Key lifting : Multi-key Fully Homomorphic Encryption in plain model without noise flooding in partial decryption

Xiaokang Dai[1,2] Wenyuan Wu[✉,2] and Yong Feng[2]

[1] University of Chinese Academy of Sciences, Beijing, 100049 China
[2] Chongqing Key Laboratory of Automated Reasoning and Cognition,Chongqing Institute of Green and Intelligent Technology, Chongqing, 400714, China
daixiaokang@cigit.ac.cn wuwenyuan@cigit.ac.cn yongfeng@cigit.ac.cn

Abstract. Multi-key Fully Homomorphic Encryption(MKFHE) based on Learning With Error(LWE) usually lifts ciphertexts of different users to new ciphertexts under a common public key to enable homomorphic evaluation. The main obstacle of current MKFHE schemes in applications is huge ciphertext expansion cost especially in data intensive scenario. For example, for a boolean circuit with input length N, multiplication depth L, security parameter λ , the number of additional encryptions introduced to obtain ciphertext expansion is $O(N\lambda^6 L^4)$.

In this paper we present a framework to solve this problem that we call Key-Lifting Multi-key Fully Homomorphic Encryption (KL-MKFHE). By introducing a key lifting procedure, the number of encryption for a local user is pulled back to $O(N)$ as single-key fully homomorphic encryption(FHE). Moreover, current MKFHE schemes are often based on Common Reference String model(CRS). In our LWE-based scheme, CRS is removed by using the leakage resilient property of the leftover hash lemma(LHL). In particular, we noticed that as long as our encryption scheme is leakage-resilient, the partial decryption does not need to introduce noise flooding technique, and the semantic security of fresh ciphertext can also be guaranteed, which greatly reducing the modulus q and the computational overhead of the entire scheme.

Due to the structural properties of polynomial rings, such LWE-based scheme cannot be trivially transplanted to RLWE-based scheme. We give a RLWE-based KL-MKFHE under Random Oracle Model(ROM) by introduing a bit commitment protocol.

Keywords: Multi-key homomorphic encryption · LWE · RLWE · Leakage resilient cryptography.

1 Introduction

Fully Homomorphic Encryption(FHE). The concept of FHE was proposed by Rivest et al. [41], within a year of publishing of the RSA scheme [42]. The first truly fully homomorphic scheme was proposed by Gentry in his doctoral dissertation [20]in 2009. Based on Gentry's ideas, a series of FHE schemes have

been proposed [21] [44] [9] [19] [23] [14] [13], and their security and efficiency have been continuously improved. FHE is suitable to the problem of unilateral outsourcing computations. However in the case of multiple data providers, in order to support homomorphic evaluation, data must be encrypted by a common public key. Due to privacy of data, it is unreasonable to require participants to use other people's public keys to encrypt their own data.

Threshold fully holomorphic encryption(Th-FHE**).**After giving the first fully homomorphic encryption scheme, also, for the situation of multiple participants, Gentry gives the corresponding strategy : first, all participants execute a secure multi-party computation protocol to obtain a public key which all data are encrypted by, and then ciphertext evaluation is performed. After the evaluation is completed, all participants execute another secure MPC protocol to obtain the result. Obviously, the threshold was initially added to FHE only to support multiple users, while the later Th-FHE is more concerned with the flexibility of the access strategy in order to cope with different application scenarios. In addition, there are two main ways to initialize the master public key(used to encrypt plaintext) of Th-FHE. First, [25] [8] assumes that there is a central authority, which generates the master public key, and disperses the private key (using Secret Sharing scheme) to each participant. Encryption and operation of data are all under the same master public key, when decryption is required, the set of participants that satisfy the access control structure obtains the result through a round of interactive decryption. [8] further proposes the concept of Universal Thresholdizer, which for any fully homomorphic encryption scheme, it can be converted into a threshold fully homomorphic encryption supporting monotonic access control structure in a black-box manner.

The second method is for the parties to generate the master key in a distributed manner, where there is no central authority. For example, [38] adds a threshold function to the integer homomorphic scheme [18], and uses a distributed manner to generate the master public key and private key, without a central setup. Although adopting the black box method for the construction process, the distributed key generation process is quite complicated, which consists of three steps, firstly generating the private key, then the private key of the squeezed circuit, and finally the master public key. These three processes all need to repeatedly invoke the distributed bit generation, the comparison, and the multiplication protocols. Based on the key homomorphic property, [6] generates the master public key through two rounds of interaction in a distributed manner, and the master private key is the sum of the individual private keys. In order to match the public and private keys and ensure the security of the private key, a common reference string(CRS) needs to be introduced, and decryption requires everyone to provide the private key, which is actually a (n-n)Th-FHE. [17] introduces homomorphic encryption in order to optimize the preprocessing stage (such preprocessing is typically based on the classic circuit randomization technique of Beaver [7], it can be done by evaluating in parallel many small circuits of small multiplicative depth), and, a common reference string also needs to be introduced.

Multi-key Fully Homomorphic Encryption(MKFHE**).** To deal with privacy of multiple data providers, López-Alt et al. proposed the concept of MKFHE in [27] and construct the first MKFHE scheme based on modified-NTRU [43]. Conceptually, it is an enhancement of the FHE on function that allows data provider to encrypt data independent from other participants, its key generation and data encryption are done locally. To get the evaluated result, all participants are required to execute a round of threshold decryption protocol.

After López-Alt et al. proposed the concept of MKFHE, many schemes were proposed. In 2015, Clear and McGoldrick [15] constructed a GSW [23] LWE-based MKFHE. This scheme defined the master private key as the concatenation of all keys, and constructed a masking scheme to converts the ciphertext under individual public key to total key by introducing CRS and circular-LWE assumptions, which only supports single-hop computation. In 2016, Mukherjee and Wich [37], Perkert and shiehian [39], Brakerski and Perlman [11] constructed MKFHE scheme based on GSW respectively. [37] simplified the mask scheme of [15], and focused on constructing a two-round MPC protocol. Different methods in [39] and [11] were put forward dedicatedly to constructing a multi-hop MKFHE. [11] introduced bootstrapping to realize ciphertext expansion, thereby realizing the multi-hop function. [39] realized multi-hop function through ingenious construction. It is worth mentioning that all MKFHE schemes constructed based on the LWE scheme require a ciphertext expansion procedure.

1.1 Motivation

We note that the biggest difference between Th-FHE and MKFHE in form is that MKFHE allows participants to encrypt data with their own public keys, and does not require interaction during the initialization phase, while Th-FHE needs to introduce a dealer or generate the master key pair in a distributed manner. Conceptually, it is clear that MKFHE is more concise, and a series of work [10] [37] [5] shows that MKFHE is an excellent base tool for building round-optimal MPC. MKFHE seems attractive, but its actual construction involves some cumbersome operations and some unavoidable assumptions. Below we describe some details of the MKFHE scheme, and give our goal in the last paragraph of this subsection.

Ciphertext expansion is expensive : Although the MKFHE based on LWE can use LHL to remove CRS, in order to convert the ciphertext under different keys to the ciphertext under same key(ciphertext expansion), participants and the computing server need to do a lot of preparatory work. For ciphertext expansion, it is necessary to encrypt the random matrix $\mathbf{R} \in \mathbb{Z}_q^{m \times m}$ of each ciphertext. For a boolean circuit with input length N, multiplication depth L, security parameter λ, $m = n \log q + \omega(\log \lambda)$, the additional encryption operation introduced is $O(N\lambda^6 L^4)$, which is $O(N)$ for single-key FHE. For computing-sensitive participants, this is a lot of overhead.

CRS looks inevitable : Due to compact structure on polynomial ring and some interesting parallel algorithm such as SIMD, it is generally believed that FHE scheme based on RLWE is more efficient than FHE based on LWE. This is the reason why most current MKFHE schemes, such as [12] [36] are constructed based on RLWE.

Leftover Hash Lemma (LHL) over integer ring $\mathbb{Z}$ enjoys the leakage resilient property : It can transform an average quality random sources into higher quality [24] which can be used to get rid of CRS as [10] does. However, regularity lemma [29] over polynomial rings do not have corresponding properties, as [16]mentioned if the j-th Number theoretical Transfer(NTT) coordinate of each ring element in $\mathbf{x} = (x_1, \ldots, x_l)$ is leaked, then the j-th NTT coordinate of $a_{l+1} = \sum a_i x_i$ is defined, so a_{l+1} is very far from uniform, yet this is only a 1/n leakage rate. Therefore, it seems to be more difficult to remove CRS for RLWE-based MKFHE.

Noise flooding leads to extremely large module q : As far as we know so far, whether it is MKFHE or Th-FHE, a great noise needs to be introduced in the decryption stage to cover up the partial decryption result, otherwise, the individual key may be leaked. In order to make the result of partial decryption simulatable, assuming that the noise accumulated after the evaluation is $\mathbf{e}_{eval}$ and the private key is $\mathbf{s}$, the flooding noise e_{sm} must satisfy $\langle \mathbf{e}_{eval}, \mathbf{s} \rangle / e_{sm} = \mathsf{negl}(\lambda)$. At this time, in order to ensure the correctness of the decryption result, module q needs to satisfy $q \geq 4e_{sm}$. Thus noise flooding results in a q that is exponentially larger than the q in a single-key FHE.

Therefore, MKFHE as a general framework, although conceptually attractive, is not suitable for some specific scenarios. Especially in the era of mobile Internet, data providers often do not know each other, and sometimes it is difficult to convince them there is a dealer or the randomness of common reference string generated by a third party. At the same time, it is unreasonable to require the data provider to do $O(N\lambda^6 L^4)$ encryption on personal terminal.

In response to the above problems, we propose our goal: we consider multi-user *trust-sensitive* and *computationally-sensitive* scenario.

- Without CRS, we do not assume the existence of a dealer or a common reference string
- The data provider does as many encryptions as the single-key homomorphic scheme($O(N)$ for the circuit with input length N).
- $q = O(2^{\lambda+L} B_\chi)$ of the same size as the single-key homomorphic scheme, while $q = O(2^{\lambda L} B_\chi)$ for those schemes introduced noise flooding.

1.2 Related works

Except sum type of key structure [6], concatenation structure were studied in [15] [39] [37] [11] [12] together with CRS. [4] removed CRS from a higher dimension, instead of using LHL or regularity lemma, they based on Multiparty

Homomorphic Encryption and modified the initialization method of its root node to achieve this purpose, more details please refer to [4]. [10] is the first scheme using leakage resilient property of LHL to get rid of CRS, which has the concatenation total key structure, and ciphertext expansion is essential. All of the above schemes introduced noise flooding technology.

We present a comparison of some properties in related work in Table 1.

Scheme	Key structure	CRS	Noise flooding	Interaction(setup phase)
THFHE [6]	Sum	✓	✓	✓
MKFHE [12]	Concatenation	✓	✓	
MKFHE [37]	Concatenation	✓	✓	
MKFHE [10]	Concatenation		✓	✓
scheme#1	Sum			✓
scheme#2	Sum	ROM	✓	✓

Table 1.

1.3 Our Results

For *trust-sensitive* and *computationally-sensitive* scenario, we appropriately *tighten and loosen* the original definition of MKFHE, and introduce the concept of KL-MKFHE which is more suitable for such scenarios. Following this concept, we construct the first KL-MKFHE scheme based on LWE in the plain model.

Since regularity lemma [30] on rings has no corresponding leakage resilient properties, we cannot apply the LWE construction routine trivially to RLWE-based MKFHE. As a compromise, we introduce a round of bit commitment protocol to guarantee the independence of each participants, and construct the corresponding KL-MKFHE based on ROM.

We give a brief review of the new concept and two scheme below and explain how we remove noise flooding in the partial decryption phase.

The concept of KL-MKFHE :
Different from previous definition [37], we abandon ciphertext expansion procedure, instead, introducing a key lifting procedure which has the same functionality with ciphertext expansion, but at a lower cost. In addition to the properties that required by MKFHE, such as *Correctness, Compactness, semantic security, Simulatability of decryption*, KL-MKFHE should satisfy the following two additional properties :

- **Locally Computationally Compactness :** A leveled KL-MKFHE is locally computationally compact if the participants do the same number of encryptions as the single-key FHE scheme.
- **Low round complexity :** For k participants, only two round interaction is allowed in Key lifting procedure.

Here we feel compelled to explain that we are not proposing a new definition for the purpose of grandstanding or bells and whistles. The definitions of MKFHE and Th-FHE are good and suitable for many application scenarios. But as we mentioned in the previous subsection, the current schemes do not fit the scenario very well. Strictly classified by definition, the schemes(*scheme#1*) that we give are neither MKFHE (we introduce interactions during initialization) nor Th-FHE (each party uses a different key to encrypt data). That's why we introduced the concept of KL-MKFHE. For more details, please refer to Section 3.1.

Leakage resistance implies a smaller q :

We noticed that, in the partial decryption phase, introducing large noise to cover up the information of the private key, is essentially to ensure the security of the plaintext. But adding noise is just one way to achieve it. In particular, we observe that if the encryption scheme is leakage resistant, the same purpose can be achieved by just increasing the significant bits of private key appropriately.

Assuming that the output length of the circuit to be evaluated is Δ, if no noise flooding is introduced, the information of private key leaked in the partial decryption phase is $\Delta \log q$ bits. Because our private key $\mathbf{s} \in \{0,1\}^\star$ is a binary vector, as long as our encryption scheme is leakage resistant, we only need to add $\Delta \log q$ bits to the length of the original $\mathbf{s}$, which can also ensure the security of the plaintext. Thus, the previous $q = 2^{\lambda L} B_\chi$ in [6] [37] [12] can be reduced to $q = 2^{\lambda + L} B_\chi$ in our scheme. Refer to Section 4.6 for a detailed discussion.

***scheme#1*: LWE-based KL-MKFHE under plain model :**

The security of *scheme#1* is based on the LWE assumption. The master private key is the sum of the private keys of all participants. We note that previous MKFHE or Th-FHE schemes [35] [6] adopt this key structure are all based on the CRS model. Without CRS, our solution is simpler and more efficient in construction : For a circuit with an input length N, our scheme requires local users to perform $O(N)$ encryption operations, while is $O(N\lambda^6 L^4)$ for those schemes that require ciphertext expansion. We bound the participants k by $\mathsf{poly}(\lambda)$, because a larger k will results in longer private key, which further leads to higher memory cost.

We give a comparison with schemes [10] [39] [6] in Table 2. For detailed security and parameters, please refer to Section 4.

***scheme#2*: RLWE-based KL-MKFHE under ROM :**

Same as the scheme in [12], *scheme#2* is based on circular-RLWE. We introduce a bit commitment protocol to guarantee the randomness of each participant's public key. Due to the sum key structure, the dimension of $\mathbf{t} \otimes \mathbf{t}$ is independent of the number of participant, so the ciphertext relinearization algorithm pulls the ciphertext after tensor product back to initial dimension by one shot, in addition, the "one shot algorithm" introduces less noise. We note that, as we mention before, regularity lemma on polynomial ring : $\mathbb{Z}(x)/x^d + 1$ does not

Scheme	Space			Time	Interaction(setup phase)	CRS
	PubKey + EvalKey	Ciphertext	Module q	Extra encryption		
MKFHE [39]	$\tilde{O}(\lambda^6 L^4(k+N\lambda^3 L^2))$	$\tilde{O}(Nk^2\lambda^6 L^4)$	$2^{\lambda L}B_\chi$	$\tilde{O}(N\lambda^{14}L^9)$	-	✓
MKFHE [10]	$\tilde{O}(k^4\lambda^{15}L^{11})$	$\tilde{O}(Nk^4\lambda^8 L^6)$	$2^{\lambda L}B_\chi$	$\tilde{O}(Nk^3\lambda^{15}L^{10})$	2 rounds	
Th-FHE [6]	$\tilde{O}(\lambda^6 L^4)$	$\tilde{O}(N\lambda^6 L^4)$	$2^{\lambda L}B_\chi$	-	1 rounds	✓
scheme#1	PubKey + EvalKey : $\tilde{O}((k\lambda(\lambda+L)+\Delta)\lambda(\lambda+L)^2)$ Ciphertext : $\tilde{O}(N(k\lambda(\lambda+L)+\Delta)^2\lambda^4(\lambda+L)^4)$ Module q : $2^{\lambda+L}B_\chi$			-	2 rounds	

Table 2. The notation $\tilde{O}$ hides logarithmic factors. Space column denotes the bit size of public, evaluation key and ciphertext; the Extra encryption column denotes the number of multiplication operations over $\mathbb{Z}_q$; λ denotes the security parameter, k participants number, B_χ the initial LWE noise; N, L, Δ denotes the input length, depth, and output length of the circuit respectively.

Remark : In the [39] [10] [6], n represents the dimension of the LWE problem, in order to compare under the same security level, we replace n with λ and L by $n = \Omega(\lambda \log q/B_\chi)$. To achieve 2^λ security against known lattice attacks, one must have $n = \Omega(\lambda \log q/B_\chi)$. For our parameter settings $q = 2^{\lambda+L}B_\chi$, thus we would have $n = \Omega(\lambda(\lambda+L))$, while $n = \Omega(\lambda^2 L)$ for the previous scheme by noise flooding.

enjoy the leakage resilient property, we have to introduce smudging noise in partial decryption phase as other RLWE-based MKFHE.

We compared with [12] in terms of memory and computational overhead, the results are shown in Table 3.

Scheme	Space		Time		Interaction(Setup phase)	CRS
	Evalkey	Ciphertext	Relinear	Mult		
MKFHE [12]	$\tilde{O}(kd)$	$\tilde{O}(kd)$	$\tilde{O}(k^2d)$	$\tilde{O}(k^2d)$	-	Yes
scheme#2	$\tilde{O}(kd)$	$\tilde{O}(d)$	$O(1)$	$\tilde{O}(d)$	2 rounds	ROM

Table 3. The notation $\tilde{O}$ hides logarithmic factors, k denotes the number of participants; d denotes the degree of the RLWE problem. The Evalkey and Ciphertext columns denote the asymptotic storage overhead, dominated by k and d. The Relinear and Mult columns denotes the number of scalar operation over $\mathbb{Z}_q$.

2 Preliminaries

2.1 Notation:

Let λ denote security parameter, $\mathsf{negl}(\lambda)$ a negligible function parameterized by λ. Vectors are represented by lowercase bold letters such as $\mathbf{v}$, unless otherwise specified, vectors are row vectors by default, and matrices are represented by uppercase bold letters such as $\mathbf{M}$. $[k]$ denotes the set of integers $\{1, \ldots, k\}$. If X is a distribution, then $a \leftarrow X$ denotes that value a according to the distribution X. If X is a finite set, then $a \leftarrow U(X)$ denotes that the value of a is uniformly sampled from X. For two distribution X, Y, we use $X \overset{\mathsf{stat}}{\approx} Y$ to represent X and Y are statistically indistinguishable ,while $X \overset{\mathsf{comp}}{\approx} Y$ are computationally indistinguishable.

In order to decompose elements in $\mathbb{Z}_q$ into binary, we review the Gadget matrix [32] [3] here, let $\mathbf{G}^{-1}(\cdot)$ be the computable function that for any

$$\mathbf{M} \in \mathbb{Z}_q^{m \times n}, \text{ We have } \mathbf{G}^{-1}(\mathbf{M}) \in \{0,1\}^{ml \times n}, \text{ where } l = \lceil \log q \rceil$$

Let $\mathbf{g} = (1, 2, \ldots, 2^{l-1}) \in \mathbb{Z}_q^l$, $\mathbf{G} = \mathbf{I}_m \otimes \mathbf{g} \in \mathbb{Z}_q^{m \times ml}$, it satisfies $\mathbf{G}\mathbf{G}^{-1}(\mathbf{M}) = \mathbf{M}$.

Definition 1. *A distribution ensemble $\{\mathcal{D}_n\}_{n \in [N]}$ supported over integer, is called B-bounded if :*

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

Lemma 2 (Smudging lemma [6]). *Let $B_1 = B_1(\lambda)$, and $B_2 = B_2(\lambda)$ be positive integers and let $e_1 \in [-B_1, B_1]$ be a fixed integer, let $e_2 \in [-B_2, B_2]$ be chosen uniformly at random, Then the distribution of e_2 is statistically indistinguisable from that of $e_2 + e_1$ as long as $B_1/B_2 = \mathsf{negl}(\lambda)$.*

2.2 The Small Integer Solution(SIS) Problem

The Small Integer Solution(SIS) problem was introduced by Ajtai in the seminal work [1] which presented a family of one-way function based on SIS problem. Subsequent series of works [31] [34] [22] [33] have made efforts to reduce the size of q, the definition below comes from [34]:

Definition 3 (in [34]). *The small integer solution problem $\mathsf{SIS}_{m,n,q,\beta}$ (in the ℓ_∞ norm) is : given an integer q, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a real β, find a nonzero integer vector $\mathbf{z} \in \mathbb{Z}^m / \{\mathbf{0}\}$ such that $\mathbf{A}\mathbf{z}^T = 0 \mod q$ and $||\mathbf{z}||_\infty < \beta$*

[33] proved that solving the $\mathsf{SIS}_{m,n,q,\beta}$ problem is at least as hard as approximating lattice problems in the worst case on lattices :

Theorem 4 (in [33]). *Let n and $m = \mathsf{poly}(n)$ be integers, let β be reals, let $Z = \{\mathbf{z} \in \mathbb{Z}^m : ||\mathbf{z}||_\infty < \beta\}$, and let $q > \beta \cdot n^\delta$ for some constant $\delta > 0$. Then solving (on the average, with non-negligible probability) $\mathsf{SIS}_{m,n,q,\beta}$ with parameters m, n, q, β and solution set $Z/\{\mathbf{0}\}$ is at least as hard as approximating lattice problems in the worst case on n dimensional lattices to within $\gamma = \tilde{O}(\beta\sqrt{n})$.*

2.3 The Learning With Error(LWE) Problem

The Learning With Error problem was introduced by Regev [40].

Definition 5 (Decision-LWE). *Let λ be security parameter, for parameters $n = n(\lambda)$ be an integer dimension, $q = q(\lambda) > 2$ be an integer, and a distribution $\chi = \chi(\lambda)$ over $\mathbb{Z}$, the $\mathsf{LWE}_{n,q,\chi}$ problem is to distinguish the following distribution:*

- *$\mathcal{D}_0$: the jointly distribution $(\mathbf{A}, \mathbf{z}) \in (\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^n)$ is sampled by $\mathbf{A} \leftarrow U(\mathbb{Z}_q^{m\times n})$ $\mathbf{z} \leftarrow U(\mathbb{Z}_q^n)$*
- *$\mathcal{D}_1$: the jointly distribution $(\mathbf{A}, \mathbf{b}) \in (\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^n)$ is computed by $\mathbf{A} \leftarrow U(\mathbb{Z}_q^{m\times n})$ $\mathbf{b} = \mathbf{sA} + \mathbf{e}$, where $\mathbf{s} \leftarrow U(\mathbb{Z}_q^n)$ $\mathbf{e} \leftarrow \chi^m$*

The Decision-$\mathsf{LWE}_{n,q,\chi}$ assumption assuming that $\mathcal{D}_0 \overset{\mathsf{comp}}{\approx} \mathcal{D}_1$. Regev [40] proved that for certain module q and Gaussian error distributions χ the Decision-$\mathsf{LWE}_{n,q,\chi}$ problem is true as long as certain worst case lattice problems are hard to solve using a quantum algorithm.

2.4 The Ring Learning With Error(RLWE) Problem

Lyubaskevsky, Peikert and Regev defines The Decision-RLWE problem in [28] as follows:

Definition 6 (Decision-RLWE). *Let λ be a security parameter. For parameters $d = d(\lambda)$, where d is a power of 2, $q = q(\lambda) > 2$,and a distribution $\chi = \chi(\lambda)$ over $R = \mathbb{Z}[x]/x^d + 1$, let $R_q = R/qR$, the Decision-$\mathsf{RLWE}_{d,q,\chi}$ problem is to distinguish the following distribution:*

- *$\mathcal{D}_0$: the jointly distribution $(a, z) \in R_q^2$ is sampled by $(a, z) \leftarrow U(R_q^2)$.*
- *$\mathcal{D}_1$: the jointly distribution $(a, b) \in R_q^2$ is computed by $a \leftarrow U(R_q)$, $b = as+e$, where $s \leftarrow U(R_q)$, $e \leftarrow \chi$.*

[28] gave a reduction from the $\mathsf{RLWE}_{d,q,\chi}$ problem to the Gap-SVP problem on an ideal lattice, which is now generally considered to be intractable. Specially, [28] indicated that The $\mathsf{RLWE}_{n,q,\chi}$ problem is also infeasible when s is sampled from nosie distribution χ. In homomorphic encryption, this property is especially popular, because the low-norm s introduces less noise during homomorphic computation.

2.5 Dual-GSW(DGSW) Encryption scheme

The DGSW scheme [10] and GSW scheme is similar to Dual-Regev scheme and Regev scheme resp. which is defined as follows:

- $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda, 1^L)$: For a given security parameter λ, circuit depth L, choose a appropriate lattice dimension $n = n(\lambda, L)$, $m = n \log q + \omega(\lambda)$, a discrete Gaussian distribution $\chi = \chi(\lambda, L)$ over $\mathbb{Z}$, which is bouned by B_χ, module $q = \mathsf{poly}(n) \cdot B_\chi$ satisfying the $\mathsf{LWE}_{n,q,\chi,B_\chi}$ problem, Output $\mathsf{pp} = (n, m, q, \chi, B_\chi)$ as the initial parameters.

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$: Let $\mathsf{sk} = \mathbf{t} = (-\mathbf{s}, 1)$, $\mathsf{pk} = (\mathbf{A}, \mathbf{b})$, where $\mathbf{s} \leftarrow U\{0,1\}^{m-1}$, $\mathbf{A} \leftarrow U(\mathbb{Z}_q^{m-1\times n})$, $\mathbf{b} = \mathbf{sA} \mod q$
- $\mathbf{C} \leftarrow \mathsf{Enc}(\mathsf{pk}, u)$: Input public key pk and plaintext u, choose a random matrix $\mathbf{R} \leftarrow U(\mathbb{Z}_q^{n\times w})$, $w = ml$, $l = \lceil \log q \rceil$ and an error matrix $\mathbf{E} \leftarrow \chi^{m\times w}$, Output the ciphertext :

$$\mathbf{C} = \begin{pmatrix} \mathbf{A} \\ \mathbf{b} \end{pmatrix} \mathbf{R} + \mathbf{E} + u\mathbf{G}, \text{ where } \mathbf{G} \text{ is a gadget Matrix.}$$

- $u \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathbf{C})$: Input private key sk, ciphertext $\mathbf{C}$, let $\mathbf{w} = (0, \ldots, \lceil q/2 \rceil) \in \mathbb{Z}_q^m$, $v = \langle \mathbf{tC}, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle$, output $u' = \lceil \frac{v}{q/2} \rceil$.

Homomorphic addition and multiplication : For ciphertext $\mathbf{C}_1$, $\mathbf{C}_2 \in \mathbb{Z}_q^{m\times w}$, let $\mathbf{C}_{\mathsf{add}} = \mathbf{C}_1 + \mathbf{C}_2$, $\mathbf{C}_{\mathsf{mult}} = \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$. It is easy to verify that $\mathbf{C}_{\mathsf{add}}$ and $\mathbf{C}_{\mathsf{mult}}$ are ciphertext of $u_1 + u_2$ and $u_1 u_2$, respectively.

For the security and correctness of the DGSW scheme, please refer to [10]. Compared with the GSW scheme, DGSW scheme has bigger ciphertext, which is $O(n^2 \log^3 q)$, while $O(n^2 \log q)$ for GSW scheme. As [10] mentioned, DGSW scheme makes it more convenient to use the leakage resilient property of LHL to remove CRS.

2.6 Multi-Key Fully Homomorphic Encryption

We review the definition of MKFHE in detail here, the main purpose of which is to compare with the definition of KL-MKFHE we proposed later.

Definition 7. *Let λ be the security parameter, L be the circuit depth, and k be the number of participants. A Leveled multi-key fully homomorphic encryption scheme consists of a tuple of efficient probabilistic polynomial time algorithms* MKFHE=(Init, Gen, Enc, Expand, Eval, Dec) *which defines as follows.*

- $\mathsf{pp} \leftarrow \mathsf{Init}(1^\lambda, 1^L)$ *: Input security parameter λ, circuit depth L, output system paremeter* pp*. We assume that all algorithm take* pp *as input.*
- $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(\mathsf{pp}, \mathsf{id})$ *: Input* pp*, identity* id*, output a key pair for participant i.*
- $c_i \leftarrow \mathsf{Enc}(\mathsf{pk}_i, u_i)$ *: Input* pk_i *and u_i, output ciphertext c_i.*
- $\bar{c}_i \leftarrow \mathsf{Expand}(\mathsf{pk}, c_i)$*:Input the ciphertext c_i of participant i, the public key set* $\mathsf{pk} = \{\mathsf{pk}_i\}_{i\in[k]}$ *of all participants, output expanded ciphertext $\bar{c}_i$ which is under $f(\mathsf{sk}_i, \ldots \mathsf{sk}_k)$ whose structure is undefined.*
- $\bar{c}_{eval} \leftarrow \mathsf{Eval}(\bar{c}, \mathcal{C})$*:Input circuit $\mathcal{C}$, the set of all ciphertext $\bar{c} = \{\bar{c}_1 \ldots \bar{c}_N\}$ while N is the input length of circuit $\mathcal{C}$, output evaluated ciphertext $\bar{c}_{eval}$*
- $u \leftarrow \mathsf{Dec}(\bar{c}_{eval}, f(\mathsf{sk}_1 \ldots \mathsf{sk}_k))$: *Input evaluated ciphertext $\bar{c}_{eval}$, total private key function $f(\mathsf{sk}_1 \ldots \mathsf{sk}_k)$, output u*

Remark :

1. The $\mathsf{Expand}(\cdot)$ algorithm is not necessary. For example, in the RLWE-based MKFHE scheme, the ciphertext expansion procedure is trivial, but in the LWE-based MKFHE scheme, the ciphertext expansion is a complicated and time-consuming process.
2. The private key function $f(\mathsf{sk}_1, \ldots, \mathsf{sk}_k)$ represents an organization form, or a certain function, which is not unique, for example, it can be the concatenation of all keys or the sum of all keys.

Properties implicated in the definition of MKFHE : For the above definition, each participant is required in key generation and encryption phase independently to generates their own keys and completes the encryption operation without interaction between participants. These two phases are similar to single-key homomorphic encryption, the computational overhead is independent of k and only related to λ and L, only in the decryption phase, interaction is involved when participants perform a round of decryption protocol.

3 Key Lifting Multi-key Fully Homomorphic Encryption(KL-MKFHE) scheme

3.1 The definition of KL-MKFHE

In order to cope with *computationally-sensitive* and *trust-sensitive scenarios*, we appropriately *tighten and loosen* the definition of MKFHE, we abandon ciphertext expansion procedure and introduce a **Key lifting** procedure. In addition, a tighter bound is required on the amount of local computation, as a compromise, we allow a small amount of interaction during **Key lifting**.

Definition 8. *A leveled KL-MKFHE scheme is a tuple of probabilistic polynomial time algorithm* (Init, Gen, KeyLifting, Enc, Eval, Dec)*, which can be divided into two phases, online phase:* KeyLifting *and* Dec*, where interaction is allowed between participants, but the rounds should be constant, local phase :* Init*,* Gen*,* Enc*, and* Eval*, whose operations do not involve interaction. These five algorithms are described as follows:*

- $\mathsf{pp} \leftarrow \mathsf{Init}(1^\lambda, 1^L)$*:Input security parameter* λ*, circuit depth* L*, output public parameters* pp*.*
- $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(\mathsf{pp})$*:Input public parameter* pp*, output the key pair of participant* i
- $\mathsf{hk}_i \leftarrow \mathsf{KeyLifting}(\mathsf{pk}_i, \mathsf{sk}_i)$*: Input key pair of participants* i*, output the hybrid key* hk_i *of* i*.*
- $c_i \leftarrow \mathsf{Enc}(\mathsf{hk}_i, u_i)$*: Input plaintext* u_i *and* hk_i*, output ciphertext* c_i
- $\hat{c} \leftarrow \mathsf{Eval}(\mathcal{C}, S)$*: Input circuit* $\mathcal{C}$*, ciphertext set* $S = \{c_i\}_{i\in[N]}$ *, output ciphertext* $\hat{c}$
- $u \leftarrow \mathsf{Dec}(\hat{c}, f(\mathsf{sk}_1 \ldots \mathsf{sk}_k))$*: Input evaluated ciphertext* $\hat{c}$*,* $f(\mathsf{sk}_1 \ldots \mathsf{sk}_k)$*, output* $\mathcal{C}(u_i)_{i\in[N]}$*.*

Remark : KL-MKFHE does not have ciphertext expansion procedure, indeed, the inputed ciphertext set S in $\mathsf{Eval}(\cdot)$ is encrypted by participants under their own hybrid key hk_i which are different among participants, however, the resulting ciphertext c_i supports homomorphic evaluation without extra modification.

we require KL-MKFHE to satisfy the following properties:

Locally Computationally Compactness : *A leveled KL-MKFHE is locally computationally compact if the participants do the same number of encryptions as the single-key FHE scheme.*

Low round complexity : *Only constant round interaction is allow in* $\mathsf{KeyLifting}(\cdot)$ *procedure.*

IND-CPA security of encryption : *Let* λ *be the security parameter,* $L = \mathsf{poly}(\lambda)$ *is the circuit depth, for any probabilistic polynomial time adversary* $\mathcal{A}$*, he can distinguish the following two distributions with negligible advantage.*

$$\Pr\left[\,\mathcal{A}(\mathsf{pp},\mathsf{pk},\mathsf{Enc}(\mathsf{pk},1)) - \mathcal{A}(\mathsf{pp},\mathsf{pk},\mathsf{Enc}(\mathsf{pk},0)) \neq 0\,\right] = \mathsf{negl}(\lambda).$$

Correctness and Compactness : *A leveled KL-MKFHE scheme is correct if for a given security parameter* λ*, circuit depth* L*, participants* k*, we have the following*

$$\Pr\left[\,\mathsf{Dec}(f(\mathsf{sk}_1 \ldots \mathsf{sk}_k), \hat{c}) \neq \mathcal{C}(u_1 \ldots u_N)\,\right] = \mathsf{negl}(\lambda).$$

probability is negligible, where $\mathcal{C}$ *is a circuit with input length* N *and depth length less than* L*. A leveled KL-MKFHE scheme is compact, if the size* $\hat{c}$ *of evaluated ciphertext is bounded by* $\mathsf{poly}(\lambda, L, k)$*, but independent of circuit size.*

4 *scheme#1*: a KL-MKFHE scheme based on DGSW in plain model

Our first scheme is based on DGSW, please refer to Section 2.5 for related background. In this section, we first introduce the key lifting process, then describe the entire scheme, and finally give parameter analysis and security proof.

4.1 Key lifting procedure

Following the definition of KL-MKFHE, it requires the ciphertext encrypted by hybrid key hk which is outputted by $\mathsf{KeyLifting}(\cdot)$ algorithm and different among participants, to support homomorphic evaluation without extra modification. We achieve this property by allowing two round interaction between participants.

Key Lifting

- $\{\mathsf{hk}_i\}_{i\in[k]} \leftarrow \mathsf{KeyLifting}(\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i\in[k]})$: Input the DGSW key pair $\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i\in[k]}$ of all participants, where $\mathsf{pk}_i = (\mathbf{A}_i, \mathbf{b}_{i,i})$, $\mathbf{A}_i \leftarrow U(\mathbb{Z}_q^{(m-1)\times n})$, $\mathbf{s}_i \leftarrow U\{0,1\}^{m-1}$, $\mathbf{b}_{i,i} = \mathbf{s}_i\mathbf{A}_i \mod q$. Assuming there is a broadcast channel, all participants are engaged in the following two interaction :

- First round : i broadcasts $(\mathbf{A}_i, \mathbf{b}_{i,i})$ and receives all $\{\mathbf{A}_j, \mathbf{b}_{j,j}\}_{j\in[k]}$.
- Second round : i generates and disclose $\{\mathbf{b}_{i,j}\}_{j\in[k]}$, where $\mathbf{b}_{i,j} = \mathbf{s}_i\mathbf{A}_j \mod q$

After above two round interaction, i receives $\{\mathbf{b}_{j,i}\}_{j\in[k]}$

$$\text{let } \mathbf{b}_i = \sum_{j=1}^{k} \mathbf{b}_{j,i},\ i \text{ output hybrid key } \mathsf{hk}_i = (\mathbf{A}_i, \mathbf{b}_i)$$

Let $\bar{\mathbf{t}} = (-\mathbf{s}, 1)$, $\mathbf{s} = \sum_{i=1}^{k} \mathbf{s}_i$, for ciphertext $\mathbf{C}_i$, $\mathbf{C}_j$ encrypted by hybrid key hk_i, hk_j respectively :

$$\mathbf{C}_i = \begin{pmatrix} \mathbf{A}_i \\ \mathbf{b}_i \end{pmatrix} \mathbf{R}_1 + \mathbf{E}_1 + u_i\mathbf{G}, \qquad \mathbf{C}_j = \begin{pmatrix} \mathbf{A}_j \\ \mathbf{b}_j \end{pmatrix} \mathbf{R}_2 + \mathbf{E}_2 + u_j\mathbf{G},$$

obviously we have $\bar{\mathbf{t}}\mathbf{C}_i \approx u_i\bar{\mathbf{t}}\mathbf{G}$, $\bar{\mathbf{t}}\mathbf{C}_j \approx u_j\bar{\mathbf{t}}\mathbf{G}$(omit small error). Therefore, although $\mathbf{C}_i$ and $\mathbf{C}_j$ are encrypted by different hybrid keys, they correspond to the same decryption key $\bar{\mathbf{t}}$.

As we will point out later, however, this structure will draw some security concern. First, semi-malicious adversary may generates matrix $\mathbf{A}$ with trapdoor, then $\mathbf{s}$ is leaked. Second, semi-malicious adversary j may generate $\mathbf{b}_{j,i}$ adaptively after seeing $\mathbf{b}_{i,i}$, then the public key $\mathbf{b}_i$ of participant i may not distributed as requirement. We note that as long as our encryption scheme is leakage-resistant, properly lengthening private key $\mathbf{s}$ can guarantee the semantic security of the scheme even if part of $\mathbf{s}$ is leaked. This is the result we derived from [10]. We remedy second problem by increasing the noise bounds in the last row of the noise matrix $\mathbf{E}$. This is the result we derived from [6]. We discuss the security of the scheme in Section 4.5

4.2 The entire scheme

In order to compare the impact of introducing noise flooding, our *scheme#1* prepares two parameter settings, one corresponds to noise flooding and the other corresponds to leakage-resilient scenarios, especially q and key length.
scheme#1 is based on the DGSW scheme, containing the following five algorithm (Init, Gen, KeyLifting, Enc, Eval, Dec)

- $\mathsf{pp} \leftarrow \mathsf{Init}(1^\lambda, 1^L)$: Let λ be security parameter, L circuit depth, Δ circuit output length, lattice dimension $n = n(\lambda, L)$, noise distribution χ over $\mathbb{Z}$, $e \leftarrow \chi$, where $|e|$ is bounded by B_χ with overwhelming probability, modulus $q = 2^{\lambda L}B_\chi$(with noise flooding) or $2^{\lambda+L}B_\chi$, $k = \mathsf{poly}(\lambda)$, $m = kn\log q + \lambda$(with noise flooding) or $(kn + \Delta)\log q + \lambda$, suitable choosing above parameters to make $\mathsf{LWE}_{n,m,q,B_\chi}$ is infeasible. Output $\mathsf{pp} = (k, n, m, q, \chi, B_\chi)$
- $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(\mathsf{pp})$: Input pp, output the DGSW key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$ of participants i, where $\mathsf{pk}_i = (\mathbf{A}_i, \mathbf{b}_{i,i})$, $\mathbf{A}_i \leftarrow U(\mathbb{Z}_q^{(m-1)\times n})$, $\mathbf{s}_i \leftarrow U\{0,1\}^{m-1}$, $\mathbf{b}_{i,i} = \mathbf{s}_i\mathbf{A}_i \mod q$.

- $\mathsf{hk}_i \leftarrow$ **Key Lifting** : All participants are engaged in the **Key lifting procedure 4.1**, output the hybrid key hk_i.
- $\mathbf{C}_i \leftarrow \mathsf{Enc}(\mathsf{hk}_i, u_i)$: Input hybrid key hk_i, plaintext u_i, output ciphertext $\mathbf{C}_i = \begin{pmatrix} \mathbf{A}_i \\ \mathbf{b}_i \end{pmatrix} \mathbf{R} + \mathbf{E} + u_i \mathbf{G}$, where $\mathbf{R} \leftarrow \chi^{n \times ml}$, $l = \lceil \log q \rceil$, $\mathbf{E} = \begin{pmatrix} \mathbf{E}_0 \\ \mathbf{e}_1 \end{pmatrix}$, $\mathbf{E}_0 \leftarrow \chi^{(m-1)\times ml}$, $\mathbf{e}_1 \leftarrow \chi'^{\,ml}$, χ' is a distribution over $\mathbb{Z}$, satisfying $||\mathbf{e}_1||_\infty$ is bounded by $2^{\lambda^{\epsilon_1}} B_\chi$, $\epsilon_1 \in (0, \frac{1}{2})$, $\mathbf{G} = \mathbf{I}_m \otimes \mathbf{g}$ is a gadget matrix.
- $\hat{\mathbf{C}} \leftarrow \mathsf{Eval}(S, \mathcal{C})$: Input the ciphertext $S = \{\mathbf{C}_i\}_{i\in[N]}$ which are encrypted by hybrid key $\{\mathsf{hk}_i\}_{i\in[k]}$, circuit $\mathcal{C}$ with input length N, output $\hat{\mathbf{C}}$.

Homomorphic addition and multiplication

Let $\bar{\mathbf{t}} = (\sum_{i=1}^{k} \mathbf{s}_i, -1)$

- $\mathbf{C}_{\mathsf{add}} \leftarrow \mathsf{Add}(\mathbf{C}_1, \mathbf{C}_2)$: Input ciphertext $\mathbf{C}_1$, $\mathbf{C}_2$, output $\mathbf{C}_{\mathsf{add}} = \mathbf{C}_1 + \mathbf{C}_2$, Obviously $\bar{\mathbf{t}}\mathbf{C}_{\mathsf{add}} \approx (u_1 + u_2)\bar{\mathbf{t}}\mathbf{G}$
- $\mathbf{C}_{\mathsf{mult}} \leftarrow \mathsf{Mult}(\mathbf{C}_1, \mathbf{C}_2)$: Input ciphertext $\mathbf{C}_1$, $\mathbf{C}_2$, output $\mathbf{C}_{\mathsf{mult}} = \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$, Obviously $\bar{\mathbf{t}}\mathbf{C}_{\mathsf{mult}} \approx u_1 u_2 \bar{\mathbf{t}}\mathbf{G}$

Distributed decryption Similar to [37], the decryption procedure is a distributed procedure :

- $\mathsf{LocalDec}$: Input $\hat{\mathbf{C}}$, let $\hat{\mathbf{C}} = \begin{pmatrix} \mathbf{C}_0 \\ \mathbf{c}_1 \end{pmatrix}$, where $\mathbf{C}_0 \in \mathbb{Z}_q^{(m-1)\times ml}$, $\mathbf{c}_1 \in \mathbb{Z}_q^{ml}$, i computes $\beta_i = \langle \mathbf{s}_i,\ \mathbf{C}_0\mathbf{G}^{-1}(\mathbf{w}^T) \rangle$, and set $\gamma_i = \beta_i + e_i''$(with flooding noise) or $\gamma_i = \beta_i$, where $\mathbf{w} = (0, \ldots, 0, \lceil q/2 \rceil) \in \mathbb{Z}_q^m$, $e_i'' \leftarrow \chi''$ is a distribution over $\mathbb{Z}$, satisfying $|e_i''| < 2^{L\lambda^{\epsilon_2}} B_\chi$, $\epsilon_2 \in (\frac{1}{2}, 1)$, then i broadcast γ_i
- $\mathsf{FinalDec}$: After received $\{\gamma_i\}_{i\in[k]}$, let $\gamma = \sum_{i=1}^{k} \gamma_i + \langle \mathbf{c}_1,\ \mathbf{G}^{-1}(\mathbf{w}^T) \rangle$, output $u = \lceil \frac{\gamma}{q/2} \rfloor$

4.3 Bootstrapping

In order to eliminate the dependence on the circuit depth to achieve fully homomorphism, we need to use Gentry's bootstrapping technology. It is worth noting that the bootstrapping procedure of *scheme#1* is the same as single-key homomorphic scheme: After **Key lifting** procedure, participant i uses hybrid key hk_i to encrypt s_i to obtain evaluation key evk_i. Because evk_i and $\hat{\mathbf{C}}$ are both ciphertexts under $\bar{\mathbf{t}} = (-\sum_{i=1}^{k} \mathbf{s}_i, 1)$, homomorphic evaluation of the decryption circuit could be executed directly as $\hat{\mathbf{C}}$ are need to be refresh. Therefore, in order to evaluate any depth circuit, we only need to set the initial parameters to satisfy the homomorphic evaluation of the decryption circuit.

However, for those MKFHE schemes that requires ciphertext expansion, additional ciphertext expansion is required, for the reason that $\hat{\mathbf{C}}$ is the ciphertext under $\bar{\mathbf{t}}$, but $\{\mathsf{evk}_i\}_{i\in[k]}$ are the ciphertext under $\{\mathbf{t}_i\}_{i\in[k]}$. This is another large amount of computational overhead, because in order to expand $\{\mathsf{evk}_i\}_{i\in[k]}$, participant i needs to encrypt the random matrix of the ciphertext corresponding to evk_i.

4.4 Correctness analysis

To illustrate the correctness of *scheme#1*, we first study the accumulation of noise:

$$\text{Let } \mathbf{s} = \sum_{i=1}^{k} \mathbf{s}_i,\ \mathbf{t} = (-\mathbf{s}, 1),\ \text{for fresh ciphertext } \mathbf{C} = \begin{pmatrix} \mathbf{A}_i \\ \mathbf{b}_i \end{pmatrix} \mathbf{R} + \begin{pmatrix} \mathbf{E}_0 \\ \mathbf{e}_1 \end{pmatrix} + u\mathbf{G}$$

we have $\mathbf{tC} = \mathbf{e}_1 + \mathbf{sE}_0 + u\mathbf{tG}$, let $\mathbf{e}_{\mathsf{init}} = \mathbf{e}_1 + \mathbf{sE}_0$, Obviously $||\mathbf{e}_{\mathsf{init}}||_\infty < (2^{\lambda^{\epsilon_1}} + km)B_\chi$.

After L depth circuit evaluation, let $\mathbf{e}_L = (ml)^L \mathbf{e}_{\mathsf{init}}$. According to the distributed decryption of *scheme#1* we have :

– with noise flooding $q = 2^{\lambda L} B_\chi$.

$$\gamma = \sum_{i=i}^{k} \gamma_i + \mathbf{c}_1 \mathbf{G}^{-1}(\mathbf{w}^T) = \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle + \sum_{i=1}^{k} e_i'' + u\lfloor \frac{q}{2} \rceil \tag{1}$$

Obviously $|e_i''| > \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle$, for taking the logarithm of both sides:

$$\log e_i'' = \lambda^{\epsilon_2} L \tag{2}$$

$$\log \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle = \log((kn\lambda^2 L^2)^L (2^{\lambda^{\epsilon_1}} + k^2 n\lambda L)) = O(L + \lambda^{\epsilon_1}) \tag{3}$$

Let :

$$e_{\mathsf{final}} = \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle + \sum_{i=1}^{k} e_i''$$

Thus :

$$e_{\mathsf{final}} < \sum_{i=1}^{k+1} e_i'' = \mathsf{poly}(\lambda) \cdot 2^{\lambda^{\epsilon_2} L} B_\chi$$

– without noise flooding $q = 2^{\lambda + L} B_\chi$.

$$\gamma = \sum_{i=i}^{k} \gamma_i + \mathbf{c}_1 \mathbf{G}^{-1}(\mathbf{w}^T) = \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle + u\lfloor \frac{q}{2} \rceil \tag{4}$$

$e_{\mathsf{final}} = \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle$, $\log e_{\mathsf{final}} = O(L + \lambda^{\epsilon_2})$.

In order to decrypt correctly, it requires $e_{\mathsf{final}} < \frac{q}{4}$. For our parameter settings, with noise flooding $q = 2^{\lambda L} B_\chi$, $m = kn \log q + \lambda$, or without $q = 2^{\lambda + L} B_\chi$, $m = (kn + \Delta) \log q + \lambda$, requirements are fulfilled.

4.5 Semantic Security of Encryption against Semi-Malicious Adversary

We assume that the adversary is semi-malicious, that is to say, he can generate parameters adaptively and does not need to strictly execute the steps of the protocol. For a more formal definition, please refer to [6]. First, we prove that DGSW is leakage-resilient, and second, we prove *scheme#1*'s semantic security.

DGSW is leakage-resilient

The DGSW scheme and GSW scheme is similar to Dual-Regev scheme and Regev scheme resp. [10] has proved that it is leakage-resilient, here, for completeness, we repeat it. Let χ be LWE noise distribution bounded by B_χ, χ' a distribution over $\mathbb{Z}$ bounded by $B_{\chi'}$, satisfying $B_\chi / B_{\chi'} = \mathsf{negl}(\lambda)$.

Lemma 9 (in [10]). *Let $\mathbf{A}_i \in \mathbb{Z}_q^{(m-1)\times n}$ be uniform, and let $\mathbf{A}_j$ for all $j \neq i$ be chosen by a rushing adversary after seeing $\mathbf{A}_i$. Let $\mathbf{s}_i \leftarrow \{0,1\}^{m-1}$ and $\mathbf{b}_{i,j} = \mathbf{s}_i \mathbf{A}_j$. Let $\mathbf{r} \in \mathbb{Z}_q^n$ be uniform, $\mathbf{e} \leftarrow \chi^{m-1}$, $e' \leftarrow \chi'$. Then under the LWE assumption, the vector $\mathbf{c} = \mathbf{A}_i \mathbf{r} + \mathbf{e}$ and number $c' = \langle \mathbf{b}_{i,i}, \mathbf{r} \rangle + e'$ are (jointly) pseudorandom, even given the $\mathbf{b}_{i,j}$'s for all $j \in [k]$ and the view of the adversary that generated the $\mathbf{A}_j$'s.*

We omit the proof here, more details please refer to [10].

The semantic security of *scheme#1*

For a honest player i, he generates $\mathbf{A}_i \leftarrow U(\mathbb{Z}_q^{(m-1)\times n})$, $\mathbf{b}_{i,j} \leftarrow \{0,1\}^n$ as the protocol specification, but a semi-malicious adversary may generates it arbitrarily and adaptively. For arbitrary $\mathbf{A}_i$, the leakage-resilient property of DGSW guarantees the semantic security. Here, we deal with what happens when $\mathbf{b}_{i,j}$ generated adaptively. Note that in addition to an output tape, a semi-malicious adversary also has a witness tape, whenever the adversary produces a new protocol message m, it must also write to its witness tape some pair (x, r) of input x and randomness r.

In *scheme#1*, we require that for each $\mathbf{b}_{i,j}$, participant i must know what the output $\mathbf{s}_i$ on its corresponding witness tape is, and $||\mathbf{s}_i||_\infty$ is bounded by B_{sis}. Let B_{sis} be the bound keeping the $\mathsf{SIS}_{m,n,q,B_{\mathsf{sis}}}$ problem hard, according to Theorem 4, if $B_{\mathsf{sis}} \ll q^{n/m}$, the problem is vacuously hard, most likely, such solutions do not exists, if $B_{\mathsf{sis}} \gg \gamma^m \cdot q^{n/m}$, this is an instance of $\mathsf{approx\text{-}SVP}$ with exponential approximation factor γ, which can be solved by LLL [26], somewhere in between these bounds is where cryptography takes place, typically for $B_{\mathsf{sis}} = q^{n/m} \cdot \mathsf{poly}(\lambda)$. For our parameter Settings $B_{\mathsf{sis}} = q^{n/m} \cdot \mathsf{poly}(\lambda) = \mathsf{poly}(\lambda)$. We complete the simulation by constructing a reduction from *scheme#1* to the DGSW scheme. Consider the following Game:

1. Challenger generates $\mathsf{pk}_{\mathsf{DGSW}} = (\mathbf{A}, \mathbf{b}_1)$ where $\mathbf{A} \leftarrow U(\mathbb{Z}_q^{(m-1)\times n})$, $\mathbf{b}_1 = \mathbf{s}_1 \mathbf{A}$, $\mathbf{s}_1 \leftarrow U\{0,1\}^{m-1}$ and send $\mathsf{pk}_{\mathsf{DGSW}}$ to adversary $\mathcal{A}$

2. $\mathcal{A}$ adaptively chooses $\{\mathbf{b}_i\}_{i\in[k]/1}$ where $\mathbf{b}_i = \mathbf{s}_i\mathbf{A}$ and $||\mathbf{s}_i||_\infty < B_{\mathsf{sis}}$ after seeing $\mathsf{pk}_{\mathsf{DGSW}}$, chooses a bit $u \in \{0,1\}$ and sets $\mathsf{hk}_{scheme\#1} = (\mathbf{A}, \mathbf{b})$, where $\mathbf{b} = \sum_{i=1}^{k} \mathbf{b}_i$, then send $\mathsf{hk}_{scheme\#1}$ and u to Challenger.
3. Challenger chooses a bit $\alpha \in \{0,1\}$, if $\alpha = 0$, set $\mathbf{C}_{scheme\#1} \leftarrow \mathsf{Enc}(\mathsf{hk}_{scheme\#1}, u)$, otherwise $\mathbf{C}_{scheme\#1} \leftarrow U(\mathbb{Z}_q^{m\times ml})$, and send $\mathbf{C}_{scheme\#1}$ to $\mathcal{A}$
4. After receiving $\mathbf{C}_{scheme\#1}$, $\mathcal{A}$ output bit $\bar{\alpha}$, if $\bar{\alpha} = \alpha$, then $\mathcal{A}$ wins.

Lemma 10. *Let* $\mathsf{Adv} = |Pr[\bar{\alpha} = \alpha] - \frac{1}{2}|$ *denote* $\mathcal{A}$*'s advantage in winning the game, If* $\mathcal{A}$ *can win the game with advantage* Adv*, then* $\mathcal{A}$ *can distinguish between the ciphertext distribution of* DGSW *and the uniform random distribution with the same advantage.*

Proof. We construct $\mathbf{C}_{scheme\#1}$ by DGSW.$\mathsf{Enc}(\mathsf{pk}_{\mathsf{DGSW}}, u)$:

1. First, Challenger generates $\mathsf{pk}_{\mathsf{DGSW}}$ like the step 1 of above Game, set $\mathbf{C}_{\mathsf{DGSW}} = \mathsf{DGSW.Enc}(\mathsf{pk}_{\mathsf{DGSW}}, u)$ send the both to $\mathcal{A}$.
2. $\mathcal{A}$ generates $\{\mathbf{s}_i\}_{i\in[k]/1}$, let $\mathbf{s}' = \sum_{i=2}^{k} \mathbf{s}_i$, $\mathbf{C}_{\mathsf{DGSW}} = \begin{pmatrix} \mathbf{C}_0 \\ \mathbf{c}_1 \end{pmatrix}$, $\mathbf{C}' = \begin{pmatrix} \mathbf{C}_0 \\ \mathbf{c}_1 + \mathbf{c}_1' \end{pmatrix}$, where $\mathbf{c}_1' = \mathbf{s}'\mathbf{C}_0$, obviously $\mathbf{C}' = \begin{pmatrix} \mathbf{A} \\ \mathbf{b} \end{pmatrix} \mathbf{R} + \begin{pmatrix} \mathbf{E}_0 \\ \mathbf{e}_1 + \mathbf{s}'\mathbf{E}_0 \end{pmatrix}$.

For our parameter settings $||\mathbf{e}_1||_\infty < 2^{\lambda^{\epsilon_1}} B_\chi$, $||\mathbf{s}'\mathbf{E}_0||_\infty < kmB_\chi B_{\mathsf{sis}}$, thus $\mathbf{s}'\mathbf{E}_0/\mathbf{e}_1 = \mathsf{negl}(\lambda)$, we have $\mathbf{C}' \overset{\mathsf{stat}}{\approx} \mathbf{C}_{scheme\#1}$, if $\mathcal{A}$ can distinguish between $\mathbf{C}_{scheme\#1}$ and uniform random distribution by advantage Adv, then he can distinguish between DGSW.$\mathsf{Enc}(\mathsf{pk}_{\mathsf{DGSW}}, u)$ and the uniform random distribution with the same advantage.

Remark: we require k to be bounded by $\mathsf{poly}(\lambda)$, because if a larger k is introduced, it will lead to a larger smudging error, which further leads to a larger q. For our choice of $q = 2^{\lambda L} B_\chi$(with noise flooding) or $q = 2^{\lambda+L} B_\chi$, the corresponding approximation factor of the SVP problem is $\tilde{O}(2^{\lambda L})$ or $\tilde{O}(2^{\lambda+L})$

4.6 Noise flooding technology VS. Leakage resilient property in partial decryption

We note that the introduction of noise flooding in the partial decryption phase is essentially to guarantee the semantic security of fresh ciphertext, and noise flooding achieves this by masking the private key information hidden in the partial decryption noise. For partial decryption to be simulatable, the magnitude of the noise introduced needs to be exponentially larger than the noise after the homomorphic evaluation. At the same time, as mentioned in [37], masking techniques based on noise flooding can only guarantee weak simulatable properties : input all private keys $\{\mathsf{sk}_j\}_{j\in[k]/i}$ except sk_i, evaluated result u_{eval}, ciphertext $\hat{\mathbf{C}}$, it can simulate the local decryption result γ_i, while for stronger security requirements : input any private key set $\{\mathsf{sk}_j\}_{j\in S}$, S is any subset of $[k]$, evaluated

result u_{eval} and ciphertext $\hat{\mathbf{C}}$, to simulate $\{\gamma_i\}_{i\in U,\ U=[k]-S}$, it don't know how to achieve it.

To illustrate how our approach works, let's first look at how the noise flooding technique works. During the partial decryption phase:

$$\gamma_i = \langle \mathbf{s}_i, \hat{\mathbf{C}} \rangle + e_i'' = u\lfloor \frac{q}{2} \rceil + \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle + \sum_{j\neq i}^{k} e_j'' - \langle \mathbf{c}_1, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle - \sum_{j\neq i}^{k} \gamma_j$$

For a simulator $\mathcal{S}$, input $\{\mathsf{sk}_j\}_{j\in[k]/i}$, evaluated result u, ciphertext $\hat{\mathbf{C}}$, output simulated γ_i'

$$\gamma_i' = u\lfloor \frac{q}{2} \rceil + \sum_{j\neq i}^{k} e_j'' + \langle \mathbf{c}_1, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle - \sum_{j\neq i}^{k} \gamma_j.$$

In order to make the partial decryption process simulatable :

$$\langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T) \rangle + \sum_{j\neq i}^{k} e_j'' \overset{\mathsf{stat}}{\approx} \sum_{j\neq i}^{k} e_j''$$

is required. In short, the noise e_i'' is introduced to cover up some information(private key and the noise in initial ciphertext) of participant i contained in $\mathbf{e}_L$, thus the partial decryption result of participant i can be simulated, providing other participants information.

To illustrate what information is contained in $\mathbf{e}_L$, let's look at how $\mathbf{e}_L$ is generated. For the initial DGSW ciphertext, we have :

$$\mathbf{C}_1 = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{b}_1 \end{pmatrix} \mathbf{R}_1 + \mathbf{E}_1 + u_1\mathbf{G}, \qquad \mathbf{C}_2 = \begin{pmatrix} \mathbf{A}_2 \\ \mathbf{b}_2 \end{pmatrix} \mathbf{R}_2 + \mathbf{E}_2 + u_2\mathbf{G},$$

After performing a homomorphic multiplication operation, we obtain:

$$\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2) = \left[\begin{pmatrix} \mathbf{A}_1 \\ \mathbf{b}_1 \end{pmatrix} \mathbf{R}_1 + \mathbf{E}_1 + u_1\mathbf{G} \right] \mathbf{G}^{-1}(\mathbf{C}_2)$$

$$= \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{b}_1 \end{pmatrix} \mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) + \mathbf{E}_1\mathbf{G}^{-1}(\mathbf{C}_2) + u_1 \begin{pmatrix} \mathbf{A}_2 \\ \mathbf{b}_2 \end{pmatrix} \mathbf{R}_2 + u_1\mathbf{E}_2 + u_1u_2\mathbf{G}$$

Let :

$$\Pi_1 = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{b}_1 \end{pmatrix} \mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) + u_1 \begin{pmatrix} \mathbf{A}_2 \\ \mathbf{b}_2 \end{pmatrix} \mathbf{R}_2$$

$$\delta_1 = \mathbf{E}_1\mathbf{G}^{-1}(\mathbf{C}_2) + u_1\mathbf{E}_2$$

we have $\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2) = \Pi_1 + \delta_1 + u_1u_2\mathbf{G}$. δ_1 is the noise after the first homomorphic evaluation. For the ciphertexts $\mathbf{C}_3, \mathbf{C}_4$ of the same level, we have $\mathbf{C}_3\mathbf{G}^{-1}(\mathbf{C}_4) = \Pi_1' + \delta_1' + u_3u_4\mathbf{G}$, where Π_1', δ_1' and Π_1, δ_1 have the same structure.

Let:

$$\mathbf{C}_{L_1} = \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2), \qquad \mathbf{C}'_{L_1} = \mathbf{C}_3\mathbf{G}^{-1}(\mathbf{C}_4)$$
$$\delta_2 = \delta_1\mathbf{G}^{-1}(\mathbf{C}'_{L_1}) + u_1u_2\delta'_1$$

we have $\mathbf{C}_{L_1}\mathbf{G}^{-1}(\mathbf{C}'_{L_2}) = \mathit{\Pi}_2 + \delta_2 + u_1u_2u_3u_4\mathbf{G}$. Therefore, for the ciphertext at level L, we have :

$$\mathbf{C}_L = \mathbf{C}_{L-1}\mathbf{G}^{-1}(\mathbf{C}'_{L-1}) = \mathit{\Pi}_L + \delta_L + u_{L-1}u'_{L-1}\mathbf{G}$$
$$\delta_L = \delta_{L-1}\mathbf{G}^{-1}(\mathbf{C}'_{L-1}) + u_{L-1}\delta'_{L-1}$$

To find out what information δ_L contains, first, we observe $\delta_1 = \mathbf{E}_1\mathbf{G}^{-1}(\mathbf{C}_2)+ u_1\mathbf{E}_2$. Here, we prove :

$$\delta_1 \overset{\mathsf{stat}}{\approx} \mathbf{E}_1\mathbf{G}^{-1}(\mathbf{C}_2)$$

Proof. First, according to the LWE assumption, replace $\mathbf{G}^{-1}(\mathbf{C}_2)$ with $\mathbf{M} \leftarrow \mathbf{U}\{0,1\}^{ml\times ml}$. When $u_1 = 0$, it is proved, otherwise $u_1 = 1$, let $\delta_1(i,j)$ be the i-th row, j-th column element of δ_1. We have :

$$\delta_1(0,0) = z_1e_1 + z_2e_2 + \cdots + z_{ml}e_{ml} + e_{ml+1}$$
$$\mathbf{E}_1\mathbf{M}(0,0) = z_1e_1 + z_2e_2 + \cdots + z_{ml}e_{ml}$$

where $\{z_i\}_{i\in[ml]} \leftarrow \{0,1\}$, $(e_1, \cdots, e_{ml})$ is the first row of $\mathbf{E}_1$, $e_{ml+1} = \mathbf{E}_2(0,0)$, and $\{e_i\}_{i\in[ml]} \leftarrow N(0,\sigma^2)$. Suppose, the number of 1s in $\{z_i\}_{i\in[ml]}$ is ϵ. Thus, we have :

$$\delta_1(0,0) \sim N(0,(\epsilon+1)\sigma^2)$$
$$\mathbf{E}_1\mathbf{M}(0,0) \sim N(0,\epsilon\sigma^2)$$

The respective probability density functions of $\delta_1(0,0)$ and $\mathbf{E}_1\mathbf{M}(0,0)$ are f_{δ_1}, $f_{\mathbf{E}_1\mathbf{M}}$:

$$f_{\delta_1} = \frac{1}{2\pi\sqrt{\epsilon+1}\sigma}\exp\{-\frac{x^2}{2(\epsilon+1)\sigma^2}\}$$
$$f_{\mathbf{E}_1\mathbf{M}} = \frac{1}{2\pi\sqrt{\epsilon}\sigma}\exp\{-\frac{x^2}{2\epsilon\sigma^2}\}$$

Let $f_{\delta_1} = f_{\mathbf{E}_1\mathbf{M}}$, the solution $x = \sqrt{\epsilon(\epsilon+1)\ln\frac{\epsilon+1}{\epsilon}}\sigma$. The statistical distance of $\delta_1(0,0)$ and $\mathbf{E}_1\mathbf{M}(0,0)$ is :

$$SD(\delta_1(0,0), \mathbf{E}_1\mathbf{M}(0,0)) = \int_{-x}^{x} f_{\mathbf{E}_1\mathbf{M}} - f_{\delta_1}dx$$
$$= 2\int_{-\infty}^{-x} f_{\delta_1} - f_{\mathbf{E}_1\mathbf{M}}dx$$
$$< 2\int_{-\infty}^{-x} f_{\delta_1}dx$$

By the Lemma 4 in [2].

Lemma 11 (in [2]). *Let χ denote the Gaussian distribution with standard deviation σ and mean zero. Then, for all $C > 0$, it holds that:*

$$\Pr[e \leftarrow \chi : |e| > C \cdot \sigma] \leq \frac{2}{C\sqrt{2\pi}} \exp\{-\frac{C^2}{2}\}.$$

Let $C = \sqrt{\epsilon(\epsilon+1)\ln\frac{\epsilon+1}{\epsilon}}$, We have :

$$\begin{aligned} 2\int_{-\infty}^{-x} f_{\delta_1} dx &< \frac{2}{C\sqrt{2\pi}} \exp\{-\frac{C^2}{2}\} \\ &= \frac{2}{C\sqrt{2\pi}} \exp\{-\frac{1}{2}\epsilon(\epsilon+1)\ln\frac{\epsilon+1}{\epsilon}\} \\ &= \frac{2}{C\sqrt{2\pi}} \exp\{-\frac{\epsilon+1}{2}\} \end{aligned}$$

On average, the number of 1s in $\{z_i\}_{i\in[ml]}$ is $ml/2$, thus statistical distance of $\delta_1(0,0)$ and $\mathbf{E}_1\mathbf{M}(0,0)$:

$$SD(\delta_1(0,0), \mathbf{E}_1\mathbf{M}(0,0)) < \frac{2}{C\sqrt{2\pi}} \exp\{-\frac{ml+1}{4}\} = \mathsf{negl}(\lambda).$$

for $ml = (kn + \Delta)\lambda(\lambda + L)$, we completed the proof. For other item of $\delta_1(i,j)$ and $\mathbf{E}_1\mathbf{M}(i,j))$ the statement also holds.

According to the results we proved above, we point out that the multiplication of DGSW is asymmetric, that is, the noise $\mathbf{E}_2$ of the right ciphertext $\mathbf{C}_2$ in the ciphertext $\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$ is masked by the noise $\mathbf{E}_1$ in the left ciphertext $\mathbf{C}_1$. Similarly, the noise $\mathbf{E}_4$ of $\mathbf{C}_4$ in $\mathbf{C}_3\mathbf{G}^{-1}(\mathbf{C}_4)$ is masked by the noise $\mathbf{E}_3$ of $\mathbf{C}_3$ on the leftside. For the noise $\delta_2 = \delta_1\mathbf{G}^{-1}(\mathbf{C}'_{L_1}) + u_{L_1}\delta'_1$ of the second level, δ'_1 is masked by δ_1, and similarly the noise $\delta_L = \delta_{L-1}\mathbf{G}^{-1}(\mathbf{C}'_{L-1}) + u_{L-1}\delta'_{L-1}$ of the L-th level, δ'_{L-1} is masked by δ_{L-1}. We illustrate this continuous process in Figure 1.

If the input length of circuit is N and the depth is L, as long as $L > \log N$, then the noise δ_L of the ciphertext $\mathbf{C}_L$ of the L-th level only contains the information of noise $\mathbf{E}_t(t \in [N])$ in a certain initial ciphertext. At this point, we only need to left-multiply $\mathbf{C}_L$ by a ciphertext $Enc(1)$ whose plaintext is 1, and let $\mathbf{C}_{clear} = Enc(1)\mathbf{G}^{-1}(\mathbf{C}_L)$. Thus, the noise δ_{clear} in $\mathbf{C}_{clear}$ does not contain any information about the noise $\{\mathbf{E}_i\}_{i\in[N]}$ in the initial ciphertext $\{\mathbf{C}_i\}_{i\in[N]}$

Decrypting $\mathbf{C}_{clear}$, we have :

$$\bar{\mathbf{t}}\mathbf{C}_{clear} = \bar{\mathbf{t}}\delta_{clear} + u_L\bar{\mathbf{t}}\mathbf{G}.$$

Let $\mathbf{e}_L = \bar{\mathbf{t}}\delta_{clear}$, therefore, $\langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle \in \mathbb{Z}_q$ leaks participant i's private key $\mathbf{s}_i$ with at most $\log q$ bits. For a circuit with output length Δ, the entire partial decryption leaks $\Delta \log q$ bits of $\mathbf{s}_i$. Because the DGSW scheme is leakage-resilient, as long as we set the key length reasonably $m = (kn + \Delta)\log q + \lambda$, the initial ciphertext $\{\mathbf{C}_i\}_{i\in[N]}$ is semantically secure.

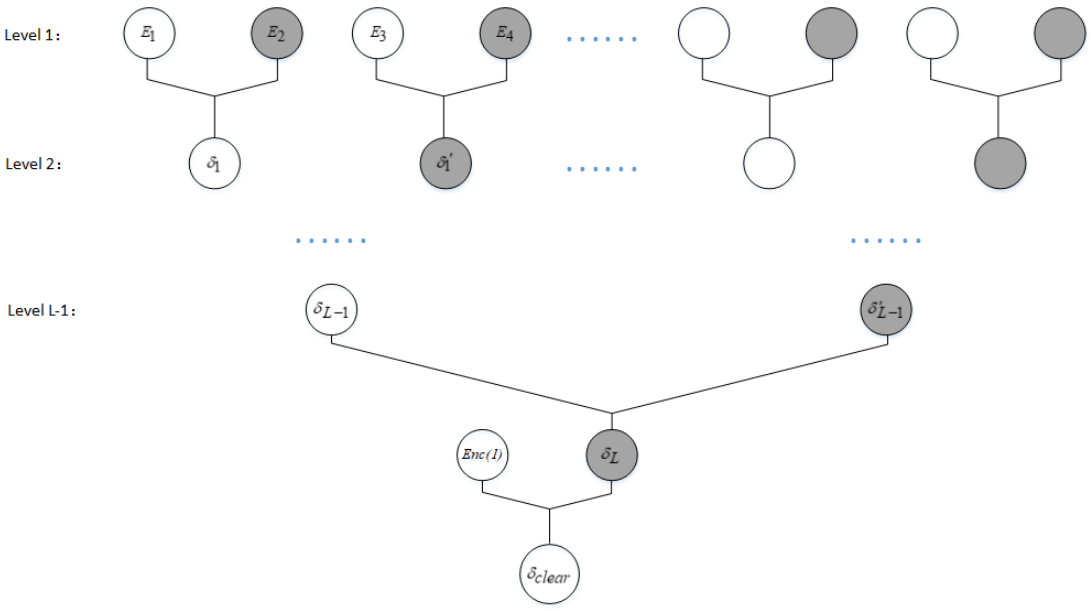

Fig. 1. circuit

Here, the reader might think that doing so would result in a key that is longer than using noise flooding. We point out that as long as the output length Δ of circuit satisfies $\Delta < \frac{k\lambda^3L^2-k\lambda(\lambda+L)^2}{\lambda+L}$, the length of the private key will not be longer than when using noise flooding. Let $m' = (kn+\Delta)\log q'+\lambda$, $q' = 2^{\lambda+L}B_\chi$, while with noise flooding $m = kn\log q+\lambda$, $q = 2^{\lambda L}B_\chi$. In order to make $m' < m$, only $\Delta < \frac{k\lambda^3L^2-k\lambda(\lambda+L)^2}{\lambda+L}$ is required, thus for circuits with small output fields, our scheme does not lead to longer keys.

Here, for completeness and for comparison with no noise flooding, we give a proof of the simulability of partial decryption of *scheme#1* with noise flooding introduced. According to equation 1 we have $\gamma = \sum_{i=1}^{k}\gamma_i + \langle \mathbf{c}_1, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle$

$$\text{thus } \gamma_i = u\lfloor\frac{q}{2}\rceil + \langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle + \sum_{i=1}^{k} e_i'' - \langle \mathbf{c}_1, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle - \sum_{j\neq i}^{k}\gamma_j$$

For a simulator $\mathcal{S}$, input $\{\mathsf{sk}_j\}_{j\in[k]/i}$, evaluated result u, ciphertext $\hat{\mathbf{C}}$, output simulated γ_i'

$$\gamma_i' = u\lfloor\frac{q}{2}\rceil + \sum_{i=1}^{k} e_i'' + \langle \mathbf{c}_1, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle - \sum_{j\neq i}^{k}\gamma_j.$$

For our parameter settings, we have :

$$|\sum_{i=1}^{k} e_i''| < k \cdot 2^{L\lambda^{\epsilon_2}} B_\chi$$

$$\langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle = (kn\lambda^2 L^2)^L (2^{\lambda^{\epsilon_1}} + k^2 n\lambda L) B_\chi = 2^{O(L+\lambda^{\epsilon_1})}$$

$$\text{thus } |\langle \mathbf{e}_L, \mathbf{G}^{-1}(\mathbf{w}^T)\rangle / \sum_{i=1}^{k} e_i''| = \mathsf{negl}(\lambda)$$

we have $\gamma_i \overset{\mathsf{stat}}{\approx} \gamma_i'$.

5 *scheme#2*: KL-MKFHE based on RLWE in ROM

It is regrettable that general polynomial ring $R : \mathbb{Z}[x]/f(x)$ cannot enjoy the leak resilient property of the LHL on the integer ring $\mathbb{Z}$. This means that we cannot transplant the above construction process trivially to RLWE-based FHE. Indeed, [16] pointed out that for $\mathbf{x} = (x_1, \ldots, x_l) \in R^l$, if the j-th NTT coordinate of each $x_{i,i\in[l]}$ is leaked, then the j-th NTT coordinate of $a_{l+1} = \sum_{i=1}^{l} a_i x_i$ is defined, thus a_{l+1} is far from random, although the leakage ratio is only 1/n. We also noticed a trivial solution : for $\mathbf{a}, \mathbf{s} \in R_q^l$, $b = \langle \mathbf{a}, \mathbf{s}\rangle \in R_q$, b leaks information about $\mathbf{s}$ at most $n \log q$ bits, therefore, as long as we set l long enough, for example, $l = l + n \log q$, then obviously b is close to uniformly random, but this will result in a extremely large key, thus it is not practical.

To ensure the independence of the $\{a_i\}_{i\in[k]}$ generated by each participant, we simply added a round of bit commitment protocol. Under the ROM, the cryptographic hash function is used to ensure the independence of $\{a_i\}_{i\in[k]}$. Let $H : \{0,1\}^\star \to \{0,1\}^\lambda$ be a cryptography hash function, $a_i \in R_q$, $H(a_i) = \delta_i$. For a given $\delta \in \{0,1\}^\lambda$, an adversary $\mathcal{A}$ sends a query $x \in \{0,1\}^\star$ to H, which happens to have probability $\Pr[H(x) = \delta] = \frac{1}{2^\lambda}$. Let Adv denotes the probability that $\mathcal{A}$ finds a collision after making $q_{\mathsf{ro}} = \mathsf{poly}(\lambda)$ queries, Obviously $\mathsf{Adv} = \mathsf{negl}(\lambda)$, we have the following result.

Lemma 12. *For a given $\delta \in \{0,1\}^\lambda$, k probabilistic polynomial time(ppt) adversary $\mathcal{A}$, Each $\mathcal{A}$ makes $q_{ro} = \mathsf{poly}(\lambda)$ queries to H, let $\overline{\mathsf{Adv}}$ denotes the probability of finding a collision, then: $\overline{\mathsf{Adv}} = \mathsf{negl}(\lambda)$*

For *scheme#2*, we only describe its key generation and re-linearization procedure in detail, the encryption, evaluation and decryption algorithm is similar to other RLWE-based MKFHE schemes.

Key generation with bit commitment.

k participants perform the following steps to get their own public key and evaluation key

1. $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$:Input security parameter λ, circuit depth L, output $\mathsf{pp} = (d, q, \chi, B_\chi)$, which χ is an noise distribution over $R : \mathbb{Z}[x]/x^d + 1$, satisfying $e \leftarrow \chi$, $||e||_\infty^{can}$ is bounded by B_χ, and $\mathsf{RLWE}_{d,q,\chi,B_\chi}$ is infeasible.
2. i generates $\Phi_i = \{a_i, \mathbf{d}_i, \mathbf{f}_i\}$ where $a_i \leftarrow U(R_q)$ is used for public key, $\mathbf{d}_i$, $\mathbf{f}_i \leftarrow U(R_q^l)$ for evaluation key, and commitment $\Psi_i = \{\delta_i, \epsilon_i, \zeta_i\}$, $\delta_i = H(a_i)$, $\epsilon_i = H(\mathbf{d}_i)$, $\zeta_i = H(\mathbf{f}_i)$, broadcast Ψ_i.
3. After all $\{\Psi_i\}_{i\in[k]}$ are public, i discloses Φ_i.
4. After receiving $\{\Phi_j\}_{j\in[k]/i}$, i broadcast $\{b_i,\ \mathbf{h}_i\}$, where $b_i = as_i + e_1$, $\mathbf{h}_i = \mathbf{d}s_i + \mathbf{e}_2$, $a = \sum_{i=1}^k a_i$, $\mathbf{d} = \sum_{i=1}^k \mathbf{d}_i$, $(s_i,\ e_1,\ \mathbf{e}_2) \leftarrow \chi^{l+2}$.
5. After receiving $\{b_j,\ \mathbf{h}_j\}_{j\in[k]/i}$, i output $\mathsf{pk}_i = (a, b)$ and evaluation key $\mathsf{evk}_i = (\mathbf{h}_i, \boldsymbol{\eta}_i, \boldsymbol{\theta}_i)$

$$b = \sum_{i=1}^{k} b_i \qquad \boldsymbol{\eta}_i = \mathbf{d}r_i + \mathbf{e}_3 + s_i\mathbf{g}$$
$$\boldsymbol{\theta}_i = \mathbf{f}s_i + \mathbf{e}_4 + r_i\mathbf{g} \qquad (r_i,\ \mathbf{e}_3,\ \mathbf{e}_4) \leftarrow \chi^{2l+1}$$

Re-linearization ciphertext

Multiplying two ciphertext $\mathbf{c}_1,\ \mathbf{c}_2 \in R_q^2$, which under the same private key $\mathbf{t} = (1, s)$, $s = \sum_{i=1}^k s_i$, resulting $\mathbf{c}_{\mathsf{mult}} = \mathbf{c}_1 \otimes \mathbf{c}_2 \in R_q^4$, where its corresponding private key is $\mathbf{t} \otimes \mathbf{t} = (1, s, s, s^2)$. In order to re-linearize $\mathbf{c}_{\mathsf{mult}}$, we need to construct the ciphertext of s^2 under $\mathbf{t}$. Let total evaluation key $\boldsymbol{\Pi} = (\boldsymbol{\eta}, \boldsymbol{\theta}, \mathbf{h})$.

$$\text{where } \boldsymbol{\eta} = \sum_{i=1}^{k} \boldsymbol{\eta}_i \quad \boldsymbol{\theta} = \sum_{i=1}^{k} \boldsymbol{\theta}_i \quad \mathbf{h} = \sum_{i=1}^{k} \mathbf{h}_i$$

Let $\mathbf{k} = (\mathbf{k}_0, \mathbf{k}_1),\ \mathbf{k}_0 = -\boldsymbol{\theta}\mathbf{g}^{-1}(\mathbf{h}) \in R_q^l,\ \mathbf{k}_1 = (\boldsymbol{\eta} + \mathbf{f}\mathbf{g}^{-1}(\mathbf{h})) \in R_q^l$, obviously $\mathbf{k}_0 + \mathbf{k}_1 s \approx s^2\mathbf{g}$ (omit small error). Let $\mathbf{c}_{\mathsf{mult}} = (c_0, c_1, c_2, c_3)$.

$$\begin{aligned}\langle \mathbf{c}_{mult}, \mathbf{t} \otimes \mathbf{t} \rangle &= c_0 + (c_1 + c_2)s + s^2 c_3 \\ &= c_0 + (c_1 + c_2)s + s^2\mathbf{g}\mathbf{g}^{-1}(c_3) \\ &= c_0 + \mathbf{k}_0\mathbf{g}^{-1}(c_3) + (c_1 + c_2 + \mathbf{k}_1\mathbf{g}^{-1}(c_3))s.\end{aligned}$$

Let $\mathbf{c}_{\mathsf{linear}} = (c_0', c_1'),\ c_0' = c_0 + \mathbf{k}_0\mathbf{g}^{-1}(c_3),\ c_1' = c_1 + c_2 + \mathbf{k}_1\mathbf{g}^{-1}(c_3)$, output $\mathbf{c}_{\mathsf{linear}}$ as re-linearized ciphertext. The algorithm defines as follows:

– $\mathbf{c}_{\mathsf{linear}} \leftarrow \mathsf{Relinear}(\mathbf{c}_{\mathsf{mult}}, \{\mathsf{evk}_i\}_{i\in[k]})$: Input $\mathbf{c}_{\mathsf{mult}} \in R_q^4$, evaluation key $\{\mathsf{evk}_i\}_{i\in[k]}$, perform the following algorithem, output $\mathbf{c}_{\mathsf{linear}} = (c_0', c_1')$.

Due to the sum structure of keys, the dimension of $\mathbf{t} \otimes \mathbf{t}$ is independent of participants k, thus above algorithm pulls the tensor product ciphertext back to initial dimension by one shot, and introduces less noise than those keys with concatenation structure.

Ciphertext Relinearization

Input: $\mathbf{c}_{\mathsf{mult}} = (c_0, c_1, c_2, c_3) \in R_q^4,\ \{\mathsf{evk}_i\}_{i\in[k]} = \{\mathbf{h}_i,\ \boldsymbol{\eta}_i,\ \boldsymbol{\theta}_i\}_{i\in[k]}$

Output: $\mathbf{c}_{\mathsf{linear}} = (c_0', c_1') \in R_q^2$

1: $\boldsymbol{\eta} \leftarrow \sum_{i=1}^{k} \boldsymbol{\eta}_i,\ \boldsymbol{\theta} \leftarrow \sum_{i=1}^{k} \boldsymbol{\theta}_i,\ \mathbf{h} \leftarrow \sum_{i=1}^{k} \mathbf{h}_i$

2: $\mathbf{k}_0 \leftarrow -\boldsymbol{\theta}\mathbf{g}^{-1}(\mathbf{h}),\ \mathbf{k}_1 \leftarrow \boldsymbol{\eta} + \mathbf{f}\mathbf{g}^{-1}(\mathbf{h})$

3: $c_0' \leftarrow c_0 + \mathbf{k}_0\mathbf{g}^{-1}(c_3),\ c_1' \leftarrow c_1 + c_2 + \mathbf{k}_1\mathbf{g}^{-1}(c_3)$

4: **Output:** (c_0', c_1')

5: **End**.

6 Conclusions

For the LWE-based MKFHE in order to alleviate the overhead of the local participants, we proposed the concept of KL-MKFHE which introduced a **Key lifting** procedure, getting rid of expensive ciphertext expansion operation and construct a DGSW style KL-MKFHE under plain model. Our *scheme#1* is more friendly to local participants than previous scheme, for which the local encryption $O(N\lambda^6 L^4)$ is reduced to $O(N)$, and by abandoning noise flooding, it compress q from $2^{\lambda L}B_\chi$ to $2^{\lambda+L}B_\chi$, reducing the computational scale of the entire scheme. However, the key length depends on the number of participants and the amount of leakage, which limits the application of the scheme to some extent. Further work will focus on compressing the key length.

For the multi-key homomorphic scheme based on RLWE, although the computation overhead of the local participants is not large: to perform re-linearization, only one ring element needs to be encrypted, the common random string is always an insurmountable hurdle. We introduced bit commitment to ensure the independence of the $\{a_i\}_{i\in[k]}$ generated by each participant under ROM. Constructing RLWE-type MKFHE under plain model is the future direction.

References

1. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: 28th ACM STOC. pp. 99–108. ACM Press (May 1996)
2. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology 9(3), 169–203 (2015)
3. Alperin-Sheriff, J., Peikert, C.: Faster bootstrapping with polynomial error. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 297–314. Springer, Heidelberg (Aug 2014)
4. Ananth, P., Jain, A., Jin, Z., Malavolta, G.: Multi-key fully-homomorphic encryption in the plain model. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 28–57. Springer, Heidelberg (Nov 2020)
5. Ananth, P., Jain, A., Jin, Z., Malavolta, G.: Unbounded multi-party computation from learning with errors. pp. 754–781. LNCS, Springer, Heidelberg (2021)
6. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction

via threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (Apr 2012)
7. Beaver, D.: Efficient multiparty protocols using circuit randomization. In: Feigenbaum, J. (ed.) CRYPTO'91. LNCS, vol. 576, pp. 420–432. Springer, Heidelberg (Aug 1992)
8. Boneh, D., Gennaro, R., Goldfeder, S., Jain, A., Kim, S., Rasmussen, P.M.R., Sahai, A.: Threshold cryptosystems from threshold fully homomorphic encryption. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018. pp. 565–596. Springer International Publishing, Cham (2018)
9. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Goldwasser, S. (ed.) ITCS 2012. pp. 309–325. ACM (Jan 2012)
10. Brakerski, Z., Halevi, S., Polychroniadou, A.: Four round secure computation without setup. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 645–677. Springer, Heidelberg (Nov 2017)
11. Brakerski, Z., Perlman, R.: Lattice-based fully dynamic multi-key FHE with short ciphertexts. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 190–213. Springer, Heidelberg (Aug 2016)
12. Chen, H., Dai, W., Kim, M., Song, Y.: Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 395–412. ACM Press (Nov 2019)
13. Cheon, J.H., Kim, A., Kim, M., Song, Y.S.: Homomorphic encryption for arithmetic of approximate numbers. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 409–437. Springer, Heidelberg (Dec 2017)
14. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part I. LNCS, vol. 10031, pp. 3–33. Springer, Heidelberg (Dec 2016)
15. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (Aug 2015)
16. Dachman-Soled, D., Gong, H., Kulkarni, M., Shahverdi, A.: Towards a ring analogue of the leftover hash lemma. Journal of Mathematical Cryptology 15(1), 87–110 (2021)
17. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 643–662. Springer, Heidelberg (Aug 2012)
18. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) Advances in Cryptology – EUROCRYPT 2010. pp. 24–43. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
19. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144 (2012), https://eprint.iacr.org/2012/144
20. Gentry, C.: A fully homomorphic encryption scheme. Stanford university (2009)
21. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st ACM STOC. pp. 169–178. ACM Press (May / Jun 2009)
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008)

23. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (Aug 2013)
24. Impagliazzo, R., Levin, L.A., Luby, M.: Pseudo-random generation from one-way functions (extended abstracts). In: 21st ACM STOC. pp. 12–24. ACM Press (May 1989)
25. Jain, A., Rasmussen, P.M.R., Sahai, A.: Threshold fully homomorphic encryption. Cryptology ePrint Archive, Paper 2017/257 (2017), https://eprint.iacr.org/2017/257, https://eprint.iacr.org/2017/257
26. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische annalen 261(ARTICLE), 515–534 (1982)
27. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Karloff, H.J., Pitassi, T. (eds.) 44th ACM STOC. pp. 1219–1234. ACM Press (May 2012)
28. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (May / Jun 2010)
29. Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 35–54. Springer, Heidelberg (May 2013)
30. Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-lwe cryptography. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 35–54. Springer (2013)
31. Micciancio, D.: Almost perfect lattices, the covering radius problem, and applications to ajtai's connection factor. SIAM Journal on Computing 34(1), 118–169 (2004)
32. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (Apr 2012)
33. Micciancio, D., Peikert, C.: Hardness of sis and lwe with small parameters. In: Annual Cryptology Conference. pp. 21–39. Springer (2013)
34. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. In: 45th FOCS. pp. 372–381. IEEE Computer Society Press (Oct 2004)
35. Mouchet, C., Troncoso-Pastoriza, J., Hubaux, J.P.: Computing across trust boundaries using distributed homomorphic cryptography. Cryptology ePrint Archive, Paper 2019/961 (2019), https://eprint.iacr.org/2019/961, https://eprint.iacr.org/2019/961
36. Mouchet, C., Troncoso-Pastoriza, J.R., Bossuat, J.P., Hubaux, J.P.: Multiparty homomorphic encryption from ring-learning-with-errors. PoPETs 2021(4), 291–311 (Oct 2021)
37. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 735–763. Springer, Heidelberg (May 2016)
38. Myers, S., Sergi, M., abhi shelat: Threshold fully homomorphic encryption and secure computation. Cryptology ePrint Archive, Paper 2011/454 (2011), https://eprint.iacr.org/2011/454, https://eprint.iacr.org/2011/454
39. Peikert, C., Shiehian, S.: Multi-key fhe from lwe, revisited. In: Theory of Cryptography Conference. pp. 217–238. Springer (2016)

40. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 84–93. ACM Press (May 2005)
41. Rivest, R.L., Adleman, L., Dertouzos, M.L., et al.: On data banks and privacy homomorphisms. Foundations of secure computation 4(11), 169–180 (1978)
42. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
43. Stehlé, D., Steinfeld, R.: Making ntru as secure as worst-case problems over ideal lattices. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 27–47. Springer (2011)
44. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (May / Jun 2010)