cryptography

Article

Multiparty Delegated Quantum Computing

Elham Kashefi [1,2] and Anna Pappa [1,3,*]

[1] School of Informatics, University of Edinburgh, Edinburgh EH89AB, UK; ekashefi@gmail.com
[2] Laboratoire d'Informatique de Paris 6-CNRS, Université Pierre et Marie Curie, 75005 Paris, France
[3] Department of Physics and Astronomy, University College London, London WC1E6BT, UK
* Correspondence: a.pappa@ucl.ac.uk

Received: 29 May 2017; Accepted: 26 July 2017; Published: 30 July 2017

Abstract: Quantum computing has seen tremendous progress in the past few years. However, due to limitations in the scalability of quantum technologies, it seems that we are far from constructing universal quantum computers for everyday users. A more feasible solution is the delegation of computation to powerful quantum servers on the network. This solution was proposed in previous studies of blind quantum computation, with guarantees for both the secrecy of the input and of the computation being performed. In this work, we further develop this idea of computing over encrypted data, to propose a multiparty delegated quantum computing protocol in the measurement-based quantum computing framework. We prove the security of the protocol against a dishonest server and against dishonest clients, under the assumption of common classical cryptographic constructions.

Keywords: quantum cryptography; secure multiparty quantum computation; composability

1. Introduction

Since the early days of quantum computing and cryptography, research has been focused on finding secure communication protocols for different cryptographic tasks. However, the no-go results for bit commitment [1,2] and oblivious transfer [3] soon provided evidence that it is not possible to guarantee perfect security against any type of quantum adversary. In fact, the authors of [4] showed that any non-trivial protocol that implements a cryptographic primitive necessarily leaks information to a dishonest player. It directly follows that two-party unitaries, and by consequence multi-party ones, cannot be used without further assumptions, to securely implement cryptographic protocols. An important question that arises then is what are the cryptographic assumptions that are needed in order to achieve secure multiparty computation.

Dupuis et al. [5,6] examined the case of two-party computation and showed that in order to guarantee security, access to an AND gate for every gate that is not in the Clifford group plus a final SWAP gate between the two parties' registers are required. In the multiparty setting, Ben-Or et al. [7] rely on an honest majority assumption in order to build a verifiable quantum secret sharing scheme that is the basis of the multiparty quantum computation. From a different perspective, much research in quantum computing has been focused on delegation of computation [8–11]. This is because the current state-of-the-art is still far from constructing scalable quantum devices, and it seems that the first quantum networks will rely on the use of a limited number of powerful quantum servers. Common requirements from delegated computation schemes are that they are provably secure (the input of the computation remains private), blind (the computation performed is hidden from the server) and verifiable (honest participants can verify the correctness of the computation).

In this work, we extend previous research on quantum computing, by examining the problem of secure delegation of a multiparty quantum computation to a powerful server. More specifically, we suppose that a number of clients holding some quantum input state want to perform a unitary operation on it, but are lacking the computational abilities to do so. They would therefore like to delegate the computation to a server, while keeping their quantum inputs, quantum outputs and the performed computation secret. In the proposed protocol, the quantum operations required from the clients are limited to creating $|+\rangle$ states and applying X gates and rotations around the z-axis.

As already mentioned, in order to provide any type of security in the multiparty setting, we need to make some assumptions about the dishonest parties. In this work, we will need two assumptions. First, we will assume that the clients have secure access to classical multiparty functionalities, which we will treat as oracles. This is a common construction in classical secure multiparty computation and uses assumptions on the participating parties, like honest majority or difficulty inverting specific one-way functions. The second assumption is that a set of malicious clients cannot corrupt the server, and the other way around. We therefore prove security against two adversarial models, against a dishonest server and against a coalition of dishonest clients. To achieve this type of security, all clients contribute to some form of quantum encryption process, while at the same time, they commit to the values that they use for the encryption. Security in the more general scenario, where a server and some clients collaborate to cheat, remains an open question (however, see [12] for a relevant model with only one client, where the server is also allowed to provide an input).

A big advantage over previous protocols [7] is that quantum communication between all clients is no longer required in order to provide security against dishonest participants. By using a remote state preparation procedure, we manage to remove any quantum communication between clients, making our protocol adaptable to a client/server setting. More interestingly, the quantum communication from the clients to the server can be done in single-qubit rounds, not necessitating any quantum memory from the clients. Furthermore, all quantum communication takes place in the preparation (offline) phase, which makes the computation phase much more efficient, since it only requires classical communication.

Finally, we should note that in this work, we are focusing on proving security against malicious quantum adversaries in order to provide a simple protocol for quantum multiparty computation. As such, no guarantee is given on the correctness of the computation outcome. However, it is normal to assume that in future quantum networks, the quantum servers would want to maintain a good reputation and provide the correct outcome to the clients in case the results get cross-checked with other competing servers. In principle though, it seems possible to add verification processes in our protocol, by enforcing honest behaviour, following the work of [6,11].

2. Materials and Methods

2.1. Measurement-Based Quantum Computing

Delegated computation is commonly studied in the Measurement Base Quantum Computing (MBQC) model [8], where a computation is described by a set of measurement angles on an entangled state. A formal way to describe an MBQC computation was proposed in [13] and is usually referred to as an MBQC pattern. In the general case of quantum input and quantum output, such a pattern is defined by a set of qubits (V), a subset of input qubits (I), a subset of output qubits (O) and a sequence of measurements $\{\phi_j\}$ acting on qubits in $O^c := V \setminus O$. Due to the probabilistic nature of the measurements, these angles need to be updated according to a specific structure. This structure is described by the flow f of the underlying graph G of the entangled state. The flow is a function from measured qubits to non-input qubits along with a partial order over the nodes of the graph such that each qubit j is X-dependent on qubit $f^{-1}(j)$ and Z-dependent on qubits i for which $j \in N_G(f(i))$, where $N_G(j)$ is the set of neighbours of node j in graph G. We will denote the former set of qubits by S_j^X and the latter set of qubits by S_j^Z. If we define the outcome of a measurement on qubit i as s_i, then the new measurement angle for qubit j is:

$$\phi'_j = (-1)^{s_j^X} \phi_j + s_j^z \pi \tag{1}$$

where $s_j^X = \bigoplus_{i \in S_j^X} s_i$ and $s_j^Z = \bigoplus_{i \in S_j^Z} s_i$ (we will use this notation throughout the paper).

2.2. Multiparty Delegated Quantum Computing

In this section, we will give some necessary definitions for multiparty delegated quantum computing protocols. We will consider multiple clients $C_1, ..., C_n$ that have registers $\mathcal{C}_1, ..., \mathcal{C}_n$. To allow the computation to be done in a delegated way, we also introduce the notion of a server S, who is responsible for performing the computation of a unitary U on input ρ_{in}. The server has register $\mathcal{S}$, but no legal input to the computation. We also denote with $\mathcal{D}(\mathcal{A})$ the set of all possible quantum states (i.e., positive semi-definite operators with Trace 1) in register $\mathcal{A}$. We denote the input state (possibly entangled with the environment R) as:

$$\rho_{in} \in \mathcal{D}(\mathcal{C}_1 \otimes \cdots \otimes \mathcal{C}_n \otimes \mathcal{R}) \tag{2}$$

In what follows, we consider that the input registers of the participants also contain the classical information necessary for the computation. We denote with $\mathcal{L}(\mathcal{A})$ the set of linear mappings from $\mathcal{A}$ to itself, and we call a superoperator $\Phi : \mathcal{L}(\mathcal{A}) \to \mathcal{L}(\mathcal{B})$ that is completely positive and trace preserving a quantum operation. Finally, we will denote with $\mathbb{I}_{\mathcal{A}}$ and $\mathbf{1}_{\mathcal{A}}$ the totally mixed state and the identity operator, respectively, in register $\mathcal{A}$.

In order to analyse the properties of the protocol to follow, we will first consider an abstract system that takes as input ρ_{in} and the computation instructions for implementing U in MBQC (i.e., the measurement angles $\{\phi_j\}$) and outputs a state ρ_{out}. We will call such a resource an ideal functionality because this is what we want to implement. In order to allow the server to act dishonestly, we also allow the ideal functionality to accept input from the server, which dictates the desired deviation in the form of quantum input and classical information (Figure 1). In the case where the server is acting honestly, the ideal functionality is outputting the correct output $\rho_{out} = (U \otimes \mathbf{1}_{\mathcal{R}}) \cdot \rho_{in}$, where for the ease of use, we will write $U \cdot \rho$ instead of $U\rho U^\dagger$ each time we talk about applying a unitary operation U to a quantum state ρ.

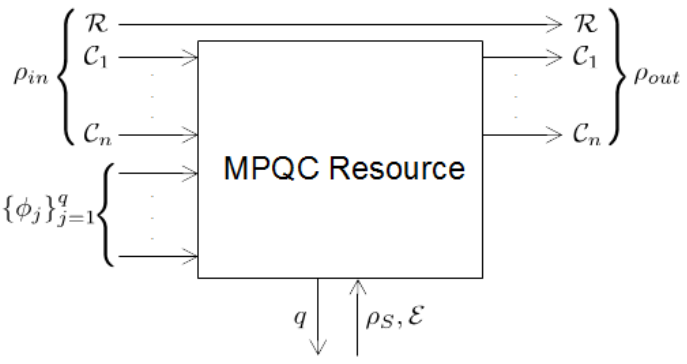

Figure 1. The ideal functionality that implements a multiparty quantum computation (MPQC), given by measurement angles $\{\phi_j\}_{j=1}^q$ on input ρ_{in}.

Definition 1 (MPQC Delegated Resource)**.** *A multiparty quantum computation (MPQC) delegated resource gets input ρ_{in} and computation instructions from n clients and leaks the size of the computation q to the server. The server can then decide to input a quantum map and a quantum input. The output of the computation is $\rho_{out} = (U \otimes \mathbf{1}_{\mathcal{R}}) \cdot \rho_{in}$ if the server does not deviate; otherwise, the output is defined by the quantum deviation on the inputs of the server and the clients.*

A protocol can in general be modelled by a sequence of local quantum operations on the participants' registers together with some oracle calls, which are joint quantum operations on the registers. Here, we will consider the delegated version of communication protocols [5–7], where n clients are delegating the computation to a server.

Definition 2. *We denote a t-step delegated protocol with oracle calls between clients C_k ($k \in [n]$) and a server S, with $\pi\mathcal{O} = (\{\pi_k\}_{k=1}^{n}, \pi_S, \mathcal{O})$. If we denote by $\mathcal{C}_k$ and $\mathcal{S}$ the registers of the clients and the server, respectively, then:*

- *Each client's strategy π_k, $k \in [n]$ consists of a set of local quantum operators $(L_1^k, \dots, L_t^k)$ such that $L_i^k : \mathcal{L}(\mathcal{C}_k) \to \mathcal{L}(\mathcal{C}_k)$ for $1 \leq i \leq t$.*
- *The server's strategy π_S consists of a set of local quantum operators $(L_1^S, \dots, L_t^S)$ such that $L_i^S : \mathcal{L}(\mathcal{S}) \to \mathcal{L}(\mathcal{S})$ for $1 \leq i \leq t$.*
- *The oracle $\mathcal{O}$ is a set of global quantum operators $\mathcal{O} = (\mathcal{O}_1, \dots, \mathcal{O}_t)$ such that $\mathcal{O}_i : \mathcal{L}(\mathcal{C}_1 \otimes \cdots \otimes \mathcal{C}_n \otimes \mathcal{S}) \to \mathcal{L}(\mathcal{C}_1 \otimes \cdots \otimes \mathcal{C}_n \otimes \mathcal{S})$ for $1 \leq i \leq t$.*

Therefore, at each step i of the protocol $\pi\mathcal{O}$, all participants apply their local operations, and they also jointly apply a global $\mathcal{O}_i$. If we define the joint quantum operation $L_i = L_i^1 \otimes \cdots \otimes L_i^n \otimes L_i^S$, then the quantum state at step i when the input state ρ_{in} is defined by Equation (2) is:

$$\rho_i(\rho_{in}) := (\mathcal{O}_i L_i \dots \mathcal{O}_1 L_1) \cdot \rho_{in} \tag{3}$$

At the end of the protocol, the output will be:

$$\pi\mathcal{O}(\rho_{in}) = (\mathcal{O}_t L_t \dots \mathcal{O}_1 L_1) \cdot \rho_{in} \tag{4}$$

A commonly-used oracle is the communication oracle that transmits information from one participant to the other, by just moving the state between registers. Due to no-cloning, however, when the information transmitted is a quantum state, we require that the oracle also erases the transmitted information from the original party's register.

Another oracle that we will use in our protocol is what we call the computation oracle, which can be thought of as a classical ideal functionality (i.e., a box that takes inputs from all parties and outputs the correct outputs of the functionality on the specific inputs). All classical multiparty computation in this work will be done with the help of such a computation oracle. Under standard cryptographic assumptions, there exist classical protocols for building such oracles that emulate in a composable way any classical multiparty functionality. In [14], this is shown for an honest majority of participants, given that they share pairwise secure classical channels and a physical broadcast channel. Since the last physical requirement is quite strong and constructions of broadcast channels in a non-physical way are usually based on authenticated Byzantine agreement, which is non-composable, we are led to relax the requirement of the honest majority to either having more than 2/3 of honest participants or to allow them access to other cryptographic constructions like oblivious transfer [15,16]. The quantum lifting theorem of Unruh [17] states that if there exists such a construction that is secure against classical adversaries, then the same also holds against quantum adversaries, therefore allowing us to replace any classical multiparty computation in a quantum protocol, by using the computation oracle. This result was proven in the universal composability framework, but it also holds in the Abstract Cryptography (AC) framework that we will use in this paper, since the two are equivalent when there is only one malicious participant (i.e., we need to build one global simulator).

2.2.1. Properties

In the following section, we will present a delegated protocol $\pi\mathcal{O}$ and claim that it emulates the MPQC resource of Figure 1. We can evaluate how well this is done by measuring how well a global distinguisher can understand whether it is interacting with one system or the other. This is quantified by using a pseudo-metric on the space of the resources that is often referred to as the distinguishing advantage of the distinguisher. The reason why a global distinguisher is used to quantify security instead of an adversary is composability. In what follows, we will use the AC composable framework [18] to model the properties of our protocol and prove its indistinguishability from the ideal functionality of Figure 1.

We will deal with three distinct cases of malicious behaviour: (a) when all clients and the server are honest, (b) when the server is dishonest and (c) when a subset $\mathcal{D}$ of clients is dishonest. For Case (a) where everyone acts honestly, we consider that the MPQC resource $\mathcal{M}$ has a filter $\perp$ blocking all dishonest behaviour, and therefore, we want to prove that:

$$d(\pi\mathcal{O}, \mathcal{M}_{\perp}) \leq \epsilon \tag{5}$$

In Case (b) when the server is acting dishonestly, we want that:

$$d(\pi_1 \ldots \pi_n \mathcal{O}, \sigma_S \mathcal{M}) \leq \epsilon \tag{6}$$

where σ_S is a simulator for the dishonest server. Finally, for Case (c), when a subset $\mathcal{D}$ of clients is dishonest, we want to prove that:

$$d(\pi_{\mathcal{H}} \mathcal{O} \pi_S, \sigma_{\mathcal{D}} \mathcal{M}_{\perp}) \leq \epsilon \tag{7}$$

where $\mathcal{H}$ is the set of honest clients and $\sigma_{\mathcal{D}}$ is a simulator for the dishonest clients.

In the above equations, the filter $\perp$ blocks input from the server's side, since the MPQC resource of Definition 1 accepts deviated behaviour only from the server. The clients, on the other hand, have the liberty to choose the quantum state that they will give as input to the MPQC resource. However, this does not mean that the clients behave honestly during the protocol. We will see that the protocol "enforces" honest behaviour on clients by asking them to secretly share their classical values in a verifiable way, in order to commit to using the same values during the protocol. This is done using Verifiable Secret Sharing (VSS) schemes, where a dealer wants to share their secret information with all parties, in such a way that a group of honest parties can reconstruct the sharing without the help of the dealer. There are two stages in a VSS protocol: the sharing stage, where the dealer and the parties exchange messages depending on the dealer's secret and the parties' chosen randomness, and the reconstruction stage, where the parties apply a function on their received messages in order to reconstruct the dealer's secret. VSS schemes can be viewed as a multiparty computation, and we can therefore use a computation oracle under the same cryptographic assumptions as previously discussed (see [19,20] for examples of VSS based on different constructions).

The distinguishing advantage of Equations (5)–(7) can reduce to simple measures of distance between the states that a distinguisher sees, when interacting with the real and ideal system. For example if the outputs of the resources are classical strings, then a distinguisher will be given strings sampled from either the probability distribution produced by the ideal or the real resource. He/she then needs to decide from which one the strings were sampled; therefore, the distinguishing advantage is equal to the total variation distance between the two probability distributions. If the outputs of the resources are quantum states, then the distinguishing advantage is given by the Helstrom measurement, which depends on the trace distance of the states of the two systems.

3. The Protocol

In this section, we propose a cryptographic protocol that constructs an MPQC resource using quantum and classical communication between the n clients and the server. We suppose that the clients want to perform a unitary U on their quantum inputs, translated in an MBQC pattern on a brickwork state using measurement angles $\{\phi_j\}_{j=1}^{q}$, where $q = |O^c|$. For simplicity, we consider that each client C_k ($k \in [n]$) has one qubit as input and one qubit as output, but it is easy to generalize to any other case. We will use the following labelling: client C_k has as input qubit "k" and as output qubit "$q + k$", while the first qubit of the second column in Figure 2 has label "$n + 1$", the last one in the second column "$2n$", etc.

We want to guarantee that the private data of the clients remain secret during the protocol. Here, each client's data consist of the quantum input and output, while we consider that the measurement angles are not known to the server (they can be known to all clients or to a specific client that delegates the computation). The protocol first starts with a process named "remote state preparation" [21]. The clients send quantum states to the server, who then entangles them and measures all but one. In the case where one of the clients has a quantum input, he/she sends that quantum state one-time padded to the server, while the rest of the clients send Z-rotated $|+\rangle$ states to the server (Algorithm 1, Figure 3). In the case of the extra "operational" qubits in $O^c \setminus I$, all clients send rotated $|+\rangle$ states to the server (Algorithm 2, Figure 4). In this way, the clients remotely prepare quantum states at the server's register that are "encrypted" using secret data from all of them, without having to communicate quantum states to each other.

However, since each client is supposed to only choose their own quantum input, and not affect the input of the other clients, the protocol should ask the clients to commit to using the same classical values for the duration of the protocol. This is done by using a VSS scheme each time a classical value is chosen by a client. In the case of the "remote state preparation" process, each time a client sends a Z-rotated $|+\rangle$ state, that rotation needs to be corrected at a later point in the protocol. In order to ensure that the "reverse" rotation is used later, the clients send many copies of randomly Z-rotated $|+\rangle$ states (Algorithm 3) and commit (via VSS) to the rotations used. They then get tested by the server and the rest of the clients on the correctness of the committed values. A similar commitment takes place for the quantum one-time pad that each client performs on their quantum input, since the classical values used affect the measurement angles of consecutive layers of computation.

At the end of the "remote state preparation" phase, the server entangles the non-measured states in a universal graph state (for example, in the brickwork state of Figure 2 [11]). Since the proposed protocol uses MBQC to compute the desired functionality, there is an unavoidable dependency between measurement angles of the qubits in different layers of computation. This means that the clients need to securely communicate between them and with the server, in order to jointly compute the updated measurement angles, taking into account the necessary corrections from the previous measurements and the dependency sets of each qubit. This procedure is purely classical and uses VSS schemes and a computation oracle to calculate the necessary values at each step of the protocol and to ensure that the clients behave honestly.

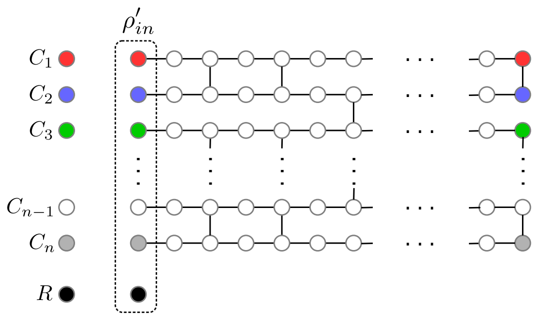

Figure 2. The brickwork state with the encrypted quantum input in the first layer of computation. The colours of the qubits denote their origin, while the encrypted input state can also be entangled with the environment R. During the computation, all qubits will be measured except the last layer.

Algorithm 1 State preparation for $j \in I$.

```
server stores states received from clients $C_k$ to distinct registers $\mathcal{S}_k \subset \mathcal{S}$ $(k = 1, \dots, n)$;
for $k = 1, \dots, n-1$
    if $k = j$ then
        break;
    if $k = n-1$ and $j = n$ then
        break;
    if $k = j-1$, then
        CNOT on $\mathcal{S}_k \otimes \mathcal{S}_{k+2}$;
    else
        CNOT on $\mathcal{S}_k \otimes \mathcal{S}_{k+1}$;
    end;
    measure state in $\mathcal{S}_k$ and get outcome $t_j^k$;
end;
if $j = n$ then
    CNOT on $\mathcal{S}_{n-1} \otimes \mathcal{S}_n$;
    measure state in $\mathcal{S}_{n-1}$ and get outcome $t_n^{n-1}$;
else
    CNOT on $(\mathcal{S}_n \otimes \mathcal{S}_j)$;
    measure state in $\mathcal{S}_n$ and get outcome $t_j^n$;
end;
```

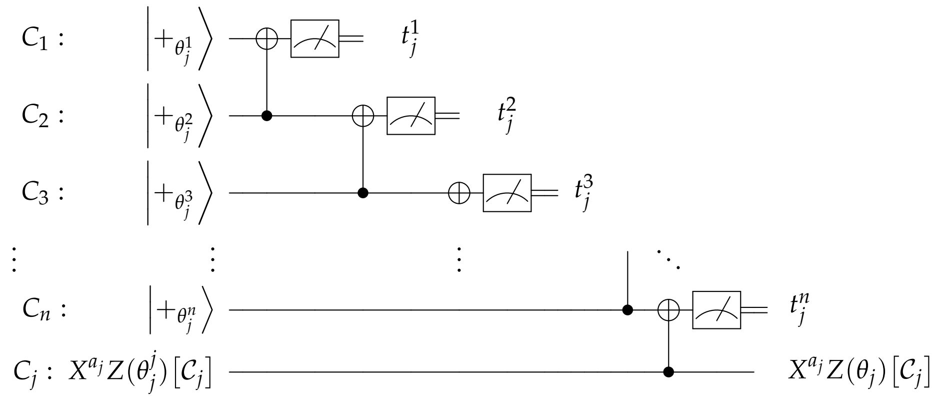
C_1 :
$\left|+_{\theta_j^1}\right\rangle$
t_j^1
C_2 :
$\left|+_{\theta_j^2}\right\rangle$
t_j^2
C_3 :
$\left|+_{\theta_j^3}\right\rangle$
t_j^3
C_n :
$\left|+_{\theta_j^n}\right\rangle$
t_j^n
C_j : $X^{a_j} Z(\theta_j^j)[\mathcal{C}_j]$
$X^{a_j} Z(\theta_j)[\mathcal{C}_j]$

Figure 3. Remote state preparation with quantum input (Algorithm 1). Client C_j performs a one-time pad on his/her register $\mathcal{C}_j$, and the result of the circuit remains one-time padded, where $\theta_j = \theta_j^j + \sum_{k=1, k \neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \theta_j^k$.

Algorithm 2 (State preparation for $j \in O^c \setminus I$)

```
server stores states received from clients $C_k$ to distinct registers $\mathcal{S}_k \subset \mathcal{S}$ $(k = 1, \dots, n)$;
for $k = 1, \dots, n-1$
    CNOT on $\mathcal{S}_k \otimes \mathcal{S}_{k+1}$;
    measure state in $\mathcal{S}_k$ and get outcome $t_j^k$;
end;
```

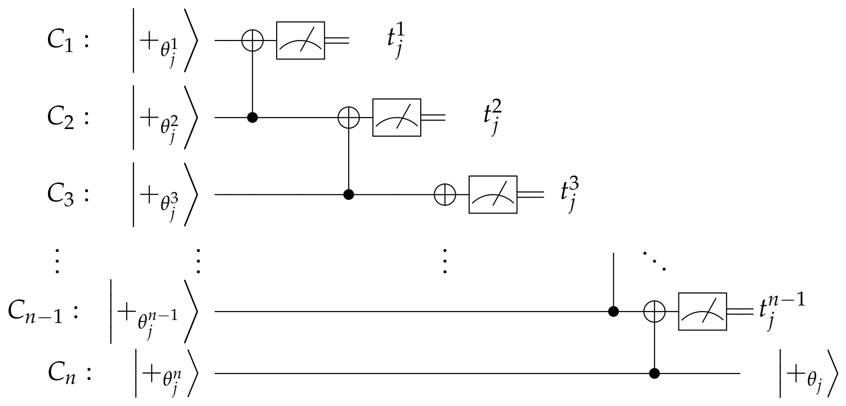
C_1 :
$\left|+_{\theta_j^1}\right\rangle$
t_j^1
C_2 :
$\left|+_{\theta_j^2}\right\rangle$
t_j^2
C_3 :
$\left|+_{\theta_j^3}\right\rangle$
t_j^3
C_{n-1} :
$\left|+_{\theta_j^{n-1}}\right\rangle$
t_j^{n-1}
C_n :
$\left|+_{\theta_j^n}\right\rangle$
$\left|+_{\theta_j}\right\rangle$

Figure 4. Remote state preparation without quantum input (Algorithm 2), where $\theta_j = \theta_j^n + \sum_{k=1}^{n-1}(-1)^{\bigoplus_{i=k}^{n-1} t_j^i}\theta_j^k$.

Finally, in the output phase, each output qubit $j \in O$ is naturally encrypted due to the corrections propagated during the computation. Qubit j is sent to the legitimate recipient C_{j-q}, while the operation that is needed to decrypt it is the $X^{s_j^X} Z^{s_j^Z}$. The classical values necessary to compute s_j^X and s_j^Z are then computed from the secret shares of all clients and sent to client C_{j-q}, who applies the necessary quantum operation.

Algorithm 3 (Enforcing honest behaviour for client C_k).

1. Client C_k sends m qubits $\left|+_{\theta_i^k}\right\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\theta_i^k}|1\rangle)$ to the server and secretly shares the values $\{\theta_i^k\}_{i=1}^m$ with all clients, using a VSS scheme.
2. The server requests the shared values from the clients for all, but one qubit, and measures in the reconstructed bases. If the bases agree with the results of the measurements, then with high probability, the remaining state is correctly formed in relation to the shared angle.

4. Analysis of the Protocol

4.1. Correctness

Theorem 1. *Algorithm 4 emulates the filtered ideal resource $\mathcal{M}_\perp$ of Figure 1.*

The validity of Theorem 1 comes directly from the correctness of the individual circuits implementing Algorithms 1 and 2, as well as the propagation of Z and X corrections through the flow of the computation. A detailed proof is given in the Appendix that shows that:

$$d(\pi\mathcal{O}, \mathcal{M}_\perp) = 0$$

therefore, a distinguisher cannot tell the difference between the real communication protocol and an interaction with the ideal MPQC resource when all participants are honest.

Algorithm 4 Multiparty quantum computing protocol.

- A quantum input ρ_{in} and measurement angles $\{\phi_j\}_{j=1}^q$ for qubits $j \in O^c$.

Preparation phase

quantum input: For $j \in I$

1. Client C_j applies a one-time pad $X^{a_j} Z(\theta_j^j)$ to his/her qubit, where $a_j \in_R \{0,1\}$ and $\theta_j^j \in_R \{l\pi/4\}_{l=0}^7$ and sends it to the server. He/she secretly shares the values a_j and θ_j^j with the other clients.
2. Each client $C_k (k \neq j)$, runs Algorithm 3 with the server. If all clients pass the test, the server at the end has $n-1$ states $\left|+_{\theta_j^k}\right\rangle = \frac{1}{\sqrt{2}}\left(\,|0\rangle + e^{i\theta_j^k}\,|1\rangle\,\right)$ for $k \neq j$.
3. The server runs Algorithm 1 and announces outcome vector $\mathbf{t}_j$.

At this point the server has the state $\rho'_{in} = \left(X^{a_1} Z(\theta_1) \otimes \cdots \otimes X^{a_n} Z(\theta_n) \otimes \mathbf{1}_{\mathcal{R}}\right) \cdot \rho_{in}$, where

$$\theta_j = \theta_j^j + \sum_{k=1,k\neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \theta_j^k \tag{8}$$

non-output/non-input qubits: For $j \in O^c \setminus I$

4. All clients C_k, $k \in [n]$ run Algorithm 3 with the server. If all clients pass the test, the server at the end has n states $\left|+_{\theta_j^k}\right\rangle = \frac{1}{\sqrt{2}}\left(\,|0\rangle + e^{i\theta_j^k}\,|1\rangle\,\right)$ for $k = 1, \ldots, n$.
5. The server runs Algorithm 2 getting outcome vector $\mathbf{t}_j$. He/she ends up with the state $\left|+_{\theta_j}\right\rangle$, where:

$$\theta_j = \theta_j^n + \sum_{k=1}^{n-1} (-1)^{\bigoplus_{i=k}^{n-1} t_j^i} \theta_j^k \tag{9}$$

output qubits: For $j \in O$, the server prepares $|+\rangle$ states.
graph state: The server entangles the $n+q$ qubits to a brickwork state by applying ctrl-Z gates.

Computation phase

non-output qubits: For $j \in O^c$

1. All clients C_k, $k = 1, \ldots, n$ choose random $r_j^k \in \{0,1\}$, which they secretly share with the other clients. Then, using a computation oracle, they compute the measurement angle of qubit j:

$$\delta_j := \phi'_j + \pi r_j + \theta_j \tag{10}$$

   where undefined values are equal to zero, or otherwise:

   - $\phi'_j = (-1)^{a_j + s_j^X} \phi_j + s_j^Z \pi + a_{f^{-1}(j)} \pi$.
   - $r_j = \bigoplus_{k=1}^n r_j^k$.
   - $s_i = b_i \oplus r_i$, for $i \leq j$.
2. The server receives δ_j and measures qubit j in basis $\{\left|+_{\delta_j}\right\rangle, \left|-_{\delta_j}\right\rangle\}$, getting result b_j. He/she announces b_j to the clients.

output qubits: For $j \in O$, the server sends the "encrypted" quantum state to client C_{j-q}. All participants jointly compute s_j^X and s_j^Z and send it to client C_{j-q}, who applies operation $Z^{s_j^Z} X^{s_j^X}$ to retrieve the actual quantum output.

4.2. Malicious Server

The proof of security against a malicious server that is allowed to deviate from the protocol, by applying operations on the data he/she receives, is based on quantum teleportation. As observed in [22], for all of the quantum states sent to the server in Algorithm 4, there exists an equivalent circuit that sends half an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ to the server and then 'teleports' to them via measurement of the quantum input. It is then easy to build a simulator for the server σ_S, which creates EPR pairs and sends half to the server and half to an MPQC resource that performs the delayed measurement for the teleportation.

Theorem 2. *Algorithm 4 is secure against a malicious server.*

Proof. In order to show that Algorithm 4 is secure against a malicious server, we will argue that Equation (6) holds for a simulator and an MPQC resource running Algorithms 5 and 6, respectively. We want to be sure that the two resources $\pi_1 \ldots \pi_n \mathcal{O}$ and $\sigma_S \mathcal{M}$ implement the same map, in other words that the outputs are indistinguishable. The equivalence comes from the fact that every time there is a quantum message sent to the server, there is an equivalent circuit based on teleportation that uses an EPR pair to transmit the same quantum state to the server.

More specifically, in Step 1 of Algorithm 4, client C_j can equivalently create an EPR pair, entangle it to their quantum input and then measure the input wire using angle θ_j^j and the EPR-half in the computational basis. The unmeasured EPR-half now contains the information of the input wire (up to corrections depending on the measurement outcomes of the other two wires). Due to no-signalling, the two measurements can also be delayed till a later stage (for example till after the server has sent some reply). This is very convenient when deciding on what angles δ_j to send to the server in Step 1 of the computation phase; since the angle θ_j can be decided later, angle δ_j can be chosen at random and then corrected by the measurement done on the entangled state (i.e., instead of choosing θ_j at random and fixing δ_j, it is equivalent to doing it the other way around).

From this, it is easy to break the procedure into two processes (see Figure 5). The first one creates the EPR pair, sends half to the server, chooses random measurement angles δ_j to send to the server and receives messages from them. The second entangles the other EPR half with the quantum input and chooses their measurement angles based on δ_j, the measurement pattern $\{\phi_j\}$ and the information received from the server. If we now name the first process the simulator σ_S and the second process the MPQC resource $\mathcal{M}$, we can see that σ_S has at no point access to any information on either the quantum input of the clients or the measurement pattern that is being implemented.

The same holds for any quantum message sent from any client to the server: there is an equivalent circuit based on teleportation that allows the client to delay the choice of measurement angle θ_j till after receiving information from the server. For a detailed proof based on a series of equivalent protocols that show the full transition from Algorithm 4 to Algorithms 5 and 6, see Appendix A. $\square$

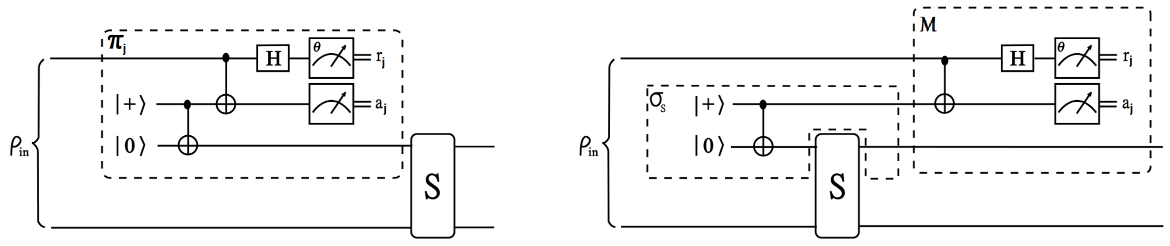

Figure 5. The equivalence between the circuit of client C_j sending his/her encrypted quantum input in Algorithm 4 and the delayed measurement circuits of the simulator and the MPQC resource of Algorithms 5 and 6 for the same action (see also [22]).

Algorithm 5 Simulator for the server.

```
non-output qubits: For $j \in O^c$
    1. $\sigma_S$ creates an EPR pair and sends one half of it to the server.
    2. $\sigma_S$ runs Algorithm 3 on behalf of the clients $C_k,\ k \neq j$ when $j \in I$ and of the clients
       $C_k,\ k \in [n-1]$ for $j \in O^c \setminus I$ sending to the server half EPR pairs and always accepting.
    3. $\sigma_S$ receives vector $\mathbf{t}_j$.
    4. $\sigma_S$ sends $\delta_j \in_R \{l\pi/4\}_{l=1}^{7}$ to the server and receives a reply $b_j \in \{0,1\}$.
output qubits: For $j \in O$
    1. $\sigma_S$ receives $n$ qubits from the server.
    2. $\sigma_S$ sends the other halves of the EPR pairs, the received quantum states, as well as $\delta_j$, $b_j$
       and $\mathbf{t}_j$ for $j = 1, \dots, q$, to the MPQC resource.
```

Algorithm 6 MPQC resource.

1. The resource receives the n qubits of ρ_{in} from the clients and all the information from σ_S.
2. For $j \in I$: the resource performs a CNOT on the corresponding EPR half with the input qubit as control and measures the EPR half in the computational basis, getting result a_j. It chooses random measurement angles $\hat{\theta}_j^k$ for the qubits coming from clients $C_k, k \neq j$, sets:

$$\hat{\theta}_j^j := \delta_j - \phi'_j - \sum_{k=1,k\neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \hat{\theta}_j^k$$

   and measures the corresponding qubits. For $j \in O^c \setminus I$, it chooses random measurement angles $\hat{\theta}_j^k$ for clients $C_k, k \in [n-1]$, sets:

$$\hat{\theta}_j^n := \delta_j - \phi'_j - \sum_{k=1}^{n-1} (-1)^{\bigoplus_{i=k}^{n-1} t_j^i} \hat{\theta}_j^k$$

   and measures the corresponding qubits. In the computation of the angles, undefined values are equal to zero, and:

   - $\phi'_j = (-1)^{a_j + s_j^X} \phi_j + s_j^Z \pi + a_{f^{-1}(j)} \pi$.
   - $r_j = \bigoplus_{k=1}^{n} r_j^k$.
   - $s_i = b_i \oplus r_i$, for $i \leq j$.
3. For $j \in O$: the resource performs corrections $Z^{s_j^Z} X^{s_j^X}$ on the remaining qubits.

We have proven that a server does not learn anything about the inputs of the clients, since the protocol emulates an ideal MPQC resource and a simulator that reproduces the view of the malicious server without having access to the clients' input. From this result, we prove at the same time that the computation is done in a blind way, meaning that the server does not know what computation he/she is performing or equivalently that the measurement pattern (i.e., the angles ϕ_j) remains hidden from him/her.

Corollary 1. *The protocol is blind against a malicious server.*

The proof of blindness follows directly from Theorem 2, since in the simulated model, the server interacts only with the simulator σ_S, which just gives input to the MPQC resource and therefore has no access to the measurement angles during the computation.

4.3. Malicious Clients

Theorem 3. *The protocol is secure against a coalition of malicious clients.*

Proof. Algorithm 7 presents a simulator σ_C that receives communication from a malicious coalition of clients on its external interface. For the ease of use, we will consider one malicious client C_c with one input qubit, but this can easily be extended by thinking of all malicious clients as one client that has multiple input qubits. It is straightforward to see that Equation (7) holds, since the malicious clients never receive quantum information from the other clients, and the only information they share is the one used in the computation oracles that are implemented using secure classical multiparty computation protocols. The quantum outcome they receive is the correct outcome of an honest server, encoded by some information (r_j) that is chosen by the malicious clients in a previous step. □

Algorithm 7 Simulator for clients.

```
1. $\sigma_C$ receives a quantum state from client $C_c$ as well as the secret shares of $a_j$ and $\theta_j^j$.
2. For all other nodes of the brickwork state, $\sigma_C$ runs Algorithm 3 with client $C_c$ and aborts
   if the secret shares of the classical values do not all match the measurement outcomes
   of the quantum states.
3. For $j \in O^c$, $\sigma_C$ receives the secret shares of the randomness $r_j^c$, chosen by the client $C_c$ and
   interacts according to the communication protocol simulating the oracle of computing $\delta_j$,
   choosing uniformly at random the value of $\delta_j$. $\sigma_C$ also replies with random $b_j$.
4. $\sigma_C$ undoes rotation $X^{a_c} Z(\theta_c^c)$ on the input qubit of $C_c$, inputs it to the Ideal MPQC resource
   and gets back the output corresponding to $C_c$.
5. Finally, $\sigma_C$ rotates the output qubit $j \in O$ corresponding to client $C_c$, applying the operation
   $Z^{s_j^Z} X^{s_j^X}$, and participates in the computation protocol to compute $s_j^X$ and $s_j^Z$ with the previously
   sent and shared values $b_j$ and $r_j$.
```

5. Conclusions

In this work, we have presented a quantum multiparty delegated protocol that provides security for clients with limited quantum abilities, therefore extending previous results on two-party [5] and multiparty [7] computation while using recent work on delegated blind computing [10,11,22]. Our protocol requires no quantum memory, entangling operations or measurement devices for the clients, only the ability to perform X gates and Z rotations. We prove security against a dishonest server or a coalition of malicious clients; it remains to study whether the proposed protocol remains secure against a dishonest coalition between clients and the server or if there is an unavoidable leakage of information. One equivalent way of studying this problem would be by extending the results of [12] in the multiparty setting, where both the parties and the server have inputs in the computation. An even more interesting question is whether we can enhance our protocol to include verifiability in a similar way as is done in [11].

The specific protocol presented here uses the measurement-based quantum computing framework, to extend the delegated blind protocol of [10] into the multiparty setting. As such, it inherits the key advantage of using MBQC over gate teleportation approaches; once the preparation phase is finished (all qubits are sent to the server and entangled in a graph state), the rest of the communication is classical. The scheme has low round complexity, both quantum and classical. The quantum communication required is linear to the number of elementary gates required to decompose the unitary that we want to apply (see Appendix of [10]). Concerning classical communication, it is not straightforward to compute the necessary bits of communication that are required to perform the VSS scheme and the computation of the angles, since they depend on the cryptographic assumptions that we make (e.g., the number of malicious parties, physical infrastructure); however, we consider that classical communication is less expensive than quantum communication, and since it seems to be growing at a worse polynomial with the parameters of the protocol [23], our scheme remains efficient. Our scheme could also be adapted to any blind computing model, for example the measurement-only model [24], since as mentioned in [22], all protocols with one-way communication from the server to a client are inherently secure due to no-signalling. We have also assumed that the clients choose to act passively maliciously, since any active dishonest activity would be detected with high probability; however, a quantitative proof of security, assuming more extensive attacks from the side of the clients would be a natural extension of this work.

Finally, a similar approach to ours has been explored for two-party computation, which uses recent advances in classical Fully Homomorphic Encryption (FHE). In [25] and in follow-up work [26], it is shown how to evaluate quantum circuits using quantum FHE; it would be very interesting to see how they can be adapted in the case of multiple parties and whether the computational and communication requirements are different from our work.

Acknowledgments: This work was supported by Grants EP/N003829/1 and EP/M013243/1 from the U.K. Engineering and Physical Sciences Research Council (EPSRC) and by the Marie Sklodowska-Curie Grant Agreement No. 705194 from the European Union's Horizon 2020 Research and Innovation program. Part of this work was done while Anna Pappa was visiting Telecom ParisTech, sponsored by a mobility grant from the Scottish Informatics and Computer Science Alliance (SICSA). The authors would also like to thank Petros Wallden and Theodoros Kapourniotis for many useful discussions.

Author Contributions: Both authors contributed to developing the theory and writing the paper.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

We first give the complete proof of Theorem 1.

Proof. We want to prove that Equation (5) holds. First, we will prove that Equations (8) and (9) are correct. We will start with Equation (9) for a server that receives n qubits $\left|+_{\theta_j^k}\right\rangle$ for $k = 1, \ldots, n$.

We will show the result by induction to the number of qubits received. For the case of two qubits $\left|+_{\theta_j^1}\right\rangle$ and $\left|+_{\theta_j^2}\right\rangle$, the server performs a CNOT operation on them (with control wire the second one). The resulting state is:

$$|0\rangle \left|+_{\theta_j^2+\theta_j^1}\right\rangle + e^{i\theta_j^1} |1\rangle \left|+_{\theta_j^2-\theta_j^1}\right\rangle$$

When the server measures the first qubit, he/she sets outcome bit $t_j^1 = 0$ when the observed state is $|0\rangle$ and $t_j^1 = 1$ when it is $|1\rangle$. Therefore, the resulting state is $\left|+_{\tilde{\theta}_j^2}\right\rangle$, where:

$$\tilde{\theta}_j^2 = \theta_j^2 + (-1)^{t_j^1} \theta_j^1$$

Therefore, Equation (9) holds for $n = 2$. Now, we will assume that the claim holds for $n - 1$, and we will prove it for n. After measurement of the qubit $n - 2$, the state of qubit $n - 1$ is $\left|+_{\tilde{\theta}_j^{n-1}}\right\rangle$, where:

$$\tilde{\theta}_j^{n-1} = \theta_j^{n-1} + \sum_{k=1}^{n-2} (-1)^{\oplus_{i=k}^{n-2} t_j^i} \theta_j^k$$

The server performs a CNOT on qubits $n - 1$ and n, resulting in the state:

$$|0\rangle \left|+_{\theta_j^n+\tilde{\theta}_j^{n-1}}\right\rangle + e^{i\tilde{\theta}_j^{n-1}} |1\rangle \left|+_{\theta_j^n-\tilde{\theta}_j^{n-1}}\right\rangle$$

The state after the measurement of the qubit $n-1$ is $\left|+_{\theta_j}\right\rangle$, where:

$$\begin{aligned}\theta_j &= \theta_j^n + (-1)^{t_j^{n-1}}\tilde{\theta}_j^{n-1} \\ &= \theta_j^n + (-1)^{t_j^{n-1}}(\theta_j^{n-1} + \sum_{k=1}^{n-2}(-1)^{\oplus_{i=k}^{n-2} t_j^i}\theta_j^k) \\ &= \theta_j^n + \sum_{k=1}^{n-1}(-1)^{\oplus_{i=k}^{n-1} t_j^i}\theta_j^k\end{aligned}$$

We have therefore proven Equation (9). What remains in order to prove Equation (8) is to see what happens when the server entangles the one-padded quantum input of client C_j with the states $\left|+_{\theta_j^k}\right\rangle$ of the rest of the clients $C_k,\ k \neq j$. If the server follows Sub-algorithm 1, it first entangles the rotated qubits of the clients $C_k,\ k \neq j$ and measures all but the last, creating a state $\left|+_{\tilde{\theta}_j}\right\rangle$. Now, we know how to compute $\tilde{\theta}_j$:

- For $j \neq n$: $\tilde{\theta}_j = \theta_j^n + \sum_{k=1,k\neq j}^{n-1}(-1)^{\oplus_{i=k}^{n-1} t_j^i}\theta_j^k$.
- For $j = n$: $\tilde{\theta}_j = \theta_n^{n-1} + \sum_{k=1}^{n-2}(-1)^{\oplus_{i=k}^{n-2} t_n^i}\theta_n^k$.

The last step of Algorithm 1 performs a CNOT on $\left|+_{\tilde{\theta}_j}\right\rangle$ with the control qubit the one-time padded input of client C_j and measures the first in the computational basis. We already have seen how the Z-rotation propagates through the CNOT gate. The X operation of the one-time pad results in a bit flip of the last measurement outcome (either n or $n-1$ according to the two cases above). Therefore, if the one-time pad on register $\mathcal{C}_j$ was $X^{a_j}Z(\theta_j^j)$, after the remote state preparation, the register $\mathcal{C}_j$ is still one-time padded with $X^{a_j}Z(\theta_j)$, where:

- For $j \neq n$:

$$\theta_j = \theta_j^j + (-1)^{t_j^n + a_j}\left(\theta_j^n + \sum_{k=1,k\neq j}^{n-1}(-1)^{\oplus_{i=k}^{n-1} t_j^i}\theta_j^k\right) = \theta_j^j + \sum_{k=1,k\neq j}^{n}(-1)^{\oplus_{i=k}^{n} t_j^i + a_j}\theta_j^k$$

- For $j = n$:

$$\theta_n = \theta_n^n + (-1)^{t_n^{n-1}+a_n}\left(\theta_n^{n-1} + \sum_{k=1}^{n-2}(-1)^{\oplus_{i=k}^{n-2} t_n^i}\theta_n^k\right) = \theta_n^n + \sum_{k=1}^{n-1}(-1)^{\oplus_{i=k}^{n-1} t_n^i + a_n}\theta_n^k$$

From the two cases above, it is obvious that for a general $j = 1, \ldots, n$, Equation (8) is true, which concludes the correctness of the preparation phase of Algorithm 4.

In the computation phase, for each qubit of the brickwork state, all clients input their data in the classical box, and the output is the measurement angle of that qubit. From the properties of MBQC and the flow of the protocol, when the entangled qubits in the preparation phase are in the $|+\rangle$ state, each angle ϕ_j needs to be adjusted to ϕ'_j as defined above. However, here each qubit is rotated by all clients by a total angle θ_j. Therefore, the measurement angle in the MBQC needs to be adjusted by θ_j. The final difference from ordinary MBQC is the insertion of some joint randomness r_j, whose effect is reversed in the consequent steps by adding the randomness to the correction of the function (see [10] for details).

Finally, in the outcome phase, due to previous corrections, the state of qubit $j \in O$ needs to be corrected by client C_{j-q} by applying an operation $Z^{s_j^Z}X^{s_j^X}$, whose classical values can be computed using a computation oracle. We have therefore proven that:

$$d(\pi\mathcal{O}, \mathcal{M}_\perp) = 0$$

meaning that a distinguisher cannot tell the difference between the real communication protocol and an interaction with the ideal MPQC resource when all participants are honest. □

Now, we will present intermediate protocols that prove Equation (6) similarly to the proof technique used in [22]. We will not include any test of correctness for the clients (Algorithm 3 and secret sharing schemes) since the technique used is based on teleportation and delayed measurements, and therefore, it is not possible for the clients to commit to the correct preparation of the quantum states beforehand. However, this does not affect the proof of security, since these protocols are artificial and used only to show that a malicious server does not have access at any step to the private data of the clients. We could have included these tests of correctness of the clients, always asking them to accept any measurement outcome of the server and therefore showing that they do not provide any further information to the server. The complete real communication protocol and the simulated one are presented in the main text. Here, we restate the communication protocol, omitting the steps where the clients' honest behaviour is checked (Algorithm A1) and provide intermediate protocols that are used to prove the equivalence of the real and ideal setting in the case of a malicious server. This will be done by a step-wise process of proving that each of the presented protocols is equivalent to the others, leading to the final one that uses a simulator for the server and the ideal resource defined in the main text.

We can now check step-by-step the equivalence of the protocols described above and argue that Equation (6) holds. We start by comparing Algorithms A1 and A2. In Algorithm A1, at Step 1, client C_j chooses a_j and θ_j^j uniformly at random from their domains and one-time pads his/her input state. In Algorithm A2, at Step 1, C_j chooses uniformly at random $\hat{\theta}_j^j$ and teleports his/her input register to the server, one-time padded with $X^{a_j}Z(\theta_j^j)$, where $\theta_j^j = \hat{\theta}_j^j + \pi r_j^j$. Since both a_j and θ_j^j occur with the same probabilities, the state that the server receives from client C_j is the same in both protocols. Similarly, in Step 2 of Algorithm A1, client C_k chooses uniformly at random θ_j^k and rotates the state $|+\rangle$ accordingly. In Step 2 of Algorithm A2, client C_j chooses uniformly $\hat{\theta}_j^k$ and teleports to the server the state $|+\rangle$ rotated by $\theta_j^k = \hat{\theta}_j^k + \pi r_j^k$. Since θ_j^k appears with the same probabilities for all clients in both protocols, the state described by Equations (A1) and (A4) that the server has are the same. The same argumentation holds for Step 4 of the two protocols; therefore, at the end of the preparation phase, the server has received exactly the same information from the clients. Finally, at the computation phase of the two protocols, the clients choose the measurement angles with the same probability.

We now check the equivalence of Algorithms A2 and A3. The main difference of Algorithm A3 is that the phase flip (measurement of r_j^k) is delayed till after the measurement of the half of EPR pair by the server. This is possible because the operation commutes with the teleportation. The states that the server holds in both protocols are the same due to no signalling. In the computation phase, in Algorithm A2, the uniformly random value $\hat{\theta}_j^k$ defines the measurement angle δ_j, while in Algorithm A3, the uniformly random value δ_j defines $\hat{\theta}_j^k$ and thus the delayed step of the teleportation.

Finally, the combined simulator and ideal resource defined in Algorithms A4 and A5 are just a separation and renaming of the preparation and computation tasks that the clients are required to do. It is easy to see that the ideal resource described in Algorithm A5 fits the requirements of the MPQC resource defined in the main text, and therefore, we have proven that the communication protocol is equivalent to the ideal resource and a simulator for a dishonest server: $\pi_1 \ldots \pi_n \mathcal{O} = \sigma_S \mathcal{M}$.

Algorithm A1 Multiparty quantum computing.

- A quantum input ρ_{in} and measurement angles $\{\phi_j\}_{j=1}^{q}$ for qubits $j \in O^c$.

Preparation phase

quantum input: For $j \in I$

1. Client C_j applies a one-time pad $X^{a_j} Z(\theta_j^j)$ to his/her qubit, where $a_j \in_R \{0,1\}$ and $\theta_j^j \in_R \{l\pi/4\}_{l=0}^7$ and sends it to the server.
2. Each client C_k $(k \neq j)$ chooses $\theta_j^k \in_R \{l\pi/4\}_{l=0}^7$ and sends $\left|+_{\theta_j^k}\right\rangle = \frac{1}{\sqrt{2}}\big(\,|0\rangle + e^{i\theta_j^k}\,|1\rangle\,\big)$ to the server.
3. The server runs Algorithm 1 and announces outcome vector $\mathbf{t}_j$.

At this point the server has the state $\rho'_{in} = \big(X^{a_1}Z(\theta_1) \otimes \cdots \otimes X^{a_n}Z(\theta_n) \otimes \mathbf{1}_{\mathcal{R}}\big) \cdot \rho_{in}$, where

$$\theta_j = \theta_j^j + \sum_{k=1, k\neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \theta_j^k \tag{A1}$$

non-output/non-input qubits: For $j \in O^c \setminus I$

4. All clients C_k, $k \in [n]$ choose $\theta_j^k \in_R \{l\pi/4\}_{l=0}^7$ and send $\left|+_{\theta_j^k}\right\rangle = \frac{1}{\sqrt{2}}\big(\,|0\rangle + e^{i\theta_j^k}\,|1\rangle\,\big)$ to the server.
5. The server runs Algorithm 2 getting outcome vector $\mathbf{t}_j$. He/she ends up with the state $\left|+_{\theta_j}\right\rangle$, where:

$$\theta_j = \theta_j^n + \sum_{k=1}^{n-1} (-1)^{\bigoplus_{i=k}^{n-1} t_j^i} \theta_j^k \tag{A2}$$

output qubits: For $j \in O$, the server prepares $|+\rangle$ states.
graph state: The server entangles the $n+q$ qubits to a brickwork state by applying ctrl-Z gates.

Computation phase

non-output qubits: For $j \in O^c$

1. All clients C_k, $k = 1, \ldots, n$ choose random $r_j^k \in \{0,1\}$ and using a computation oracle, they compute the measurement angle of qubit j:

$$\delta_j := \phi'_j + \pi r_j + \theta_j \tag{A3}$$

   where undefined values are equal to zero, or otherwise:

   - $\phi'_j = (-1)^{a_j + s_j^X} \phi_j + s_j^Z \pi + a_{f^{-1}(j)} \pi$.
   - $r_j = \bigoplus_{k=1}^{n} r_j^k$.
   - $s_i = b_i \oplus r_i$, for $i \leq j$.
2. The server receives δ_j and measures qubit j in basis $\{\left|+_{\delta_j}\right\rangle, \left|-_{\delta_j}\right\rangle\}$, getting result b_j. He/she announces b_j to the clients.

output qubits: For $j \in O$, the server sends the "encrypted" quantum state to client C_{j-q}. All participants jointly compute s_j^X and s_j^Z and send it to client C_{j-q}, who applies operation $Z^{s_j^Z} X^{s_j^X}$ to retrieve the actual quantum output.

Algorithm A2 Multiparty quantum computing (using EPR pairs).

- A quantum input ρ_{in} and measurement angles $\{\phi_j\}_{j=1}^q$ for qubits $j \in O^c$.

Preparation phase

quantum input: For $j \in I$

1. Client C_j creates an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and sends half to the server. He/she then applies a $Z(\hat{\theta}_j^j)$ rotation to his/her qubit, where $\hat{\theta}_j^j \in_R \{l\pi/4\}_{l=0}^7$, performs a CNOT on the remaining half EPR qubit with control the input qubit, and measures the input qubit in the Hadamard basis and the half EPR in the computational basis, getting outcomes r_j^j and a_j respectively.
2. Each client C_k $(k \neq j)$ creates an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and sends half to the server. He/she then chooses $\hat{\theta}_j^k \in_R \{l\pi/4\}_{l=0}^7$ and applies a $Z(\hat{\theta}_j^k)$ rotation to the remaining half EPR and then measures it in the Hadamard basis getting outcome r_j^k.
3. The server runs Algorithm 1 and announces outcome vector $\mathbf{t}_j$.

At this point the server has the state $\rho'_{in} = \left(X^{a_1}Z(\theta_1) \otimes \cdots \otimes X^{a_n}Z(\theta_n) \otimes \mathbf{1}_{\mathcal{R}}\right) \cdot \rho_{in}$, where:

$$\theta_j = \pi \bigoplus_{k=1}^{n} r_j^k + \hat{\theta}_j^j + \sum_{k=1,k\neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \hat{\theta}_j^k \tag{A4}$$

non-output/non-input qubits: For $j \in O^c \setminus I$

4. Each client C_k, $k \in [n]$ creates an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and sends half to the server. He/she then chooses $\hat{\theta}_j^k \in_R \{l\pi/4\}_{l=0}^7$ and applies a $Z(\hat{\theta}_j^k)$ rotation to the remaining half EPR followed by a Hadamard and then measures it in the computational basis getting outcome r_j^k.
5. The server runs Algorithm 2 getting outcome vector $\mathbf{t}_j$. It ends up with the state $\left|+_{\theta_j}\right\rangle$, where:
$$\theta_j = \pi \bigoplus_{k=1}^{n} r_j^k + \hat{\theta}_j^n + \sum_{k=1}^{n-1} (-1)^{\bigoplus_{i=k}^{n-1} t_j^i} \hat{\theta}_j^k \tag{A5}$$

output qubits: For $j \in O$, the server prepares $|+\rangle$ states.
graph state: The server entangles the $n+q$ qubits to a brickwork state by applying ctrl-Z gates.

Computation phase

non-output qubits: For $j \in O^c$

1. The clients use a computation oracle to send the measurement angle of qubit j to the server:
$$\delta_j := \phi'_j + \pi r_j + \theta_j \tag{A6}$$
where undefined values are equal to zero or otherwise:
   - $\phi'_j = (-1)^{a_j + s_j^X} \phi_j + s_j^Z \pi + a_{f^{-1}(j)} \pi$.
   - $r_j = \bigoplus_{k=1}^{n} r_j^k$.
   - $s_i = b_i \oplus r_i$, for $i \leq j$.
2. The server measures qubit j in basis $\{\left|+_{\delta_j}\right\rangle, \left|-_{\delta_j}\right\rangle\}$ and announces result b_j.

output qubits: For $j \in O$, the server sends the "encrypted" quantum state to client C_{j-q}. All participants jointly compute s_j^X and s_j^Z and send it to client C_{j-q}, who applies operation $Z^{s_j^Z} X^{s_j^X}$ to retrieve the actual quantum output.

Algorithm A3 Multiparty quantum computing (using EPR pairs and delaying teleportation).

- A quantum input ρ_{in} and measurement angles $\{\phi_j\}_{j=1}^q$ for qubits $j \in O^c$.

Preparation phase

quantum input: For $j \in I$

1. Client C_j creates an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and sends half to the server. He/she then performs a CNOT on the remaining half EPR qubit with control the input qubit and measures the former in the computational basis, getting outcome a_j.
2. Each client C_k $(k \neq j)$ creates an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and sends half to the server.
3. The server runs Algorithm 1 and announces outcome vector $\mathbf{t}_j$.

non-output/non-input qubits: For $j \in O^c \setminus I$

4. Each client C_k, $k \in [n]$ creates an EPR pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and sends half to the server.
5. The server runs Algorithm 2 getting outcome vector $\mathbf{t}_j$.

output qubits: For $j \in O$, the server prepares $|+\rangle$ states.
graph state: The server entangles the $n + q$ qubits to a brickwork state by applying ctrl-Z gates.

Computation phase

non-output qubits: For $j \in O^c$

1. The computation oracle sends a random angle $\delta_j \in_R \{l\pi/4\}_{l=0}^7$ to the server, who measures qubit j in basis $\{\left|+_{\delta_j}\right\rangle, \left|-_{\delta_j}\right\rangle\}$ and announces result b_j.
2. For $j \in I$, the computation oracle chooses random measurement angles $\hat{\theta}_j^k$ for clients $C_k, k \neq j$ and sets:
$$\hat{\theta}_j^j := \delta_j - \phi'_j - \sum_{k=1,k\neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \hat{\theta}_j^k \tag{A7}$$
while for $j \in O^c \setminus I$, the computation oracle chooses random measurement angles $\hat{\theta}_j^k$ for clients $C_k, k \in [n-1]$ and sets:
$$\hat{\theta}_j^n := \delta_j - \phi'_j - \sum_{k=1}^{n-1} (-1)^{\bigoplus_{i=k}^{n-1} t_j^i} \hat{\theta}_j^k \tag{A8}$$
where undefined values are equal to zero, or otherwise:
   - $\phi'_j = (-1)^{a_j + s_j^X} \phi_j + s_j^Z \pi + a_{f^{-1}(j)} \pi$.
   - $r_j = \bigoplus_{k=1}^{n} r_j^k$.
   - $s_i = b_i \oplus r_i$, for $i \leq j$.
3. The clients measure the respective qubits in the received measurement bases.

output qubits: For $j \in O$, the server sends the "encrypted" quantum state to client C_{j-q}. All participants jointly compute s_j^X and s_j^Z and send it to client C_{j-q}, who applies operation $Z^{s_j^Z} X^{s_j^X}$ to retrieve the actual quantum output.

Algorithm A4 Simulator for server.

```
non-output qubits: For $j \in O^c$
    1. $\sigma_S$ creates $n$ EPR pairs and sends one half of each to the server.
    2. $\sigma_S$ receives vector $\mathbf{t}_j$.
    3. $\sigma_S$ sends $\delta_j \in_R \{l\pi/4\}_{l=1}^{7}$ to the server and receives a reply $b_j \in \{0,1\}$.
output qubits: For $j \in O$
    1. $\sigma_S$ receives $n$ qubits from the server.
    2. $\sigma_S$ sends the other halves of the EPR pairs, the received quantum states, as well as $\delta_j$, $b_j$
       and $\mathbf{t}_j$ for $j = 1, \ldots, q$, to the MPQC resource.
```

Algorithm A5 MPQC resource.

```
1. The resource receives the $n$ qubits of $\rho_{in}$ from the clients, measurement angles $\{\phi_j\}_{j=1}^{q}$ and all
   the information from $\sigma_S$.
2. For $j \in I$: the resource performs a CNOT on the corresponding EPR half with the input qubit
   as control and measures the EPR half in the computational basis, getting result $a_j$. It chooses
   random measurement angles $\hat{\theta}_j^k$ for the qubits coming from clients $C_k, k \neq j$, sets:

       $\hat{\theta}_j^j := \delta_j - \phi'_j - \sum_{k=1, k\neq j}^{n} (-1)^{\bigoplus_{i=k}^{n} t_j^i + a_j} \hat{\theta}_j^k$

   and measures the corresponding qubits. For $j \in O^c \setminus I$, it chooses random measurement angles
   $\hat{\theta}_j^k$ for clients $C_k, k \in [n-1]$ and sets:

       $\hat{\theta}_j^n := \delta_j - \phi'_j - \sum_{k=1}^{n-1} (-1)^{\bigoplus_{i=k}^{n-1} t_j^i} \hat{\theta}_j^k$

   where undefined values are equal to zero, or otherwise:
       • $\phi'_j = (-1)^{a_j + s_j^X} \phi_j + s_j^Z \pi + a_{f^{-1}(j)} \pi$.
       • $r_j = \bigoplus_{k=1}^{n} r_j^k$.
       • $s_i = b_i \oplus r_i$, for $i \leq j$.
3. For $j \in O$: the resource performs corrections $Z^{s_j^Z} X^{s_j^X}$ on the remaining qubits.
```

References

1. Lo, H.-K.; Chau, H.F. Is quantum bit commitment really possible? *Phys. Rev. Lett.* **1997**, *78*, 3410–3413.
2. Mayers, D. Unconditionally secure quantum bit commitment is impossible. *Phys. Rev. Lett.* **1997**, *78*, 3414–3417.
3. Lo, H.-K. Insecurity of quantum secure computations. *Phys. Rev. A* **1997**, *56*, 1154–1162.
4. Salvail, L.; Schaffner, C.; Sotakova, M. On the Power of Two-Party Quantum Cryptography. In Proceedings of the ASIACRYPT 2009, Tokyo, Japan, 6–10 December 2009; Volume 5912, pp. 70–87.
5. Dupuis, F.; Nielsen, J.B.; Salvail, L. Secure two-party quantum evaluation of unitaries against specious adversaries. In Proceedings of the CRYPTO 2010, Santa Barbara, CA, USA, 15–19 August 2010; Volume 6223, pp. 685–706.
6. Dupuis, F.; Nielsen, J.B.; Salvail, L. Actively Secure Two-Party Evaluation of any Quantum Operation. In Proceedings of the CRYPTO 2012, Santa Barbara, CA, USA, 19–23 August 2012; pp. 794–811.
7. Ben-Or, M.; Crépeau, C.; Gottesman, D.; Hassidim, A.; Smith, A. Secure Multiparty Quantum Computation with (Only) a Strict Honest Majority. In Proceedings of the IEEE FOCS 2006, Berkeley, CA, USA, 21–24 October 2006; pp. 249–260.

8. Raussendorf, R.; Briegel, H.J. A One-Way Quantum Computer. *Phys. Rev. Lett.* **2001**, *86*, 5188.
9. Leung, D.W. Quantum computation by measurements. *Int. J. Quantum Inf.* **2004**, *2*, 33–43.
10. Broadbent, A.; Fitzsimons, J.F.; Kashefi, E. Universal blind quantum computation. In Proceedings of the FOCS 2009, Atlanta, GA, USA, 25–27 October 2009; pp. 517–526.
11. Fitzsimons, J.F.; Kashefi, E. Unconditionally Verifiable Blind Computation. *arXiv* **2012**, arXiv:1203.5217.
12. Kashefi, E.; Wallden, P. Garbled Quantum Computation. *arXiv* **2016**, arXiv:1606.06931.
13. Danos, V.; Kashefi, E. Determinism in the one-way model. *Phys. Rev. A* **2006**, *74*, 052310.
14. Canetti, R. Universally composable security: A new paradigm for cryptographic protocols. In Proceedings of the FOCS 2001, Las Vegas, NV, USA, 14–17 October 2001; pp. 136–147.
15. Ishai, Y.; Prabhakaran, M.; Sahai, A. Founding cryptography on oblivious transfer–efficiently. In Proceedings of the CRYPTO 2008, Santa Barbara, CA, USA, 17–21 August 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 572–591.
16. Goldwasser, S.; Lindell, Y. Secure Computation without Agreement. In *Distributed Computing (DISC 2002)*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2508.
17. Unruh, D. Universally Composable Quantum Multiparty Computation. In Proceedings of the EUROCRYPT 2010, French Riviera, 30 May–3 June 2010; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6110, pp. 486–505.
18. Maurer, U.; Renner, R. Abstract cryptography. In *Innovations in Computer Science*; Tsinghua University Press: Beijing, China, 2011.
19. Kumaresan, R.; Patra, A.; Rangan, C.P. The round complexity of verifiable secret sharing: The statistical case. In Proceedings of the ASIACRYPT 2010, Singapore, 5–9 December 2010; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6477, pp. 431–447.
20. Laud, P.; Pankova, A. Verifiable Computation in Multiparty Protocols with Honest Majority. In Proceedings of the Provable Security (ProvSec 2014), Hong Kong, China, 9–10 October 2014; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2014; Volume 8782.
21. Dunjko, V.; Kashefi, E.; Leverrier, A. Universal Blind Quantum Computing with Weak Coherent Pulses. *Phys. Rev. Lett.* **2012**, *108*, 200502.
22. Dunjko, V.; Fitzsimons, J.F.; Portmann, C.; Renner, R. Composable security of delegated quantum computation. In Proceedings of the ASIACRYPT 2014, Kaoshiung, Taiwan, 7–11 December 2014; pp. 406–425.
23. Hirt, M.; Nielsen, J.B. Upper Bounds on the Communication Complexity of Optimally Resilient Cryptographic Multiparty Computation. In Proceedings of the ASIACRYPT 2005, Chennai, India, 4–8 December 2005; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3788.
24. Morimae, T.; Fujii, K. Blind quantum computation protocol in which Alice only makes measurements. *Phys. Rev. A* **2013**, *87*, 050301.
25. Broadbent, A.; Jeffery, S. Quantum homomorphic encryption for circuits of low T-gate complexity. In Proceedings of the CRYPTO 2015, Santa Barbara, CA, USA, 16–20 August 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 609–629.
26. Dulek, Y.; Schaffner, C.; Speelman, F. Quantum homomorphic encryption for polynomial-sized circuits. *arXiv* **2016**, arXiv:1603.09717v1.