Dynamic multi-key FHE in asymmetric key setting from LWE

sen Dong [1]

[1]the State Key Laboratory of Public Big Data and College of Computer Science and Technology

October 30, 2023

Abstract

We have improved an article called 'Multi-key FHE from LWE, revisited' in TCC'16 and proposed a Dynamic multi-key FHE in asymmetric key setting from LWE. Can you give me some suggestions for modification? thanks very much!

Dynamic multi-key FHE in asymmetric key setting from LWE

Yuling Chen, Sen Dong, Tao Li, Yilei Wang and Huiyu Zhou

***Abstract*—Multi-key Fully homomorphic encryption (MFHE) schemes allow computation on the encrypted data under different keys. However, traditional multi-key FHE schemes based on Learning with errors (LWE) have the undesirable property that is the number of keys has to be fixed in advance. A dynamic multi-key FHE scheme is the most versatile variant which the information about the participants is not required before key generation. To support further homomorphic computation on extended ciphertexts and ciphertexts encrypted under additional keys, Peikert and Shiehian (TCC '16) proposed a leveled dynamic multi-key FHE scheme. Nevertheless, it introduces the circular-security assumption for the LWE parameters to ensure its security, which provides weaker security to a certain extent. The problem of how to construct a LWE-based dynamic multi-key FHE scheme is still open. To address the above problem, in this work, we present a dynamic multi-key FHE scheme based on the LWE assumption in public key setting. The ciphertext can be extended and performed homomorphic evaluation with the ciphertexts encrypted under additional keys. Compared with current constructions, our proposed method requires fewer "local" memory and the process of ciphertext extension is distributed. Our proposed method provides a new way to extend the ciphertext such that the ciphertext homomorphism computation is more efficient. Our scheme is proven to be secure under standard LWE assumptions without using the circular-security assumption.**

***Index Terms*—multi-key, Fully homomorphic encryption, Learning with errors, public key setting, ciphertext extension, distributed, circular-security, Peikert, Shiehian, dynamic.**

I. Introduction

FULLY homomorphic encryption (FHE) scheme allows arbitrary computation on the encrypted data and fully-homomorphic encryption is one of the holy grails of modern cryptography. Rivest, Adleman and Dertouzos [1] firstly proposed that adopting homomorphic encryption to protect data privacy, and then it became an open problem. Later, more and more researchers show strong interests to this challenge. Particularly, since Gentry made a breakthrough realization and constructed the first FHE scheme based on ideal lattice [2][3], several improved variants have been proposed [4][5][6][7][8][9][10][11]. Based on the circuit depth of homomorphic evaluation, fully homomorphic encryption schemes can be divided into two categories: pure fully homomorphic encryption and leveled fully homomorphic encryption. A "pure" FHE scheme allows the circuits of unlimited depths to be evaluated, while a leveled FHE scheme allows an evaluator to evaluate the circuit of the limited depth L , and the parameters of the scheme depends on L. Although a leveled FHE scheme may not meet the requirements of an arbitrary depth circuit, a leveled FHE scheme using a polynomial depth circuit is more efficient in practical applications.

Gentry, Sahai and Waters [12] presented a simpler and more elegant leveled FHE scheme using an approximate eigenvector method called GSW13. The security of GSW13 is based on the Learning with Errors (LWE) problem, introduced by Regev in 2005 [13]. A multi-key FHE scheme is more practical than that of single-key. To overcome the limitation of the single-key FHE schemes, Lopez-Alt et al. [14] developed a multi-key FHE scheme based on a variant of the NTRU cryptosystem. However, its security is based on a new and somewhat nonstandard assumption [15], which is not the commonly seemed LWE problem. Later, Clear and McGoldrick [16] proposed a LWE-based multi-key FHE scheme based on the variant of GSW13. Subsequently, Mukherjee and Wichs [17] proposed a new multi-key FHE scheme based on the Clear-McGoldrick work and constructed a two-round MPC protocol upon their work. However, it is known that the construction of these two works and the other variants [18][19][20][21] is only *static* (i.e., single-hop for keys), which means that no further homomorphic computation can be carried out on the evaluated ciphertexts when the evaluation is completed. Specifically, in addition to ensuring the security of encryption scheme based on mathematical theory, the encrypter usually updates the secret keys to prevent them from being leaked from inside. Supposing there is a fresh ciphertext that encrypted under the additional keys when the update operation occurs, the homomorphic computation of the fresh ciphertext and the processed ciphertexts (extended or evaluated) cannot be excuted properly. Therefore, compared with the *static* multi-key FHE scheme, the dynamic multi-key FHE scheme is more practical and available in applications.

This work was supported in part by the National Natural Science Foundation of China under Grant 61962009, in part by the Major Scientific and Technological Special Project of Guizhou Province under Grant 20183001, in part by the Science and Technology Support Plan of Guizhou Province under Grant [2020]2Y011 and in part by the Foundation of Guangxi Key Laboratory of Cryptography and Information Security under Grant GCIS202118.

Yuling Chen is with the State Key Laboratory of Public Big Data and College of Computer Science and Technology, Guizhou University, Guiyang, 550025, China (e-mail: ylchen3@gzu.edu.cn).

Sen Dong is with the the State Key Laboratory of Public Big Data and College of Computer Science and Technology, Guizhou University, Guiyang, 550025, China, also with the Guangxi Key Laboratory of Cryptography and Information Security, Guilin University Of Electronic Technology, Guilin, 541004, China (e-mail: midmountain@sina.com).

Tao Li is with the State Key Laboratory of Public Big Data and College of Computer Science and Technology, Guizhou University, Guiyang, 550025, China (e-mail: litao_2019@qfnu.edu.cn).

Yilei Wang is with the school of Qufu Normal University, Qufu, 273165, China (e-mail: wang_yilei2019@qfnu.edu.cn).

Huiyu Zhou is with the school of Informatics, University of Leicester, Leicester LE1 7RH, United Kingdom (e-mail: hz143@leicester.ac.uk).

Dynamic multi-key FHE. Compared to *static* multi-key FHE schemes, a dynamic (i.e. multi-hop) multi-key scheme should satify following properties:

1) It allows one to execute the homomorphic evaluation on the ciphertexts encrypted under multiple keys.

2) It can extend a (fresh, extended or evaluated) ciphertext to concatenation keys including additional keys.

3) It Supports the resulting ciphertext to perform the further homomorphic computation with the ciphertexts encrypted under additional keys.

Currently, there also exist many schemes about multi-hop and multi-key FHE. For example, Brakerski and Perlman [22] construct a (unbounded) dynamic multi-key FHE scheme and focus on minimizing the size of ciphertexts (Note that the ciphertexts are LWE vectors). However, it has a restriction of performing an expensive bootstrapping technique in the process of extending ciphertexts and the homomorphic multiplication/NAND operation, which results the encrypted secret keys much larger. Later, Peikert and Shiehian put forward a (leveled) dynamic multi-key FHE scheme [23], which can also be an unbound dynamic multi-key FHE scheme using the bootstrapping technique. Besides, the ciphertexts grow quadratically in the number of the associated keys, requiring more "local" memory. In addition, it uses the circular-security assumption for the LWE parameters, thereby providing weaker security. Recently, Biswas and Dutta [24] proposed a LWE-based construction of a dynamic multi-key FHE scheme based on Peikert and Shiehian's work without the circular-security assumption. Moreover, their scheme gave a different ciphertext structure. However, the special structure of the extended ciphertexts can be decrypted without the participation of the key holder of the additional keys.

As we mentioned above, the dynamic multi-key FHE scheme is more desirable. The existing schemes, however, extend the ciphertexts on the cloud, which requires the cloud to have higher computing capabilities. This will cause the ciphertexts provider to pay more to the cloud service provider. Specifically, in traditional dynamic multi-key FHE schemes, ciphertext extension is executed on the cloud, which means that the cloud not only needs to do homomorphic computation on ciphertexts, but also needs to extend the ciphertexts to additional keys. If there is a dynamic multi-key FHE scheme that all the secret key holders can take participate in for the extension of ciphertexts (all participants complete this work interactively) while ensuring the security of the scheme. Although it may bring extra overhead in communication, it can effectively reduce the work of the cloud, so that the resources are more concentrated on homomorphic computation. When the participants fix their computing resources and costs, it will reduce the costs paid by the participants to the cloud service providers. Obviously, it also make the ciphertext extension faster since the ciphertext extension is distributed to multiple participants instead of only excuting on the cloud individually. In summary, existing dynamic multi-key FHE schemes mainly focus on basic functionality of ciphertexts extension without considering other costs. The dynamic multi-key FHE schemes that only perform homomorphic computation on the cloud will be more desirable.

A. Technical Overview

The **scheme #2** in [23] is a (leveled) dynamic multi-key FHE scheme in the symmetric key setting. The fresh ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{n\times m}$ is a GSW ciphertext encrypted under secret key $\boldsymbol{t} \in \chi^n$ and the extended ciphertexts also are GSW ciphertexts with no extra components, so the standard multiplication/NAND homomorphic operations can be performed normally.

Suppose there exists a (fresh, evaluated or extended) ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{nk\times nkl}$ that has been encrypted under a concatenation key $\boldsymbol{t} \in \chi^{nk}$. In order to preserve the GSW relation for the concatenation of the secret keys when the additional key $\boldsymbol{t}^* \in \chi^n$ occurs, they extend the ciphertext to another GSW ciphertext

$$\hat{\boldsymbol{C}} = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{X} \\ \boldsymbol{O} & \boldsymbol{X}^* \end{bmatrix} \in \mathbb{Z}_q^{n(k+1)\times n(k+1)l}$$

that is encrypted under the new concatenation key $(\boldsymbol{t}, \boldsymbol{t}^*) \in \chi^{n(k+1)}$ for the same message. Although the extended ciphertext has no extra components, other public parameters are needed in the process of ciphertext extension and the security of these public parameters are based on the circular-security assumption for LWE. Essentially, in order to ensure the invisibility of the plaintext u to other encryption participants, there also exists an extra "junk" term $\boldsymbol{b}\cdot(\boldsymbol{I}_k \otimes \boldsymbol{R})$ where $\boldsymbol{R} \leftarrow \{0,1\}^{m\times n^2 l}$ is a uniformly random matrix, unlike the one introduced by decrypting a ciphertext with a wrong secret key in the other works. Particularly, the extra "junk" term is also included in the public parameters and will be cancelled when decrypting the ciphertext with the new concatenation key.

We pointed out that, however, their encryption scheme is in symmetric key setting in the above scheme. Although the setting of public parameters will result in a smaller ciphertext, the public parameters require more "local" memory. Besides, in a symmetric encryption scheme, if the key is hijacked by an adversary, the scheme is no longer secure. On the contrary, in asymmetric encryption scheme, even if the public key is obtained, the adversary cannot decrypt the ciphertext to obtain any information. Therefore, compared to the symmetric encryption scheme, the encryption scheme in public-key setting is more secure. In addition, they use circular-security assumptions to ensure the security of public parameters. Note that the circular -security is a strong assumption which makes the scheme weaker from the security point of view.

B. Our Contributions

To overcome the above difficulties, we make the contributions as follows:

1) Public Key Setting: We put forward a dynamic multi-key FHE scheme without using a reference matrix. Instead of using the circular-security assumption to ensure the security of the public parameters, we use the product of a public key and a uniformly random matrix to hide the secret key. Besides, our scheme works in the public-key setting, rather than the symmetric key setting.

2) Smaller Public Parameters: In the process of ciphertext extension, only one public parameter is required, and the public matrix is relatively small. Therefore, less "local" memory is required for public parameters. The comparison of our scheme with the the current learning with errors (LWE) based multi-key FHE schemes is provided in Table I.

3) Computational Complexity Is More Efficiency: The extended ciphertext structure in our scheme is

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{X} \\ \boldsymbol{O} & \boldsymbol{Y} \end{bmatrix} \in \mathbb{Z}_q^{nk \times nkl}.$$

We first proposed a concept called distributed ciphertext extension. The computation and ciphertext extension that are originally done independently by the cloud are now jointly participated by all parties and completed by each participant interactively. This improvement reduces the work of the cloud and improves the efficiency of ciphertext extension.

Except for these contributions described above, we also retain some of the properties in our scheme. Our scheme is a dynamic FHE scheme in which the ciphertexts can be homomorphically computed under several keys and the results are available in further computation under additional keys. Moreover, our scheme is also suitable for dynamic on-the-fly MPC. In addition, our scheme can support unbounded homomorphic computations for any polynomial number of keys using a "bootstrap"method.

II. Preliminaries

A. Notion

Negligible Function. For a parameter λ and a positive polynomial $poly(\lambda)$, if there exists a function $\mathcal{F}(\lambda) = 1/poly(\lambda)$, we call $\mathcal{F}$ is negligible, written as $negl(\lambda)$.

Matrices, Vectors and Sets. Matrices and vectors are represented by bold uppercase letters (e.g., $\boldsymbol{A}$) and lower-case bold letters (e.g., $\boldsymbol{a}$) respectively, the i^{th} element of vectors by the notation of $\boldsymbol{a}[i]$.$||\boldsymbol{a}||_\infty$ and $||\boldsymbol{a}||_1$ represent the maximum norm and 1-norm respectively where $||\boldsymbol{a}||_\infty = max_i |\boldsymbol{a}_i|$ and $||\boldsymbol{a}||_1 = \sum_i \boldsymbol{a}_i$. The inner product of two vectors $\boldsymbol{a}, \boldsymbol{b}$ for some dimension n is written as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$. We define $[k] \stackrel{\text{def}}{=} \{1, 2, ..., k\}$ for any non-negative integer k.

Distributions. If χ is sampled uniformly or normally from the probability distribution $\mathcal{D}$, we denote by $\chi \stackrel{\$}{\leftarrow} \mathcal{D}$. For the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ if they are computationally indistinguishable or statistically indistinguishable, we denote by $\mathcal{D}_0 \underset{c}{\approx} \mathcal{D}_1$ and $\mathcal{D}_0 \underset{s}{\approx} \mathcal{D}_1$ respectively.

Definition 1 (B-bounded distributions (**Definition 2** [12])). A distribution ensemble $\{\mathcal{D}_n\}_{n \in N}$ supported over the integers, is called B-bounded if

$$Pr_{e \stackrel{\$}{\leftarrow} \mathcal{D}_n}[|e > B|] = negl(n).$$

Kronecker Products. Given an $m_1 \times n_1$ matrix $\boldsymbol{A}$ and an $m_2 \times n_2$ matrix $\boldsymbol{B}$. Kronecker product of the two matrices denoted by $\otimes$ is defined as

$$\boldsymbol{A} \otimes \boldsymbol{B} := (\, \boldsymbol{a}_{1,1}\boldsymbol{B}, \boldsymbol{a}_{1,2}\boldsymbol{B}, ..., \boldsymbol{a}_{1,n_1}\boldsymbol{B}, \boldsymbol{a}_{2,1}\boldsymbol{B}, ..., \boldsymbol{a}_{2,n_1}\boldsymbol{B}, \\ ..., \boldsymbol{a}_{m_1,1}\boldsymbol{B}, ..., \boldsymbol{a}_{m_1,n_1}\boldsymbol{B} \,)$$

where $\boldsymbol{A} \otimes \boldsymbol{B}$ is an $m_1 m_2 \times n_1 n_2$ matrix and the $a_{i,j}$ is the $(i,j)^{th}$ element of $\boldsymbol{A}$. There are some properties of the Kronecker product used throughout this paper as follows:

1) If $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same size:

$$(\boldsymbol{A} + \boldsymbol{B}) \otimes \boldsymbol{C} = \boldsymbol{A} \otimes \boldsymbol{C} + \boldsymbol{B} \otimes \boldsymbol{C}$$

2) If the matrix products $\boldsymbol{AC}$ and $\boldsymbol{BD}$ exist:

$$(\boldsymbol{A} \otimes \boldsymbol{B}) \cdot (\boldsymbol{C} \otimes \boldsymbol{D}) = \boldsymbol{AC} \otimes \boldsymbol{BD}$$

3) for any matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ of compatible dimensions:

$$(\boldsymbol{A} \otimes \boldsymbol{B}) = (\boldsymbol{A} \otimes \boldsymbol{I}_{height\,(\boldsymbol{B})}) \cdot (\boldsymbol{I}_{width(\boldsymbol{A})} \otimes \boldsymbol{B}) \\ = (\boldsymbol{I}_{height(\boldsymbol{A})} \otimes \boldsymbol{B}) \cdot (\boldsymbol{A} \otimes \boldsymbol{I}_{width(\boldsymbol{B})}).$$

B. Learning With Errors

The Learning with Errors (LWE) problem was pointed out by Regev [13]. It has a decisional variant denoted by $DLWE_{n,m,q,\chi}$ will be used in our paper. Given (a polynomial number of) independent samples $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ sampled either from the LWE distribution $A_{S,\chi}$ or the uniform distribution, the $DLWE_{n,m,q,\chi}$ problem is to distinguish which these samples come from. The *DLWE* assumption is that these two distributions are computationally indistinguishable for any *PPT* adversary. It is known that if the discrete Gaussian distribution χ with parameter $\alpha q \geq 2\sqrt{n}$ over $\mathbb{Z}$, the $DLWE_{n,m,q,\chi}$ in the average-case is as hard as the approximation lattices problems in the worst-case with approximation factors of $\widetilde{O}(n/\alpha)$ by the classical or quantum reductions [13][25][26][27].

TABLE I: Comparison of multi-key FHE schemes

	$\|pk\|$	$\|CT\|$	CRM	Dynamic	Assumption	Bootstrap in each level
Clear and McGoldrick	$\widetilde{O}(nL^2)$	$\widetilde{O}(n^2k^2L^2)$	YES	NO	LWE	NO
Brakerski and Perlman	$\widetilde{O}(n^3)$	$\widetilde{O}(nk)$	YES	YES	LWE&Circular security	YES
Mukherjee and Wichs	$\widetilde{O}(nL^2)$	$\widetilde{O}(n^2k^2L^2)$	YES	NO	LWE	YES
Scheme #1 of Peikert and Shiehian	$\widetilde{O}\left(n(K+L)^2\right)$	$\widetilde{O}\left(n^3k(K+L)^4\right)$	YES	YES	LWE	NO
Scheme #2 of Peikert and Shiehian	$\widetilde{O}\left(n^4(K+L)^4\right)$	$\widetilde{O}\left(n^2k^2(K+L)^2\right)$	YES	YES	LWE&Circular security	NO
Ours	$\widetilde{O}\left(n^3(K+L)^2\right)$	$\widetilde{O}\left(n^2k^2(K+L)^2\right)$	NO	YES	LWE	NO

Here k is the actual number of the secret keys associated with the ciphertext, K denotes a designed upper bound on k, L represents the maximum depth of the boolean circuits homomorphically evaluated (without bootstrapping), and n is the dimension of the underlying LWE problem used for security. The $\widetilde{O}$ notation hides the factors of the form $\log poly(n, k, l)$ for some polynomial function . The $|pk|$ and $|CT|$ represent the size of pk and the ciphertext respectively, where all the sizes are in bits. CRM denotes whether or not the common reference matrix is needed in the scheme.

Definition 2 (Decisional $LWE_{n,m,q,\chi}$ problem). Suppose λ is the security parameter. Let $n = n(\lambda)$, $q = q(\lambda) \geq 2$ and the error distribution $\chi = \chi(\lambda)$ over $\mathbb{Z}$. The decisional learning with errors problem is to distinguish the following distributions:

Distribution 0: The i^{th} sample $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ is sampled uniformly from the random distributions where $\boldsymbol{a}_i \xleftarrow{\$} \mathbb{Z}_q^n$ and $b_i \xleftarrow{\$} \mathbb{Z}_q$.

Distribution 1: The i^{th} sample $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ is made up of uniformly sampling $\boldsymbol{a}_i \xleftarrow{\$} \mathbb{Z}_q^n$ and computing $b_i \xleftarrow{\$} \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i$ where $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$ is generated uniformly and $e_i \xleftarrow{\$} \chi$ is sampled from the error distribution.

Definition 3 (Decisional $LWE_{n,m,q,\chi}$assumption). Decisional $LWE_{n,m,q,\chi}$ assumption holds if

$$|Pr[\mathcal{A}(\boldsymbol{a}, b) = 1 : (\boldsymbol{a}, b) \leftarrow Distribution0]| - |Pr[\mathcal{A}(\boldsymbol{a}, b) = 1 : (\boldsymbol{a}, b) \leftarrow Distribution1]| = negl(n)$$

for any *PPT* adversary $\mathcal{A}$.

But in this work, it is convenient to use another form of decisional LWE as Peikert and Shiehian mentioned in their scheme.The $DLWE_{n\text{-}1,m,q,\chi}$ is computationally equivalent to the $DLWE_{n,m,q,\chi}$ problem and the LWE samples in this form are indistinguishable from uniform assuming the hardness of $DLWE_{n,m,q,\chi}$ problem.

C. Gadget Matrix

For convenience, we use the gadget matrix [26] and some definitions in [23] throughout this work.

The gadget matrix is used to decompose the vectors or the matrices(over $\mathbb{Z}_q$) into short vectors or matrices(over $\mathbb{Z}$). The standard gadget vector is

$$\boldsymbol{g} = \left(1, 2, 4, 8, ..., 2^{l-1}\right) \in \mathbb{Z}_q^l$$

where $l = \lceil \log q \rceil$. And $\boldsymbol{g}^{-1} : \mathbb{Z}_q \rightarrow \{0,1\}^l$ is the bit decomposition function, which outputs a binary column vector over $\mathbb{Z}$ consisting of the binary representation if its elements. It is obvious that $\boldsymbol{g} \cdot \boldsymbol{g}^{-1}[\boldsymbol{a}] = \boldsymbol{a}$ and we define

$$[\boldsymbol{a}]\boldsymbol{g}^{-t} \stackrel{\text{def}}{=} \boldsymbol{g}^{-1}[\boldsymbol{a}]^t$$

which outputs a binary row vector and satisfies $[\boldsymbol{a}]\boldsymbol{g}^{-t} \cdot \boldsymbol{g}^t = \boldsymbol{a}$. According to these definitions, it is obvious that

$$(\boldsymbol{I}_n \otimes \boldsymbol{g}) \cdot \left(\boldsymbol{I}_n \otimes \boldsymbol{g}^{-1}\right)[\boldsymbol{A}] = \boldsymbol{A},$$

$$[\boldsymbol{A}]\left(\boldsymbol{I}_n \otimes \boldsymbol{g}^{-t}\right) \cdot (\boldsymbol{I}_n \otimes \boldsymbol{g}^t) = \boldsymbol{A}$$

where the $(\boldsymbol{I}_n \otimes \boldsymbol{g})$ is exactly the n-row gadget matrix $\boldsymbol{G}$ and $\left(\boldsymbol{I}_n \otimes \boldsymbol{g}^{-1}\right)[\cdot]$ is exactly the bit-decomposition operation $\boldsymbol{G}^{-1}$ on height-n matrices or vectors.

D. Cryptographic Definitions

Definition 4. A leveled dynamic multi-key FHE variant of GSW is a tuple of *PPT* algorithm (*Setup, Keygen, Encrypt, Decrypt, Extend, Eval*) having the following properties:

$params \leftarrow Setup\left(1^\lambda, 1^L, 1^k\right)$: Given a security parameter λ, the maximum circuit depth L, a bound k on the number of keys, it outputs the system public parameter *params*.

$(pk, sk) \leftarrow Keygen(params)$: This algorithm is used to generate the public key *pk* and the secret key *sk*.

$\boldsymbol{C} \leftarrow Encrypt(pk, u)$: On input the public key *pk* and a single bit message $u \in \{0,1\}$, it outputs the ciphertext $\boldsymbol{C}$ that encrypts *u* corresponding to *pk*.

$u \leftarrow Decrypt((sk_1, sk_2, ..., sk_k,), \boldsymbol{C})$: Given the concatenation key $SK = (sk_1, sk_2, ..., sk_k,)$ and a (fresh, extended or evaluated) ciphertext under *SK* corresponding to $PK = (pk_1, pk_2, ..., pk_k,)$, it recovers and outputs the message $u \in \{0,1\}$.

$\boldsymbol{C}' \leftarrow Eval(PK, \mathbb{C}, \boldsymbol{C}_1, \boldsymbol{C}_2, ..., \boldsymbol{C}_s)$: Given a boolean circuit $\mathbb{C}$ of maximum depth L along with s (fresh, extended or evaluated) ciphertexts wires, outputs an evaluated ciphertext $\boldsymbol{C}'$ that implicitly contains a reference to each public key associated with $\boldsymbol{C}_i$ where $1 \leq i \leq s$.

$\hat{\boldsymbol{C}} \leftarrow Extend(pk, \boldsymbol{C})$: On the input of a (fresh, extended or evaluated) ciphertext corresponding to the message $u \in \{0,1\}$ under $SK = (sk_1, sk_2, ..., sk_{k-1},)$, and a *PK* consist of the public key *pk* and the public extension matrix, it outputs an extended ciphertext $\hat{\boldsymbol{C}}$ corresponding to the message $u \in \{0,1\}$ under $SK = (sk_1, sk_2, ..., sk_k,)$.

These algorithms should satisfy compactness and correctness functionality properties as follows:

Compactness. We say a dynamic multi-key FHE scheme is compact if the length of $\hat{\boldsymbol{C}}$ is independent of $\boldsymbol{C}$ and s instead of depending polynomially on λ, k and L. In other words, $\left|\hat{\boldsymbol{C}}\right| \leq poly(\lambda, k, L)$ if there exists a polynomial $p(\cdot,\cdot,\cdot)$.

Correctness. A leveled dynamic multi-key FHE scheme is correct if for the security parameter λ, a bound k on the numbers of keys, for a circuit of depth at most L having N input wires and a ciphertext sequences $(\boldsymbol{C}_i)_{i\in[N]}$ corresponding to a same key set S (S is made up of $\left(pk_j, sk_j\right) \leftarrow Keygen(params)$) for each $j \in [k]$, $\boldsymbol{C}_i$ is generated as $\boldsymbol{C}_i \leftarrow Encrypt\left(pk_j, u_j\right)$ where $i \in [N]$, $j \in [k]$ and $u \in \{0,1\}$, the following formula has a probability of $negl(\lambda)$:

$$Pr[\ Decrypt(sk_s, Eval(PK, \mathbb{C}, \boldsymbol{C}_1, \boldsymbol{C}_2, ..., \boldsymbol{C}_s)) \neq \mathbb{C}(u_1, u_1, ..., u_N)\ ] = negl(\lambda),$$

where *Decrypt* is given those secret keys sk_s from a fixed key set S corresponding to the public keys referenced by all the ciphertexts.

GSW Linear combination. This operation takes the GSW ciphertexts $\boldsymbol{C}_{i,j}$ which is the encryption of the individual entries $\boldsymbol{M}[i,j]$ where $\boldsymbol{M} \in \mathbb{Z}_q^{m\times m}$ and a plaintext vector $\boldsymbol{v} \in \mathbb{Z}_q^m$ as inputs. This operation outputs a "pseudo ciphertext"$\boldsymbol{C}_{lc}$ satisfying $\boldsymbol{tC} \approx \boldsymbol{vM}$.

Property 1.(Linear combination, **Property 5.3** [17]) Let $\boldsymbol{M} \in \{0,1\}^{m\times m}$ be a matrix and for $i \in [m]$, $j \in [m]$, let $\boldsymbol{C}_{i,j} \in \mathbb{Z}_q^{n\times m}$ be a $\beta - noise$ GSW encryption of $\boldsymbol{M}[i,j]$ under

a secret key $\boldsymbol{t} \in \mathbb{Z}_q^n$ and $\boldsymbol{v} \in \mathbb{Z}_q^m$ be some vector (not necessarily short). Then there is a polynomial-time deterministic algorithm

$$\boldsymbol{C}_{lc} = GSW.LComb\left(\left(\boldsymbol{C}_{1,1}, ..., \boldsymbol{C}_{m,m}\right), \boldsymbol{v}\right)$$

which outputs $\boldsymbol{C}_{lc} \in \mathbb{Z}_q^{n \times m}$ such that $\boldsymbol{t}\boldsymbol{C}_{lc} = \boldsymbol{v}\boldsymbol{M} + \boldsymbol{e}$ where $|\boldsymbol{e}|_\infty \leq m^3\beta$.

Implementation. The algorithm *GSW.LComb* $\left(\left(\boldsymbol{C}_{1,1}, ..., \boldsymbol{C}_{i,j}, ..., \boldsymbol{C}_{m,m}\right), \boldsymbol{v}\right)$ is implemented as follows:

For each $i \in [m]$, $j \in [m]$ defines a matrix $\boldsymbol{Z}_{i,j} \in \mathbb{Z}_q^{n \times m}$ as follows:

$$\boldsymbol{Z}_{i,j}[a,b] := \begin{cases} \boldsymbol{v}[i], & \text{when } a = n \text{ and } b = j \\ 0, & \text{otherwise} \end{cases}$$

and then output $\boldsymbol{C}_{lc} \in \mathbb{Z}_q^{n \times m}$ where

$$\boldsymbol{C}_{lc} = \sum_{i=1,j=1}^{m,m} \boldsymbol{C}_{i,j}\boldsymbol{G}^{-1}\left(\boldsymbol{Z}_{i,j}\right).$$

III. Concrete construction

In this section, we show how to construct a multi-hop, multi-key FHE scheme *MFHE* = (*Setup, Keygen, Encrypt, Decrypt, Extend, Eval*) from LWE for single bit message in the public-key setting.

A. Basic Encryption Scheme

params $\leftarrow$ *Setup* $\left(1^\lambda, 1^L, 1^k\right)$: On input of a security parameter $\lambda \in \mathbb{N}$, a maximum depth $L \in \mathbb{N}$ of the evaluated circuit and a bound k on the number of keys, it chooses the lattice dimension parameters $n = n(\lambda L)$, a modulus q and $B_\chi - bounded$ for $B_\chi = \Theta(n)$ standard discrete Gaussian error distribution where $\chi = \chi(\lambda L)$ with parameter $2\sqrt{n}$. We will explain later how to choose the modulus q in order to get correct decryption. Then it sets $m = nl$ where $l = \lceil \log q \rceil$ and chooses a random matrix $\boldsymbol{B} \in \mathbb{Z}_q^{n-1 \times m}$. It outputs the $params = (q, n, m, \chi, B_\chi, \boldsymbol{B})$. We stress that all the other algorithms implicitly get params as input even if we usually do not write this explicitly.

$(PK, sk) \leftarrow$ *Keygen* $(params)$: We separate *Keygen* to two sub-algorithms to generate secret key and pubic key along with a public extension matrix respectively:

1) Sample $\bar{\boldsymbol{t}} \xleftarrow{\$} \chi^{n-1}$ randomly from the standard discrete Gaussian error distribution and then output $sk = \boldsymbol{t} = (-\bar{\boldsymbol{t}}, 1) \in \chi^n$.

2) Sample $\boldsymbol{e} \xleftarrow{\$} \chi^m$ and compute $\boldsymbol{b} = \bar{\boldsymbol{t}}\boldsymbol{B} + \boldsymbol{e} \in \mathbb{Z}_q^m$. Then set $pk = \boldsymbol{A} = \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{b} \end{bmatrix}$, we can observe that $\boldsymbol{t} \cdot \boldsymbol{A} = \boldsymbol{e}$. In addition, unlike the current dynamic multi-key schemes, it also sets $\boldsymbol{P} = (\boldsymbol{\omega} \otimes \boldsymbol{A}\boldsymbol{R}) + (\boldsymbol{I}_n \otimes \boldsymbol{t} \otimes \boldsymbol{g})$ as the public extension key. It finally outputs the public key $PK = (\boldsymbol{A}, \boldsymbol{P})$.

$\boldsymbol{C} \leftarrow$ *Encrypt* (pk, u) : To encrypt a message $u \in \{0,1\}$, sample a uniformly random matrix $\boldsymbol{R} \xleftarrow{\$} \{0,1\}^{m \times m}$ as the randomness. Then output the encryption of message u as $\boldsymbol{C} \in \mathbb{Z}_q^{n \times m}$ where

$$\boldsymbol{C} = \boldsymbol{A}\boldsymbol{R} + u\boldsymbol{G}.$$

It is obvious that $\boldsymbol{C}$ is simply a GSW ciphertext encrypting u under secret key $\boldsymbol{t}$:

$$\begin{aligned} \boldsymbol{t}\boldsymbol{C} &= \boldsymbol{t}\boldsymbol{A}\boldsymbol{R} + u\boldsymbol{t}\boldsymbol{G} \\ &= \boldsymbol{e}\boldsymbol{R} + u\boldsymbol{t}\boldsymbol{G} \\ &= \boldsymbol{e}' + u\left(\boldsymbol{t} \otimes \boldsymbol{g}\right). \end{aligned}$$

Observe that a fresh ciphertext $\boldsymbol{C}$ is generated by encrypting the message u with the pk $\boldsymbol{A}$ with corresponding $sk = \boldsymbol{t}$. Recall that $\boldsymbol{t} \cdot \boldsymbol{A} = \boldsymbol{e}$ and $||\boldsymbol{e}||_\infty \leq B_\chi$. It is obvious that $\boldsymbol{t}\boldsymbol{C} = \boldsymbol{e}' + u\boldsymbol{t}\boldsymbol{G}$ where $\boldsymbol{e}' = \boldsymbol{e}\boldsymbol{R}$ which implies $||\boldsymbol{e}'|| \leq mB_\chi$. Therefore, the ciphertext $\boldsymbol{C}$ is $mB_\chi - noisy$ encryption of u under secret key $\boldsymbol{t}$. We define this value as initial noise $\beta := mB_\chi$.

$\hat{\boldsymbol{C}} \leftarrow Extend\left(params, PK, \boldsymbol{C}\right)$: This algorithm takes as input a (fresh, evaluated or extended) ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{nk \times nkl}$ that encrypts u under a concatenation key $sk = \boldsymbol{t} = \left(\boldsymbol{t}_1, \boldsymbol{t}_2, ..., \boldsymbol{t}_k \in \chi^{nk}\right)$ where $\boldsymbol{t}_i \in \chi^n$ for $i \in [k]$. Note that $\boldsymbol{C}$ satisfies $\boldsymbol{t}\boldsymbol{C} = \boldsymbol{E}_{\boldsymbol{C}} + u\left(\boldsymbol{t} \otimes \boldsymbol{g}\right)$ with noise $||\boldsymbol{E}_{\boldsymbol{C}}||_\infty$. This algorithm extends the ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{nk \times nkl}$ to $\hat{\boldsymbol{C}} \in \mathbb{Z}_q^{n(k+1) \times n(k+1)l}$ under an additional secret key $sk = \boldsymbol{t}^* \in \chi^n$ corresponding to the public extension matrix $\boldsymbol{P}$. The extended ciphertext $\hat{\boldsymbol{C}}$ is a GSW encryption of the message $n \in \{0,1\}$ under the new extended secret key $sk = \hat{\boldsymbol{t}} = (\boldsymbol{t}, \boldsymbol{t}^*) \in \chi^{n(k+1)}$. We will give a detailed description of the specific implementation steps below.

$u \leftarrow Decrypt\left(SK, \boldsymbol{C}\right)$: The ciphertexts in our scheme is GSW ciphertexts, so that this is an ordinary GSW decryption. For simplicity, we just describe how GSW decryption works here. To decrypt a message u , the decrypter lets $\boldsymbol{\omega} = (0, 0, ..., q/2)$, then computes $v = \boldsymbol{t}\boldsymbol{C}\boldsymbol{G}^{-1}\left(\boldsymbol{\omega}^T\right) = \boldsymbol{E}_{\boldsymbol{C}}\left(\boldsymbol{\omega}^T\right) + u\lceil q/2 \rceil$ where $\boldsymbol{E}_{\boldsymbol{C}}\left(\boldsymbol{\omega}^T\right)$ is the noise term whose bound is $m\beta$. If v is closer to 0 as opposed to 0, the decryption result is 0; otherwise, the result is 1.

$\boldsymbol{C} \leftarrow Eval\left(PK, \mathbb{C}, \boldsymbol{C}_1, \boldsymbol{C}_2, ..., \boldsymbol{C}_s\right)$: The ciphertexts above are just GSW ciphertexts (with no extra information), so homomorphic addition and multiplication work as the GSW scheme:

$$\begin{aligned} &GSW.Add\left(\boldsymbol{C}_1, \boldsymbol{C}_2\right) : \text{Output } \boldsymbol{C}_1 + \boldsymbol{C}_2 \in \in \mathbb{Z}_q^{nk \times nkl} \\ &GSW.Mult\left(\boldsymbol{C}_1, \boldsymbol{C}_2\right) : \text{Output } \boldsymbol{C}_1\boldsymbol{G}^{-1}\left(\boldsymbol{C}_2\right) \in \mathbb{Z}_q^{nk \times nkl} \\ &GSW.NAND\left(\boldsymbol{C}_1, \boldsymbol{C}_2\right) : \text{Output } \boldsymbol{G} - \boldsymbol{C}_1\boldsymbol{G}^{-1}\left(\boldsymbol{C}_2\right) \in \mathbb{Z}_q^{nk \times nkl} \end{aligned}$$

Therefore, we only need to show how to extend the ciphertexts to additional keys.

B. Ciphertext Extension

As we have mentioned above, a dynamic (e.g., multi-hop for keys), multi-key FHE scheme supports an arbitrary ciphertext of $\mathbb{C}\left(u_1, u_2, ..., u_k\right)$ under the concatenation key extending to an additional key. Briefly speaking, a dynamic multi-key FHE scheme must be able to support the extension of the evaluated ciphertext and/or the extended ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{nk \times nkl}$ and satisfy the GSW decryption, namely,

$$\begin{aligned} \hat{\boldsymbol{t}}\hat{\boldsymbol{C}} &= \left(\boldsymbol{t}, \boldsymbol{t}^*\right)\hat{\boldsymbol{C}} \\ &= \boldsymbol{E}_{\hat{\boldsymbol{C}}} + u\left(\boldsymbol{t}, \boldsymbol{t}^*\right)\boldsymbol{G} \\ &= \boldsymbol{E}_{\hat{\boldsymbol{C}}} + u\left(\left(\boldsymbol{t}, \boldsymbol{t}^*\right) \otimes \boldsymbol{g}\right) \end{aligned}$$

where $(\boldsymbol{t}_1, \boldsymbol{t}_2, ..., \boldsymbol{t}_k) \in \chi^{nk}$ and $\boldsymbol{t}^* \in \chi^n$ represents the concatenation of k individual secret keys $\boldsymbol{t}_i \in \chi^n$ and the additional key respectively, and $u \in \{0,1\}$ is a plaintext corresponding to the evaluated or the extended ciphertext.

In order to achieve this goal, we extend $\boldsymbol{C} \in \mathbb{Z}_q^{nk \times nkl}$ to an additional secret key $\boldsymbol{t}^* \in \chi^n$ for which we know the associated public matrix $\boldsymbol{P}^*$ and generate a ciphertext $\hat{\boldsymbol{C}} \in \mathbb{Z}_q^{n(k+1) \times n(k+1)l}$ that encrypts u under $\hat{\boldsymbol{t}} = (\boldsymbol{t}, \boldsymbol{t}^*) \in \chi^{n(k+1)}$. Besides, the ciphertext $\hat{\boldsymbol{C}}$ is the GSW construction.

We generate the extension ciphertext as

$$\hat{\boldsymbol{C}} = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{X} \\ \boldsymbol{O} & \boldsymbol{Y} \end{bmatrix} \in \mathbb{Z}_q^{n(k+1) \times n(k+1)l}$$

where the extension ciphertext is the same as [23] in structure, but we must declare here that the components are different because of our design, which is why we claim the homomorphism evaluation of extended ciphertexts is more efficient than their work.

Notice that by construction,

$$\hat{\boldsymbol{t}}\hat{\boldsymbol{C}} = (\boldsymbol{t}\boldsymbol{C}\ \boldsymbol{t}\boldsymbol{X} + \boldsymbol{t}^*\boldsymbol{Y}) + \boldsymbol{E}_{\hat{\boldsymbol{C}}}.$$

Below we show how to construct the $\boldsymbol{X}$ and $\boldsymbol{Y}$ to satisfy

$$\boldsymbol{t}\boldsymbol{X} + \boldsymbol{t}^*\boldsymbol{Y} \approx u\,(\boldsymbol{t}^* \otimes \boldsymbol{g})$$

so that

$$\begin{aligned} \hat{\boldsymbol{t}}\hat{\boldsymbol{C}} &= (u\,(\boldsymbol{t} \otimes \boldsymbol{g})\ \ u\,(\boldsymbol{t}^* \otimes \boldsymbol{g})) + \boldsymbol{E}_{\hat{\boldsymbol{C}}} \\ &= (u\,(\boldsymbol{t}, \boldsymbol{t}^*)) \otimes \boldsymbol{g} + \boldsymbol{E}_{\hat{\boldsymbol{C}}} \\ &= u\hat{\boldsymbol{t}}\boldsymbol{G} + \boldsymbol{E}_{\hat{\boldsymbol{C}}} \end{aligned}$$

satisfy our proposition.

Particularly, we stress that the construction of $\boldsymbol{X}$ and $\boldsymbol{Y}$ in our scheme is inspired by previous works [16][17][23]. In detail, let $\boldsymbol{Y}'$ be the encryption of plaintext u under $\boldsymbol{t}^*$. But in order to avoid the key holder of $\boldsymbol{t}^*$ being able to recover u individually, we use an unrelated matrix to "blind" $\boldsymbol{Y}'$ to get the final $\boldsymbol{Y}$ in our scheme. And then we let $\boldsymbol{X}$ be the encryption of the unrelated matrix under $\boldsymbol{t} \in \chi^n$, so whether the extended ciphertext can be decrypted or not will be decided by the key holder of $\boldsymbol{t}$ and $\boldsymbol{t}^*$ where $\boldsymbol{t}$ is a concatenation key composed of k individual secret keys.

We construct $\boldsymbol{X}$ and $\boldsymbol{Y}$ in two steps:

Constructing Y. Suppose that there exists a ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{nk \times nkl}$ that satisfies

$$\boldsymbol{t}\boldsymbol{C} = \boldsymbol{E}_{\boldsymbol{C}} + u\boldsymbol{t}\boldsymbol{G} \tag{1}$$

where $\boldsymbol{t}$ represents the concatenation key. Then we want to find an equation satisfies

$$\begin{aligned} \boldsymbol{t}\boldsymbol{X} &\approx -''\boldsymbol{Blind}'' \\ \boldsymbol{t}^*\boldsymbol{Y} &\approx u\boldsymbol{t}^*\boldsymbol{G} +'' \boldsymbol{Blind}'' \end{aligned}$$

so that

$$\begin{aligned} \hat{\boldsymbol{t}}\hat{\boldsymbol{C}} &= (\boldsymbol{t}, \boldsymbol{t}^*) \begin{bmatrix} \boldsymbol{C} & \boldsymbol{X} \\ \boldsymbol{O} & \boldsymbol{Y} \end{bmatrix} \\ &= (\boldsymbol{t}\boldsymbol{C}\ \boldsymbol{t}\boldsymbol{X} + \boldsymbol{t}\boldsymbol{Y}^*) + \boldsymbol{E} \\ &= (u\boldsymbol{t}\boldsymbol{G}\ u\boldsymbol{t}^*\boldsymbol{G}) + \boldsymbol{E} \\ &= u\,(\boldsymbol{t}, \boldsymbol{t}^*\boldsymbol{G} + \boldsymbol{E}) \\ &= u\hat{\boldsymbol{t}}\boldsymbol{G} + \boldsymbol{E}. \end{aligned}$$

If we set $\bar{\boldsymbol{C}} = \boldsymbol{C} \cdot (\boldsymbol{e}_n^t \otimes \boldsymbol{I}_l) \in \mathbb{Z}_q^{nk \times l}$ consist of the last l columns of $\boldsymbol{C}$, then

$$\boldsymbol{t}\bar{\boldsymbol{C}} \approx u\boldsymbol{g} \tag{2}$$

with error $\boldsymbol{E}_{\bar{\boldsymbol{C}}}$ (same as $\boldsymbol{E}_{\boldsymbol{C}}$). Different from Equation 2, we establish a relationship between the additional key $\boldsymbol{t}^*$ and the plaintext $u \in \{0,1\}$ to satisfy

$$\boldsymbol{t}^*\bar{\boldsymbol{C}} \approx u\boldsymbol{g}$$

so that we can construct matrix $\boldsymbol{Y}$ to satify $\boldsymbol{t}^*\boldsymbol{Y} \approx u\boldsymbol{t}^*\boldsymbol{G}$, which is in line with our proposition above.

Firstly, we break $\boldsymbol{C} \cdot (\boldsymbol{e}_n^t \otimes \boldsymbol{I}_l) \in \mathbb{Z}_q^{nk \times l}$ into k rows submatrices $\bar{\boldsymbol{C}}^i \in \mathbb{Z}_q^{n \times l}$, i.e.,

$$\bar{\boldsymbol{C}} = \begin{bmatrix} \bar{\boldsymbol{C}}^1 \\ \bar{\boldsymbol{C}}^2 \\ \vdots \\ \bar{\boldsymbol{C}}^k \end{bmatrix}$$

Then, every encryption participant sets its own

$$\begin{aligned} \boldsymbol{P}_i &= \boldsymbol{A}^*\boldsymbol{R}_i + \left(\boldsymbol{\omega}^T \otimes \boldsymbol{t}_i \otimes \boldsymbol{g}\right) \\ &= \boldsymbol{A}^*\boldsymbol{R}_i + \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{O} \\ \vdots \\ \boldsymbol{t}_i \otimes \boldsymbol{g} \end{bmatrix} \end{aligned}$$

to compute $\left(\bar{\boldsymbol{C}}^i\right)' \in \mathbb{Z}_q^{n \times l}$ where $i \in [k]$, $\boldsymbol{R}_i \in \{0,1\}^{m \times m}$, $\boldsymbol{\omega} = (0, 0, .., 1)^n$ and $\boldsymbol{A}^*$ is the public key with respect to $\boldsymbol{t}^*$ and then we get the final $\left(\bar{\boldsymbol{C}}\right)' \in \mathbb{Z}_q^{n \times l}$:

1) Computing $\left(\bar{\boldsymbol{C}}^i\right)' \in \mathbb{Z}_q^{n \times l}$

$$\begin{aligned} \left(\bar{\boldsymbol{C}}^i\right)' &= \boldsymbol{P}_i \cdot \boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right) \\ &= \left(\boldsymbol{A}^*\boldsymbol{R}_i + \left(\boldsymbol{\omega}^T \otimes \boldsymbol{t}_i \otimes \boldsymbol{g}\right)\right) \cdot \boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right) \\ &= \boldsymbol{A}^*\boldsymbol{R}_i \cdot \boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right) + \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{O} \\ \vdots \\ \boldsymbol{t}_i \otimes \boldsymbol{g} \end{bmatrix} \cdot \boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right) \\ &= \boldsymbol{A}^*\boldsymbol{R}_i \cdot \boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right) + \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{O} \\ \vdots \\ \boldsymbol{t}_i \cdot \bar{\boldsymbol{C}}^i \end{bmatrix} \end{aligned}$$

2) Computing $\bar{\boldsymbol{C}}' \in \mathbb{Z}_q^{n\times l}$:

$$\begin{aligned}\bar{\boldsymbol{C}}' &= \sum_{i=1}^{k}\left(\bar{\boldsymbol{C}}^i\right)' \\ &= \sum_{i=1}^{k}\boldsymbol{P}_i\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right) \\ &= \boldsymbol{A}^*\boldsymbol{R}_1\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^1\right)+\boldsymbol{A}^*\boldsymbol{R}_2\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^2\right)+...+ \\ &\boldsymbol{A}^*\boldsymbol{R}_i\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right)+\begin{bmatrix}\boldsymbol{O}\\ \boldsymbol{O}\\ \vdots\\ \boldsymbol{t}_1\cdot\bar{\boldsymbol{C}}^1+\boldsymbol{t}_2\cdot\bar{\boldsymbol{C}}^2+...+\boldsymbol{t}_k\cdot\bar{\boldsymbol{C}}^k\end{bmatrix} \\ &= \sum_{i=1}^{k}\boldsymbol{A}^*\boldsymbol{R}_i\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right)+\begin{bmatrix}\boldsymbol{O}\\ \boldsymbol{O}\\ \vdots\\ \sum_{i=1}^{k}\boldsymbol{t}_i\cdot\bar{\boldsymbol{C}}^i\end{bmatrix}\end{aligned}$$

Every encryption participant needs to broadcast its $\left(\bar{\boldsymbol{C}}^i\right)' \in \mathbb{Z}_q^{n\times l}$ after the calculation so that all participants can do the summarization operation, and we can find an interesting property here

$$\begin{aligned}\sum_{i=1}^{k}\boldsymbol{t}_i\bar{\boldsymbol{C}}^i &= \boldsymbol{t}\boldsymbol{C} \\ &= u\boldsymbol{g}+\boldsymbol{E}_{\boldsymbol{C}}\end{aligned}$$

and

$$\begin{aligned}&\boldsymbol{t}^*\bar{\boldsymbol{C}}' \\ &= \sum_{i=1}^{k}\boldsymbol{t}^*\boldsymbol{A}^*\boldsymbol{R}_i\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right)+\boldsymbol{t}^*\begin{bmatrix}\boldsymbol{O}\\ \boldsymbol{O}\\ \vdots\\ \sum_{i=1}^{k}\boldsymbol{t}_i\cdot\bar{\boldsymbol{C}}^i\end{bmatrix} \\ &= \sum_{i=1}^{k}\boldsymbol{t}^*\boldsymbol{A}^*\boldsymbol{R}_i\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right)+(-\bar{\boldsymbol{t}}^*,1)\begin{bmatrix}\boldsymbol{O}\\ \boldsymbol{O}\\ \vdots\\ \sum_{i=1}^{k}\boldsymbol{t}_i\cdot\bar{\boldsymbol{C}}^i\end{bmatrix} \\ &= u\cdot\boldsymbol{g}+\boldsymbol{E}_{\bar{\boldsymbol{C}}'}\end{aligned}$$

where $\boldsymbol{E}_{\bar{\boldsymbol{C}}'} \le km\beta+\boldsymbol{E}_{\bar{\boldsymbol{C}}}$.

Due to this discovery, we use the public matrix $\boldsymbol{P}^* = (\boldsymbol{\omega}^*\otimes\boldsymbol{A}^*\boldsymbol{R}^*)+(\boldsymbol{I}_n\otimes\boldsymbol{t}^*\otimes\boldsymbol{g})$ associated with $\boldsymbol{t}^*$ where $\boldsymbol{\omega}^* = (1,1,..,1)^n$, $\boldsymbol{R}_i \in \{0,1\}^{m\times m}$ and define

$$\boldsymbol{s} := \left(\boldsymbol{I}_n\otimes\boldsymbol{I}_n\otimes\boldsymbol{g}^{-1}\right)\cdot\left(\bar{\boldsymbol{C}}'\otimes\boldsymbol{I}_n\right)\cdot\boldsymbol{\Pi}$$

where $\boldsymbol{\Pi}$ is a permutation matrix of order nl satisfying $(\boldsymbol{A}\otimes\boldsymbol{B})\boldsymbol{\Pi} = (\boldsymbol{B}\otimes\boldsymbol{A})$, we observe that

$$\begin{aligned}&\boldsymbol{t}^*\cdot\boldsymbol{P}^* \\ &= \boldsymbol{t}^*\cdot((\boldsymbol{\omega}^*\otimes\boldsymbol{A}^*\boldsymbol{R}^*)+(\boldsymbol{I}_n\otimes\boldsymbol{t}^*\otimes\boldsymbol{g})) \\ &= (\boldsymbol{\omega}^*\otimes\boldsymbol{e}^*\boldsymbol{R}^*)+(\boldsymbol{t}^*\otimes\boldsymbol{t}^*\otimes\boldsymbol{g})\end{aligned}$$

so that

$$\begin{aligned}\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s} &= (\boldsymbol{\omega}^*\otimes\boldsymbol{e}^*\boldsymbol{R}^*)\cdot\boldsymbol{s}+ \\ &(\boldsymbol{t}^*\otimes\boldsymbol{t}^*\otimes\boldsymbol{g})\cdot\left(\boldsymbol{I}_n\otimes\boldsymbol{I}_n\otimes\boldsymbol{g}^{-1}\right)\cdot\left(\bar{\boldsymbol{C}}'\otimes\boldsymbol{I}_n\right)\cdot\boldsymbol{\Pi} \\ &= (\boldsymbol{\omega}^*\otimes\boldsymbol{e}^*\boldsymbol{R}^*)\cdot\boldsymbol{s}+(\boldsymbol{t}^*\otimes\boldsymbol{t}^*)\cdot\left(\bar{\boldsymbol{C}}'\otimes\boldsymbol{I}_n\right)\cdot\boldsymbol{\Pi} \\ &= (\boldsymbol{\omega}^*\otimes\boldsymbol{e}^*\boldsymbol{R}^*)\cdot\boldsymbol{s}+\left(\boldsymbol{t}^*\cdot\bar{\boldsymbol{C}}'\right)\otimes\boldsymbol{t}^*\cdot\boldsymbol{\Pi} \\ &= (\boldsymbol{\omega}^*\otimes\boldsymbol{e}^*\boldsymbol{R}^*)\cdot\boldsymbol{s}+\boldsymbol{t}^*\otimes\boldsymbol{E}_{\bar{\boldsymbol{C}}'}+u(\boldsymbol{t}^*\otimes\boldsymbol{g}). \qquad (3)\end{aligned}$$

Since the $\left(\boldsymbol{I}_n\otimes\boldsymbol{I}_n\otimes\boldsymbol{g}^{-1}\right)\cdot\left(\bar{\boldsymbol{C}}'\otimes\boldsymbol{I}_n\right) \in \{0,1\}^{n^2l\times nl}$ then

$$\begin{aligned}&\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s} \\ &= \boldsymbol{E}_{\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s}}+u(\boldsymbol{t}^*\otimes\boldsymbol{g}) \qquad (4)\end{aligned}$$

where $\boldsymbol{E}_{\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s}} \le n^3l\beta+B_\chi\boldsymbol{E}_{\bar{\boldsymbol{C}}'}$.

According to equations above, we know that if every participant broadcasts its $\left(\bar{\boldsymbol{C}}^i\right)' \in \mathbb{Z}_q^{n\times l}$ and then the key holder of $\boldsymbol{t}^*$ can recover the plaintext u by investigating if or not it is a malicious participant.

To avoid this risk and ensure the security of our scheme, we randomly select a participant j where $j \in [1,k]$ to perform only the computation. Specifically, the participant j can receive computation results broadcast by other participants, but it does not broadcast its own computation results so that

$$\begin{aligned}\left(\bar{\boldsymbol{C}}'\right)^{broadcast} &= \sum_{i=1}^{k\backslash j}\left(\left(\bar{\boldsymbol{C}}^{i'}\right)\right)^{broadcast} \\ &= \sum_{i=1}^{k\backslash j}\boldsymbol{A}^*\boldsymbol{R}_i\cdot\boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right)+\begin{bmatrix}\boldsymbol{O}\\ \boldsymbol{O}\\ \vdots\\ \sum_{i=1}^{k\backslash j}\boldsymbol{t}_i\cdot\bar{\boldsymbol{C}}^i\end{bmatrix} \\ &\neq \bar{\boldsymbol{C}}' \qquad (5)\end{aligned}$$

According to Equation (5), the key holder of $\boldsymbol{t}^*$ can not get any information about the plaintaext u even if it is a malicious participant. We also can observe that because of

$$\boldsymbol{t}_j\cdot\boldsymbol{A}^* \neq \boldsymbol{E} \qquad (6)$$

the j^{th} participant also can not recover u. Hence, the plaintext u is invisible to the key holder of $\boldsymbol{t}^{i^{th}}$ and $\boldsymbol{t}^*$ in the process of extension ciphertext formation.

Secondly, we need to add a "Blind" item to $\boldsymbol{Y}$ recalling how we want to construct $\boldsymbol{X}$ and $\boldsymbol{Y}$ with our description above. Since the public key is

$$pk = \boldsymbol{A} = \begin{bmatrix}\boldsymbol{B}\\ \boldsymbol{b}\end{bmatrix}$$

and suppose $\boldsymbol{C}$ is the GSW encryption of message 0 under $pk_1 = \boldsymbol{A}_1$ so that

$$\boldsymbol{C} = \boldsymbol{A}_1\boldsymbol{R}_1 = \begin{bmatrix}\boldsymbol{B}_1\boldsymbol{R}_1\\ \boldsymbol{b}_1\boldsymbol{R}_1\end{bmatrix}$$

for some random matrix $\boldsymbol{R} \in \{0,1\}^{m\times m}$. Later, assuming we are given $pk_2 = \boldsymbol{A}_2 = \begin{bmatrix}\boldsymbol{B}\\ \boldsymbol{b}_2\end{bmatrix}$ corresponding to the additional secret key $\boldsymbol{t}_2$. Then

$$\boldsymbol{t}_2\boldsymbol{C} = -\bar{\boldsymbol{t}}_2\boldsymbol{B}\boldsymbol{R}_1+\boldsymbol{b}_1\boldsymbol{R}_1$$

Because of $\boldsymbol{b}_2 = \bar{\boldsymbol{t}}_2\boldsymbol{B} + \boldsymbol{e}_2$, we observe that

$$\begin{aligned}\boldsymbol{t}_2\boldsymbol{C} &= (\boldsymbol{e}_2 - \boldsymbol{b}_2)\boldsymbol{R}_1 + \boldsymbol{b}_1\boldsymbol{R}_1 \\ &= \boldsymbol{e}_2\boldsymbol{R}_1 + (\boldsymbol{b}_1 - \boldsymbol{b}_2)\boldsymbol{R}_1 \end{aligned} \tag{7}$$

From Equation (4), if we simply set $\boldsymbol{Y} = \boldsymbol{P}^* \cdot \boldsymbol{s}$, the key holder of $\boldsymbol{t}^*$ can recover the plaintext u indenpendently. Therefor, we need to add a "blind" item to $\boldsymbol{Y}$ as we mentioned above. Then we combine Equation (3) with Equation (7) and let $\boldsymbol{C} = \boldsymbol{A}_i\boldsymbol{M}_i$ as the "Blind" term, i.e., we set

$$\boldsymbol{Y} = ((\boldsymbol{\omega}^* \otimes \boldsymbol{A}_i\boldsymbol{M}_i) + \boldsymbol{P}^*) \cdot \boldsymbol{s}$$

where $\boldsymbol{M}_i \in \{0,1\}^{m\times m}$. We observe that

$$\begin{aligned}&\boldsymbol{t}^*\boldsymbol{Y} \\ &= \boldsymbol{t}^* \cdot (\boldsymbol{\omega}^* \otimes \boldsymbol{A}_j\boldsymbol{M}_j) \cdot \boldsymbol{s} + \boldsymbol{t}^* \cdot \boldsymbol{P}^* \cdot \boldsymbol{s} \\ &= (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_j - \boldsymbol{b}_*) \cdot \boldsymbol{M}_j + \boldsymbol{e}_*\boldsymbol{M}_j \cdot \boldsymbol{s})) + \boldsymbol{E}_{\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s}} + u\,(\boldsymbol{t}^* \otimes \boldsymbol{g}) \\ &= (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_j - \boldsymbol{b}_*) \cdot \boldsymbol{M}_j \cdot \boldsymbol{s})) + (\boldsymbol{\omega}^* \otimes (\boldsymbol{e}^* \cdot \boldsymbol{M}_j)) \\ &+ \boldsymbol{E}_{\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s}} + u\,(\boldsymbol{t}^* \otimes \boldsymbol{g}) \\ &= (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_j - \boldsymbol{b}_*) \cdot \boldsymbol{M}_j \cdot \boldsymbol{s})) + u\,(\boldsymbol{t}^* \otimes \boldsymbol{g}) + \boldsymbol{E}_{\boldsymbol{Y}} \end{aligned} \tag{8}$$

where $\boldsymbol{E}_{\boldsymbol{Y}} \le n^3 l\beta + \boldsymbol{E}_{\boldsymbol{t}^*\cdot\boldsymbol{P}^*\cdot\boldsymbol{s}} \le 2n^3 l\beta + B_\chi \boldsymbol{E}_{\bar{\boldsymbol{C}}'}$.

If there exist a matrix $\boldsymbol{X}$ that can let $\boldsymbol{tX} = -(\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_j - \boldsymbol{b}_*) \cdot \boldsymbol{M}_j \cdot \boldsymbol{s}))$, then the Equation (8) can be satisfied and the extension ciphertext is a GSW ciphertext in structure.

Constructing X. From Equation (5) and Equation (6) above, we know that any participant, including $\boldsymbol{t}^*$, cannot acquire any information about plaintext u through $\boldsymbol{Y}$ individually because of the "blind" term. Hence, we can construct $\boldsymbol{X}$ through the "blind" term. Specifically, the j^{th} participant uses the GSW linear combination to generate a ciphertext $\boldsymbol{C}_{j\text{-}lc}$ based on $\boldsymbol{M}_j \in \{0,1\}^{m\times m}$ to get $\boldsymbol{X}$. Therefore, we define

$$\begin{aligned}\boldsymbol{C}_{j\text{-}lc} := GSW.LComb(\ (\ \boldsymbol{C}^{1,1}, ..., \boldsymbol{C}^{m,m}\ ) \in (\ \mathbb{Z}_q^{n\times m}\ )^{m^2}, \\ \boldsymbol{b}_* - \boldsymbol{b}_j\ ) \in \mathbb{Z}_q^{n\times m}\end{aligned}$$

where $\boldsymbol{C}^{a,b} \in \mathbb{Z}_q^{n\times m}$ is the GSW encryption of the each element of the private random matrix $\boldsymbol{M}_j \in \{0,1\}^{m\times m}$, i.e.,

$$\boldsymbol{C}^{a,b} \leftarrow GSW.Encrypt\,(pk, \boldsymbol{M}_j[a,b]) \in \mathbb{Z}_q^{n\times m}$$

It can be seen from **Property 1** that

$$\boldsymbol{t}_j\boldsymbol{C}_{j\text{-}lc} = (\boldsymbol{b}_* - \boldsymbol{b}_j) \cdot \boldsymbol{M}_j + \boldsymbol{e}$$

where $\boldsymbol{e} \le m^3\beta$.

So if we define

$$\boldsymbol{X} := \begin{bmatrix} \boldsymbol{O} \\ \vdots \\ (\boldsymbol{\omega}^* \otimes \boldsymbol{C}_{j\text{-}lc}) \cdot \boldsymbol{s} \\ \vdots \\ \boldsymbol{O} \end{bmatrix} \in \mathbb{Z}_q^{nk\times nkl}$$

where $\boldsymbol{O}$ is the zero matrix of order $n \times nl$. Then, we can have

$$\begin{aligned}&\boldsymbol{tX} \\ &= \boldsymbol{t}_j \cdot (\boldsymbol{\omega}^* \otimes \boldsymbol{C}_{j\text{-}lc}) \cdot \boldsymbol{s} \\ &= (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_* - \boldsymbol{b}_j) \cdot \boldsymbol{M}_j + \boldsymbol{e})) \cdot \boldsymbol{s} \\ &= (\boldsymbol{\omega}^* \otimes \boldsymbol{e}) \cdot \boldsymbol{s} + (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_* - \boldsymbol{b}_j) \cdot \boldsymbol{M}_j)) \cdot \boldsymbol{s} \\ &= \boldsymbol{E}_{\boldsymbol{X}} + (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_* - \boldsymbol{b}_j) \cdot \boldsymbol{M}_j)) \cdot \boldsymbol{s}\end{aligned}$$

Since $\boldsymbol{\omega}^* = \{1,1,...,1\}^n$, so $\boldsymbol{E}_{\boldsymbol{X}} \le m^3n^3l\beta$.

Since

$$\begin{aligned}&\boldsymbol{tX} + \boldsymbol{t}^*\boldsymbol{Y} \\ &= (\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_* - \boldsymbol{b}_j) \cdot \boldsymbol{M}_j)) \cdot \boldsymbol{s} + \\ &(\boldsymbol{\omega}^* \otimes ((\boldsymbol{b}_j - \boldsymbol{b}_*) \cdot \boldsymbol{M}_j)) \cdot \boldsymbol{s} + u\,(\boldsymbol{t}^* \otimes \boldsymbol{g}) + \boldsymbol{E}_{\boldsymbol{X}} + \boldsymbol{E}_{\boldsymbol{Y}} \\ &= u\,(\boldsymbol{t}^* \otimes \boldsymbol{g}) + \boldsymbol{E}_{\boldsymbol{X}+\boldsymbol{Y}}\end{aligned}$$

where $\boldsymbol{E}_{\boldsymbol{X}+\boldsymbol{Y}} \le m^3n^3l\beta + 2n^3l\beta + B_\chi\boldsymbol{E}_{\bar{\boldsymbol{C}}'}$. Finally, we have

$$\begin{aligned}\hat{\boldsymbol{t}}\hat{\boldsymbol{C}} &= (\boldsymbol{t}, \boldsymbol{t}^*)\begin{bmatrix}\boldsymbol{C} & \boldsymbol{X} \\ \boldsymbol{O} & \boldsymbol{Y}\end{bmatrix} \\ &= (\boldsymbol{tC}\ \ \boldsymbol{tX} + \boldsymbol{t}^*\boldsymbol{Y}) \\ &= u\hat{\boldsymbol{t}}\boldsymbol{G} + \boldsymbol{E}_{\hat{\boldsymbol{C}}}\end{aligned}$$

which indicates that $\hat{\boldsymbol{C}}$ is a GSW ciphertext corresponding to the message $u \in \{0,1\}$ under the secret key $sk = \hat{\boldsymbol{t}} = (\boldsymbol{t}, \boldsymbol{t}^*) \in \chi^{n(k+1)}$ with error $\left|\left|\boldsymbol{E}_{\hat{\boldsymbol{C}}}\right|\right|_\infty = \max\{||\boldsymbol{E}_{\boldsymbol{C}}||_\infty, ||\boldsymbol{E}_{\boldsymbol{X}+\boldsymbol{Y}}||_\infty\} \le (m+2)^3n^3l\beta + k\beta^2 + B_\chi||\boldsymbol{E}_{\boldsymbol{C}}||_\infty$. So, in our construction, the error bound for extension is equal to $poly\,(n,k,l) + B_\chi||\boldsymbol{E}_{\boldsymbol{C}}||_\infty$, which is a multiple of the original error $||\boldsymbol{E}_{\boldsymbol{C}}||_\infty$ by a factor B_χ. We can still extend a ciphertext under additional multiple keys while incurring increase in the error by a factor of B_χ.

In order to make the process of ciphertext extension more clear, not just in the mathematical expression, we will use an algorithm to explain this process and a figure illustrate our proposed scheme below.

The algorithm below clearly illustrates the process of ciphertext extension in our scheme. In Algorithm 1, the input parameter $\boldsymbol{C}$ of the ciphertext extension function is a simply GSW ciphertext. The input parameter *Bro_flag* is TRUE means that except for a certain participant, other participants need to broadcast their calculation results in the process of ciphertext extension when additional keys occur. In addition, when some participants are offline, as long as it has published the parameter $\boldsymbol{P}_i = \boldsymbol{A}^*\boldsymbol{R}_i + \left(\boldsymbol{\omega}^T \otimes \boldsymbol{t}_i \otimes \boldsymbol{g}\right)$ required by the ciphertext extension in advance, the ciphertext extension can still proceed normally. The security of the whole process can be guaranteed as mentioned above and be explained in the following section of security analysis.

In Figure 1, we compare the traditional multi-key FHE scheme with ours. In the traditional multi-key FHE scheme, after the plaintext is encrypted, the ciphertext is uploaded to the cloud via the internet. When it is necessary to perform homomorphic computation on the ciphertext encrypted under different keys, the cloud needs to extend the ciphertexts to the concatenation secret key to ensure that the evaluated ciphertexts can be decrypted correctly. The process of ciphertext extension increases the workload of the cloud, which requires ciphertexts provider to pay more to the cloud service provider. In this article, we propose a distribution method to extend ciphertexts through distributed computing. Participants interactively perform the process of ciphertext extension through the internet, and the cloud only needs to do homomorphic computation. Obviously, this improvement addresses the disadvantages of the traditional multi-key FHE scheme we presented above.

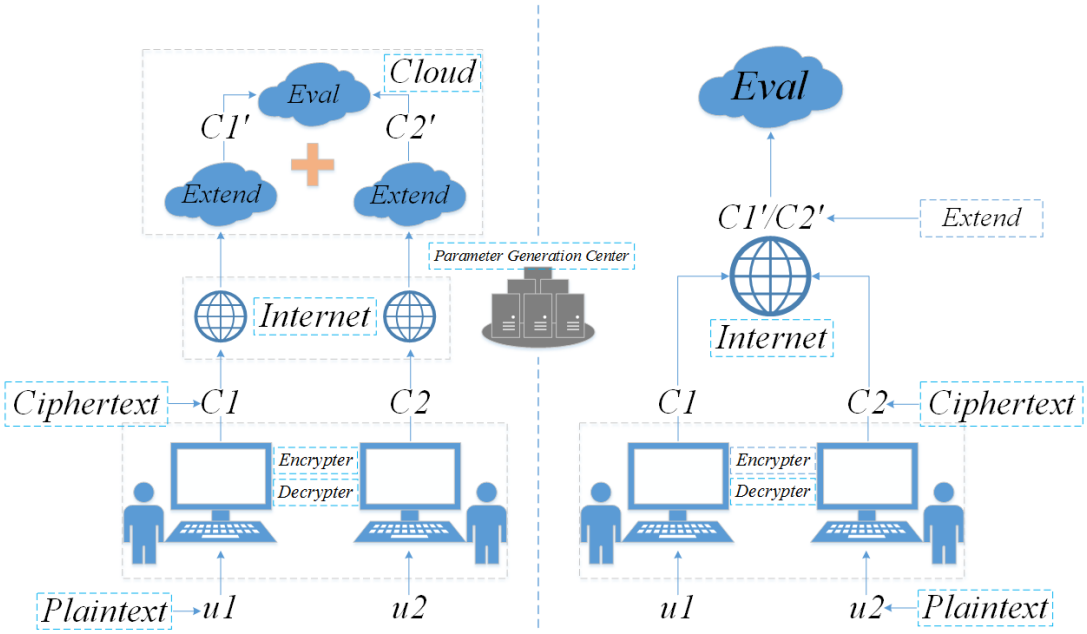


Fig. 1: A comparison between the traditional multi-key FHE scheme (left) and ours (right).

C. Parameters Setting

Now we bound the worst-case error growth when homomorphically evaluating a depth L circuit of NAND gates. Suppose there are two ciphertext (fresh, extended or evaluated) $\boldsymbol{C}_1 \in \mathbb{Z}_q^{nk\times nkl}$ and $\boldsymbol{C}_2 \in \mathbb{Z}_q^{nk\times nkl}$ be the encryption of u_1 and u_2 respectively under the concatenation key $\boldsymbol{t} = (\boldsymbol{t}_1, \boldsymbol{t}_2, ..., \boldsymbol{t}_k) \in \chi^{nk}$, satisfying Equation (1) with the error bound by E . As GSW13 mentioned, the homomorphic computation of two ciphertexts by NAND gates has the error bounded by $(nkl+1)E = poly(n,k,l)E$ where $poly(n,k,l)$ enotes a polynomial function in n, k and l.

When we extend a ciphertext with the error bounded by $\boldsymbol{E}^*$, the final ciphertext has the error bounded by $(m+2)^3 n^3 l\beta + k\beta^2 + B_\chi \boldsymbol{E}^*$. Therefore, for any depth L homomorphic computation on ciphertexts encrypted under k keys, the result has the error bounded by $poly(n,k,l)^{k+l}\boldsymbol{E}^*$. Therefore, it suffices to choose a modulus $q \geq 4poly(n,k,l)^{k+l}\boldsymbol{E}^*$ following the previous work. Recall that $l = \Theta(\log q) = \widetilde{O}(k+d)$, where $\widetilde{O}$ hides the logarithmic terms and χ is a discrete Gaussian distribution with the error bound $B_\chi = \Theta(n)$. The LWE problem with this parameterizations is hard and corresponds to a worst-case approximation factor of $poly(n,k,l)^{k+l}$ for n-dimensional lattice problems.

IV. Security Analysis

In this section, we discuss the security of our scheme $MFHE = (Setup, Keygen, Encrypt, Decrypt, Extend, Eval)$. Compared with **scheme #2** in [23], the biggest difference is that we release the circular-security assumption in the process of the ciphertext extension. We prove that our scheme is IND-CPA secure without the circular-security assumption.

Theorem 4.1 Our scheme $MFHE = (Setup, Keygen, Encrypt, Decrypt, Extend, Eval)$ described in Section 3 is IND-CPA secure under the decisional $LWE_{n-1,m,q,\chi}$ problem.

Proof. We prove that the public extension key and the ciphertexts are indistinguishable in the real world from the ideal world for any *PPT* adversary $\mathcal{A}$. Let we consider the hybrid experiments in the real word and the idea world respectively as follows:

Game 0: This is the real IND-CPA game played between a challenger *CH* and an adversary $\mathcal{A}$. More precisely:

1) The challenger *CH* runs the algorithm $Setup\left(1^\lambda, 1^L, 1^k\right)$ to generate the public params and $Keygen(params)$ to obtain key pair (PK, sk) where PK include a public key $\boldsymbol{A} = \begin{bmatrix}\boldsymbol{B}\\ \boldsymbol{b}\end{bmatrix} \in \mathbb{Z}_q^{n\times m}$ and a public extension key $\boldsymbol{P} = (\boldsymbol{\omega}\otimes\boldsymbol{AR}) + (\boldsymbol{I}_n \otimes \boldsymbol{t}\otimes\boldsymbol{g}) \in \mathbb{Z}_q^{n\times n^2 l}$, secret key $sk = \boldsymbol{t} = (-\bar{\boldsymbol{t}}, 1) \in \chi^n$ and $\boldsymbol{R} \in \{0,1\}^{m\times m}$ is a random matrix. Then the challenger *CH* sends PK to the *PPT* adversary $\mathcal{A}$.

2) The adversary $\mathcal{A}$ chooses a pair of message u_0, $u_2 \in \{0,1\}$ for the challenger *CH*.

3) The challenger *CH* chooses a random bit $b \in \{0,1\}$ and a random matrix $\boldsymbol{R} \in \{0,1\}^{m\times m}$, then runs the encryption algorithm $Encrypt(pk, u)$ to generate ciphertext $\boldsymbol{C}_b = \boldsymbol{AR} + u_b\boldsymbol{G}$ and sends the challenge ciphertext $\boldsymbol{C}_b$ to adversary $\mathcal{A}$.

4) Finally, the adversary $\mathcal{A}$ guesses the bit for b as b' in the polynomial time and then sends bit b' to challenger *CH*. The **Game 0** outputs 1 if b' = b and 0 otherwise.

Game 1: This is a hybrid experiment in the ideal word, unlike the hybrid experiment **Game 0**, **Game 1** has the following facts:

1) The challenger runs the corresponding algorithm to obtain key pair (PK, sk) and chooses a uniformly random matrix $\boldsymbol{U} \in \mathbb{Z}_q^{n\times m}$, then generates the public extension key $\boldsymbol{P} = (\boldsymbol{\omega}\otimes\boldsymbol{U}) + (\boldsymbol{I}_n \otimes \boldsymbol{t}\otimes\boldsymbol{g}) \in \mathbb{Z}_q^{n\times n^2 l}$ and sends PK to the *PPT* adversary $\mathcal{A}$.

Algorithm 1 The Process of Ciphertext Extension with Multiple Participants

Input:
Function Ciphertext_Extend $(\boldsymbol{C}, \boldsymbol{P}^*, \boldsymbol{A}^*, \text{Bro\_flag})$;
Output:
//construct $\boldsymbol{Y}$.
for each $i \in [1,k]$ **do**
  break $\bar{\boldsymbol{C}} = \boldsymbol{C} \cdot (\boldsymbol{e}_n^t \otimes \boldsymbol{I}_l)$ to k rows sub-matrices $\bar{\boldsymbol{C}}^i$;
  set $\boldsymbol{P}_i = \boldsymbol{A}^* \boldsymbol{R}_i + (\boldsymbol{\omega}^T \otimes \boldsymbol{t}_i \otimes \boldsymbol{g})$; // $\boldsymbol{R}_i \in \{0,1\}^{m \times m}$ is an uniformly random matrix.
  compute $\left(\bar{\boldsymbol{C}}^i\right)' = \boldsymbol{P}_i \cdot \boldsymbol{g}^{-1}\left(\bar{\boldsymbol{C}}^i\right)$;
  if Bro_flag **then**
    broadcast $\left(\bar{\boldsymbol{C}}^i\right)'$;
  else
    compute $\bar{\boldsymbol{C}}' = \sum_{i=1}^{k} \left(\bar{\boldsymbol{C}}^i\right)'$;
    set $\boldsymbol{s} := \left(\boldsymbol{I}_n \otimes \boldsymbol{I}_n \otimes \boldsymbol{g}^{-1}\right) \cdot \left(\bar{\boldsymbol{C}}' \otimes \boldsymbol{I}_n\right) \cdot \boldsymbol{\Pi}$;
    compute Trans_matrix $\boldsymbol{Y}_1 = \boldsymbol{P}^* \cdot \boldsymbol{s}$;
    sample an uniformly matrix $\boldsymbol{M} \in \{0,1\}^{m \times m}$;
    compute Trans_matrix $\boldsymbol{Y}_2 = (\boldsymbol{\omega}^* \otimes \boldsymbol{A}_i \boldsymbol{M}_i) \cdot \boldsymbol{s}$;
    set $\boldsymbol{Y} = \boldsymbol{Y}_1 + \boldsymbol{Y}_2$; //finish $\boldsymbol{Y}$.

    //construct $\boldsymbol{X}$.
    call function to get $\boldsymbol{C}^{(a,b)} = \text{GSW.Encrypt}(pk, \boldsymbol{M}[a,b])$ for each element of $\boldsymbol{M}$;
    call function to get $\boldsymbol{C}_{lc} = \text{GSW.LComb}\left(\left(\boldsymbol{C}^{1,1}, ..., \boldsymbol{C}^{m,m}\right), \boldsymbol{b}_* - \boldsymbol{b}\right)$;
    broadcast $\boldsymbol{C}_{lc}$; //finish $\boldsymbol{X}$.
  end if
end for

construct $\hat{\boldsymbol{C}}$ by $\boldsymbol{C}, \boldsymbol{X}$ and $\boldsymbol{Y}$ according to the format;

return $\hat{\boldsymbol{C}}$;

2) The challenger *CH* chooses a random bit $b \in \{0,1\}$ and an uniformly random matrix $\boldsymbol{U} \in \mathbb{Z}_q^{n \times m}$, then generates the challenge ciphertext $\boldsymbol{C}_b = \boldsymbol{U} + u_b \boldsymbol{G}$ and sends the challenge ciphertext $\boldsymbol{C}_b$ to adversary $\mathcal{A}$.

We define the probability that the adversary guesses bit b correctly as $Pr[S_i]$. It is found that the advantage of $\mathcal{A}$ is $Pr[S_1] = \frac{1}{2}$ in the ideal world is owing to the challenge ciphertext $\boldsymbol{C}_b = \boldsymbol{U} + u_b \boldsymbol{G}$ and the public extension key $\boldsymbol{P} = (\boldsymbol{\omega} \otimes \boldsymbol{U}) + (\boldsymbol{I}_n \otimes \boldsymbol{t} \otimes \boldsymbol{g})$ are uniformly random and independent of the message as $\boldsymbol{U} \in \mathbb{Z}_q^{n \times m}$ is an uniformly random matrix. Our scheme is IND-CPA secure under the decisional $LWE_{n-1,m,q,\chi}$ problem if the advantage of $\mathcal{A}$ satisfies **Lemma 1** in these two worlds.

Lemma 1. $|Pr[S_0] - Pr[S_1]| = |Pr[S_0] - \frac{1}{2}| = \varepsilon$

where ε is negligible.

Proof. In the real word **Game 0**, the ciphertext is generated as $\boldsymbol{C}_b = \boldsymbol{AR} + u_b \boldsymbol{G}$ where $\boldsymbol{A}$ is the public key and each column of $\boldsymbol{A}$ is a LWE sample. By contrast, the ciphertext form is $\boldsymbol{C}_b = \boldsymbol{U} + u_b \boldsymbol{G}$ in the ideal word **Game 1** where $\boldsymbol{U} \in \mathbb{Z}_q^{n \times m}$ is an uniformly random matrix. Let define the two distributions "real" and "ideal" as

$$real = \{\boldsymbol{C}_b \in \mathbb{Z}_q^{n \times m} : \boldsymbol{C}_b = \boldsymbol{AR} + u_b \boldsymbol{G} \text{ where } \boldsymbol{A} \in \mathbb{Z}_q^{n \times m} \text{ is a LWE sample and } \boldsymbol{R} \in \{0,1\}^{m \times m} \text{ is a random matrix } \}$$

$$ideal = \{\boldsymbol{C}_b \in \mathbb{Z}_q^{n \times m} : \boldsymbol{C}_b = \boldsymbol{U} + u_b \boldsymbol{G} \text{ where } \boldsymbol{U} \in \mathbb{Z}_q^{n \times m} \text{ is an uniformly random matrix } \}$$

where $u_b \in \{0,1\}$. Then if a distinguisher $\mathcal{A}$ can distinguish these two distributions "real" and "ideal" , then $\mathcal{A}$ can distinguish $\boldsymbol{AR} \in \mathbb{Z}_q^{n \times m}$ from $\boldsymbol{U} \in \mathbb{Z}_q^{n \times m}$. Therefore we can obtain that

$$Pr[S_0] = Pr[\mathcal{A}(\boldsymbol{C}_b) = 1 : \boldsymbol{C}_b \leftarrow real]$$

$$Pr[S_1] = Pr[\mathcal{A}(\boldsymbol{C}_b) = 1 : \boldsymbol{C}_b \leftarrow ideal].$$

As we mentioned above, our scheme is IND-CPA secure if the advantage of $\mathcal{A}$ is

$$|Pr[S_0] - Pr[S_1]| = negligible$$

among the real and ideal worlds, namely,

$$|\, Pr[\mathcal{A}(\boldsymbol{C}_b) = 1 : \boldsymbol{C}_b \leftarrow real] - Pr[\mathcal{A}(\boldsymbol{C}_b) = 1 : \boldsymbol{C}_b \leftarrow ideal] \,| = negligible$$

for these two distributions, it is the advantage of $\mathcal{A}$ o distinguish the matrix $\boldsymbol{AR} \in \mathbb{Z}_q^{n \times m}$ from the uniformly random matrix $\boldsymbol{U} \in \mathbb{Z}_q^{n \times m}$. This is negligible by the following **Lemma 2**:

Lemma 2 [12]. Let $params = (n, q, m, \chi)$ be such that the $LWE_{n,q,\chi}$ assumption holds. Then, for $m = O(n \log q)$, $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ and $\boldsymbol{R} \in \{0,1\}^{m \times m}$, the joint distribution $(\boldsymbol{A}, \boldsymbol{A} \cdot \boldsymbol{R})$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{n \times m}$.

Consequently, $\boldsymbol{A} \cdot \boldsymbol{R}$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{n \times m}$ under the decisional *LWE* assumption, hence the advantage of $\mathcal{A}$ in distinguishing matrix $\boldsymbol{A} \cdot \boldsymbol{R} \in \mathbb{Z}_q^{n \times m}$ from $\boldsymbol{U} \in \mathbb{Z}_q^{n \times m}$ is negligible, namely, $|\, Pr[\mathcal{A}(\boldsymbol{C}_b) = 1 : \boldsymbol{C}_b \leftarrow real] - Pr[\mathcal{A}(\boldsymbol{C}_b) = 1 : \boldsymbol{C}_b \leftarrow ideal] \,| = negligible$ so that $|Pr[S_0] - Pr[S_1]| = negligible$. On the other hand, from **Lemma 2**, we can observe that the secret key $\boldsymbol{t} \in \chi^n$ is computationally hided by $\boldsymbol{A} \cdot \boldsymbol{R} \in \mathbb{Z}_q^{n \times m}$. It means that the public extension key and every ciphertext are uniformly random and independent of the messages. Therefore, **Theorem 4.1** holds and our scheme is IND-CPA secure.

V. Conclusion

In this work, we have proposed a dynamic multi-key FHE scheme from LWE. Compared with the traditional dynamic multi-key FHE scheme, we effectively solve the shortcomings of the existing multi-key FHE schemes, that is, we use a dietribution method to reduce the workload of the cloud and the cost of the encrypter on the cloud. We have shown that our construction is comparable with the other multi-key FHE

schemes with respect to public parameters length, ciphertext size, assumption and so on. Furthermore, our scheme is only based on LWE without other assumptions and has a light public key, which makes the process of ciphertext extension more efficient. In addition, unlike other existing dynamic multi-key FHE schemes in the symmetric key setting, our scheme works in asymmetric key setting. However, our dynamic multi-key FHE construction is of single bit encryption. In future, we will focus more on studying dynamic, multi-bit and multi-key FHE design from LWE. Furthermore, in addition to the GSW13 FHE scheme, we would like to further explore using different FHE schemes to design more efficient dynamic multi-key FHE schemes from LWE.

REFERENCES

[1] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of Secure Computation, Academia Press*, pp. 169–179, 1978.

[2] C. Gentry, "Fully homomorphic encryption using ideal lattices," vol. 9, 01 2009, pp. 169–178.

[3] C. Gentry, "Toward basing fully homomorphic encryption on worst-case hardness," in *Advances in Cryptology-crypto, Cryptology Conference, Santa Barbara, Ca, Usa, August*, 2010.

[4] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "V.: Fully homomorphic encryption over the integers," vol. 2009, 01 2009, p. 616.

[5] N. Smart and F. Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes," *IACR Cryptol. ePrint Arch.*, vol. 2009, p. 571, 2009.

[6] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme," 2011.

[7] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, "Fully homomorphic encryption over the integers with shorter public keys," in *Proceedings of the 31st Annual Conference on Advances in Cryptology*, ser. CRYPTO-11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 487–504.

[8] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *Proceedings of the 31st Annual Conference on Advances in Cryptology*, ser. CRYPTO-11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 505–524.

[9] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) lwe," in *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, 2011, pp. 97–106.

[10] J. H. Cheon and J. Kim, "A hybrid scheme of public-key encryption and somewhat homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 5, pp. 1052–1063, 2017.

[11] J. C. Bajard, P. Martins, L. Sousa, and V. Zucca, "Improving the efficiency of svm classification with fhe," *IEEE transactions on information forensics and security*, vol. 15, pp. 1709–1722, 2020.

[12] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," *Proceedings of Advances in Cryptology-Crypto*, vol. 8042, 08 2013.

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

[14] 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 Forty-Fourth Annual ACM Symposium on Theory of Computing*, ser. STOC-12. New York, NY, USA: Association for Computing Machinery, 2012, pp. 1219–1234.

[15] J. Hoffstein, J. Pipher, and J. H. Silverman, "Ntru: A ring-based public key cryptosystem," in *Proceedings of the Third International Symposium on Algorithmic Number Theory*, ser. ANTS-III. Berlin, Heidelberg: Springer-Verlag, 1998, pp. 267–288.

[16] M. Clear and C. McGoldrick, "Multi-identity and multi-key leveled fhe from learning with errors," vol. 9216, 08 2015, pp. 630–656.

[17] P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key fhe," in *EUROCRYPT (2)*. Springer, 2016, pp. 735–763.

[18] E. Kim, H.-S. Lee, and J. Park, "Towards round-optimal secure multiparty computations: Multikey fhe without a crs," *International Journal of Foundations of Computer Science*, vol. 31, no. 02, pp. 157–174, 2020. [Online]. Available: https://doi.org/10.1142/S012905412050001X

[19] H. Wang, Y. Feng, Y. Ding, and S. Tang, "A multi-key smc protocol and multi-key fhe based on some-are-errorless lwe," *Soft Comput.*, vol. 23, no. 5, pp. 1735–1744, Mar. 2019. [Online]. Available: https://doi.org/10.1007/s00500-017-2896-9

[20] H. Chen, I. Chillotti, and Y. Song, "Multi-key homomorphic encryption from tfhe," vol. 11922, pp. 446–472, 2019.

[21] P. Ananth, A. Jain, Z. Jin, and G. Malavolta, *Multi-key Fully-Homomorphic Encryption in the Plain Model*, 12 2020, pp. 28–57.

[22] Z. Brakerski and R. Perlman, "Lattice-based fully dynamic multi-key fhe with short ciphertexts," in *Proceedings, Part I, of the 36th Annual International Cryptology Conference on Advances in Cryptology-CRYPTO 2016 - Volume 9814*. Berlin, Heidelberg: Springer-Verlag, 2016, pp. 190–213.

[23] C. Peikert and S. Shiehian, "Multi-key fhe from lwe, revisited," in *Proceedings, Part II, of the 14th International Conference on Theory of Cryptography-Volume 9986*. Berlin, Heidelberg: Springer-Verlag, 2016, pp. 217–238.

[24] R. D. Chinmoy Biswas, "Dynamic multi-key fhe in symmetric key setting from lwe without using common reference matrix," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–14, 2021.

[25] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé, "Classical hardness of learning with errors," *CoRR*, vol. abs/1306.0281, 2013. [Online]. Available: http://arxiv.org/abs/1306.0281

[26] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," vol. 2011, 01 2011, p. 501.

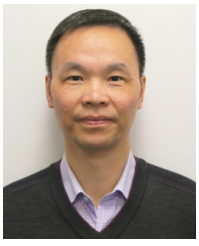
[27] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem: Extended abstract," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, ser. STOC-09. New York, NY, USA: Association for Computing Machinery, 2009, pp. 333–342. [Online]. Available: https://doi.org/10.1145/1536414.1536461

Yuling Chen received her BS from Taishan University, Taian, PR China, in 2006; MS from Guizhou University, Guiyang, PR China, in 2009; PHD from Guizhou University, Guiyang, PR China, in 2021. She is now an associate professor in State Key Laboratory of Public Big Data, Guizhou University, Guiyang, PR China. Her recent research interests include cryptography and information safety, Blockchain.

Sen Dong is currently a Graduate Student with the College of Computer Science and Technology, Guizhou University. His research interests include blockchain, privacy protection, cryptography, and information security.

Tao Li is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Guizhou University, Guiyang city, Guizhou province, China. He received a Bachelor degree in Computer Science from Shandong Normal University (2001), and a Master degree in Software Enginery from Dalian University of Technology (2007), respectively. His research interests include information security, cryptography, and blockchain technology.

Yilei Wang received her PhD degree from Shandong University in 2014. Currently she is an associate professor of Qufu Normal University. Her research interests include blockchain security and game theory.



Huiyu Zhou received a Bachelor of Engineering degree in Radio Technology from Huazhong University of Science and Technology of China (1990), and a Master of Science degree in Biomedical Engineering from University of Dundee of United Kingdom (2002), respectively. He was awarded a Doctor of Philosophy degree in Computer Vision from Heriot-Watt University, Edinburgh, United Kingdom (2006). Dr. Zhou currently is a full Professor at School of Informatics, University of Leicester, United Kingdom. He has published over 350 peer-reviewed papers in the field.