

# Classical Homomorphic Encryption for Quantum Circuits

Urmila Mahadev\*

April 10, 2018

## 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.

## 1 Introduction

Is it possible for a weak client to use a more powerful device (a server) to run a computation, while maintaining privacy of the data on which the computation is run? This is a major question in classical cryptography, and was answered affirmatively in [Gen09] by the construction of a fully homomorphic encryption scheme. In such a scheme, a client can provide a server with encrypted data, on top of which the server is able to run a classical computation.

It is natural to ask the analogous question in the case of a quantum server, especially since early quantum computers are expected to be cloud-based. This question can be formalized in two different ways, depending upon the power of the client. Ideally the client would be classical, and access the quantum server over the internet. A different formulation would allow the client to carry out restricted quantum operations and send these qubits to the server. The latter formulation of the question has been well studied, and there are results showing how to carry out blind computation of a general BQP circuit, both allowing interaction between the client and the server ([BFK08],[ABOE08],[FK17],[ABOEM17]) and without interaction ([DSS16]). In these results, the quantum operations of the client are restricted in some sense; either the client does not need to run a general BQP circuit or the client may only require a constant sized quantum register. Unfortunately, the total number of quantum operations carried out by the client scales at least as fast as the *size* (rather than the depth) of the quantum circuit being delegated, and it is an open question whether the quantum computation performed by the client can be made even marginally less dependent on the size of the delegated circuit.

In this paper we study the first formulation, which asks whether it is possible for a purely classical client to use a BQP server to run a private quantum computation. In other words, does there exist a classical method of hiding data while still allowing an arbitrary quantum circuit to be applied on the data? There has been significantly less progress on this variant of the question of homomorphic encryption. We give the first protocol which achieves the goal of outsourcing quantum computation with a classical client. Note that this means that the input and output of the

---

\*Department of Computer Science, UC Berkeley, USA. Supported by Templeton Foundation Grant 52536, ARO Grant W911NF-12-1-0541, NSF Grant CCF-1410022 and MURI Grant FA9550-18-1-0161. Email: mahadev@cs.berkeley.edu.

delegated circuit are necessary classical, which is not a restriction for standard quantum computations. However, our scheme can be immediately extended to allow for quantum data as input: now the client must be quantum, but the encryption keys remain classical.

To build our protocol, we use the fact that blindly computing a quantum circuit boils down to the ability of a quantum server to perform a CNOT gate (a reversible XOR gate) controlled by classically encrypted data: we need a method in which a quantum server holding a classically encrypted bit  $s$  can apply  $\text{CNOT}^s$  to a quantum state (i.e. apply the CNOT gate if and only if  $s = 1$ ). This method must not leak information about the bit  $s$ . This observation is not new to this paper, and is in fact quite well known ([BJ14], [DSS16]). However, all previous results proposed quantum functions to enable the server to perform such an operation. Here we propose a classical function, which we call an encrypted CNOT operation.

Our encrypted CNOT operation relies on a two to one function for which the two preimages of a given point hide the bit  $s$ . While the idea of using two to one functions to constrain quantum provers was introduced in [BCM<sup>+</sup>18], here we use two to one functions for functionality instead. We show that the function we require can be built out of a classical homomorphic encryption scheme which satisfies several straightforward requirements; we call such schemes *quantum capable* encryption schemes. Next, by combining two existing homomorphic encryption schemes based on learning with errors ([GSW13], [GPV07]), we build a quantum capable homomorphic encryption scheme. Our main result is as follows (stated formally in Theorem 5.2 and Theorem 6.2):

**Theorem 1.1 (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.*

In our protocol, the total amount of computation done by the client is proportional to the depth of the delegated circuit rather than its size. Moreover, as in classical homomorphic encryption schemes, this overhead can be made independent of the circuit size with stronger assumptions regarding the circular security of the classical encryption scheme.

## 1.1 Related Work

There have been several results proposing quantum homomorphic encryption schemes, for both general ([DSS16]) and restricted ([BJ14], [OTF15], [TKO<sup>+</sup>16], [LC17], [NS17]) BQP circuits. These results all require a restricted quantum client and satisfy the definition of quantum homomorphic encryption given in [BJ14], which is strictly weaker than the classical definition of homomorphic encryption. The main difference lies in the reusability of keys: quantum keys cannot be reused. Each time the client wishes to delegate a quantum circuit, he must generate fresh keys. In [DSS16] (the only known scheme which achieves non interactive blind delegation of general quantum circuits) the size of the key generation procedure is proportional to the size of the delegated circuit. Although the quantum circuit used to generate the keys is simpler than the delegated circuit, the client still must run this polynomial size circuit each time he delegates a quantum computation.

In contrast, leveled classical homomorphic encryption schemes have classical keys and the size of the key generation process corresponds to the depth of the delegated circuit. The result in this paper satisfies the classical definition of homomorphic encryption: the client only needs to run a single classical computation of size corresponding to the depth of the delegated circuit in order to delegate as many quantum computations as he wishes. Furthermore, our scheme has the same functionality as [DSS16], as it can be immediately extended to use quantum states as input while retaining the advantage of classical keys.

In terms of more recent work, there have been two papers ([MDMF17], [CCKW18]) 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 interaction.

## 2 Overview

We will begin with a simple encryption scheme, the Pauli one time pad, for quantum states and show how to extend it into a quantum leveled fully homomorphic encryption. To do this, we will show how a universal set of quantum gates (Pauli, Clifford and Toffoli gates) can be applied on a state encrypted with the Pauli pad. Most of the focus will be on the Toffoli gate. To begin, we present the Pauli one time pad encryption, which is not new to this paper (it was also used in [BJ15], [DSS16]).

### 2.1 Pauli One Time Pad

We can encrypt an  $l$  qubit quantum state  $|\psi\rangle$  by choosing random Pauli operators in  $\mathbb{P}_l$  and applying them on a state. The client first chooses a *Pauli key*  $P \in \mathbb{P}_l$  at random. The client then sends  $P|\psi\rangle$  to the server. Note that if  $|\psi\rangle$  is a standard basis state,  $P|\psi\rangle$  can be represented as a classical string, since the phase portion of  $P$  has no effect on  $|\psi\rangle$ . The shared state held by the client and server after the server receives the state is:

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

where the last register containing  $zx$  is held by the client. The client's decoding process is simple: he simply uses the keys  $z, x$  to apply a Pauli operator  $(Z^z X^x)^\dagger$  to the state he receives from the server. To see why the Pauli one time pad hides the state from the server, we can just consider the server's part of the above state (the last register containing  $z, x$  above is traced out):

$$\frac{1}{|\mathbb{P}_l|} \sum_{P \in \mathbb{P}_l} P |\psi\rangle \langle\psi| P^\dagger = \frac{1}{2^l} \mathcal{I} \quad (2)$$

The equality is due to the Pauli mixing lemma, which we state below (and prove in Section B.1):

**Lemma 2.1 (Pauli Mixing)** *For a matrix  $\rho$  on two spaces  $A, B$*

$$\frac{1}{|\mathbb{P}_l|} \sum_{P \in \mathbb{P}_l} (P \otimes \mathcal{I}_B) \rho (P \otimes \mathcal{I}_B)^\dagger = \frac{1}{2^l} \mathcal{I}_A \otimes \text{Tr}_A(\rho)$$

This information theoretically secure encryption scheme can be easily transformed into a computationally secure encryption scheme by giving the server the Pauli key encrypted using a classical homomorphic encryption scheme. 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. We now describe how quantum gates can be applied by the server on top of this computationally secure Pauli encryption scheme.

### 2.2 Clifford and Pauli Gate Application

Pauli gates can be applied on a one time padded state without applying any operators to the state itself; only the Pauli keys need to be homomorphically updated by the server. For example, to apply a Pauli operation  $Z^a X^b$  to a state  $|\psi\rangle$  which is one time padded with the Pauli operator  $Z^z X^x$ , the server simply updates the encrypted keys

from  $(z, x)$  to  $(z \oplus a, x \oplus b)$ . For more details, see Section A.1.

Applying a Clifford gate involves both a gate application and a Pauli key update. To apply a Clifford gate  $C$  on a state  $|\psi\rangle$  encoded by  $Z^z X^x$ , the server first applies  $C$ . This maps the state to:

$$C Z^z X^x |\psi\rangle = C(Z^z X^x) C^\dagger C |\psi\rangle = Z^{\hat{z}} X^{\hat{x}} C |\psi\rangle \quad (3)$$

The equality follows because Clifford operators preserve Pauli operators by conjugation. The server completes the Clifford application by homomorphically updating the encrypted Pauli keys from  $(z, x)$  to  $(\hat{z}, \hat{x})$ . For more details, see Section A.2.

### 2.3 Toffoli Gate Application

Assume we would like to apply a Toffoli gate to  $Z^z X^x |\psi\rangle$  (a 3 qubit state). Applying a Toffoli directly yields:

$$T Z^z X^x |\psi\rangle = T(Z^z X^x) T^\dagger T |\psi\rangle \quad (4)$$

Unfortunately,  $T(Z^z X^x) T^\dagger$  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 one of the Pauli keys). For more details, see Section A.3.

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 holding a ciphertext  $c$  encrypting a bit  $s$  use the ciphertext to apply  $\text{CNOT}^s$  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.

### 2.4 Encrypted CNOT Operation

We now present the key idea in this paper: we show how a BQP server can apply  $\text{CNOT}^s$  if he holds a ciphertext  $c$  encrypting  $s$ . We call this procedure an *encrypted CNOT operation*. So far, we have only required a generic classical homomorphic encryption scheme in order to encrypt the Pauli keys. However, the encrypted CNOT operation will require the underlying homomorphic encryption scheme to have certain properties, which we will describe as they are used.

We begin by imagining an ideal scenario in which there exist finite sets  $\mathcal{R}, \mathcal{Y}$  and a pair of functions  $f_0, f_1 : \{0, 1\} \times \mathcal{R} \rightarrow \mathcal{Y}$  with the following properties. First, both  $f_0, f_1$  are injective and their images are equal. 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  $c$ ). Moreover, there exists a trapdoor which allows inversion of both functions.

If the server held such a pair of functions, the encrypted CNOT operation is straightforward and proceeds as follows. Assume the server would like to apply  $\text{CNOT}^s$  to a 2 qubit state  $|\psi\rangle = \sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b\rangle$ . 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:

$$\frac{1}{\sqrt{2|\mathcal{R}|}} \sum_{a,b,\mu \in \{0,1\}, r \in \mathcal{R}} \alpha_{ab} |a, b\rangle |\mu, r\rangle |f_a(\mu, r)\rangle \quad (5)$$

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:

$$\sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b\rangle |\mu_a\rangle |r_a\rangle \quad (6)$$

At this point, the server XORs  $\mu_a$  into the second register. Since  $\mu_0 \oplus \mu_1 = s$ , this is equivalent to applying the operation  $\text{CNOT}^s$ :

$$\sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b \oplus \mu_a\rangle |\mu_a\rangle |r_a\rangle = \sum_{a,b \in \{0,1\}} \alpha_{ab} (\mathcal{I} \otimes X^{\mu_0}) \text{CNOT}_{1,2}^s |a, b\rangle \otimes |\mu_a, r_a\rangle \quad (7)$$

Finally, the server removes the interference by applying a Hadamard transform on the registers containing  $\mu_a, r_a$  and measuring to obtain  $d$ . If we let  $(\mu_a, r_a)$  denote the concatenation of the two values, the resulting state (up to a global phase) is

$$(Z^{d \cdot ((\mu_0, r_0) \oplus (\mu_1, r_1))} \otimes X^{\mu_0}) \text{CNOT}_{1,2}^s \sum_{a,b \in \{0,1\}} \alpha_{ab} |a, b\rangle \quad (8)$$

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

We now construct the two to one function pair used above from a classical homomorphic encryption scheme (which we will call HE) with certain properties. 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  $c$ :  $f_0 = f_1 \oplus_H c$  (where  $\oplus_H$  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  $\mu_0, r_0$  from a ciphertext  $\text{Enc}_{pk}(\mu_0; r_0)$  ( $\text{Enc}_{pk}(\mu_0; r_0)$  denotes the encryption of a bit  $\mu_0$  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 (5) with the corresponding weighted superposition:

$$\sum_{a,b,\mu \in \{0,1\}, r} \alpha_{ab} \sqrt{D(\mu, r)} |a\rangle |b\rangle |\mu, r\rangle |f_a(\mu, r)\rangle \quad (9)$$

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

$$\sum_{\mu \in \{0,1\}, r} \sqrt{D(\mu, r)} |\mu, r\rangle \quad (10)$$

Due to the negligible statistical distance between  $f_0(D)$  and  $f_1(D)$ , when the last register of (9) is measured to obtain  $y$ , with high probability there exist  $\mu_0, r_0, \mu_1, r_1$  such that  $y = f_0(\mu_0, r_0) = f_1(\mu_1, r_1)$ , which implies that the state collapses to (6) and that

$$\text{Enc}_{pk}(\mu_0; r_0) = \text{Enc}_{pk}(\mu_1; r_1) \oplus_H c \quad (11)$$

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.

## 2.5 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 2.4, 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  $\epsilon$ . 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 2.4) 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 (10). 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 Section 4.

Finally, we describe how to connect this weaker definition to the encrypted CNOT operation given in Section 2.4. 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 2.4. 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.



## 2.6 Example of a Quantum Capable Classical Encryption Scheme

In Section 5 (Theorem 5.2), we show that an existing classical fully homomorphic encryption scheme is quantum capable. We use the structure of the scheme from [GSW13], which is a leveled fully homomorphic encryption scheme built by extending the vector ciphertexts of [Reg05] to matrices. The resulting encryption scheme does satisfy the ciphertext form requirement (as described in Section 2.5), but since the underlying encryption scheme ([Reg05]) does not have the randomness recoverability property, neither does [GSW13]. We therefore alter the scheme from [GSW13] to use the dual encryption scheme of [Reg05], which was introduced in [GPV07] and allows randomness recovery, as the underlying encryption scheme. We call the resulting scheme DualHE and the underlying scheme of [GPV07] Dual.

We use the scheme DualHE as an instantiation of the scheme we called HE in Section 2.5. 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 Section 5, we first describe the scheme Dual from [GPV07] and we then show how to extend it to DualHE using [GSW13]. Next, we show that DualHE satisfies the ciphertext form requirement and that a ciphertext under DualHE can be converted to a ciphertext under Dual. In Theorem 5.1, we use these properties to show that DualHE is a classical leveled fully homomorphic encryption scheme. Finally, we show in Theorem 5.2 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).

## 2.7 Extension to Leveled 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 [Gen09]): 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 2.5).

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 ciphertexts 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 Section 6, proving that quantum capable classical encryption schemes can be used to build a quantum leveled fully homomorphic encryption scheme (see Theorem 6.2 for a formal statement).

**Paper Organization** We begin with a background section (Section 3), which covers the necessary information about quantum gates and the classical cryptographic primitives we will require. In Section 4, we define quantum capable encryption schemes by listing the requirements that a classical homomorphic encryption scheme must satisfy in order to be used to evaluate quantum circuits, as described in Section 2.5. We use this definition to formally prove the correctness (in Claim 4.3) of the encrypted CNOT operation given in Section 2.4. Section 5 covers Section 2.6: we provide an example of a classical homomorphic encryption scheme which is quantum capable. In Section 6, we formally show how to extend a quantum capable classical leveled fully homomorphic encryption scheme to a quantum leveled fully homomorphic encryption scheme (as described in Section 2.7).

### 3 Preliminaries

Throughout this paper, we borrow notation and definitions from [ABOE08], [ABOEM17] and [BCM<sup>+</sup>18]. Parts of the following sections are also taken from these sources.

#### 3.1 Notation

For all  $q \in \mathbb{N}$  we let  $\mathbb{Z}_q$  denote the ring of integers modulo  $q$ . We represent elements in  $\mathbb{Z}_q$  using numbers in the range  $(-\frac{q}{2}, \frac{q}{2}] \cap \mathbb{Z}$ . We denote by  $[x]_q$  the unique integer  $y$  s.t.  $y = x \pmod{q}$  and  $y \in (-\frac{q}{2}, \frac{q}{2}]$ . For  $x \in \mathbb{Z}_q$  we define  $|x| = |[x]_q|$ . For a vector  $\mathbf{u} \in \mathbb{Z}_q^n$ , we write  $\|\mathbf{u}\|_\infty \leq \beta$  if each entry  $u_i$  in  $\mathbf{u}$  satisfies  $|u_i| \leq \beta$ . Similarly, for a matrix  $U \in \mathbb{Z}_q^{n \times m}$ , we write  $\|U\|_\infty \leq \beta$  if each entry  $u_{i,j}$  in  $U$  satisfies  $|u_{i,j}| \leq \beta$ .

We use the terminology of polynomially bounded, super-polynomial, and negligible functions. A function  $n : \mathbb{N} \rightarrow \mathbb{R}_+$  is *polynomially bounded* if there exists a polynomial  $p$  such that  $n(\lambda) \leq p(\lambda)$  for all  $\lambda \in \mathbb{N}$ . A function  $n : \mathbb{N} \rightarrow \mathbb{R}_+$  is *negligible* (resp. *super-polynomial*) if for every polynomial  $p$ ,  $p(\lambda)n(\lambda) \rightarrow_{\lambda \rightarrow \infty} 0$  (resp.  $n(\lambda)/p(\lambda) \rightarrow_{\lambda \rightarrow \infty} \infty$ ).

We generally use the letter  $D$  to denote a distribution over a finite domain  $X$ . We write  $U$  for the uniform distribution. We write  $x \leftarrow D$  to indicate that  $x$  is sampled from distribution  $D$ , and  $x \leftarrow_U X$  to indicate that  $x$  is sampled uniformly from the set  $X$ . For two distributions  $D_1$  and  $D_2$  over the same finite domain  $X$ , the Hellinger distance between  $D_1$  and  $D_2$  is

$$H^2(D_1, D_2) = 1 - \sum_{x \in X} \sqrt{D_1(x)D_2(x)}. \quad (12)$$

#### 3.2 Homomorphic Encryption

The following definitions are modified versions of definitions from [BV14] and [BV13]. A homomorphic (public-key) encryption scheme  $\text{HE} = (\text{HE.Keygen}, \text{HE.Enc}, \text{HE.Dec}, \text{HE.Eval})$  is a quadruple of PPT algorithms which operate as follow:



- **Key Generation.** The algorithm  $(pk, evk, sk) \leftarrow \text{HE.Keygen}(1^\lambda)$  takes a unary representation of the security parameter and outputs a public key encryption key  $pk$ , a public evaluation key  $evk$  and a secret decryption key  $sk$ .
- **Encryption.** The algorithm  $c \leftarrow \text{HE.Enc}_{pk}(\mu)$  takes the public key  $pk$  and a single bit message  $\mu \in \{0, 1\}$  and outputs a ciphertext  $c$ . The notation  $\text{HE.Enc}_{pk}(\mu; r)$  will be used to represent the encryption of a bit  $\mu$  using randomness  $r$ .
- **Decryption.** The algorithm  $\mu^* \leftarrow \text{HE.Dec}_{sk}(c)$  takes the secret key  $sk$  and a ciphertext  $c$  and outputs a message  $\mu^* \in \{0, 1\}$ .
- **Homomorphic Evaluation** The algorithm  $c_f \leftarrow \text{HE.Eval}_{evk}(f, c_1, \dots, c_l)$  takes the evaluation key  $evk$ , a function  $f : \{0, 1\}^l \rightarrow \{0, 1\}$  and a set of  $l$  ciphertexts  $c_1, \dots, c_l$ , and outputs a ciphertext  $c_f$ . It must be the case that:

$$\text{HE.Dec}_{sk}(c_f) = f(\text{HE.Dec}_{sk}(c_1), \dots, \text{HE.Dec}_{sk}(c_l)) \quad (13)$$

with all but negligible probability in  $\lambda$ .

A homomorphic encryption scheme is said to be secure if it meets the following notion of semantic security:

**Definition 3.1 (CPA Security)** A scheme HE is IND-CPA secure if, for any polynomial time adversary  $\mathcal{A}$ , there exists a negligible function  $\mu(\cdot)$  such that

$$\text{Adv}_{\text{CPA}}[\mathcal{A}] \stackrel{\text{def}}{=} |\Pr[\mathcal{A}(pk, evk, \text{HE.Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, evk, \text{HE.Enc}_{pk}(1)) = 1]| = \mu(\lambda) \quad (14)$$

where  $(pk, evk, sk) \leftarrow \text{HE.Keygen}(1^\lambda)$ .

We now define two desirable properties of homomorphic encryption schemes:

**Definition 3.2 (Compactness and Full Homomorphism)** A homomorphic encryption scheme HE is compact if there exists a polynomial  $s$  in  $\lambda$  such that the output length of  $\text{HE.Eval}$  is at most  $s$  bits long (regardless of  $f$  or the number of inputs). A compact scheme is (pure) fully homomorphic if it can evaluate any efficiently computable boolean function. A compact scheme is leveled fully homomorphic if it takes  $1^L$  as additional input in key generation, and can only evaluate depth  $L$  Boolean circuits where  $L$  is polynomial in  $\lambda$ . A scheme is quantum pure or leveled fully homomorphic if we refer to quantum circuits rather than boolean circuits.

### 3.3 Learning with Errors and Discrete Gaussians

This background section on the learning with errors problem is taken directly from [BCM<sup>+</sup>18]. For a positive real  $B$  and positive integers  $q$ , the truncated discrete Gaussian distribution over  $\mathbb{Z}_q$  with parameter  $B$  is supported on  $\{x \in \mathbb{Z}_q : \|x\| \leq B\}$  and has density

$$D_{\mathbb{Z}_q, B}(x) = \frac{e^{-\frac{\pi\|x\|^2}{B^2}}}{\sum_{x \in \mathbb{Z}_q, \|x\| \leq B} e^{-\frac{\pi\|x\|^2}{B^2}}} . \quad (15)$$

We note that for any  $B > 0$ , the truncated and non-truncated distributions have statistical distance that is exponentially small in  $B$  [Ban93, Lemma 1.5]. For a positive integer  $m$ , the truncated discrete Gaussian distribution over  $\mathbb{Z}_q^m$  with parameter  $B$  is supported on  $\{x \in \mathbb{Z}_q^m : \|x\| \leq B\sqrt{m}\}$  and has density

$$\forall x = (x_1, \dots, x_m) \in \mathbb{Z}_q^m, \quad D_{\mathbb{Z}_q^m, B}(x) = D_{\mathbb{Z}_q, B}(x_1) \cdots D_{\mathbb{Z}_q, B}(x_m) . \quad (16)$$

**Lemma 3.3** *Let  $B$  be a positive real number and  $q, m$  be positive integers. Let  $\mathbf{e} \in \mathbb{Z}_q^m$ . The Hellinger distance between the distribution  $D = D_{\mathbb{Z}_q^m, B}$  and the shifted distribution  $D + \mathbf{e}$  satisfies*

$$H^2(D, D + \mathbf{e}) \leq 1 - e^{\frac{-2\pi\sqrt{m}\|\mathbf{e}\|}{B}}, \quad (17)$$

and the statistical distance between the two distributions satisfies

$$\|D - (D + \mathbf{e})\|_{TV}^2 \leq 2\left(1 - e^{\frac{-2\pi\sqrt{m}\|\mathbf{e}\|}{B}}\right). \quad (18)$$

**Definition 3.4** *For a security parameter  $\lambda$ , let  $n, m, q \in \mathbb{N}$  be integer functions of  $\lambda$ . Let  $\chi = \chi(\lambda)$  be a distribution over  $\mathbb{Z}$ . The  $\text{LWE}_{n,m,q,\chi}$  problem is to distinguish between the distributions  $(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{q})$  and  $(\mathbf{A}, \mathbf{u})$ , where  $\mathbf{A}$  is uniformly random in  $\mathbb{Z}_q^{n \times m}$ ,  $\mathbf{s}$  is a uniformly random row vector in  $\mathbb{Z}_q^n$ ,  $\mathbf{e}$  is a row vector drawn at random from the distribution  $\chi^m$ , and  $\mathbf{u}$  is a uniformly random vector in  $\mathbb{Z}_q^m$ . Often we consider the hardness of solving LWE for any function  $m$  such that  $m$  is at most a polynomial in  $n \log q$ . This problem is denoted  $\text{LWE}_{n,q,\chi}$ . When we write that we make the  $\text{LWE}_{n,q,\chi}$  assumption, our assumption is that no quantum polynomial-time procedure can solve the  $\text{LWE}_{n,q,\chi}$  problem with more than a negligible advantage in  $\lambda$ .*

As shown in [Reg05, PRS17], for any  $\alpha > 0$  such that  $\sigma = \alpha q \geq 2\sqrt{n}$  the  $\text{LWE}_{n,q,D_{\mathbb{Z}_q,\sigma}}$  problem, where  $D_{\mathbb{Z}_q,\sigma}$  is the discrete Gaussian distribution, is at least as hard as approximating the shortest independent vector problem (SIVP) to within a factor of  $\gamma = \tilde{O}(n/\alpha)$  in worst case dimension  $n$  lattices. This is proven using a quantum reduction. Classical reductions (to a slightly different problem) exist as well [Pei09, BLP<sup>+</sup>13] but with somewhat worse parameters. The best known (classical or quantum) algorithm for these problems run in time  $2^{\tilde{O}(n/\log \gamma)}$ . For our construction we assume hardness of the problem against a quantum polynomial-time adversary in the case that the noise ratio  $\gamma$  is a super polynomial function in  $n$ . This is a commonly used assumption in cryptography (i.e. it is used in homomorphic encryption schemes such as [GSW13]).

We use the fact that it is possible to generate LWE samples  $(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e})$  such that there is a trapdoor allowing recovery of  $\mathbf{s}$  from the samples.

**Theorem 3.5 (Theorem 5.1 in [MP11])** *Let  $n, m \geq 1$  and  $q \geq 2$  be such that  $m = \Omega(n \log q)$ . There is an efficient randomized algorithm  $\text{GENTRAP}(1^n, 1^m, q)$  that returns a matrix  $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$  and a trapdoor  $t_{\mathbf{A}}$  such that the distribution of  $\mathbf{A}$  is negligibly (in  $n$ ) close to the uniform distribution. Moreover, there is an efficient algorithm  $\text{INVERT}$  that, on input  $\mathbf{A}, t_{\mathbf{A}}$  and  $\mathbf{A}\mathbf{s} + \mathbf{e}$  where  $\|\mathbf{e}\| \leq q/(C_T\sqrt{n \log q})$  and  $C_T$  is a universal constant, returns  $\mathbf{s}$  and  $\mathbf{e}$  with overwhelming probability over  $(\mathbf{A}, t_{\mathbf{A}}) \leftarrow \text{GENTRAP}$ .*

### 3.4 Quantum Computing Preliminaries

We will use the  $X, Y$  and  $Z$  Pauli operators:  $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ ,  $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$  and  $Y = iXZ$ . The  $l$ -qubits Pauli group consists of all elements of the form  $P = P_1 \otimes P_2 \otimes \dots \otimes P_l$  where  $P_i \in \{I, X, Y, Z\}$ , together with the multiplicative factors  $-1$  and  $\pm i$ . We will use a subset of this group, which we denote as  $\mathbb{P}_l$ , which includes all operators  $P = P_1 \otimes P_2 \otimes \dots \otimes P_l$  but not the multiplicative factors. We will use the fact that Pauli operators anti commute;  $ZX = -XZ$ .

Let  $\mathfrak{C}_l$  denote the  $l$ -qubit Clifford group. Recall that it is a finite subgroup of unitaries acting on  $l$  qubits generated by the Hadamard matrix  $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ , by  $K = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$ , and by controlled-NOT (CNOT) which maps  $|a, b\rangle$  to  $|a, a \oplus b\rangle$  (for bits  $a, b$ ). The Clifford group is characterized by the property that it maps the Pauli group  $\mathbb{P}_l$  to itself, up to a phase  $\alpha \in \{\pm 1, \pm i\}$ . That is:  $\forall C \in \mathfrak{C}_l, P \in \mathbb{P}_l : \alpha C P C^\dagger \in \mathbb{P}_l$

The Toffoli gate  $T$  maps  $|a, b, c\rangle$  to  $|a, b, c \oplus ab\rangle$  (for  $a, b, c \in \{0, 1\}$ ). We will use the fact that the set consisting of the Toffoli gate and the Hadamard gate is a universal gate set for quantum circuits ([Shi03]).

For density matrices  $\rho, \sigma$ , the trace distance  $\|\rho - \sigma\|_{tr}$  is equal to:

$$\|\rho - \sigma\|_{tr} = \frac{1}{2} \text{Tr}(\sqrt{(\rho - \sigma)^2}) \quad (19)$$

The following lemma relates the Hellinger distance (as given in (12)) and the trace distance of superpositions:

**Lemma 3.6** *Let  $X$  be a finite set and  $D_1, D_2$  be distributions over  $X$ . Let*

$$|\psi_1\rangle = \sum_{x \in X} \sqrt{D_1(x)} |x\rangle \quad \text{and} \quad |\psi_2\rangle = \sum_{x \in X} \sqrt{D_2(x)} |x\rangle .$$

*Then*

$$\| |\psi_1\rangle \langle \psi_1| - |\psi_2\rangle \langle \psi_2| \|_{tr} = \sqrt{1 - (1 - H^2(D_1, D_2))^2} .$$

## 4 Quantum Capable Homomorphic Encryption Schemes

As described in Section 2.5, a classical leveled fully homomorphic encryption scheme is quantum capable if an encrypted CNOT operation can be applied with respect to any ciphertext which occurs during the computation. We begin by formalizing the notion of such a ciphertext. This definition is dependent on the depth parameter  $L$  (see Definition 3.2). Let  $\mathcal{F}_L$  be the set of all functions which can be computed by circuits of depth  $L$ . Assume for convenience that all such functions have domain  $\{0, 1\}^l$  and range  $\{0, 1\}$ .

**Definition 4.1** *For a classical leveled fully homomorphic encryption scheme  $HE$ , let  $C_{HE}$  be the set of all ciphertexts which can occur during the computation:*

$$C_{HE} = \{HE.Eval_{evk}(f, HE.Enc_{pk}(\mu_1), \dots, HE.Enc_{pk}(\mu_l)) \mid f \in \mathcal{F}_L, \mu_1, \dots, \mu_l \in \{0, 1\}\} \quad (20)$$

We now define quantum capable homomorphic encryption schemes:

**Definition 4.2 (Quantum Capable Homomorphic Encryption Schemes)** *Let  $\lambda$  be the security parameter. Let  $HE$  be a classical leveled fully homomorphic encryption scheme.  $HE$  is quantum capable if there exists an encryption scheme  $AltHE$  such that the following conditions hold for all ciphertexts  $c \in C_{HE}$ .*

1. *There exists an algorithm  $HE.Convert$  which on input  $c$  produces an encryption  $\hat{c}$  under  $AltHE$ , where both  $c$  and  $\hat{c}$  encrypt the same value.*
2.  *$AltHE$  allows the XOR operation to be performed homomorphically. Moreover, the XOR operation is efficiently invertible using only the public key of  $AltHE$ .*
3. *There exists a distribution  $D$  which may depend on the parameters of  $HE$  and satisfies the following conditions:*

(a) *The Hellinger distance between the following two distributions is negligible in  $\lambda$ :*

$$\{AltHE.Enc_{pk}(\mu; r) \mid (\mu, r) \xleftarrow{\$} D\} \quad (21)$$

*and*

$$\{AltHE.Enc_{pk}(\mu; r) \oplus_H \hat{c} \mid (\mu, r) \xleftarrow{\$} D\} \quad (22)$$

*where  $\oplus_H$  represents the homomorphic XOR operation.*

(b) It is possible for a BQP server to create the following superposition given access to the public key  $pk$ :

$$\sum_{\mu \in \{0,1\}, r} \sqrt{D(\mu, r)} |\mu, r\rangle \quad (23)$$

(c) Given  $y = \text{AltHE.Enc}_{pk}(\mu_0; r_0)$  where  $(\mu_0, r_0)$  is sampled from  $D$ , it must be possible to compute  $\mu_0, r_0$  given the secret key and possibly additional trapdoor information (which can be computed as part of the key generation procedure).

For convenience, we will assume that AltHE and HE have the same public/secret key; this is the case in the example quantum capable scheme we provide in Section 5 and also simplifies Section 6. However, this assumption is not necessary.

We prove the following claim, which formalizes the encrypted CNOT operation given in Section 2.4:

**Claim 4.3** *Let HE be a quantum capable homomorphic encryption scheme and let  $c \in C_{HE}$  be a ciphertext encrypting a bit  $s$ . Consider a BQP machine with access to  $c$  and a state  $|\psi\rangle$  on two qubits. The BQP machine can compute a ciphertext  $y = \text{AltHE.Enc}_{pk}(\mu_0, r_0)$ , a string  $d$  and a state within negligible trace distance of the following ideal state:*

$$(Z^{d \cdot ((\mu_0, r_0) \oplus (\mu_1, r_1))} \otimes X^{\mu_0}) \text{CNOT}_{1,2}^s |\psi\rangle \quad (24)$$

for  $(\oplus_H$  is the homomorphic XOR operation)

$$\text{AltHE.Enc}_{pk}(\mu_0, r_0) = \text{AltHE.Enc}_{pk}(\mu_1, r_1) \oplus_H \text{HE.Convert}(c) \quad (25)$$

**Proof of Claim 4.3:** The server first computes  $\hat{c} = \text{HE.Convert}(c)$ . He then applies the encrypted CNOT operation, as described in Section 2.4. Recall that in Section 2.4, we used  $f_0$  to denote the encryption function of AltHE and  $f_1$  to denote the shift of  $f_0$  by the homomorphic XOR (which we denote as  $\oplus_H$ ) of  $\hat{c}$ . At the stage of (9), the server holds the following state:

$$\sum_{a,b,\mu \in \{0,1\}, r} \alpha_{ab} \sqrt{D(\mu, r)} |a\rangle |b\rangle |\mu, r\rangle |f_a(r)\rangle \quad (26)$$

$$= \sum_{a,b,\mu \in \{0,1\}, r} \alpha_{ab} \sqrt{D(\mu, r)} |a\rangle |b\rangle |\mu, r\rangle |\text{AltHE.Enc}_{pk}(\mu; r) \oplus_H a \cdot \hat{c}\rangle \quad (27)$$

Fix  $a = 1$  and  $b$  and consider the resulting state:

$$\sum_{\mu_0 \in \{0,1\}, r_0} \sqrt{D(\mu_1, r_1)} |1, b\rangle |\mu_1, r_1\rangle |\text{AltHE.Enc}_{pk}(\mu_0; r_0)\rangle \quad (28)$$

where  $\mu_1, r_1$  are defined with respect to  $\mu_0, r_0$  and  $\hat{c}$  as in (25). We can now apply Lemma 3.6 with reference to the distributions in (21) and (22), which are negligibly close. As a result, we obtain that the following state is within negligible trace distance of (28):

$$\sum_{\mu_0 \in \{0,1\}, r_0} \sqrt{D(\mu_0, r_0)} |1, b\rangle |\mu_1, r_1\rangle |\text{AltHE.Enc}_{pk}(\mu_0; r_0)\rangle \quad (29)$$

It follows immediately that the state in (27) is within negligible trace distance of the following state:

$$\sum_{\mu_0 \in \{0,1\}, r_0} \sum_{a,b \in \{0,1\}} \alpha_{ab} \sqrt{D(\mu_0, r_0)} |a, b\rangle |\mu_a, r_a\rangle |\text{AltHE.Enc}_{pk}(\mu_0; r_0)\rangle \quad (30)$$

Observe that, when measured, the state in (30) collapses exactly to the state in (6). The statement of Claim 4.3 follows.

□

## 5 Example of Quantum Capable Homomorphic Encryption Scheme

This section is dedicated to showing that the dual of the fully homomorphic encryption scheme in [GSW13] is quantum capable. We begin by presenting a scheme called Dual in Section 5.1, which is the dual of the encryption scheme from [Reg05]. In Section 5.2, we use the framework of [GSW13] to extend Dual to a scheme called DualHE, which we prove in Theorem 5.1 is a classical leveled fully homomorphic encryption scheme. Finally, in Section 5.3 (Theorem 5.2), we prove that DualHE is quantum capable .

We begin by listing our initial parameters. Let  $\lambda$  be the security parameter. All other parameters are functions of  $\lambda$ . Let  $q \geq 2$  be a power of 2. Let  $n, m \geq 1$  be polynomially bounded functions of  $\lambda$ , let  $N = (m + 1) \log q$  and let  $\beta_{init}$  be a positive integer such that the following conditions hold:

1.  $m = \Omega(n \log q)$  ,
  2.  $2\sqrt{n} \leq \beta_{init}$
- (31)

### 5.1 Dual Encryption Scheme

We first describe the dual scheme of [Reg05]. This scheme was originally given in [GPV07], but the presentation below is taken from Section 5.2.2 in [Pei15]. This scheme will eventually serve as the scheme AltHE in Definition 4.2.

#### Scheme 5.1 Dual Encryption Scheme [GPV07]

- *Dual.KeyGen:* Choose  $\mathbf{e}_{sk} \in \{0, 1\}^m$  uniformly at random. Using the procedure  $\text{GENTRAP}(1^n, 1^m, q)$  from Theorem 3.5, sample a random trapdoor matrix  $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ , together with the trapdoor information  $t_{\mathbf{A}}$ . The secret key is  $\mathbf{sk} = (-\mathbf{e}_{sk}, 1) \in \mathbb{Z}_q^{m+1}$  and the trapdoor is  $t_{\mathbf{A}}$ . The public key is  $\mathbf{A}' \in \mathbb{Z}_q^{(m+1) \times n}$ , which is the matrix composed of  $\mathbf{A}$  (the first  $m$  rows) and  $\mathbf{A}^T \mathbf{e}_{sk} \bmod q$  (the last row).
- *Dual.Enc<sub>pk</sub>( $\mu$ ):* To encrypt a bit  $\mu \in \{0, 1\}$ , choose  $\mathbf{s} \in \mathbb{Z}_q^n$  uniformly and create  $\mathbf{e} \in \mathbb{Z}_q^{m+1}$  by sampling each entry from  $D_{\mathbb{Z}_q, \beta_{init}}$ . Output  $\mathbf{A}'\mathbf{s} + \mathbf{e} + (0, \dots, 0, \mu \cdot \frac{q}{2}) \in \mathbb{Z}_q^{m+1}$ .
- *Dual.Dec<sub>sk</sub>( $\mathbf{c}$ ):* To decrypt, compute  $b' = \mathbf{sk}^T \mathbf{c} \in \mathbb{Z}_q$ . Output 0 if  $b'$  is closer to 0 than to  $\frac{q}{2} \bmod q$ , otherwise output 1.

We make a few observations:

- For a ciphertext  $c$  with error  $\mathbf{e}$  such that  $\|\mathbf{e}\| < \frac{q}{4\sqrt{m+1}}$ , the decryption procedure will operate correctly (since  $\mathbf{sk}^T \mathbf{A}' = 0$ ).
- The trapdoor  $t_{\mathbf{A}}$  can be used to recover the randomness  $\mathbf{s}, \mathbf{e}$  from a ciphertext. To see this, note that the first  $m$  entries of the ciphertext can be written as  $\mathbf{A}\mathbf{s} + \mathbf{e}'$ , where  $\mathbf{e}' \in \mathbb{Z}_q^m$ . Therefore, the inversion algorithm INVERT in Theorem 3.5 outputs  $\mathbf{s}, \mathbf{e}$  on input  $\mathbf{A}\mathbf{s} + \mathbf{e}'$  and  $t_{\mathbf{A}}$  as long as  $\|\mathbf{e}'\| < \frac{q}{C_T \sqrt{n \log q}}$  for  $C_T$  the universal constant in Theorem 3.5.
- This scheme is naturally additively homomorphic; adding two ciphertexts encrypting  $\mu_0$  and  $\mu_1$  results in a ciphertext encrypting  $\mu_0 \oplus \mu_1$ .

## 5.2 Leveled Fully Homomorphic Encryption Scheme from Dual

We can extend the scheme Dual into a leveled fully homomorphic encryption scheme DualHE in the same way that the standard LWE scheme from [Reg05] is extended in [GSW13]. Namely, we map the ciphertexts to matrices and encrypt the bit  $\mu$  in a matrix (the key generation procedure remains the same). We begin with the required preliminaries and then describe the scheme DualHE. Next, we prove a property of DualHE which is crucial for quantum capability: the ciphertexts retain the same form throughout the computation. Finally, we show in Theorem 5.1 that for a strengthened version of the parameters in (31), DualHE a leveled fully homomorphic encryption scheme.

To describe this scheme, we will require two operations used in [GSW13]. The first is the linear operator  $\mathbf{G} \in \mathbb{Z}_q^{(m+1) \times N}$  ( $N = (m+1) \log_2 q$ ), which converts a binary representation back to the original representation in  $\mathbb{Z}_q^{m+1}$ . More precisely, consider the  $N$  dimensional vector  $\mathbf{a} = (a_{1,0}, \dots, a_{1,l-1}, \dots, a_{m+1,0}, \dots, a_{m+1,l-1})$ , where  $l = \log_2 q$ .  $\mathbf{G}$  performs the following mapping:

$$\mathbf{G}(\mathbf{a}) = \left( \sum_{j=0}^{\log_2 q - 1} 2^j \cdot a_{1,j}, \dots, \sum_{j=0}^{\log_2 q - 1} 2^j \cdot a_{m+1,j} \right) \quad (32)$$

Observe that  $\mathbf{G}$  is well defined even if  $\mathbf{a}$  is not a 0/1 vector. We will call the non linear inverse operation  $G^{-1}$ , which converts  $\mathbf{a} \in \mathbb{Z}_q^{m+1}$  to its binary representation (a vector in  $\mathbb{Z}_2^N$ ).  $G^{-1}$  can also be applied to a matrix by converting each column. Note that  $\mathbf{G}G^{-1}$  is the identity operation. In terms of homomorphic evaluation, we will only consider the NAND gate, since we are only concerned with applying Boolean circuits. The scheme can be extended to arithmetic circuits over  $\mathbb{Z}_q$ , as described in further detail in [GSW13].

The description of the scheme given below is derived from talks ([Bra], [Wic]) describing the scheme in [GSW13]. It is equivalent to the description given in [GSW13], but is more convenient for our purposes.

### Scheme 5.2 DualHE: Classical Leveled FHE Scheme from Dual

- *DualHE.KeyGen*: This procedure is the same as *Dual.KeyGen*.
- *DualHE.Enc<sub>pk</sub>( $\mu$ )*: To encrypt a bit  $\mu \in \{0, 1\}$ , choose  $\mathbf{S} \in \mathbb{Z}_q^{n \times N}$  uniformly at random and create  $\mathbf{E} \in \mathbb{Z}_q^{(m+1) \times N}$  by sampling each entry from  $D_{\mathbb{Z}_q, \beta_{init}}$ . Output  $\mathbf{A}'\mathbf{S} + \mathbf{E} + \mu\mathbf{G} \in \mathbb{Z}_q^{(m+1) \times N}$ .
- *DualHE.Eval( $\mathbf{C}_0, \mathbf{C}_1$ )*: To apply the NAND gate, on input  $\mathbf{C}_0, \mathbf{C}_1$  output  $\mathbf{G} - \mathbf{C}_0 \cdot G^{-1}(\mathbf{C}_1)$ .

For quantum capability, we will also require the following algorithm, which converts a ciphertext under DualHE to a ciphertext under Dual:

- *DualHE.Convert( $\mathbf{C}$ )*: Output column  $N$  of  $\mathbf{C}$

Given this algorithm, we can state decryption in terms of Dual<sup>1</sup>:

- *DualHE.Dec<sub>sk</sub>( $\mathbf{C}$ )*: Output *Dual.Dec<sub>sk</sub>(DualHE.Convert( $\mathbf{C}$ ))*

---

<sup>1</sup>This is equivalent to the decryption algorithm of [GSW13], which is as follows. Let  $\mathbf{u} = (0, \dots, 0, 1) \in \mathbb{Z}_q^{m+1}$ . To decrypt, compute  $b' = \mathbf{sk}^T \mathbf{C} G^{-1}(\frac{q}{2} \mathbf{u})$ . Output 0 if  $b'$  is closer to 0 than to  $\frac{q}{2} \bmod q$ , otherwise output 1.



### 5.2.1 Ciphertext Form

We will rely on the fact that, throughout the computation of a Boolean circuit of depth  $L$ , a ciphertext encrypting a bit  $\mu$  can be written in the following form:

$$\mathbf{A}'\mathbf{S} + \mathbf{E} + \mu\mathbf{G} \quad (33)$$

where  $\|\mathbf{E}\|_\infty \leq \beta_{init}(N+1)^L$ . This is clearly the structure of the ciphertext immediately after encryption and we now show that this ciphertext structure is maintained after the NAND operation. The NAND operation is performed by computing:

$$\mathbf{G} - \mathbf{C}_0 \cdot G^{-1}(\mathbf{C}_1) \quad (34)$$

Assume the ciphertexts we begin with are  $\mathbf{C}_b = \mathbf{A}'\mathbf{S}_b + \mathbf{E}_b + \mu_b\mathbf{G}$  for  $b \in \{0, 1\}$ . Using the fact that  $\mathbf{G}G^{-1}$  is the identity operation, it is easy to see that the result of the NAND operation is:

$$\mathbf{A}'\mathbf{S}' + \mathbf{E}' + (1 - \mu_0\mu_1)\mathbf{G} \quad (35)$$

for

$$\mathbf{S}' = -\mathbf{S}_0 \cdot G^{-1}(\mathbf{C}_1) - \mu_0\mathbf{S}_1 \quad (36)$$

$$\mathbf{E}' = -\mathbf{E}_0 \cdot G^{-1}(\mathbf{C}_1) - \mu_0\mathbf{E}_1 \quad (37)$$

Note that if both  $\|\mathbf{E}_0\|_\infty$  and  $\|\mathbf{E}_1\|_\infty$  are at most  $\beta$ , then  $\|\mathbf{E}'\|_\infty \leq \beta(N+1)$ . It follows that if the scheme DualHE is used to compute a circuit of depth  $L$ ,  $\|\mathbf{E}\|_\infty \leq \beta_{init}(N+1)^L$  for all ciphertexts throughout the computation.

### 5.2.2 Encryption Conversion

We now use the above property to prove the correctness of DualHE.Convert. Assume we begin with a ciphertext  $\mathbf{C} = \mathbf{A}'\mathbf{S} + \mathbf{E} + \mu\mathbf{G}$  under DualHE. The ciphertext  $\mathbf{c}$  (under Dual) will be column  $N$  of  $\mathbf{C}$ . To see why this is correct, first note that all individual columns of  $\mathbf{A}'\mathbf{S} + \mathbf{E}$  are of the form  $\mathbf{A}'\mathbf{s} + \mathbf{e}$ . Second, observe that column  $N$  of  $\mu\mathbf{G}$  is equal to  $(0, \dots, 0, \mu \cdot \frac{q}{2})$ .

### 5.2.3 Proof of Correctness and Security of Scheme 5.2

The above two sections allow us to easily prove the following theorem:

**Theorem 5.1** *Let  $\lambda$  be the security parameter. There exists a function  $\eta_c$  which is logarithmic in  $\lambda$  such that DualHE is IND-CPA secure and leveled fully homomorphic under the hardness assumption  $\text{LWE}_{n,q,D_{\mathbb{Z}_q},\beta_{init}}$  if the conditions in (31) as well as the following condition are satisfied:*

$$\beta_{init}(N+1)^{\eta_c} < \frac{q}{4(m+1)}. \quad (38)$$

*Proof:* We prove that DualHE is IND-CPA secure by relying on the hardness of  $\text{LWE}_{n,m,q,D_{\mathbb{Z}_q},\beta_{init}}$  (i.e. the hardness of LWE with a superpolynomial noise ratio), which implies that the ciphertext is computationally indistinguishable from a uniform string. We can use this LWE assumption as long as the public key  $\mathbf{A}'$  is statistically indistinguishable from a uniformly random matrix (see Section 3.3). Since  $\mathbf{A}$  is selected from a distribution which is statistically indistinguishable from the uniform distribution and  $m = \Omega(n \log q)$ ,  $\mathbf{A}'$  is statistically indistinguishable from uniform due to the leftover hash lemma in [ILL89] (see [Reg05] or [Pei15] for more details).

We now show that DualHE is leveled fully homomorphic. From Section 5.2.1, it is clear that the evaluation of

the NAND operation is correct and that DualHE is compact. Let  $\eta_c$  be larger than the depth of the decryption circuit of DualHE, which is logarithmic in  $\lambda$ . If we show that the decryption procedure operates correctly after evaluation of a circuit of depth  $\eta_c$ , the standard bootstrapping technique<sup>2</sup> of [Gen09] can be used to turn DualHE into a leveled fully homomorphic encryption scheme. Due to Section 5.2.1, we can assume a ciphertext resulting from a circuit of depth  $\eta_c$  can be written as  $\mathbf{A}'\mathbf{S} + \mathbf{E} + \mu\mathbf{G}$ , where  $\|\mathbf{E}\|_\infty \leq \beta_{init}(N+1)^{\eta_c}$ . It is easy to check that the decryption procedure operates correctly as long as

$$\|\mathbf{E}\|_\infty < \frac{q}{4(m+1)} \quad (39)$$

The condition in (39) is implied by the condition in (38).  $\square$

### 5.3 Quantum Capability of DualHE

We now prove the following theorem:

**Theorem 5.2** *Let  $\lambda$  be the security parameter, let  $\eta_c$  be the logarithmic function in Theorem 5.1, and let  $\eta$  be an arbitrary logarithmic function in  $\lambda$ . Assume the choice of parameters satisfies the conditions in (31) as well as the following condition:*

$$\beta_{init}(N+1)^{\eta+\eta_c} < \frac{q}{4(m+1)}. \quad (40)$$

*Under the hardness assumption of  $\text{LWE}_{n,q,D_{\mathbb{Z}_q},\beta_{init}}$ , the scheme DualHE is quantum capable.*

Observe that the only change in parameters involved in making DualHE quantum capable is increasing the circuit depth by an additive logarithmic factor; this does not change the underlying computational assumption of the hardness of learning with errors with a superpolynomial noise ratio.

*Proof:* To prove Theorem 5.2, we begin by noting that DualHE is leveled fully homomorphic by Theorem 5.1. We now show that DualHE is quantum capable, by listing the requirements for quantum capability and proving that each holds. The scheme corresponding to AltHE will be Dual (Section 5.1). Recall the definition of  $C_{\text{DualHE}}$  from Definition 4.1. For all ciphertexts  $c \in C_{\text{DualHE}}$ :

1. *There exists an algorithm  $\text{DualHE.Convert}_{pk}$  which on input  $c$  produces an encryption  $\hat{c}$  under Dual, where both  $c$  and  $\hat{c}$  encrypt the same value.*

See Section 5.2.2.

2. *Dual allows the XOR operation to be performed homomorphically. Moreover, the XOR operation is efficiently invertible using only the public key of AltHE.*

See Section 5.1.

3. *There exists a distribution  $D$  which satisfies the following conditions:*

- (a) *The Hellinger distance between the following two distributions is negligible in  $\lambda$ :*

$$\{\text{Dual.Enc}_{pk}(\mu; r) \mid (\mu, r) \xleftarrow{\$} D\} \quad (41)$$

---

<sup>2</sup>A pure fully homomorphic encryption scheme can be obtained by assuming circular security. A leveled fully homomorphic encryption scheme can be obtained by producing a string of public and secret keys and encrypting each secret key under the next public key - see Section 4.1 of [Gen09].

and

$$\{\text{Dual.Enc}_{pk}(\mu; r) \oplus_H \hat{c} | (\mu, r) \xleftarrow{\$} D\} \quad (42)$$

where  $\oplus_H$  represents the homomorphic XOR operation.

Let

$$\beta_f = \beta_{init}(N+1)^{\eta_c + \eta} \quad (43)$$

The distribution  $D$  will sample  $\mu, s$  uniformly at random and will sample  $\mathbf{e}$  from the discrete Gaussian distribution  $D_{\mathbb{Z}_q^{m+1}, \beta_f}$ . Assume that  $\hat{c} = \text{DualHE.Convert}(c) = \mathbf{A}'\mathbf{s}' + \mathbf{e}' + (0, \dots, 0, s \cdot \frac{q}{2})$  where  $\|\mathbf{e}'\| \leq \beta_{init}(N+1)^{\eta_c} \sqrt{m+1}$ . We can assume this since we know the format of the ciphertext throughout the computation (see Section 5.2.1). The two distributions corresponding to (41) and (42) are as follows:

$$\{\mathbf{A}'\mathbf{s} + \mathbf{e} + (0, \dots, 0, \mu \cdot \frac{q}{2}) | (\mu, s, \mathbf{e}) \xleftarrow{\$} D\} \quad (44)$$

and

$$\{\mathbf{A}'(\mathbf{s} + \mathbf{s}') + \mathbf{e} + \mathbf{e}' + (0, \dots, 0, \mu \cdot \frac{q}{2}) | (\mu, s, \mathbf{e}) \xleftarrow{\$} D\} \quad (45)$$

The Hellinger distance between the distributions in (44) and (45) is equal to the distance between the following two distributions:

$$\{\mathbf{e} | \mathbf{e} \xleftarrow{\$} D_{\mathbb{Z}_q^{m+1}, \beta_f}\} \quad (46)$$

and

$$\{\mathbf{e} + \mathbf{e}' | \mathbf{e} \xleftarrow{\$} D_{\mathbb{Z}_q^{m+1}, \beta_f}\} \quad (47)$$

Since  $\|\mathbf{e}'\| \leq \beta_{init}(N+1)^{\eta_c} \sqrt{m+1}$  and  $\frac{\beta_f}{\beta_{init}(N+1)^{\eta_c}}$  is equal to the superpolynomial function  $(N+1)^\eta$ , Lemma 3.3 shows that the distance between the two distributions is negligible.

- (b) *It is possible for a BQP server to create the following superposition:*

$$\sum_{\mu \in \{0,1\}, r} \sqrt{D(\mu, r)} |\mu, r\rangle \quad (48)$$

In this case,  $r = (s, \mathbf{e})$ .  $D$  samples  $\mu$  and  $s$  according to the uniform distribution and samples  $\mathbf{e}$  according to the discrete Gaussian distribution  $D_{\mathbb{Z}_q^{m+1}, \beta_f}$ . It is easy for a BQP server to create a superposition over a discrete Gaussian (see Lemma 3.12 in [Reg05]<sup>3</sup>).

- (c) *Given  $y = \text{AltHE.Enc}_{pk}(\mu_0; r_0)$  where  $(\mu_0, r_0)$  is sampled from  $D$ , it must be possible to compute  $\mu_0, r_0$  given the secret key and possibly additional trapdoor information (which can be computed as part of the key generation procedure).*

We first show that  $\mu_0, r_0$  can be recovered from  $y$ . Assume  $y$  has error  $\mathbf{e} \in \mathbb{Z}_q^{m+1}$ . Since  $\mathbf{e}$  is sampled from  $D_{\mathbb{Z}_q^{m+1}, \beta_f}$ ,  $\|\mathbf{e}\| \leq \sqrt{m+1}\beta_f$ . Therefore, it is possible to compute  $\mu_0$  as long as  $\beta_f < \frac{q}{4(m+1)}$  (see Section 5.1). Second, it is possible to recover the randomness  $r_0$  of  $y$  as long as the lattice trapdoor is applicable. As stated in Theorem 3.5, the lattice trapdoor is applicable if  $\beta_f < \frac{q}{C_T \sqrt{n(m+1) \log q}}$ .

Combining these two conditions, we require that:

$$\beta_f < \min\left(\frac{q}{C_T \sqrt{n(m+1) \log q}}, \frac{q}{4(m+1)}\right) = \frac{q}{4(m+1)} \quad (49)$$

---

<sup>3</sup>Taken from [BCM<sup>+</sup>18] - specifically, the state can be created using a technique by Grover and Rudolph ([GR02]), who show that in order to create such a state, it suffices to have the ability to efficiently compute the sum  $\sum_{x=c}^d D_{\mathbb{Z}_q, B_P}(x)$  for any  $c, d \in \{-\lfloor \sqrt{B_P} \rfloor, \dots, \lceil \sqrt{B_P} \rceil\} \subseteq \mathbb{Z}_q$  and to within good precision. This can be done using standard techniques used in sampling from the normal distribution.

The equality follows since  $m = \Omega(n \log q)$ . Given the definition of  $\beta_f$  in (43), this condition is satisfied by (40).

□

## 6 Extension to Leveled Fully Homomorphic Encryption

We now present the construction of a quantum leveled fully homomorphic encryption scheme from a quantum capable classical leveled fully homomorphic encryption scheme, as described in Section 2.7. We first provide a full description of the scheme (Section 6.1) and then proceed to proving that it is a quantum leveled FHE scheme. Correctness of evaluation follows almost immediately from the correctness of the encrypted CNOT operation (see Claim 4.3), while CPA security follows along the lines described in Section 2.7.

Assume there are  $L$  levels of the quantum circuit to be computed, where each level consists of Clifford gates, followed by a layer of non intersecting Toffoli gates. Note that this circuit arrangement increases the depth of the original circuit by a factor of at most 2. Let the depth of the classical circuit corresponding to each level of this circuit be  $L_c$  (this includes decrypting and recovering randomness from ciphertexts corresponding to the encrypted CNOT operations from the previous level, performing the Pauli key updates corresponding to the encrypted CNOT operations from the previous level, and performing the Pauli key updates corresponding to the Clifford and Toffoli gates of the current level). See Section 2.7 for a reminder of the above description.

The scheme is quite straightforward: the server initially receives a quantum standard basis state encrypted under Pauli keys (which can be thought of as a one time padded classical string), along with the Pauli keys encrypted under a quantum capable classical homomorphic encryption scheme (which we call HE) and a string of evaluation keys (one for each level). Each evaluation key consists of the evaluation key of HE, encrypted secret key/ trapdoor information and a fresh public key for each level. Recall that the evaluation key is of this form since we need to use a fresh public/ secret key pair for each encrypted CNOT operation; this allows encryption of the secret key/ trapdoor of each level under a new, independent public key (see Section 2.7). The server then applies Toffoli and Hadamard gates (which compose a universal gate set) as described in Section 2. Finally, the decryption consists of the client first decrypting the encryptions of the Pauli keys, and then using the Pauli keys to decrypt the final measurement result sent by the server.

**Scheme 6.1 Quantum Leveled Fully Homomorphic Encryption** Let HE be a classical leveled fully homomorphic encryption scheme which is quantum capable for depth  $L_c$ .

- $QHE.KeyGen(1^\lambda, 1^L)$ :
  1. For  $1 \leq i \leq L + 1$ , let  $(pk_i, evk_i, sk_i, t_{sk_i}) = HE.Keygen(1^\lambda, 1^{L_c})$ , where  $t_{sk_i}$  is the trapdoor information required for randomness recovery from ciphertexts.
  2. The public key  $pk$  is  $pk_1$  and the secret key  $sk$  is  $sk_{L+1}$ . The evaluation key  $evk$  consists of  $(evk_1, \dots, evk_{L+1})$  as well as  $(pk_{i+1}, HE.Enc_{pk_{i+1}}(sk_i), HE.Enc_{pk_{i+1}}(t_{sk_i}))$  for  $1 \leq i \leq L$ .
- $QHE.Enc_{pk}(m)$ : For a message  $m \in \{0, 1\}^\lambda$ , the encryption is  $(Z^z X^x |m\rangle, HE.Enc_{pk_1}(z, x))$ <sup>4</sup>, where  $z, x \in \{0, 1\}^\lambda$  are chosen at random. Note that  $Z^z X^x |m\rangle$  can be represented as the classical string  $x \oplus m$ .

---

<sup>4</sup>Observe that this encryption can immediately be extended to quantum states by replacing  $m$  with a  $\lambda$  qubit state  $|\psi\rangle$ . The decryption can also be extended in the same manner.

- *QHE.Dec<sub>sk</sub>*: The input is a classical message  $m \in \{0, 1\}^\lambda$  and encryptions of  $z, x \in \{0, 1\}^\lambda$  under  $pk_{L+1}$ . The encryptions are first decrypted using  $sk_{L+1}$  to obtain  $z, x$ . The decrypted message is  $Z^z X^x |m\rangle$ , which can be represented as  $x \oplus m$ .
- *QHE.Eval*: Clifford gates and Toffoli gates are applied to an encrypted state as follows:
  1. To apply a Clifford gate, the Clifford is applied to the Pauli one time padded input state and the encrypted Pauli keys are homomorphically updated according to which Clifford gate was applied (see Section A.2).
  2. To apply a Toffoli gate:
    - (a) The Toffoli gate is applied to the Pauli one time padded state. Assume the Toffoli is applied on top of the Pauli one time pad  $Z^z X^x \in \mathbb{P}_3$ .
    - (b) The Pauli key encryptions are homomorphically updated according to  $P_{zx}$  (see Section A.1).
    - (c) Three encrypted CNOT operations are used to correct  $C_{zx}$  (see Section A.3 for details on  $C_{zx}$  and  $P_{zx}$ ). As part of each operation, the Pauli key encryptions are homomorphically updated (see Claim 6.3 for a full description of how this is done).

## 6.1 CPA Security

In this section, we prove the following theorem:

**Theorem 6.1** *The scheme presented in Section 6.1 is IND-CPA secure.*

*Proof:* To prove CPA security as defined in Definition 3.1, we show that for any polynomial time adversary  $\mathcal{A}$ , there exists a negligible function  $\mu(\cdot)$  such that

$$\text{Adv}_{\text{CPA}}[\mathcal{A}] = |\Pr[\mathcal{A}(pk, evk, \text{QHE.Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, evk, \text{QHE.Enc}_{pk}(1)) = 1]| = \mu(\lambda) \quad (50)$$

where  $(pk, evk, sk) \leftarrow \text{QHE.Keygen}(1^\lambda)$ .

The only difficulty in proving (50) is that encryptions of  $sk_1$  and  $t_{sk_1}$  are also given to the attacker as part of the evaluation key; we need to prove that this information can be replaced with encryptions of 0. This can be done via standard techniques in proving security of leveled homomorphic encryption schemes (see Section 4.1 in [Gen09]). We include the proof for completeness.

We proceed through  $L$  hybrids. In the final hybrid, the attacker is given only the public key  $pk_1$  and the evaluation key  $evk' = (evk_1, pk_2, evk_2, \dots, pk_{L+1}, evk_{L+1})$  (along with many encryptions of 0). CPA security at this point follows immediately (by replacing the encryptions of  $z, x$  with 0 and then using Lemma 2.1). The hybrids are as follows:

Hyb<sub>L+1</sub>: The evaluation key is as described in Section 6.1.

For  $1 \leq i \leq L$ , where  $i$  is decreasing:

Hyb <sub>$i$</sub> : The evaluation key is the same as in Hyb <sub>$i+1$</sub> , except  $\text{HE.Enc}_{pk_{i+1}}(t_{sk_i})$  and  $\text{HE.Enc}_{pk_{i+1}}(sk_i)$  are replaced with encryptions of 0.

Note that in Hyb <sub>$i$</sub> , the evaluation key does not contain secret key or trapdoor information corresponding to public keys  $pk_i, \dots, pk_{L+1}$ . More specifically, the evaluation key in Hyb<sub>1</sub> is  $evk'$  (all the encryptions of secret keys and trapdoors have been replaced by encryptions of 0).

First,  $\text{Hyb}_{L+1}$  is computationally indistinguishable from  $\text{Hyb}_L$  due to the CPA security of HE under  $pk_{L+1}$  (note that encryptions of  $sk_{L+1}$  and  $t_{sk_{L+1}}$  were not provided as part of the evaluation key). For all  $1 \leq i \leq L-1$ ,  $\text{Hyb}_{i+1}$  is indistinguishable from  $\text{Hyb}_i$  due to the CPA security of HE under  $pk_{i+1}$ . This is because  $\text{Hyb}_{i+1}$  has no secret key or trapdoor information corresponding to  $pk_{i+1}$ .

It follows that there exists a negligible function  $\mu_C$  such that the CPA security of QHE

$$|\Pr[\mathcal{A}(pk, evk, \text{QHE.Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, evk, \text{QHE.Enc}_{pk}(1)) = 1]| \quad (51)$$

can be upper bounded as follows:

$$\begin{aligned} \dots &\leq \mu_C L + |\Pr[\mathcal{A}(pk, evk', \text{QHE.Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, evk', \text{QHE.Enc}_{pk}(1)) = 1]| \\ &\leq \mu_C (L+1) \end{aligned} \quad (52)$$

where  $(pk, evk, sk) \leftarrow \text{QHE.Keygen}(1^\lambda)$  and  $evk' = (evk_1, pk_2, evk_2, \dots, pk_{L+1}, evk_{L+1})$ .

□

## 6.2 Quantum Leveled FHE

In this section, we prove the following theorem:

**Theorem 6.2** *The scheme QHE presented in Section 6.1 is a quantum leveled fully homomorphic encryption scheme.*

Combining Theorem 6.2 with Theorem 5.2 provides the main result of our paper (stated informally in Theorem 1.1). To prove Theorem 6.2, we need to prove that QHE can evaluate depth  $L$  quantum circuits. This is taken care of by the following claim:

**Claim 6.3** *Assume the underlying classical encryption scheme HE of QHE is quantum capable for depth  $L_c$ . Then there exist  $z', x' \in \{0, 1\}^2$  such that a BQP machine with access to a ciphertext  $c$  encrypting  $s$  under  $pk_i$ , a quantum state  $Z^z X^x |\psi\rangle$  on two qubits, ciphertexts encrypting  $z, x$  under  $pk_i$ , and the evaluation key of QHE can compute a state within negligible trace distance of the following ideal state*

$$\text{CNOT}_{1,2}^s Z^{z'} X^{x'} |\psi\rangle \langle\psi| (Z^{z'} X^{x'})^\dagger (\text{CNOT}_{1,2}^s)^\dagger \quad (53)$$

as well as the encryptions of  $z', x'$  under  $pk_{i+1}$ .

*Proof:* Let  $c_{z,x,pk_i}$  be the concatenation of four ciphertexts, each encrypting a single bit of  $z, x$  under  $pk_i$ . The server applies the following operations:

1. As described in Section 2.4, the server applies the encrypted CNOT operation to the two qubit state  $Z^z X^x |\psi\rangle$  using the ciphertext  $\hat{c} = \text{HE.Convert}(c)$ . According to Claim 4.3, the server will obtain a ciphertext  $y = \text{AltHE.Enc}_{pk}(\mu_0, r_0)$ , a string  $d \in \{0, 1\}^m$  and a state within negligible trace distance of the following ideal state:

$$(Z^{d \cdot ((\mu_0, r_0) \oplus (\mu_1, r_1))} \otimes X^{\mu_0}) \text{CNOT}_{1,2}^s |\psi\rangle \quad (54)$$

where  $\text{AltHE.Enc}_{pk}(\mu_0; r_0) = \text{AltHE.Enc}_{pk}(\mu_1; r_1) \oplus_H \hat{c}$  and  $\oplus_H$  is the homomorphic XOR operation.

2. The server uses  $pk_{i+1}$  to compute  $\text{HE.Enc}_{pk_{i+1}}(c_{z,x,pk_i})$  and  $\text{HE.Enc}_{pk_{i+1}}(\hat{c}, y, d)$ .



3. The server computes the encryption of  $z, x$  under  $pk_{i+1}$  by homomorphically running the decryption circuit on inputs  $\text{HE.Enc}_{pk_{i+1}}(sk_i)$  and  $\text{HE.Enc}_{pk_{i+1}}(c_{z,x,pk_i})$ .
4. The server homomorphically computes  $(\mu_0, r_0)$  and  $(\mu_1, r_1)$ , using the ciphertexts encrypting  $t_{sk_i}, sk_i, \hat{c}, y, d$  (all encrypted with HE under public key  $pk_{i+1}$ ). The server then uses this result, along with the ciphertexts encrypting  $z, x, d$ , to homomorphically compute  $z' = z + (d \cdot ((\mu_0, r_0) \oplus (\mu_1, r_1)), 0)$  and  $x' = x + (0, \mu_0)$ . The result of this computation is the encryption of  $z', x'$  with HE under  $pk_{i+1}$ .

□

Theorem 6.2 follows from Theorem 6.1 and Claim 6.3:

**Proof of Theorem 6.2:** Theorem 6.1 shows QHE is IND-CPA secure. From the description of the scheme (Scheme 6.1), it is clear that QHE is compact. Since the number of Toffoli gates is polynomial in  $\lambda$ , Claim 6.3 (along with the triangle inequality) implies that the trace distance of the server's final state from the ideal (correct) final state is at most negligible in  $\lambda$ .

□

## 7 Acknowledgments

Thanks to Dorit Aharonov, Zvika Brakerski, Sanjam Garg, Stacey Jeffery, Zeph Landau, Umesh Vazirani and Thomas Vidick for many useful discussions.

## References

- [ABOE08] Dorit Aharonov, Michael Ben-Or, and Elad Eban. Interactive Proofs For Quantum Computations. *Arxiv preprint arXiv:0810.5375*, 2008.
- [ABOEM17] Dorit Aharonov, Michael Ben-Or, Elad Eban, and Urmila Mahadev. Interactive Proofs for Quantum Computations. *Arxiv preprint 1704.04487*, 2017.
- [Ban93] Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, 1993.
- [BCM<sup>+</sup>18] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. Certifiable randomness from a single quantum device. *Arxiv preprint 1804.00640*, 2018.
- [BFK08] Anne Broadbent, Joseph F. Fitzsimons, and Elham Kashefi. Universal blind quantum computation. *Arxiv preprint arXiv:0807.4154*, 2008.
- [BJ14] Anne Broadbent and Stacey Jeffery. Quantum homomorphic encryption for circuits of low T-gate complexity. *Arxiv preprint arXiv:1412.8766*, 2014.
- [BJ15] Anne Broadbent and Stacey Jeffery. Quantum homomorphic encryption for circuits of low  $t$ -gate complexity. Cryptology ePrint Archive, Report 2015/551, 2015. <http://eprint.iacr.org/2015/551>.
- [BLP<sup>+</sup>13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 575–584. ACM, 2013.

- [Bra] Zvika Brakerski. Fully homomorphic encryption. Simons Institute. <https://www.youtube.com/watch?v=08IvJAIvGJo>.
- [BV13] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based fhe as secure as pke. Cryptology ePrint Archive, Report 2013/541, 2013. <http://eprint.iacr.org/2013/541>.
- [BV14] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. Siam Journal on Computing, Vol 43, No 2, pp. 831-871, 2014. <http://epubs.siam.org/doi/pdf/10.1137/120868669>.
- [CCKW18] Alexandru Cojocaru, Leo Colisson, Elham Kashefi, and Petros Wallden. Delegated pseudo-secret random qubit generator, 2018.
- [DSS16] Yfke Dulek, Christian Schaffner, and Florian Speelman. *Quantum Homomorphic Encryption for Polynomial-Sized Circuits*, pages 3–32. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.
- [FK17] Joseph F. Fitzsimons and Elham Kashefi. Unconditionally verifiable blind quantum computation. *Phys. Rev. A*, 96:012303, Jul 2017.
- [Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. [crypto.stanford.edu/craig](http://crypto.stanford.edu/craig).
- [GPV07] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. Cryptology ePrint Archive, Report 2007/432, 2007. <http://eprint.iacr.org/2007/432>.
- [GR02] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions, 2002.
- [GSW13] Craig Gentry, Amit Sahai, and Brent 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>.
- [ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 12–24, New York, NY, USA, 1989. ACM.
- [LC17] Ching-Yi Lai and Kai-Min Chung. On statistically-secure quantum homomorphic encryption, 2017.
- [MDMF17] Atul Mantri, Tommaso F. Demarie, Nicolas C. Menicucci, and Joseph F. Fitzsimons. Flow ambiguity: A path towards classically driven blind quantum computation. *Phys. Rev. X*, 7:031004, Jul 2017.
- [MP11] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. Cryptology ePrint Archive, Report 2011/501, 2011. <http://eprint.iacr.org/2011/501>.
- [NS17] Michael Newman and Yaoyun Shi. Limitations on transversal computation through quantum homomorphic encryption, 2017.
- [OTF15] Yingkai Ouyang, Si-Hui Tan, and Joseph F. Fitzsimons. Quantum homomorphic encryption from quantum codes, 2015.

- [Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC*, pages 333–342, 2009.
- [Pei15] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. <http://eprint.iacr.org/2015/939>.
- [PRS17] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-lwe for any ring and modulus. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 461–473. ACM, 2017.
- [Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing, STOC '05*, pages 84–93, New York, NY, USA, 2005. ACM.
- [Shi03] Yaoyun Shi. Both toffoli and controlled-not need little help to do universal quantum computing. *Quantum Info. Comput.*, 3(1):84–92, January 2003.
- [TKO<sup>+</sup>16] Si-Hui Tan, Joshua A. Kettlewell, Yingkai Ouyang, Lin Chen, and Joseph F. Fitzsimons. A quantum approach to homomorphic encryption. scientific reports. Scientific Reports, 6., 2016.
- [Wic] Daniel Wichs. Homomorphic commitments and signatures. Simons Institute. <https://www.youtube.com/watch?v=1cQP1QYVjAI>.

## A One Time Pad Gate Application

The descriptions below are taken from [ABOEM17].

### A.1 Applying Pauli Operators

To apply a Pauli operator  $Z^a X^b$ , only the Pauli keys need to be updated. This is because the effect of updating the Pauli keys is applying a Pauli operator to the state:

$$\frac{1}{|\mathbb{P}_n|} \sum_{z,x \in \{0,1\}^n} Z^z X^x |\psi\rangle \langle\psi| (Z^z X^x)^\dagger \otimes |z+a, x+b\rangle \langle z+a, x+b| \quad (55)$$

is equal to

$$\frac{1}{|\mathbb{P}_n|} \sum_{z,x \in \{0,1\}^n} Z^{z+a} X^{x+b} |\psi\rangle \langle\psi| (Z^{z+a} X^{x+b})^\dagger \otimes |zx\rangle \langle zx| \quad (56)$$

which is equal to

$$\frac{1}{|\mathbb{P}_n|} \sum_{z,x \in \{0,1\}^n} Z^a X^b Z^z X^x |\psi\rangle \langle\psi| (Z^a X^b Z^z X^x)^\dagger \otimes |zx\rangle \langle zx| \quad (57)$$

since the global phases which come with commuting Pauli operators cancel out. Therefore, the server can update his encrypted Pauli keys by homomorphically adding  $a$  to the  $Z$  Pauli key and  $b$  to the  $X$  Pauli key.

## A.2 Applying Clifford Operators

Here we show that the Pauli one time pad encryption allows transversal application of Clifford gates. In other words, the server can apply the Clifford gate  $C$  on top of  $Z^z X^x |\psi\rangle$ . The resulting state is:

$$\frac{1}{|\mathbb{P}_n|} \sum_{z,x \in \{0,1\}^n} CZ^z X^x C^\dagger C |\psi\rangle \langle \psi| C^\dagger (CZ^z X^x C^\dagger)^\dagger \otimes |z, x\rangle \langle z, x| \quad (58)$$

Let  $CZ^z X^x C^\dagger = Z^{\hat{z}} X^{\hat{x}}$  (such a Pauli exists because Clifford operators preserve Pauli operators by conjugation). Now if the Pauli keys are updated from  $z, x$  to  $\hat{z}, \hat{x}$ , the state is:

$$\frac{1}{|\mathbb{P}_n|} \sum_{z,x \in \{0,1\}^n} Z^{\hat{z}} X^{\hat{x}} C |\psi\rangle \langle \psi| C^\dagger (Z^{\hat{z}} X^{\hat{x}})^\dagger \otimes |\hat{z}, \hat{x}\rangle \langle \hat{z}, \hat{x}| \quad (59)$$

which (after relabeling  $\hat{z}, \hat{x}$  as  $z, x$ ) is equal to

$$\frac{1}{|\mathbb{P}_n|} \sum_{z,x \in \{0,1\}^n} Z^z X^x C |\psi\rangle \langle \psi| C^\dagger (Z^z X^x)^\dagger \otimes |z, x\rangle \langle z, x| \quad (60)$$

Therefore, a Clifford can be applied on top of a Pauli one time pad by two operations: a Clifford applied on the state, as well as an update of the Pauli keys. The server can again update the Pauli keys homomorphically.

## A.3 Applying Toffoli Operators

The same trick used in the Clifford case (i.e. the key update in conjunction with the gate application) cannot be used to apply a Toffoli gate on top of the Pauli one time pad. This is because a Toffoli operator maps Pauli operators to Clifford operators:

$$\begin{aligned} TZ^{z_1} X^{x_1} \otimes Z^{z_2} X^{x_2} \otimes Z^{z_3} X^{x_3} T^\dagger &= CNOT_{1,3}^{x_2} CNOT_{2,3}^{x_1} \hat{Z}_{1,2}^{z_3} Z^{z_1+x_2 z_3} X^{x_1} \otimes Z^{z_2+x_1 z_3} X^{x_2} \otimes Z^{z_3} X^{x_1 x_2 + x_3} \\ &= C_{zx} P_{zx} \end{aligned} \quad (61)$$

where  $\hat{Z}$  is the controlled phase gate:

$$\hat{Z} |a, b\rangle = (-1)^{ab} |a, b\rangle \quad (62)$$

$$\hat{Z}^{z_3} = (\mathcal{I} \otimes H) CNOT_{1,2}^{z_3} (\mathcal{I} \otimes H) \quad (63)$$

If the server applies a Toffoli gate, a Pauli key update can only remove  $P_{zx}$ . The state will still have  $C_{zx}$  acting on it. Observe that  $C_{zx}$  consists only of CNOT gates and 2 Hadamard gates. The CNOT gates are difficult for the server to remove, since each is to a power which comes from the Pauli key (which is unknown to the server, since the server holds only the classically encrypted Pauli keys).

## B Additional Proofs

### B.1 Proof of Pauli Mixing Lemma (Lemma 2.1)

This proof is taken from [ABOEM17].

**Lemma 2.1 (Pauli Mixing)** For a matrix  $\rho$  on two spaces  $A, B$

$$\frac{1}{|\mathbb{P}_l|} \sum_{P \in \mathbb{P}_l} (P \otimes \mathcal{I}_B) \rho (P \otimes \mathcal{I}_B)^\dagger = \frac{1}{2^l} \mathcal{I}_A \otimes \text{Tr}_A(\rho)$$

Proof of **Lemma 2.1** : First, we write  $\rho$  as:

$$\sum_{ij} |i\rangle \langle j|_A \otimes \rho_{ij}$$

It follows that:

$$\text{Tr}_A(\rho) = \sum_i \rho_{ii}$$

Next, observe that:

$$\sum_{P \in \mathbb{P}_l} P |i\rangle \langle j| P^\dagger = \sum_{z, x \in \{0,1\}^l} X^x Z^z |i\rangle \langle j| (X^x Z^z)^\dagger \quad (64)$$

$$= \sum_{z, x \in \{0,1\}^l} \left( \sum_{z \in \{0,1\}^l} (-1)^{z \cdot (i \oplus j)} \right) X^x |i\rangle \langle j| (X^x)^\dagger \quad (65)$$

This expression is 0 if  $i \neq j$ . If  $i = j$ , we obtain  $2^l \mathcal{I}$ . Plugging in this observation to the expression in the claim, we have:

$$\frac{1}{|\mathbb{P}_l|} \sum_{P \in \mathbb{P}_l} (P \otimes \mathcal{I}_B) \rho (P \otimes \mathcal{I}_B)^\dagger = \frac{1}{|\mathbb{P}_l|} \sum_{ij} \sum_{P \in \mathbb{P}_l} P |i\rangle \langle j|_A P^\dagger \otimes \rho_{ij} \quad (66)$$

$$= \frac{1}{|\mathbb{P}_l|} \sum_i \sum_{P \in \mathbb{P}_l} P |i\rangle \langle i|_A P^\dagger \otimes \rho_{ii} \quad (67)$$

$$= \frac{2^l}{|\mathbb{P}_l|} \mathcal{I} \otimes \sum_i \rho_{ii} \quad (68)$$

$$= \frac{1}{2^l} \mathcal{I} \otimes \text{Tr}_A(\rho) \quad (69)$$

□