prod_{w \in N_G(v)} \mathsf{Z}_w \;\middle|\; V_{\text{trap}} \subseteq V, \forall v \notin V_{\text{trap}} : |N_G(v) \cap V_{\text{trap}}| \equiv 0 \pmod 2 \right\}.$$

The last condition ensures that there are no vertices with a single Z in the respective stabiliser. In this way, every test can be identified with the subset of vertices which act as traps, or equivalently with the complement, the subset of vertices which act as *holes*, i.e. vertices on which the respective stabiliser equals the identity and which can therefore be ignored by the decision function of the trappified scheme. In the following we will also write $V_{\text{trap}}(T)$ and $V_{\text{hole}}(T)$ as shorthands for these two sets of vertices.

Analogously, we write $V_{\text{error}}(\mathsf{E})$ for the set of vertices on which the error E is not equal to the identity (and therefore equals the Pauli Z). This makes it easy to give a short description of the relation R:

$$R : (T, \mathsf{E}) \mapsto |V_{\text{error}}(\mathsf{E}) \cap V_{\text{trap}}(T)| \pmod 2.$$

Handling Errors on Even-degree Vertices. As described in Section 2.3, for all even-degree vertices $v \in V$, the test T with $V_{\text{hole}}(T) = \{v\}$ is indeed dummyless. Generalising this concept, for any independent set V^* of even-degree vertices, we can define a dummyless test T with $V_{\text{hole}}(T) = V^*$. Similarly to the construction of

tests in [22], any (fractional) colouring of the vertices of a graph G gives rise to a distribution of independent sets of G, and therefore also a distribution of independent sets of even-degree vertices and tests. To this end, let $\mathcal{D}$ be a distribution of independent sets of G such that

$$\forall v \in V : \Pr_{I \leftarrow \mathcal{D}}[v \in I] \geq \frac{1}{\chi_f(G)},$$

where $\chi_f(G)$ is the fractional chromatic number of G. This distribution exists by definition of the fractional chromatic number. Consider the test strategy given by the distribution $\mathcal{D}_{\text{even}}$ of tests in $\mathcal{T}_{\text{dummyless}}$ described as follows:

1. Sample an independent set: $V_1 \leftarrow \mathcal{D}$.
2. Restrict the set to even-degree vertices: $V_2 = V_1 \cap V_{\text{even}}(G)$, where $V_{\text{even}}(G) = \{v \in V \mid \deg(v) \equiv 0 \pmod 2\}$.
3. Choose a random subset to determine the location of holes: $V_3 \leftarrow \mathcal{U}(\wp(V_2))$.
4. Perform the dummyless test T determined by $V_{\text{hole}}(T) = V_3$.

As the following Lemma shows, this strategy allows for a detection rate of errors that affect even-degree vertices that scales inversely with the fractional chromatic number of the graph.

Lemma 5 (Even-degree Error Detection). *The above-mentioned test strategy* $\left(\frac{1}{2\chi_f(G)}\right)$*-detects the error set* $\mathcal{E}_{even} = \{E \in \mathcal{G}_V^{\mathsf{Z}} \mid V_{error}(E) \cap V_{even} \neq \emptyset\}$*, i.e.*

$$\forall E \in \mathcal{E}_{even} : \quad \mathbb{E}_{T \leftarrow \mathcal{D}_{even}}[|V_{error}(E) \cap V_{trap}(T)| \equiv 1 \pmod 2] \geq \frac{1}{2\chi_f(G)}.$$

Proof. Let $E \in \mathcal{E}_{\text{even}}$. Then, by definition of the test distribution, it holds that

$$\begin{aligned}
&\mathbb{E}_{T \leftarrow \mathcal{D}_{\text{even}}}[|V_{\text{error}}(E) \cap V_{\text{trap}}(T)| \equiv 1 \pmod 2] \\
&\geq \mathbb{E}_{V_3 \leftarrow \mathcal{U}(\wp(V_2))}[|V_{\text{error}}(E) \cap V_3| \equiv 1 \pmod 2 \mid V_{\text{error}}(E) \cap V_2 \neq \emptyset] \\
&\qquad \cdot \Pr_{V_1 \leftarrow\!\$\, \mathcal{D}}[V_{\text{error}}(E) \cap V_1 \cap V_{\text{even}}(G) \neq \emptyset] \\
&\geq \frac{1}{2} \cdot \frac{1}{\chi_f(G)},
\end{aligned}$$

which concludes the proof. ∎

Handling Errors on Odd-degree Vertices. Since all errors acting non-trivially on even-degree vertices are already handled in the previous case, it remains to detect errors that affect only odd-degree vertices and act as the identity on even-degree vertices.

To this end, we construct a specific type of test. For $k \geq 2$, let $(v_1, \ldots, v_k) \in V^k$ be a chain of vertices in G satisfying the following conditions:

1. The end vertices are of odd degree: $\deg(v_1) \equiv \deg(v_k) \equiv 1 \pmod 2$.
2. All intermediate vertices are of even degree: $\deg(v_2) \equiv \cdots \equiv \deg(v_{k-1}) \equiv 0 \pmod 2$.
3. Only subsequent vertices are neighbours in G:

$$\forall i, j \in \{1, \ldots, k\} : \quad \{v_i, v_j\} \in E(G) \Leftrightarrow |i - j| = 1.$$

It is easy to verify that under these conditions there exists a valid dummyless test T with $V_{\text{hole}}(T) = \{v_1, \ldots, v_k\}$. Note, that there might not be a chain of this type in G for any pair of odd-degree vertices as end points. However, it is possible to connect any two odd-degree vertices through a chain of chains that might traverse other odd-degree vertices at the end and starting points of chains. In this way, it is possible to choose a "spanning tree" of $(|V_{\text{odd}}(G)| - 1)$ chains that connects all odd-degree vertices in the graph G.

Define the set of errors on odd-degree nodes only as $\mathcal{E}_{\text{odd}} = \{E \in \mathcal{G}_V^{\mathsf{Z}} \mid V_{\text{error}}(\mathsf{E}) \cap V_{\text{even}}(G) = \emptyset \wedge V_{\text{error}}(\mathsf{E}) \cap V_{\text{odd}}(G) \neq \emptyset\}$ and let $\mathsf{E} \in \mathcal{E}_{\text{odd}} \setminus \{\mathsf{E}^*\}$. Then, there must exist two odd-degree vertices $v_1 \in V_{\text{error}}(\mathsf{E})$ and $v_2 \notin V_{\text{error}}(\mathsf{E})$. But then at least one of the chains connecting v_1 and v_2 with start in a vertex affected by the error E and end in a vertex unaffected by E. Since all intermediate vertices are of even degree and therefore unaffected by E, the test given by this chain detects E. This essentially shows the following statement.

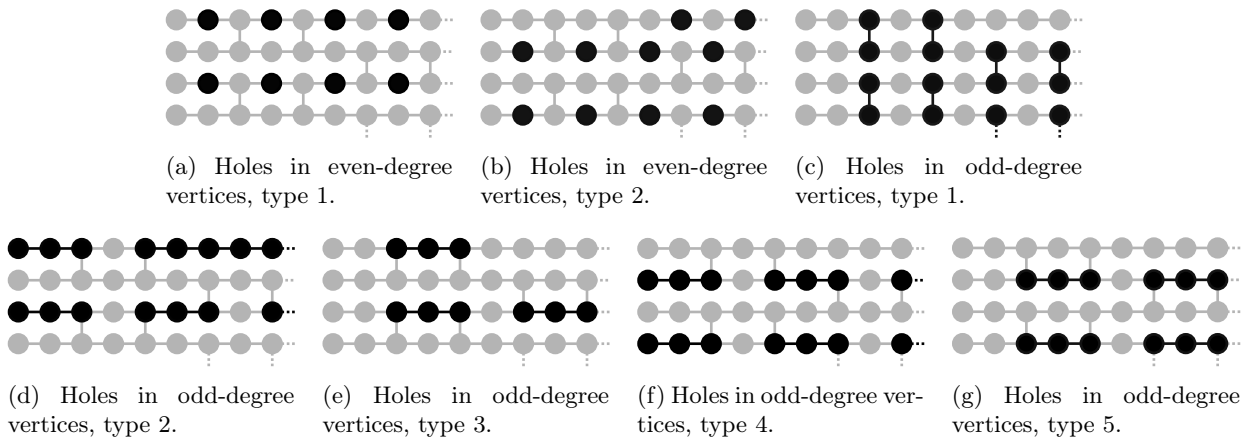
Lemma 6 (Odd-degree Error Detection). *There exists an efficient testing strategy that* $\left(\frac{1}{|V_{odd}(G)|-1}\right)$*-detects errors in* $\mathcal{E}_{odd} \setminus \{\mathsf{E}^*\}$.

Combining the testing strategies from Lemma 5 and Lemma 6 immediately yields the following result for testing strategies on general graphs.

Lemma 7 (Error Detection on General Graphs). *For any graph* G*, there exists an efficient testing strategy that* ε*-detects* $\mathcal{E} = \mathcal{G}_V^{\mathsf{Z}} \setminus \{\mathsf{I}, \mathsf{E}^*\}$*, where*

$$\begin{aligned}\varepsilon &= \frac{1}{2\chi_f(G)(|V_{odd}(G)|-1)}\left(\frac{1}{2\chi_f(G)} + \frac{1}{|V_{odd}(G)|-1}\right)^{-1} \\ &\geq \frac{1}{2}\min\left\{\frac{1}{2\chi_f(G)}, \frac{1}{|V_{odd}(G)|-1}\right\}.\end{aligned}$$

This already shows that the detection rate that is achievable on general graphs decreases at most linearly in the number of vertices of the graph. This lower bound is however far from tight in many cases. In fact, even for universal graph states a constant lower bound is possible as the following result shows.



(a) Holes in even-degree vertices, type 1.

(b) Holes in even-degree vertices, type 2.

(c) Holes in odd-degree vertices, type 1.

(d) Holes in odd-degree vertices, type 2.

(e) Holes in odd-degree vertices, type 3.

(f) Holes in odd-degree vertices, type 4.

(g) Holes in odd-degree vertices, type 5.

Fig. 1: The seven types of dummyless tests for the brickwork graph. A trap configuration is sampled by randomly choosing one of the seven types, and then in cases 1a-1b sampling uniformly at random a subset of marked vertices as holes, and in cases 1c-1g sampling uniformly at random a subset of marked chains as holes.

Lemma 8 (Error Detection on the Brickwork State). *Let* G *be a brickwork graph. Then, there exists an efficient testing strategy that* $(1/14)$*-detects* $\mathcal{E} = \mathcal{G}_V^{\mathsf{Z}} \setminus \{\mathsf{I}, \mathsf{E}^*\}$.

Proof Sketch. To detect errors affecting even-degree vertices, use the strategy from Lemma 5. As the brickwork graph is bipartite, this will yield a detection rate of $1/4$.

To detect errors on odd-degree vertices, follow the strategy from Lemma 6, but use chains that can be tested in parallel to boost the detection rate. There are five classes of chains between odd-degree vertices that can each be run at the same time. One class consists of all vertical chains, and the other four of horizontal chains where every class contains chains only in every second row and only every second horizontal chain on these rows. By testing random subsets of these classes of chains, the detection rate in this case is lower bounded by $1/10$.

Optimal switching between these two strategies (with probabilities $2/7$ and $5/7$) yields an overall detection rate of $1/14$. The different types of tests on the brickwork graph are depicted in Figure 1. ■

3 Collaborative State Preparation

Following the approach outlined in § 1.3, we now turn to the design of a composably secure protocol for implementing the preparation of the input states required by the dummyless protocols introduced in § 2.3. The Collaborative Remote State Preparation Protocol 2 presented here will allow n Clients to collaboratively construct an encrypted state on the Server whose encryption key is held by a purely classical party called the Orchestrator. It guarantees that no malicious coalition including up to $n-1$ Clients and the Server (but not the Orchestrator) has any knowledge about the final state.

This security property is captured formally as follows. The Remote State Preparation Resource 2 (or RSP) allows one party called the Sender to prepare a quantum state on a device held by another party called the Receiver. Its simplest instantiation requires only a direct quantum channel between the two participants but more interesting scenarios can be considered, for example using untrusted relays or additional participants. We specify this resource for our specific case, i.e. sending states in the $X-Y$ plane.

Resource 2 Remote State Preparation

Inputs: The Sender has as input an angle $\theta \in \Theta = \left\{\frac{k\pi}{4}\right\}_{k\in\{0,\dots,7\}}$.
Computation by the Resource: The Resource prepares and sends the state $|+_\theta\rangle$ to the Receiver.

The goal of the Collaborative Remote State Preparation Protocol is then to construct this Remote State Preparation Resource 2 between the Orchestrator and the Server using one Quantum Channel Resource between each Client and the Server and one Secure Classical Channel Resource between each Client and the Orchestrator. This latter Resource transmits faithfully and privately any classical message from the sender to the receiver, while only leaking the size of the message to an eavesdropper.

Protocol 2 Collaborative Remote State Preparation

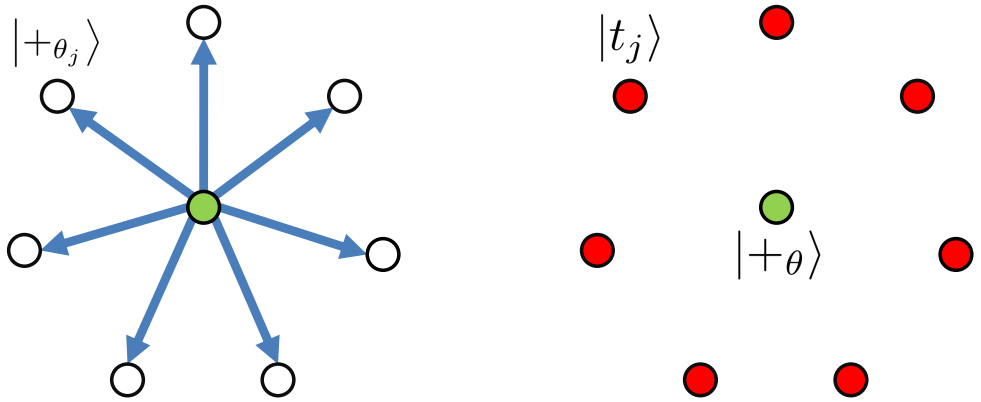
Input: The Orchestrator has as input an angle $\theta \in \Theta$. The Server and Clients have no input.
Protocol:
- Client j samples uniformly at random $\theta_j \in_R \Theta$ and sends $\left|+_{\theta_j}\right\rangle$ to the Server.
- Client j sends θ_j to the Orchestrator using a Secure Classical Channel.
- For each $j \neq n$, the Server applies $\mathsf{CNOT}_{n,j}$ between the qubits n and j, with the first being the control and the second the target. It measures the target qubit j in the computational basis with measurement outcome t_j. It sends the vector $\boldsymbol{t}$ containing all the measurement outcomes to the Orchestrator.
- The Orchestrator computes $\theta' = \theta_n + \sum_{j\in[n-1]}(-1)^{t_j}\theta_j$ and sends a correction $(b, (-1)^b\theta - \theta')$ to the Server, who applies $\mathsf{X}^b\mathsf{Z}((-1)^b\theta - \theta')$ to the unmeasured qubit, keeping it as output.

We can now state the main result of this section, namely the correctness and security of Protocol 2 in the AC framework. Both properties are proven independently below.

Theorem 4 (Security of Collaborative Remote State Preparation). *Protocol 2 perfectly constructs the Remote State Preparation Resource 2 from Secure Classical Channel Resources between each Client and the Orchestrator, for malicious coalitions that include the Server and at most $n-1$ Clients.*

Proof of Correctness. The state of the central qubit after an honest execution of Protocol 2 before the correction sent by the Client is $|+_{\theta'}\rangle$ with:

$$\theta' = \theta_n + \sum_{j\in[n-1]}(-1)^{t_j}\theta_j. \tag{7}$$



(a) The Server receives the qubits and applies CNOT gates. The central qubit n is the control, the rest are targets.

(b) The Server measures all qubits but the central one in the computational basis and gets outcomes $t_j \in \{0, 1\}$.

Fig. 2: Collaborative Remote State Preparation for eight qubits. All qubits start in the state $|+_{\theta_j}\rangle$.

It is sufficient to prove this for a pure state $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$ as control. We apply a CNOT gate with $|\phi\rangle$ as control and $|+_{\hat{\theta}}\rangle$ with $\hat{\theta} \in \Theta$ as target, followed by a measurement of this second qubit in the computational basis. Let $t \in \{0, 1\}$ be the measurement result. After tracing out the second qubit post-measurement, the system is in the following state:

$$\begin{aligned}\sqrt{2}\,\langle 0|_2\, \mathsf{X}_2^t \mathsf{CNOT}_{1,2}\,|\phi\rangle\,|+_{\hat{\theta}}\rangle &= \langle 0|_2\, \mathsf{X}_2^t \Big(\alpha|00\rangle + \alpha e^{i\hat{\theta}}|01\rangle + \beta|11\rangle + \beta e^{i\hat{\theta}}|10\rangle\Big) \\ &= \langle 0|_2 \left(\alpha|0\rangle + \beta e^{i\hat{\theta}}|1\rangle\right)|t\rangle + e^{i\hat{\theta}}\,\langle 0|_2 \left(\alpha|0\rangle + \beta e^{-i\hat{\theta}}|1\rangle\right)|t \oplus 1\rangle \\ &= \mathsf{Z}(\hat{\theta})\,|\phi\rangle\,\langle 0|t\rangle + e^{i\hat{\theta}}\mathsf{Z}(-\hat{\theta})\,|\phi\rangle\,\langle 0|t \oplus 1\rangle\end{aligned}$$

Therefore, the result of this single step is $\mathsf{Z}((-1)^t\hat{\theta})\,|\phi\rangle$ up to a global phase. Replacing the result above in the sequence of CNOT's and measurements performed by the Server where the control is qubit n and the targets are qubits $j \neq n$ yields the desired value for θ'. Finally, the rotation correction $(-1)^b\theta - \theta'$ sent by the Orchestrator, along with X^b, transform the value of the final state into $|+_\theta\rangle$. ■

Proof of Security. We first construct a Simulator against an adversarial Server and a coalition of $n-1$ Clients, which represents the worst case. The Server expects to receive n qubits and a final correction after transmitting the measurement results. The Simulator has single-query oracle access to the Remote State Preparation Resource 2 for state set $\{|+_\theta\rangle\}_{\theta\in\Theta}$. It receives a state from this resource, without the corresponding classical description, and must make the Server accept this state as its output at the end of the interaction. The actions of this Simulator are described in Simulator 1. Let h be the index associated to the honest Client.

We can now prove that no Distinguisher can tell apart the following two situations with one honest client: (i) the ideal resource interacting with the Simulator, and (ii) the real scenario.

Data and transcripts available to the Distinguisher. By construction, the Distinguisher fixes θ the angle of the desired state to be prepared at the Server output-interface. It also fixes the value of all θ_j for $j \neq h$ both in the real and ideal scenarios and has perfect knowledge of the states sent by malicious parties. It does not have access to θ_h as this is fixed by the honest client protocol.

Simulator 1 Malicious Server and $n-1$ Clients

1. The Simulator calls the Remote State Preparation Resource 2 and receives a state $|+_\theta\rangle$.
2. It then emulates the behaviour of the n Quantum Channel Resources:
   - For indices $j \neq h$, it simply forwards the state from corrupted Client j to the Server;
   - For index h, it samples uniformly at random $\theta_h \in_R \Theta$ and $b_h \in_R \{0,1\}$, and sends an encrypted version $\mathsf{Z}(\theta_h)\mathsf{X}^{b_h}(|+_\theta\rangle)$ of the state received from the RSP Resource.
3. It then emulates the Secure Classical Channel Resources and receives from each corrupted Client $j \neq h$ a value θ_j
4. It receives from the Server a bit-string of measurement results $\mathbf{t} \in [n-1]$.
5. After extending the bit-string $\mathbf{t}$ with $t_n = 0$, it computes θ' using Equation 7 and sends the correction $(t_h \oplus b_h, -\theta')$ to the Server (by impersonating the Orchestrator) and halts.

Before sending the values for the measurement outcomes, the Distinguisher receives from the non-corrupted party the state $|+_{\theta_h}\rangle$ in the real case and the state $\left|+_{(-1)^{b_h}\theta+\theta_h}\right\rangle$ in the ideal case. After sending the bit-string $\mathbf{t}$, regardless of how it was chosen, the Distinguisher receives a bit and an angle corresponding to the corrections chosen by either the Orchestrator or the Simulator. In the first case this is equal to $(b, (-1)^b\theta - \theta')$ and in the second case $(t_h \oplus b_h, -\theta')$ with b being chosen uniformly at random and θ' being computed in the exact same way in both settings (see Equation 7).

The remaining parameters in the real case and ideal cases are the received honest state, the associated measurement outcome, the X-correction bit and the Z-correction angle. This gives us the following variables that are in the hands of the Distinguisher (rows are labeled by the meaning of the corresponding data in the real setting):

	Real world	**Ideal world**
Orchestrator-chosen output angle	θ	θ
Server's received quantum state	$\|+_{\theta_h}\rangle$	$\left\|+_{(-1)^{b_h}\theta+\theta_h}\right\rangle$
Measurement result bit	t_h	t_h
Orchestrator correction bit	b	$b_h \oplus t_h$
Orchestrator correction angle	$(-1)^b\theta - (-1)^{t_h}\theta_h$	$-(-1)^{t_h}\theta_h$

Indistinguishability of data and transcripts for the Distinguisher. To finish the security proof, we need to show that the distributions of the above data and transcripts are statistically indistinguishable in both scenarios. To do this, we will perform a series of row-wide operations and eliminate the parameters of the corrupted parties so that we are left with a new set of variables that will be trivially indistinguishable. The reversibility of each operation and its dependency on values that are known to the Distinguisher guarantees that it can always undo them.

First, multiply the final angle by $(-1)^{t_h}$ and use this angle to apply a rotation to the state. This transforms the above values into:

Real world	**Ideal world**
θ	θ
$\left\|+_{(-1)^{b\oplus t_h}\theta}\right\rangle$	$\left\|+_{(-1)^{b_h}\theta}\right\rangle$
t_h	t_h
b	$b_h \oplus t_h$
$(-1)^{b\oplus t_h}\theta - \theta_h$	$-\theta_h$

Note that in both cases, the value for θ_h only appears in the last row term. Since it is chosen uniformly at random both final terms follow the same distribution, meaning that they give no distinguishing advantage. We can therefore safely omit them in the rest of the process:

Real world	Ideal world
θ	θ
$\left\|+_{(-1)^{b\oplus t_h}\theta}\right\rangle$	$\left\|+_{(-1)^{b_h}\theta}\right\rangle$
t_h	t_h
b	$b_h \oplus t_h$

Since b in the first row is a bit sampled uniformly at random, we can substitute it with $b \oplus t_h$ without changing the distribution.[9] We arrive at

Real world	Ideal world
θ	θ
$\left\|+_{(-1)^{b}\theta}\right\rangle$	$\left\|+_{(-1)^{b_h}\theta}\right\rangle$
t_h	t_h
$b \oplus t_h$	$b_h \oplus t_h$

Because the b and b_h are uniformly random bits, the above two distributions are identical, which concludes the proof. ■

4 Quantum Secure Multi-Party Computation

We present in this section an extension of the SDQC Protocol 1 from Section 2.3 based on the trappified schemes in the $X-Y$ plane. We consider here that n Clients want to perform a joint MBQC computation on private classical inputs, receiving at the end either the same classical output or an abort message. There are two steps in the SDQC protocol which must be modified: the preparation of a state which is compatible with the SDQC protocol and does not leak any information to coalitions of malicious parties, and the classical interaction between with the server to drive the computation and tests. If these components are available, the composable security of the SDQC protocol ensures that the multi-party version is also secure.

The second step is purely classical once the state and computation have been fixed and we will use a Classical SMPC Resource to handle it. This Resource will also sample the trappified canvas and embed the Client's desired computation into it. Hence, no malicious coalition will be able to learn where the tests are located among the blind computations. The first step will make use of the Collaborative RSP Protocol 2 from the previous section, replacing the Orchestrator by calls to the Classical SMPC Resource. The n Clients will use it to prepare rotated $|+\rangle$ states on the Server such that the encryption angle θ is unknown to any malicious coalition, which protects the blindness of each computation.

Our resulting Secure Delegated Quantum Secure Multi-Party Computation Protocol with Classical IO (Protocol 3) is therefore an information-theoretic upgrade of the Classical SMPC functionality. This is the best one can hope for without an honest majority since it is impossible in that case to construct an information-theoretically secure Quantum SMPC protocol. Crucially, no additional computational assumptions are used beyond what is required to construct the Classical SMPC Resource. This modularity means that we can instantiate our protocol using any post-quantum secure assumption which is capable of constructing a Classical SMPC.

Quantum Secure Multi-Party Computation Resource. Our protocol will construct the following Quantum Secure Multi-Party Computation Resource 3. It has $n+1$ interfaces, one for each Player and the last one for an Eavesdropper. It allows n Players to perform a collectively defined quantum computation C over their private classical inputs with the guarantee that their computation is either executed properly, in which case Player j receives the correct classical output, or it is aborted altogether. It is allowed to leak a known value l_ρ about the Players' computation and input on the Eavesdropper's filtered interface.

[9] This is the hidden reason for the additional encryption via X^b in the protocol.

Resource 3 Quantum Secure Multi-Party Computation with Classical IO

Inputs:
- Player j sends a classical bit-string x_j. It can also input two bits f_j and c_j as a filtered interface.
- The n Players send the classical description of a quantum polynomial-time computation C with classical inputs and outputs.
- The Eavesdropper can input two bits e and c as a filtered interface.

Computation by the Resource:
- If $e = 1$, the Resource sends the leakage l_ρ to the Eavesdropper's interface.
- If $c = 1$ or there exists j such that $c_j = 1$, the Resource sends Abort to all Players j such that $c_j = 0$.
- It computes $O = \mathsf{C}(x)$, where x is the concatenation of strings x_j.
- If there exists $j \in [n]$ such that $f_j = 1$, it sends O to Player j.
- If there has been no abort at this stage, it sends the outputs O to all other Players j in a similar fashion.

In order to construct this resource, we will make use of its classical equivalent. Our protocol will in the end be an information theoretical upgrade of the following Resource.

Classical Secure Multi-Party Computation Resource. Resource 4 allows n Players to provide their private inputs and perform a collectively defined computation C on them with the guarantee that the computation is performed properly. We assume that it keeps an internal state between calls.

Resource 4 Classical Secure Multi-Party Computation

Inputs:
- Player j sends a classical bit-string x_j. It can also input two bits f_j and c_j as a filtered interface.
- The n Players send the description of a classical polynomial-time computation C.

Computation by the Resource:
- If there exists j such that $c_j = 1$, the Resource sends Abort to all Players j such that $c_j = 0$.
- It computes $O = C(x)$, where x is the concatenation of strings x_j.
- If there exists $j \in [n]$ such that $f_j = 1$, it sends O to Player j.
- If there has been no abort at this stage, it sends the outputs O to all other Players j in a similar fashion.

Delegated QSMPC Protocol. Our final protocol will be built upon the two presented earlier. In an execution the Trappified Delegated Blind Computation Protocol 1, the Client can perform all of its classical interactions with the Server via a Classical SMPC Resource 4 if it provides this resource with its input and computation (angles and flow). This resource is then responsible for sampling all the secret parameters – angles, bits, order of test and computation runs, which tests to perform – and simply instructs the Client to prepare specific states to send to the Server. Since only rotated $|+\rangle$ states are required for this verification protocol, this step can further be replaced by an instance of the Remote State Preparation Resource 2 for states $\{|+_\theta\rangle\}_{\theta\in\Theta}$, as sending a state from this set is a perfect protocol constructing the RSP Resource. We can then finally replace this resource by the Collaborative Remote State Preparation Protocol 2, in which the Orchestrator is played by the Classical SMPC Resource.

In essence, the Classical SMPC together with the Collaborative RSP emulate the behaviour of the honest Client in an execution of the Trappified Delegated Blind Computation Protocol, whose tests – described in Section 2.4 – needed to be tailored specifically to require only the preparation of rotated $|+\rangle$ states. The full description is given below in Protocol 3. We continue to refer to the Classical SMPC Resource as the Orchestrator for simplicity, since in the Abstract Cryptography framework there is no formal difference between an honest party and an interactive Resource.

Protocol 3 Secure Delegated Quantum Secure Multi-Party Computation with Classical IO

Public Information:

- $G = (V, E, I, O)$, a graph with input and output vertices I and O respectively;
- $\{I_j\}_{j\in[n]}$, a partition of the input vertices, with each I_j being associated to Client j.
- $\boldsymbol{P}$, a trappified scheme on graph G;
- $\preceq_G$, a partial order on the set V of vertices;
- N, d, w, parameters representing the number of runs, the number of computation runs, and the number of tolerated failed tests.

Clients' Inputs:

- Each Client j has as input a classical bit-string $x_j \in \{0,1\}^{|I_j|}$.
- The n Clients collaboratively have as input a set of angles $\{\phi_i\}_{i\in V}$ and a flow f which induces an ordering compatible with $\preceq_G$.

Protocol:

1. The Clients send their input x_j to the Orchestrator, together with the computation angles $\{\phi_i\}_{i\in V}$ and flow f. Let x be the concatenation of all x_j.
2. The Orchestrator and the Server perform an execution of the Trappified Delegated Blind Computation Protocol 1. Instead of having the Orchestrator send rotated states during the UBQC execution, they perform for each state an instance of the Collaborative State Preparation Protocol 2 together with the n Clients.
   (a) The Orchestrator samples uniformly at random a subset $C \subset [N]$ of size d representing the computation runs.
   (b) For $k \in [N]$:
      i. If $k \in C$, the Orchestrator sets the computation for the run to $(\{\phi_i\}_{i\in V}, f)$ with input x. Otherwise, the Orchestrator samples a test (T, σ, τ) from the trappified scheme $\boldsymbol{P}$.
      ii. The Orchestrator and Server execute the chosen run with the UBQC Protocol 4. For each qubit sent during the execution of the protocol, they instead execute the Collaborative RSP Protocol 2 together with the n Clients.
      iii. If the run is a test, the Orchestrator checks whether it passed.
   (c) If the number of failed tests is greater than w, the Orchestrator sets the output to $(\perp, \mathsf{Rej})$.
   (d) Otherwise, let O be the majority vote on the output results of the computation runs. The Orchestrator sets the output to (O, Acc).
3. The Orchestrator sends its set output to all Clients.

Extending the Functionality. The presentation above restricts how the input and output are treated for simplicity's sake and any additional efficient classical pre- and post-processing steps can be performed by the Orchestrator with no impact on the security of the protocol.

Removing the Correction in the Collaborative RSP Protocol used with UBQC. The final step of the Collaborative RSP Protocol calls for the Orchestrator to instruct the Server to apply a correction $\mathsf{X}^b\mathsf{Z}((-1)^b\theta - \theta')$ to a state which in the honest case is equal to $|+_{\theta'}\rangle$, for a random value of $b \in_R \{0,1\}$ and the Orchestrator's desired angle θ. This is required to make the protocol simulatable against a malicious coalition – otherwise, the Simulator has no way of transmitting the correct state to the Server. However, in Protocol 3 these qubits are used in an execution of the UBQC Protocol, in which the Orchestrator requests that the Server measures the qubit in the basis $\{|\pm_\delta\rangle\}$ for $\delta = \phi' + \theta + r\pi$. Together, the unitary operations on this qubit in the honest case can be written as

$$\mathsf{Z}(-\delta)\mathsf{E}\mathsf{X}^b\mathsf{Z}((-1)^b\theta - \theta')\mathsf{Z}(\theta')\,|+\rangle \otimes |\psi\rangle$$

for a state $|\psi\rangle$ representing the rest of the state and the graph entangling operation E. Then, this is equal to

$$\mathsf{Z}(-\phi' - \theta - r\pi)\mathsf{E}\mathsf{Z}(\theta - (-1)^b\theta')\mathsf{Z}((-1)^b\theta')\,|+\rangle \otimes |\psi\rangle = \mathsf{Z}(-\phi' - (-1)^b\theta' - r\pi)\mathsf{E}\mathsf{Z}((-1)^b\theta')\,|+\rangle \otimes |\psi\rangle\,.$$

By performing the change of variables $\hat{\theta} = (-1)^b\theta'$, which is drawn from the same distribution, we recover the state in the original UBQC Protocol, with no correction from the Orchestrator:

$$\mathsf{Z}(-\phi' - \hat{\theta} - r\pi)\mathsf{EZ}(\hat{\theta})\,|+\rangle \otimes |\psi\rangle\,.$$

Therefore in the full protocol, requesting and applying the correction are unnecessary steps, either for correctness or security, since the states with or without these corrections are equal.

Security of QSMPC. We now prove the correctness and security of our QSMPC protocol using the composition of AC resources and protocols.

Theorem 5 (Security of Delegated Quantum SMPC). *Suppose that the Trappified Delegated Blind Computation Protocol 1 ϵ_V-constructs the Secure Delegated Quantum Computation with Classical IO Resource 1 for leak l_ρ. Then Protocol 3 ϵ_V-constructs the Quantum Secure Multi-Party Computation with Classical IO Resource 3 from an interactive Classical Secure Multi-Party Computation Resource 4 for the same leak l_ρ, against malicious coalitions that include at most the Server and $n-1$ Clients.*

Proof. This proof is very simple and works by retracing in reverse order the high-level description of the protocol in the worst case with $n-1$ malicious Clients in collusion with a malicious Server

We first use the security of the Collaborative RSP Protocol as expressed in Theorem 4 to replace each instance of this protocol with a call to the RSP Resource 2, at no security cost. The Secure Classical Channel Resources from the Clients to the Orchestrator come for free since this party is now replaced by the Classical SMPC Resource in our protocol.

We can then replace these Resources with a direct quantum communication channel between the Orchestrator and the Server, since this protocol perfectly implements the RSP Resource. We obtain as a result exactly an execution of the UBQC Protocol 4 between the Orchestrator and the Server in step 2.b.ii of Protocol 3. The whole step 2 of Protocol 3 is then exactly an execution of Protocol 1 between the Orchestrator and the Server.

We then use the fact that this protocol ϵ_V-constructs the Secure Delegated Quantum Computation with Classical IO Resource and replace it by a call to that resource with a cost of ϵ_V.

In this final stage, the Clients send their desired computation and inputs to the Orchestrator, which only forwards the concatenated input to the SDQC Resource. This Resource leaks the value l_ρ to the Server and returns the correct value to the Orchestrator if there has been no abort from the Server. The Orchestrator then sends back this output to the malicious Clients if they desire to receive it first. If there has been no abort at this stage, the Orchestrator finally transmits the output to the honest Clients as well. Therefore merging the Orchestrator – a Classical SMPC Resource – and the SDQC Resource yields exactly the behaviour of the desired QSMPC Resource between the n Clients and the Server. ■

5 Discussion

5.1 Comparison with Other QSMPC Protocols

Table 1 below gives a comparison of our protocol with the peer-to-peer protocols of [9] and [29], and with the more recent semi-delegated protocol of [1]. We note n is the number of parties, d the depth of the computation (MBQC for our paper, circuit for [29] and $\{\mathsf{T}, \mathsf{CNOT}\}$-depth for [9]), t the number of T gates, c the number of CNOT gates, C_{dist} the code distance used in [29] and η a statistical security parameter. The values below correspond to the simple case where each player has a single qubit of input.

Security guarantees. Reference [9] achieve an information-theoretic upgrade of a Classical SMPC to the quantum domain, secure against an arbitrary number of corrupted parties. On the other hand, the protocol from [1] is only computationally-secure since it relies on a Fully-Homomorphic Encryption Scheme on top of the Classical SMPC, but it is also secure against arbitrary corruptions. The protocol of [29] constructs an information-theoretically secure Quantum SMPC but suffers from an artificial blow-up in the number of

	Dulek et al. [9]	**Lipinska et al. [29]**	**Alon et al. [1]**	**This work**
Security	Stat. upgrade of CSMPC	Stat.	Comp. (FHE + CSMPC)	Stat. upgrade of CSMPC
Abort	Unanimous	Unanimous	Identifiable	Unanimous
Composability	Composable	Stand-Alone	Stand-Alone	Composable
Max adversaries	$n-1$	$\left\lfloor \frac{C_{dist}-1}{2} \right\rfloor$	$n-1$	$n-1$
Protocol nature	Symmetric	Symmetric	Semi-Delegated	Delegated
Network topology	Q and C: Complete	Q and C: Complete	Q and C: Complete	Q: Star / C: Complete
Q operations	FTQC	FTQC	FTQC	Cl: Single Qubit S: FTQC
Classical SMPC	Clifford Computation, Operations in $\mathbb{Z}_2$, CT	CT	Clifford Computation, FHE verification	Operations in $\mathbb{Z}_8$, $\mathbb{Z}_2$, CT
Rounds (C)	$\mathcal{O}(d+\eta(N+t))$	$d+2$	$\mathcal{O}(1)$	$d+3$
Rounds (Q)	Par: $\mathcal{O}(nd)$ Seq: $\mathcal{O}(n(n+t+c))$	Par: 3 (2 if C output) Seq: $\mathcal{O}(\eta^2(n+t))$	Par: $\mathcal{O}(n^4)$	Par: 1 Seq: $\mathcal{O}(\eta nd)$
Size of Q memory	Par: $\mathcal{O}(\eta^2(n+t)))$ Seq: $\mathcal{O}(\eta^2 n)$	Par: $\mathcal{O}(\eta^2 n(n+t))$ Seq: $\mathcal{O}(n^2)$	Par: $\mathcal{O}(tn^9\eta^2)$	Cl: 0 S (par): $\mathcal{O}(\eta n^2 d)$ S (seq): $\mathcal{O}(nd)$

Table 1: Comparison with [9,29,1]. Q stands for quantum and C for classical. The abbreviations Cl and S stand for Client and Server respectively. Stat. means statistical, FTQC stands for Fault-Tolerant Quantum Computer and CT for Coin-Toss.

participants and exchanged qubits.[10] The protocols of [29,1] are proven secure in the Stand-Alone Model, whereas ours and that of [9] are fully composable. On top of blindness, all protocols provide verifiability with unanimous abort apart from that of [1] which achieves the stronger notion of identifiable abort.[11]

Communication requirements. One key advantage of our protocol over the others lies in its delegated nature, where only one participant needs a full fault-tolerant quantum computer while the rest only perform very limited quantum operations, compared with the symmetric setup in [9,29] where all participant has requires fault-tolerance. The protocol of [1] can be considered semi-delegated in the sense that the brunt of the quantum computation is performed by a single player. However, all players must have the ability to perform arbitrary Cliffords on large states and cannot do so without having at their disposal a full fault-tolerant quantum computer. This is also reflected in the network topology: whereas the best performance in [9,29,1] can only be reached by using a complete quantum and classical communication graph, we only need a star graph for quantum communications. While the network topology of [9] and [29] can also be star-shaped – with

[10] It is based on error-correcting codes and the size of the code must correspond to the number of players n. The maximum number of cheaters tolerated by the protocol is the number of correctable errors $\left\lfloor \frac{C_{dist}-1}{2} \right\rfloor$, which by the quantum Singleton bound [33] is at most $\lfloor \frac{n-1}{4} \rfloor$. In their example, 7 players are required for implementing a two-party computation since the code that is used is of size 7 and corrects 1 error. This leads to a situation where 5 participants that don't have inputs nor outputs must still exchange messages and none can be malicious if one of the players with inputs is.

[11] A protocol satisfies the unanimous abort property if all honest players abort at the same time, as compared with selective aborts where the Adversary can choose which players will abort separately. On top of that, identifiable abort means that all honest players agree on the malicious party responsible for the failure of the protocol.

one player acting as a router – this would degrade their performance in terms of quantum communication rounds.

Usage of Classical Primitives. Regarding classical primitives, [29] only requires secure coin-tossing and authenticated broadcast channels (information-theoretically secure since they can rely on an honest majority). We only use our Classical SMPC to perform coin-tossing, basic string operations (array lookup) and computations in $\mathbb{Z}_8$ and $\mathbb{Z}_2$. The Classical SMPC is more complex in [9,1] since it must be able to sample uniformly at random and perform computations on the classical descriptions of arbitrary Cliffords.

Rounds of communication. We can now quantify more precisely the number of classical rounds of communication or calls to the Classical SMPC resource, quantum rounds of communication, and size of quantum memory required by each participant in the protocol. [9] calls the Classical SMPC very often: a constant number of times for each input qubit and gate in the circuit. But the most costly part is the generation of ancillary magic states (for implementing T gates via gate-teleportation), which requires $\mathcal{O}(\eta(n+t))$ invocations of the Classical SMPC. Our protocol simply uses $d+3$ calls to this Resource, 2 for setting up the state and 2 for the key-release step. This is equivalent to the classical communication requirements of [29], where they only need $d+2$ classical broadcasts per participant (with one for setting up the shared randomness and another for the state preparation). If all quantum communications are done in parallel in [29], it can be further parallelised to only require a constant number of classical broadcast rounds. The protocol of [1] uses FHE (classical and quantum) to perform the computation and consequently the number of calls to the Classicl SMPC is only constant. We note that using a classical primitive called functional encryption, where a party in possession of an evaluation key can recover the clear-text of a function of the encrypted values (and only that), would allow to attain the same result for our construction by allowing the Server to compute the next measurement angle as a function of the encrypted secrets and previous measurement results.

The protocol of [9] requires numerous rounds of quantum communication as they need to send encoded states around for the verification of inputs and T and CNOT gates. After parallelisation the total cost is $\mathcal{O}(nd)$ quantum rounds. [1] aims to remove the circuit dependency in the number of rounds, obtaining $\mathcal{O}\big(n^4\big)$ quantum rounds in the worst case in the case where the protocol is parallelised.[12]

Qubit Count and Memory Requirements. [29] seeks to optimise the quantum memory requirement of players and therefore their communication is done sequentially, yielding $\mathcal{O}\big(\eta^2(n+t)\big)$ quantum rounds. Parallelisation lowers it to 3 (or 2 for classical outputs), at a higher quantum memory cost for all parties. Our protocol is optimal as there is only a single quantum round (in the parallel case): sending to the Server all states required for the collaborative state preparation phase.

Finally, the number of qubits required by [9] during the computation phase is $\mathcal{O}(\eta(n+t))$ for each participant (they encode each of their input qubits, ancillae and magic states using $\mathcal{O}(\eta)$ qubits). However they use $\mathcal{O}\big(\eta^2(n+t)\big)$ additional qubits in the offline phase to prepare the ancillary qubits (if the quantum communications are performed in parallel). On the other hand, [29] reduces the number of qubits for each participant to $\mathcal{O}\big(n^2\big)$ for sequential quantum communication, but this blows up to $\mathcal{O}\big(\eta^2 n(n+t)\big)$ if parallelised. The construction from [1] uses a compiler that adds automatically a cost of $\mathcal{O}\big(n^2\big)$ for each base qubit. The costly double encryptions and multiple layers of traps, in particular for the magic state distillation procedure, yields a total quantum memory cost per participant of at least $\mathcal{O}\big(tn^9\eta^2\big)$ (this is a weak lower bound). In our paper the Server needs $\mathcal{O}(nd)$ qubits to perform each blind computation or test. Each qubit in these graphs is generated using n qubits via the Collaborative RSP Protocol and the computations and tests are repeated $\mathcal{O}(\eta)$ times in total, resulting in a total qubit cost of $\mathcal{O}\big(\eta n^2 d\big)$ for parallelised quantum communication but only $\mathcal{O}(nd)$ if the rounds are performed sequentially. However, the Clients can prepare these states on the fly and the Clients do not need quantum memory.

[12] They send states along a path of size n^2 in the communication graph of the parties, and remove a party if it doesn't deliver a packet before resending the states along a different path of the same size. In the worst case where there are $n-1$ malicious players which do not want to get caught cheating, they can drop $(n-1)(n-2)/2$ packets without being disconnected from the communication graph.

5.2 Impossibility of Single-Qubit Privacy Amplification on the Whole Bloch Sphere

The construction and security of Protocol 3 relies on the composition of a collaborative encryption gadget with the regular robust VBQC protocol driven by the Orchestrator.

The crucial features of the collaborative encryption are that (i) a single honest Client providing a random state from the allowed input set is enough to randomize the output of the gadget, and (ii) no information about the state provided by an honest Client leaks to the Server. These are the two properties that were shown to hold in § 3.

It can also be seen that those do not hold whenever the set of input states for the clients not only comprise the 8 $|+_\theta\rangle$ states in the $X-Y$ plane but also the computational basis states $|0\rangle, |1\rangle$. The reason is that in such case, if the central qubit is set to a computational basis state, it cannot be randomized by the states provided by other clients.

While this specific failure is contingent to the chosen transformation implemented by our gadget, we will show here that it is indeed a more generic problem that gadgets fulfilling (i-ii) have in common, thereby restricting these "gadget-assisted" approaches to verification of classical input classical output computations.

First, we give a mathematical definition of (i):

Definition 4 (Randomizing gadget). *Let P be a protocol with two Clients and one Server such that it takes a quantum state at each Client's input interface, and produces a quantum state at the Server's output interface together with a common classical bit string at each of the parties output interface.*

We say that this gadget is randomizing *whenever conditionned on the value of the common bit string, the linear maps implemented by the protocol when one of the two input states is fixed is invertible for pure input states.*

The motivation for this definition is simple: whenever one of the input is fixed, then the other one is enough to randomize the output at the Server's side. The role of the common bit string shared by all the parties at the end of the protocol is to allow the possibility of having a linear map that depends on this bit string as it is the case in our construction. As a consequence, the output state at the Server's interface might not be normalized in order to encapsulate the probability of a specific common bit string to be produced by the protocol.

The following Lemma 10 shows that for a fixed common string, there will always exist a specific state for one of the two inputs such that the map will not be invertible. This can result in one of the two following cases. Either the output is a fixed non-zero quantum state or it produces the null vector. In the former, this implies that the gadget is not able to correctly produce random states required by the VBQC protocol to be secure. In the latter, observing a specific common bit string excludes some input state for the honest Client, thereby also violating the assumptions required to obtain the security of the whole protocol.

Lemma 9. *Every two-dimensional linear subspace $V \subset \mathbb{C}^{2\times 2}$ contains at least one nonzero, singular matrix.*

Proof. Let $A, B \in V$ form a basis of V. If A or B is singular, the claim is trivial. Assume henceforth that both A and B are invertible.

For $\alpha \in \mathbb{C}$, let $C_\alpha = A + \alpha B$. Clearly, $C_\alpha \in \mathrm{span}(\{A, B\})$. Since A and B are linearly independent, $C_\alpha \neq 0$. It further holds that

$$\begin{aligned}
\det(C_\alpha) &= \det \begin{bmatrix} a_{11}+\alpha b_{11} & a_{12}+\alpha b_{12} \\ a_{21}+\alpha b_{21} & a_{22}+\alpha b_{22} \end{bmatrix} \\
&= (a_{11}+\alpha b_{11})(a_{22}+\alpha b_{22}) - (a_{12}+\alpha b_{12})(a_{21}+\alpha b_{21}) \\
&= \alpha^2(b_{11}b_{22} - b_{12}b_{21}) + \alpha(a_{11}b_{22} + b_{11}a_{22} - a_{12}b_{21} - b_{12}a_{21}) + a_{11}a_{22} - a_{12}a_{21} \\
&= \alpha^2 \det(B) + \alpha(a_{11}b_{22} + b_{11}a_{22} - a_{12}b_{21} - b_{12}a_{21}) + \det(A).
\end{aligned}$$

As $\det(B) \neq 0$, this is a polynomial of degree 2 in the variable α. By the fundamental theorem of algebra, this polynomial admits at least one complex root. ∎

Lemma 10. *There exists no linear map $\Xi : \mathbb{C}^{2\times 2} \to \mathbb{C}^2$ such that for all nonzero $v \in \mathbb{C}^2$ both $\Xi(\cdot \otimes v)$ and $\Xi(v \otimes \cdot)$ are invertible.*

Proof. Assume the existence of such a map Ξ. By the rank-nullity theorem, it holds then that

$$\dim(\mathrm{Ker}(\Xi)) = \dim(\mathbb{C}^{2\times 2}) - \dim(\mathrm{Im}(\Xi)) \geq 2.$$

By Lemma 9, there exists a rank-one matrix $C \in \mathrm{Ker}(\Xi)$. We can rewrite $C = vw^T = v \otimes w$ with nonzero vectors $v, w \in \mathbb{C}^2$. It follows that $\Xi(v \otimes w) = 0$ which contradicts the invertibility of $\Xi(\cdot \otimes w)$ and $\Xi(v \otimes \cdot)$. ■

This leads us to conclude that such gadget assisted approaches will inherently be limited to classical I/O computations.

5.3 Open Questions

This work closes a gap between the circuit and MBQC models regarding secure multi-party computations. It shows that both are able to perform the required lift from classical to quantum in a statistically secure way, in spite of the more stringent requirements the delegation imposes on what clients can do. Yet, this is only partially satisfactory as we do not consider the quantum input/output case. This specific question was considered by some of the authors. This lead to designing a protocol that was similar in spirit to the one presented here, but where the Collaborative Remote State Preparation would not only be able to prepare states in the $X - Y$ plane, but also dummy qubits. An attack on this protocol is analysed in § C. It's discovery initiated the current work using dummyless verification as a way to avoid it. Yet, we also show in § 5.2 that such approach based on Collaborative Remote State Preparation outside a single plane is not likely to succeed, thereby leaving open the question of how to perform Delegated QSMPC with quantum I/O.

Other open questions regarding SMPC in the MBQC model include the verification of sampling with possibly better than polynomial security bounds. The question of the delegation of fault-tolerant computation in the MBQC model is also a long standing open question that we believe can benefit from the theoretical tools developed in [22] and from an approach similar to the one exemplified in this work.

Finally, [30] showed how to blindly delegate quantum computations with trusted rotations, even if both state preparations and measurements are untrusted, but left open the question whether verification is possible in this setting. The difficulty of verification seems to stem from the fact that (i) their analysis concerns states in the $X-Y$ plane, but not dummy states, and (ii) the remotely prepared states are blind, but not necessarily verifiable. While this work does not overcome the second obstacle, it shows that verification is indeed possible without the remote preparation of dummy states, and therefore constitutes a step towards the solution of this open problem.

Acknowledgments. The authors would like to thank Michael Oliveira for discussions about the results from Section 5.2. TK was supported by a Leverhulme Early Career Fellowship. HO was partially funded by the Hybdrid HPC Quantum Initiative. EK, DL, and HO acknowledge the support by ANR research grant ANR-21-CE47-0014 (SecNISQ). LM is grateful for support from the grant BPI France Concours Innovation PIA3 projects DOS0148634/00 and DOS0148633/00 – Reconfigurable Optical Quantum Computing.

References

1. Alon, B., Chung, H., Chung, K.M., Huang, M.Y., Lee, Y., Shen, Y.C.: Round efficient secure multiparty quantum computation with identifiable abort (Nov 2020), https://eprint.iacr.org/2020.1464
2. Bartusek, J.: Secure quantum computation with classical communication. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography. pp. 1–30. Springer International Publishing, Cham (2021)
3. Bartusek, J., Coladangelo, A., Khurana, D., Ma, F.: On the round complexity of secure quantum computation. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021. pp. 406–435. Springer International Publishing, Cham (2021)

4. Ben-Or, M., Crepeau, C., Gottesman, D., Hassidim, A., Smith, A.: Secure multiparty quantum computation with (only) a strict honest majority. In: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science. pp. 249–260. FOCS '06, IEEE Computer Society, Washington, DC, USA (2006). https://doi.org/10.1109/FOCS.2006.68, http://dx.doi.org/10.1109/FOCS.2006.68
5. Broadbent, A., Fitzsimons, J., Kashefi, E.: Universal blind quantum computation. In: IEEE (ed.) 50th Annual IEEE Symposium on Foundations of Computer Science (2009)
6. Cramer, R., Damgrd, I.B., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing. Cambridge University Press, USA, 1st edn. (2015), https://dl.acm.org/doi/book/10.5555/2846411
7. Crépeau, C., Gottesman, D., Smith, A.: Secure multi-party quantum computation. In: Proceedings of the Thiryfourth Annual ACM Symp. on Theory of Computing. p. 643. STOC '02, ACM, New York, NY, USA (2002). https://doi.org/10.1145/509907.510000, http://doi.acm.org/10.1145/509907.510000
8. Danos, V., Kashefi, E.: Determinism in the one-way model. Phys. Rev. A **74**, 052310 (Nov 2006). https://doi.org/10.1103/PhysRevA.74.052310, http://link.aps.org/doi/10.1103/PhysRevA.74.052310
9. 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.) Advances in Cryptology – EUROCRYPT 2020. pp. 729–758. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_25
10. Dunjko, V., Fitzsimons, J.F., Portmann, C., Renner, R.: Composable security of delegated quantum computation. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology – ASIACRYPT 2014. pp. 406–425. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
11. Dupuis, F., Fehr, S., Lamontagne, P., Salvail, L.: Adaptive versus non-adaptive strategies in the quantum setting with applications. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016. pp. 33–59. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
12. Dupuis, F., Nielsen, J.B., Salvail, L.: Secure Two-Party Quantum Evaluation of Unitaries against Specious Adversaries. Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_37, http://dx.doi.org/10.1007/978-3-642-14623-7_37
13. Dupuis, F., Nielsen, J.B., Salvail, L.: Actively secure two-party evaluation of any quantum operation. In: Advances in Cryptology–CRYPTO 2012, pp. 794–811. Springer (2012)
14. Fehr, S., Katz, J., Song, F., Zhou, H.S., Zikas, V.: Feasibility and completeness of cryptographic tasks in the quantum world. In: Sahai, A. (ed.) Theory of Cryptography. pp. 281–296. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
15. Ferracin, S., Kapourniotis, T., Datta, A.: Reducing resources for verification of quantum computations. Physical Review A **98**(2), 022323 (2018)
16. Fitzsimons, J.F., Kashefi, E.: Unconditionally verifiable blind quantum computation. Phys. Rev. A **96**, 012303 (Jul 2017). https://doi.org/10.1103/PhysRevA.96.012303, https://link.aps.org/doi/10.1103/PhysRevA.96.012303
17. Gheorghiu, A., Kapourniotis, T., Kashefi, E.: Verification of quantum computation: An overview of existing approaches. Theory of Computing Systems **63**(4), 715–808 (May 2019). https://doi.org/10.1007/s00224-018-9872-3, https://doi.org/10.1007/s00224-018-9872-3
18. Hallgren, S., Smith, A., Song, F.: Classical cryptographic protocols in a quantum world. International Journal of Quantum Information **13**(04), 1550028 (2015). https://doi.org/10.1142/S0219749915500288, https://www.worldscientific.com/doi/abs/10.1142/S0219749915500288
19. Hein, M., Eisert, J., Briegel, H.J.: Multi-party entanglement in graph states (2003)
20. Houshmand, M., Houshmand, M., Tan, S.H., Fitzsimons, J.: Composable secure multi-client delegated quantum computation. arXiv preprint arXiv:1811.11929 (2018)
21. Kapourniotis, T., Dunjko, V., Kashefi, E.: On optimising quantum communication in verifiable quantum computing (2015), presented at AQIS'15 conference
22. Kapourniotis, T., Kashefi, E., Leichtle, D., Music, L., Ollivier, H.: Unifying quantum verification and error-detection: Theory and tools for optimisations. arxiv:2206.00631 (2022)
23. Kapourniotis, T., Kashefi, E., Music, L., Ollivier, H.: Delegating multi-party quantum computations vs. dishonest majority in two quantum rounds (2021)
24. Kashefi, E., Music, L., Wallden, P.: The quantum cut-and-choose technique and quantum two-party computation (2017)
25. Kashefi, E., Pappa, A.: Multiparty delegated quantum computing. Cryptography **1**(2), 1–20 (7 2017). https://doi.org/10.3390/cryptography1020012
26. Kashefi, E., Wallden, P.: Garbled quantum computation. Cryptography **1**(1), 6 (2017)
27. Kashefi, E., Wallden, P.: Optimised resource construction for verifiable quantum computation. Journal of Physics A: Mathematical and Theoretical; preprint arXiv:1510.07408 (2017), http://iopscience.iop.org/10.1088/1751-8121/aa5dac

28. Leichtle, D., Music, L., Kashefi, E., Ollivier, H.: Verifying bqp computations on noisy devices with minimal overhead. Phys. Rev. X Quantum **2**(040302) (2021)
29. Lipinska, V., Ribeiro, J., Wehner, S.: Secure multi-party quantum computation with few qubits. arXiv e-prints arXiv:2004.10486 (Apr 2020)
30. Ma, Y., Kashefi, E., Arapinis, M., Chakraborty, K., Kaplan, M.: QEnclave – a practical solution for secure quantum cloud computing. npj Quantum Information **8**(1), 128 (2022)
31. Maurer, U.: Constructive cryptography – a new paradigm for security definitions and proofs. In: Mödersheim, S., Palamidessi, C. (eds.) Theory of Security and Applications. pp. 33–56. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
32. Maurer, U., Renner, R.: Abstract cryptography. In: Innovations in Computer Science. pp. 1 – 21. Tsinghua University Press (jan 2011), https://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/14.html
33. Rains, E.M.: Nonbinary quantum codes. IEEE Transactions on Information Theory **45**(6), 1827–1832 (1999). https://doi.org/10.1109/18.782103
34. Raussendorf, R., Briegel, H.J.: A one-way quantum computer. Phys. Rev. Lett. **86**, 5188–5191 (May 2001). https://doi.org/10.1103/PhysRevLett.86.5188, http://link.aps.org/doi/10.1103/PhysRevLett.86.5188
35. Unruh, D.: Universally composable quantum multi-party computation. In: Gilbert, H. (ed.) Advances in Cryptology – EUROCRYPT 2010. pp. 486–505. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
36. Yao, A.: How to generate and exchange secrets. In: Foundations of Computer Science, 1986., 27th Annual Symposium on. pp. 162–167. IEEE (1986)

Auxiliary Supporting Material

A The Abstract Cryptography Framework

Abstract Cryptography is a framework for defining and proving the security of cryptographic protocols, first introduced in [32,31]. Its main advantage compared to so-called Stand-Alone Models such as [18] is that any system that follows the structure defined by the framework is inherently composable, in the sense that if two protocols are secure separately, the framework guarantees at an abstract level that their sequential or parallel composition is also secure. It is equivalent to the Quantum Universal Composability (Q-UC) Model of [35] if a single Adversary controls all corrupted parties – which is the case in this work. Therefore any protocol which is secure in the Q-UC model is also secure in the AC model considered here. We refer the reader to [10] for a more in-depth presentation.

In this framework, the purpose of a secure protocol π is, given a number of available resources $\mathcal{R}$, to construct a new resource – written as $\pi\mathcal{R}$. This new resource can be itself reused in a future protocol.

The actions of all honest players in a given protocol are represented as a sequence of efficient CPTP maps acting on their internal quantum registers – which may contain communication registers, both classical and quantum. An n-party quantum protocol is therefore described by $\pi = (\pi_1, \ldots, \pi_n)$ where π_j is the aforementioned sequence of efficient CPTP maps executed by party j, called the *converter* of party j.

A *resource* $\mathcal{R}$ is described as a sequence of CPTP maps with an internal state. It has input and output interfaces describing which party may exchange states with it. Some interfaces may be filtered, meaning that they are only accessible to a corrupted party.[13] It works by having the party sending it a given state at one of its input interfaces, applying the specified CPTP map after all input interfaces have been initialised and then outputting the resulting state at its output interfaces in a specified order. Classical resources are modelled by considering that the input state is measured in the computational basis upon reception and the output is a measurement result on its internal state.

In order to define the security of a protocol, we need to give a pseudo-metric on the space of resources. The security analysis then consists of considering a special type of converters called *distinguishers*. The distinguisher's aim is to discriminate between resources $\mathcal{R}_0$ and $\mathcal{R}_1$ which have the same input and output interfaces. It attaches to the inputs and outputs of one of the resources, interacting with it according to its own – possibly adaptive – strategy, and outputs a single bit indicating its guess as to which resource it had access to. Two resources are said to be indistinguishable if no distinguisher can make this guess with good probability.

Definition 5 (Indistinguishability of Resources). *Let $\epsilon(\eta)$ be a function of security parameter η and $\mathcal{R}_0$ and $\mathcal{R}_1$ be two resources with same input and output interfaces. The resources are ϵ-statistically-indistinguishable if, for all distinguishers $\mathcal{D}$, we have:*

$$\left|\Pr[b = 1 \mid b \leftarrow \mathcal{D}\mathcal{R}_0] - \Pr[b = 1 \mid b \leftarrow \mathcal{D}\mathcal{R}_1]\right| \leq \epsilon \tag{8}$$

We then write $\mathcal{R}_0 \underset{stat,\epsilon}{\approx} \mathcal{R}_1$.

The correctness of a protocol protocol π applied to resource $\mathcal{R}$ can be expressed as the indistinguishability between the resource $\pi\mathcal{R}$ and a desired target resource $\mathcal{S}$.

The security of the protocol is captured by the fact that the resources remain indistinguishable if we allow some parties to deviate in the sense that they are no longer forced to use the converters defined in the protocol but can use any other CPTP maps instead. This is done by removing the converters for those parties in Equation 8, keeping only $\pi_{M^c} = \{\pi_j\}_{j \notin M}$ where M is the set of corrupted parties. On the other side, there must exist a converter called a *simulator* which attaches to the interfaces of $\mathcal{S}$ for corrupted parties $j \in M$ and aims to reproduce the transcript of honest players interacting with the corrupted ones. The security is formalised as follows in Definition 6.

Definition 6 (Construction of Resources). *Let $\epsilon(\eta)$ be a function of security parameter η. We say that an n-party protocol π ϵ-statistically-constructs resource $\mathcal{S}$ from resource $\mathcal{R}$ against adversarial patterns $\mathsf{P} \subseteq \wp([N])$ if:*

[13] In this paper filtered input interfaces consist of single bits, set to 0 in the default, honest case.

1. *It is correct:* $\pi\mathcal{R} \underset{stat,\epsilon}{\approx} \mathcal{S}\perp$*, where* $\perp$ *filters the malicious interfaces;*
2. *It is secure for all subsets of corrupted parties in the pattern* $M \in \mathsf{P}$*: there exists a converter called simulator* σ_M *such that* $\pi_{M^c}\mathcal{R} \underset{stat,\epsilon}{\approx} \mathcal{S}\sigma_M$*.*

We can now present the General Composition Theorem (Theorem 1 from [32]).

Theorem 6 (General Composition of Resources). *Let* $\mathcal{R}$*,* $\mathcal{S}$ *and* $\mathcal{T}$ *be resources,* α*,* β *and* id *be protocols (where protocol* id *does not modify the resource it is applied to). Let* $\circ$ *and* $\mid$ *denote respectively the sequential and parallel composition of protocols and resources. Then the following implications hold:*

- *The protocols are* sequentially composable*: if* $\alpha\mathcal{R} \underset{stat,\epsilon_\alpha}{\approx} \mathcal{S}$ *and* $\beta\mathcal{S} \underset{stat,\epsilon_\beta}{\approx} \mathcal{T}$ *then* $(\beta \circ \alpha)\mathcal{R} \underset{stat,\epsilon_\alpha+\epsilon_\beta}{\approx} \mathcal{T}$
- *The protocols are* context-insensitive*: if* $\alpha\mathcal{R} \underset{stat,\epsilon_\alpha}{\approx} \mathcal{S}$ *then* $(\alpha \mid \mathsf{id})(\mathcal{R} \mid \mathcal{T}) \underset{stat,\epsilon_\alpha}{\approx} (\mathcal{S} \mid \mathcal{T})$

Combining the two properties presented above yields concurrent composability (the distinguishing advantage cumulates additively as well).

The computational versions of these definitions are obtained by quantifying over quantum polynomial time parties. Composing a statistically-secure protocol with a computationally-secure protocol is possible provided that the simulator for the statistically-secure one runs in expected polynomial time. The resulting protocol is of course only computationally-secure.

Comments on the Security Framework. First, we always consider in this work a single Adversary controlling all the corrupted parties. As explained above, it is therefore possible to instantiate all purely classical Resources using any classical protocol which is secure in the Q-UC framework of [35] with the same security guarantees. It is also possible to instantiate them with any classical UC-secure protocol whose security relies on a quantum-hard problem thanks to Theorem 18 (Quantum Lifting Theorem – Computational) from [35].

Also, it is impossible to have fairness of output distribution in the case of a dishonest majority, the malicious parties can always choose to receive their output before the honest players. This is modelled in the resources by a filtered bit f_j at each player's interface, indicating that it receives the output before others. The corrupted players can then decide to make the honest players abort before receiving their output.

B Measurement-Based Quantum Computing

The protocols in the present paper relies on Measurement-Based Quantum Computing (or MBQC). The MBQC model of computation emerged from the gate teleportation principle. It was shown in [34] that any quantum computation can be implemented by performing single-qubit measurements on a type of entangled states called graph states.

Given a graph $G = (V, E)$, and input and output vertices $I, O \subseteq V$, the corresponding graph state is generated by initialising a qubit in state $|+\rangle$ for each vertex in V and performing entangling operator CZ between qubits whose vertices are linked by an edge in E. The qubits are measured according to an order given by a function $f : O^c \to I^c$ called the flow of the computation.

We define the rotation operator around the Z axis of the Bloch sphere by an angle θ as $\mathsf{Z}(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$ and $|+_\theta\rangle = \mathsf{Z}(\theta)|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\theta}|1\rangle)$. For approximate universality, we can restrict the set of angles to $\Theta = \left\{\frac{k\pi}{4}\right\}_{k\in\{0,\dots,7\}}$ [5]. The measurement associated to an angle $\phi \in \Theta$ is given by the basis $|\pm_\phi\rangle$. We consider in this paper that this measurement is performed by rotating the state to be measured using the operation $\mathsf{Z}(-\phi)$ and then measuring in the X-basis.

Later measurement may depend on the outcomes of previous measurements. Let $\{\phi(v)\}_{v\in O^c}$ be a set of default measurement angles for non-output qubits. Let $S_X(v)$ and $S_Z(v)$ be respectively the X and Z dependency-sets for qubit v.[14] The measurement result $s(w)$ for qubit $w \in S_X(v) \cup S_Z(v)$ induces Pauli

[14] These sets are also given by the flow, see [19,8] for details.

corrections on qubit v which are equivalent to measuring qubit v with corrected angle $\phi'(v) = (-1)^{s_X(v)}\phi(v) + \pi s_Z(v)$, where $s_X(v) = \bigoplus_{w \in S_X(v)} s(w)$ and $s_Z(v) = \bigoplus_{w \in S_Z(v)} s(w)$.

The special case of classical inputs is handled by adding an angle $x(v)\pi$ to the measurement angle $\phi(v)$ of input qubit $v \in I$. Classical outputs correspond to the case where all qubits are measured.

The classical input-output computation is defined by a graph $G = (V, E)$, input and output vertices $I, O \subseteq V$, a set of default measurement angles $\{\phi(v)\}_{v \in V}$ and a flow function $f : O^c \to I^c$. To perform the computation, one generates the graph state associated to G, performs the measurements with angles $\phi'(v)$ using the default angles and the flow. The outcome is defined by bit-string $\{s(v)\}_{v \in O}$.

B.1 Universal Blind Quantum Computing

An MBQC computation can easily be delegated by a Client to a quantum Server by having the Server create the graph state and perform measurements instructed by the Client. Universal Blind Quantum Computation (or UBQC) [5] is an upgrade of delegated MBQC which guarantees that the Server does not learn anything about the computation besides the computation graph, order of measurements and position of output qubits.

This is achieved as follows. The Client hides the computation by sending rotated states $\left|+_{\theta(v)}\right\rangle$ instead of $|+\rangle$ for each qubit v. The effect of this rotation is cancelled by a corresponding rotation of the measurement angle. An additional parameter $r(v)$ adds an extra $r(v)\pi$ rotation to the measurement angle in order to hide the measurement outcome. This can be classically accounted for by the Client by setting $s(v) = r(v) \oplus b(v)$, where $s(v)$ is used as above in MBQC for corrections while $b(v)$ is the outcome returned by the Server. The measurement angle sent by the Client is then $\delta(v) = \phi'(v) + \theta(v) + r(v)\pi + x(v)\pi$, where x is its input bit-string, extended by 0 on non-input vertices.

We give the full UBQC protocol for classical inputs and outputs in Protocol 4.

Protocol 4 Classical Input-Output Universal Blind Quantum Computation

Public Information:
- $G = (V, E, I, O)$, a graph with input and output vertices I and O respectively;
- $\preceq_G$, a partial order on the set V of vertices;

Client's Inputs: A bit-string $x \in \{0,1\}^{|I|}$ and the classical description of a unitary U as default measurement angles $\{\phi(v)\}_{v \in V}$ and a flow f which induces an ordering compatible with $\preceq_G$.

The Protocol:
1. The Client, for each vertex $v \in V$, samples at random $\theta(v) \in_R \Theta$ and sends $\left|+_{\theta(v)}\right\rangle$ to the Server.
2. The Server receives the qubits one-by-one and applies the entangling operations CZ that correspond to the edges of the graph G.
3. For each qubit $v \in V$, following the partial order of the flow:
   (a) The Client samples at random $r(v) \in_R \{0,1\}$, calculates and sends an angle to the Server:
   $$\delta(v) = \phi'(v) + \theta(v) + r(v)\pi + x(v)\pi. \tag{9}$$
   (b) The Server measures in the $\left\{\left|+_{\delta(v)}\right\rangle, \left|-_{\delta(v)}\right\rangle\right\}$ basis and returns to the Client outcome $b(v)$. The Client sets $s(v) = b(v) \oplus r(v)$.
4. The Client sets the bit-string $\{s(v)\}_{v \in O}$ as its output.

C Post-Mortem of Previous Protocol

A subset of the authors of the current paper proposed an earlier protocol for QSMPC [23]. We show here the limits of the design and discuss possible paths towards fixing it.

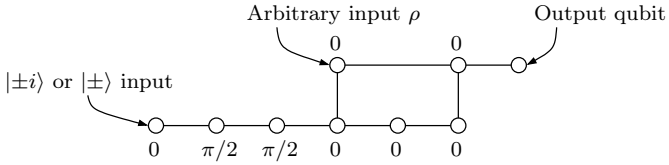
State-Selective Flipping Attack. The principle of the previous protocol was to separate the computation in two parts. The first section, which is blind only and not verifiable, is responsible for preparing the verifiable graph state from [27], i.e. a single graph state which includes traps. This requires to prepare both rotated qubits and dummies. The Collaborative RSP prepares only rotated states which must then either be transformed into dummies $\{|0\rangle, |1\rangle\}$ or left undisturbed (for computation and trap qubits). This is done with a blind computation on all these qubits, the additional qubits required for this computation being also generated using the Collaborative RSP. This is essentially a way to extend the Collaborative RSP to a bigger set of states.

The blindness of this gadget is proven in the Abstract Cryptography framework, so it would seem that it can be composed with the single-Client SDQC protocol to yield QSMPC in the same way as in the current work. However, this is not the case since we do not verify that the Server acts honestly so the final state is correct up to a global deviation. In general this deviation depends on the state that is being prepared, in particular the deviation can depend on whether the final state is a dummy or computation/trap qubit. Conscious of this, [23] exhibit a number of sufficient conditions on the computation so that this global deviation is independent of the secret state the the Clients prepare collaboratively.

These conditions are as follows:

1. The inputs in the graph of the Clients' desired computation have degree 1.
2. All measurement angles are from the set $\{k\pi/2\}_{0\leq k\leq 3}$, i.e. the computation is Clifford.
3. The graph, flow of this computation and the angles of all vertices beyond the first layer of the gadget are independent of the final desired state of the qubits.

This final condition restrains a lot the possible types of computations that can be performed in this step but [23] proposes a scheme which seems to satisfy them. We recall it here for completeness. The MBQC pattern is given by the following graph and angles.



The qubit which must be transformed is denoted ρ (upper left qubit) and the lower left qubit's state is chosen depending on whether the upper qubit should be turned into a dummy or not. We refer to [23] for details. In order to be correct, it requires an additional correction step after this computation. The correction depends on the state of the second input qubit and the measurement outcome of the last qubit in the lower line.

2nd qubit input	Outcome	Correction	Effect
$\|+_i\rangle$	0	Y	I
$\|+_i\rangle$	1	Y	I
$\|-_i\rangle$	0	I	I
$\|-_i\rangle$	1	I	I
$\|+\rangle$	0	X	H
$\|+\rangle$	1	Z	H
$\|-\rangle$	0	Z	H
$\|-\rangle$	1	X	H

Unfortunately, this correction depends on the final state. More precisely, by flipping the value reported as measurement outcome, the Server can apply an Y operation on dummy qubits and leave the rest unaffected. This flips selectively the state of dummies only, even if the server does not know that the qubit being prepared is in fact a dummy. We show in the next section how this breaks the verifiability of the protocol. The main take-away is that any correction applied after the computation does not also depend on the final state.

A potential patch was constructed using a more compact setup which seemingly satisfies all conditions. The graph that is used consists of a three vertex line for each final qubit. The first qubit in the line is

measured either with an angle $\pi/2$ for dummy vertices or 0 for other positions. The second vertex will always be measured with an angle of $\pi/2$. This is presented below in Figure 3.

(a) Measurement pattern for rotated qubits. (b) Measurement pattern for dummy qubits.

Fig. 3: DBQC measurement pattern applied to each qubit in the verifiable graph. The vertices surrounded with squares are inputs, round vertices are measured, diamond vertices are outputs. Blue vertices correspond to a measurement angle of $\pi/2$ while white vertices are measured with angle 0.

In the first case, the operation that is applied is $\mathsf{HZ}(\pi/2)\mathsf{HZ}(\pi/2) = \mathsf{X}(\pi/2)\mathsf{Z}(\pi/2)$, which has the effect of transforming the state $|+\rangle$ first into $|+_{\pi/2}\rangle$ with the Z-rotation and then into $|0\rangle$ via the X-rotation. Note that this does not correspond to a Hadamard gate since it transforms $|-\rangle$ into $-i\,|1\rangle$, but it is sufficient for our purposes. In the second case, the operation correspond to $\mathsf{X}(\pi/2)$, which has no effect when applied to a $|+\rangle$ state up to a global phase. Since all qubits are rotated $|+\rangle$ states, the rotation of the last qubit in the three vertex line graph re-encodes the state if the final state is not a dummy. This yields the full set of states required by the SDQC protocol.

Note that once again, the conditions appear to be satisfied. Also, there are no post-processing steps beyond the standard MBQC flow corrections. However, here the input states do not span the full 8 rotated states, but are always considered as $|+\rangle$ states and they are re-encrypted via the rotation of the last qubit. By applying Z on the input before the computation and the output after the computation, it is also possible to selectively flip dummies only: the two Zs will cancel out for rotated qubits, but the second Z will have no effect on dummies while the first Z will flip the dummy.

Attack from Selective Dummy Flipping. We describe here an attack on the VBQC scheme of [27], assuming that the Adversary can flip the value of dummy qubits (without affecting the computation and traps). We assume for this section knowledge about the Dotted-Triple Graph construction of [27]. Consider a line graph of two qubits and its transformation in a Dotted-Triple Graph. This graph contains two primary locations with three qubits and one added location with nine qubits.

Through the application of CZ gates to construct the graph, flipping the value of a dummy is equivalent to applying Z on all adjacent qubits. For a given qubit in the graph, the global effect is I if an even number of adjacent dummies are flipped, and Z if an odd number of adjacent dummies are flipped. As we wish to disrupt the computation but not affect traps, the key to our attack is to use the difference in the number of dummies in the neighbourhood of traps and computation qubits. Traps are only linked to dummies while a computation qubit will always have at least one other computation qubit among its neighbours. As shown in Figure 4 below, we selectively flip added vertices so that each primary vertex is linked to exactly two attacked added vertices.

In that case, since the primary trap qubits are only linked to dummies, the attack does not trigger either trap (if one of the middle qubits that is attacked is a trap, the effect of the attack on this trap is I as explained above). However, the attack may either affect two dummies linked to the primary computation qubits, in which case there is no attack since the effects cancel out, or one added dummy and the added computation qubit. Then, the effect on the added computation qubit is I but the attacked dummies will apply a Z operation on primary computation qubits on both sides of the link. If we assume fixed (but unknown) attack positions, whether this attack succeeds in modifying the computation depends only on the colouring that is used, while never triggering any trap. The probability of success is equal to 2/3: the attack succeeds

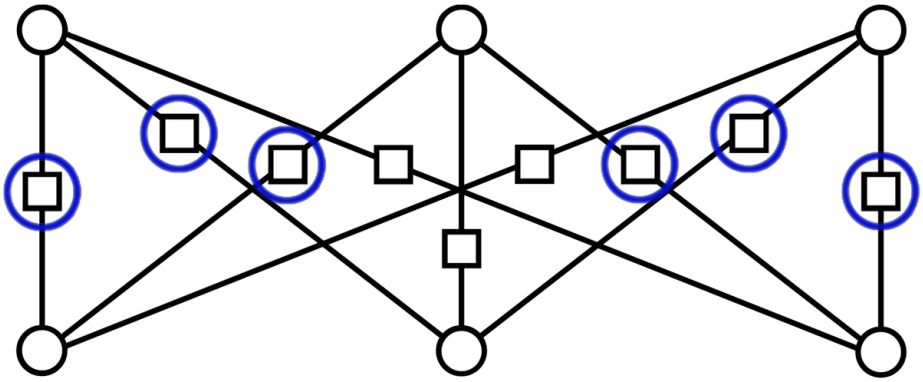
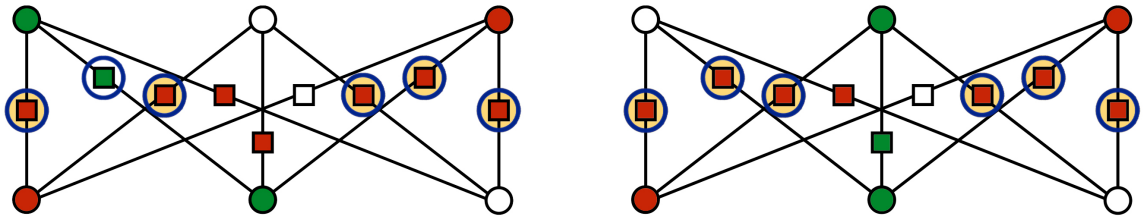


Fig. 4: Example of attack layout where each top and bottom primary qubit is attached to exactly two attacked added qubits. Qubits that have been chosen for the attack are circled in blue.

if the computational added qubit is chosen for the attack, there are 6 possible choices of attack configuration and each added qubit is left untouched by 2 out of the 6 attack configurations. We give in Figure 5 two possible colourings, ones in which the attack has no effect one the computation while the other corrupts it.



(a) Z attack on both primary computational qubits due to odd number of attacked added dummies.

(b) No attack on either primary computational qubit due to even number of attacked added dummies.

Fig. 5: Two colourings of the previous graph (computational qubits are green, traps are white and dummies are red) for the same attacked qubits but a different effect on primary computational qubits. Attacked qubits are circled in blue, which translates to an X effect on dummies (yellow-filled circle) and no effect on added computational qubits (empty circle). The primary trap qubits are never affected by the attack since they are always attached to an even number of attacked added dummies.

Extension and Take-away. Essentially, allowing an attack to depend on the nature of the qubits introduces new attacks compared to those that are possible in the plain VBQC Protocol. We have shown above that even a simple attack of this type is sufficient to break verifiability.

This attack is not specific to the construction of [27] and can also be applied to the robust SDQC Protocol of [28]. There, a trap is also always linked to dummies but computation qubits are never linked to dummies (the test and computation graphs are separated as in the current work). Applying the selective flip and apply a Z on all neighbours will corrupt the computation but leave traps unaffected.

There are two main ingredients to these attacks: (i) the possibility for the server to selectively attack qubits depending on whether they are dummies or rotated qubits, and (ii) the difference in the neighbourhoods of computation and trap qubits in terms of dummies. Regarding the first point, the sufficient conditions above are very restrictive as to what types of computations can be performed to generate a wider range of states. Together with the result from Section 5.2, this is a strong indication that starting from a Collaborative RSP for a restricted set of states and expanding it is hard to construct securely. As for the second point, if it is possible to construct an SDQC protocol in which the effect of flipping any number of dummies is the same on the tests and computations, then this attack would have no effect beyond what the SDQC protocol already protects. The current paper provides a solution to this problem by removing dummies altogether in the case of classical inputs and outputs. Other directions can be explored as well in order to construct a protocol which resits these attacks and handles quantum inputs and outputs.

We note that at no point do we break the theorem from [23] proving the sufficient conditions for constructing an MBQC gadget for blindly generating an SDQC resource state up to state-independent deviations. However, it shows that the following conditions were implicitly assumed in the proof: (i) the starting states should span the full range of rotated $|+_\theta\rangle$ states, and (ii) any post-processing should be independent of the final state.