Delegating Quantum Computation in the Quantum Random Oracle Model

Jiayu Zhang[1]*

Boston University (jyz16@bu.edu)

Abstract. A delegation scheme allows a computationally weak client to use a server's resources to help it evaluate a complex circuit without leaking any information about the input (other than its length) to the server. In this paper, we consider delegation schemes for quantum circuits, where we try to minimize the quantum operations needed by the client. We construct a new scheme for delegating a large circuit family, which we call "C+P circuits". "C+P" circuits are the circuits composed of Toffoli gates and diagonal gates. Our scheme is non-interactive, requires small amount of quantum computation from the client (proportional to input length but independent of the circuit size), and can be proved secure in the quantum random oracle model, without relying on additional assumptions, such as the existence of fully homomorphic encryption. In practice the random oracle can be replaced by an appropriate hash function or block cipher, for example, SHA-3, AES.
This protocol allows a client to delegate the most expensive part of some quantum algorithms, for example, Shor's algorithm. The previous protocols that are powerful enough to delegate Shor's algorithm require either many client side quantum operations or the existence of FHE. The protocol requires asymptotically fewer quantum gates on the client side compared to running Shor's algorithm locally.
To hide the inputs, our scheme uses an encoding that maps one input qubit to multiple qubits. We then provide a novel generalization of classical garbled circuits ("reversible garbled circuits") to allow the computation of Toffoli circuits on this encoding. We also give a technique that can support the computation of phase gates on this encoding.
To prove the security of this protocol, we study key dependent message(KDM) security in the quantum random oracle model. KDM security was not previously studied in quantum settings.

Keywords: Quantum Computation Delegation· Quantum Cryptography· Garbled Circuit· Quantum Random Oracle· KDM Security

1 Introduction

In computation delegation, there is a client holding secret data φ and the description of circuit C that it wants to apply, but it doesn't have the ability to

* Supported in part by NSF awards IIS-1447700 and AF-1763786

[0] The full version of this paper can be found at http://arxiv.org/abs/1810.05234

compute $C(\varphi)$ itself. A delegation protocol allows the client to compute $C(\varphi)$ with the help from a more computationally powerful server. The delegation is *private* if the server cannot learn anything about the input φ during the protocol. After some communications, the client can decrypt the response from the server and get the computation result (see Figure 1.) This problem is important in the quantum setting: it's likely that quantum computers, when they are built, will be expensive, and made available as a remote service. If a client wants to do some quantum computation on secret data, a quantum computation delegation protocol is needed.

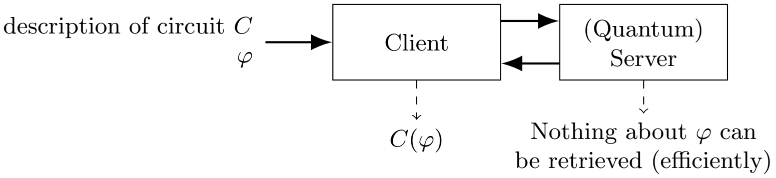
description of circuit C
φ
Client
(Quantum) Server
$C(\varphi)$
Nothing about φ can be retrieved (efficiently)

Fig. 1. Delegation of (quantum) computation

Delegation of computation is a central problem in modern cryptography, and has been studied for a long time in classical settings. Related works include multiparty computation, fully homomorphic encryption(FHE), etc. In the study of delegation, there are two key aspects: privacy and authenticity. This paper will focus on privacy.

We want the delegation protocol to be useful, efficient and secure. Previous work falls into two classes: some protocols have information-theoretical security, but they either can only support a small circuit class or require huge client side quantum resources (including quantum memories, quantum gates and quantum communications); other protocols rely on classical fully homomorphic encryption(FHE). This raises the following question:

Is it possible to delegate quantum computation for a large circuit family, with small amount of quantum resources on the client side, without assuming classical FHE?

In the classical world, Yao's garbled circuit answers this question. Garbled circuit is also a fundamental tool in many other cryptographic tasks, like multiparty computation and functional encryption.

Note When designing quantum cryptographic protocols, one factor that we care about is the "quantum resources" on the client side. The "quantum resources" can be defined as the sum of the cost of the following: (1)the size of quantum memory that the client needs; (2)the number of quantum gates that the client needs to apply; (3)the quantum communication that the client needs to make. Note that if the input (or computation, communication) is partly quantum and partly classical, we only consider the quantum part. Since the classical part is usually much easier to implement than the quantum part, as long as the classical part is polynomial, it's reasonable to ignore it and only consider the complexity of quantum resources. And we argue that it's better to consider the "client side quantum resources" instead of considering only the quantum memory size or quantum gates: on the one hand, we do not know which type of quantum computers will survive in the future, so it's better to focus on the cost estimate that is invariant to them; on the other hand, there may be some way to compose the protocol with other protocols to reduce the memory size, or simplify the gate set.

1.1 Our Contributions

In this paper we develop a non-interactive (1 round) quantum computation delegation scheme for "C+P circuits", the circuits composed of Toffoli gates and diagonal gates. We prove the following:

Theorem 1. *It's possible to delegate C+P circuits non-interactively and securely in the quantum random oracle model, and the client requires* $O(\eta N_q + N_q^2)$ *quantum* CNOT *gates as well as polynomial classical computation, where* N_q *is the number of qubits in the input and* η *is the security parameter.*

We will give a more formal statement in Section 6. The client's quantum circuit size can in fact be bounded by $O(\kappa N_q)$ where κ is the key length of the cryptographic primitives we use. Our current proof of security requires setting $\kappa = \eta + 4N_q$ where η is the actual security parameter. However, we conjecture the same protocol can be proven secure for $\kappa = O(\eta)$, leading to the following conjecture:

Conjecture 1. It's possible to delegate C+P circuits non-interactively and securely in the quantum random oracle model, using the same protocol as Theorem 1, and the client side quantum resources are $O(\eta N_q)$ CNOT gates, where N_q is the number of qubits in the input and η is the security parameter.

We argue that our protocol is important for three reasons: (1)The client only needs small quantum resources. Here we say "small" to mean the quantum resources only depend on the key length and the input size, and are independent of the circuit size. (2)Its security can be proven in the quantum random oracle model, without assuming some trapdoor one-way function. Many protocols before, for example, [11][15] are based on classical FHE and therefore rely on some kinds of lattice cryptographic assumptions, for example, LWE assumption. Our protocol is based on the quantum random oracle (therefore based on hash functions in practice), and this provides an alternative, incomparable assumption on which we can base the security of quantum delegation. (3)Our protocol introduces some new ideas and different techniques, which may be useful in the study of other problems.

Our protocol can be applied to Shor's algorithm. The hardest part of Shor's algorithm is the Toffoli part applied on quantum states, so the client can use this protocol securely with the help of a remote quantum server.

Corollary 1. *It's possible to delegate Shor's algorithm on input of length n within one round of communication in the quantum random oracle model, where the client requires $O(\eta n + n^2)$ CNOT gates plus $\tilde{O}(n)$ quantum gates. Assuming Conjecture 1, the number of CNOT gates is $O(\eta n)$.*

If the client runs the factoring algorithm by itself, the quantum operations it needed will be $\omega(n^2)$, and the exact complexity depends on the multiplication methods.

The security proof for our protocol heavily uses the concept of KDM security, which was not previously studied in the quantum setting. We therefore also initiate a systematic study of KDM security in the quantum random oracle model. We point out that although there already exists classical KDM secure encryption scheme in the random oracle model[4], the security in the quantum random oracle model still needs an explicit proof. We complete its proof in this paper. Furthermore, we generalize KDM security to quantum KDM security, and construct a protocol for it in the quantum random oracle model.

1.2 Related Work

To delegate quantum computation, people raised the concepts of blind quantum computation[6] and quantum homomorphic encryption(QHE)[7]. These two concepts are a little different but closely related: in quantum homomorphic encryption, no interaction is allowed and the circuits to be evaluated are known by the server. While in blind quantum computation, interactions are usually allowed and the circuits are usually only known by the client.

The concept of blind quantum computation was first raised in [2]. And [6] gave a universal blind quantum computation protocol, based on measurement-based quantum computation(MBQC)[18]. What's more, secure assisted quantum computation based on quantum one-time pad(QOTP) technique was raised in [9], with which we can easily apply Clifford gates securely but T gates are hard to implement and require interactions.

Quantum homomorphic encryption is the homomorphic encryption for quantum circuits. Based on QOTP and classical FHE, [7] studied the quantum homomorphic encryption for circuits with low T gate complexity. Later [11] constructed a quantum homomorphic encryption scheme for polynomial size circuits. But it still requires some quantum computing ability on the client side to prepare the evaluation gadgets, and the size of gadgets is propotional to the number of T gates. Recently Mahadev constructed a protocol[15], which achieves fully quantum homomorphic encryption, and what makes this protocol amazing is that the client can be purely classical, which hugely reduces the burden on the client side.

Another viewpoint of these protocols is the computational assumptions needed. With interactions, we can do blind quantum computation for universal quantum circuits information theoretically(IT-) securely. But for non-interactive protocols, [24] gave a limit for IT-secure QHE, which implies IT-secure quantum FHE is impossible. But it's still possible to design protocols for some non-universal circuit families.[14] gave a protocol for IQP circuits, and [23] gave a protocol for circuit with logarithmic number of T gates.

On the other hand, [7][11][15] rely on classical FHE. The current constructions of classical FHE are all based on various kinds of lattice-based cryptosystems, and the most standard assumption is the Learning-With-Error(LWE) assumption.

Table 1 compares different protocols for quantum computation delegation.

Protocol	Circuit class	Client's quantum resources	Assumption
QOTP[9]	Clifford	$O(N_q)$ Pauli operations	-
[6]	All	$O(L)$ Rounds: Circuit Depth	-
[15]	All	$O(N_q)$ Pauli operations	FHE
[14]	IQP	$O(N_q)$	-
[23]	Clifford+small number of T gates	Exponential in the number of T gates	-
This paper	C+P	$O(\eta N_q)$(Conjectured) $O(\eta N_q + N_q^2)$(Proved) CNOT operations	Quantum ROM

Table 1. L is the number of gates in the circuits, N_q is the number of qubits in the input, η is the security parameter.

1.3 Techniques

A different encoding for hiding quantum states with classical keys In many previous protocols, the client hides a quantum state using "quantum one time pad": $\rho \to \mathsf{X}^a\mathsf{Z}^b(\rho)$, where a, b are two classical strings. After taking average on a, b, the encrypted state becomes a completely mixed state. In our protocol, we use the following mapping to hide quantum states, which maps one qubit in the plaintext to κ qubits in the ciphertext:

$$\mathsf{Et}_{k_0,k_1} : |0\rangle \to |k_0\rangle , |1\rangle \to |k_1\rangle$$

where k_0, k_1 are chosen uniformly at random in $\{0, 1\}^\kappa$ and distinct.

We can prove for all possible input states, if we apply this operator on each qubit, after taking average on all the possible keys, the final results will be exponentially close to the completely mixed state.

Reversible garbled circuits The main ingredient in our construction is "reversible garbled circuit". In the usual construction of Yao's garbled table, the server can feed the input keys into the garbled table, and get the output keys; then in the decoding phase, it uses an output mapping to map the keys to the result. This well-studied classical construction does not work for quantum states. Even if the original circuit is reversible, the evaluation of Yao's garbled circuit is not! To use it on quantum states, besides the original garbled table, we add another table from the output keys to the input keys. This makes the whole scheme reversible, which means we can use it on quantum states and the computation result won't be entangled with auxiliary qubits. For security, we remove the output mappings. In the context of delegation, these are kept by the client.

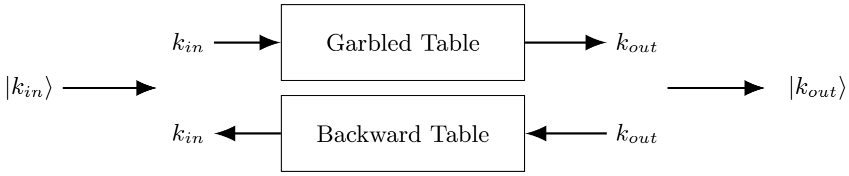

Fig. 2. Reversible garbled table

Note The proof of security of this scheme is subtle. The extra information included to allow the reversible computation introduces encryption cycles among the keys. We address the problem by studying key-dependent message security in the quantum setting. We show that a KDM-secure encryption scheme exists in the quantum random oracle model, and use this result to prove the security of our reversible garbled circuit construction.

Phase gates The reversible garbled circuit allows evaluating Toffoli circuits. To handle phase gates, instead of applying $|k_{in}\rangle \to |k_{out}\rangle$, we can make the garbled table implement the following transformation(where m is chosen randomly):

$$|k_0\rangle \to |k_0\rangle |m\rangle , |k_1\rangle \to |k_1\rangle |m+1\rangle \qquad (1)$$

Then the server can apply a "qudit Z gate" $\sum_i \omega_n^i |i\rangle \langle i|$ (define $\omega_n := e^{\mathrm{i}\pi/n}$) on the second register, where $i \in \mathbb{Z}_n$ goes through all the integers in $\mathbb{Z}_n$.(This operation can be done efficiently.) This will give us:

$$|k_0\rangle \to \omega_n^m |k_0\rangle |m\rangle , |k_1\rangle \to \omega_n^{m+1} |k_1\rangle |m+1\rangle$$

Then it applies (1) again to erase the second register. After removing the global phase the result is the same as the output of applying a phase gate $R_Z(\frac{\pi}{n}) = |0\rangle \langle 0| + \omega_n |1\rangle \langle 1|$.

1.4 Organisation

This paper is organized as follows. Section 2 contains some background for this paper. In Section 3 we discuss the encoding scheme. In Section 4 we give our construction of the quantum computation delegation protocol for C+P circuits. In Section 5 we prove the security of classical KDM secure scheme in the quantum random oracle model, as the preparation for the security proof of the main protocol. Then in Section 6 we discuss the security of our protocol. Section 7.1 turns this delegation scheme to a fully blind protocol, and Section 7.2 shows how to use our protocol on Shor's algorithm. Section 8 generalizes KDM security to quantum settings, constructs a quantum KDM secure protocol and proves its security. Then we discuss the open questions and complete this paper.

2 Definitions and Preliminaries

2.1 Basics of Quantum Computation

In this section we give a simple introduction for quantum computing, and clarify some notations in this paper. For more detailed explanations, we refer to [16].

In quantum computing, a pure state is described by a unit vector in a Hilbert space. A qubit, or a quantum bit, in a pure state, can be described by a vector $|\varphi\rangle \in \mathbb{C}^2$. The symbols $|\cdot\rangle$ and $\langle\cdot|$ are called Dirac symbols. A qudit is described by a vector $|\varphi\rangle \in \mathbb{C}^d$.

But a quantum system isn't necessarily in a pure state. When the quantum system is open, we need to consider mixed states. To describe both pure and mixed states, the state of a qubit is described by a density matrix in $\mathbb{C}^{2\times 2}$. A density matrix is a trace-one positive semidefinite complex matrix. The density matrix that corresponds to pure state $|\varphi\rangle$ is $|\varphi\rangle\langle\varphi|$, and we abbreviate it as φ.

For an n-qubit state, its density matrix is in $\mathbb{C}^{2^n\times 2^n}$. The space of density operators in system $\mathcal{S}$ is denoted as $\mathbb{D}(\mathcal{S})$. Note that we use $\mathbb{E}$ for the notation of the expectation value.

A quantum operation on pure states can be described by a unitary transform $|\varphi\rangle \to U|\varphi\rangle$. And an operation on mixed states can be described by a superoperator $\rho \to \mathcal{E}(\rho) = \mathrm{tr}_R(U(\rho\otimes|0\rangle\langle 0|)U^\dagger))$. We use calligraphic characters like $\mathcal{D}$, $\mathcal{E}$ to denote superoperators, and use the normal characters like U, D to denote unitary transforms. We also use Sans-serif font like $\mathsf{X}, \mathsf{Z}, \mathsf{Et}$ to denote quantum operations: When they are used as $\mathsf{Et}|\varphi\rangle$ they mean unitary operations(applied on Dirac symbols without parentheses), and when used as $\mathsf{Et}(\rho)$ they mean superoperators.

The quantum gates include X, Y, Z, CNOT, H, T, $\mathsf{Toffoli}$ and so on. What's more, denote $R_Z(\theta) = |0\rangle\langle 0| + e^{\mathrm{i}\theta}|1\rangle\langle 1|$, where i is the imaginary unit. Denote $\omega_n = e^{\mathrm{i}\pi/n}$, we can write $R_Z(k\pi/n) = |0\rangle\langle 0| + \omega_n^k|1\rangle\langle 1|$. Since the i will be used as the symbol for indexes and "inputs", we avoid using $e^{\mathrm{i}\pi/n}$ in this paper, and use ω_n instead.

The trace distance of two quantum states is defined as $\Delta(\rho,\sigma) = \frac{1}{2}|\rho-\sigma|_{\mathrm{tr}}$, where $|\cdot|_{\mathrm{tr}}$ is the trace norm.

2.2 Encryption with Quantum Adversaries

A quantum symmetric key encryption scheme contains three mappings: $\mathsf{KeyGen}(1^\kappa) \to sk$, $\mathsf{Enc}_{sk} : \mathbb{D}(\mathcal{M}) \to \mathbb{D}(\mathcal{C})$, $\mathsf{Dec}_{sk} : \mathbb{D}(\mathcal{C}) \to \mathbb{D}(\mathcal{M})$.[17]

In this paper, we need to use the symmetric key encryption scheme with key tags, which contains four mappings: KeyGen, Enc, Dec, Ver. The scheme has a key verification procedure $\mathsf{Ver} : \mathcal{K} \times \mathbb{D}(\mathcal{C}) \to \{\perp, 1\}$.

A quantum symmetric key encryption scheme with key tags is correct if:

1. $\forall \rho \in \mathbb{D}(\mathcal{R} \otimes \mathcal{S})$, $\mathbb{E}_{sk \leftarrow \mathsf{KeyGen}(1^\kappa)} |(\mathsf{I} \otimes \mathsf{Dec}_{sk})((\mathsf{I} \otimes \mathsf{Enc}_{sk})(\rho)) - \rho|_{\mathrm{tr}} = \mathsf{negl}(\kappa)$
2. $\forall \rho \in \mathbb{D}(\mathcal{R} \otimes \mathcal{S})$, $\Pr_{sk \leftarrow \mathsf{KeyGen}(1^\kappa)}(\mathsf{Ver}(sk, (\mathsf{I} \otimes \mathsf{Enc}_{sk})(\rho)) = \perp) = \mathsf{negl}(\kappa)$, and $\Pr_{sk \leftarrow \mathsf{KeyGen}(1^\kappa), r \leftarrow \mathsf{KeyGen}(1^\kappa)}(\mathsf{Ver}(r, (\mathsf{I} \otimes \mathsf{Enc}_{sk})(\rho)) = 1) = \mathsf{negl}(\kappa)$

Here the encryption and decryption are all on system S, and R is the reference system.

Sometimes we also need to encrypt the messages with multiple keys, and require that (informally) an adversary can only get the message if it knows all the keys. In symmetric multi-key encryption scheme with key tags, $\mathsf{KeyGen}(1^\kappa)$ is the same as the symmetric single-key scheme, $\mathsf{Enc}_{k_1, k_2, \cdots k_i}$ encrypts a message under keys $K = (k_1, k_2, \cdots k_i)$, $\mathsf{Dec}_{k_1, k_2, \cdots k_i}$ decrypts a ciphertext given all the keys $k_1, k_2, \cdots k_i$, and $\mathsf{Ver}(k, i, c) \to \{\perp, 1\}$ verifies whether k is the i-th key used in the encryption of c.

The next problem is to define "secure" formally. The concept of indistinguishability under chosen plaintext attack (IND-CPA) was introduced in [3][13]. Let's first review the security definitions in the classical case.

Definition 1. *For a symmetric key encryption scheme, consider the following game, called "IND-CPA game", between a challenger and an adversary $\mathscr{A}$:*

1. *The challenger runs* $\mathsf{KeyGen}(1^\kappa) \to sk$ *and samples* $b \leftarrow_r \{0, 1\}$.
2. *The adversary gets the following classical oracle, whose input space is* $\mathcal{M}$:
   (a) *The adversary chooses* $m \in \mathcal{M}$, *and sends it into the oracle.*
   (b) *If* $b = 1$, *the oracle outputs* $\mathsf{Enc}(m)$. *If* $b = 0$, *it outputs* $\mathsf{Enc}(0^{|m|})$.
3. *The adversary tries to guess* b *with some distinguisher* $\mathcal{D}$. *Denote the guessing result as* b'.

The distinguishing advantage is defined by $\mathsf{Adv}^{IND-CPA}(\mathscr{A}, \kappa) = |\Pr(b' = 1|b = 1) - \Pr(b' = 1|b = 0)|$.

And we call it an one-shot IND-CPA game if the adversary can only query the oracle once. Similarly we can define the distinguishing advantage $\mathsf{Adv}^{IND-CPA-oneshot}(\mathscr{A}, \kappa) = |\Pr(b' = 1|b = 1) - \Pr(b' = 1|b = 0)|$

Definition 2. *We say a protocol is IND-CPA secure against quantum adversaries if for any BQP adversary* $\mathscr{A}$ *which can run quantum circuits as the distinguisher but can only make classical encryption queries, there exists a negligible function* negl *such that* $\mathsf{Adv}^{IND-CPA}(\mathscr{A}, \kappa) = \mathsf{negl}(\kappa)$. *And we call it one-shot IND-CPA secure against quantum adversaries if* $\mathsf{Adv}^{IND-CPA-oneshot}(\mathscr{A}, \kappa) = \mathsf{negl}(\kappa)$.

Note that the "IND-CPA security against quantum adversaries" characterizes the security of a protocol against an adversary who has the quantum computing ability in the distinguishing phase but can only run the protocol classically.

For quantum cryptographic schemes, we use the formulation in [7].

Definition 3. *For a symmetric key encryption scheme, consider the following game, called "qIND-CPA game", between a challenger and an adversary $\mathscr{A}$:*

1. *The challenger runs* $\mathsf{KeyGen}(1^\kappa) \to sk$ *and samples* $b \leftarrow_r \{0,1\}$.
2. *The adversary gets the following oracle, whose input space is* $\mathcal{D}(\mathcal{M})$*:*
   (a) *The adversary chooses* $\rho \in \mathbb{D}(\mathcal{M} \otimes \mathcal{R})$*. The adversary sends system* $\mathcal{M}$ *to the oracle, and keeps* $\mathcal{R}$ *as the reference system.*
   (b) *If* $b = 1$*, the oracle applies* Enc *on* $\mathcal{M}$ *and sends it to the adversary. The adversary will hold the state* $(\mathsf{Enc} \otimes \mathsf{I})(\rho)$*. If* $b = 0$*, the oracle encrypts* $0^{|m|}$ *and the adversary gets* $(\mathsf{Enc} \otimes \mathsf{I})(0^{|m|} \otimes \rho_R)$*, where* ρ_R *is the density operator of subsystem* $\mathcal{R}$*.*
3. *The adversary tries to guess* b *with some distinguisher* $\mathcal{D}$*. Denote the guessing output as* b'*.*

The distinguishing advantage is defined by $\mathsf{Adv}^{qIND-CPA}(\mathscr{A}, \kappa) = |\Pr(b' = 1|b = 1) - \Pr(b' = 1|b = 0)|$.

And we call it an one-shot qIND-CPA game if the adversary can only query the oracle once. Similarly we can define the distinguishing advantage $\mathsf{Adv}^{qIND-CPA-oneshot}(\mathscr{A}, \kappa) = |\Pr(b' = 1|b = 1) - \Pr(b' = 1|b = 0)|$.

Definition 4. *A protocol is qIND-CPA secure if for any BQP adversary* $\mathscr{A}$*, there exists a negligible function* negl *such that* $\mathsf{Adv}^{qIND-CPA}(\mathscr{A}, \kappa) = \mathsf{negl}(\kappa)$.

What's more, we call it one-shot qIND-CPA secure if for any BQP adversary $\mathscr{A}$*, there exists a negligible function* negl *such that* $\mathsf{Adv}^{qIND-CPA-oneshot}(\mathscr{A}, \kappa) = \mathsf{negl}(\kappa)$.

In the definition of qIND-CPA security, the adversary can query the encryption oracle with quantum states, and it can also run a quantum distinguisher.

Key Dependent Message security In the definitions above the plaintexts do not depend on the secret keys. There is another type of security called "key-dependent message (KDM) security", where the adversary can get encryptions of the secret keys themselves. We will need to study this type of security in the proof of our main theorem, but we defer the definitions and further discussions to Section 5.

2.3 Delegation of Quantum Computation, and Related Problems

There are three similar concepts: delegation of quantum computation, quantum homomorphic encryption[7] and blind quantum computation[2][6].

The differences of these three concepts are whether the interaction is allowed, and which party knows the circuit. The delegation of quantum computation and blind quantum computation protocols are interactive. For quantum homomorphic encryption, the interaction is not allowed. If we focus on non-interactive protocols, their difference is which party knows the circuit: in blind quantum computation, the circuit is only known by the client but not the server; in homomorphic encryption, the circuit is known by the server but not necessarily known by the client. In our paper, we use "delegation of quantum computation" to mean that the circuit is known by both parties but the input is kept secret.

A non-interactive quantum computation delegation protocol BQC on circuit family $\mathcal{F} = \{F_n\}$ contains 4 mappings:

$\mathsf{BQC.KeyGen}(1^\kappa, 1^N, 1^L) \rightarrow (sk)$: The key generation algorithm takes the key length κ, input length N and circuit length L and returns the secret key.

$\mathsf{BQC.Enc}_{sk}^C : \mathbb{D}(\mathcal{M}) \rightarrow \mathbb{D}(\mathcal{C})$. Given the encryption key and the public circuit in $\mathcal{F} = \cup\{F_n\}$, this algorithm maps inputs to ciphertexts.

$\mathsf{BQC.Eval}^C : \mathbb{D}(\mathcal{C}) \rightarrow \mathbb{D}(\mathcal{C}')$. This algorithm maps ciphertexts to some other ciphertexts, following the instructions which may be contained in $\mathcal{C}$.

$\mathsf{BQC.Dec}_{sk} : \mathbb{D}(\mathcal{C}') \rightarrow \mathbb{D}(\mathcal{M}')$. This algorithm decrypts the ciphertexts and stores the outputs in $\mathcal{M}$.

Here we put N, L into the KeyGen algorithm, which are needed in our protocol. We put C on the superscript to mean the circuit is known by both parties.

Definition 5. *The security (IND-CPA, qIND-CPA, etc) of the non-interactive delegation of computation protocol is defined to be the security of its encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc})$.

2.4 Quantum Random Oracle Model

A classical random oracle is an oracle of a random function $\mathcal{H} : \{0,1\}^\kappa \rightarrow \{0,1\}^\kappa$ which all parties can query with classical inputs. It returns independent random value for different inputs, and returns fixed value for the same input. In practice, a random oracle is usually replaced by a hash function.

A quantum random oracle allows the users to query it with quantum states: the users can apply the map $\mathcal{H} : |a\rangle |b\rangle \rightarrow |a\rangle |\mathcal{H}(a) \oplus b\rangle$ on its state. The quantum random oracle was raised in [5]. It becomes the security proof model for many post-quantum cryptographic scheme[21]. On the other hand, the application of the quantum random oracle in quantum cryptographic problems is not very common, and as far as we know, our work is the first application of it in the delegation-stype problems.

The security definitions in the quantum random oracle model are the same as Definitions 2 and 4. Here we assume the adversary can only make polynomial number of random oracle queries, but the queries can be quantum states. Then by the "Random Oracle Methodology" we can conjecture the protocol is also secure in the standard model, when the random oracle is replaced by a hash function in practice. As with proofs in the classical random oracle model, interpreting these security claims is subtle, since there exist protocols that are secure in the random oracle model but insecure in any concrete initialization of hash function.[8]

This paper focuses on the quantum cryptographic protocols in the quantum random oracle model. As far as we know, the assumption of a quantum random oracle is incomparable to any trapdoor assumption. We do not know any construction of public key encryption based on solely quantum random oracle. What's more, in our proof, the random oracle doesn't need to be "programmable"[12].

2.5 Garbled Table

We make a simple introduction of Yao's garbled table [22] here. The garbled table construction will be the foundation of our protocol.

Garbled table is a powerful technique for the randomized encoding of functions. When constructing the garbled circuit of some circuit C, the client picks two keys for each wire, and denotes them as k_b^w, where $b \in \{0,1\}$, and w is the index of the wire.

The garbled table is based on a symmetric key encryption scheme with key tags. For gate g, suppose its input wires are w_1, w_2, and the output wire is v. The client constructs the following table:

$$\mathsf{Enc}_{k_0^{w_1},k_0^{w_2}}(k^v_{g(0,0)}) \tag{2}$$

$$\mathsf{Enc}_{k_0^{w_1},k_1^{w_2}}(k^v_{g(0,1)}) \tag{3}$$

$$\mathsf{Enc}_{k_1^{w_1},k_0^{w_2}}(k^v_{g(1,0)}) \tag{4}$$

$$\mathsf{Enc}_{k_1^{w_1},k_1^{w_2}}(k^v_{g(1,1)}) \tag{5}$$

And it picks a random permutation in S_4 to shuffle them.

If the server is given the garbled table for some gate, and given a pair of input keys, it can evaluate the output keys: it can try each row in the garbled table and see whether the given keys pass the verification. If they pass, use them to decrypt this row and get the output keys.

By providing the input keys and the garbled table for each gate in the circuit, the server can evaluate the output keys for the whole circuit. And in the randomized encoding problem the client also provides the mapping from the output keys to the corresponding values on some wires: $k_b^w \to b$, for some set of ws. The server can know the output values on these revealed wires, but the values on other wires are hidden. This construction has wide applications in classical world, for example, it allows an NC^0 client to delegate the evaluation of a circuit to the server.

3 The Encoding For Hiding Quantum States With Classical Keys

Let's first discuss the encoding operator, Et, to "hide" the quantum states. For each qubit in the input, the client picks two random different keys $k_0, k_1 \in \{0,1\}^\kappa$ and encodes the input qubit with the following operator:

$$\mathsf{Et}_{k_0,k_1} : |0\rangle \to |k_0\rangle\,, |1\rangle \to |k_1\rangle$$

The dimensions of two sides are not the same, but we can add some auxiliary qubits on the left side. As long as k_0, k_1 are distinct, this operator is unitary.

For pure quantum state $|\varphi\rangle = \sum \alpha_{i_1 i_2 \cdots i_N} |i_1 i_2 \cdots i_N\rangle$, given key set $K = \{k_i^n\}$, where $n \in [N]$, $i \in \{0,1\}$, if we apply this operator on each qubit, using keys $\{k_0^n, k_1^n\}$ for the n-th qubit, we get:

$$\mathsf{Et}_K |\varphi\rangle = \sum \alpha_{i_1 i_2 \cdots i_n} |k_{i_1}^{(1)} k_{i_2}^{(2)} \cdots k_{i_n}^{(N)}\rangle$$

The following lemma shows that if the keys are long enough, chosen randomly and kept secret, this encoding is statistically secure, in other words, the mixed state after we take average on all the possible keys, is close to the completely mixed state with exponentially small distance:

Lemma 1. *Suppose $\rho \in \mathbb{D}(\mathcal{S} \otimes \mathcal{R})$, $\mathcal{S} = (\mathbb{C}^2)^{\otimes N}$. Suppose we apply the* Et *operation on system $\mathcal{S}$ with key length κ, after taking average on all the valid keys, we get*

$$\sigma = \frac{1}{(2^\kappa(2^\kappa - 1))^N} \sum_{\forall n \in [N], k_0^n, k_1^n \in \{0,1\}^\kappa, k_0^n \neq k_1^n} (\mathsf{Et}^{\mathcal{S}}_{\{k_i^n\}} \otimes \mathsf{I})(\rho)$$

then we have $\Delta(\sigma, (\frac{1}{2^{\kappa N}}\mathsf{I}) \otimes \mathrm{tr}_{\mathcal{S}}(\rho)) \leq (\frac{1}{2})^{\kappa-4}N$

Thus such an encoding keeps the input secure against unbounded adversaries. We put the detailed proof in the full version of this paper.

Since Et is a unitary mapping, given K and $\mathsf{Et}_K(\rho)$, we can apply the inverse of Et and get ρ: $\mathsf{Et}_K^{-1}(\mathsf{Et}_K(\rho)) = \rho$. Note that when applying Et we enlarge the space by appending auxiliary qubits, and when applying Et^{-1} we remove these auxiliary qubits.

Fact 1 Et *can be implemented with only* CNOT *operations.*

Proof. First implement mapping $|0\rangle \to |0^\kappa\rangle, |1\rangle \to |k_0 \oplus k_1\rangle$. This can be done by CNOT the input into the places where $k_0 \oplus k_1$ has bit value 1. Then apply X gates on the places where k_0 has bit value 1. This will xor k_0 into these registers and complete the mapping $|0\rangle \to |k_0\rangle, |1\rangle \to |k_1\rangle$.

The quantum computation delegation protocol that we will discuss in the next section will use this encoding.

4 A Quantum Computation Delegation Protocol for C+P Circuits

In this section, we use Et encoding and a new technique called "reversible garbled circuit" to design a quantum computation delegation protocol.

4.1 C+P Circuits and the Relation to Toffoli Depth

[20] defined "almost classical" circuits. Here we rename it to "C+P" circuits, abbreviating "classical plus phase".

Definition 6 ([20]). *C+P is the family of quantum circuits which are composed of Toffoli gates and diagonal gates.*

We can prove it's possible to decompose this type of circuits into simpler gates. We put the proof in the full version of this paper.

Proposition 1. *Any C+P circuit can be decomposed to Toffoli gates and single qubit phase gates. Furthermore, it can be approximated by Toffoli gates and single qubit phase gates of the form $R_Z(\frac{\pi}{n}) = |0\rangle\langle 0| + \omega_n |1\rangle\langle 1|, n \in \mathbb{N}_+$, where ω_n is the nth root of unity. To approximate a circuit of length L of Toffoli gates and single qubit phase gates to precision ϵ, we only need Toffoli gates and phase gates in the form of $R_Z(\frac{\pi}{2^d}), d \in [D]$, where $D = \Theta(\log \frac{L}{\epsilon})$.*

We consider D as a fixed value in this paper. Since ϵ depends exponentially on D, a small D in practice should be enough and it will at most add a logarithmic term in the complexity.

{C+P, H} is a complete basis for quantum circuits. Our work implies a delegation scheme whose round complexity equals the H-depth of a given circuit. Previous works on quantum computation delegation generally focused on {Clifford, T} basis. (The exception is [14], which works for IQP circuits.) With the exception of Mahadev's FHE-based scheme[15], their complexity of client side quantum gates increases with the circuit's T-depth.

As far as we know, there is no general way to transform a Toffoli circuit into the {Clifford, T} basis such that its T depth is smaller than the Toffoli depth of the original ciruit, without blowing up the circuit width exponentially. We formalize this statement as a conjecture:

Conjecture 2. For any polynomial time algorithm that transforms Toffoli circuits into the {Clifford, T} basis, there exists a sequence of inputs with increasing Toffoli depths for which the algorithm's outputs have T depth $\Omega(d)$, where d denotes the Toffoli depths of the original circuits.

Working with the {C+P, H} basis allows us to design efficient protocols for delegating Shor's algorithm (which has low H-depth). Previously, this was only possible using FHE-based schemes.

4.2 Protocol Construction

We now describe our protocol that supports our main results. This protocol gets a public description of a C+P circuit as well as a secret quantum state.

The idea comes from Yao's Garbled Circuit construction. We have discussed the construction in section 2.5. The garbled circuit construction is commonly used for randomized encodings of classical circuits, but it's not applicable to quantum circuits. In this paper we will show how to do the reversible garbling for C+P circuits. Let's first discuss the ideas briefly.

One big difference of classical operations and quantum operations is in quantum world, the operations have to be reversible. Firstly, we will consider the garbling of Toffoli gates. In classical world, the garbled tables can contain non-reversible gates, for example, AND gate, OR gate. But in quantum world, we have to start with the Toffoli gate, which is reversible, and contains 3 input wires and 3 output wires.

However, even if the underlying circuit is reversible, if we try to use the classical garbled table construction on a quantum circuit, the garbled circuits is still not reversible, and it's not possible to use it to implement the quantum operations. Note that we need two levels of reversibility here: the circuit to be garbled needs to be reversible, and the garbled circuit itself has to be reversible too, even if it calls the random oracle as a black box.

Thus we propose a new garbling technique, which is a reversible garbling of reversible circuits: when constructing the garbled tables, instead of just creating one table for each gate, the client can construct two tables, in one table it encrypts the output keys with the input keys, and in the other table it encrypts the input keys with the output keys! This construction will make the garbled circuit reversible: we will show, the garbled circuit evaluation mapping can be applied on quantum states unitarily.

But another problem arises: If we simply replace the garbled circuit in the randomized encoding problem with "reversible garbled circuit", it's not secure any more. But it turns out, if we remove the output mapping, it becomes secure again, under some reasonable assumptions. And that gives us a delegation protocol.

The full protocol is specified in Protocol 1. Below we give more details.

Reversible garbling of Toffoli gates First recall that in the classical garbled circuit, the evaluation operation on each garbled gate takes the input keys, decrypts the table and computes the corresponding output keys:

$$k_{in} \to k_{out}$$

This mapping is classical, and there is a standard way to transform a classical circuit to a quantum circuit, by introducing auxiliary output registers, and keeping the input:

$$U : |k_{in}\rangle |c\rangle \xrightarrow{\text{garbled gate}} |k_{in}\rangle |k_{out} \oplus c\rangle \tag{6}$$

We use the second register as the output register, and c is its original value. This mapping computes the output keys from the garbled table and xors them to the second register.

This mapping is unitary, and we can also put superpositions on the left-hand side of (6). However, when it is used directly on quantum states, the inputs and outputs will be entangled together. Explicitly, for a specific Toffoli gate, we use $k_u^{w1}, k_v^{w2}, k_w^{w3}$ to denote the keys of the input wires $w1, w2, w3$ which correspond to the input (u, v, w); for the output part the keys are $k_u^{v1}, k_v^{v2}, k_w^{v3}$. If we apply (6) directly, we get:

$$\begin{aligned} U : &|k_u^{w1}\rangle |k_v^{w2}\rangle |k_w^{w3}\rangle |c_1\rangle |c_2\rangle |c_3\rangle \\ &\to |k_u^{w1}\rangle |k_v^{w2}\rangle |k_w^{w3}\rangle |k_u^{v1} \oplus c_1\rangle |k_v^{v2} \oplus c_2\rangle |k_{w\oplus uv}^{v3} \oplus c_3\rangle \end{aligned}$$

But what we need is the following mapping:

$$U : |k_u^{w1}\rangle |k_v^{w2}\rangle |k_w^{w3}\rangle \to |k_u^{v1}\rangle |k_v^{v2}\rangle |k_{w\oplus uv}^{v3}\rangle \tag{7}$$

Which means, we need to disentangle and erase the input registers from the output registers. Note that, again, both sides should be understood as superpositions of different keys. And recall that for each Toffoli gate there are eight possible combinations of input keys, and this mapping should work for all the eight combinations.

To erase the input from the output, we can use two mappings: $|k_{in}\rangle |0\rangle \rightarrow |k_{in}\rangle |k_{out}\rangle$ and $|k_{in}\rangle |k_{out}\rangle \rightarrow |0\rangle |k_{out}\rangle$. Both operations have the same form as equation (6). (For the second step, we could view the k_{out} as the input, k_{in} as c, and get $|k_{out}\rangle |k_{in} \oplus k_{in}\rangle$) So we can use two garbled tables for this "reversible garbled table"!

Assume CL is some multiple key encryption scheme with key tags. The client puts the encryption outputs $\mathsf{CL.Enc}_{k_{in}}(k_{out})$ into a table(there are eight rows in this table), and shuffles them randomly; this is the forward table. And it puts the encryption outputs $\mathsf{CL.Enc}_{k_{out}}(k_{in})$ into a table and shuffles to get the backward table. This construction will allow the server to implement (7), even on superpositions of input keys.

We note that we do not need to consider the detailed operations for decrypting each garbled table, and the existence of such operations comes from quantize the classical mapping as (6).

For the encoding of the inputs, recall that in the usual garble table construction, the client encrypts each bit in the inputs with the mapping:

$$0 \rightarrow k_0, 1 \rightarrow k_1 \tag{8}$$

To make it quantum, instead of replacing the classical bits with the corresponding keys, the client uses Et operator to hide the inputs. And we notice that (8) is a special case of Et where the input is classical.

Phase gates Now the protocol works for Toffoli gates. But what if there are phase gates?

From Proposition 1, we only need to consider the single qubit phase gates in the form of $R_Z(\frac{\pi}{n}), n \in \mathbb{Z}_+$. Suppose we want to implement such a gate on some wire, where the keys are k_0, k_1, corresponding to values 0 and 1, as discussed in the last subsection.

To implement $R_Z(\frac{\pi}{n})$, the client first picks a random integer $m \in \mathbb{Z}_n$. What it is going to do is to create a table of two rows, put $\mathsf{CL.Enc}_{k_0}(m)$ and $\mathsf{CL.Enc}_{k_1}(m+1)$ into the table and shuffle it. When the server needs to evaluate $R_Z(\frac{\pi}{n})$, it will first decrypt the garbled table and write the output on an auxiliary register $|0\rangle$. So it can implement the following transformation:

$$|k_0\rangle \rightarrow |k_0\rangle |m\rangle , |k_1\rangle \rightarrow |k_1\rangle |m+1\rangle \tag{9}$$

This step is similar to implementing equation (6).

Then it applies a "qudit Z gate" $\sum_i \omega_n^i |i\rangle \langle i|$ on the second register, where $i \in \mathbb{Z}_n$ goes through all the integers in $\mathbb{Z}_n$.(This operation can be done efficiently.) This will give us:

$$|k_0\rangle \rightarrow \omega_n^m |k_0\rangle |m\rangle , |k_1\rangle \rightarrow \omega_n^{m+1} |k_1\rangle |m+1\rangle$$

Then it applies (9) again to erase the second register. After removing the global phase the result is the same as the output of applying a phase gate $R_Z(\frac{\pi}{n}) = |0\rangle\langle 0| + \omega_n |1\rangle\langle 1|$.

What's more, since m is chosen randomly the garbled gate won't reveal the keys. (This fact is contained in the security proof.)

4.3 Protocol Design

In this section we formalize this garbled circuit based quantum computation delegation protocol. Let's call it GBC.

We index the wires in the circuit as follows: If two wires are separated by a single qubit phase gate, we consider them as the same wire; otherwise (separated by a Toffoli gate, or disjoint), they are different wires. Suppose we have already transformed the circuit using Fact 1 so that there is no controlled phase gate. For a circuit with N input bits and L gates, the number of wires is at most $N + 3L$.

Protocol 1 *The protocol* GBC*, with* CL *being the underlying classical encryption scheme, for a circuit C which is composed of Toffoli gates and phase gates in the form of $R_Z(\frac{\pi}{n})$, is defined as:*

Key Generation $\mathsf{GBC.KeyGen}(1^\kappa, 1^N, 1^L)$*: Sample keys $K = (k_b^l)$,*
$k_b^l \leftarrow \mathsf{CL.KeyGen}(1^\kappa)$*, $b \in \{0,1\}, l \in [N+3L]$.*

Encryption $\mathsf{GBC.Enc}_K^C(\rho)$*: Output $(\mathsf{Et}_{K_{in}}(\rho), \mathsf{TAB}_{\mathsf{CL}}^C(K))$. (Note that with the reference system, the first part is $(\mathsf{I} \otimes \mathsf{Et}_{K_{in}})(\rho_{RS})$.)*

Evaluation $\mathsf{GBC.Eval}^C(c)$*, where $c = (\rho_q, tabs)$: Output $\mathsf{EvalTAB}_{\mathsf{CL}}^C(\rho_q, tabs)$*

Decryption $\mathsf{GBC.Dec}_K(\sigma)$*: Suppose the output keys in K are K_{out}. Apply the map $\mathsf{Et}_{K_{out}}^{-1}(\cdot)$ on σ and return the result.*

$\mathsf{TAB}_{\mathsf{CL}}^C(K)$ and $\mathsf{EvalTAB}_{\mathsf{CL}}^C(\rho_q, tabs)$ appeared in this protocol are defined as follows:

Protocol 2 $\mathsf{TAB}_{\mathsf{CL}}^C(K)$*, where K is the set of keys:*

Suppose circuit C is composed of gates $(g_i)_{i=1}^L$. This algorithm returns $(tab_{g_i})_{i=1}^L$, where tab_g is defined as follows:

1. *If g is a Toffoli gate: Suppose g has controlled input wires $w1, w2$ and target wire $w3$, and the corresponding output wires are $v1, v2, v3$. Suppose the corresponding keys in K are $\{k_b^w\}, w \in \{w1, w2, w3, v1, v2, v3\}, b \in \{0,1\}$:*
Create $table1$ as follows: For each triple $u, v, w \in \{0,1\}^3$, add the following as a row:
$$\mathsf{CL.Enc}_{k_u^{w1}, k_v^{w2}, k_w^{w3}}(k_u^{v_1} || k_v^{v_2} || k_{w\oplus uv}^{v_3})$$
and pick a random permutation in S_8 to shuffle this table.
Create $table2$ as follows: For each triple $u, v, w \in \{0,1\}^3$, add the following as a row:
$$\mathsf{CL.Enc}_{k_u^{v_1}, k_v^{v_2}, k_{w\oplus uv}^{v_3}}(k_u^{w1} || k_v^{w2} || k_w^{w3})$$
and pick another random permutation in S_8 to shuffle this table.
Return $(table1, table2)$

2. *If g is a phase gate, Suppose g is a phase gate $R_Z(\frac{\pi}{n})$ on wire w:*
   Sample $m_0 \leftarrow_r \mathbb{Z}_n$, $m_1 = m_0 + 1$. Create table1 as follows: For each $u \in \{0,1\}$, add the following as a row:

$$\mathsf{CL.Enc}_{k_u^w}(m_u)$$

   and pick a random permutation in S_2 to shuffle this table.
   Return table1.

Protocol 3 $\mathsf{EvalTAB}^C_{\mathsf{CL}}(\rho, tab)$*:*

Suppose circuit C is composed of gates $(g_i)_{i=1}^L$. For each gate g in C, whose corresponding garbled gate is tab_g in tab:

If g is a Toffoli gate, with input wires $w1, w2, w3$, output wires $v1, v2, v3$: Suppose $tab_g = (tab1, tab2)$, where $tab1$ is the table from input keys to output keys, and $tab2$ is from output keys to input keys. Suppose $\rho \in \mathbb{D}(\mathcal{S}_g \otimes \mathcal{S}')$, where $\mathcal{S}_g$ is the system that is currently storing the keys on the input wires of g, and $\mathcal{S}'$ is the remaining systems:

1. *Introduce three auxiliary registers and denote the system as S'_g. Use $tab1$ to apply the following mapping on S_g, as discussed in the section 4.2:*

$$|k_u^{w1}\rangle\,|k_v^{w2}\rangle\,|k_w^{w3}\rangle\,|0\rangle\,|0\rangle\,|0\rangle \to |k_u^{w1}\rangle\,|k_v^{w2}\rangle\,|k_w^{w3}\rangle\,|k_u^{v1}\rangle\,|k_v^{v2}\rangle\,|k_{w\oplus uv}^{v3}\rangle$$

2. *Use $tab2$ to apply the following mapping on $\mathcal{S}_g \otimes \mathcal{S}'_g$, as discussed in the section 4.2:*

$$|k_u^{w1}\rangle\,|k_v^{w2}\rangle\,|k_w^{w3}\rangle\,|k_u^{v1}\rangle\,|k_v^{v2}\rangle\,|k_{w\oplus uv}^{v3}\rangle \to |0\rangle\,|0\rangle\,|0\rangle\,|k_u^{v1}\rangle\,|k_v^{v2}\rangle\,|k_{w\oplus uv}^{v3}\rangle$$

3. *Remove system $\mathcal{S}_g$, rename $\mathcal{S}'_g$ as $\mathcal{S}_g$. Denote the final state as the new ρ.*

If g is a phase gate on wire w in the form of $R_Z(\frac{\pi}{n})$, : Suppose $\rho \in D(\mathcal{S}_g \otimes \mathcal{S}')$, where $\mathcal{S}_g$ is the system that stores the keys on the input wire of g, and S' is the remaining systems:

1. *Use tab_g to implement the mapping $|k^w\rangle\,|0\rangle \to |k^w\rangle\,|m\rangle$, where m is the decrypted output.*
2. *Apply $\sum_i \omega_n^i\,|i\rangle\,\langle i|$ on the system of m.*
3. *Use tab_g to implement the mapping $|k^w\rangle\,|m\rangle \to |k^w\rangle\,|0\rangle$.*

The following two theorems summarize its correctness and efficiency:

Theorem 2. *Protocol* GBC *is a correct non-interactive quantum computation delegation protocol for C+P circuits.*

Theorem 3. *In* GBC *protocol, the quantum resources required on the client side are $O(\kappa N_q)$* CNOT *gates, where κ stands for the key length used in the protocol, N_q is the size of quantum states in the input, which are independent of the size of the circuit.*

Here we use N_q instead of N because we want to consider the case where some part of the input is classical and some part of it is quantum. To make the protocol secure we may need to choose κ depending on N_q. This is discussed with more details in Section 6.

This means the quantum resources of this protocol are independent of the circuit to be evaluated! In practice the size of the circuit may be a large polynomial of the input size, and our protocol will not be affected by this.

4.4 Structure of the Security Proofs

The structure of the security proofs is as follows. First we study the key dependent message security in the quantum world, and design a protocol which we call the KDMP protocol. Note that this part is not about the garbling scheme.

Then for the garbling scheme, we first state Proposition 2, which is the IND-CPA security of our garbling scheme. And we state a lemma about the security of the garbling scheme, which is the Lemma 2. The proofs use a reduction to the security of the KDMP protocol. And the proofs are in the full version.

Then we prove the security of our garbling scheme (Theorem 6) from Proposition 2 and lemma 2. This part is given in the main content.

5 KDM Security of Classical Encryption against Quantum Attack

As we can see, in GBC protocol there are encryption cycles. So to make the protocol secure, for the underlying encryption scheme CL, the usual security definition is not enough and we need at least KDM security. In this section, we will first discuss the key dependent message security(KDM security) in quantum world, and give an encryption scheme KDMP that is KDM-secure against quantum adversaries. These results will be the foundation for the security proof of the GBC protocol.

In classical world, KDM security was discussed in several papers, for example, [4, 1]. [4] gave a classical KDM secure encryption scheme in the random oracle model, and [1] constructed KDM secure protocols in the standard model, based on some hard problems, for example, Learning-With-Error.

5.1 KDM Security in the Classical World

As a part of the preliminaries, we repeat the definition of the security game of the classical KDM security[4][1].

Definition 7. *The KDM-CPA game is defined similar to the IND-CPA game, except that (1)in the first step the challenger runs* $\mathsf{KeyGen}(1^\kappa)$ *for* N *times to generate* $\mathcal{K} = \{sk_i\}_{i\in[N]}$*,* N *is less than a polynomial of the security parameter.(2)the client is allowed to query the encryption oracle with a function* $f \in \mathcal{F}$*, a message* m*, and an index* i *of the keys, and the encryption oracle returns* $\mathsf{Enc}_{sk_i}(f(K, m))$ *or* $\mathsf{Enc}_{sk_i}(0^{|f(K,m)|})$*, depending on* b*. Note that the outputs of functions in* $\mathcal{F}$ *should be fixed-length, otherwise* $|f(K, m)|$ *is not well-defined.*

5.2 KDM Security in the Quantum World

The attack for the KDM security can be adaptive, which means, the adversary can make encryption queries after it receives some ciphertexts. But in our work we only need to consider the non-adaptive setting. What's more, we only need to consider the symmetric key case. To summarize, the game between the adversary and the challenger can be defined as:

Definition 8 (naSymKDM Game).
The symmetric key non-adaptive KDM game naSymKDM for function family $\mathcal{F}$ against a quantum adversary $\mathscr{A}$ in the quantum random oracle model with parameters (κ, L, T, q) is defined as follows.

1. *The challenger chooses bit $b \leftarrow_r \{0,1\}$ and samples $K = \{sk_i\}_{i=1}^{L}$, $sk_i \leftarrow \mathsf{KeyGen}(1^\kappa)$.*
2. *The adversary and the challenger do the following T times, non-adaptively, which means, the challenger will only send out the answers in step (b) after it has received all the queries:*
   (a) *The adversary picks index i, function $f \in \mathcal{F}$ and message $msg \in \{0,1\}^*$, and sends them to the challenger. The size of msg should be compatible with f.*
   (b) *If $b = 1$, the challenger gives $c = \mathsf{Enc}_{sk_i}(f(K, msg))$ to the adversary. If $b = 0$, the challenger gives $c = \mathsf{Enc}_{sk_i}(0^{|f(K,msg)|})$.*
3. *The adversary tries to guess b using distinguisher $\mathcal{D}$ and outputs b'. Here $\mathcal{D}$ is a quantum operation and can query the oracle with quantum states. Suppose $\mathcal{D}$ will query the random oracle for at most q times.*

f can also query the random oracle, and it only makes queries on classical states. What's more, the output of functions in $\mathcal{F}$ should have a fixed length, otherwise $|f(K, m)|$ will not be well-defined.

The guessing advantage is defined as $\mathsf{Adv}_{\mathcal{F}}^{naSymKDM}(\mathscr{A}_{(L,T,q)}, \kappa) = |\Pr(b' = 1|b = 1) - \Pr(b' = 1|b = 0)|$.

Definition 9. *A symmetric key encryption scheme is nonadaptive KDM secure for circuit family $\mathcal{F}$ against quantum adversaries in the quantum random oracle model if for any BQP adversary,*

$$\mathsf{Adv}_{\mathcal{F}}^{naSymKDM}(\mathscr{A}_{(L(\kappa),T(\kappa),q(\kappa))}, \kappa) = \mathsf{negl}(\kappa)$$

Where $L(\kappa), T(\kappa), q(\kappa)$ are polynomial functions that may depend on the adversary.

5.3 A KDM Secure Protocol in the Quantum Random Oracle Model

In the quantum random oracle model, we can give a construction of the classical KDM secure encryption scheme KDMP. Here "classical" means the encryption and decryption are purely classical. But the distinguisher may query the quantum random oracle in superposition.

Protocol 4 *We can construct a symmetric KDM secure encryption scheme* KDMP *that has key tags in the quantum random oracle model, where we denote the random oracle as* $\mathcal{H}$*:*

$\mathsf{KDMP.KeyGen}(1^\kappa)$*: Output* $sk \leftarrow_r \{0,1\}^\kappa$
$\mathsf{KDMP.Enc}_{sk}(m)$*:* $R_1, R_2 \leftarrow_r \{0,1\}^\kappa$*, output ciphertext* $c = (R_1, \mathcal{H}(sk||R_1) \oplus m)$ *and key tag* $(R_2, \mathcal{H}(sk||R_2))$
$\mathsf{KDMP.Dec}_{sk}(c)$*: Output* $\mathcal{H}(sk||c_1) \oplus c_2$*, where* c_1 *and* c_2 *are from* $c = (c_1, c_2)$*.*
$\mathsf{KDMP.Ver}(k, tag)$*: Suppose* $tag = (tag1, tag2)$*, output* 1 *if* $\mathcal{H}(k||tag1) = tag2$*, and* $\perp$ *otherwise.*

Since the execution of this protocol is classical, the correctness can be proved classically and is obvious. We refer to [4] here and write it out explicitly for convenience.

Theorem 4 (Correctness). KDMP *is a correct symmetric key encryption scheme with key tags in the quantum random oracle model.*

The security under classical random oracle model has been proven. But here we study the quantum random oracle, so although the protocol is almost the same, we still need a new proof.

Theorem 5 (Security). *Define* $\mathcal{F}[q']$ *as the set of classical functions that query the random oracle at most* q' *times. For any adversary which can query the random oracle quantumly at most q times, we have*

$$\mathsf{Adv}^{naSymKDM}_{\mathsf{KDMP},\mathcal{F}[q']}(\mathscr{A}_{(L,T,q)}, \kappa) \leq \mathsf{poly}(q, q', L, T)2^{-0.5\kappa}$$

where poly *is a fixed polynomial.*

We put the proof in the full version of this paper.

6 Security of GBC Protocol

In this section we discuss the security of protocol GBC. First we need to construct a classical encryption scheme CL as its underlying scheme. The construction is very similar to the KDMP scheme, except that this is multi-key and the KDMP scheme is single-key. We will use it as the underlying scheme of GBC.

6.1 Construction of the Underlying Classical Encryption Scheme

Protocol 5 *The underlying multi-key encryption scheme* CL *is defined as:*

$\mathsf{CL.KeyGen}(1^\kappa)$*: Output* $sk \leftarrow_r \{0,1\}^\kappa$
$\mathsf{CL.Enc}_{k_1,k_2,k_3}(m)$*:* $R_1, R_2, R_3, R_4, R_5, R_6 \leftarrow_r \{0,1\}^\kappa$*, output*

$$(R_1, R_2, R_3, \mathcal{H}(k_1||R_1) \oplus \mathcal{H}(k_2||R_2) \oplus \mathcal{H}(k_3||R_3) \oplus m), \quad (10)$$

$$((R_4, \mathcal{H}(k_1||R_4)), (R_5, \mathcal{H}(k_2||R_5)), (R_6, \mathcal{H}(k_3||R_6))) \quad (11)$$

where $\mathcal{H}$ *is the quantum random oracle.*

$\mathsf{CL.Dec}_{k_1,k_2,k_3}(c)$: *Suppose* $c = (R_1, R_2, R_3, c_4)$. *Output* $(\mathcal{H}(k_1||R_1) \oplus \mathcal{H}(k_2||R_2) \oplus \mathcal{H}(k_3||R_3) \oplus c_4)$.

$\mathsf{CL.Ver}(k, i, c)$: *Suppose the ith key tag in c is* $tag_i = (R_i, r)$. *Output* 1 *if* $r = \mathcal{H}(k||R_i)$, *and* $\perp$ *otherwise.*

We choose not to define and discuss the security of this scheme, but use it as a "wrapper" of the KDMP scheme. In the security proof we will "unwrap" its structure and base the proof on the security of KDMP scheme.

6.2 Security of GBC against Classical or Quantum Attack

In this subsection we give the security statements of GBC. First, we can show, when used on classical inputs, $\mathsf{GBC}_{\mathsf{CL}}$ is secure:

Proposition 2. $\mathsf{GBC}_{\mathsf{CL}}$*, where* CL *is defined as Protocol 5, is one-shot IND-CPA secure against quantum adversary (that is, secure when used to encrypt one* classical *input) in the quantum random oracle model. Explicitly, if the distinguisher that the adversary uses makes at most q queries to the quantum random oracle, the input size is N and the size of circuit C is L,*

$$\mathsf{Adv}^{IND-CPA-oneshot}_{\mathsf{GBC}^{C}_{\mathsf{CL}}}(\mathscr{A}, \kappa) \leq \mathsf{poly}(q, N, L)2^{-0.5\kappa}$$

Where poly *is a fixed polynomial that does not depend on* $\mathscr{A}$ *or the parameters.*

The detailed proof is in the full version of this paper.

But we meet some difficulty when we try to prove the qIND-CPA security (that is, the security for quantum inputs). We leave it as a conjecture:

Conjecture 3. $\mathsf{GBC}_{\mathsf{CL}}$ is one-shot qIND-CPA secure in the quantum random oracle model.

But if we use a longer key, we can prove its security.

Theorem 6. *For any BQP adversary* $\mathscr{A}$*, there exists a negligible function* negl *such that:*

$$\mathsf{Adv}^{qIND-CPA-oneshot}_{\mathsf{GBC}_{\mathsf{CL}}}(\mathscr{A}, \kappa) = \mathsf{negl}(\kappa - 4N_q)$$

where N_q *is the size of quantum states in the input.*

In other words, denote GBC' as the protocol of taking $\kappa = \eta + 4N_q$ as the key length in the GBC protocol, we can prove GBC' is one-shot qIND-CPA secure with respect to security parameter η. So we prove:

Theorem 7. *There exists a delegation protocol for C+P gate set that is one-shot qIND-CPA secure in the quantum random oracle model, and the client requires* $O(\eta N_q + N_q^2)$ *quantum* CNOT *gates as well as polynomial classical computation, where* N_q *is the number of qubits in the input and* η *is the security parameter.*

Although we don't have a proof for Conjecture 3, we conjecture it is true, since this protocol seems to be a very natural generalization from classical to quantum. We leave it as an open problem. The main obstacle here is its security cannot be reduced to the semantic security of classical garbled circuits easily: the adversary gets many superpositions of keys. We have to prove it using different techniques, which leads to Theorem 6.

From Theorem 6 we know when we take $\kappa \geq 4N_q$ and consider $\kappa - 4N_q$ as the security parameter the security has been proved. So when the circuit size $L = \omega(N_q^2)$ the quantum resources for the client to run this protocol are smaller than running the circuit itself anyway.

What's more, although our proof requires the quantum random oracle model, we conjecture that this protocol is still secure when we replace the random oracle with practical hash functions or symmetric key encryption schemes:

Conjecture 4. When we replace the quantum random oracle in $\mathsf{GBC}_{\mathsf{CL}}$ with practical hash functions or symmetric key encryption schemes, such as versions of SHA-3 or AES with appropriate input and output sizes, the security statements still hold.

6.3 Security Proof

IND-CPA security of Protocol 1 The proof of Proposition 2 is postponed into the full version of this paper. The proof is based on Theorem 5, which is about KDM security of Protocol 4. The structure of our scheme, when used classically, can be seen as a special case of the KDM function. But the definition of IND-CPA security for protocol GBC is still different from the KDM game security: in GBC we are trying to say the inputs of Et are hidden, but KDM security is about the encrypted messages in the garbled table. So it doesn't follow from the security of KDMP protocol trivially.

Discussions of the qIND-CPA security To prove Theorem 6, we use a different security proof technique, which enables us to base the qIND-CPA advantage on the IND-CPA advantage and a classical "hard-to-compute" lemma. This technique enables us to argue about the security of a quantum protocol using only security results in the classical settings.

We need to prove the keys that are not "revealed" are "hard to compute". Then we expand the expression of the qIND-CPA advantage, write it as the sum of exponential number of terms and we can observe that their forms are the same as the probability of "computing the unrevealed keys". We can prove these terms are all exponentially small, thus we get a bound for the whole expression.

Lemma 2. *For any C+P circuit C, $|C| = L$, any adversary that uses distinguisher $\mathcal{D}$ which can query the quantum random oracle q times (either with classical or quantum inputs), given the reversible garbled table and input keys corresponding to one input, it's hard to compute the input keys corresponding to* *other input. Formally, for any $i \neq j, |\varphi_i\rangle$, we have*

$$\mathbb{E}_K \mathbb{E}_R \operatorname{tr}((\mathsf{Et}_K |j\rangle)^\dagger \mathcal{D}(\mathsf{Et}_K(|i\rangle \langle i|) \otimes \varphi_i \otimes \mathsf{TAB}^C_{\mathsf{CL}}(K,R))(\mathsf{Et}_K |j\rangle)) \\ \leq \mathsf{poly}(q,N,L)2^{-0.5\kappa} \quad (12)$$

where poly *is a fixed polynomial that does not depend on $\mathscr{A}$ or the parameters, N is the size of inputs, and R denotes the randomness used in the computation of $\mathsf{TAB}^C_{\mathsf{CL}}(K)$, including the random oracle outputs, the random paddings and the random shuffling. And $\mathsf{TAB}^C_{\mathsf{CL}}(K,R)$ is the output of $\mathsf{TAB}^C_{\mathsf{CL}}(K)$ using randomness R, and since R is given as a parameter there will be no randomness inside.*

Note that since we have already fixed all the randomness, $\mathsf{TAB}^C_{\mathsf{CL}}(K,R)$ is pure. We also note that this can be seen as a classical lemma since $|i\rangle$, $|j\rangle$ are all in computational basis. We postpone the proof into the full version.

Let's prove Theorem 6 from Proposition 2 and Lemma 2. We will expand the the expression of the input state and qIND-CPA advantage, and each term in the cross terms can be bounded by (12).

Proof (of Theorem 6). First, suppose the state that the adversary uses is $|\varphi\rangle = \sum_i c_i |i\rangle |\varphi_i\rangle$, where i is in the input system, $i \in I$ where I is the set of non-zero term($c_i \neq 0$), $|I| \leq 2^{N_q}$ and $|\varphi_i\rangle$ is in the reference system. Additionally assume c_is are all real numbers and $|\,|i\rangle|\varphi_i\rangle\,| = 1$. We can only consider pure states since we can always write a mixed state as a probability ensemble of pure states.

Then we can assume the distinguisher $\mathcal{D}$ is a unitary operation D on the output and auxiliary qubits, followed by a measurement on a specific output qubit. So we can write $\mathcal{D}(\rho) = \operatorname{tr}_R(D(\rho \otimes |0\rangle\langle 0|)D^\dagger)$, where $|0\rangle\langle 0|$ stands for big enough auxiliary qubits. Let's use $\mathcal{E}_{proj}(\rho)$ to denote the operation of projecting ρ onto the computational basis. Denote the projection operator onto the $|0\rangle\langle 0|$ space as P_0, we have

$$\begin{aligned} &\mathsf{Adv}^{qIND-CPA-oneshot}_{\mathsf{GBC}}(\mathscr{A},\kappa) && (13)\\ =&|\Pr(\mathcal{D}(\mathbb{E}_K \mathsf{GBC.Enc}_K(\varphi)) = 1)) - \Pr(\mathcal{D}(\mathbb{E}_K \mathsf{GBC.Enc}_K(0^N)) = 1)| && (14)\\ \leq&|\Pr(\mathcal{D}(\mathbb{E}_K \mathbb{E}_R(\rho)) = 1)) - \Pr(\mathcal{D}(\mathbb{E}_K \mathbb{E}_R(\mathcal{E}_{proj}(\rho))) = 1)|+ \\ &|\Pr(\mathcal{D}(\mathbb{E}_K \mathsf{GBC.Enc}_K(\mathcal{E}_{proj}(\varphi))) = 1)) - \Pr(\mathcal{D}(\mathbb{E}_K \mathsf{GBC.Enc}_K(0^N)) = 1)| && (15)\end{aligned}$$

Here we write $\rho := (\mathsf{Et}_K \otimes \mathsf{I})(\varphi) \otimes \mathsf{TAB}(K,R)$.

Let's first compute the first term.

$$\begin{aligned} &|\Pr(\mathcal{D}(\mathbb{E}_K \mathbb{E}_R(\rho)) = 1)) - \Pr(\mathcal{D}(\mathbb{E}_K \mathbb{E}_R(\mathcal{E}_{proj}(\rho))) = 1))| && (16)\\ =&|\operatorname{tr}(P_0(\mathbb{E}_K \mathbb{E}_R D(\rho \otimes |0\rangle\langle 0|)D^\dagger)) - \operatorname{tr}(P_0(\mathbb{E}_K \mathbb{E}_R D(\mathcal{E}_{proj}(\rho) \otimes |0\rangle\langle 0|)D^\dagger))| && (17)\end{aligned}$$

The first term inside can be expanded as

$$
\begin{aligned}
&\mathbb{E}_K \, \mathbb{E}_R \, D(\rho \otimes |0\rangle \langle 0|)D^\dagger \quad (18)\\
=&\mathbb{E}_K \, \mathbb{E}_R \, D((\mathsf{Et}_K \otimes \mathsf{I})(\varphi) \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger \quad (19)\\
=&\mathbb{E}_K \, \mathbb{E}_R \, D((\mathsf{Et}_K \otimes \mathsf{I})((\sum_i c_i \, |i\rangle \, |\varphi_i\rangle)(\sum_i c_i^\dagger \, \langle i| \, \langle \varphi_i|))\\
&\qquad (\mathsf{Et}_K \otimes \mathsf{I})^\dagger \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger \quad (20)
\end{aligned}
$$

Denote $|x_i\rangle = \mathsf{Et}_K \, |i\rangle \otimes |\varphi_i\rangle$, we can simplify the expression:

$$
\begin{aligned}
(20) =&\mathbb{E}_K \, \mathbb{E}_R \, D(\sum_i c_i \, |x_i\rangle \sum_i c_i^\dagger \, \langle x_i| \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger \quad (21)\\
=&\mathbb{E}_K \, \mathbb{E}_R \, D(\sum_i |c_i|^2 \, |x_i\rangle \, \langle x_i| \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger\\
&+ \mathbb{E}_K \, \mathbb{E}_R \, D(\sum_{i \neq j} c_i c_j^\dagger \, |x_i\rangle \, \langle x_j| \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger \quad (22)\\
=&\mathbb{E}_K \, \mathbb{E}_R \, D(\mathcal{E}_{proj}(\rho) \otimes |0\rangle \langle 0|)D^\dagger\\
&+ \mathbb{E}_K \, \mathbb{E}_R \, D(\sum_{i \neq j} c_i c_j^\dagger \, |x_i\rangle \, \langle x_j| \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger \quad (23)
\end{aligned}
$$

Substitute it into (17), we get

$$
\begin{aligned}
&(17)\\
=&| \, \mathbb{E}_K \, \mathbb{E}_R \, \mathrm{tr}(P_0 D(\sum_{i \neq j} c_i c_j^\dagger \, |x_i\rangle \, \langle x_j| \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|)D^\dagger)| \quad (24)\\
=&| \sum_{i \neq j} c_i c_j^\dagger \, \mathbb{E}_K \, \mathbb{E}_R (\langle x_j| \, \langle \mathsf{TAB}(K,R)| \, \langle 0| \, D^\dagger P_0 D(|x_i\rangle \, |\mathsf{TAB}(K,R)\rangle \, |0\rangle))| \quad (25)\\
\leq&\sqrt{\sum_{i \neq j} c_i^2 {c_j^\dagger}^2} \sqrt{\sum_{i \neq j} | \, \mathbb{E}_K \, \mathbb{E}_R \, \langle 0| \, \langle \mathsf{TAB}(K,R)| \, \langle x_j| \, D^\dagger P_0 D \, |x_i\rangle \otimes |\mathsf{TAB}(K,R)\rangle \, |0\rangle \, |^2}\\
&\quad (26)\\
\leq&\sqrt{\sum_{i \neq j} \mathbb{E}_K \, \mathbb{E}_R \, |(\langle 0| \otimes \langle \mathsf{TAB}(K,R)| \, \langle x_j|)D^\dagger P_0 D(|x_i\rangle \otimes |\mathsf{TAB}(K,R)\rangle \, |0\rangle)|^2} \quad (27)
\end{aligned}
$$

The magic of this technique actually happens between (24) and (25): first we move $\sum_{i \neq j} c_i c_j^\dagger$ out by linearity, then after rotating terms inside the trace, an expression which talks about applying D on some state becomes an expression for the probability of applying $\{D^\dagger P_0 D, D^\dagger P_1 D\}$ on $|x_i\rangle$ and getting $|x_j\rangle$.

By Lemma 2, consider the operation $\mathcal{E}$ defined as follows: expand the space and apply D, make a measurement with operators $\{P_0, P_1\}$, and apply $D^\dagger$. Let

$\mathcal{E}_0 = D^\dagger P_0 D(\cdot \otimes |0\rangle \langle 0|) D^\dagger P_0 D$, and $\mathcal{E}_1 = D^\dagger P_1 D(\cdot \otimes |0\rangle \langle 0|) D^\dagger P_1 D$. We have:

$$\begin{aligned}
&\mathbb{E}_K \, \mathbb{E}_R (\mathrm{tr}((\mathsf{Et}_K |j\rangle)^\dagger \mathcal{E}_0 (\mathsf{Et}_K(i) \otimes \varphi_i \otimes \mathsf{TAB}(K,R)) \mathsf{Et}_K |j\rangle)) \tag{28}\\
&+ \mathrm{tr}((\mathsf{Et}_K |j\rangle)^\dagger \mathcal{E}_1 (\mathsf{Et}_K(|i\rangle \langle i|) \otimes \varphi_i \otimes \mathsf{TAB}(K,R)) \mathsf{Et}_K |j\rangle)) \tag{29}\\
\le &\mathsf{poly}(q,N,L) 2^{-0.5\kappa} \tag{30}
\end{aligned}$$

With this, we can bound the inner part of (27) further:

$$\begin{aligned}
&\mathbb{E}_K \, \mathbb{E}_R \, |(\langle 0| \otimes \langle \mathsf{TAB}(K,R)| \langle x_j|) D^\dagger P_0 D (|x_i\rangle \otimes |\mathsf{TAB}(K,R)\rangle |0\rangle)|^2 \tag{31}\\
= &\mathbb{E}_K \, \mathbb{E}_R \, |(\langle 0| \otimes \langle \mathsf{TAB}(K,R)| \, ((\mathsf{Et}_K |j\rangle) \otimes |\varphi_j\rangle)^\dagger \\
&\qquad D^\dagger P_0 D (\mathsf{Et}_K |i\rangle \otimes |\varphi_i\rangle) \otimes |\mathsf{TAB}(K,R)\rangle |0\rangle)|^2 \tag{32}\\
\le &\mathbb{E}_K \, \mathbb{E}_R \, \mathrm{tr}((\mathsf{Et}_K |j\rangle)^\dagger \mathcal{E}_0 (\mathsf{Et}_K(|i\rangle \langle i|) \otimes \varphi_i \otimes \mathsf{TAB}(K,R) \otimes |0\rangle \langle 0|) \mathsf{Et}_K |j\rangle) \tag{33}\\
\le &\mathsf{poly}(q,N,L) 2^{-0.5\kappa} \tag{34}
\end{aligned}$$

Substitute it back into (27), we will know

$$\begin{aligned}
&|\Pr(\mathcal{D}(\mathbb{E}_K \, \mathbb{E}_R(\rho)) = 1) - \Pr(\mathcal{D}(\mathbb{E}_K \, \mathbb{E}_R(\mathcal{E}_{proj}(\rho))) = 1)| \tag{35}\\
\le &2^{N_q} \mathsf{poly}(q,N,L) 2^{-0.25\kappa} \tag{36}
\end{aligned}$$

The second term in (15) can be bounded by Proposition 2. $\mathcal{E}_{proj}(\rho)$ is a classical state so we have

$$\begin{aligned}
&|\Pr(\mathcal{D}(\mathbb{E}_K \, \mathsf{GBC.Enc}_K(\mathcal{E}_{proj}(\varphi))) = 1) - \Pr(\mathcal{D}(\mathbb{E}_K \, \mathsf{GBC.Enc}_K(0^N)) = 1)|\\
\le &\mathsf{poly}(q,N,L) 2^{-\kappa}
\end{aligned}$$

Combining these two inequalities we have

$$\mathsf{Adv}_{\mathsf{GBC}}^{qIND-CPA-oneshot}(\mathscr{A}, \kappa) \le \mathsf{poly}(q,N,L) 2^{-0.25(\kappa - 4N_q)}$$

6.4 Standard Model

In the last section we prove the security in the quantum random oracle model. In practice, the random oracle can usually be replaced with hash functions, and we claim that our protocol is not an exception (Conjecture 4). In our protocol, it's more natural to use a symmetric key encryption scheme directly: the usage of the random oracle in our protocol is on the symmetric multi-key encryption scheme with key tags, and the key verification can be replaced with the "point-and-permute" technique from the classical garbled circuit.

When using symmetric key encryption instead of the random oracle, since in our protocol we use affine functions in KDM game, we need at least that the symmetric key encryption is secure against quantum adversaries under KDM game for affine functions. Although this is a strong assumption, it's still reasonable in practice.

7 Applications

7.1 Blind Quantum Computation for C+P Circuits

Protocol 1 is a quantum computation delegation protocol. But since the circuit can be put into inputs, we can turn it into a blind quantum computation protocol, where the server doesn't know either input state or the circuit to be applied. If we only want to hide the type of gates in the circuit, our original protocol actually already achieves it. But if we also want to hide the circuit topology, we need to do more. The adversaries should only know the fact that the circuit is a C+P circuit, the input size and an upper bound on the circuit size. In this subsection we are going to construct a universal machine $\mathcal{U}$ such that for all the C+P circuit C, $C(\rho) = \mathcal{U}(C, \rho)$. What's more, we want $\mathcal{U}$ to be in C+P so that we can use our protocol on $\mathcal{U}$.

Suppose the size of input is N and the phase gates are all in the form of $R_Z(\pi/2^d), d \in [D]$. Then there are $N^3 + ND$ possible choices for each gate. Thus a $\log(N^3+ND)$ bits description is enough for each gate. For the server-side evaluation, a bad implementation may lead to $N^3 + ND$ extra cost, and we can do a simple preprocessing on the circuit to reduce it: We can first introduce three auxiliary wires, and convert C to a form that only contains three types of gates: (1)$R_Z(\pi/2^d)$ (2)a SWAP operation between a normal wire and an auxiliary wire (3)a Toffoli gate on the auxiliary wires. After this transformation, the number of choices of the gates is only $3N + 1 + ND$. Thus we can describe each gate by a string of length $\log(3N + 1 + ND)$. And given the description of g, the operation of $\mathcal{U}$ is a series of multi-controlled gate operations, where the control wires correspond to the gate description and the target wires are the wires in the original circuit. And this multi-controlled multi-target operation is also in C+P and it can be transformed to the standard form of $\mathsf{Toffoli}$ and phase gates.

Since $\mathcal{U}$ itself is a C+P circuit, we can delegate it by applying Protocol 1. Then the original circuit will be indistinguishable from the identity circuit, which means we know nothing beyond some information on its size.

7.2 Delegation of Shor's Algorithm

Shor's algorithm contains two parts: first we apply lots of Toffoli gates on $|+\rangle^{\otimes n} \otimes |M\rangle$, where M is, for example, the number to be factored, and $n = \log M$; then measure, apply quantum Fourier transform and measure again. From [10][19] we know the quantum Fourier transform is actually easy to implement: a quantum Fourier transform on n qubits has time complexity $\tilde{O}(n)$. The main burden of Shor's algorithm is the Toffoli part. ([19] contains resource estimates on the elliptic curve version.) With this protocol we can let the server do the Toffoli part of Shor's algorithm without revealing the actual value of the input.

Explicitly, suppose the client wants to run Shor's algorithm on M while also wants to keep M secret, the client can use the following protocol:

Protocol 6 *Protocol for delegation of Shor's algorithm:*

Suppose ShorToff is the Toffoli gate part of Shor's algorithm, and its length is L.

1. *The client samples* $K \leftarrow \mathsf{GBC.KeyGen}(1^\kappa, 1^{2n}, 1^L)$*. Then the client prepares* $(\rho, tab) \leftarrow \mathsf{GBC.Enc}_K^{ShorToff}(|+\rangle^{\otimes n} \otimes |M\rangle)$ *and sends it to the server.*
2. *The server evaluates* $\mathsf{GBC}_{\mathsf{CL}}.\mathsf{Eval}^{ShorToff}(\rho, tab)$ *and sends it back to the client.*
3. *The client decrypts with* $\mathsf{GBC.Dec}_K$*. Then it does quantum Fourier transform itself and measures to get the final result.*

So the quantum resources on the client side are only $O(\kappa n)$ CNOT gates plus $\tilde{O}(n)$ gates for quantum Fourier transform, and it can delegate Shor's algorithm to the server side securely.

Theorem 8. *Protocol 6 can be used to delegate Shor's algorithm securely and non-interactively, in the quantum random oracle model(without assuming trapdoor one-way functions), and for n bit inputs, the amount of quantum resources on the client side are quasi-linear quantum gates plus* $O(\kappa n)$ CNOT *gates (assuming Conjecture 3,* $\kappa = \eta$*, or under the current security proof,* $\kappa = \eta + 4n$*).*

For comparison, if the client runs Shor's algorithm locally, the client needs to perform $\omega(n^2 \log n)$ Toffoli gates, and the exact form depends on the multiplication method it uses. Schoolbook multiplication leads to $O(n^3)$ complexity; if it uses fast multiplication method, the complexity is still $\omega(n^2 \log n)$ and it has a big hidden constant.

8 Quantum KDM Security

As a natural generalization of our discussion of KDM-security, we formalize the quantum KDM security and construct a protocol in this section. Previously when we discuss the KDM security the function f and message m are classical; here we further generalize them to include quantum states and operations.

Definition 10. *A symmetric key non-adaptive quantum KDM game naSymQKDM for function family* $\mathcal{F}$ *in the quantum random oracle model is defined as follows:*

1. *The challenger chooses bit* $b \leftarrow_r \{0,1\}$ *and samples* $K = \{sk_i\}_{i=1}^N$*,* $sk_i \leftarrow \mathsf{KeyGen}(1^\kappa)$*.*
2. *The adversary and the challenger repeat the following for L times,non-adaptively, in other words, the challenger should only sends out the answers in step (b) after it receives all the queries:*
   - *(a) The adversary picks index i, function* $f \in \mathcal{F}$ *and message* $\rho \in \mathbb{D}(\mathcal{R} \otimes \mathcal{M})$*, and sends system* $\mathcal{M}$ *to the challenger.*
   - *(b) If* $b = 1$*, the challenger returns* $c = \mathsf{Enc}_{sk_i}(f(K, \rho_m))$ *to the adversary. If* $b = 0$*, the challenger returns* $c = \mathsf{Enc}_{sk_i}(0^{|f(K,\rho_m)|})$*.*
3. *The adversary tries to guess b with some distinguisher* $\mathcal{D}$*, and outputs* b'*.*

Note that $\mathcal{F}$ can be quantum operations and can query the random oracle with quantum states. The output of functions in $\mathcal{F}$ should be fixed-lengthed, otherwise $|f(K, m)|$ will not be well-defined.

The guessing advantage is defined as $\mathsf{Adv}^{naSymQKDM}(\mathscr{A}, \kappa) = |\Pr(b' = 1|b = 1) - \Pr(b' = 1|b = 0)|$

Definition 11. *A symmetric key quantum encryption scheme is nonadaptively qKDM-CPA secure for function $\mathcal{F}$ if for any BQP adversary $\mathscr{A}$,*

$$\mathsf{Adv}_{\mathcal{F}}^{naSymQKDM}(\mathscr{A}, \kappa) = \mathsf{negl}(\kappa)$$

8.1 Protocol Design

Protocol 7 *A Quantum KDM Secure Protocol in the Quantum Random Oracle Model:*

Key Generation $\mathsf{QKDM.KeyGen}(1^\kappa)$*:* $sk \leftarrow \{0,1\}^\kappa$*.*
Encryption $\mathsf{QKDM.Enc}_{sk}(\rho)$*: Sample* $a, b \in_r \{0,1\}^N$*, where* N *is the length of inputs.*
Output $(\mathsf{X}^a\mathsf{Z}^b(\rho), \mathsf{KDMP.Enc}_{sk}(a, b))$*.*
Decryption $\mathsf{QKDM.Dec}_{sk}((\rho, c))$*: First compute* $a, b \leftarrow \mathsf{KDMP.Dec}_{sk}(c)$*, then output* $\mathsf{X}^a\mathsf{Z}^b(\rho)$

Theorem 9. *Protocol 7 is nonadaptively qKDM-CPA secure for functions in* $\mathcal{F}[\mathsf{poly}]$ *in the quantum random oracle model, where* $\mathcal{F}[\mathsf{poly}]$ *is the function family that makes at most* $\mathsf{poly}(\kappa)$ *queries to the quantum random oracle.*

We put its proof in the full version of this paper.

9 Open Problems

One obvious open problem in our paper is to prove Conjecture 3, the qIND-CPA security without additional requirement on κ. We believe this is true, but we can only prove the security when $\kappa - 4N_q = \eta$. And another further research direction is to base these protocols directly on the assumptions in the standard model, for example, the existence of hash functions or symmetric key encryption schemes that are exponentially KDM secure for affine functions against a quantum adversary. We can also study how to optimize this protocol, and how efficient it is compared to other protocols based on the quantum one-time pad. One obvious route is to make use of the optimization techniques for classical garbled circuits.

Another open question is whether this protocol is useful in other problems than Shor's algorithm. Lots of previous works studied quantum circuits on $\{\mathsf{Clifford}, \mathsf{T}\}$ gate set, and our work shows $\{\text{C+P}, \mathsf{H}\}$ is also important and worth studying. There are not many works on converting quantum circuits into layers of C+P gates and H gates, and it's possible that some famous quantum algorithms which require a lot of T gates, after converted into $\{\text{C+P}, \mathsf{H}\}$ gate set, can have small H depth. This problem is still quite open, and further research is needed here.

What's more, KDM security in quantum settings is an interesting problem. This paper gives some initial study on it, but there are still a lot of open questions. Is it possible to construct quantum KDM secure protocol in the standard model? Could quantum cryptography help us design classical KDM secure scheme?Again, further research is needed here.

This paper also gives some new ideas on constructing secure quantum encryption schemes without using trapdoor functions. Although there is some result[24] on the limit of information-theoretically secure quantum homomorphic encryption, in our work we use the quantum random oracle and make the circuits available to the client, the limit doesn't hold any more. So here comes lots of interesting problems on the possibility and impossibility of quantum computation delegation: What is the limit for non-interactive information-theoretically secure delegation of quantum computation, where the circuit is public/private, with/without quantum ROM? If we allow small amount of quantum/classical communication, does it lead to something different?

Acknowledgements

The author would like to thank Prof. Adam Smith, NSF funding and anonymous reviewers.

References

1. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems pp. 595–618 (2009)
2. Arrighi, P., Salvail, L.: Blind quantum computation. International Journal of Quantum Information **04** (10 2003)
3. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. (01 1997)
4. Black, J., Rogaway, P., Shrimpton, T.: Encryption-scheme security in the presence of key-dependent messages. In: Nyberg, K., Heys, H. (eds.) Selected Areas in Cryptography. pp. 62–75. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
5. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. pp. 41–69. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
6. Broadbent, A., Fitzsimons, J., Kashefi, E.: Universal blind quantum computation. In: Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science. pp. 517–526. FOCS '09, IEEE Computer Society, Washington, DC, USA (2009)
7. Broadbent, A., Jeffery, S.: Quantum homomorphic encryption for circuits of low t-gate complexity. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology – CRYPTO 2015. pp. 609–629. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
8. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. Journal of the ACM **51** (02 1970). https://doi.org/10.1145/1008731.1008734

9. Childs, A.M.: Secure assisted quantum computation. Quantum Info. Comput. **5**(6), 456–466 (Sep 2005)
10. Cleve, R., Watrous, J.: Fast parallel circuits for the quantum fourier transform (06 2000)
11. Dulek, Y., Schaffner, C., Speelman, F.: Quantum homomorphic encryption for polynomial-sized circuits. In: Proceedings, Part III, of the 36th Annual International Cryptology Conference on Advances in Cryptology — CRYPTO 2016 - Volume 9816. pp. 3–32. Springer-Verlag, Berlin, Heidelberg (2016)
12. Fischlin, M., Lehmann, A., Ristenpart, T., Shrimpton, T., Stam, M., Tessaro, S.: Random oracles with(out) programmability. pp. 303–320 (12 2010). https://doi.org/10.1007/978-3-642-17373-8-18
13. Goldwasser, S., Micali, S.: Probabilistic encryption &amp; how to play mental poker keeping secret all partial information. In: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing. pp. 365–377. STOC '82, ACM, New York, NY, USA (1982)
14. Lai, C.Y., Chung, K.M.: On statistically-secure quantum homomorphic encryption (09 2018)
15. Mahadev, U.: Classical homomorphic encryption for quantum circuits. CoRR (2017)
16. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, New York, NY, USA, 10th edn. (2011)
17. Okamoto, T., Tanaka, K., Uchiyama, S.: Quantum public-key cryptosystems. In: Bellare, M. (ed.) Advances in Cryptology — CRYPTO 2000. pp. 147–165. Springer Berlin Heidelberg, Berlin, Heidelberg (2000)
18. Raussendorf, R.: Measurement-based quantum computation with cluster states. International Journal of Quantum Information **07**(06), 1053–1203 (2009)
19. Roetteler, M., Naehrig, M., Svore, K., Lauter, K.: Quantum resource estimates for computing elliptic curve discrete logarithms (06 2017)
20. Selinger, P.: Quantum circuits of t-depth one. Phys. Rev. A **87**, 042302 (Apr 2013)
21. Unruh, D.: Post-quantum security of fiat-shamir (2017), https://eprint.iacr.org/2017/398
22. Yao, A.C.: How to generate and exchange secrets. In: 27th Annual Symposium on Foundations of Computer Science (sfcs 1986). pp. 162–167 (Oct 1986). https://doi.org/10.1109/SFCS.1986.25
23. Yingkai Ouyang, Si-Hui Tan, J.F.: Quantum homomorphic encryption from quantum codes (2015)
24. Yu, L., Perez-Delgado, C.A., Fitzsimons, J.: Limitations on information theoretically secure quantum homomorphic encryption **90** (06 2014)