2018 IEEE 59th Annual Symposium on Foundations of Computer Science

Classical Homomorphic Encryption for Quantum Circuits

Urmila Mahadev
Department of Computer Science, UC Berkeley
mahadev@berkeley.edu

***Abstract*—We present the first leveled fully homomorphic encryption scheme for quantum circuits with classical keys. The scheme allows a classical client to blindly delegate a quantum computation to a quantum server: an honest server is able to run the computation while a malicious server is unable to learn any information about the computation. We show that it is possible to construct such a scheme directly from a quantum secure classical homomorphic encryption scheme with certain properties. Finally, we show that a classical homomorphic encryption scheme with the required properties can be constructed from the learning with errors problem.**

I. Introduction

Can a classical client delegate a desired quantum computation to a remote quantum server while hiding all data from the server? Quantum secure classical encryption schemes do not immediately answer this question; they provide a way of hiding data, but not of computing on the data. This question is particularly relevant to proposals for quantum computing in the cloud.

The classical analogue of this task, in which the client is a weak classical machine and the server is more powerful (but still classical), was solved in 2009 with the celebrated construction of homomorphic encryption schemes ([1]). Unfortunately, these schemes are built only to handle classical computations on the encrypted data; the prospect of applying computations in superposition over encrypted bits seems to be much more difficult. This difficulty arises from the fact that all classical homomorphic encryption schemes require (for security) that each bit has many possible different encryptions. This property appears to preclude the quantum property of interference: interference requires that elements of a superposition representing the same bit string, but with opposite amplitudes, cancel out. If these elements have different encryptions, interference cannot happen, thereby preventing one of the key advantages of quantum algorithms.

Due to these obstacles, the question of quantum homomorphic encryption was weakened by allowing a quantum client ([2]). This variant has been well studied in recent years ([2], [3], [4], [5], [6], [7]) and has led to advancements in the field of delegated quantum computing. However, the model has a number of shortcomings. The principal issue is that the quantum client relies on quantum evaluation keys, which are not reusable; the client must generate fresh keys each time he wishes to delegate a quantum computation. Unfortunately, existing protocols ([7]) require the client to generate a number of quantum keys proportional to the size of the quantum circuit being applied.[1]

The related question of blind quantum computation predated the question of quantum homomorphic encryption and has also been extensively studied, beginning with [8]. Blind quantum computation and quantum homomorphic encryption have the same goal of carrying out a computation on encrypted data, but blind computation allows multiple rounds of interaction between the client and server, while homomorphic encryption allows only one round of interaction. Even in the weaker model of blind computation, a quantum client has been a necessity so far (at a minimum, the client must be able to prepare certain constant qubit states [9]/[10]).

In this paper, we return to the original question of quantum homomorphic encryption by providing a homomorphic encryption scheme for quantum computations with a classical client. To do this, we show that certain classical homomorphic encryption schemes can be lifted to the quantum setting; they can be used in a different way to allow for homomorphic evaluation of quantum circuits. It follows that all properties of classical homomorphic encryption schemes (such as reusability of keys and circular security) also hold in the quantum setting. This scheme is the first (and currently only) to allow blind quantum computation between a classical client and a quantum server.[2]

To build our homomorphic encryption scheme, we begin with the fact that blindly computing a quantum circuit can be reduced to the ability of a quantum server to perform a CNOT gate (a reversible XOR gate) controlled by classically encrypted data. More specifically, we need a procedure which takes as input the classical encryption of a bit s, which we denote as $\mathrm{Enc}(s)$, a 2 qubit state $|\psi\rangle = \sum_{a,b\in\{0,1\}} \alpha_{ab} |a,b\rangle$ and outputs the following state:

$$\mathrm{CNOT}^s |\psi\rangle = \sum_{a,b\in\{0,1\}} \alpha_{ab} |a, b \oplus a \cdot s\rangle \qquad (1)$$

[1]The client is still restricted in some sense (in comparison to the server); for example, the client may not need to run a general BQP circuit or may only require a constant sized quantum register.

[2]There have been two papers ([11], [12]) proposing delegated blind quantum computation protocols between a classical client and a quantum server. Both of these results differ from ours in that they do not claim security (i.e. blindness) against a malicious quantum server for the delegation of general quantum computations. Moreover, both results require multiple rounds of interaction between the client and the server.

2575-8454/18/$31.00 ©2018 IEEE
DOI 10.1109/FOCS.2018.00039

We call this procedure an encrypted CNOT operation. Of course, the output state $\text{CNOT}^s |\psi\rangle$ will have to be suitably encrypted (to avoid revealing s); this is accomplished by the Pauli one time pad, which we will describe a bit later in this introduction. The key step of the encrypted CNOT operation is the extraction of a classically encrypted bit into a quantum superposition. We now show how this extraction can be done by relying on the classical cryptographic primitive of trapdoor claw-free function pairs.

A trapdoor claw-free function pair is a pair of injective functions f_0, f_1 which have the same image, are easy to invert with access to a trapdoor, and for which it is computationally difficult to find any pair of preimages (x_0, x_1) with the same image ($f_0(x_0) = f_1(x_1)$). Such a pair of preimages is called a claw, hence the name claw-free. These functions are particularly useful in the quantum setting, due to the fact that a quantum machine can create a uniform superposition over a random claw (x_0, x_1): $\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$. This superposition can be used to obtain information which is conjectured to be hard to obtain classically: the quantum machine can obtain *either* a string $d \neq 0$ such that $d \cdot (x_0 \oplus x_1) = 0$ or one of the two preimages x_0, x_1. In [13], this advantage was introduced and used to create a scheme for verifiable quantum supremacy and to show how to generate information theoretic randomness from a single quantum device. This advantage was further used in [14] to classically verify general quantum computations.

Here, we show that if a bit s is encrypted using a trapdoor claw-free function pair $f_0, f_1 : \{0,1\} \times \mathcal{R} \to \mathcal{Y}$, it can easily be stored in superposition, as required for the encrypted CNOT operation. First, we need to be precise about the encryption: we say that the function pair f_0, f_1 encrypts the bit s if each claw $(\mu_0, r_0), (\mu_1, r_1)$ (for $\mu_0, \mu_1 \in \{0,1\}, r_0, r_1 \in \mathcal{R}$) hides the bit s as follows: $\mu_0 \oplus \mu_1 = s$. We also make a simplifying assumption, for the purposes of this introduction, regarding the 2 qubit state $|\psi\rangle$ to which the encrypted CNOT operation will be applied: we assume that the second qubit of the state $|\psi\rangle$ is fixed to 0 (in other words, $|\psi\rangle$ can be written as $\sum_{a\in\{0,1\}} \alpha_a |a, 0\rangle$). This assumption allows us to highlight the key step of the encrypted CNOT operation, which is the extraction of an encrypted bit into a superposition, and allows us to replace (1) with:

$$\text{CNOT}^s |\psi\rangle = \sum_{a\in\{0,1\}} \alpha_a |a, a \cdot s\rangle \tag{2}$$

Given the special form of the trapdoor claw-free encryption, conversion to a superposition is quite straightforward. Using the encryption f_0, f_1, the prover can entangle the state $|\psi\rangle$ with a random claw, creating the following state:

$$\sum_{a\in\{0,1\}} \alpha_a |a\rangle |\mu_a, r_a\rangle = \sum_{a\in\{0,1\}} \alpha_a |a\rangle |\mu_0 \oplus a \cdot s, r_a\rangle \tag{3}$$

Observe that the bit s is now stored in superposition, in essentially the form required in (2), although it is hidden (due to μ_0, r_0, r_1). As mentioned earlier, the output of the encrypted CNOT operation must be suitably encrypted, and μ_0, r_0, r_1 will eventually be part of this encryption, as we will see shortly. To briefly give some intuition about these parameters, note that μ_0 serves as a classical one time pad to hide the bit s, and r_0, r_1 are required since the bit s is in superposition; therefore, just a classical one time pad does not suffice.

We now return to the encryption scheme we will use: as in previous blind quantum computation schemes, our homomorphic encryption scheme relies on the Pauli one time pad encryption scheme for quantum states ([15]), which is the quantum analogue of the classical one time pad. Recall the one time pad method of classical encryption: to encrypt a string m, it is XORed with a random string r. Just as l classical bits suffice to hide an l bit string m, $2l$ classical bits suffice to hide an l bit quantum state $|\psi\rangle$. The Pauli one time pad requires the Pauli X and Z operators, defined as follows:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{4}$$

A single qubit state $|\psi\rangle$ is information theoretically encrypted by choosing bits $z, x \in \{0,1\}$ (called the Pauli keys) at random and applying the Pauli one time pad $Z^z X^x$, creating the following encryption:

$$Z^z X^x |\psi\rangle \tag{5}$$

To convert this encryption to a computationally secure encryption, the encrypted Pauli keys are included as part of the encryption (the resulting encryption is two part, consisting of the quantum state in (5) and classically encrypted bits $\text{Enc}(z), \text{Enc}(x)$).

Given the Pauli one time pad, we can now precisely state the goal of our encrypted CNOT operation described above. The encrypted CNOT operation is a quantum operation which takes as input $\text{Enc}(s)$ and a 2 qubit state $|\psi\rangle$ and outputs the state $Z^z X^x \text{CNOT}^s |\psi\rangle$ as well as classical encryptions of $z, x \in \{0,1\}^2$.[3] Recall the final state of the encrypted CNOT operation (as given in (3)):

$$\sum_{a\in\{0,1\}} \alpha_a |a\rangle |\mu_0 \oplus a \cdot s, r_a\rangle \tag{6}$$

$$= (\mathcal{I} \otimes X^{\mu_0} \otimes \mathcal{I}) \sum_{a\in\{0,1\}} \alpha_a |a\rangle |a \cdot s, r_a\rangle \tag{7}$$

The result of performing a Hadamard measurement on the final register (containing r_a) to obtain a string d is:

$$(Z^{d\cdot(r_0\oplus r_1)} \otimes X^{\mu_0}) \sum_{a\in\{0,1\}} \alpha_a |a\rangle |a \cdot s\rangle \tag{8}$$

To complete the operation, the server must compute the encryptions of $d \cdot (r_0 \oplus r_1)$ and μ_0. This can be done via a classical homomorphic computation as long as the server is given the encryption of the trapdoor of the function pair f_0, f_1.

So far, we have shown that if $\text{Enc}(s)$ is equal to a pair of trapdoor claw-free functions f_0, f_1 (for which each claw hides the bit s), the encrypted CNOT operation is straightforward.

[3] Each application of the encrypted CNOT operation results in $z, x \in \{0,1\}^2$ sampled uniformly at random.

However, in order to apply quantum operations homomorphically to the Pauli one time pad encryption scheme described above, we require that $\text{Enc}(s)$ is an encryption under a classical homomorphic encryption scheme. Therefore, to obtain a homomorphic encryption scheme for quantum circuits, the above idea must be generalized: we must show how to perform the encrypted CNOT operation if $\text{Enc}(s)$ is a ciphertext of a classical homomorphic encryption scheme which may not have the trapdoor claw-free structure described earlier. We do so by showing that if the classical encryption scheme satisfies certain properties, the function pair f_0, f_1 can be constructed given only $\text{Enc}(s)$. We call such classical encryption schemes *quantum capable*. The two main results of this paper are that a quantum capable scheme can be constructed from the learning with errors problem by combining two existing classical encryption schemes and that quantum capable schemes can be used for homomorphic evaluation of quantum circuits. Combined, they provide the following theorem:

Theorem I.1 (Informal)**.** *Under the assumption that the learning with errors problem with superpolynomial noise ratio is computationally intractable for an efficient quantum machine, there exists a quantum leveled fully homomorphic encryption scheme with classical keys.*

A. Outline

We first describe the Pauli one time pad encryption scheme and show how quantum gates can be applied homomorphically, with the goal of reducing quantum homomorphic encryption to the encrypted CNOT operation described in the introduction. We then describe how the encrypted CNOT operation works, in the case that $\text{Enc}(s)$ is equal to a trapdoor claw-free function pair f_0, f_1 which hides the encrypted bit s. Next, we show how a classical encryption of a bit s can be used to build f_0, f_1 if the encryption scheme has certain properties and we use these properties to describe how such quantum capable homomorphic encryption schemes are defined. We conclude by describing how to combine two existing classical homomorphic encryption schemes ([16], [17]) to form a quantum capable scheme, and then showing how a quantum capable scheme can be used to build a quantum leveled fully homomorphic encryption scheme with classical keys. This paper is an overview of our result; formal statements and complete proofs are deferred to the full version ([18]).

II. Reduction to the Encrypted CNOT Operation

An l qubit quantum state $|\psi\rangle$ is Pauli one time padded by choosing $z, x \in \{0,1\}^l$ at random and applying $Z^z X^x$ to $|\psi\rangle$, creating $Z^z X^x |\psi\rangle$. The bit strings z, x are called the *Pauli keys* and are retained by the client. Once the client sends the encrypted state to the server, the shared state held by the client and server is (the last register containing zx is held by the client):

$$\frac{1}{2^{2l}} \sum_{z,x \in \{0,1\}^l} Z^z X^x |\psi\rangle \langle\psi| (Z^z X^x)^\dagger \otimes |zx\rangle \langle zx| \quad (9)$$

The client's decoding process is simple: he simply uses the keys z, x to apply the Pauli operator $(Z^z X^x)^\dagger$ to the state he receives from the server. A nice property is that, in the case that $|\psi\rangle$ is a standard basis state $|m\rangle$, the quantum one time pad is the same as the classical one time pad:

$$Z^z X^x |m\rangle \langle m| X^x Z^z = |m \oplus x\rangle \langle m \oplus x| \quad (10)$$

It follows that the encoding and decoding of a standard basis state can be performed classically.

The key property of the Pauli one time pad used in blind computing is the fact that it can be used hide a quantum state entirely: to the server, who has no knowledge of the Pauli keys, a Pauli one time padded quantum state is equal to the maximally mixed state (the identity $\mathcal{I}$), as stated in the following lemma:

Lemma II.1 (Pauli Mixing). *For a matrix ρ on two spaces A, B*

$$\frac{1}{2^{2l}} \sum_{z,x \in \{0,1\}^l} (Z^z X^x \otimes \mathcal{I}_B) \rho (Z^z X^x \otimes \mathcal{I}_B)^\dagger = \frac{1}{2^l} \mathcal{I}_A \otimes Tr_A(\rho)$$

The information theoretically secure Pauli one time pad encryption scheme can be easily transformed into a computationally secure encryption scheme ([2]). To do so, the client simply encrypts his classical Pauli keys (z, x) (using a classical homomorphic encryption scheme) and includes the encryption $\text{Enc}(z, x)$ as part of the encryption of the state $|\psi\rangle$; the encryption is now a two part encryption, containing classically encrypted keys and the Pauli one time padded quantum state $Z^z X^x |\psi\rangle$. To decode, the client requests both the Pauli key encryptions and the quantum state. He first decrypts the Pauli key encryptions to obtain the Pauli keys, and then applies the inverse of the Pauli keys to the quantum state.

A. Homomorphic Gate Application

In order to apply quantum gates homomorphically to the computationally secure Pauli encryption scheme, the server will need to run two separate computations: a classical homomorphic computation on the Pauli keys and a quantum computation on the one time padded state. In this section, we show how the server is able to perform the following transformations for all gates V in a universal set of gates (our universal set will be the Clifford group along with the Toffoli gate):

$$\text{Enc}(z, x) \to \text{Enc}(z', x') \quad (11)$$

$$Z^z X^x |\psi\rangle \to Z^{z'} X^{x'} V |\psi\rangle \quad (12)$$

1) Homomorphic Application of Pauli and Clifford Gates: To achieve our goal, we simply need to show that the transformations in (11)/(12) can be computed if V is a Clifford gate or a Toffoli gate. To provide intuition, we also include the case in which V is a Pauli gate $Z^a X^b$. In this case, the server only performs the classical part of the parallel computation (in (11)): he homomorphically updates his Pauli keys from $\text{Enc}(z, x)$ to $\text{Enc}(z', x')$, for $(z' = z \oplus a, x' = x \oplus b)$. The

server has now effectively applied $Z^a X^b$ to his one time padded state, since

$$Z^z X^x |\psi\rangle = Z^{z' \oplus a} X^{x' \oplus b} |\psi\rangle = Z^{z'} X^{x'} Z^a X^b |\psi\rangle \quad (13)$$

The equality follows up to a global phase since Pauli operators anti commute.

We continue to the case in which V is equal to a Clifford gate C. Clifford gates are applied by two parallel computations: a homomorphic Pauli key update and a quantum operation on the one time padded state. The server applies the gate C to his one time padded state, resulting in $CZ^z X^x |\psi\rangle$. We now take advantage of the fact that the Clifford group preserves the Pauli group by conjugation: for all z, x, there exist z', x' such that

$$CZ^z X^x |\psi\rangle = Z^{z'} X^{x'} C |\psi\rangle \quad (14)$$

To complete the Clifford application, the server homomorphically updates his Pauli keys from $\text{Enc}(z, x)$ to $\text{Enc}(z', x')$.

2) Homomorphic Application of the Toffoli Gate: The Toffoli gate is more complicated. It cannot be applied by parallel quantum/classical operations by the server, as was done for Clifford gates. This is because it does not preserve Pauli operators by conjugation; applying a Toffoli directly to a 3 qubit one time padded state yields:

$$TZ^z X^x |\psi\rangle = T(Z^z X^x) T^\dagger T |\psi\rangle \quad (15)$$

The correction $T(Z^z X^x) T^\dagger$ is not a Pauli operator, as was the case for Clifford operators; it is instead a product of Pauli and Clifford operators, where the Clifford operator involves Hadamard gates and gates of the form $\text{CNOT}^{b_{zx}}$ (b_{zx} is a bit which depends on the Pauli keys z, x). Since the correction is a Clifford gate and not a Pauli gate, it cannot be removed by a simple homomorphic Pauli key update by the server.

In order to complete the application of the Toffoli gate, the server will need to remove the operators $\text{CNOT}^{b_{zx}}$ up to Pauli operators. Since the server holds the encrypted Pauli keys, we can assume the server can compute an encryption of b_{zx}. Therefore, we have reduced the question of applying a Toffoli gate on top of a one time padded state to the following question: can a BQP server use a ciphertext c encrypting a bit s to apply CNOT^s to a quantum state (up to Pauli operators)? In our setting specifically, s will be a function of the Pauli keys of the one time padded state.

III. Encrypted CNOT Operation

We now present the key idea in this paper: we show how a BQP server can apply CNOT^s if he holds a ciphertext c encrypting a bit s. We call this procedure an *encrypted CNOT operation*. We first show how to perform this operation in an ideal scenario in which the ciphertext c is a trapdoor claw-free function pair hiding the bit s. We then generalize to the case in which c is a ciphertext from a classical homomorphic encryption scheme which satisfies certain properties, which we will describe as they are used.

In our ideal scenario, there exists finite sets $\mathcal{R}, \mathcal{Y}$ and the ciphertext c is equal to a trapdoor claw-free function pair $f_0, f_1 : \{0,1\} \times \mathcal{R} \to \mathcal{Y}$ with one additional property. As a reminder of trapdoor claw-free function pairs, recall that both f_0, f_1 are injective and their images are equal. There also exists a trapdoor which allows for efficient inversion of both functions. Note that we have introduced an extra bit in the domain; this bit of the preimage will be used to hide the bit s. The property we require is as follows: for all $\mu_0, \mu_1 \in \{0,1\}$ and $r_0, r_1 \in \mathcal{R}$ for which $f_0(\mu_0, r_0) = f_1(\mu_1, r_1)$, $\mu_0 \oplus \mu_1 = s$ (s is the value encrypted in the ciphertext c).

Our encrypted CNOT operation boils down to the ability to extract an encrypted bit from a classical encryption and instead store it in superposition in a quantum state. The claw-free function pair f_0, f_1 described above serves as a classical encryption for the bit s which immediately allows this task. At a high level (we discuss the details in the next paragraph), this is because it is possible for the server to compute the following superposition, for a random claw $(\mu_0, r_0), (\mu_1, r_1)$:

$$\frac{1}{\sqrt{2}} \sum_{b \in \{0,1\}} |\mu_b\rangle |r_b\rangle \quad (16)$$

Since $\mu_0 \oplus \mu_1 = s$, the above state can be written as:

$$(X^{\mu_0} \otimes \mathcal{I}) \frac{1}{\sqrt{2}} \sum_{b \in \{0,1\}} |b \cdot s\rangle |r_b\rangle \quad (17)$$

Therefore, the server is able to easily convert a classical encryption of s (which is in the form a trapdoor claw-free function pair) to a quantum superposition which contains s.

It is quite straightforward to use the above process to apply the encrypted CNOT operation: the superposition over the claw in (16) is simply entangled with the first qubit of the quantum state on which the CNOT is to be applied. We now describe this process in detail. Assume the server would like to apply CNOT^s to a 2 qubit state $|\psi\rangle = \sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b\rangle$. The server begins by entangling the first qubit of the state $|\psi\rangle$ with a random claw of f_0, f_1, which proceeds as follows. The server uses the first qubit of $|\psi\rangle$ to choose between the functions f_0, f_1 in order to create the following superposition:

$$\frac{1}{\sqrt{2|\mathcal{R}|}} \sum_{a,b,\mu \in \{0,1\}, r \in \mathcal{R}} \alpha_{ab} |a, b\rangle |\mu, r\rangle |f_a(\mu, r)\rangle \quad (18)$$

Now the server measures the final register to obtain $y \in \mathcal{Y}$. Let $(\mu_0, r_0), (\mu_1, r_1)$ be the two preimages of y ($f_0(\mu_0, r_0) = f_1(\mu_1, r_1) = y$). The remaining state is:

$$\sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b\rangle |\mu_a\rangle |r_a\rangle \quad (19)$$

Recall that to apply CNOT^s, the value $a \cdot s$ must be added to the register containing b. This is where the structure in (17) (which relies on the fact that $\mu_0 \oplus \mu_1 = s$) comes in to play: to add $a \cdot s$, the server XORs μ_a into the second register, which essentially applies the operation CNOT^s:

$$\sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b \oplus \mu_a\rangle |\mu_a\rangle |r_a\rangle \quad (20)$$

$$= \sum_{a,b \in \{0,1\}} \alpha_{ab} (\mathcal{I} \otimes X^{\mu_0}) \text{CNOT}^s_{1,2} |a, b\rangle \otimes |\mu_a, r_a\rangle \quad (21)$$

Finally, the server applies a Hadamard transform on the registers containing μ_a, r_a and measures to obtain d. If we let (μ_a, r_a) denote the concatenation of the two values, the resulting state (up to a global phase) is

$$(Z^{d\cdot((\mu_0,r_0)\oplus(\mu_1,r_1))}\otimes X^{\mu_0})\text{CNOT}^s_{1,2}\sum_{a,b\in\{0,1\}}\alpha_{ab}\,|a,b\rangle \quad (22)$$

In order to complete the encrypted CNOT operation, the server requires an encryption of the trapdoor of the functions f_0, f_1. The server can then homomorphically compute the bits μ_0 and $d\cdot((\mu_0,r_0)\oplus(\mu_1,r_1))$ and use these bits to update his Pauli keys.

A. Trapdoor Claw-free Pair Construction

So far, we have shown how to apply the encrypted CNOT operation in the case that the ciphertext c encrypting the bit s is a trapdoor claw-free function pair f_0, f_1 which hides s. To build a homomorphic encryption scheme, we need to show how the encrypted CNOT operation can be applied if c instead comes from a classical homomorphic encryption scheme, which we call HE. We now show that if HE satisfies certain properties, the function pair f_0, f_1 hiding the bit s can be constructed (by the server) using c.

The function f_0 will be the encryption function of HE. The function f_1 is the function f_0 shifted by the homomorphic XOR of the ciphertext c encrypting the bit s: $f_0 = f_1 \oplus_H c$ ($\oplus_H$ is the homomorphic XOR operation). To ensure that f_0, f_1 are injective, we require that HE has the property of randomness recoverability: there must exist a trapdoor which allows recovery of μ_0, r_0 from a ciphertext $\text{Enc}(\mu_0; r_0)$ ($\text{Enc}(\mu_0; r_0)$ denotes the encryption of a bit μ_0 with randomness r_0). We also require that the homomorphic XOR operation is efficiently invertible using only the public key of HE.

Unfortunately, the images of the functions f_0, f_1 are not equal. We will instead require the weaker (but still sufficient) condition that there exists a distribution D over the domain of the functions such that $f_0(D)$ and $f_1(D)$ are statistically close. We replace (18) with the corresponding weighted superposition:

$$\sum_{a,b,\mu\in\{0,1\},r}\alpha_{ab}\sqrt{D(\mu,r)}\,|a\rangle\,|b\rangle\,|\mu,r\rangle\,|f_a(\mu,r)\rangle \quad (23)$$

To do so, we require that the server can efficiently create the following superposition:

$$\sum_{\mu\in\{0,1\},r}\sqrt{D(\mu,r)}\,|\mu,r\rangle \quad (24)$$

Due to the negligible statistical distance between $f_0(D)$ and $f_1(D)$, when the last register of (23) is measured to obtain y, with high probability there exist μ_0, r_0, μ_1, r_1 such that $y = f_0(\mu_0, r_0) = f_1(\mu_1, r_1)$, which implies that the state collapses to (19) and that

$$\text{Enc}(\mu_0; r_0) = \text{Enc}(\mu_1; r_1) \oplus_H c \quad (25)$$

Since $\oplus_H$ is the homomorphic XOR operation, $\mu_0 \oplus \mu_1 = s$.

There is one remaining issue with the encrypted CNOT operation described above. The requirements above must hold for a classical ciphertext c which occurs at any point during the classical computation on the encrypted Pauli keys. However, in many classical homomorphic encryption schemes, the format of the ciphertext c changes throughout the computation. We know of several schemes for which the above requirements hold for a freshly encrypted ciphertext, but we do not know of any schemes which satisfy the requirements during a later stage of computation. The next section addresses this complication by sufficiently weakening the above requirements while preserving the functionality of the encrypted CNOT operation.

IV. Quantum Capable Classical Homomorphic Encryption Schemes

In this section, we define quantum capable homomorphic encryption schemes, i.e. classical leveled fully homomorphic encryption schemes which can be used to evaluate quantum circuits. To justify why we must weaken the requirements listed in Section III, we begin with a description of the ideal high level structure of a quantum capable homomorphic encryption scheme. In many classical homomorphic encryption schemes, the encryption of a bit b can be thought of as a random element of a subset S_b, perturbed by some noise term ϵ. As the computation progresses, we will require that the structure of the ciphertext remains the same; it must still be a random element of S_b, but the noise term may grow throughout the computation. We will also require that the homomorphic XOR operation is natural, in the sense that the noise of the output ciphertext is simply the addition of the two noise terms of the input ciphertexts. If these two conditions hold (invariance of the ciphertext form and the existence of a natural XOR operation), deriving a distribution $f_0(D)$ over ciphertexts which remains roughly the same after shifting by the homomorphic XOR of the ciphertext c (as needed in Section III) is straightforward. We simply choose D to sample the noise term from a discrete Gaussian distribution with width sufficiently larger than the magnitude of the noise term of the ciphertext c.

Unfortunately, we do not know of a classical homomorphic encryption scheme which satisfies both the conditions (ciphertext form and natural XOR operation) at once. To account for this difficulty, we define a quantum capable homomorphic encryption schemes as follows. We call a classical homomorphic encryption scheme HE quantum capable if there exists an alternative encryption scheme AltHE which satisfies the following conditions. First, given a ciphertext c under HE, it must be possible for the server to convert c to a ciphertext $\hat{c}$ under AltHE. The conversion process must maintain the decrypted value of the ciphertext. Second, AltHE must have a natural homomorphic XOR operation (which is also efficiently invertible). Third, there must exist a distribution $f_0(D)$ over encryptions under AltHE which must remain almost the same after shifting by the homomorphic XOR of $\hat{c}$ and must allow efficient construction of the superposition in (24). In addition,

it must be possible to both decrypt and recover randomness from ciphertexts under AltHE given the appropriate secret key and trapdoor information. This definition is formalized in the full version of this paper ([18]).

Finally, we describe how to connect this weaker definition to the encrypted CNOT operation given in Section III. We begin with a quantum capable homomorphic encryption scheme HE, which is used to encrypt the Pauli keys. HE satisfies the ciphertext form requirement but may not have a natural XOR operation. Each time the server needs to apply an encrypted CNOT operation (controlled by a ciphertext c encrypting a bit s under HE), he will convert c to a ciphertext $\hat{c}$ under AltHE, which does have a natural XOR operation. Using AltHE (rather than HE) the server performs the operations described in Section III. Upon obtaining his measurement results (denoted as y and d), the server will encrypt both $\hat{c}$ and y, d under HE. The server will then use the secret key and trapdoor information of AltHE, which are provided to him as encryptions under HE, to homomorphically recover the randomness and decrypted values from both y and $\hat{c}$. This encrypted information can be used to homomorphically compute the Pauli key updates. The entire classical homomorphic computation is done under HE.

V. Example of a Quantum Capable Classical Encryption Scheme

In the full version of this paper ([18]), we show that an existing classical fully homomorphic encryption scheme is quantum capable. We use the structure of the scheme from [17], which is a leveled fully homomorphic encryption scheme built by extending the vector ciphertexts of [19] to matrices. The resulting encryption scheme does satisfy the ciphertext form requirement (as described in Section IV), but since the underlying encryption scheme ([19]) does not have the randomness recoverability property, neither does [17]. We therefore alter the scheme from [17] to use the dual encryption scheme of [19], which was introduced in [16] and allows randomness recovery, as the underlying encryption scheme. We call the resulting scheme DualHE and the underlying scheme of [16] Dual.

We use the scheme DualHE as an instantiation of the scheme we called HE in Section IV. Although the underlying scheme Dual does have a natural XOR operation, the extension to matrices compromises the XOR operation; once the ciphertexts are matrices, addition is performed over a larger field. Luckily, it is easy to convert a ciphertext of DualHE to a ciphertext of Dual. We therefore use Dual as AltHE.

In the full version of this paper, we first describe the scheme Dual from [16] and we then show how to extend it to DualHE using [17]. Next, we show that DualHE satisfies the ciphertext form requirement and that a ciphertext under DualHE can be converted to a ciphertext under Dual. We then use these properties to show that DualHE is a classical leveled fully homomorphic encryption scheme. Finally, we show that DualHE is quantum capable with only a small modification of parameters (the underlying assumption for both the classical FHE and the quantum capable instantiation is the hardness of learning with errors with a superpolynomial noise ratio).

VI. Extension to Quantum Leveled Fully Homomorphic Encryption

We have so far provided a quantum fully homomorphic encryption scheme with classical keys under the assumption of circular security: the server must be provided the encrypted secret key and trapdoor information in order to update his encrypted Pauli keys after each encrypted CNOT operation. Our notion of circular security here will be slightly stronger than the standard notion, due to the encryption of the trapdoor (instead of just the secret key). As an alternative to assuming circular security, we can build a quantum leveled fully homomorphic encryption scheme by employing a technique which is commonly used in classical homomorphic encryption schemes (Section 4.1 in [1]): we will encrypt the secret key and trapdoor information under a fresh public key. In other words, the i^{th} level secret key sk_i and its corresponding trapdoor information are encrypted under a fresh public key pk_{i+1} and given to the server as part of the evaluation key. The computation of the Pauli key updates corresponding to the encrypted CNOT operations of level i is performed under pk_{i+1} (i.e. the corresponding $\hat{c}, y$ and d from each encrypted CNOT operation in level i will be encrypted, by the server, under pk_{i+1} - see the last paragraph of Section IV).

Note that with the introduction of the leveled scheme, we can see the classical portion of the quantum homomorphic computation as follows. Each level of the quantum computation can be thought of as a series of Clifford gates followed by one layer of non intersecting Toffoli gates, finishing with a layer of non intersecting encrypted CNOT operations. It follows that the classical homomorphic computation of level i consists of first decrypting and recovering randomness from the cipertexts corresponding to the encrypted CNOT operations from level $i-1$, then performing the Pauli key updates corresponding to the encrypted CNOT operations from level $i-1$ and finally performing the Pauli key updates corresponding to the Clifford and Toffoli gates of level i. The ciphertexts which result from this computation are then used as the control bits for the layer of encrypted CNOT operations of level i.

Intuitively, this leveled approach is secure since each secret key is protected by the semantic security of the encryption scheme under an independent public key. To prove security, we start with the final level of encryption. If there are L levels of the circuit, then there will be no trapdoor or secret key information provided corresponding to pk_{L+1}. It follows that all encryptions under pk_{L+1} can be replaced by encryptions of 0; now there is no encrypted information provided corresponding to sk_L. Then all encryptions under pk_L can be replaced by encryptions of 0, and we can continue in this manner until we reach pk_1, which will imply security of encryptions under the initial public key pk_1. The scheme and proof of security are presented in the full version ([18]), proving that quantum

capable classical encryption schemes can be used to build a quantum leveled fully homomorphic encryption scheme.

ACKNOWLEDGMENT

Thanks to Dorit Aharonov, Zvika Brakerski, Sanjam Garg, Stacey Jeffery, Zeph Landau, Umesh Vazirani and Thomas Vidick for many useful discussions. The author is supported by Templeton Foundation Grant 52536, ARO Grant W911NF-12-1-0541, NSF Grant CCF-1410022 and MURI Grant FA9550-18-1-0161.

REFERENCES

[1] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009, crypto.stanford.edu/craig.

[2] A. Broadbent and S. Jeffery, "Quantum homomorphic encryption for circuits of low t-gate complexity," Cryptology ePrint Archive, Report 2015/551, 2015, http://eprint.iacr.org/2015/551.

[3] Y. Ouyang, S.-H. Tan, and J. F. Fitzsimons, "Quantum homomorphic encryption from quantum codes," 2015.

[4] S.-H. Tan, J. A. Kettlewell, Y. Ouyang, L. Chen, and J. F. Fitzsimons, "A quantum approach to homomorphic encryption. scientific reports," Scientific Reports, 6., 2016.

[5] C.-Y. Lai and K.-M. Chung, "On statistically-secure quantum homomorphic encryption," 2017.

[6] M. Newman and Y. Shi, "Limitations on transversal computation through quantum homomorphic encryption," 2017.

[7] Y. Dulek, C. Schaffner, and F. Speelman, *Quantum Homomorphic Encryption for Polynomial-Sized Circuits*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 3–32.

[8] A. M. Childs, "Secure assisted quantum computation," *Quantum Info. Comput.*, vol. 5, no. 6, pp. 456–466, Sep. 2005.

[9] A. Broadbent, J. F. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," *Arxiv preprint arXiv:0807.4154*, 2008.

[10] D. Aharonov, M. Ben-Or, and E. Eban, "Interactive Proofs For Quantum Computations," *Arxiv preprint arXiv:0810.5375*, 2008.

[11] A. Mantri, T. F. Demarie, N. C. Menicucci, and J. F. Fitzsimons, "Flow ambiguity: A path towards classically driven blind quantum computation," *Phys. Rev. X*, vol. 7, p. 031004, 07 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.7.031004

[12] A. Cojocaru, L. Colisson, E. Kashefi, and P. Wallden, "Delegated pseudo-secret random qubit generator," 2018.

[13] Z. Brakerski, P. Christiano, U. Mahadev, U. Vazirani, and T. Vidick, "Certifiable randomness from a single quantum device," *Arxiv preprint 1804.00640*, 2018.

[14] U. Mahadev, "Classical verification of quantum computations," 2018.

[15] A. Ambainis, M. Mosca, A. Tapp, and R. de Wolf, "Private quantum channels," in *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, ser. FOCS '00. Washington, DC, USA: IEEE Computer Society, 2000, pp. 547–. [Online]. Available: http://dl.acm.org/citation.cfm?id=795666.796592

[16] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," Cryptology ePrint Archive, Report 2007/432, 2007, http://eprint.iacr.org/2007/432.

[17] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," Cryptology ePrint Archive, Report 2013/340, 2013, http://eprint.iacr.org/2013/340.

[18] U. Mahadev, "Classical Homomorphic Encryption for Quantum Circuits," *Arxiv preprint arXiv:1708.02130*, 2017.

[19] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, ser. STOC '05. New York, NY, USA: ACM, 2005, pp. 84–93.