Quantum Information and Computation, Vol. 16, No. 1&2 (2016) 0061–0086

QUANTUM-ENHANCED SECURE DELEGATED CLASSICAL COMPUTING

VEDRAN DUNJKO
Institut fur Theoretische Physik, Universitat Innsbruck, Technikerstrase, 25, A-6020 Innsbruck, Austria
Division of Molecular Biology, Ruder Boskovic Institute, Bijenicka cesta 54, 10002 Zagreb, Croatia

THEODOROS KAPOURNIOTIS
School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh, UK

ELHAM KASHEFI
School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh, UK

Received February 16, 2015
Revised October 5, 2015

We present a family of quantumly-enhanced protocols to achieve unconditionally secure delegated classical computation where the client and the server have both their classical and quantum computing capacity limited. We prove the same task cannot be achieved using only classical protocols. This extends the work of Anders and Browne on the computational power of correlations to a security setting. In doing so we are able to highlight the power of online quantum communication as we prove the same task could not be achieved using pre-shared (offline) quantum correlations.

Keywords: Delegated Computing, Quantum Communication, Foundation of Quantum Theory

Communicated by: R Jozsa & R de Wolf

1 Introduction

The concept of delegated quantum computing is a quantum extension of the classical task of computing with encrypted data without decrypting them first [1]. Many quantum protocols [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] address this challenge for a futuristic quantum client-server setting achieving a wide range of security properties. The central challenge of all these protocols to be applicable for classical tasks (such as secure multi party computation [14, 15] or fully homomorphic encryption [16]) is the requirement of a server with a universal quantum computer. By restricting the task to classical computation only, we derive a family of protocols for unconditionally secure delegation of any classical computation to a remote server that has access to basic quantum devices, currently available in any scientific or commercial labs. Concretely, we present how a client with access to a non-universal classical gate (e.g. parity gate) could securely delegate the computation of a universal gate (e.g. NAND gate) to an untrusted server with capability of manipulating a single qubit. We note that, in this work, the security of the protocol pertains to the privacy, or confidentiality, of the protocol - we

require that no information about the client's input leaks to the server.[a]

The general idea behind our protocol for the secure computation of the universal NAND gate is based on the following simple fact presented for the first time in [17] which was further utilised for a multi-party cryptographic setting in [18], the role of contextuality in computational speedup in [19] and the relation of entangled quantum states and multi-party computational games in [20]. Let M^0 to denote a Pauli-X measurement and M^1 a Pauli-Y, then the three qubit measurement $M^a \otimes M^b \otimes M^{a\oplus b}$ of the GHZ state (denoted in this paper as $|\Psi\rangle = 1/\sqrt{2}(|001\rangle - |110\rangle)$) computes NAND$(a, b)$ (see more details in Section 2). We then show how instead of switching the measurements (based on the input a and b) one can simply apply the pre-rotation operation based on a, b and $a \oplus b$ to the GHZ state and then the Pauli-X measurements of all three qubits achieve the same task. This will allow us to achieve a client-server scenario where the client effectively chooses the measurement basis by this pre-rotation while hiding his secret input bits (a and b). The next step for obtaining the full security property is the application of additional random Z gates to hide the outcome computed by the server. These hiding steps lead to the necessity of quantum communication (as we prove next) and as a result we can replace the requirement of GHZ state with sequential rotation and one final measurement on a single qubit state as well. In other words if we denote the $\pi/2$ rotation along the Z axis by S then we prove that the local operators of the form

$$S^{\dagger^{a\oplus b}} S^b S^a |+\rangle$$

encode the input of the client in the resource state, $|+\rangle$, while permitting the server to perform the other operations required to compute the NAND gate. In this paper, for any unitary U and bit x, with U^x we denote the identity $\mathbb{1}$ (for $x = 0$) and U (for $x = 1$). Since all the information of the client is encoded in the phase of the states, additional randomly chosen Z gates achieve a full one-time pad of the client's information, which can easily be decoded by the client by a bit-flip (for details see **Protocols 1 - 4**). Here we present specific protocols based on various manipulations of the single qubit $|+\rangle$ and three qubits entangled GHZ state, however one could easily adapt these protocols to cover various encodings necessary for the specific noise model or available resources within a particular implementation platform as we have recently done for an experimental demonstration of our protocol in an optical setting [21].

The actual setting of our protocols (a restricted client with XOR gate only), on its own, is not a realistic set-up. The potential advantage of the employment of such quantum scheme in a classical secure multi-party protocol for reducing the overall overhead should be explored elsewhere. However one could think of our protocols as a game scenario that exhibits the power of quantum communication. It is straightforward to prove that purely classical players (i.e. a classical protocol with classical client and server) could not win deterministically the game of computing securely the NAND of encoded input bits (**Theorem 1**). The proof is based on a reduction to the impossibility of computing a non-linear function (e.g NAND) with a linear function (e.g Parity) given a generic advice string.[b] On the other hand to prove that

[a] Stronger notions of security would also require the verifiability of such a computation, that is, a mechanism which provides a means for the client to check that the output of the computation is correct. In this work, we do not require verifiability.

[b] By a generic advice string we mean any string which does not depend on the input bits.

having pre-shared quantum correlation or equivalently using offline quantum communication (quantum states independent of classical inputs) will also not lead to a winning strategy, a completely different proof technique had to to be developed (**Theorem 2** and **3**). Through a series of lemmas we show any quantum offline protocol for secure delegated NAND computing could be reduced to a simple protocol with one round only, where the classical communication will be necessarily dependent on the client's secret inputs. Then the correctness criteria for a deterministic secure NAND computation is proven to be equivalent to perfect discrimination of the classical encoded messages and hence the leakage of client's secret, see **Lemma 7**. To the best of our knowledge this is the first time that a security game manifests the structural difference between static pre-shared correlation versus a dynamic quantum communication.[c]

2 Secure NAND Protocols

There are three types of protocols that we introduce here, to address various implementation scenarios. These families achieve the same goal and differ only in the required quantum gadgets of the client. Following the construction steps explained in the introduction, in the first family of the protocols, it is assumed that client can create or have secure access to some simple (few qubits) entangled states. On the other hand, in the second family it is assumed that the client is able to measure the flying qubit that it receives through an untrusted channel to perform its desired universal computation. In the third setting, the client needs only to have the capacity to perform simple single qubit rotations. Importantly, in all three scenarios the classical computation of the client is restricted to XOR operations.

For all of our SecureNAND protocols will always assume that the client is honest, and behaves as specified by the protocols. In our setting, the server is the untrusted party. However, we will require and that the protocols are correct - yield correct outputs in the case of an honest server. This property corresponds to completeness in the terminology of interactive proof systems, whereas we do not require soundness (verifiability, in the context of delegated computing).

Aside from correctness, regarding the guarantees for the honest client, we require the protocols are secure, specifically that they reveal no information about the client's input (aside from the input's size) to the server. This property, also known as blindness, is defined formally as follows.

Definition 1. *We will say that any SecureNAND protocol is secure (also referred to as* blind*) if the quantum states sent during the execution of the protocol from the client to the server, once averaged over the client's internal secret parameters, do not depend on the secret input bits to the client i.e. the inputs to the NAND gate which is to be computed in a secure delegated fashion. In other words, the averaged states sent by the client are the same for any choice of the inputs.*

In other words, the system the server receives from the client could have been generated by the server without receiving any information from the client.

[c] While obviously quantum communication could be achieved using shared entanglement and classical communication, however the restricted client in our set up could not employ teleportation. This is in fact chosen so that to highlight the differences.

We first prove that it is impossible to achieve the similar task of secure delegated computing of our protocols by removing the quantum requirement.

Theorem 1. *No classical protocol, in which the client is restricted to XOR computations can delegate deterministically computation of NAND to a server while keeping the blindness.*

Proof. We prove this result first for the case of two rounds of communication, and no initial shared randomness. Any such protocol will have the following three stages: client's encoding, server's computation, and client's decoding.

client's encoding. In this stage, the only thing the client can do is to compute $C_1(a, b, \overrightarrow{x})$, where a, b are the input bits, $\overrightarrow{x}$ is a random bit string (of any length) and C_1 is a computation which can be implemented using only XOR gates. However, the state $C_1(a, b, \overrightarrow{x})$ must be independent from a and b to maintain blindness when averaged over all $\overrightarrow{x}$.

server's computation. The only thing the server can do is to apply some computable function S on $C_1(a, b, \overrightarrow{x})$, thus returning $S(C_1(a, b, \overrightarrow{x}))$.

client's decoding. The only thing the client can do is to run some function C_2, on all the data he has, which is implementable using XOR gates only:

$$C_2(a, b, \overrightarrow{x}, S(C_1(a, b, \overrightarrow{x}))) = NAND(a, b) \text{ (correctness)}$$

and the output must (deterministically) be the NAND of the inputs.

Let $c = C_1(a', b', \overrightarrow{x'})$ be some constant the client may send to the server. Then, because of blindness it must hold that for all a, b there must exist $\overrightarrow{x}(a, b)$, which depends on a, b such that

$$C_1(a, b, \overrightarrow{x}(a, b)) = c.$$

To see this, note that if the client could send c, but not for some inputs a'' and b'', then upon receiving c the server learns something about the input, namely that it is not a'', b'', which violates blindness. Note also that since all the computations the client can perform use only XOR gates (and without the loss of generality, reversible), the client *can* compute $\overrightarrow{x}(a, b)$ given a, b using only XOR operations. But then, by the correctness of the protocol we have that

$$C_2(a, b, \overrightarrow{x}(a, b), S(c)) = NAND(a, b) \text{ (correctness)}.$$

But $S(c)$ is constant as well. This implies that given a fixed string $S(c)$ the client can compute the NAND of any input using just XOR gates, which is not possible.

This argument can be further generalized to a setting with shared randomness and many rounds of communication. It is easy to see that the randomness cannot help as the protocol must be deterministic (hence work for any sampling of the joint random variable), whereas using multiple rounds (all of which must be independent of the input, from the viewpoint of the server) just yields a longer constant string (analogous to $S(c)$) using which the client can compute the $NAND$ on her own, which is again impossible $\square$.

We proceed now with constructing the family of quantum protocols for the same task. While the simplest protocol is demonstrated last, we present the sequence of adapting the

GHZ game as shown in [17] to a security scenario to present a simple security and correctness proof. Then with each new family we reduce the client's requirement while maintaining the same properties.

2.1 Preparing client

Protocol 1 Entangled-based Preparing client SecureNAND

- Input (to client): two bits a, b
- Output (from client): $\neg(a \wedge b)$
- The Protocol:
  - client's round
    1. $r \in_{\mathsf{R}} \{0, 1\}$
    2. client generates
    $$|\Psi'\rangle = Z_1^r \left(S_1^\dagger\right)^a \left(S_2^\dagger\right)^b \left(S_3^\dagger\right)^{a \oplus b} |\Psi\rangle$$
    and sends it to the server.
  - server's round
    1. server measures the qubits 1,2 and 3, with respect to the observables X_1, X_2, and X_3, obtaining outcomes b_1, b_2 and b_3, respectively.
    2. server sends b_1, b_2, b_3 to client
  - client's round
    1. client computes
    $$out = b_1 \oplus b_2 \oplus b_3 \oplus r \tag{1}$$
    2. client outputs *out*.

In Protocol 1 the client generates a GHZ state of 3 qubits which are rotated depending on the values of the inputs a, b, $a \oplus b$ and a random bit r. Qubits are sent through an untrusted quantum channel from client to server who applies a Pauli-X measurement on the qubits and sends the classical result to the client via an untrusted classical channel. The client produces the final output by applying classical XOR gates between the received classical bits and the random bit. In what follows we denote a random selection of an element of a set by $\in_{\mathsf{R}}$.

We will say any SecureNAND protocol is *correct* if for every run of the protocol where both players are honest (adhere to the protocol) and for all inputs a, b we have

$$out = \neg(a \wedge b) = 1 \oplus (ab)$$

This definition will be used for all the presented protocols in this paper. Throughout this paper we will be using the notation for the logical *and* between two bits a, b as $a \wedge b$ and ab interchangeably.

Lemma 2. *Protocol 1 is correct.*

Proof. First note that the protocol is correct if the following eigenstate equality is true for all binary variables a, b, r

$$X_1X_2X_3|\Psi'\rangle = (-1)^{1\oplus ab\oplus r}|\Psi'\rangle \tag{2}$$

as this equality guarantees that the parity of the outcomes of stabiliser measurements of server (in basis $X_1X_2X_3$) equals $1 \oplus ab \oplus r$ which implies client will decode the correct outcome in Equation 1 of Protocol 1. For simplicity define the following notion for Pauli observable

$$P^i = \begin{cases} X & \text{if } i{=}0 \\ Y & \text{if } i{=}1 \end{cases}$$

we then have the following commutation relations $\forall\ b, r \in \{0, 1\}$

$$\begin{aligned} P^bZ^r &= (-1)^r Z^r P^b \\ P^bS^r &= (-1)^{(b\oplus 1)r} S^r P^{b\oplus r} \\ P^b\left(S^\dagger\right)^r &= (-1)^{br}\left(S^\dagger\right)^r P^{b\oplus r} \end{aligned}$$

and in particular

$$X\left(S^\dagger\right)^r = \left(S^\dagger\right)^r P^r$$

and hence as stated in [22] we obtain that $\forall\ a, b \in \{0, 1\}$

$$P_1^a P_2^b P_3^{a\oplus b}|\Psi\rangle = (-1)^{(1\oplus ab)}|\Psi\rangle$$

We proceed to show that Equation (2) holds:

$$\begin{aligned} X_1X_2X_3|\Psi'\rangle &= X_1X_2X_3Z_1^r\left(S_1^\dagger\right)^a\left(S_2^\dagger\right)^b\left(S_3^\dagger\right)^{a\oplus b}|\Psi\rangle \\ &= \left[X_1Z_1^r\left(S_1^\dagger\right)^a\right]_1\left[X_2\left(S_2^\dagger\right)^b\right]_2\left[X_3\left(S_3^\dagger\right)^{a\oplus b}\right]_3|\Psi\rangle \\ &= (-1)^r\left[Z_1^r\left(S_1^\dagger\right)^a P_1^a\right]_1\left[\left(S_2^\dagger\right)^b P_2^b\right]_2\left[\left(S_3^\dagger\right)^{a\oplus b} P_3^{a\oplus b}\right]_3|\Psi\rangle \\ &= (-1)^r Z_1^r\left(S_1^\dagger\right)^a\left(S_2^\dagger\right)^b\left(S_3^\dagger\right)^{a\oplus b} P_1^aP_2^bP_3^{a\oplus b}|\Psi\rangle \\ &= (-1)^{1\oplus ab\oplus r} Z_1^r\left(S_1^\dagger\right)^a\left(S_2^\dagger\right)^b\left(S_3^\dagger\right)^{a\oplus b}|\Psi\rangle = (-1)^{1\oplus ab\oplus r}|\Psi'\rangle \end{aligned}$$

In the derivation above we have simply used the trivial commutativity of operators acting on disjoint subsystems. So the Lemma holds $\square$.

Next, we prove the blindness of this protocol.

In the remainder of this paper we will use the following short-hand notation:

$$\boxed{X} := |X\rangle\langle X|$$

for all labels X. This is a non-standard notation used here for the sake of brevity.

Lemma 3. *Protocol 1 is blind.*

Proof. For fixed input bits a and b the state the server receives from the client can be written as

$$\rho_S = \sum_r \frac{1}{2} Z_1^r \eta Z_1^r \tag{3}$$

where

$$\eta = \left(S_1^\dagger\right)^a \left(S_2^\dagger\right)^b \left(S_3^\dagger\right)^{a\oplus b} |\Psi\rangle\langle\Psi| (S_1)^a (S_2)^b (S_3)^{a\oplus b}$$

Note that η can be written as $\mathbf{S}|\Psi\rangle\langle\Psi|\mathbf{S}^\dagger$, where

$$\mathbf{S} = \left(S_1^\dagger\right)^a \left(S_2^\dagger\right)^b \left(S_3^\dagger\right)^{a\oplus b}$$

The operator $\mathbf{S}$ does not depend on the r_i variables, and is diagonal in the computational basis so it commutes with Pauli Z operators and hence we have

$$\rho_S = \mathbf{S}\left(\textstyle\sum_r \frac{1}{2} Z_1^r |\Psi\rangle\langle\Psi| Z_1^r\right)\mathbf{S}^\dagger$$

The operator $\sum_r \frac{1}{2} Z_1^r |\Psi\rangle\langle\Psi| Z_1^r$ can explicitly be written as

$$\frac{1}{2}\left(\frac{1}{2}\left(|001\rangle\langle 001| + |110\rangle\langle 110| - |001\rangle\langle 110| - |110\rangle\langle 001|\right) + \frac{1}{2}\left(|001\rangle\langle 001| + |110\rangle\langle 110| + |001\rangle\langle 110| + |110\rangle\langle 001|\right)\right)$$

which in our notation is equal to $\frac{1}{2}\left(\boxed{001} + \boxed{110}\right)$. Hence the operator is diagonal in the computational basis, and again commutes with $\mathbf{S}$ so we get:

$$\rho_s = \left(\textstyle\sum_r \frac{1}{2} Z_1^r |\Psi\rangle\langle\Psi| Z_1^r\right)\mathbf{S}\mathbf{S}^\dagger = \frac{1}{2}\left(\boxed{001} + \boxed{110}\right)$$

This state is independent from a and b and the lemma is proved □.

2.2 Measuring client

In Protocol 2 server generates a GHZ state of 3 qubits. The qubits are sent through an untrusted quantum channel from the server to the client. The client applies a Pauli-X or Pauli-Y measurement on the qubits depending on the classical inputs a and b and their classical XOR and produces the final output by applying classical XOR gates between the measurement outputs.

Lemma 4. *Protocol 2 is blind and correct.*

Proof. The correctness of this protocol follows directly from the result in [22]. The blindness of the protocol trivially follows from the fact that no information is sent from the client to the server, thus the protocol is blind in all no signaling theories (including standard Quantum Mechanics) □.

Protocol 2 Entangled-based Measuring client SecureNAND

- Input (to client): two bits a, b
- Output (from client): $\neg(a \wedge b)$
- The Protocol:
  - server's round
    1. The server prepares the state $|\Psi\rangle$ and sends it to the client
  - client's round
    1. The client computes $c = a \oplus b$, measures the qubits 1,2 and 3, with respect to the observables P^a, P^b, and P^c, obtaining outcomes b_1, b_2 and b_3, respectively.
    2. client computes

$$out = b_1 \oplus b_2 \oplus b_3 \tag{4}$$

    3. client outputs *out*.

2.3 Bounce Protocol

In Protocol 3 we reduce the requirements on the client side, which no longer has to measure or prepare states, but rather only modify locally the GHZ state of 3 qubits prepared by the server. The client applies single-qubit quantum operators depending on the values of the inputs a, b, $a \oplus b$ and 3 classical random bits. The client sends the rotated qubits to the server via an untrusted quantum channel. The server applies a Pauli-X measurement on the qubits and sends the classical result to the client via an untrusted classical channel. The client produces the final output by applying classical XOR gates between the received classical bits and the random bits.

Lemma 5. *Protocol 3 is correct.*

Proof. The correctness is directly obtained from Lemma 2 on the correctness of Protocol 1. To see this note that the states the server performs the measurements on are identical in the two protocols, up to the existence of possible $Z_2^{r_2}$ and $Z_3^{r_3}$ rotations on the second and third qubit. Since we have

$$\begin{aligned} XZ^r &= (-1)^r Z^r X, \text{and} \\ YZ^r &= (-1)^r Z^r Y, \end{aligned}$$

these rotations cause an additional (multiplicative) phase of $(-1)^{r_2 \oplus r_3}$. But this is compensated for in the modified decoding of the client (see Equation 5 in Protocol 3) so the output is correct in this protocol as well $\square$.

Lemma 6. *Protocol 3 is blind.*

Protocol 3 Entangled-based Bounce SecureNAND

- Input (to client): two bits a, b
- Output (from client): $\neg(a \wedge b)$
- The Protocol:
  - server's round
    1. The server prepares the state $|\Psi\rangle$ and sends it to the client
  - client's round
    1. client receives the state $|\Psi\rangle$ from the server.
    2. client generates $r_1, r_2, r_3 \in_{\mathsf{R}} \{0, 1\}$
    3. client modifies the state $|\Psi\rangle$ to $|\Psi'\rangle$ as follows

$$|\Psi'\rangle = Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \left(S_1^\dagger\right)^a \left(S_2^\dagger\right)^b \left(S_3^\dagger\right)^{a\oplus b} |\Psi\rangle$$

       and sends it to the server.
  - server's round
    1. server measures the qubits 1,2 and 3, with respect to the observables X_1, X_2, and X_3, obtaining outcomes b_1, b_2 and b_3, respectively.
    2. server sends b_1, b_2, b_3 to client
  - client's round
    1. client computes

$$out = b_1 \oplus b_2 \oplus b_3 \oplus r_1 \oplus r_2 \oplus r_3. \tag{5}$$

    2. client outputs *out*.

Proof. For fixed input a and b the final state server obtains in the protocol can be written as

$$\rho_S^{fin} = \sum_{r_1,r_2,r_3} \frac{1}{8} \left(Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \eta \left(Z_1^{r_1} Z_3^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right)$$

with

$$\eta = \left(\left(S_1^\dagger\right)^a \left(S_2^\dagger\right)^b \left(S_3^\dagger\right)^{a\oplus b} \otimes \mathbb{1}_S \right) \rho_S^{init} \left((S_1)^a (S_2)^b (S_3)^{a\oplus b} \otimes \mathbb{1}_S \right)$$

where ρ_S^{init} is any state the malevolent server could have initially prepared. In the expression above, we have made no assumptions on the dimensionality of the initial state the server may have prepared, and we only assume that the local operations of the client are correct, single qubit operations, acting on three distinct qubits.

Note further that the actions of the client are only on a subsystem of the whole system in the state ρ_S^{init}, signifying that the server might have prepared an entangled state, and sent only a subsystem to the client to be modified, while keeping the remainder of the system. To simplify the state we could commute the Z operators with the phase $S^\dagger$ operators since the parameters of the phase operators do not depend on r_i values. Introducing the shorthand

$$\mathbf{S} = \left(\left(S_1^\dagger\right)^a \left(S_2^\dagger\right)^b \left(S_3^\dagger\right)^{a\oplus b} \otimes \mathbb{1}_S \right)$$

we can rewrite the state of the server's system as

$$\rho_S^{fin} = (\mathbf{S} \otimes \mathbb{1}_S) \sum_{r_1,r_2,r_3} \frac{1}{8} \left(Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \rho_S^{init} \left(Z_1^{r_1} Z_3^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \left(\mathbf{S}^\dagger \otimes \mathbb{1}_S\right)$$

The state ρ_S^{init} has two partitions - the partition corresponding to the subsystem the server sends to the client, and the subsystem he keeps. Thus ρ_S^{init} can be written (in the Pauli operator basis) as

$$\sum_{i,j} \alpha_{i,j} \underbrace{\sigma_i}_{C} \otimes \underbrace{\sigma_j}_{S'}$$

where C denotes the subsystem sent to the client, and S' the subsystem kept by the server, and σ_i and σ_j denote general Pauli operators acting on the two respective subsystems. Next, we have the following derivation:

$$\begin{aligned}
& \sum_{r_1,r_2,r_3} \frac{1}{8} \left(Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \rho_S^{init} \left(Z_1^{r_1} Z_3^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \\
= & \sum_{r_1,r_2,r_3} \frac{1}{8} \left(Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \sum_{i,j} \alpha_{i,j} \underbrace{\sigma_i}_{C} \otimes \underbrace{\sigma_j}_{S'} \left(Z_1^{r_1} Z_3^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \\
= & \frac{1}{8} \sum_{i,j} \alpha_{i,j} \left(\sum_{r_1,r_2,r_3} Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \sigma_i Z_1^{r_1} Z_3^{r_2} Z_3^{r_3} \right) \otimes \sigma_j
\end{aligned}$$

Note that since both X and Y anticommute with Z, the expression

$$\sum_{r_1,r_2,r_3} Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \sigma_i Z_1^{r_1} Z_3^{r_2} Z_3^{r_3}$$

is non-zero only if all the single qubit operators making up σ_i are either Z or identity, and in both cases diagonal in the computational basis. Thus, we can write the final expression of the derivation above as

$$\sum_{i,j} \alpha'_{i,j} \sigma'_i \otimes \sigma_j$$

where σ'_i is diagonal in the computational basis. So, overall, for the state of the server's system we have

$$\begin{aligned} & (\mathbf{S} \otimes \mathbb{1}_S) \sum_{r_1,r_2,r_3} \frac{1}{8} \left(Z_1^{r_1} Z_2^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \rho_S^{init} \left(Z_1^{r_1} Z_3^{r_2} Z_3^{r_3} \otimes \mathbb{1}_S\right) \left(\mathbf{S}^\dagger \otimes \mathbb{1}_S\right) \\ = \; & (\mathbf{S} \otimes \mathbb{1}_S) \sum_{i,j} \alpha'_{i,j} \sigma'_i \otimes \sigma_j \left(\mathbf{S}^\dagger \otimes \mathbb{1}_S\right) \\ = \; & \sum_{i,j} \alpha'_{i,j} \left(\mathbf{S} \otimes \mathbb{1}_S\right) \sigma'_i \otimes \sigma_j \left(\mathbf{S}^\dagger \otimes \mathbb{1}_S\right) \end{aligned}$$

and since σ'_i commute with $\mathbf{S}$ we get:

$$\sum_{i,j} \alpha'_{i,j} \sigma'_i \mathbf{S}\mathbf{S}^\dagger \otimes \sigma_j = \sum_{i,j} \alpha'_{i,j} \sigma'_i \otimes \sigma_j$$

Since $\alpha'_{i,j}$ is independent from a and b, this state is independent from a and b and the lemma is proved □.

2.4 Single Qubit Protocols

Here, we give variants of a new class of secure NAND protocols which only require single qubit manipulations. Similarly to the variants we have given for the GHZ-based protocols, the single qubit protocol can also be modified to a client preparation or a measuring client protocol. In the former, it is the client which would prepare the initial $|+\rangle$ state, whereas in the measuring client protocol, the client would perform the final measurements. Similar to the entangled-based scenario, all variations of protocols are blind and correct as a simple consequence of the Single Qubit Bounce SecureNAND protocol (that we describe next).

In Protocol 4, the server generates a single qubit state ($|+\rangle$) and sends it via an untrusted quantum channel to the client who applies a series of single qubit rotation operator depending on the values of the inputs a, b, $a \oplus b$, and a classical random bit. The client sends the rotated qubit to the server via untrusted quantum channel. Sever applies a Pauli-X measurement on the qubit and sends the classical result to the client via an untrusted classical channel. The client produces the final output by applying classical XOR gates between the received classical bit and the random bit and constant bit 1. The correctness and blindness are directly obtained from the proof for the entangled-based protocols. To see the correctness note that if the server was honest, it is a straightforward calculation to see the state of the qubit the server receives is

$$Z^r Z^{a \wedge b} |+\rangle$$

Then the result of the measurement performed by the server is $s = r \oplus a \wedge b$, and the decoding produces $out = 1 \oplus a \wedge b$ as required. To see the security, note that the most general strategy of the server is to prepare a bipartite state $\pi_{1,2}$ and send the first subsystem to the client. Then the state of the server system (up to a normalisation factor), once the client performed his round is

$$\sum_r \left(Z^r Z^{a \wedge b} \otimes \mathbb{1}_2\right) \pi_{1,2} \left(Z^r Z^{a \wedge b} \otimes \mathbb{1}_2\right) = \sum_{r'} \left(Z^{r'} \otimes \mathbb{1}_2\right) \pi_{1,2} \left(Z^{r'} \otimes \mathbb{1}_2\right)$$

Protocol 4 Single Qubit Bounce SecureNAND

- Input (to client): two bits a, b
- Output (from client): $\neg(a \wedge b)$
- The Protocol:
  - server's round
    1. The server prepares the state $|+\rangle$ and sends it to the client
  - client's round
    1. client receives the state $|+\rangle$ from the server.
    2. client generates $r \in_{\mathsf{R}} \{0, 1\}$
    3. client modifies the state $|+\rangle$ to $|\Psi\rangle$ as follows

$$|\Psi\rangle = Z^r S^a S^b \left(S^\dagger\right)^{a \oplus b} |+\rangle$$

       and sends it to the server.
  - server's round
    1. The server measures the qubit with respect to the X basis, obtaining the outcome s
    2. server sends s to client
  - client's round
    1. client computes

$$out = s \oplus r \oplus 1 \tag{6}$$

    2. client outputs *out*.

where $r' = r \oplus a \wedge b$. Since r is distributed uniformly at random, so is r' so the state above does not depend on a or b.

3 No-go Result

The main contribution of our paper is to prove the optimality of the quantum protocols of the last section. We prove that it is impossible to achieve the similar task of the secure delegated NAND computing if one attempts to remove any quantum communication. Next we show that the communicated quantum states must also depend on the classical input of the client as it is done in our protocols. More precisely, we will show that any quantum protocol where a XOR-restricted client computes $NAND(a, b)$, by initially sending a quantum state ρ to the server, followed by classical communication only, can be perfectly blind and perfectly correct only if the state ρ depends on the input bits a, b of the client. The protocols without this dependence, so where all the quantum communication can be done independently from the input of the client (hence can be done before the client decides on her input bits), we call quantum off-line protocols. Thus, we show that a blind quantum-offline protocol with a

XOR-restricted client is not possible ?[d] We begin by addressing protocols with two rounds of communication between the client and the server. By round we refer to an instance of either the client sending a message to the server, or the server sending a message to the client. Since the last message, for it to have any meaning, must come from the server, the order of the two rounds is client $\rightarrow$ server, followed by server $\rightarrow$ client. The generic description of a potential secure NAND quantum offline protocol with two rounds is given later in Protocol 7. In order to prove the impossibility of obtaining such a protocol we prove several lemmas proving first the impossibility of a particular class of somehow 'minimal' NAND quantum offline protocols (see Protocol 5 and 6 below). Following this, we present the reduction between these protocols i.e. if a generic protocol of type Protocol 7 is possible then so is the minimal protocol, hence proving the impossibility of obtaining any offline quantum protocol. Finally we extend our argument to the multi-rounds scenario.

These types of protocols are intimately linked to the composability of secure NAND computations in a larger computation ?[e] Note that since, for the second layer of any computation, the client does not know the inputs in advance (since she cannot compute them herself) but knows the encryption of the outputs in advance, thus, quantum offline protocols are necessary and probably sufficient for the composition of NANDs in a larger computation, without requiring additional run-timethe multi-round scenarios communication. The case where run-time communication is allowed will be studied presently. Note also that it does not matter what function, which in tandem with XOR and NOT gates forms a universal set, we use. For simplicity, here we focus on AND. To shorten our expressions, in this section we will be predominantly use ab to denote $a \wedge b$ the logical AND operator of two bits a and b.

The simple quantum offline secure AND computation with two rounds of communication (Simple AND QO2, Protocol 5) is the most natural first attempt, which is inspired by information-theoretic considerations - since the client's input is two bits a and b, hence the quantum state encodes two bits of x and y. Therefore to hide the two bits in the quantum state, additional randomness of two bits r_1 and r_2 is needed.

Recall that the correctness of these protocols are defined by requesting $out = ab$, and blindness is defined by the equation

$$\sum_{\mathbf{x}} \boxed{m(a,b)} \otimes \rho^{\mathbf{x}} = \eta \qquad \forall a, b,$$

where a, b are the input bits, $m(a, b)$ the classical message which may depend on the input, ρ^x a quantum state which depends on some random parameters x (but may also depend on a, b), and η is a positive-semidefinite operator, independent from a, b. For simplicity, we are omitting any normalisation factors, so η may be of non-unit trace.

Lemma 7. *No Simple SecureAND QO2 can be correct and blind.*

Proof. As in any Simple SecureAND QO2 protocol the client sends two classical bits of information to the server (here denoted a', b'), without the loss of generality, we may assume

[d] Note, in contrast, that the blind quantum computing protocol in [4] is quantum-offline, as the initial qubits the client sends are chosen uniformly at random. However, in this protocol, the computing power of the client is beyond just XOR gates.

[e] We do not explicitly address the security issues of composability of our protocols. However, note that our obtained lower bounds on what is possible implies also that the impossibility results will also hold true in any composable security setting.

Protocol 5 Simple SecureAND QO2

The functionality of the Small AND protocol:

- Input (to the client): two bits a, b
- Output (from the client): $(a \wedge b)$
- The Protocol:
  - client's round
    1. client generates a quantum state $\rho^{x,y}_{r_1,r_2}$, characterized by random bits x, y, r_1, r_2 and sends it to the server.
    2. client receives her input bits a, b.
    3. client computes $m_c = (x \oplus a, y \oplus b)$ and sends it to the server.
  - server's round
    1. server performs a (generalized) measurement of $\rho^{x,y}_{r_1,r_2}$, parametrized by m_c. He obtains the outcome m_s and sends it to the client.
  - client's round
    1. client computes $out = m_s \oplus r_1 \oplus r_2$.
    2. client outputs out.

that the message the server returns to the client is a single bit measurement outcome of one of four (generalised) measurements (one for each message (a', b')) which we denote $M^{a',b'}(\rho^{x,y}_{r_1,r_2})$. The correctness of the protocol entails that

$$M^{a',b'}(\rho^{x,y}_{r_1,r_2}) = (a' \oplus x)(b' \oplus y) \oplus r_1 \oplus r_2$$

For clarity we briefly comment on the equation above. Since, for message (a', b') the server performs a generalised two-outcome measurement, this measurement can be represented by the POVM elements $\Pi_0^{a',b'}, \Pi_1^{a',b'}$ (which are positive operators summing to the identity), corresponding to outcomes 0 and 1, respectively. Then the equation above means that

$$Tr(\Pi^{a',b'}_{(a'\oplus x)(b'\oplus y)\oplus r_1\oplus r_2}\rho^{x,y}_{r_1,r_2}) = 1$$

Then, by taking $r = r_1 \oplus r_2$ and defining $\rho^{x,y}_r = 1/2(\rho^{x,y}_{0,r} + \rho^{x,y}_{1,1\oplus r})$ we get, by linearity, that

$$M^{a',b'}(\rho^{x,y}_r) = (a' \oplus x)(b' \oplus y) \oplus r,$$

or equivalently,

$$Tr(\Pi^{a',b'}_{(a'\oplus x)(b'\oplus y)\oplus r}\rho^{x,y}_r) = 1$$

and also that

$$Tr(\Pi^{a',b'}_{(a'\oplus x)(b'\oplus y)\oplus r}\rho^{x,y}_{r\oplus 1}) = 0$$

The two equations above immediately entail that $\rho^{x,y}_r$ and $\rho^{x,y}_{r\oplus 1}$ must be (mixtures of mutually) orthogonal states, which we denote as

$$\rho^{x,y}_r \perp \rho^{x,y}_{r\oplus 1}$$

But, more generally, the equations above imply that two states $\rho_r^{x,y}$ and $\rho_{r'}^{x',y'}$ must be in orthogonal subspaces, whenever any of the sub/superscripts differ. To see this, we will consider the remaining cases separately. First, assume that $r = r'$, but $x \neq x'$ and/or $y \neq y'$. Then if we set $a' = x \oplus 1$ and $b' = y \oplus 1$ we see that

$$M^{a',b'}(\rho_r^{x,y}) = (a' \oplus x)(b' \oplus y) \oplus r = 1 \oplus r$$

but

$$M^{a',b'}(\rho_r^{x',y'}) = (a' \oplus x')(b' \oplus y') \oplus r = r$$

so the outcomes *deterministically* differ, meaning that the two states must be in orthogonal subspaces. We have already seen that the same conclusion follows if $r \neq r'$, and $x = x'$ and $y = y'$. The next case is when $r \neq r'$, and either $x \neq x'$ or $y \neq y'$ (but one is an equality). Assume that $x = x'$, $y \neq y'$ and $r = 0$. Then if we set $a' = x = x'$ we see that

$$M^{x,b'}(\rho_0^{x,y}) = (x \oplus x)(b' \oplus y) = 0$$

and

$$M^{x,b'}(\rho_1^{x,y'}) = (x \oplus x')(b' \oplus y') \oplus 1 = 1$$

Similarly, if $r = 1$ we get opposite results, and if $x \neq x'$ and $y = y'$ we get the same by setting $b' = y = y'$. Finally, we must consider the case when all the parameters differ. First, assume $r = 0$, then by setting $a' = x$ and $b' = 1 \oplus y$ we get

$$M^{a',b'}(\rho_0^{x,y}) = (x \oplus x)(b' \oplus y) = 0$$
$$M^{a',b'}(\rho_1^{x',y'}) = (x \oplus x')(1 \oplus y \oplus y') \oplus 1 = 1$$

since $y \neq y'$, if $r = 1$ then the first equation above would yield 1, and the last would yield 0, since $1 \oplus y \oplus y' = 0$. Thus we can conclude that the states $\{\rho_r^{x,y}\}_{x,y,r}$ are all in orthogonal subspaces. But this means, in particular, that the states $1/4(\sum_{r_1,r_2} \rho_{r_1,r_2}^{x,y})$ are in orthogonal subspaces for all x, y which implies that there exists a measurement which perfectly reveals x and y given any $\rho_{r_1,r_2}^{x,y}$. Thus, the server can perfectly learn x and y and, given the classical message of the client, the inputs of the client, and the protocol is not blind □.

In the above proof we have quickly concluded that the two bits r_1, r_2 are superfluous and one will suffice (which is intuitive as only one random bit is needed to one-time pad the one bit outcome). This gives us the definition of the next general family of protocols (Small AND QO2, Protocol 6) as we describe below and will refer to later.

Lemma 8. *No small SecureAND QO2 can be correct and blind.*

Proof. Obvious from the proof of impossibility of simple AND QO2, where we have actually reduced simple to small protocols □.

3.1 Generalization: QO2

In order to prove a reduction between the general case of Protocol 7 and the simple scenario of Protocol 6 we start with a supposedly given blind and correct QO2 protocol and iteratively

Protocol 6 Small SecureAND QO2

The functionality of the Small AND protocol:

- Input (to the client): two bits a, b
- Output (from the client): $(a \wedge b)$
- The Protocol:
  - client's round
    1. client generates a quantum state $\rho_r^{x,y}$, characterized by random bits x, y, r and sends it to the server.
    2. client receives her input bits a, b.
    3. client computes $m_c = (x \oplus a, y \oplus b)$ and sends it to the server.
  - server's round
    1. server performs a (generalized) measurement of $\rho_r^{x,y}$, parametrized by m_c. He obtains the outcome m_s and sends it to the client
  - client's round
    1. client computes $out = m_s \oplus r$.
    2. client outputs out.

construct a blind correct small QO2, using a sequence of claims which define increasingly simpler protocols.

Theorem 2. *If there exists a blind, correct SecureAND QO2 then there exists a blind correct* Small *SecureAND QO2.*

The objects which appear in the protocol (which differ from the objects in the small QO2) are as follows:

$\rho^{\mathbf{x}}$, with $\mathbf{x} = (x_1, \ldots, x_n)$ – the quantum state parametrized by n bits
$m_c = \mathrm{XOR}_E(a, b, \mathbf{x})$ – the m bit message from the client
m_s, the k bit message from the server
$ab = out = \mathrm{XOR}_D(a, b, \mathbf{x}, m_s)$ – the calculation of the output

Lemma 9. *Nothing is gained from using multi-bit m_s.*

Proof. Note that since the client is restricted to computing XOR operations, we can dissect

$$\mathrm{XOR}_D(a, b, \mathbf{x}, m_s)$$

and see that it must be of the form

$$\mathrm{XOR}_D(a, b, \mathbf{x}, m_s) = \mathrm{XOR}'_D(a, b, \mathbf{x}) \oplus \bigoplus_{j \in I \subseteq [k]} [m_s]_j,$$

where $[m_s]_j$ is the j^{th} bit of the k-bit message m_s. That is, it is a mod 2 addition of something which does not depend on the server's message, and the mod 2 addition of some of the bits

Protocol 7 SecureAND QO2

The functionality of the AND protocol:

- Input (to the client): two bits a, b
- Output (from the client): $(a \wedge b)$
- The Protocol:
  - client's round
    1. client generates a quantum state $\rho^{\mathbf{x}}$, characterised by a sequence of random parameters $\mathbf{x} = (x_1, \ldots, x_n)$, and sends it to the server.
    2. client receives her input bits a, b (the client could have had her bits all along. It is however the defining property of quantum-offline protocols that the parameters $\mathbf{x}$ are independent from a, b).
    3. client computes an XOR-computable function
    $$m_c = \mathrm{XOR}_E(a, b, \mathbf{x})$$
    (E for encryption) of the input and the random parameters. Note that it would be superfluous for the client to generate additional random values at this stage - they could be part of $\mathbf{x}$, without influencing the state the client generates.
    4. client sends m_c to the server.
  - server's round
    1. server performs a (generalized) measurement of $\rho^{\mathbf{x}}$, parametrized by m_c. He obtains the outcome m_s and sends it to the client.
  - client's round
    1. client computes an XOR-computable function
    $$out = \mathrm{XOR}_D(a, b, \mathbf{x}, m_s)$$
    (D for decryption).
    2. client outputs *out*.

of the message responded by the server. Since the form of the message (*i.e* the explicitly description of the function XOR_D) is public, being in the protocol description, the protocol remains secure and correct if the server himself computes the bit $\bigoplus_{j \in I \subseteq [k]} [m_s]_j$, and returns this to the client. Thus, for every correct, blind QO2 there exists a correct blind $\mathrm{QO2}_1$ where the server's message comprises only one bit. The remainder of the claims assumes we are dealing with a $\mathrm{QO2}_1$ protocol □.

Lemma 10. *No random parameters which do not appear in the encryption or decryption are needed.*

Proof. Let $S \subset [n]$ be a subset of indices of the random parameters which appear in

either encryption (as variables of XOR_E) or decryption (XOR_D), and let $S' = [n] \setminus S$ be the subset which does not appear. Then, by exchanging the state $\rho^{\mathbf{x}}$ with the state

$$(\rho')^{\mathbf{x}'} = \sum_{x_j | j \in S'} \frac{1}{2^{|S'|}} \rho^{\mathbf{x}}$$

in a $\mathrm{QO2}_1$ protocol it is easy to see we again obtain a protocol (which we refer to as $\mathrm{QO2}_2$) which is correct and blind. In $\mathrm{QO2}_2$ protocols, all the random parameters appear either in the decryption or encryption. The remainder of the claims assumes we are dealing with a $\mathrm{QO2}_2$ protocol $\square$.

Lemma 11. *No more than one random parameter which appears only in the decryption is needed.*

Proof. Let $S_{D \setminus E} \subset [n]$ be the set of indices of random parameters which appear only in the decryption, that is, as a variable of the function XOR_D. Without the loss of generality, we will assume that the last k indices are such. Then $\mathrm{XOR}_D(a, b, \mathbf{x}, m_s)$ (due to the restrictions on the client) can be written as:

$$\mathrm{XOR}_D(a, b, \mathbf{x}, m_s) = \mathrm{XOR}'_D(a, b, m_s, x_1 \ldots, x_{n-k}) \oplus x_{n-k+1} \oplus \cdots \oplus x_n,$$

Then, by exchanging the state $\rho^{\mathbf{x}}$ with the state

$$(\rho')^{x_1,\ldots,x_{N-k},x} = \sum_{\substack{x_j | j \in S_{D \setminus E} \\ s.t. \\ \oplus_j x_j = x}} \frac{1}{2^{|S_{D \setminus E}|-1}} \rho^{\mathbf{x}}$$

in a $\mathrm{QO2}_2$ protocol we again obtain a protocol (which we refer to as $\mathrm{QO2}_3$) which is correct and blind. Blindness is trivial, as the sum over all the random parameters for the state $\rho^{\mathbf{x}}$ yields the same density operator as the sum over all random parameters for the state $(\rho')^{x_1,\ldots,x_{n-k},x}$ (and no message correlated to the summed up random parameters is sent from the client to the server). Correctness holds as the correctness of the (original) $\mathrm{QO2}_2$ protocol only depended on the parity of the k random parameters, and the construction above preserves this $\square$.

In $\mathrm{QO2}_3$ protocols, at most one random parameter appears in the decryption only. The remainder of the claims assumes we are dealing with a $\mathrm{QO2}_3$ protocol.

Lemma 12. *The client's input bits a and b do not need to appear in the decryption function.*

Proof. In general the decryption function of the client (for $\mathrm{QO2}_3$) protocols attains the form

$$\mathrm{XOR}_D(a, b, \mathbf{x}, m_s) = \mathrm{XOR}'_D(a, b, m_s) \oplus \bigoplus_{j \in S_{E \cap D}} x_j \oplus x_n \text{ or}$$
$$\mathrm{XOR}_D(a, b, \mathbf{x}, m_s) = \mathrm{XOR}'_D(a, b, m_s) \oplus \bigoplus_{j \in S_{E \cap D}} x_j$$

where $S_{E \cap D}$ is the set of indices of random parameters which appear in both the decryption and encryption function, and x_n may appear only in the decryption function. Here, we have assumed without the loss of generality that it is the last random parameter that (possibly)

appears only in the decryption function. First, we show that at least one random parameter must appear in the decryption, meaning that either x_n must appear or $S_{E\cap D}$ is non-empty (or both). Assume this is not the case. Then we have

$$\mathrm{XOR}_D(a,b,\mathbf{x},m_s)=\mathrm{XOR}'_D(a,b,m_s)$$

and this must be equal to ab by the correctness of the protocol. But, due to the restrictions of the client we have

$$\mathrm{XOR}'_D(a,b,m_s)=\mathrm{XOR}''_D(a,b)\oplus m_s=ab \text{ or}$$
$$\mathrm{XOR}'_D(a,b,m_s)=\mathrm{XOR}''_D(a,b)=ab$$

The latter is not possible as no function computable using only XOR can yield the output ab, so

$$\mathrm{XOR}'_D(a,b,m_s)=\mathrm{XOR}''_D(a,b)\oplus m_s=ab\Leftrightarrow$$
$$m_s=ab\oplus\mathrm{XOR}''_D(a,b).$$

The function $\mathrm{XOR}''_D(a,b)$ can only be one of six functions, which are such that either a or b appear in the decryption:

$$\mathrm{XOR}''_D(a,b)=a;\ \mathrm{XOR}''_D(a,b)=1\oplus a$$
$$\mathrm{XOR}''_D(a,b)=b;\ \mathrm{XOR}''_D(a,b)=1\oplus b;$$
$$\mathrm{XOR}''_D(a,b)=a\oplus b;\ \mathrm{XOR}''_D(a,b)=1\oplus a\oplus b.$$

But, for all of these functions we have that $ab\oplus\mathrm{XOR}''_D(a,b)$ is correlated to a,b, hence not blind. For example $a\oplus b\oplus ab=a\vee b$, so if the server obtains $m_s=0$ this means $a=b=0$. Thus, for the protocol to be blind, at least one random parameter must appear in the decryption.

Let j be the index of this random parameter. Then x_j either appears or does not appear in the encryption. First assume x_j appears in the encryption, and let $\mathrm{XOR}''_D(a,b)=a$. Then by modifying XOR_D in such a way that it no longer depends on a (by substituting $\mathrm{XOR}''_D(a,b)$ with 0 in the definition of XOR_D) and by modifying the encryption function in such a way that all instances of x_j are substituted with $x_j\oplus\mathrm{XOR}''_D(a,b)$, we obtain a new protocol, in which the inputs a,b no longer appear in the decryption function. This protocol is correct, as the initial protocol was correct for all possible inputs and random variables, and all we have done is a substitution of variables. Since, from the perspective of the server, both $x_j\oplus\mathrm{XOR}''_D(a,b)$ and x_j are equally distributed (uniformly at random), the protocol is blind as well.

Consider now the case where x_j does not appear in the encryption (thus no random parameters appearing in the encryption appear in the decryption), and let $\mathrm{XOR}''_D(a,b)$ be the function which appears in the evaluation of the decryption, and is not constant. Then, we need to modify the messages the client sends, and the measurement the server does. Let m_c be the message the client sends in the original protocol. Then, in the modified protocol, the client will send the message $(m_c,\mathrm{XOR}''_D(a,b)\oplus y)$, where y is a new random bit. The server will perform the same measurement as in the original protocol, as defined by m_c but will output $m_s^{new}=m_s^{original}\oplus\mathrm{XOR}''_D(a,b)\oplus y$. Note that this process can be viewed as a redefinition of the measurement the server does. the client decrypts almost the same as in the original protocol, altered by substituting $\mathrm{XOR}''_D(a,b)$ with 0, and by XORing with y . So

we have:

The original decryption in original protocol :
$out = \mathrm{XOR}''_D(a,b) \oplus m_s^{original} \oplus x_j$
The new decryption in new protocol :
$0 \oplus m_s^{new} \oplus x_j \oplus y = m_s^{original} \oplus \mathrm{XOR}''_D(a,b) \oplus y \oplus x_j \oplus y = out.$

Thus, the new protocol is also correct. To see that it is blind, note that the only piece of additional information given to the server, relative to the original protocol is the bit $\mathrm{XOR}''_D(a,b) \oplus y$. However, since y is chosen uniformly at random, this reveals no extra information so the protocol is blind as well.

Thus for every $\mathrm{QO2}_3$ blind correct protocol, there exists a blind correct $\mathrm{QO2}_4$ protocol where the inputs of the client do not appear in the decryption function $\square$.

To summarise, to this point we have shown that we only need to consider protocols in which the server's output is a single bit, at most one random parameter which appears in the decryption (but not in encryption) is used, and the decryption function does not take the inputs of the client as parameters. Additionally we have shown that we only need the random parameters which appear either in encryption or decryption. Next, we deal with the size of the client's messages, and the number of required random parameters appearing in the encryption.

Consider the encryption, and the generated quantum state in the protocol:

$m_c = \mathrm{XOR}_E(a,b,\mathbf{x})$ $-$ the m bit message from the client
$\rho^{\mathbf{x}},\ for\ \mathbf{x} = (x_1,\ldots,x_n)$ $-$ the quantum state parametrized by n bits.

and let $(m_c)_j$ denote the j^{th} bit of the m bit message m_c.

Lemma 13. *No single isolated random variables are needed.*

Proof. Assume that, for some j and k we have, $(m_c)_j = x_k$. Then, the protocol reveals x_k. But this means that if we fix $x_k = 0$ (that is, by dropping that random parameter from the protocol) we yield again a blind correct protocol (with one less random parameter). We get the same if the negation of x_k appears. By repeating this, we obtain a protocol for which no part of the message is equal to a single random parameter, or its negation $\square$.

Lemma 14. *No arbitrary XOR functions of random variables are needed.*

Proof. Next, assume that for some j and k,l we have, $(m_c)_j = x_k \oplus x_l$. Then, we can introduce the variable $x_{k,l} = x_k \oplus x_l$, and substitute all instances of x_l in the protocol with $x_{k,l} \oplus x_l$. This again yields a correct blind protocol, with the same number of random parameters as the original protocol. However, the modified protocol has the new variable $x_{k,l}$ appearing in $(m_c)_j$ isolated, so it (by the argument in the last paragraph) be dropped from the protocol.

We can perform analogous substitutions whenever arbitrary XOR functions of random parameters appear in isolation: for a function $b \oplus x_{k_1} \oplus \cdots \oplus x_{k_p}$ we can define the substituting variable $x^b_{k_1,\ldots,k_p} = b \oplus x_{k_1} \oplus \cdots \oplus x_{k_p}$, and substitute all instances of x_{k_1} with $x^b_{k_1,\ldots,k_p} \oplus b \oplus$

$x_{k_2} \oplus \cdots x_{k_p}$. Thus we retain exactly the same number of random parameters, but $x^b_{k_1,\ldots,k_p}$ now appears in isolation. So, this variable can be dropped.

Thus, for any $\mathrm{QO2}_4$ protocol, there exists a protocol (blind and correct) where no functions of random parameters appear in isolation in m_c.

Thus, each entry of m_c is of the form $\mathrm{XOR}(a, b, x_1, \ldots x_n)$, where this function is not constant in a or b (or both). However, it is clear that this function cannot be constant in all the random parameters $\mathbf{x}$ as otherwise the protocol would not be blind $\square$.

We can now complete the main proof of the impossibility of quantum offline protocol by showing how the redundancies could be removed.

Proof of Theorem 2. Define

$$(m_c)_j = \mathrm{XOR}(a, b) \oplus \bigoplus_{k \in S_j \subseteq [N]} x_k$$
$$(m_c)_{k \neq j} = \mathrm{XOR}(a, b) \oplus \bigoplus_{k \in S_k \subseteq [N]} x_k$$

Then, the XOR of those two entries reveals the XOR of the random parameters with indices in the intersection $S_j \cap S_k$. Let

$$\tilde{x} = \bigoplus_{k \in S_j \subseteq [N]} x_k \oplus \bigoplus_{k \in S_k \subseteq [N]} x_k = \bigoplus_{k \in S_k \cap S_j \subseteq [N]} x_l$$

Then the original protocol is equally blind as the protocol (we will call it MOD1 for modification 1) in which the message element $(m_c)_k$ is substituted with $\tilde{x}$ and the server, upon the receipt of the message redefines $(m_c)_k := (m_c)_j \oplus x$.

For simplicity, assume that $S_k \cap S_j = \{1, 2, \ldots l\}$. If we further modify MOD1 to MOD2 by substituting all instances of x_1 in this protocol with $\tilde{x} \oplus x_2 \ldots x_l$ we obtain a protocol in which $\tilde{x}$ is a randomly chosen variable, and note that it appears isolated in message element $(m_c)_k$. Thus, it can by the arguments we presented earlier, be dropped from the protocol, by setting it to zero. Note that analogous transformations of the protocol can be done if the XOR functions on two positions differ by a bit flip.

Hence, we only need to consider protocols where each function of a, b in the message of the client appears only once, where functions which differ by a bit flip can be considered duplicates as well. There are only three XOR computable non-constant functions of two binary parameters, up to a bit flip:

$$\mathrm{XOR}(a, b) = a,\ \mathrm{XOR}(a, b) = b,\ \mathrm{XOR}(a, b) = a \oplus b$$

Thus, the message the client sends to the server, without the loss of generality, is of the form:

$$m_c = (a \oplus \bigoplus_{k \in S_1 \subseteq [n]} x_k, b \oplus \bigoplus_{k \in S_2 \subseteq [n]} x_k, a \oplus b \oplus \bigoplus_{k \in S_3 \subseteq [n]} x_k)$$

Now, we can eliminate any single one of the three, and for our purposes of reduction to the small QO2 protocol, we will eliminate the last one. Note that

$$(m_c)_3 = (m_c)_1 \oplus (m_c)_2 \oplus \bigoplus_{k \in S_1 \subseteq [n]} x_k \oplus \bigoplus_{k \in S_2 \subseteq [N]} x_k \oplus \bigoplus_{k \in S_3 \subseteq [N]} x_k,$$

and that the server can obtain

$$\tilde{x} = \bigoplus_{k \in S_1 \subseteq [n]} x_k \oplus \bigoplus_{k \in S_2 \subseteq [n]} x_k \oplus \bigoplus_{k \in S_3 \subseteq [n]} x_k$$

by XORing the three bits of the client's message. Thus, similarly to the approach we used earlier, the protocol can be further modified in such a way that $\tilde{x}$ is given as the third bit of the message. Furthermore, by substitution, the third bit can be eliminated as well. Thus we obtain the third modification of the protocol, in which the client's message is of the form

$$m_c = (a \oplus \bigoplus_{k\in S_1\subseteq[n]} x_k, b \oplus \bigoplus_{k\in S_2\subseteq[n]} x_k)$$

with $S_1 \cup S_2 = [n]$. Note $S_1 \neq S_2$ as otherwise the protocol would not be blind. Let S_{DE} be the subset of indices of the random parameters which appear in the decryption and encryption. Then all the random parameters in $S_1 \setminus (S_2 \cup S_{DE})$ can be substituted by only one random parameter $\tilde{x}_1$ which is the mod 2 sum of random parameters indexed in $S_1 \setminus (S_2 \cup S_{DE})$. Additionally, the quantum state the client sends to the server needs to be averaged over all states where the mod 2 sum of random parameters indexed in $S_1 \setminus (S_2 \cup S_{DE})$ is zero (for $\tilde{x}_1 = 0$) and for the case it is one (for $\tilde{x}_1 = 1$). The same can be done for all the random parameters in $S_2 \setminus (S_1 \cup S_D E)$, generating the single random parameter $\tilde{y}_1$ appearing only in $(m_c)_2$. The indices in S_{DE} must appear either in S_1 or in S_2. Let $p_1 \dots p_q$ be the set which appears in both. Then we can substitute these random parameters with one $\tilde{p} = p_1 \oplus \cdots \oplus p_q$ by again modifying the state the client sends to the server, by averaging over those states for which $p = 0$ or $p = 1$. Similarly can be done for those indices in S_{DE} which appear only in $(m_c)_1$ (same for $(m_c)_2$) resulting in one random parameter $\tilde{x}_2$ ($\tilde{y}_2$). Thus we obtain the protocol in which the client sends

$$m_c = (a \oplus \tilde{x}_1 \oplus \tilde{x}_2 \oplus p, b \oplus \tilde{y}_1 \oplus \tilde{y}_2 \oplus p)$$

and the decryption is given with:

$$out = m_s \oplus \tilde{x}_2 \oplus \tilde{y}_2 \oplus p \oplus r$$

where r was the random parameter not appearing in the encryption, and the quantum state is parametrized with:

$$\rho^{\tilde{x}_1,\tilde{x}_2,\tilde{y}_1,\tilde{y}_2,p,r}$$

We will refer to such protocols as $\mathrm{QO2}_5$ protocols. Note that

$$M^{\alpha,\beta}(\rho^{\tilde{x}_1,\tilde{x}_2,\tilde{y}_1,\tilde{y}_2,p,r}) = (\alpha \oplus \tilde{x}_1 \oplus \tilde{x}_2 \oplus p)(\beta \oplus \tilde{y}_1 \oplus \tilde{y}_2 \oplus p) \oplus \tilde{x}_2 \oplus \tilde{y}_2 \oplus p \oplus r$$

and equivalently that

$$M^{\alpha,\beta}(\rho^{\tilde{x}'_1,\tilde{x}'_2,\tilde{y}'_1,\tilde{y}_2,p',r'}) = (\alpha \oplus \tilde{x}'_1 \oplus \tilde{x}'_2 \oplus p')(\beta \oplus \tilde{y}'_1 \oplus \tilde{y}'_2 \oplus p') \oplus \tilde{x}'_2 \oplus \tilde{y}'_2 \oplus p' \oplus r'.$$

Therefore we obtain the following relation:

$$\begin{aligned} &M^{\alpha,\beta}(\rho^{\tilde{x}_1,\tilde{x}_2,\tilde{y}_1,\tilde{y}_2,p,r}) = M^{\alpha,\beta}(\rho^{\tilde{x}'_1,\tilde{x}'_2,\tilde{y}'_1,\tilde{y}'_2,p',r'}) \; if \\ &\tilde{x}_1 \oplus \tilde{x}_2 \oplus p = \tilde{x}'_1 \oplus \tilde{x}'_2 \oplus p', \; and \\ &\tilde{y}_1 \oplus \tilde{y}_2 \oplus p = \tilde{y}'_1 \oplus \tilde{y}'_2 \oplus p' \; and \\ &\tilde{x}_2 \oplus \tilde{y}_2 \oplus p \oplus r = \tilde{x}'_2 \oplus \tilde{y}'_2 \oplus p' \oplus r'. \end{aligned}$$

Since the state ρ is parametrized by 6 independent parameters and we have three independent equations, this implies that there are 8 differing equivalency classes (as defined by the three

equalities) over the set of all possible random parameters. The equivalency classes can be represented by three bits c_1, c_2, c_3 as follows:

$$(c_1, c_2, c_3) \equiv \{(\tilde{x}_1, \tilde{x}_2, \tilde{y}_1, \tilde{y}_2, p, r) | \tilde{x}_1 \oplus \tilde{x}_2 \oplus p = c_1 \\ \tilde{y}_1 \oplus \tilde{y}_2 \oplus p = c_2, \tilde{x}_2 \oplus \tilde{y}_2 \oplus p \oplus r = c_3\}$$

We can then define the states ρ, averaged per equivalency class:

$$\rho^{c_1,c_2,c_3} = 1/8 \sum_{(\tilde{x}_1,\tilde{x}_2,\tilde{y}_1,\tilde{y}_2,p,r)\in(c_1,c_2,c_3)} \rho^{\tilde{x}_1,\tilde{x}_2,\tilde{y}_1,\tilde{y}_2,p,r}$$

Note that the first bit of the message the client sends to the server in $\mathrm{QO2}_5$ is given with $(a \oplus x_1 \oplus x_2 \oplus p)$ which is equal to c_1. Similarly, the second bit $(b \oplus y_1 \oplus y_2 \oplus p)$ is equal to c_2. The decryption is given with $out = m_s \oplus x_2 \oplus y_2 \oplus p \oplus r$ which is equal to $m_s \oplus c_3$. This gives us a protocol in which the client sends

$$m_c = (a \oplus c_1, b \oplus c_2)$$

and the decryption is given with:

$$out = m_s \oplus c_3$$

where c_3 was the random parameter not appearing in the encryption, and the quantum state is parametrized with:

$$\rho^{c_1,c_2,c_3}$$

This protocol is correct by construction, and it is also blind as the classical messages the client sends are the same as in the $\mathrm{QO2}_5$ protocol, and the quantum state is averaged over the degrees of freedom which do not appear in the abbreviated protocol - but then the averaging over the remaining free parameters yields the same state on the server's side as in the $\mathrm{QO2}_5$ protocol. Thus it is blind as well.

But this is also a small QO2 protocol. Thus, symbolically, we have shown:

$$\exists \mathrm{QO2} \rightarrow \exists \mathrm{QO2}_1 \rightarrow \exists \mathrm{QO2}_2 \rightarrow \exists \mathrm{QO2}_3 \rightarrow \exists \mathrm{QO2}_4 \rightarrow \exists \mathrm{QO2}_5 \rightarrow \exists \text{ small } \mathrm{QO2}$$

which implies the proof of the main theorem since we have already proven no small QO2 protocol exists □.

3.2 Multi rounds

In the definition of QO2 protocols, we have explicitly demanded that client and server use only two rounds of classical communication to achieve the desired functionality. That is, after the quantum offline preparation stage, client sends one classical message to server, to which server responds. This offers the possibility that including multiple rounds of classical communication may circumvent the no-go result of Theorem 2. Now we show that this is not the case. To develop the proper intuition, first consider the very first possible extension - that the protocol ends with client sending an extra message to server. This trivially cannot help, as client's output then cannot depend on whatever server does. Next consider the case where client is allowed to send an additional message to server, to which server responds. To clarify this case, we shall use the notation of Protocol 7 (SecureAND QO2). Consider

the client's round in protocol 7 in which the client computes the output *out*. In the most general setting of a 4 round, instead of computing an output, client stores server's message m_s, computes some XOR_{E2} function of m_s a, b and perhaps new random parameters, which she then forwards back to server. Denote this message m_c^2. Note that the function XOR_{E2} is specified by the protocol (that is, it is known to server), and that it can only be a combination of XOR functions and negations of its arguments. It can also be a multi-bit XOR function: if $m_c^2 = (y_1, \ldots, y_k)$ is a $k-$ bit message, each bit y_l is of the form $XOR_{E2_l}(a, b, m_s, \mathbf{x}, r)$ (where r are random bits), where some of the arguments may appear with a negation, or may not appear at all. Then each bit y_l of m_c^2 can be written as

$$XOR_{E2_l}(a, b, m_s, \mathbf{x}, r) = XOR_{E2_l'}(a, b, \mathbf{x}, r) \oplus XOR_{E2_l''}(m_s) \tag{7}$$

where we have just separated the function into parts which depend on m_s an which do not. We can do this since all operations that the client can do, commute. Now, since the server knows all the component functions and m_s, and since the protocol is by assumption blind, the part $XOR_{E2_l'}(a, b, \mathbf{x}, r)$ must be independent from a, b, when averaged over $\mathbf{x}$ and r, otherwise it would reveal information to server. We emphasize two properties of $XOR_{E2_l'}(a, b, \mathbf{x}, r)$. First it does not depend on m_s, and second it does not reveal anything about the input. Hence, client could have sent $XOR_{E2_l'}(a, b, \mathbf{x}, r)$ as a part of m_c, and the protocol would be extended by having server compute

$$XOR_{E2_l}(a, b, m_s, \mathbf{x}, r) = XOR_{E2_l'}(a, b, \mathbf{x}, r) \oplus XOR_{E2_l''}(m_s) \tag{8}$$

himself, after his measurement, without jeopardizing blindess or correctness. This shows that any 4 round blind and correct quantum offline protocol can be reduced to a two round protocol. However, this argument trivially generalizes: in an $2n$ round protocol, at client's k^{th} interaction step, any computation client does on server's prior responses, the inputs and random parameters, can be split into parts which depend on server's input and those which do not. Since server knows his responses, and the protocol is blind, the parts which do not depend on server's input cannot reveal any information about client's inputs, and also (by definition) do not depend on server's responses. Hence, client could have sent all of them in the first round of communication, and delegate the computations to server while maintaining security and correctness. This shows that any blind and correct, quantum offlice, $n-$round SecureAND protocol (denoted QOn) implies the existence of $QO2$. Since we have shown that the latter is impossible, so are QOn protocols. This proves our ultimate no-go result.

Theorem 3. *Blind, correct quantum-offline, $n-$round SecureAND protocols are impossible for every number of rounds n.*

4 Conclusion

In this work, we have considered the problem of secure quantum delegated classical computation, in the setting of a client with minimal computational capabilities. In particular, the client we consider is, on the classical side, restricted only to XOR operations and random bit generation. This is arguably a minimal setting for the client where security can be obtained - XOR gates and random bits suffice for a one-time pad of the classical information of the

client, and, at least for the simpler task of transmitting of confidential information, both are neccesary.

We have first shown that no fully classical delegated computation protocol can enable such a client to compute the NAND gate on two bits (which is necessary for universal classical computation). Following this, inspired by the results in [17], we have shown through a family of protocols, how various types of minimal quantum capabilities on the side of the client and server do allow for the computation of the NAND gate. The simplest protocol only requires the client to prepare a single qubit state - however, the state of the qubit depends on the inputs of the client. That is, the required quantum state cannot be prepared before the input is known to the client, and thus the protocol is not quantum-offline. We have then proven that this is not a restriction of our approach, but rather that quantum-offline SecureNAND protocols are impossible. This stands in contrast to protocols for secure delegated quantum computation, where quantum-offline protocols are possible [4], while the cost of achieving that is the necessity of the client to perform non-linear operations (beyond the XOR gate).

The motivation for the setting we consider is predominantly theoretical, and serves to investigate the scenario in which quantum effects can help in computational problems, and, equally importantly, how. In our setting, the client effectively exploits the additional freedom of a single-qubit system, namely, to be in a coherent superposition of two states (achieved in the temporal sequence of gates the client applies), to obtain the outcome of the computation. Alternatively, in the GHZ-based settings, it is the entanglement which helps achieve the required input-output correlations obtained by measurements on three distinct qubits. This opens up the question of what fundamental properties or resources of quantum theory allow for the shown enhancement. Indeed our demonstration of the potential power of quantum communication in the setting with a limited-client and untrusted-server seems to be closely related to the work of [19] on the power of contextuality. In the latter it was shown that a necessary resource for a multi-party quantum computation of any non-linear function where each party could only perform classical linear gates (such as XOR) together with local quantum operations is contextuality. Further work in [20] studies the optimality of non-local resources, e.g. generalized GHZ states, in winning multi-party quantum computation games in a related setting. We leave as a future work to explore further this connection to extend our result to multi-party settings and for the general function evaluation.

Nonetheless, we do not preclude the possibility that the results of this work may have a practical impact - the cost of XOR or NAND operations for a client depends on the setting. In particular, in the case of computation over encrypted data (homomorphic encryption), the solution for XOR-only computation is simple. In contrast, NAND-based, universal computation over encrypted data is notoriously hard. It is only recently that computationally secure solutions have been found for this general problem (fully homomorphic encryption) [16], and thus far, the proposed protocols are still impractical. It is thus conceivable that hybrid approaches to secure delegated computation may be possible, raising the security (or reducing the complexity) of classical schemes, at the price of a small amount of quantum capabilities. The possibility of this, based on the approaches we have presented in this work, is a part of ongoing research.

References

1. R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. *Found. Secure Computation*, 1978.
2. A. Childs. Secure assisted quantum computation. *Quant. Inf. Comput.*, 5, 2005.
3. P. Arrighi and L. Salvail. Blind quantum computation. *Int. J. Quant. Inf.*, 4, 2006.
4. A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computing. In *FOCS*, 2009.
5. D. Aharonov, M. Ben-Or, and E. Eban. Interactive proofs for quantum computations. In *ICS*, 2010.
6. T. Morimae, V. Dunjko, and E. Kashefi. Ground state blind quantum computation on aklt state. arXiv:1009.3486, 2011.
7. T. Morimae and K. Fujii. Blind topological measurement-based quantum computation. *Nature communications*, 3, 2012.
8. V. Dunjko, E. Kashefi, and A. Leverrier. Blind quantum computing with weak coherent pulses. *Phys. Rev. Lett.*, 108, 2012.
9. T. Morimae and K. Fujii. Blind quantum computation for alice who does only measurements. *Phys. Rev. A*, 87, 2013.
10. A. Mantri, C. Perez-Delgado, and J. Fitzsimons. Optimal blind quantum computation. *Phys. Rev. Lett.*, 111, 2013.
11. V. Giovannetti, L. Maccone, T. Morimae, and T. Rudolph. Efficient universal blind quantum computation. *Phys. Rev. Lett.*, 111, 2013.
12. B. Reichardt, F. Unger, and U. Vazirani. Classical command of quantum systems. *Nature*, 496, 2013.
13. K. Fisher, A. Broadbent, L. Shalm, Z. Yan, J. Lavoie, R. Prevedel, T. Jennewein, and K. Resch. Quantum computing on encrypted data. *Nature communications*, 5, 2014.
14. Andrew C. Yao. Protocols for secure computations. In *FOCS*, 1982.
15. S. Goldwasser. Multi party computations: past and present. In *PODC*, 1997.
16. C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, 2009.
17. J. Anders and D. E. Browne. Computational power of correlations. *Phys. Rev. Lett.*, 102, 2009.
18. K. Loukopoulos and D. E. Browne. Secure multiparty computation with a dishonest majority via quantum means. *Phys. Rev. A*, 81, 2010.
19. R. Raussendorf. Contextuality in measurement-based quantum computation. *Physical Review A*, 88(2), 2013.
20. M. J. Hoban, E. T. Campbell, K. Loukopoulos, and D. E. Browne. Non-adaptive measurement-based quantum computation and multi-party bell inequalities. *New Journal of Physics*, 13(2):023014, 2011.
21. S. Barz, V. Dunjko, F. Schlederer, M. Moore, E. Kashefi, and I. A. Walmsley. Secure delegated classical computing exploiting coherence. arXiv:1501.06730, 2015.
22. Janet Anders and Dan E. Browne. Computational power of correlations. *Phys. Rev. Lett.*, 102:050502, Feb 2009.