Hindawi
Security and Communication Networks
Volume 2021, Article ID 6695304, 7 pages
https://doi.org/10.1155/2021/6695304

Research Article

Two-Party Secure Computation for Any Polynomial Function on Ciphertexts under Different Secret Keys

Bingbing Jiang

Computer Science and Technology Department, Nanjing University, Nanjing, China

Correspondence should be addressed to Bingbing Jiang; njubing.jiang@gmail.com

Received 8 December 2020; Revised 8 January 2021; Accepted 8 February 2021; Published 22 February 2021

Academic Editor: Jinguang Han

Multikey fully homomorphic encryption proposed by Lopez-Alt et al. (STOC12) is a significant primitive that allows one to perform computation on the ciphertexts encrypted by multiple different keys independently. Then, several schemes were constructed based on decisional small polynomial ratio or learning with errors. These schemes all require an expansion algorithm to transform a ciphertext under a single key into an encryption of the same message under a set of keys. To achieve the expansion algorithm without interaction with these key-keepers, their encryption algorithm not only outputs a ciphertext of a plaintext but also exports auxiliary information generated from the randomness used in the former encryption process. Beyond that, the size of the ciphertext encrypted by multiple keys increases linearly or quadratically in the number of participants. In this paper, we studied the problem whether someone can directly perform arbitrary computation on ciphertexts encrypted by different keys without any auxiliary information in the output of the encryption algorithm and an increase in the size of the ciphertext in the expansion algorithm. To this end, we proposed a novel and simple scheme of secure computation on ciphertexts under two different keys directly without any auxiliary information. In other words, each party just provides its own ciphertexts encrypted by the GSW scheme (CRYPTO13). In the procedure of executing evaluation on these ciphertexts, the size of the new ciphertext remains the same as that of the GSW ciphertext.

1. Introduction

The concept of multikey fully homomorphic encryption was proposed by Lopez-Alt et al. [1], which allows someone to perform arbitrary computations on the ciphertexts encrypted by multiple different secret keys. Specifically, each party independently encrypts input x_i, to obtain a ciphertext $c_i = \mathrm{Enc}_{pk_i}(x_i)$, and one can homomorphically evaluate an arbitrary function on these encrypted data without interaction between them. After this, there has been a lot of research [2–12] for its assumptions, functionalities, and performance.

The main application of multikey FHE is that a plurality of parties is informed to engage in a computing task after they have submitted their data. This is a significant difference from the applications of the traditional (single-key) encryption schemes. For example, two hospitals want to cooperate and study the influence factors of some disease. However, the data of these patients has been encrypted and stored in their own servers ahead of this cooperation. How could an evaluation algorithm be performed directly on these ciphertexts without decrypting them? In [1], Lopez-Alt et al. focused on a problem whereby a (untrusted) cloud server wants to perform some computations over data from multiple clients without interacting with them after each client transmits their own (encrypted) input to the cloud and other clients. In the scheme proposed by Lopez-Alt et al. [1], although a ciphertext only contains an encryption of a plaintext, the size of a ciphertext under multiple secret keys becomes much larger than that of the original ciphertext and its security is based on the nonstandard assumption. The ciphertext's length is related to the number of participants where the former increases at least linearly in the later. In the scheme of Clear and McGoldrick [3], an encryption of a message contains a universal mask U generated by another public-key encryption scheme. Also, the ratio of the size of

the ciphertext under multiple keys and that under single key grow quadratically with an increase in the number of the associated participants. Afterwards, Mukherjee and Wichs [2] proposed an optimized scheme with a simple generation of the universal mask. However, there is still auxiliary information in the encryption algorithm and the ratio remains quadratic. Following the previous works, there are two independent researches about multikey fully homomorphic encryption introduced by Brakerski and Perlman [5] and Peikert and Shiehian [4], respectively. In the former scheme, although the authors replaced the algorithm of the universal mask with the bootstrapping algorithm, the ciphertext's growth rate was still linear and their evaluation keys were generated by the previous multikey fully homomorphic encryption schemes. There are two versions in the paper in [4]. In the first scheme, the encryption of a message contains a commitment of the message and an encryption of the randomness used in the former commitment algorithm. The ratio becomes linear. In the second one, the encryption algorithm only outputs a ciphertext of a message, but the ratio becomes quadratic and the evaluation keys are generated by the first scheme. In [13], the growth rate is quadratic, and the output of the encryption algorithm also contains auxiliary information except a ciphertext of a plaintext. Recently, Chen et al. [6] proposed a multikey FHE scheme based on the ring-LWE (Learning with Errors) assumption, in which their ciphertext-extension algorithm only generates the evaluated keys for the scheme with multiple keys but the size of the ciphertext under multiple keys also relies on the number of associated parties.

The first multikey fully homomorphic encryption was proposed by Lopez-Alt et al., but their solution is based on nonstandard assumptions. Subsequent solutions, despite being based on standard cryptographic assumptions (LWE), have two common shortcomings. The first shortcoming is that they require the encryption of not only the plaintext but also random numbers that have been used; namely, $c = \mathrm{Enc}(\mathrm{pk}, m, r)$, and $U = \mathrm{Enc}(\mathrm{pk}, r)$. Each ciphertext must be attached with additional information U. The second one is that the length of the ciphertext increases linearly or quadratically with the number of participants. In this paper, our main research problem is how to directly perform secure computation on ciphertext data c directly provided by each user without any additional information U. These ciphertext data are encrypted with different secret keys. Our main focus here is the case of encryption with two different keys. We begin by taking the GSW13 encryption scheme [14] into consideration as we notice that the main process of its decryption algorithm is the inner product of two vectors; that is, $\langle c, v\rangle = m\ d + e$, where d is a large constant. As such, if we want to calculate the product of ciphertexts c_1 and c_2 encrypted with different secret keys, we only need to calculate $c_1 \cdot c_2^T$. This is because $v_1^T \cdot c_1 \cdot c_2^T \cdot v_2 = (m_1 d + e_1)(m_2 d + e_2) = m_1 m_2 d^2 + d(m_1 e_2 + m_2 e_1) + e_1 e_2$. The final result is desirable, with $m_1 m_2$ being one of its factors. However, there is another problem: the constant factor becomes d^2, and small noises e_1 and e_2 are also multiplied by a large number. Therefore, we must find a way to decrease the constant factor to d, while keeping the noises within an acceptable range. Because the noise in the ciphertext grows with an increase in the number of addition and multiplication operations, when it increased to some value defined by the public parameters, it may cause incorrect decryption of the output ciphertext. Therefore, we should reduce the noise growth in evaluation.

Our approach is to decrypt $c_1 \cdot c_2$ in two steps without directly multiplying it by two secret keys. Instead, a single secret key is first used to decrypt it, that is, $v_1^T \cdot c_1 \cdot c_2^T = (m_1 d + e_1) c_2$ (denoted as tc_1), before tc_1/d is calculated and rounded to obtain $tc = m_1 c_2$. Finally, another secret key is used to decrypt tc for the final plaintext $m_1 m_2$. During the process, noises have been kept at a low level without being multiplied by a large constant factor. To sum up, the above description explains how to perform the multiplication operation on ciphertexts encrypted with two different keys. The addition operation can be transformed to the multiplication operation; that is, $c_1 + c_2 = (c_1 \cdot c_2') + (c_1' \cdot c_2)$, where c_1' and c_2' are encrypted from plaintext 1 with different secret keys. Till this step, we completed the addition and multiplication operations on ciphertexts encrypted with two different secret keys. However, this scheme has a shortcoming: the multiplication operation can only be performed once as the result of the multiplication operation on the ciphertexts encrypted with two different secret keys cannot be multiplied by other ciphertexts. In order to enable the support of polynomial calculation, we can write any polynomial f with $u + v$ inputs $(x_1, \ldots, x_u, y_1, \ldots, y_v)$ as follows: $f = \sum_{i=1}^{w} (f_i \cdot g_i)$, where the inputs of f_i are $x_1, \ldots, x_u$, and the inputs of g_i are $y_1, \ldots, y_v$. In this way, we can first use the single-key fully homomorphic encryption scheme to calculate f_i and g_i to obtain intermediate results and then calculate the final results with our proposed method. Therefore, our secure computation only involves the GSW13 encryption scheme without the requirement for additional information U. Moreover, unlike previous schemes where a ciphertext's size grows linearly or quadratically as the number of secret keys increases, the ciphertext in our scheme always maintains its original size.

Our Contributions. We proposed a protocol that allows one to perform any polynomial functions on the GSW ciphertexts under two different keys directly. Unlike the previous works, each party just provides the GSW ciphertexts without anything auxiliary of the private inputs and the size of the new ciphertext remains invariant when executing evaluations on these ciphertexts. In our *Addition* and *Multiplication* algorithms on ciphertexts under two different keys, the noise increases linearly. Compared to the scheme in [1], our scheme is based on the standard assumption. Our scheme reduces the size of the ciphertext under a single key from $\mathcal{O}(n^4 \log^4 q)$ in [2, 3] to $\mathcal{O}(n^2 \log^2 q)$, where n is the lattice dimension and q is a modulus. Compared to the scheme in [5], our scheme does not require the expensive technique of bootstrapping to transform a ciphertext under a single key to a ciphertext under a set of keys. In the first scheme of [4], the size of the ciphertext under a single key is $\mathcal{O}(n^3 \log^3 q)$. The second scheme of [4] requires its first scheme to generate a public key with larger size. Different

from the scheme in [6], the size of the public key in our scheme is the same as that of the GSW13 scheme, whereas it is $\mathcal{O}(\log q)$ times the size of the GSW13 scheme.

2. Related Work

In the scheme proposed by Lopez-Alt et al. [1], although a ciphertext only contains an encryption of a plaintext, the size of a ciphertext under multiple secret keys becomes much larger than that of an original ciphertext and their security is based on the nonstandard assumption. The ciphertext's length is related with the number of participants where the former increases at least linearly in the latter. In the scheme of Clear and McGoldrick [3], an encryption of a message contains a universal mask U generated by another public-key encryption scheme. Also, the ratio of the size of the ciphertext under multiple keys and that under single key grow quadratically with an increase in the number of the associated participants. Afterwards, Mukherjee and Wichs [2] proposed an optimized scheme with a simple generation of the universal mask. However, there is still auxiliary information in the encryption algorithm and the ratio remains quadratic. Following the previous works, there are two independent researches about multikey fully homomorphic encryption introduced by Brakerski and Perlman [5] and Peikert and Shiehian [4], respectively. In the former scheme, although the authors replaced the algorithm of the universal mask with the bootstrapping algorithm, the ciphertext's growth rate was still linear and their evaluation keys were generated by the previous multikey fully homomorphic encryption schemes. There are two versions in the paper in [4]. In the first scheme, the encryption of a message contains a commitment of the message and an encryption of the randomness used in the former commitment algorithm. The ratio becomes linear. In the second one, the encryption algorithm only outputs a ciphertext of a message, but the ratio becomes quadratic and the evaluation keys are generated by the first scheme. In [13], the growth rate is quadratic and the output of the encryption algorithm also contains auxiliary information except a ciphertext of a plaintext. Recently, Chen et al. [6] proposed a multikey FHE scheme based on the ring-LWE assumption, in which their ciphertext-extension algorithm only generates the evaluated keys for the scheme with multiple keys but the size of the ciphertext under multiple keys also has a relationship with the number of associated parties.

3. Preliminary

3.1. Learning with Errors, SIVP, and GapSVP. Regev firstly introduced the Learning with Errors (LWE) problem in 2005 and showed that the hardness of LWE can be reduced quantum to the lattice hard problems. Then, Peikert introduced an efficient classical reduction between LWE and the lattice intractable problems. The details are given below.

Definition 1. (Learning with Errors). Let λ be the security parameter, let $n = n(\lambda)$ be an integer dimension of a lattice, let $q = q(\lambda) \geq 2$ be an integer, and let $\chi = \chi(\lambda)$ be an error distribution over $\mathbb{Z}$.

(i) (Searchable LWE) Sample $\mathbf{s} \longleftarrow \mathbb{Z}_q^n$ uniformly and then draw $\mathbf{a}_i \longleftarrow \mathbb{Z}_q^n$ uniformly, $e_i \longleftarrow \chi$. Set $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i$. The searchable LWE is to find $\mathbf{s}$, given $m = m(\lambda)$ samples $\{(\mathbf{a}_i, b_i)\}_{i=1}^m$, called $\mathrm{LWE}_{n,m,q,\chi}$.

(ii) (Decision LWE) The decision LWE, denoted as $\mathrm{LWE}_{n,q,\chi}$, is to distinguish two distributions: The first one is a uniform distribution over $\mathbb{Z}_q^{n+1}$. The second is that one first samples $\mathbf{s} \longleftarrow \mathbb{Z}_q^n$ and then draws $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ by sampling $\mathbf{a}_i \longleftarrow \mathbb{Z}_q^n$ uniformly, $e_i \longleftarrow \chi$, and setting $b_i = \langle \mathbf{a}_i, \mathbf{s}_i \rangle + e_i$.

The Learning with Errors (LWE) assumption is that $\mathrm{LWE}_{n,m,q,\chi}$ ($\mathrm{LWE}_{n,q,\chi}$) is intractable.

Definition 2. ($\mathrm{SIVP}_{\gamma(n)}$). Let Λ be an n−dimension lattice. The $\mathrm{SIVP}_{\gamma(n)}$ problem is to output n linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ such that $\max_i\{v_i\} \leq \gamma(n) \cdot \lambda_n$, where $\lambda_n = \min_r\{r\colon \dim(\mathrm{span}(B(0,r) \cap \Lambda)) \geq n\}$.

Definition 3. ($\mathrm{GapSVP}_{\gamma(n)}$). Let Λ be an n−dimension lattice and let d be a real number. $\mathrm{GapSVP}_{\gamma(n)}$ is to distinguish whether $\lambda_1 < d$ or $\lambda_1 \geq \gamma(n) \cdot d$, where λ_1 is the length of the shortest vector in Λ.

Definition 4. (B-bounded distributions). A distribution ensemble $\{\chi_n\}_{n\in\mathbb{N}}$ over the integers is called B-bounded distribution if

$$\Pr_{e \longleftarrow \chi_n}[|e| > B] = \mathrm{negl}(n). \tag{1}$$

Theorem 1. *Let $q = q(n)$ be either a prime power or a product of small (size poly(n)) distinct primes, $B \geq \omega(\log n) \cdot \sqrt{n}$, and χ is an efficient sampleable B-bounded distribution. If there exists an efficient algorithm solving the $LWE_{n,q,\chi}$ problem, then*

There is an efficient quantum algorithm for $GapSVP_{O(nq/B)}$ on any n-dimension lattice

There is an efficient classical algorithm that solves $GapSVP_{O(nq/B)}$ on any n-dimension lattice

In both cases, if one also considers solving $LWE_{n,q,\chi}$ with subpolynomial advantage, then request $B \geq \widetilde{O}(n)$ and $\gamma(n) \geq \widetilde{O}(n^{1.5}q/B)$.

3.2. Fully Homomorphic Encryption. A fully homomorphic encryption is a tuple of algorithms (**Gen**, **Enc**, **Dec**, **Eval**) described as follows:

$(\mathrm{pk}, \mathrm{sk}, \mathrm{evk}) \longleftarrow \mathbf{Gen}(1^\lambda)$: on the security parameter λ, output a public key pk, a secret key sk, and a public evaluation key evk.

$c \longleftarrow \mathbf{Enc}(\mathrm{pk}, \mu)$: encrypt a message μ from the plaintext space and output a ciphertext c.

$\mu \longleftarrow \mathbf{De\ c}(\mathrm{sk}, c)$: decrypt a valid ciphertext c and output a corresponding message μ; otherwise, output a special symbol $\perp$.

$c_f \longleftarrow \mathbf{Eval}(\mathrm{evk}, f, c_1, \ldots, c_l)$: input the public evaluation key evk, a function f, and a sequence of ciphertexts $c_1, \ldots, c_l$ which are responding to the sequence of plaintexts $\mu_1, \ldots, \mu_l$; output a valid ciphertext c_f responding to the message $f(\mu_1, \ldots, \mu_l)$.

We say that a scheme $\Pi = (\mathbf{Gen}, \mathbf{Enc}, \mathbf{De\ c}, \mathbf{Eval})$ is fully homomorphic if it satisfies the following properties:

Homomorphism: denote a class of all arithmetic circuits over GF(2) as $\mathbb{C}$. If for arbitrary circuit $f \in \mathbb{C}$, the following inequation holds:

$$\Pr\left[\mathbf{Dec}\left(\mathrm{sk}, \mathbf{Eval}\left(\mathrm{evk}, f, c_1, \ldots, c_l\right)\right) \neq f\left(\mu_1, \ldots, \mu_l\right)\right] = \mathrm{negl}(\lambda). \tag{2}$$

Compactness: if there exists a polynomial $p = \mathrm{poly}(\lambda)$, it holds that the output length of **Eval** is at most $p(\lambda)$ bits without relation to the function f or the numbers of inputs.

3.3. Multikey Fully Homomorphic Encryption

Definition 5. (multikey FHE). A multikey FHE is a tuple of algorithms (**Setup**, **Keygen**, **Encrypt**, **Expand**, **Eval**, **Decrypt**) described as follows:

params $\longleftarrow$ **Setup** $(1^\lambda, 1^d)$: on the security parameter λ and the circuit depth d, the setup algorithm outputs the system parameters params. We assume that all the other algorithms take params as an input implicitly.

$(\mathrm{sk}, \mathrm{pk}) \longleftarrow$ **Keygen** (params): generate secret key sk and public key pk.

$c \longleftarrow$ **Encrypt** (pk, μ): take public key pk and a message μ as an input and output for a ciphertext c.

$\widehat{c} \longleftarrow$ **Expand** $(\mathrm{pk}_1, \ldots, \mathrm{pk}_N, i, c)$: on a sequence of N public keys and a fresh ciphertext c under the i–th key pk_i, it outputs an expanded ciphertext $\widehat{c}$.

$\widehat{c}$: =**Eval** $(\mathrm{params}, \mathscr{C}, (\widehat{c}_1, \ldots, \widehat{c}_l))$: given a Boolean circuit $\mathscr{C}$ of depth $\leq d$ along with l expanded ciphertexts $\widehat{c}_1, \ldots, \widehat{c}_l$, output an evaluated ciphertext $\widehat{c}$.

μ: =**Decrypt** $(\mathrm{params}, (\mathrm{sk}_1, \ldots, \mathrm{sk}_N), \widehat{c})$: take some ciphertext $\widehat{c}$ and a sequence of N secret keys as an input and output a message μ.

The following properties hold:

Semantic Security of Encryption. For any polynomial $d = d(\lambda)$ and any two messages μ_0, μ_1, the distribution $(\mathrm{params}, \mathrm{pk}, \mathbf{Encrypt}(\mathrm{pk}, \mu_0))$ is computationally indistinguishable from the distribution $(\mathrm{params}, \mathrm{pk}, \mathbf{Encrypt}\ (\mathrm{pk}, \mu_1))$, where $\mathrm{params} \longleftarrow \mathbf{Setup}\ (1^\lambda, 1^d)$, $(\mathrm{sk}, \mathrm{pk}) \longleftarrow \mathbf{Keygen}\ (\mathrm{params})$.

Correctness and Compactness. Let params $\longleftarrow$ **Setup** $(1^\lambda, 1^d)$. Consider any sequence of N correctly generated key pairs $\{(\mathrm{pk}_i, \mathrm{sk}_i) \longleftarrow \mathbf{Keygen}(\mathrm{params})\}_{i\in[N]}$ and l–tuple of messages $(\mu_1, \ldots, \mu_l)$. For any sequence of indices $(I_1, \ldots, I_l)$ where each $I_i \in [N]$, let $\{c_i \longleftarrow \mathbf{Encrypt}(\mathrm{pk}_{I_i}, \mu_i)\}_{i\in[l]}$ be encryptions of the messages μ_i under the I_i–th public key and let $\{\widehat{c}_i \longleftarrow \mathbf{Expan\ d}((\mathrm{pk}_1, \ldots, \mathrm{pk}_N), I_i, c_i)\}_{i\in[l]}$ be the corresponding expanded ciphertexts. Let $\mathscr{C}$ be any Boolean circuit of depth $\leq d$ and let $\widehat{c}$: = **Eval**$(\mathscr{C}, (\widehat{c}_1, \ldots, \widehat{c}_l))$ be the evaluated ciphertext. Then the following holds:

Correctness of Expansion. $\forall i \in [l]$, **Decrypt** $((\mathrm{sk}_1, \ldots, \mathrm{sk}_N), \widehat{c}_i) = \mu_i$.

Correctness of Evaluation. **Decrypt** $((\mathrm{sk}_1, \ldots, \mathrm{sk}_N), \widehat{c}) = \mathscr{C}(\mu_1, \ldots, \mu_l)$.

Compactness. There exists a polynomial $p(\cdot)$ such as $|\widehat{c}| \leq p(\lambda, d, N)$. In other words, the size of $\widehat{c}$ should be independent of $\mathscr{C}$ and l but can depend on λ, d, N.

4. A Scheme of Evaluation on Two-Key Ciphertexts for Any Polynomial

In this section, we formally describe our fully homomorphic encryption scheme. At the beginning, we introduce three operations used in the encryption algorithm for slow noise growth. Consider three vectors $\mathbf{a} = (a_0, \ldots, a_{n-1}) \in \mathbb{Z}_q^n$, $\alpha = (\alpha_0, \ldots, \alpha_{N-1}) \in \{0,1\}^N$, and $\beta = (\beta_0, \ldots, \beta_{N-1}) \in \mathbb{Z}_q^N$.

$\mathrm{BitDecomp}(a) = (a_{0,0}, a_{0,1}, \ldots, a_{0,l-1}, a_{1,0}, \ldots, a_{1,l-1}, \ldots, a_{n-1,l-1})$, where $a_{i,j}$ is the j-th element of the binary representation of a_i.

$\mathrm{BitDecomp}^{-1}(\alpha) = (\sum_{i=0}^{l-1} 2^i \alpha_i,\ \sum_{i=l}^{2l-1} 2^{i-l} \alpha_i, \ldots,\ \sum_{i=(n-1)l}^{N-1} 2^{i-(n-1)l} \alpha_i)$, where $\alpha \in \{0,1\}^N$.

$\mathrm{Flatten}(\beta) = \mathrm{BitDecomp}(\mathrm{BitDecomp}^{-1}(\beta))$.

We can see that $\mathrm{BitDecomp}(\cdot)$ expands each element of a vector to its binary representation, $\mathrm{BitDecomp}^{-1}(\cdot)$ can be seen as the inverse operation of $\mathrm{BitDecomp}(\cdot)$, and it makes each l element of a vector to a number in $\mathbb{Z}_q$. These three operations on a matrix are that they are performed on each column vector of the matrix. That is,

$$\mathrm{BitDecomp}(A) = \begin{bmatrix} \mathrm{BitDecomp}(A_0) \\ \vdots \\ \mathrm{BitDecomp}(A_{n-1}) \end{bmatrix}.$$

$\mathrm{BitDecomp}^{-1}(\cdot)$ and $\mathrm{Flatten}(\cdot)$ on a matrix are similar to that.

Our scheme consists of the following probabilistic polynomial time algorithms (**Setup**, **Gen**, **Enc**, **Dec**, **Add**, **Mult**, **Add2**, **Mult2**, and **Dec2**).

Setup $(1^\lambda, 1^L)$: let λ be the security parameter and let L be the max circuit depth. Choose appropriate LWE parameters: modulus $q = q(\lambda, L)$, lattice dimension $n = n(\lambda, L)$, and error distribution $\chi = \chi(\lambda, L)$. Choose parameter $m = O(n \log q)$. Set params $= (q, n, \chi, m)$. Let $l = \lfloor \log q \rfloor + 1$ and $N = n \times l$.

Gen (params): choose randomly $\mathbf{t} \longleftarrow \mathbb{Z}_q^{n-1}$. Choose a random matrix $B \longleftarrow \mathbb{Z}_q^{m\times(n-1)}$ and a vector $\mathbf{e} \longleftarrow \chi^m$. Set $\mathbf{b} = B \cdot \mathbf{t} + \mathbf{e}$. Output the secret key $\mathrm{sk} = \mathbf{s} = (1, -t_1, \ldots, -t_{n-1}) \in \mathbb{Z}_q^n$ and the public key

pk = $A = [\mathbf{b}|B]$. Let $\mathbf{v} = \text{Powerof2}(\mathbf{s})$ (note that $A \cdot \mathbf{s} = \mathbf{e}$.)

Enc (params, pk, μ): choose randomly a matrix $R \longleftarrow \{0,1\}^{N\times m}$. Then encrypt the message μ as follows:

$$C = \text{Flatten}\left(\mu \cdot I_N + \text{BitDecomp}\left(R \cdot A\right)\right) \in \mathbb{Z}_q^{N\times N}. \tag{3}$$

Output the ciphertext C.

Dec (params, sk, C): let $v_i \in ((q/4), (q/2)]$. Output $\mu = \lfloor \langle C_i, \mathbf{v}\rangle / v_i \rceil$.

Add (params, pk, C_1, C_2): to add two ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N\times N}$, output $\text{Flatten}(C_1 + C_2)$.

Mult (params, pk, C_1, C_2): to multiply two ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N\times N}$, output $\text{Flatten}(C_1 \cdot C_2)$.

Mult2 (params, $\text{pk}_1, \text{pk}_2, C_1, C_2$): these two keys are independently generated from the algorithm Gen () on the common parameters. If C_1 is not encrypted under pk_1 or C_2 is not under pk_2, then output $\perp$. Otherwise, output $C_{1,l-1} \cdot C_{2,l-1}^T$.

Add2 (params, $\text{pk}_1, \text{pk}_2, C_1, C_2$): if C_1 is not encrypted under pk_1 or C_2 is not under pk_2, then output $\perp$. Otherwise, set C_1' and C_2' as encryptions of message 1 under pk_1 and pk_2, respectively, and output $C_{1,l-1} \cdot (C_{2,l-1}')^T + C_{1,l-1}' \cdot C_{2,l-1}^T$.

Dec2 (parmas, $C, \text{sk}_1, \text{sk}_2$): if C is an evaluated ciphertext from two ciphertexts under the public keys pk_1 and pk_2, respectively, then the first secret key sk_1 holder computes $\text{tempc}_1 = \mathbf{v}_1^T \cdot C$ and sends it to the sk_2 holder. Similarly, the sk_2 holder computes $\text{tempc}_2 = C \cdot \mathbf{v}_2$ and sends it to the first holder. Then, the sk_1 holder outputs $\mathbf{v}_1^T \cdot \text{tempc}_2$ and the sk_2 holder outputs $\text{tempc}_1 \cdot \mathbf{v}_2$.

The evaluation algorithm **Eval**$(\cdot)$ that performs a depth-L circuit computations on polynomial GSW ciphertexts can be composed of **Add** and **Multi** operations.

5. Evaluation on Two-Key FHE Ciphertexts

5.1. Multiplication. Assume that $\mathbf{C}_1$ is a GSW ciphertext of the message μ_1 under the public key pk_1 and $\mathbf{C}_2$ is that of μ_2 under pk_2. $\mathbf{s}_1$ and $\mathbf{s}_2$ are secret keys corresponding to pk_1 and pk_2, respectively. Set $\mathbf{v}_i = \text{Powerof2}(\mathbf{s}_i)$, $i = 1, 2$. This function Powerof2() transforms a vector $(a_0, \ldots, a_{n-1})$ into a new vector $(a_0, 2a_0, \ldots, 2^{l-1}a_0, \ldots, a_{n-1}, \ldots, 2^{l-1}a_{n-1})$, where l is the length of the binary representation of the modulus q.

Mult2 $(\mathbf{C}_1, \mathbf{C}_2) = \mathbf{c}_1 \cdot \mathbf{c}_2^T$, where $\mathbf{c}_i = \mathbf{C}_i[l-1, \cdot]$, $i = 1, 2$.

De c$2(\mathbf{v}_1, \mathbf{v}_2, \mathbf{C})$: Compute $(\lfloor \mathbf{v}_1^T \cdot \mathbf{C}/2^{l-1} \rceil)^T + c'$, denoted as **tc**, where c' is the $(l-1)$−th row of a ciphertext of a message 0 under the secret key $\mathbf{v}_2$ such that $\lfloor (\langle c', \mathbf{v}_2\rangle / 2^{l-1}) \rceil$. Output $\lfloor \langle \mathbf{tc}, \mathbf{v}_2\rangle / 2^{l-1} \rceil$.

Theorem 2. *Suppose that* $\mathbf{C}_1, \mathbf{C}_2$ *are ciphertexts under the secret keys* $\mathbf{v}_1, \mathbf{v}_2$*, respectively. If* $\mathbf{C}$ *is obtained from* **Mult**$2(\mathbf{C}_1, \mathbf{C}_2)$ *or* **A dd** $2(\mathbf{C}_1, \mathbf{C}_2)$*, the probability of the decryption algorithm* **De c**$2(\cdot)$ *on inputs* $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{C}\}$ *running correctly is negligible. That is, there exists a negligible function* $negl(\cdot)$ *on the security parameter* λ*, satisfying the following inequation:*

$$\Pr\left[\mathbf{Dec}2\left(\mathbf{v}_1, \mathbf{v}_2, \mathbf{C}\right) \neq \mu_1\mu_2 \;\middle|\; \begin{array}{c} \left(\text{pk}_i, \text{sk}_i\right) \longleftarrow \mathbf{GSW.Gen}\left(1^\lambda\right), \quad i = 1, 2; \\ v_i = \text{Powerof}\,2\left(\text{sk}_i\right), \quad i = 1, 2; \\ \mu_1, \mu_2 \longleftarrow \{0,1\}, \mathbf{C}_i = \mathbf{GSW.Enc}\left(\text{pk}_i, \mu_i\right); \\ \mathbf{C} = \mathbf{Mult}2\left(\mathbf{C}_1, \mathbf{C}_2\right), \end{array}\right] \leq \text{negl}\left(\lambda\right). \tag{4}$$

Proof. Obviously, $\mathbf{C}_i \cdot \mathbf{v}_i = \mu_i \cdot \mathbf{v}_i + \mathbf{e}_i$, $i = 1, 2$. We also know that the first l elements of $\mathbf{v}_i$ are $(1, 2, \ldots, 2^{l-1})$. Thus, we can decrypt the ciphertext C_i as $\mu_i = \lfloor \langle C_i[l-1, \cdot], \mathbf{v}_i\rangle / 2^{l-1} \rceil$. Set $\mathbf{c}_i = C_i[l-1, \cdot]$ and $e_i = \mathbf{e}_i[l-1]$, $i = 1, 2$. So, $\langle \mathbf{c}_i, \mathbf{v}_i\rangle = \mu_i \cdot 2^{l-1} + e_i$. Running the first part of the decryption algorithm, we can obtain that $\mathbf{tc} = (\lfloor \mathbf{v}_1^T \cdot \mathbf{C}/2^{l-1} \rceil)^T + c' = (\lfloor \mathbf{v}_1^T \cdot \mathbf{c}_1 \cdot \mathbf{c}_2^T / 2^{l-1} \rceil)^T + c' = (\lfloor \mathbf{c}_2^T \cdot (\mu_1 \cdot 2^{l-1} + e_i)/2^{l-1} \rceil)^T + c' = \mu_1 \mathbf{c}_2 + c'$. After the second part, we can get $\lfloor \langle \mathbf{tc}, \mathbf{v}_2\rangle / 2^{l-1} \rceil = \lfloor \mu_1 \langle \mathbf{c}_2, \mathbf{v}_2\rangle / 2^{l-1} + \langle c', \mathbf{v}_2\rangle / 2^{l-1} \rceil = \mu_1\mu_2$ That is to say, one-time multiplication on two ciphertexts under different secret keys only increases doubly the size of noise because the noise in the intermediate ciphertext **tc** can be viewed as that in an addition to two GSW ciphertexts under the same secret key. Therefore, the ciphertexts obtained from this multiplication algorithm can be decrypted correctly.

We can easily find that one-time multiplication causes a double increase of noise. Thus, scaling up the parameters or appending something auxiliary is undesired. We can directly perform one-time multiplication on two ciphertexts encrypted by two different keys without adjusting anything of the original GSW scheme.

5.2. Addition. We can achieve the *Addition* operation by using the operation *Multiplication*. That is, **A dd** $2(\mathbf{C}_1, \mathbf{C}_2) = \mathbf{Mult}2(\mathbf{C}_1, \overline{\mathbf{C}}_2) + \mathbf{Mult}2(\overline{\mathbf{C}}_1, \mathbf{C}_2)$, where $\overline{\mathbf{C}}_i$ is a ciphertext of message 1 under the secret key $\mathbf{v}_i$, $i = 1, 2$.

According to Theorem 2, after one-time operation *Multiplication* on two ciphertexts under different secret keys, the noise increases doubly. Thus, one-time operation

Addition causes the noise to increase quadruply, which is faster than that of *Multiplication*. It is not hard to find that the ciphertext $\overline{\mathbf{C}}_i$ is unnecessary to preserve the privacy of the plaintext, an exact number 1. Therefore, when constructing $\overline{\mathbf{C}}_i$, we can set the randomness to zero. That is to say, $\overline{\mathbf{C}}_i$ is a special "ciphertext" of the plaintext 1 without noise. This change makes both the operations *Addition* and *Multiplication* have the same growth of the noise.

Note that the **Add2** operation not only supports the input of two ciphertexts under different secret keys but also processes the input of one ciphertext obtained from the **Add2** or **Mult2** procedure and one ciphertext under a single key as well as the input of two former-type ciphertexts. The following are the details of the operation.

Assume that C' is output by the **Add2** or **Mult2** procedure and $\mathbf{C}$ is a ciphertext under the secret key $\mathbf{v}_{b+1}$, where $b \longleftarrow \{0,1\}$. Then $\mathbf{A}\ \mathbf{DD}(\mathbf{C}, C') = \mathbf{Mult2}(\mathbf{C}, \overline{\mathbf{C}}) + C'$, where $\overline{\mathbf{C}}$ is a ciphertext of message 1 under the secret key $\mathbf{v}_{\overline{b+1}}$.

Assume that $\mathbf{C}, C'$ are both output by the **Add2** or **Mult2** procedure. Then, $\mathbf{A}\ \mathbf{DD}(\mathbf{C}, C') = \mathbf{C} + C'$. It also can extend to the case of the input of polynomial ciphertexts from the **Add2** or **Mult2** procedure.

5.3. Evaluation of Any Polynomial Function. Assume that f is an arbitrary polynomial function of $u + v$ inputs, denoted as $x_1, \ldots, x_u, y_1, \ldots, y_v$ and can be rewritten as $f(x_1, \ldots, y_v) = \sum_{i=1}^{w} g_i(x_1, \ldots, x_u) f_i(y_1, \ldots, y_v)$, where g_i and f_i are all L–bounded-depth circuits. Now, we have $u + v$ ciphertexts denoted as $\mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,u}$ under the public key pk_1 and $\mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,v}$ under the public key pk_2. So,

$$\begin{aligned}
&\mathbf{Eval}\left(\mathrm{pk}_1, \mathrm{pk}_2, \mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,u}, \mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,v}, \mathbb{C}_f\right) \\
&= \sum_{i=1}^{w} \mathbf{Mult2}\left(\mathbf{GSW}.\mathbf{Eval}\left(\mathrm{pk}_1, \mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,u}, \mathbb{C}_{g_i}\right), \mathbf{GSW}.\mathbf{Eval}\left(\mathrm{pk}_2, \mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,v}, \mathbb{C}_{f_i}\right)\right) \\
&= \mathbf{ADD}\left(\mathbf{Mult2}\left(C_{g_1}, C_{f_1}\right), \ldots, \mathbf{Mult2}\left(C_{g_w}, C_{f_w}\right)\right),
\end{aligned} \tag{5}$$

where $C_{g_i} = \mathrm{Eval}(\mathrm{pk}_1, C_{1,1}, \ldots, C_{1,u}, g_i)$ and $C_{f_i} = \mathrm{Eval}(\mathrm{pk}_2, C_{2,1}, \ldots, C_{2,v}, f_i)$.

Because g_i and f_i are all L-bounded-depth circuits, $\mathbf{C}_{g_i}$ and $\mathbf{C}_{f_i}$ can be decrypted correctly by the secret keys sk_1 and sk_2, respectively. The operations *Addition* and *Multiplication* both cause the noise to increase linearly. Therefore, the output of the algorithm **Eval** can be decrypted correctly.

6. Analysis

6.1. Correctness. Suppose that $\mathbf{C}_1$ and $\mathbf{C}_2$ are GSW ciphertexts of the plaintexts μ_1 and μ_2 under the public keys pk_1 and pk_2, respectively, so that $\mathbf{C}_i \cdot \mathbf{v}_i = \mu_i \cdot \mathbf{v}_i + \mathbf{small}_i$. These two ciphertexts are possibly fresh GSW ciphertexts and also can be evaluated ciphertexts through a circuit of the depth less than L. Also, a fresh GSW ciphertext has a B–bounded noise, namely, $|\mathbf{small}|_\infty \le B$. The error is bounded by $B(N+1)$ after one homomorphic operation. So, $\mathbf{C}_i$ is a ciphertext with $B(N+1)^L$–bounded noise. From the simple analysis in the front section, the noise in **Mult2** (C_1, C_2) is bounded by $2B(N+1)^L$. Moreover, the noise in the addition of C_1 and C_2 increases linearly as the same as that of the *Multiplication*. So, finishing one-time homomorphic operation on two ciphertexts under different encryption keys, the noise grows up to $2B(N+1)^L$. We only discuss one multiplication operation on two ciphertexts under different keys and polynomial additions on two multiplied ciphertexts. Thus, we assume that there are polynomial additions $w = \mathrm{poly}(\lambda, L)$. The final evaluated ciphertext is bounded $2wB(N+1)^L$. As long as this bound is less than $q/8$, we can decrypt the evaluated ciphertext correctly. We just set $B(N+1)^L \le (1/4)\sqrt{q/w}$. Then, it satisfies $B(N+1)^L \le q/8$ so that GSW ciphertexts can be decrypted correctly. Also, $2Bw(N+1)^L \le q/8$. We can decrypt correctly evaluated ciphertexts through quadratic computations on ciphertexts under two different keys. Now, we conclude this in the following theorem.

Theorem 3. *Given the parameters, a modulus q, a lattice dimension n, a B–bounded distribution χ, and the max circuit-depth L, set $N = n \times (\lfloor \log q \rfloor + t1)$. If $B(N+1)^L \le q/8$, we can decrypt correctly a ciphertext from evaluating a depth-L circuit.*

Theorem 4. *Given the above parameters q, n, χ, B, L, N, and w, that is, the number of additions of a quadratic function, if $B(N+1)^L \le (1/4)\sqrt{q/w}$, we can decrypt a ciphertext, that is, from performing a quadratic computations on fresh GSW ciphertexts under two different keys or evaluated ciphertexts through a depth-L circuit under two different keys.*

6.2. Security. The security of our scheme is dependent on that of the GSW scheme. The inputs of the evaluation algorithm are just the GSW-type ciphertexts, two public keys, and some common parameters without other information of private inputs. Thus, this process reveals no knowledge. In the process of the decryption, the output of the first part is indistinguishable with the uniform distribution because it adds a fresh ciphertext of message 0 and introduces a new noise in the intermediate result. So, we can conclude the following theorem.

Theorem 5. *Assume that the GSW scheme is semantically secure, and so does our scheme. That is, if there exists a*

probabilistic polynomial time adversary $\mathcal{A}$ which can distinguish the distribution of the ciphertext of the GSW scheme and the uniform distribution, we can construct another probabilistic polynomial time adversary $\mathcal{B}$ which can distinguish the distribution of the ciphertext of our scheme and the uniform distribution.

7. Conclusion

In this paper, we present an efficient algorithm of secure computation on ciphertexts under two different keys. In previous works, when evaluating multikey ciphertexts, the size of the ciphertext grows with the number of participants at a more or less linear rate. Although the size of the ciphertext remains invariant, it also provides auxiliary information of the plaintexts. We wanted to evaluate directly on the GSW ciphertexts from two parties without any auxiliary information or interaction between them. We designed a scheme in which one can directly perform any polynomial function on the GSW ciphertexts under two different keys.

Data Availability

No data were used to support this study.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC*, pp. 1219–1234, New York, NY, USA, May 2012.

[2] P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key FHE," in *Proceedings of the Part II 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 735–763, Vienna, Austria, May 2016.

[3] M. Clear and C. McGoldrick, "Multi-identity and multi-key leveled FHE from learning with errors," in *Proceedings of the Part II 35th Annual Cryptology Conference*, pp. 630–656, Santa Barbara, CA, USA, August 2015.

[4] C. Peikert and S. Shiehian, "Multi-key FHE from LWE, revisited," in *Proceedings of the Part II Theory of Cryptography-14th International Conference, TCC 2016-B*, pp. 217–238, Beijing, China, October 2016.

[5] Z. Brakerski and R. Perlman, "Lattice-based fully dynamic multi-key FHE with short ciphertexts," in *Proceedings of the Part I Advances in Cryptology-CRYPTO 2016-36th Annual International Cryptology Conference*, pp. 190–213, Santa Barbara, CA, USA, August 2016.

[6] L. Chen, Z. Zhang, and X. Wang, "Batched multi-hop multi-key FHE from ring-LWE with compact ciphertext extension," in *Proceedings of the Part II Theory of Cryptography-15th International Conference, TCC*, pp. 597–627, Baltimore, MD, USA, November 2017.

[7] W. Chongchitmate and R. Ostrovsky, "Circuit-private multi-key FHE," in *Proceedings of the Part II Public-Key Cryptography-PKC 2017-20th IACR International Conference on Practice and Theory in Public-Key Cryptography*, pp. 241–270, Amsterdam, The Netherlands, March 2017.

[8] Z. Li, C. Ma, and H. Zhou, "Multi-key FHE for multi-bit messages," *Science China Information Sciences*, vol. 61, no. 2, 2018.

[9] H. Chen, I. Chillotti, and Y. Song, "Multi-key homomophic encryption from TFHE," *IACR Cryptology ePrint Archive*, vol. 116, 2019.

[10] T. Zhou, N. Li, X. Yang, Y. Han, and W. Liu, "Efficient multi-key FHE with short extended ciphertexts and less public parameters," *IACR Cryptology ePrint Archive*, vol. 1054, 2018.

[11] B. Jiang and Y. Zhang, "Privacy-preserving min and k-th min computations with fully homomorphic encryption," in *Proceedings of the 34th IEEE International Performance Computing and Communications Conference, IPCCC*, pp. 1–8, IEEE Computer Society, Nanjing, China, December 2015.

[12] B. Jiang and Y. Zhang, "Securely min and *k*-th min computations with fully homomorphic encryption," *Science China Information Sciences*, vol. 61, no. 5, 2018.

[13] Z. Brakerski, S. Halevi, and A. Polychroniadou, "Four round secure computation without setup," in *Proceedings of the Part I Theory of Cryptography-15th International Conference*, pp. 645–677, Baltimore, MD, USA, November 2017.

[14] G. Craig, S. Amit, and B. Waters, "Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based," in *Proceedings of the Part I Advances in Cryptology-CRYPTO 2013-33rd Annual Cryptology Conference*, Santa Barbara, CA, USA, August 2013.