Batched Multi-hop Multi-key FHE from ring-LWE with Compact Ciphertext Extension

Long Chen[1,2], Zhenfeng Zhang[1,2](✉), and Xueqing Wang[1,3]

[1] University of Chinese Academy of Sciences
[2] Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences
{chenlong,zfzhang}@tca.iscas.ac.cn
[3] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences
wangxueqing@iie.ac.cn

Abstract. Traditional fully homomorphic encryption (FHE) schemes support computation on data encrypted under a single key. In STOC 2012, López-Alt et al. introduced the notion of multi-key FHE (MKFHE), which allows homomorphic computation on ciphertexts encrypted under different keys. In this work, we focus on MKFHE constructions from standard assumptions and propose a new construction of ring-LWE-based multi-hop MKFHE scheme. Our work is based on Brakerski-Gentry-Vaikuntanathan (BGV) FHE scheme where, in contrast, all the previous works on multi-key FHE with standard assumptions were based on Gentry-Sahai-Waters (GSW) FHE scheme. Therefore, our construction can encrypt a ring element rather than a single bit and naturally inherits the advantages in aspects of the ciphertext/plaintext ratio and the complexity of homomorphic operations. Moveover, our MKFHE scheme supports the Chinese Remainder Theorem (CRT)-based ciphertexts packing technique, achieves $poly(k, L, \log n)$ computation overhead for k users, circuits with depth at most L and an n dimensional lattice, and gives the first batched MKFHE scheme based on standard assumptions to our knowledge. Furthermore, the ciphertext extension algorithms of previous schemes need to perform complex computation on each ciphertext, while our extension algorithm just needs to generate evaluation keys for the extended scheme. So the complexity of ciphertext extension is only dependent on the number of associated parities but not on the number of ciphertexts. Besides, our scheme also admits a threshold decryption protocol from which a generalized two-round MPC protocol can be similarly obtained as prior works.

1 Introduction

Fully homomorphic encryption (FHE) is a very attractive cryptography primitive that allows computation on encrypted data and has numerous theoretical and practical applications [Gen09,BV11b,DPSZ12,GSW13]. In STOC 2012, López-Alt et al. introduced a notion of multi-key FHE (MKFHE) [LATV12], which is a variant of FHE allowing computation on data encrypted under different and independent keys. One of the most appealing applications of MKFHE is to construct on-the-fly multiparty computation (MPC) protocols.

López-Alt et al. [LATV12] proposed the first MKFHE construction based on the NTRU cryptosystem [HPS98], which was optimized later in [DHS16]. However, the security of this construction is based on a new and somewhat non-standard assumption on polynomial rings. Clear and McGoldrick [CM15] proposed an LWE-based MKFHE construction for an unlimited number of keys using the Gentry-Sahai-Waters (GSW) FHE scheme [GSW13,ASP14]. In EUROCRYPT 2016, Mukherjee and Wichs [MW16] presented a construction of MKFHE based on LWE that simplifies the scheme of Clear and McGoldrick [CM15] and admits a simple 1-round threshold decryption protocol. Based on this threshold MKFHE, they successfully constructed a general two-round MPC protocol upon it in the common random string model.

The schemes in [CM15,MW16] need to determine all the involved parties before the homomorphic computation and do not allow any new party to join in, which called *single-hop* MKFHE in [PS16]. Recently, Peikert and Shiehian [PS16] proposed a notion of *multi-hop* MKFHE, in which the result ciphertexts of homomorphic evaluations can be used in further homomorphic computations involving additional parties (secret keys). In multi-hop MKFHE, any party can dynamically join the homomorphic computation at any time. A similar notion named *fully dynamic* MKFHE was proposed by Brakerski and Perlman in [BP16]. A slight difference is that in fully dynamic MKFHE the bound of the number of users does not need to be input during the setup procedure.

The method to construct multi-hop MKFHE in [PS16] is maintaining commitment randomness relative to a fixed public parameter, along with an encryption of that randomness. Their homomorphic evaluation algorithm requires only a few standard GSW-style matrix operations. This comes at the cost of relatively larger ciphertexts, which grow at least quadratically in the maximum number of keys. In [BP16], Brakerski and Perlman provided a fully dynamic MKFHE scheme with an approach of extending the refresh keys to the ones under a joint secret key at first and then bootstrapping the ciphertexts by the extended refresh keys. Specifically, their multi-key ciphertexts grow only linearly in the number of different involving secret keys. In addition, they described an "on-the-fly" bootstrapping algorithm that requires only a linear amount of "local" memory. However, as [PS16] analyzed, [BP16] is comparatively inefficient since their bootstrapping is generally very costly and some efficient bootstrapping techniques such as [ASP13,HS15,DM15] seem not to be applicable here.[4] From above, one can obverse that MKFHE is still far from practical, even comparing with existing results of single key FHE.

[4] Most of practical bootstrapping techniques [ASP13,HS15,DM15] are based on ring-LWE schemes and few can be applied to the LWE-based GSW scheme like that used in [BP16].

1.1 Motivations

Encrypting a ring element. There are two most widely studied single-key FHE schemes based on standard assumptions, the BGV type scheme [BV11a,BGV12] [GHS12b,HS15] and the GSW type scheme [GSW13,BV14,ASP14]. Both of them have an LWE version and a ring-LWE version. As the analysis in [GSW13], the most efficient one among them is the ring-LWE based BGV scheme in aspects of the ciphertext/plaintext ratio and the complexity of homomorphic operations. Actually, the plaintext of ring-LWE BGV scheme is a ring element, while both the LWE version and ring-LWE version of GSW scheme can only encrypt one bit for each ciphertext according to [GSW13]. The major reason is that the GSW noise depends also on the plaintext size after a homomorphic multiplication. Consequently, MKFHE from the GSW scheme [CM15,MW16,BP16,PS16] can only encrypt a single bit even based on the ring-LWE assumption. Therefore, if we can encrypt a ring element in MKFHE schemes, the efficiency will be improved considerably.

SIMD operations. Currently, the most efficient FHE schemes are those that allow SIMD (Single Instruction Multiple Data) style operations, by packing some plaintexts into the same number of independent "slots" as the plaintext space. Gentry et al. [GHS12b] showed that if the circuit $\mathcal{C}$ has size $t = poly(\lambda)$, depth L, and average width $w = O(\lambda)$, and we set the packing parameter as $l = \Theta(\lambda)$, then we get an $O(L \cdot \log \lambda)$-depth implementation of $\mathcal{C}$ using $O(t/\lambda \cdot poly \log(\lambda))$ l-fold gates. If implementing each l-fold gates takes $\tilde{O}(L^b \lambda^c)$ time, then the total time to evaluate $\mathcal{C}$ is no more than $\tilde{O}(t \cdot L^b \cdot \lambda^{c-1})$. Smart and Vercauteren described a ciphertext-packing technique based on polynomial-CRT [SV14], and Gentry et al. [GHS12b] used the technique to achieve a nearly optimal homomorphic evaluation (up to poly-logarithmic factors). Besides, two other ciphertexts packing techniques have been proposed [BGH13,HAO15] so far, both of which are based on kinds of matrix operations rather than the algebra structure of the rings. However, it is not clear how ciphertext packing techniques can be applied to standard assumption based MKFHE schemes [CM15,MW16,BP16,PS16] so far.

Generally, since existing MKFHE schemes [CM15,MW16,BP16,PS16] from standard assumptions are all based on GSW scheme, one interesting theoretical problem is that *whether we can construct MKFHE from other existing standard assumption based single key FHE schemes?*

Compact ciphertext extension. In the MKFHE schemes [CM15,MW16,BP16,PS16], each party's messages are encrypted by different public keys at first, and the original ciphertexts correspond to different secret keys. When several parties decide to jointly evaluate a circuit, a *ciphertext extension* algorithm is used to transform the original ciphertexts to larger dimensional ciphertexts under a same new secret key which is a concatenation of all the involved parities' secret keys. Generally, the outputs of ciphertext extension can be viewed as the ciphertexts in a single-key FHE scheme with a larger dimensional secret key. After that, the circuit is finally evaluated under the new larger dimensional single-key FHE scheme.

Particularly, in [CM15,MW16], a GSW ciphertext is extended to a k times dimensional ciphertext matrix, by adding sub-blocks which are derived from the encryption of randomness. The ciphertext extension in [PS16] is similar to that in [CM15,MW16], while the additional sub-blocks are derived from commitment randomness relative to a fixed public parameter, along with an encryption of that randomness. In [BP16], the ciphertext extension of a GSW ciphertext is completed by bootstrapping the ciphertext with an extended refreshed key which needs to be generated in advance. All of their ciphertext extension algorithms need to perform complex computations for each ciphertext, which will be a heavy burden if the number of ciphertexts is large. We observe that such a ciphertext extension procedure is not needed in [LATV12]. For a standard assumption based MKFHE scheme, a natural question is whether one can directly compute homomorphic operations for the ciphertexts under different keys and reduce the dependence of the computation cost of ciphertext extension (if necessary) on the number of ciphertexts.

1.2 Our Contributions

Note that all previous MKFHE [CM15,MW16,BP16,PS16] are all constructed from the GSW scheme. In this paper, we construct a new ring-LWE based multi-hop MKFHE scheme from the BGV scheme, so our work naturally inherits the advantages of the second generation FHE [Lin]. For example, our scheme can encrypt a ring element and support the CRT-based ciphertexts packed technique. So it is much more efficient than prior works in aspects of the ciphertext/plaintext ratio, the complexity of homomorphic operations and other computation overhead. The detailed comparisons are provided in Table 1, Table 2 and Table 3 in Subsection 4.7. Similar to [PS16], a priori bound on the number of users is required at the setup phase. Our scheme also admit a threshold decryption protocol as [MW16], so a 2-round MPC can be similarly obtained from our construction.

A simple ciphertext extension is also used in our construction to transform BGV ciphertexts under different secret keys to larger dimensional ciphertexts under the concatenation of all involving secret keys, which is realized by just padding the ciphertext vectors with zeros. However, due to the structure of the BGV cryptosystem, the generation of new evaluation keys is needed. As the result, the complexity of the extension procedure is dependent only on the number of involved secret keys *but not on the number of ciphertexts*. The evaluation keys are generated in the key-generation phase, and can be pre-computed before encryption and even be publicly stored for the next time evaluation if the involved parties are unchanged. This is beneficial for a possible scenario where multiple ciphertexts are encrypted with the same key.

Generally, both the LWE version and the ring-LWE version of our construction can be provided. In the text, we choose to present the ring-LWE version. It is easy for readers to get the analogous LWE version without much effort. Moreover, our technique of constructing MKFHE can be extended to other (ring-)LWE based second generation FHE schemes such as [BV11a,Bra12], and all optimization techniques about these FHE schemes [GHS12a,GHS12c,GHPS13,ASP13] [HS14,HS15,CP16] also can apply here.

From a technical point of view, we show the evaluation key of BGV scheme can be generated from a GSW encryption of a secret key the first time. We believe this technique can help us to better understand the internal connection between these two famous FHE schemes.

1.3 Technique Overview

In the ring-LWE based BGV scheme, given level-l ciphertexts $\mathbf{c}_l = (\langle \mathbf{a}, \mathbf{z}_l \rangle + 2e + \mu, \mathbf{a}) \in R_q^2$ under the secret key $\mathbf{s}_l = (1, -\mathbf{z}_l) \in R_q^2$ and $\mathbf{c}'_l = (\langle \mathbf{a}', \mathbf{z}'_l \rangle + 2e' + \mu', \mathbf{a}') \in R_q^2$ under the secret key $\mathbf{s}'_l = (1, -\mathbf{z}'_l) \in R_q^2$, one can trivially extend them to ciphertexts $\bar{\mathbf{c}}_l = (\mathbf{c}_l, \mathbf{0}) \in R_q^4$ and $\bar{\mathbf{c}}'_l = (\mathbf{0}, \mathbf{c}'_l) \in R_q^4$ under the same secret key $\bar{\mathbf{s}}_l = (\mathbf{s}_l, \mathbf{s}'_l) \in R_q^4$ which is a concatenation of the two parties' secret keys. For extended ciphertexts, the homomorphic addition is just the vector addition. But for homomorphic multiplication, one need to compute the tensor product of the two ciphertexts, then use the evaluation key to relinearization the ciphertext. Since the corresponding secret key of $\bar{\mathbf{c}}_l \otimes \bar{\mathbf{c}}'_l \in R_q^{16}$ is $\hat{\mathbf{s}}_l = \bar{\mathbf{s}}_l \otimes \bar{\mathbf{s}}_l \in R_q^{16}$, the required evaluation key is

$$evk_l = \left\{ \left(\left\langle \mathbf{a}_{i,j}, \mathbf{z}^*_{l-1} \right\rangle + 2e_{i,j} + 2^j \hat{\mathbf{s}}_l[i], \mathbf{a}_{i,j} \right)_{i=1,\ldots,16,\ j=0,\ldots,\lfloor \log q \rfloor} \right\} \tag{1}$$

for next level secret key $\bar{\mathbf{s}}_{l-1} = (1, -\mathbf{z}^*_{l-1}) \in R_q^4$ and some $\mathbf{a}_{i,j} \in R_q^3$. So the main obstacle is to generate the evaluation key evk_l.

Generating BGV's evk from GSW scheme. Intuitively, evk_l can be viewed as a kind of "encryption" of each element of $\hat{\mathbf{s}}_l \in R_q^{16}$. Our first observation is that evk_l of the BGV scheme can be generated from a GSW encryption of $\hat{\mathbf{s}}_l$. In fact, the variant of GSW encryption for the plaintext $\hat{\mathbf{s}}_l[i]$ is

$$\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\hat{\mathbf{s}}_l[i]) = r \left(\mathbf{A}\mathbf{z}^*_{l-1} + 2\mathbf{e}, \mathbf{A} \right) + 2\mathbf{E} + \hat{\mathbf{s}}_l[i] \cdot \mathbf{G} \in R_q^{(\lfloor \log q \rfloor + 1)4 \times 4}.$$

Here

$$\mathbf{G} = \left(1, \ldots, 2^{\lfloor \log q \rfloor}\right)^T \otimes \mathbf{I}_4 \in \mathbb{Z}_q^{(\lfloor \log q \rfloor + 1)4 \times 4}$$

is the gadget matrix, $\mathbf{A} \in R_q^{4(\lfloor \log q \rfloor + 1) \times 3}$ is a random matrix, $r \in R_q$ and $\mathbf{E} \in R^{4(\lfloor \log q \rfloor + 1) \times 4}$. Note that the plaintext is encrypted in low bits, which is different from the original GSW scheme in [GSW13,ASP14]. Then the j-th row has the form $\left(\left\langle \mathbf{a}_j, \mathbf{z}^*_{l-1} \right\rangle + 2e_j + 2^j \hat{\mathbf{s}}_l[j], \mathbf{a}_j \right) \in R_q^4$ for some random vector $\mathbf{a}_j \in R_q^3$. This gives the evaluation key evk_l we need.

The next task is to generate $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\hat{\mathbf{s}}_l[i])$. Our basic idea is to take advantage of the ciphertext extension method in [CM15,MW16]. Specifically, each element of $\hat{\mathbf{s}}_l$ is a product of two elements of $\bar{\mathbf{s}}_l$, where $\bar{\mathbf{s}}_l$ is the concatenation of each party's secret key. So if one party's public key includes

$\mathsf{GSW.Enc}_{\mathbf{s}_{l-1}}(\mathbf{s}_l[i]), i = 1, 2$, it can be extended to a larger dimensional ciphertext $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\mathbf{s}_l[i])$ under the secret key $\bar{\mathbf{s}}_{l-1} = (\mathbf{s}_{l-1}, \mathbf{s}'_{l-1}) \in R_q^4$, and also $\mathsf{GSW.Enc}_{\mathbf{s}'_{l-1}}(\mathbf{s}'_l[i])$ can be extended to $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\mathbf{s}'_l[i])$. If we can homomorphically multiply $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\mathbf{s}_l[i])$ and $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\mathbf{s}'_l[i']), i, i' = 1, 2$, we get all the element of $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\hat{\mathbf{s}}_l[i]), i = 1, \ldots, 16$. Then we can derive $\lfloor \log q \rfloor + 1$ BGV ciphertexts

$$\left(\langle \mathbf{a}_j, \mathbf{z}^*_{l-1} \rangle + 2e_j + 2^j \hat{\mathbf{s}}_l[j], \mathbf{a}_j\right) \in R_q^4, j = 0, \ldots, \lfloor \log q \rfloor$$

under the secret key $\bar{\mathbf{s}}_{l-1} = (\mathbf{s}_{l-1}, \mathbf{s}'_{l-1}) = (1, -\mathbf{z}^*_{l-1})$ from each GSW encryption $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\hat{\mathbf{s}}_l[i])$, and therefore get the supposed evaluation key.

GSW Scheme with ring element plaintext. However, the plaintext of the traditional GSW scheme is in $\{0, 1\}$ while we encrypt $\hat{\mathbf{s}}_l[i] \in R_q$. When the plaintext is an element in R_q, the homomorphic multiplication can not work normally as explained before since the noise will be out of control. To deal with this problem, we propose a variant of GSW scheme with ring element plaintext. Specifically, we observe that when we compute $\mathsf{GSW.Enc}(a) \odot \mathsf{GSW.Enc}(b)$ for some $a, b \in R_q$, the noise in the result ciphertext only depends on *b but not on a*. So we can compute

$$\sum_{i=0}^{\lfloor \log q \rfloor} \mathsf{GSW.Enc}\left(\mathsf{Powersof2}(a)[i]\right) \odot \mathsf{GSW.Enc}\left(\mathsf{BitDecomp}(b)[i]\right).$$

Such a homomorphic multiplication in our GSW scheme with ring element plaintext can *only be performed once*, but it is enough for us to successfully compute $\mathsf{GSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\hat{\mathbf{s}}_l[i])$.

1.4 Organization

In Section 2, some background knowledge is provided. We introduce a special GSW scheme with ring element plaintext which is used to generate evaluation keys and existing techniques about the BGV scheme in Section 3. In Section 4, we give a formal description of our ring-LWE based MKFHE construction. Finally, in Section 5, we present a threshold decryption mechanism and a two round MPC protocol from our scheme. The conclusion is provided in Section 6.

2 Preliminaries

In this paper, we use bold lower case letters to denote vectors and bold upper case letters to denote matrices. All vectors are represented as columns. For a matrix $\mathbf{A}$, we use $\mathbf{A}[i, :]$ to denote the i-th row vector, and $\mathbf{A}[i, j]$ to denote the entry in the i-th row and j-th column. For a vector $\mathbf{a}$, $\mathbf{a}[i]$ denotes the i-th entry.

For a positive integer m, let $\Phi_m(X)$ be the m-th cyclotomic polynomial which has degree $n = \phi(m)$ where $\phi(\cdot)$ is the Euler's function. We will use the ring $R = \mathbb{Z}[X]/\Phi_m(X)$ and its localization R_N, for some modulus N. When dealing with R_N, we assume that the coefficients are in $[-N/2, N/2)$ (except for R_2 whose coefficients are in $\{0,1\}$). Given a polynomial $a \in R$, we denote by $\|a\|_\infty = \max_{0 \le j \le n-1} |a_j|$ the standard l_∞-norm and $\|a\|_1 = \sum_{j=0}^{n-1} |a_j|$ the standard l_1-norm.

2.1 Hardness Assumption

The ring-LWE problem introduced by [LPR13a] can be seen as a ring version of the LWE problem [Reg09]. Now we recall its definition. Let K be the m-th cyclotomic number field having dimension $n = \phi(m)$ and $R = \mathcal{O}_K$ be its ring of integers which embeds as a lattice. $R^\vee \subset K$ is the dual fractional ideal of R. The noise estimation can be taken with respect to the canonical embedding norm $\|a\|_\infty^{can} = \|\sigma(a)\|_\infty$, where σ is the canonical embedding defined in [LPR13a]. To map from norms in the canonical embedding to norms on the coefficients of the polynomial, we have

$$\|a\|_\infty \le c_m \|a\|_\infty^{\mathrm{can}}, \tag{2}$$

where c_m is the ring expansion factor, see [DPSZ12] for more details.

Definition 1 (Ring-LWE[LPR13a,LPR13b]). *For an $s \in R_q^\vee$ and a distribution χ over the field tensor product $K_\mathbb{R} = K \otimes_\mathbb{Q} \mathbb{R}$, a sample from the ring-LWE distribution $A_{s,\chi}$ over $R_q \times K_\mathbb{R}/qR^\vee$ is generated by choosing $a \leftarrow R_q$ uniformly at random, choosing $e \leftarrow \chi$, and outputting $(a, b = a \cdot s + e)$.*

The decisional version of the ring-LWE problem, denoted R-DLWE$_{q,\chi}$, is to distinguish with non-negligible advantage between independent samples from $A_{s,\chi}$, where s is uniformly chosen from $R_q^\vee$ once and for all, and the same number of uniformly random and independent samples from $R_q \times K_\mathbb{R}/qR^\vee$.

On the hardness, the theorem below captures reductions from GapSVP (GapSIVP) on ideal lattices to ring-LWE for certain parameters. We state the result in terms of canonical norm B-bounded distributions over the ring. Hereafter, "canonical norm" sometimes will be omit.

Definition 2 (B-bounded distribution over the ring). *A distribution ensemble $\{\chi_n\}_{n\in\mathbb{N}}$, supported over $K_\mathbb{R}$, is called (canonical norm) B-bounded if*

$$\Pr_{e \leftarrow \chi_n}\left[\|e\|_\infty^{can} > B\right] = negl\,(n)\,.$$

Theorem 1 ([LPR13a,LPR13b]). *Let R be the m-th cyclotomic ring, having dimension $n = \phi(m)$. Let $q = q(n)$, $q = 1 \bmod m$ be a poly(n)-bounded integer, and $B = \omega(\sqrt{n \log n})$. There is a poly$(n)$-time quantum reduction from $n^{\omega(1)} q/B$-approximate SIVP (or SVP) on ideal lattices in R to solve R-DLWE$_{q,\chi}$ where χ is a distribution bounded by B with overwhelming probability.*

It has been shown for ring-LWE that one can equivalently assume that s is alternatively sampled from the noise distribution χ [LPR13a].

2.2 Smudging Lemma

We rely on the following lemma, which says that adding large noise "smudges out" any small values.

Lemma 1 ([AJL$^+$12]). *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 the integer $e_2 \in [-B_2, B_2]$ be chosen uniformly at random. Then the distribution of e_2 is statistically indistinguishable from that of $e_2 + e_1$ as long as $B_1/B_2 = negl(\lambda)$.*

Similarly, when $R = \mathbb{Z}[X]/\varPhi_m(X)$, let $e_1 \in R_q$ be a fixed ring element where $\|e_1\|_\infty \leq B_1$, and e_2 be another ring element whose coefficients are chosen uniformly at random from $[-B_2, B_2]$. Then the distribution of e_2 is statistically indistinguishable from that of $e_2 + e_1$ as long as $B_1/B_2 = negl(\lambda)$.

2.3 Bit Decomposition Technique

The bit decomposition technique is first introduced in [BV11a] and widely used in FHE schemes. Let $\beta = \lfloor \log q \rfloor + 1$. We describe the subroutines as follows.

- $\mathsf{BitDecomp}(\mathbf{V} \in \mathbb{Z}_q^{n\times d})$: Decompose each coefficient of $\mathbf{V}$ in bit representation. Namely, write $\mathbf{V} = \sum_{j=0}^{\lfloor \log q \rfloor} 2^j \cdot \mathbf{U}_j$, with all $\mathbf{U}_j \in \{0,1\}^{n\times d}$, and output $\left[\mathbf{U}_0, \mathbf{U}_1, \ldots, \mathbf{U}_{\lfloor \log q \rfloor}\right] \in \{0,1\}^{n\times d\beta}$.
- $\mathsf{Powersof2}(\mathbf{V} \in \mathbb{Z}_q^{n\times d})$: Let $\mathbf{W}_j = 2^j \mathbf{V} \mod q \in \mathbb{Z}_q^{n\times d}$, $j = 0, \ldots, \lfloor \log q \rfloor$ and output $\left[\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{\lfloor \log q \rfloor}\right] \in \mathbb{Z}_q^{n\times d\beta}$.

Obviously, $\mathsf{BitDecomp}(\mathbf{U}) \cdot \mathsf{Powerof2}(\mathbf{V})^T = \mathbf{U} \cdot \mathbf{V}^T$, where $\mathbf{U}, \mathbf{V} \in \mathbb{Z}_q^{n\times d}$. Consequently, let $\mathbf{g} = \left(1, 2, \ldots, 2^{\lfloor \log q \rfloor}\right)^T \in \mathbb{Z}_q^\beta$, $\mathbf{I}_d$ be the d dimensional identity matrix and $\mathbf{G} = \mathbf{g} \otimes \mathbf{I}_d \in \mathbb{Z}_q^{d\beta\times d}$. For any matrix $\mathbf{C} \in \mathbb{Z}_q^{n\times d}$, $\bar{\mathbf{C}} = \mathsf{BitDecomp}(\mathbf{C}) \in \mathbb{Z}_q^{n\times d\beta}$ and $\bar{\mathbf{C}} \cdot \mathbf{G} = \mathbf{C}$. Moreover, when a is an element in the ring $R_q = R/qR$ where $R = \mathbb{Z}[x]/\varPhi_m[X]$, a can be represented as a vector in $\mathbb{Z}_q^n$ and we can apply $\mathsf{BitDecomp}$ and $\mathsf{Powersof2}$ algorithms to a as well.

2.4 Cryptographic Definitions

Definition 3. *A leveled multi-hop, multi-key FHE scheme is a tuple of efficient randomized algorithms (Setup, Gen, Enc, Dec, Eval) described as follows:*

- *$\mathsf{Setup}(1^\lambda, 1^K, 1^L)$: Given the security parameter λ, a bound K on the number of keys, and a bound L on the circuit depth, output a public parameter pp.*
- *$\mathsf{Gen}(pp)$: Given the public parameter pp, output public key pk_i and secret key sk_i $(i = 1, \ldots, K)$ for each party.*
- *$\mathsf{Enc}(pp, pk_i, \mu)$: Given the public key pk_i of party i and a message μ, output a ciphertext ct_i. Without loss of generality, ct_i contains the index of corresponding secret key and the level tag.*
- *$\mathsf{Dec}(pp, (sk_{i_1}, sk_{i_2}, \ldots, sk_{i_k}), ct_S)$: Given a ciphertext ct_S corresponding to a set of parties $S = \{i_1, \ldots, i_k\} \subseteq [K]$ and their secret keys $sk_{i_1}, sk_{i_2}, \ldots, sk_{i_k}$, output the message μ.*

– $\mathsf{Eval}(pp, \mathcal{C}, (ct_{S_1}, pk_{S_1}), \ldots, (ct_{S_t}, pk_{S_t}))$: *Given (a description of) a boolean circuit $\mathcal{C}$ along with t tuples (ct_{S_i}, pk_{S_i}), each comprising of a ciphertext ct_{S_i} corresponding to a set of secret keys indexed by $S_i = \{i_1, \ldots, i_{k_i}\} \subseteq [K]$ and a set of public keys $pk_{S_i} = \{pk_j, \forall j \in S_i\}$, output a ciphertext ct corresponding to the set of secret keys indexed by $S = \bigcup_{i=1}^{t} S_i \subseteq [K]$.*

Notice that the input ciphertexts of Eval can be fresh or the intermediate results of any homomorphic operations, which is allowed by the multi-hop property.

Definition 4 (Correctness). *A leveled multi-hop, multi-key FHE scheme is correct if for any circuit $\mathcal{C}$ of depth at most L having t input wires and any tuples $\{(ct_{S_i}, pk_{S_i})\}_{i\in[t]}$, letting $\mu_i = \mathsf{Dec}(sk_{S_i}, ct_{S_i})$, where $sk_{S_i} = \{sk_j, \forall j \in S_i\}$, $i = 1, \ldots t$, it holds that*

$$\Pr\left[\mathsf{Dec}(sk_S, \mathsf{Eval}(\mathcal{C}, (ct_{S_1}, pk_{S_1}), ..., (ct_{S_t}, pk_{S_t}))) \neq \mathcal{C}(\mu_1, \ldots, \mu_t)\right] = negl(\lambda),$$

where $S = \bigcup_{i=1}^{t} S_i$, $pp \leftarrow \mathsf{Setup}(1^\lambda, 1^K, 1^L)$, $(pk_j, sk_j) \leftarrow \mathsf{Gen}(pp)$ for $j \in [S]$.

Definition 5 (Compactness). *A leveled multi-hop, multi-key FHE scheme is compact if there exists a polynomial $poly(\cdot,\cdot,\cdot)$ such that in Definition 3, $|ct| \leq poly(\lambda, K, L)$. In other words, the length of ct is independent of the size of $\mathcal{C}$, but can depend polynomially on λ, K, and L.*

3 GSW Scheme with Ring Element Plaintext

In this section, we describe a variant of ring-LWE based GSW scheme with ring element plaintext, which can also be converted to a MKFHE scheme using the key extension technique in [CM15,MW16]. As explained in the introduction, this scheme will be used for the evaluation key generation in the Eval algorithm of the MKFHE scheme. The analogous LWE based scheme can be similarly constructed without effort, so we omit the description.

3.1 Basic Scheme

Here we present basic algorithms of our ring-GSW scheme. The differences between our scheme and the original ring-LWE based GSW scheme in [GSW13] include the following. First, the plaintext here is a R_q ring element instead of one bit, so our scheme do not support the general homomorphic multiplication gate. But we show that in a special case that the second plaintext has a small l_1 norm, one homomorphic multiplication is allowed. Second, the plaintext in our scheme is encrypted in low bits for the convenience of transformation to the evaluation key of the BGV scheme. Third, the decryption algorithm of our scheme is not presented, since it will not be used in our construction.

Our scheme is parameterized by an integer m (that defines the cyclotomic polynomial Φ_m and $\phi(m) = n$), a modulus $q(= poly(n))$, a small constant integer p, a (canonical norm) B-bounded discrete distribution χ in $R = \mathbb{Z}[X]/\Phi_m$ for $B \ll q$ and an integer $N = O(n \log q)$. Let $\beta = \lfloor \log q \rfloor + 1$. We use ring $R_q = R/qR$.

RGSW.Keygen(1^n): Sample $z \in R$ with a distribution χ, then we define the secret key as a vector $\mathbf{s} = (1, -z)^T \in R_q^2$. Pick a random vector $\mathbf{a} \in R_q^{2\beta}$ uniformly at random and vectors $\mathbf{e} \in R^{2\beta}$ with a distribution $\chi^{2\beta}$. Output the public key as

$$\mathbf{P} = [\mathbf{a}z + p\mathbf{e}, \mathbf{a}] = [\mathbf{b}, \mathbf{a}] \in R_q^{2\beta \times 2}.$$

RGSW.EncRand(μ,$\mathbf{P}$): This procedure is to generate the encryption of randomness that is used in the real encryption. When input $\mu \in R_q$, pick β ring elements $r_i \leftarrow \chi$ for $i = 1, \ldots, \beta$ and two vectors $\mathbf{e}'_1, \mathbf{e}'_2 \leftarrow \chi^\beta$, and output

$$\mathsf{RGSW.EncRand}_{\mathbf{s}}(\mu) = \mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2] \in R_q^{\beta \times 2},$$

where for $i = 1 \ldots \beta$,

$$\mathbf{f}_1[i] = \mathbf{b}[i]r_i + p\mathbf{e}'_1[i] + \mathsf{Powersof2}(\mu)[i] \in R_q$$

and

$$\mathbf{f}_2[i] = \mathbf{a}[i]r_i + p\mathbf{e}'_2[i] \in R_q.$$

Notice that $\mathbf{F}\mathbf{s} = p\tilde{\mathbf{e}} + \mathsf{Powersof2}(\mu)^T \in R_q^\beta$ for some small $\tilde{\mathbf{e}} \in R^\beta$. In fact, $\tilde{\mathbf{e}}[i] = \mathbf{e}[i]r_i + \mathbf{e}'_1[i] - \mathbf{e}'_2[i]z$ for $i = 1, \ldots, \beta$.

RGSW.Enc(μ,$\mathbf{P}$): On inputs $\mu \in R_q$ and the public key $\mathbf{P}$, pick a random ring element $r \xleftarrow{\$} \chi$ and an error matrix $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2] \leftarrow \chi^{2\beta \times 2}$, and output

$$\begin{aligned}\mathsf{RGSW.Enc}(\mu)_{\mathbf{s}} = \mathbf{C} &= r\mathbf{P} + p\mathbf{E} + \mu\mathbf{G} \\ &= [r\mathbf{b}, r\mathbf{a}] + p\mathbf{E} + \mu\mathbf{G} \\ &= [r\mathbf{a}z + p(r\mathbf{e} + \mathbf{e}_1), r\mathbf{a} + p\mathbf{e}_2] + \mu\mathbf{G} \in R_q^{2\beta \times 2},\end{aligned}$$

where $\mathbf{G} = (\mathbf{I}, 2\mathbf{I}, \ldots, 2^{\beta-1}\mathbf{I})^T \in R_q^{2\beta \times 2}$, and

$$\mathsf{RGSW.EncRand}(r, \mathbf{P}) = \mathbf{F} \in R_q^{\beta \times 2}.$$

Notice that $\mathbf{C} \cdot \mathbf{s} = p\tilde{\mathbf{e}} + \mu\mathbf{G}\mathbf{s} \in R_q^{2\beta}$ for some small $\tilde{\mathbf{e}}$. The corresponding decryption algorithm is not provided.

RGSW.HomAdd($\mathbf{C}_1, \mathbf{C}_2$): Addition of two ciphertext matrices is just standard addition in R_q.

RGSW.HomMult($\mathbf{C}_1, \mathbf{C}_2$): On input two ciphertexts $\mathbf{C}_1, \mathbf{C}_2 \in R_q^{2\beta \times 2}$, first computes the bit decomposition $\overline{\mathbf{C}}_1 = [\mathbf{D}_0, \ldots, \mathbf{D}_{\beta-1}]^T \in R_q^{2\beta \times 2\beta}$ of $\mathbf{C}_1$ such that $\mathbf{C}_1 = \sum_{i=0}^{\beta-1} 2^i \mathbf{D}_i$, and then present the multiplication as

$$\mathbf{C}_1 \odot \mathbf{C}_2 := \overline{\mathbf{C}}_1 \cdot \mathbf{C}_2.$$

The homomorphic multiplication can be accelerated using FFT/NTT as [DM15]. Notice that $\mathsf{RGSW.HomMult}$ operation can not always output a legal ciphertext with small noise. But in a special case that $\mathbf{C}_2$ encrypts a plaintext with a small l_1 norm, the noise in the output will be small. A rigorous analysis will be provided in Subsection 3.3.

RGSW.CTExt $(\mathbf{C}_i, \mathbf{F}_i, \{\mathbf{P}_j, j = 1, \dots, k\})$*:* given a ciphertext $\mathbf{C}_i \in R_q^{2\beta\times 2}$, an encryption of randomness $\mathbf{F}_i$ and public keys of all parties, output an extended ciphertext as

$$\bar{\mathbf{C}} = \begin{bmatrix} \mathbf{C}_i & \cdots & \mathbf{X}_1 & \cdots & 0 \\ 0 & \ddots & \vdots & & 0 \\ \vdots & & \mathbf{C}_i & & \vdots \\ & & \vdots & \ddots & \\ 0 & \cdots & \mathbf{X}_k & \cdots & \mathbf{C}_i \end{bmatrix} \in R_q^{2k\beta\times 2k} \tag{3}$$

where each sub block $\mathbf{X}_j \in R_q^{2\beta\times 2}$ is constructed from $\mathbf{F}_i$ and $\{\mathbf{P}_j\}_{j=1,\dots,k}$ as $\mathbf{X}_j[u, :] = \mathsf{BitDecomp}(\tilde{\mathbf{b}}_j[u])\mathbf{F}_i \in R_q^2$ for $u = 1, \dots, 2\beta$.

3.2 Security

The view of the attacker is the following distribution $(\mathbf{P}, \mathbf{F}, \mathbf{C})$ generated via, $(sk, pk = \mathbf{P}) \leftarrow \mathsf{RGSW.Keygen}(params)$, $\mathbf{F} \leftarrow \mathsf{RGSW.EncRand}(r, \mathbf{P})$ and $\mathbf{C} \leftarrow \mathsf{RGSW.Enc}(\mu, \mathbf{P})$. We prove semantic security of our GSW scheme with ring element plaintext by relying on the semantic security of the underlying ring-LWE scheme [LPR13a,LPR13b]. The proof consists of the following hybrids:

– First, we change the public key $\mathbf{P}$ to a random matrix $R_q^{2\beta\times 2}$ according the ring LWE assumption.
– Second, we change the encryption of randomness $\mathbf{F}$ to β ring LWE encryption of 0.
– Third, we change the encryption $\mathbf{C}$ to 2β ring LWE encryption of 0.

Finally, this distribution is completely independent of the plaintext μ which concludes the proof of security.

3.3 Noise Growth

The noise growth by the evaluation of the homomorphic operation can be analysed by the following lemma.

Lemma 2. *Let* $\beta = \lfloor \log q \rfloor + 1$ *and* $k \geq 1$. *Let* $\mathbf{s} \in R_q^{2k}$ *be a secret key. Let* $\mathbf{C}_1, \mathbf{C}_2 \in R_q^{2k\beta\times 2k}$ *be ciphertexts that encrypt* $\mu_1, \mu_2 \in R_q$ *with noise vectors* $\mathbf{e}_1, \mathbf{e}_2 \in R^{2k\beta}$*, respectively. Let* $\mathbf{C}_{add} := \mathbf{C}_1 \oplus \mathbf{C}_2$ *and* $\mathbf{C}_{mult} := \mathbf{C}_1 \odot \mathbf{C}_2$. *Then, we have*

$$\mathbf{C}_{add}\mathbf{s} = p\mathbf{e}_{add} + (\mu_1 + \mu_2)\mathbf{G}\mathbf{s},$$

$$\mathbf{C}_{mult}\mathbf{s} = p\mathbf{e}_{mult} + (\mu_1\mu_2)\mathbf{G}\mathbf{s},$$

where $\mathbf{e}_{add} := \mathbf{e}_1 + \mathbf{e}_2$ *and* $\mathbf{e}_{mult} := \overline{\mathbf{C}}_1\mathbf{e}_2 + \mu_2\mathbf{e}_1$. *In particular,* $\|\mathbf{e}_{mult}\|_\infty^{can} \leq \tilde{O}(\phi(m)k)\|\mathbf{e}_2\|_\infty^{can} + \|\mu_2\|_1\|\mathbf{e}_1\|_\infty^{can}$.

Proof. The statements for $\mathbf{C}_{add}$ can be immediately proved. For $\mathbf{C}_{mult}$, we have

$$\begin{aligned}\mathbf{C}_{mult}\mathbf{s} &= \overline{\mathbf{C}}_1 \cdot \mathbf{C}_2\mathbf{s}\\ &= \overline{\mathbf{C}}_1 \cdot (p\mathbf{e}_2 + \mu_2\mathbf{G}\mathbf{s})\\ &= p\overline{\mathbf{C}}_1 \cdot \mathbf{e}_2 + \mu_2\mathbf{C}_1\mathbf{s}\\ &= p(\overline{\mathbf{C}}_1 \cdot \mathbf{e}_2 + \mu_2\mathbf{e}_1) + (\mu_1\mu_2)\mathbf{G}\mathbf{s}.\end{aligned}$$

Remind that $\overline{\mathbf{C}}_1 = \sum_{i=1}^{\beta} 2^i\mathbf{D}_i$, where each $\mathbf{D}_i \in R_q^{2k\beta\times 2k}$ has entries with coefficients in $\{0,1\}$. So the canonical norm of them are bounded by $\phi(m)$. Then we have

$$\|\mathbf{e}_{mult}\|_\infty^{can} \le \tilde{O}(\phi(m)k)\|\mathbf{e}_2\|_\infty^{can} + \|\mu_2\|_1\|\mathbf{e}_1\|_\infty^{can}.$$

□

From the above lemma, we can see that the noise term in $\mathbf{C}_{mult}$ is only concerned with the l_1 norm of μ_2. From this observation, we get the following important corollary.

Corollary 1. *Let $\beta = \lfloor \log q\rfloor + 1$, $k \ge 1$ and $\phi(m) = n$. Let $\mathbf{C}_1, \mathbf{C}_2 \in R_q^{2k\beta\times 2k}$ be ciphertexts that encrypt $\mu_1, \mu_2 \in R_q$ with B bounded distribution noise vectors $\mathbf{e}_1, \mathbf{e}_2 \in R^{2\beta} \simeq \mathbb{Z}^{2\beta\phi(m)}$, respectively. $\mathbf{C}_{mult}$ and $\mathbf{e}_{mult}$ is defined as before. If $\|\mu_2\|_\infty \le 1$, we have $\| \mathbf{e}_{mult} \|_\infty \le \tilde{O}(n)\cdot B$.*

Proof. From Lemma 2, we have

$$\|\mathbf{e}_{mult}\|_\infty^{can} \le \tilde{O}(k\phi(m))\|\mathbf{e}_2\|_\infty^{can} + \|\mu_2\|_1\|\mathbf{e}_1\|_\infty^{can}.$$

Since $\|\mu_2\|_\infty \le 1$, $\|\mu_2\|_1 \le n$. So by (2) we have

$$\| \mathbf{e}_{mult} \|_\infty \le c_m \| \mathbf{e}_{mult} \|_\infty^{can} \le \tilde{O}(kn)\cdot B.$$

□

3.4 Correctness of Ciphertext Extension

In this subsection, we will explain the method of [CM15,MW16] to extend GSW ciphertexts corresponding to one single secret key to larger dimensional GSW ciphertexts corresponding to a concatenation of multiple keys.

Specifically, let $\mathbf{C}_i \in R_q^{2\beta\times 2}$ be a GSW ciphertext encrypting the message μ under secret key $\mathbf{s}_i = (1, -z_i)^T \in R_q^2$, i.e.,

$$\begin{aligned}\mathbf{C}_i &= r_i\,[\mathbf{a}z_i + p\mathbf{e}_i, \mathbf{a}] + \mathbf{E} + \mu\mathbf{G}\\ &= r_i\,[\mathbf{a}z_i + p\mathbf{e}_i, \mathbf{a}] + \mathbf{E} + \mu\mathbf{G} \in R_q^{2\beta\times 2}.\end{aligned} \tag{4}$$

Given a sequence of public vectors from different parties

$$\mathbf{b}_j = \mathbf{a}z_j + p\mathbf{e}_j \in R_q^{2\beta}, j = 1, \ldots, i-1, i+1, \ldots, k$$

and the i-th party's encryptions of the randomness

$$\mathsf{RGSW.EncRand}(r_i, pk_i) = \mathbf{F}_i \in R_q^{\beta \times 2},$$

we show that the $\mathbf{C}_i$ can be extended to a larger GSW ciphertext $\bar{\mathbf{C}} \in R_q^{2k\beta \times 2k}$ encrypting the same message μ under the secret key $\bar{\mathbf{s}} = (\mathbf{s}_1 | \dots | \mathbf{s}_k) \in R_q^{2k}$ for $\mathbf{s}_j = (1, -z_j)^T \in R_q^2,\ j \in [k]$, such that

$$\bar{\mathbf{C}} \cdot \bar{\mathbf{s}} = p\mathbf{e} + \mu \bar{\mathbf{G}} \bar{\mathbf{s}},$$

where $\tilde{\mathbf{e}} \in R^{2k\beta}$ is a small noise vector. Here the matrix $\bar{\mathbf{G}}$ can be written as

$$\bar{\mathbf{G}} = \Big[\mathbf{I}_{2k}, 2\mathbf{I}_{2k}, \dots, 2^{\lfloor \log q \rfloor}\mathbf{I}_{2k}\Big]^T \in R_q^{2k\beta \times 2k}.$$

Let the extended ciphertext

$$\bar{\mathbf{C}} = \begin{bmatrix} \mathbf{C}_i & \cdots & \mathbf{X}_1 & \cdots & 0 \\ 0 & \ddots & \vdots & & 0 \\ \vdots & & \mathbf{C}_i & & \vdots \\ & & \vdots & \ddots & \\ 0 & \cdots & \mathbf{X}_k & \cdots & \mathbf{C}_i \end{bmatrix} \in R_q^{2k\beta \times 2k} \tag{5}$$

to be a matrix whose sub blocks in $R_q^{2\beta \times 2}$ are all zero except the ones in the diagonal line and the ith column. Since $\mathbf{C}_i \mathbf{s}_i = p\mathbf{e}_i + \mu \mathbf{G} \mathbf{s}_i$, we also need to make sure that

$$\mathbf{X}_j \mathbf{s}_i + \mathbf{C}_i \mathbf{s}_j = p\tilde{\mathbf{e}} + \mu \mathbf{G} \mathbf{s}_j, \tag{6}$$

where $\tilde{\mathbf{e}} \in R^{2\beta}$ is a small noise vector.

Therefore, for $\mathbf{s}_i = (1, -z_i)^T$ and $\mathbf{s}_j = (1, -z_j)^T$, we can define

$$\tilde{\mathbf{b}}_j = \mathbf{b}_j - \mathbf{b}_i \in R_q^{2\beta}.$$

Let the uth row of $\mathbf{X}_j$ be

$$\mathbf{X}_j[u, :] = \mathsf{BitDecomp}(\tilde{\mathbf{b}}_j[u]) \mathbf{F}_i \in R_q^2 \tag{7}$$

for $u = 1, \dots, 2\beta$. Hence

$$\begin{aligned} \mathbf{X}_j[u, :]\mathbf{s}_i &= \Big(\mathsf{BitDecomp}(\tilde{\mathbf{b}}_j[v]) \mathbf{F}_i\Big) \mathbf{s}_i \\ &= \mathsf{BitDecomp}(\tilde{\mathbf{b}}_j[v]) \cdot \big(p\mathbf{e} + \mathsf{Powersof2}(r_i)^T\big) \\ &= pe' + \tilde{\mathbf{b}}_j[v] \cdot r_i, \end{aligned}$$

and

$$\mathbf{X}_j \mathbf{s}_i = p\mathbf{e}' + r_i \tilde{\mathbf{b}}_j \tag{8}$$

where $\mathbf{e}'$ is bounded by βB (canonical norm). According to the equation (4), we have

$$\begin{aligned}\mathbf{C}_i\mathbf{s}_j &= r_i(\mathbf{a}z_i + p\mathbf{e}) - r_i\mathbf{a}z_j + \mathbf{E}\mathbf{s}_j + \mu\mathbf{G}\mathbf{s}_j \\ &= r_i\left(\mathbf{b}_i - \mathbf{b}_j\right) + \mathbf{E}\mathbf{s}_j + \mu\mathbf{G}\mathbf{s}_j\end{aligned}.$$

Therefore, as the equation (6) holds for $\tilde{\mathbf{e}} = \mathbf{e}' + \mathbf{E}\mathbf{s}_j$ which is bounded by βB^2 (canonical norm).

Formally, the ciphertext extension algorithm can be described as follow.

– $\mathsf{RGSW.CTExt}\left(\mathbf{C}_i, \mathbf{F}_i, \{\mathbf{P}_j, j = 1, \ldots, k\}\right)$, given a ciphertext $\mathbf{C}_i \in R_q^{2\beta\times 2}$, an encryption of randomness $\mathbf{F}_i$ and public keys of all parties, output an extended ciphertext as (5) where each sub block $\mathbf{X}_j \in R_q^{2\beta\times 2}$ is constructed from $\mathbf{F}_i$ and $\{\mathbf{P}_j\}_{j=1,\ldots,k}$ as (7).

4 New Construction of ring-LWE MKFHE

In this section, we present the details of our method to extend the BGV scheme to a MKFHE scheme. As explained in Definition 3, MKFHE consists of five algorithms, i.e., MKFHE.Setup, MKFHE.Gen, MKFHE.Enc, MKFHE.Dec and MKFHE.Eval. For convenience, in the following we use $\mathsf{RGSW.Enc}_{\mathbf{s}}(\mu)$ (presented in Section 3) to denote a GSW ciphertext (which may be not fresh) that can be decrypted to μ with the secret key $\mathbf{s}$. Also we directly adopt the same subroutines such as modulus switching ModulusSwitch and key switching SwitchKey as the single key BGV scheme. For details of the original BGV scheme, see Appendix A.

4.1 Basic Schemes

MKFHE.Setup$(1^\lambda, 1^K, 1^L)$*:* Given the security parameter λ, a bound K on the number of keys, and a bound L on the circuit depth, generate the noise distribution $\chi = \chi(\lambda, K, L)$ which is a B-bounded distribution over R, L decreasing modules $q_L \gg q_{L-1} \gg \cdots \gg q_0$ for each level and a small integer p coprime with all q_l's. Let $\beta_l = \lfloor \log q_l \rfloor + 1$, and choose $L+1$ random public vectors $\mathbf{a}_l \in R_{q_l}^{2\beta_l}$ for $l = L, \ldots, 0$. All the following algorithms implicitly take the public parameter $pp = \big(R, \chi, B, \{q_l, \mathbf{a}_l\}_{l\in\{L,\ldots,0\}}, p\big)$ as input.

MKFHE.Gen$(j \in [K])$*:* Generate keys for the j-th party. For l from L down to 0, do the following:

1. Choose $z_{l,j} \leftarrow \chi$, and set $\mathbf{s}_{l,j} := (1, -z_{l,j})^T \in R_{q_l}^2$. The secret key for the j-th party is $sk_j = \{\mathbf{s}_{l,j}\}_{l\in\{L,\ldots,0\}}$.
2. Generate $2\beta_l$ ring-LWE instances

$$pt_{l,j} := [\mathbf{b}_{l,j} = \mathbf{a}_l z_{l,j} + p\mathbf{e}_{l,j}, \mathbf{a}_l] \in R_{q_l}^{2\beta_l\times 2},$$

where $\mathbf{e}_{l,j} \leftarrow \chi^{2\beta_l}$. The public key pk_j for the j-th party consists of all the $pt_{l,j},\ l = L, \ldots, 0$.

3. For $i = 1, \ldots, 2\beta_l$, compute $\mathsf{RGSW.Enc}\left(\mathsf{Powersof2}(\mathbf{s}_{l,j})[i], pt_{l-1,j}\right)$ and get

$$\begin{aligned}\Phi_{i,l,j} &= \mathsf{RGSW.Enc}_{\mathbf{s}_{l-1,j}}\left(\mathsf{Powersof2}(\mathbf{s}_{l,j})[i]\right) \\ &= r_{i,l,j}\left[\mathbf{b}_{l-1,j}, \mathbf{a}_{l-1}\right] + p\mathbf{E}_{i,l,j} + \mathsf{Powersof2}(\mathbf{s}_{l,j})[i]\mathbf{G}\end{aligned}$$

together with

$$\mathbf{F}_{i,l,j} = \mathsf{RGSW.EncRand}(r_{i,l,j}, pt_{l-1,j}) \in R_{q_l}^{\beta_l \times 2}.$$

Also compute

$$\begin{aligned}\Psi_{i,l,j} &= \mathsf{RGSW.Enc}_{\mathbf{s}_{l-1,j}}\left(\mathsf{BitDecomp}(\mathbf{s}_{l,j})[i]\right) \\ &= r'_{i,l,j}\left[\mathbf{b}_{l-1,j}, \mathbf{a}_{l-1}\right] + p\mathbf{E}'_{i,l,j} + \mathsf{BitDecomp}(\mathbf{s}_{l,j})[i]\mathbf{G}\end{aligned}$$

together with

$$\mathbf{F}'_{i,l,j} = \mathsf{RGSW.EncRand}(r'_{i,l,j}, pt_{l-1,j}) \in R_{q_l}^{\beta_l \times 2}.$$

The evaluation key generation material is

$$em_j = \left\{\left(\Phi_{i,l,j}, \mathbf{F}_{i,l,j}\right), \left(\Psi_{i,l,j}, \mathbf{F}'_{i,l,j}\right)\right\}_{i\in[2\beta_l], l\in[L]}.$$

Later, the em_j will be used to generate evaluation keys for the homomorphic evaluation algorithm.

MKFHE.Enc(pk_j, μ)*:* Given the public key pk_j of the j-th party and a message $\mu \in R_p$, choose a random ring element $r \in R_2$. Similar to the BGV scheme, the level-L ciphertext $\mathbf{c} = (c^0, c^1) \in R_{q_L}^2$ encrypts a plaintext element $\mu \in R_p$ with respect to $\mathbf{s}_L = (1, -z_L)$, where

$$c^0 = r\mathbf{b}_{L,j}[1] + pe + \mu \in R_{q_L} \text{ and } c^1 = r\mathbf{a}_L[1] + pe' \in R_{q_L}.$$

Let S be an ordered set containing all indexes of the parities that the ciphertext corresponding to. Without loss of generality, we assume that the indexes in S are always arranged from small to large and S has no duplicate elements. Here we set $S = \{j\}$. Usually, the ciphertext ct contains $\mathbf{c}$, the set S and a tag l to label the number of the level. Finally, output a tuple $ct = (\mathbf{c}, \{j\}, L)$.

MKFHE.Dec$(sk_S, ct = (\mathbf{c}, S, l))$*:* Suppose $S = \{j_1, \ldots, j_k\}$ and sk_S consists of all the parties' secret keys whose indexes are contained in S, i.e., $sk_S = \{sk_{j_1}, \ldots, sk_{j_k}\}$. Let

$$\bar{\mathbf{s}}_l = (\mathbf{s}_{l,j_1} | \mathbf{s}_{l,j_2} | \cdots | \mathbf{s}_{l,j_k}) \in R_{q_l}^{2k},$$

where $\mathbf{s}_{l,j}$ is the key of the j-th party to decrypt level-l ciphertexts. Once given a level-l ciphertext $\mathbf{c} \in R_{q_l}^{2k}$, compute

$$\mu = \langle \mathbf{c}, \bar{\mathbf{s}}_l \rangle \mod q_l \mod p.$$

MKFHE.Eval$((pk_{i_1},\ldots,pk_{i_k}), em_S, \mathcal{C}, (ct_1,\ldots,ct_t))$*:* Assume that the sequence of ciphertexts are at the same level-l (If needed, use SwitchKey and ModulusSwitch to make it so). For $j \in [t]$, parse ct_j as $(\mathbf{c}_j, S_j, l)$, let $|S_j| = k_j, S = \bigcup_{j=1}^{t} S_j = \{i_1,\ldots,i_k\}, pk_S = (pk_{i_1},\ldots,pk_{i_k})$, and thus $\mathbf{c}_j \in R_{q_l}^{2k_j}$. Then the outline of the evaluation of the Boolean circuit $\mathcal{C}$ is as follows.

1. For $j \in [t]$, compute MKFHE.CTExt$(\mathbf{c}_j, S) = \bar{\mathbf{c}}_j$ to get extended $2k$ dimensional ciphertexts which encrypts the same message under the key $\bar{\mathbf{s}}_l$. Here $\bar{\mathbf{s}}_l := (\mathbf{s}_{l,i_1},\ldots,\mathbf{s}_{l,i_k})$ is indexed by S.
2. Compute MKFHE.EVKGen$(em_S) = evk_S$ to generate the evaluation key for the extended scheme.
3. Call the two basic homomorphic operations for the extended ciphertexts MKFHE.EvalAdd$(evk_S, \bar{\mathbf{c}}_i, \bar{\mathbf{c}}_j)$ and MKFHE.EvalMult$(evk_S, \bar{\mathbf{c}}_i, \bar{\mathbf{c}}_j)$ to evaluate each gate of the circuit $\mathcal{C}$.

Note that, we have given a detailed description of the first four algorithms MKFHE.Setup, MKFHE.Gen, MKFHE.Enc and MKFHE.Dec. For MKFHE.Eval, we just provided an outline of the algorithm. In the following subsections, we will detail the ciphertext extension algorithm MKFHE.CTExt and the evaluation key generation algorithm MKFHE.EVKGen. Also, we will explain how to call the algorithm MKFHE.EvalAdd and MKFHE.EvalMult to evaluate addition and multiplication for larger dimensional ciphertexts.

4.2 The Ciphertext Extension

In this subsection, we detail the ciphertext extension algorithm MKFHE.CTExt which converts a BGV ciphertext to a larger dimensional ciphertext under a new larger dimensional secret key. In fact, the new secret key is a concatenation of secret keys from a larger set of parties.

MKFHE.CTExt(ct, S')*:* On input a ciphertext $ct = (\mathbf{c}, S, l)$ and a set of parties's indexes S' for $S \subseteq S'$, where S has k members $\{i_1, i_2, \ldots, i_k\}$ and S' has k' members $\{j_1, j_2, \ldots, j_{k'}\}$ for $k' > k$. $\mathbf{c} \in R_{q_l}^{2k}$ corresponds to the decryption key $\mathbf{s}_l \in R_{q_l}^{2k}$, so $\langle \mathbf{c}, \mathbf{s}_l \rangle \bmod q_l = pe + \mu$. Sequentially divide $\mathbf{c}$ into k sub-vectors which can be indexed by $S = \{i_1, i_2, \ldots, i_k\}$, i.e.,

$$\mathbf{c} = (\mathbf{c}_{i_1} | \mathbf{c}_{i_2} | \cdots | \mathbf{c}_{i_k}) \in R_{q_l}^{2k}$$

where each $\mathbf{c}_{i_1} \in R_{q_l}^{2}$. The extended ciphertext $\bar{\mathbf{c}} \in R_{q_l}^{2k'}$ consists of k' sequential sub-vectors of 2 dimensional, which can be indexed by $S' = \{j_1, j_2, \ldots, j_{k'}\}$, i.e.,

$$\bar{\mathbf{c}} = \left(\mathbf{c}'_{j_1} | \mathbf{c}'_{j_2} | \cdots | \mathbf{c}'_{j_{k'}} \right) \in R_{q_l}^{2k'}.$$

If an index j in S' is also included in S, we set $\mathbf{c}'_j = \mathbf{c}_j$, otherwise $\mathbf{c}'_j = 0$.

Obviously, $\bar{\mathbf{c}}$ corresponds to the secret key

$$\bar{\mathbf{s}}_l = \left(\mathbf{s}_{j_1,l} | \mathbf{s}_{j_2,l} | \cdots | \mathbf{s}_{j_{k'},l} \right) \in R_{q_l}^{2k'},$$

where $\mathbf{s}_{j,l}$ is the key of the j-th party to decrypt the level-l ciphertexts. And the decryption is performed by the inner product and modulus, i.e.,

$$\langle \bar{\mathbf{c}}, \bar{\mathbf{s}}_l \rangle = \sum_{t=1}^{k'} \langle \mathbf{c}'_{j_t}, \mathbf{s}_{j_t,l} \rangle = \sum_{\iota=1}^{k} \langle \mathbf{c}_{i_\iota}, \mathbf{s}_{i_\iota,l} \rangle = \langle \mathbf{c}, \mathbf{s}_l \rangle = pe + \mu, \tag{9}$$

and $\mu = \langle \bar{\mathbf{c}}, \bar{\mathbf{s}}_l \rangle \mod q_l \mod p$. The second equality in (9) holds because other $\mathbf{c}'_j$'s are all 0.

4.3 Homomorphic Operations

In this subsection, we explain how to perform the algorithms MKFHE.EvalAdd and MKFHE.EvalMult on extended ciphertexts when a proper evaluation key is provided. The evaluation key we needed is

$$\tau_{\bar{\mathbf{s}}'_l \to \bar{\mathbf{s}}_{l-1}} = \{\mathcal{K}_{t,\zeta}\}_{t=1,\dots,\beta_l;\zeta=1,\dots,4k^2} \tag{10}$$

for $\bar{s}'_l = \bar{s}_l \otimes \bar{s}_l$ and $\mathcal{K}_{t,\zeta} \in R_{q_l}^{2k}$ such that $\langle \mathcal{K}_{t,\zeta}, \bar{\mathbf{s}}_{l-1} \rangle = pe_{t,\zeta} + 2^{t-1}\bar{\mathbf{s}}'_l[\zeta] \in R_{q_l}$ and the canonical norm of $e_{t,\zeta}$ is small.

MKFHE.EvalAdd$(evk_S, \bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$*:* Take two (extended) ciphertexts $\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2 \in R_{q_l}^{2k}$ at the same level-l under the same $\bar{\mathbf{s}}_l$ as inputs (If needed, use SwitchKey and ModulusSwitch to make it so). First, compute $\bar{\mathbf{c}}'_3 \leftarrow \bar{\mathbf{c}}_1 + \bar{\mathbf{c}}_2 \mod q_l$ under the secret key $\bar{\mathbf{s}}_l \in R_{q_l}^{2k}$. Second, use SwitchKey$(\bar{\mathbf{c}}'_3, \tau_{\bar{\mathbf{s}}'_l \to \bar{\mathbf{s}}_{l-1}}, q_l)$ to generate ciphertext $\bar{\mathbf{c}}''_3$ under the secret key $\bar{\mathbf{s}}_{l-1}$ ($\bar{\mathbf{s}}'_l$'s coefficients include all of $\bar{\mathbf{s}}_l$'s since $\bar{\mathbf{s}}'_l = \bar{\mathbf{s}}_l \otimes \bar{\mathbf{s}}_l$ and $\bar{\mathbf{s}}_l$'s first coefficient is 1). Third, compute $\bar{\mathbf{c}}_3 =$ ModulusSwitch$(\bar{\mathbf{c}}''_3, l)$.

MKFHE.EvalMult$(evk_S, \bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$*:* Take two (extended) ciphertexts $\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2 \in R_{q_l}^{2k}$ at the same level-l under the same $\bar{\mathbf{s}}_l$. (If needed, use SwitchKey and ModulusSwitch to make it so). First, compute $\bar{\mathbf{c}}'_3 \leftarrow \bar{\mathbf{c}}_1 \otimes \bar{\mathbf{c}}_2 \mod q_l$ under the secret key $\bar{\mathbf{s}}_l \in R_{q_l}^{2k}$. Second, use SwitchKey$(\bar{\mathbf{c}}'_3, \tau_{\bar{\mathbf{s}}'_l \to \bar{\mathbf{s}}_{l-1}}, q_l)$ to generate a ciphertext $\bar{\mathbf{c}}''_3$ under the secret key $\bar{\mathbf{s}}'_l = \bar{\mathbf{s}}_l \otimes \bar{\mathbf{s}}_l$. Third, compute $\bar{\mathbf{c}}_3 =$ ModulusSwitch$(\bar{\mathbf{c}}''_3, l)$.

4.4 Evaluation Key Generation

In this subsection, we detail the evaluation key generation algorithm EVKGen, which inputs the public keys of involved parties and outputs the extended BGV evaluation key as (10). Remind that all parties share L common random public matrices $\mathbf{a}_l \in R_{q_l}^{2\beta_l}$ for $l = L, \dots, 0$ and $\beta_l = \lfloor \log q_l \rfloor + 1$. The evaluation key generation material em_j for the jth party consists of all the $\Phi_{i,l,j}$, $\Psi_{i,l,j}$, $\mathbf{F}_{i,l,j}$ and $\mathbf{F}'_{i,l,j}$ for $l = L, \dots, 0$ and $i = 1, \dots, 2\beta_l$.

MKFHE.EVKGen(em_S, pk_S*).* Notice that S contains k elements, and em_S consists of a collection of evaluation key generation materials $\{em_{j_1}, \dots, em_{j_k}\}$ and the public keys $\{pk_{j_1}, \dots, pk_{j_k}\}$ belonging to parties in S. To generate a level-l evaluation key as (10), compute as follows.

1. For each $j^* \in S$, use the GSW extend algorithm to get larger dimensional ciphertexts under a key $\bar{\mathbf{s}}_{l-1}$
$$\begin{aligned}\bar{\varPhi}_{i,l,j^*} &= \mathsf{RGSW.CTExt}\left(\varPhi_{i,l,j^*}, pk_S, \mathbf{F}_{i,l,j^*}\right)\\ &= \mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}\left(\mathsf{Powersof2}(\mathbf{s}_{l,j^*})[i]\right)\end{aligned}$$
and
$$\begin{aligned}\bar{\varPsi}_{i,l,j^*} &= \mathsf{RGSW.CTExt}\left(\varPsi_{i,l,j^*}, pk_S, \mathbf{F}'_{i,l,j^*}\right)\\ &= \mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}\left(\mathsf{BitDecomp}(\mathbf{s}_{l,j^*})[i]\right)\end{aligned}$$
where $\bar{\mathbf{s}}_{l-1} = (\mathbf{s}_{l-1,j_1}|\mathbf{s}_{l-1,j_2}|\cdots|\mathbf{s}_{l-1,j_k}) \in R_{q_l}^{2k}$.
2. Set $\bar{\mathbf{s}}_l = (\mathbf{s}_{l,j_1}|\mathbf{s}_{l,j_2}|\cdots|\mathbf{s}_{l,j_k}) \in R_{q_l}^{2k}$ and $\bar{\mathbf{s}}'_l = \bar{\mathbf{s}}_l \otimes \bar{\mathbf{s}}_l \in R_{q_l}^{4k^2}$. If we can compute $\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}_l[\zeta]\cdot\bar{\mathbf{s}}_l[\zeta'])$ from $\left\{\bar{\varPhi}_{i,l,j}, \bar{\varPsi}_{i,l,j}\right\}$ and $\left\{\bar{\varPhi}_{i',l,j'}, \bar{\varPsi}_{i',l,j'}\right\}$, where $\bar{\mathbf{s}}_l[\zeta]$ and $\bar{\mathbf{s}}_l[\zeta']$ are any two elements of $\bar{\mathbf{s}}_l$, we have the GSW encryptions of all the elements of $\bar{\mathbf{s}}'_l$ under the key $\bar{\mathbf{s}}_{l-1}$. The details of how to accomplish this task will be explained later.
3. Given the $\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}'_l[\zeta])$, compute
$$\tau_{\bar{\mathbf{s}}'_l \to \bar{\mathbf{s}}_{l-1}} = \{\mathcal{K}_{t,\zeta}\}_{t=1,\ldots,\beta_l;\zeta=1,\ldots,4k^2} \tag{11}$$
for $\mathcal{K}_\zeta \in R_{q_l}^{2k}$ such that $\langle \mathcal{K}_{t,\zeta}, \bar{\mathbf{s}}_{l-1}\rangle = pe_{t,\zeta} + 2^{t-1}\bar{\mathbf{s}}'_l[\zeta] \in R_{q_l}$. Also, the details will be provided later.

Details of Step 2. Since we need to compute the GSW encryptions of $\bar{\mathbf{s}}[\zeta]\cdot\bar{\mathbf{s}}[\zeta']$, the intuition may be the homomorphic multiplication of the GSW encryptions of $\bar{\mathbf{s}}[\zeta]$ and $\bar{\mathbf{s}}[\zeta'] \in R_q$. But the noise will be out of control in this way according to Lemma 2, because the absolute value of the message $\bar{\mathbf{s}}[\zeta']$ can be larger than $q_l/2$. Alternatively, we know that $\langle \mathsf{Powersof2}(\bar{\mathbf{s}}_l[\zeta]), \mathsf{BitDecomp}(\bar{\mathbf{s}}_l[\zeta'])\rangle = \bar{\mathbf{s}}_l[\zeta]\cdot\bar{\mathbf{s}}_l[\zeta']$. So we homomorphically compute the inner product of the GSW encryptions of $\mathsf{Powersof2}(\bar{\mathbf{s}}_l[\zeta]) = \mathsf{Powersof2}(\mathbf{s}_{l,j}[t])$ and $\mathsf{BitDecomp}(\bar{\mathbf{s}}_l[\zeta']) = \mathsf{BitDecomp}(\mathbf{s}_{l,j'}[t'])$, since $\zeta = 2(j-1)+t$ and $\zeta' = 2(j'-1)+t'$, $1 \le j \le k$, $t = 1$ or 2. Namely we compute
$$\begin{aligned}&\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}_l[\zeta]\cdot\bar{\mathbf{s}}_l[\zeta'])\\ &= \sum_{\iota=1}^{\beta_l}\left(\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}\left(\mathsf{Powersof2}(\bar{\mathbf{s}}_l[\zeta])[\iota]\right) \odot \mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}\left(\mathsf{BitDecomp}(\bar{\mathbf{s}}_l[\zeta'])[\iota]\right)\right)\\ &= \sum_{\iota=1}^{\beta_l}\left(\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}\left(\mathsf{Powersof2}(\mathbf{s}_{l,j}[t])[\iota]\right) \odot \mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}\left(\mathsf{BitDecomp}(\mathbf{s}_{l,j'}[t'])[\iota]\right)\right)\\ &= \sum_{\iota=1}^{\beta_l}\bar{\varPhi}_{\beta_l(t-1)+\iota,l,j} \odot \bar{\varPsi}_{\beta_l(t'-1)+\iota,l,j'}.\end{aligned} \tag{12}$$
The l_∞ norm of $\mathsf{BitDecomp}(\mathbf{s}_{l,j'}[t'])[\iota]$ is less than 1. According to Corollary 1, the canonical norm of the noise in the result ciphertext of homomorphic

multiplication is bounded by $\tilde{O}(n)B^*$ if the noise in the input ciphertexts is bounded by B^*. So the noise in the final output ciphertext $\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}_l[\zeta] \cdot \bar{\mathbf{s}}_l[\zeta'])$ of (12) is bounded by $\tilde{O}(n\beta_l^2)B^2$ for $\beta_l = \lfloor \log q_l \rfloor + 1$ if the noise in em_j is bounded by B.

Details of Step 3. After above procedure, we have the GSW ciphertext

$$\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}'_l[\zeta]) = \mathbf{C}_\zeta \ \in R_{q_l}^{2k\beta_l \times 2k}$$

so that

$$\mathbf{C}_\zeta \bar{\mathbf{s}}_{l-1} = p\mathbf{e} + \bar{\mathbf{s}}'_l[\zeta]\mathbf{G}\bar{\mathbf{s}}_{l-1}.$$

Since

$$\mathbf{G} = \left[\mathbf{I}_{2k}, 2\mathbf{I}_{2k}, \ldots, 2^{\lfloor \log q \rfloor}\mathbf{I}_{2k}\right]^T \in R_{q_l}^{2k\beta_l \times 2k},$$

let the $2t \cdot k + 1$th row of $\mathbf{C}_\zeta$ be $\mathbf{c}_{t,\zeta} \in R_{q_l}^{2k}$, so we have

$$\langle \mathbf{c}_{t,\zeta}, \bar{\mathbf{s}}_{l-1} \rangle = pe_{t,\zeta} + 2^{t-1}\bar{\mathbf{s}}'_l[\zeta] \in R_{q_l}$$

for some small $e_{t,\zeta}$. This is the evaluation key as (10).

4.5 Packing Ciphertexts

We show that if the underlying single key BGV ciphertexts is batched, we can get a batched multi-key FHE scheme. The extended ciphertext $\bar{\mathbf{c}} = (\mathbf{c}_1 | \ldots | \mathbf{c}_k) \in R_{q_l}^{2k}$ has $O(n)$ plaintext slots if the plaintext $\mu \in R_p$ has $O(n)$ slots by the Chinese Remainder Theorem. The $O(n)$-fold addition gate and the $O(n)$-fold multiplication gate can be evaluated directly by $\mathsf{MKFHE.EvalAdd}$ and $\mathsf{MKFHE.EvalMult}$ since the plaintext space is R_p. In the following we provide the homomorphic permutation operation. Given the extended ciphertext $\bar{\mathbf{c}} \in R_{q_l}^{2k}$, we first apply the automorphisms ρ_i as (15) to each ring element of $\mathbf{c}$. Since

$$\langle \bar{\mathbf{c}}, \bar{\mathbf{s}}_l \rangle = pe + \mu + k[X]\Phi_m[X],$$

we have the equality

$$\left\langle \bar{\mathbf{c}}[X^i], \bar{\mathbf{s}}[X^i]_l \right\rangle = pe[X^i] + \mu[X^i] + k[X^i]\Phi_m[X^i].$$

In view of $\Phi(X)$ divides $\Phi(X^i)$ for $i \in \mathbb{Z}_m^*$, $\bar{\mathbf{c}}[X^i] \in R_{q_l}^{2k}$ is an encryption of $\mu[X^i]$ under the key $\bar{\mathbf{s}}[X^i]$. So the homomorphic permutation is completed by $\mathsf{KeySwitching}$ and get an level-$(l-1)$ ciphertext which encrypts $\mu[X^i]$ under the key $\bar{\mathbf{s}}_{l-1}$.

In this case, the evaluation key generation material for the jth party should also include the $\mathsf{RGSW.Enc}\left(\mathbf{s}_{l,j}[X^i], pt_{l-1,j}\right)$ for $i \in \mathbb{Z}_m^*$. By applying the GSW ciphertext extension and extracting certain rows, we can successfully compute the evaluation key

$$\tau_{\bar{\mathbf{s}}'_l[X^i] \to \bar{\mathbf{s}}_{l-1}} = \{\mathcal{K}_{t,\zeta}\}_{t=1,\ldots,\beta_l;\zeta=1,\ldots,4k^2}$$

for $\mathcal{K}_\zeta \in R_{q_l}^{2k}$ such that $\langle \mathcal{K}_{t,\zeta}, \bar{\mathbf{s}}_{l-1} \rangle = pe_\zeta + 2^{t-1}\bar{\mathbf{s}}'_l[\zeta][X^i] \in R_{q_l}$.

4.6 Analysis

An analysis of the evaluation key generation procedure is as follows.

Lemma 3. *Assume the noise in each $\Phi_{i,l,j}$ and $\Psi_{i,l,j}$ is bounded by B, and k is the number of the parities involved in the evaluation. The noise of each evaluation key in (11) is bounded by $\tilde{O}(nk)B^2$.*

Proof. For $\beta_l = \lfloor \log q_l \rfloor + 1$, if the noise in each $\Phi_{i,l,j}$ and $\Psi_{i,l,j}$ is bounded by B, the noise in each $\bar{\Phi}_{i,l,j}$ and $\bar{\Psi}_{i,l,j}$ is bounded by $\beta_l B^2$ (canonical norm). According to Corollary 1, the noise in $\bar{\Phi}_{\beta_l(t-1)+\iota,l,j} \odot \bar{\Psi}_{\beta_l(t'-1)+\iota,l,j'}$ is bounded by $O(nk\beta_l)B^2$. So the noise in ciphertext $\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}_l[\zeta] \cdot \bar{\mathbf{s}}_l[\zeta'])$ in (12) is bounded by $O(nk\beta_l^2 B^2)$. The final evaluation key in (10) is just derived from the $\mathsf{RGSW.Enc}_{\bar{\mathbf{s}}_{l-1}}(\bar{\mathbf{s}}_l[\zeta] \cdot \bar{\mathbf{s}}_l[\zeta'])$, so the bound of noise is also $O(nk\beta_l^2)B^2 = \tilde{O}(nk)B^2$. □

An analysis of the homomorphic operation procedure is as follows.

Definition 6. *We say an (extended) BGV ciphertext $\bar{\mathbf{c}} \in R_{q_l}^{2k}$ $(k \geq 1)$ encrypts $\mu \in R_p$ under a key $\bar{\mathbf{s}}_l \in R_{q_l}^{2k}$ if $\langle \bar{\mathbf{c}}, \bar{\mathbf{s}}_l \rangle \bmod q_l = pe + \mu$.*

Lemma 4. *If the (extended) ciphertexts $\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2 \in R_{q_l}^{2k}$ $(k \geq 1)$ encrypt $\mu_1, \mu_2 \in R_p$, respectively, under a key $\bar{\mathbf{s}}_l \in R_{q_l}^{2k}$, the extended ciphertext $\bar{\mathbf{c}}_1 + \bar{\mathbf{c}}_2 \in R_{q_l}^{2k}$ encrypts $\mu_1 + \mu_2 \in R_p$ under the decryption key $\bar{\mathbf{s}}_l \in R_{q_l}^{2k}$.*

Lemma 5. *If the (extended) ciphertexts $\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2 \in R_{q_l}^{2k}$ $(k \geq 1)$ encrypt $\mu_1, \mu_2 \in R_p$, respectively, under the decryption key $\bar{\mathbf{s}}_l \in R_{q_l}^{2k}$, the extended ciphertext $\bar{\mathbf{c}}_1 \otimes \bar{\mathbf{c}}_2 \in R_{q_l}^{4k^2}$ encrypts the $\mu_1 \cdot \mu_2 \in R_p$ under the key $\bar{\mathbf{s}}'_l = \bar{\mathbf{s}}_l \otimes \bar{\mathbf{s}}_l \in R_{q_l}^{4k^2}$. Moreover, given the evaluation key as (10) where the canonical norm of $e_{t,\zeta}$ is bounded by B, we can use $\mathsf{SwitchKey}(\tau_{\bar{\mathbf{s}}'_l \to \bar{\mathbf{s}}_{l-1}}, \bar{\mathbf{c}}_1 \otimes \bar{\mathbf{c}}_2)$ to get $\bar{\mathbf{c}}^* \in R_{q_l}^{2k}$ which encrypts $\mu_1 \cdot \mu_2 \in \mathbb{Z}_p$ under the key $\bar{\mathbf{s}}_{l-1} \in R_{q_l}^{2k}$ with the noise bounded by $O(k^2\beta_l) \cdot B$. Here*

$$\bar{\mathbf{s}}_{l-1} = (\mathbf{s}_{l-1,j_1} | \mathbf{s}_{l-1,j_2} | \cdots | \mathbf{s}_{l-1,j_k}) \in R_{q_l}^{2k}, \tag{13}$$

where $\mathbf{s}_{l-1,j}$ is the key of the jth party to decrypt level-$(l-1)$ ciphertexts and the first entry of $\mathbf{s}_{l-1,j}$ is 1.

Assuming the noise in the public key pt_j and the evaluation key generation material em_j is bounded by B, the noise in the evaluation key is bounded by $\tilde{O}(kn) \cdot B$ according to Lemma 3. If the level-l ciphertexts have a noise bounded by B_l, the ciphertexts after homomorphic operations and before modulus switching have a noise bounded by $B_l^2 + \tilde{O}(k^3 n) \cdot B$ by lemma 5. Finally, we apply the Scale function. The noise is now at most

$$B_{l-1} = \frac{q_{l-1}}{q_l} \left(B_l^2 + \tilde{O}(k^3 n) \cdot B^2 \right) + \eta_{\mathsf{Scale},l}$$

where $\eta_{\mathsf{Scale},l}$ is an additive term. Let B_l be bounded by B_{max} for all l. Also we let $B_{max} \geq 2\left(\tilde{O}\left(K^3 n\right) \cdot B^2 + \eta_{\mathsf{Scale},l}\right)$ for all l and the upper bound of the parties' number K, and $q_l/q_{l-1} \geq 2 \cdot B_{max}$ for all l. Then we have

$$\begin{aligned} B_{l-1} =& \frac{q_{l-1}}{q_l}\left(B_l^2 + \tilde{O}(k^3 n) \cdot B^2\right) + \eta_{\mathsf{Scale},l} \\ \leq& \frac{q_{l-1}}{q_l} B_{max}^2 + \tilde{O}(k^3 n) \cdot B^2 + \eta_{\mathsf{Scale},l} \\ \leq& \frac{1}{2 \cdot B_{max}} B_{max}^2 + \frac{1}{2} B_{max} \\ \leq& B_{max}. \end{aligned}$$

Therefore, it is enough to set B_{max} as $poly(n, K)$ and the largest modulus q_L as $poly(n, K)^L$. For approximation factors of the presumed hardness, our scheme is $poly(n, K)^L$ due to the above analysis. So our scheme can similarly bootstrap as [BGV12].

4.7 Parameters and Comparisons

The comparisons of main properties of various schemes are provided in Table 1, Table 2 and Table 3. To ensure security, we can set the dimension of the underlying (ring-)LWE problem as $n = O(\lambda \log q_L) = \tilde{O}(\lambda L)$ for our scheme and $n = O(\lambda)$ for previous schemes, where λ is the the security parameter.

Comparison with [LATV12]. The first advantage over [LATV12] is that the security of our scheme is based on the LWE assumption or the ring-LWE assumption which is currently supported by a worst-case hardness theorem, but not on a somewhat non-standard assumption on polynomial rings such as the decisional small polynomial ratio (DSPR) assumption. The second advantage is that our construction admits a threshold decryption protocol, therefore can obtain a 2-round MPC, while only a "on-the-fly" MPC can be obtained from [LATV12]. Moreover, when [LATV12] is modified to avoid the recent sub-exponential attacks on the NTRU problem, our scheme still holds some advantages in efficiency. In fact, the attacks [ABD16,MSZ16,CJL16] have complexity $2^{\tilde{O}(\sqrt{n}/\log q)}$, where n is the degree of the ring, and q is the largest modulus in the modulus chain. To get security against attacks running in time 2^λ, we need $\log q > K \cdot L$ to support noise growth and $n > (\lambda K L)^2$ to thwart the attacks. This gives public key of size $\lambda^2 K^4 L^5$ and ciphertext of size $\lambda^2 K^3 L^3$ for [LATV12], while our ring-LWE based scheme has public key of size $\lambda^2 L^6$ and ciphertext of size $\lambda k L^2$.

Comparison with [PS16] and [BP16]. For approximation factors of the presumed hardness, our scheme is $poly(K, n)^L$ due to the above analysis, while [PS16] is $poly(K, n, L)^{K+L}$ and [BP16] is $poly(K, n)$. Comparing to [BP16], our scheme needs to take larger dimensions to compensate for larger approximation factors when L is large. But thanks to the ring element plaintext space and the

SIMD operations, our construction has much better amortized per-bit timing. Moreover, when considering the threshold decryption protocol, because of the Smudging Lemma, [PS16] and [BP16] also need exponential large modulus/error rate in λ and K as well as our scheme. In this case, [PS16] and [BP16] do not own an advantage in hardness assumptions when constructing a 2-round MPC protocol.

Scheme	Assumption	Public Key	Ciphertext/Plaintext	Key Hops	Batch
[CM15]	LWE	$\tilde{O}(\lambda^2 L^2)$	$\tilde{O}(k^2\lambda^2 L^2)$	Single	No
[CM15]	ring-LWE	$\tilde{O}(\lambda L^2)$	$\tilde{O}(k^2\lambda L^2)$	Single	No
[BP16]	LWE	$\tilde{O}(\lambda^3)$	$\tilde{O}(k\lambda)$	Multiple	No
[PS16] Scheme #1	LWE	$\tilde{O}(\lambda(K+L)^2)$	$\tilde{O}(k\lambda^3(K+L)^4)$	Multiple	No
[PS16] Scheme #2	LWE/KDM	$\tilde{O}(\lambda^4(K+L)^4)$	$\tilde{O}(k^2\lambda^2(K+L)^2)$	Multiple	No
Our Scheme	LWE	$\tilde{O}(\lambda^3 L^7)$	$\tilde{O}(k\lambda L)$	Multiple	No
Our Scheme	ring-LWE	$\tilde{O}(\lambda^2 L^6)$	$\tilde{O}(kL)$	Multiple	Yes

Table 1. Main Properties Comparisons. k denotes the actual number of parties involved in the evaluation, with a designed bound of K in [PS16]. L denotes the circuit depth that the scheme is designed to homomorphically evaluate.

Scheme	Assumption	Approximate Factor	Ciphertexts Extension	Evaluation Key Generation
[CM15]	LWE	$poly(K,\lambda)^L$	$t \cdot \tilde{O}(k\lambda^{4.37}L^{4.37})$	***
[CM15]	ring-LWE	$poly(K,\lambda)^L$	$t \cdot \tilde{O}(k\lambda L)$	***
[BP16]	LWE	$poly(K,\lambda)$	$\tilde{O}(k^2\lambda^4)$	$\tilde{O}(k\lambda^{4.37})$
[PS16] Scheme #1	LWE	$poly(K,\lambda,L)^{K+L}$	$< t \cdot \tilde{O}(k^2\lambda^4(K+L)^4)$	***
[PS16] Scheme #2	LWE/KDM	$poly(K,\lambda,L)^{K+L}$	$< t \cdot \tilde{O}(k^2\lambda^5(K+L)^4)$	***
Our Scheme	LWE	$poly(K,\lambda,L)^L$	$\tilde{O}(1)$	$\tilde{O}(k^{4.37}\lambda^{4.37}L^{7.37})$
Our Scheme	ring-LWE	$poly(K,\lambda,L)^L$	$\tilde{O}(1)$	$\tilde{O}(k^3\lambda^3 L^6)$

Table 2. Complexity of Party Extension. The meanings of the symbols are as same as Table 1. $t(\geq k)$ denotes the number of involved ciphertexts in an evaluation. The ciphertexts extension in [BP16] denotes the evluation of the circuit $\mathcal{C}(x,y) = \mathsf{NAND}(\mathsf{Dec}_x(c_1), \mathsf{Dec}_y(c_2))$, and the evaluation key generation is to generate the extended refresh key. The matrix multiplication is performed by the algorithm in [WV12], which has complexity of $O(n^{2.37})$ for n dimensional square matrices. It is hard to give an exact complexity for multiplication of rectangular matrices with the algorithm in [WV12], so we just provide the upper bound of the complexity by the naive algorithm.

Scheme	Assumption	Per Gate Complexity	Overhead
[CM15]	LWE	$\tilde{O}(k^{2.37}\lambda^{2.37}L^{2.37})$	$\tilde{O}(k^{2.37}\lambda^{2.37}L^{3.37})$
[CM15]	ring-LWE	$\tilde{O}(k^3\lambda L^2)$	$\tilde{O}(k^3\lambda L^2)$
[BP16]	LWE	$\tilde{O}(k^2\lambda^4)$	$\tilde{O}(k^2\lambda^4)$
[PS16] Scheme #1	LWE	$< \tilde{O}(k^2\lambda^5(K+L)^7)$	$< \tilde{O}(k^2\lambda^5(K+L)^7)$
[PS16] Scheme #2	LWE/KDM	$\tilde{O}(k^{2.37}\lambda^{2.37}(K+L)^{2.37})$	$\tilde{O}(k^{2.37}\lambda^{2.37}(K+L)^{2.37})$
Our Scheme	LWE	$\tilde{O}(k^3\lambda^3L^5)$	$\tilde{O}(k^3\lambda^3L^5)$
Our Scheme	ring-LWE	$\tilde{O}(k^2\lambda L^3)$	$\tilde{O}(k^2L^2)$

Table 3. Complexity of Evaluation. The meanings of the symbols are as same as Table 1 and Table 2. Also we just provide the complexity of the naive algorithm as the upper bound of rectangular matrix multiplication complexity.

5 Threshold Decryption and Two Round MPC

We now show how to implement a threshold decryption for the MKFHE construction presented in the previous section, hence a 2-round MPC protocol can be constructed according to the result of [MW16].

5.1 Definitions

Definition 7 ([MW16]). *A Threshold multi-key FHE scheme (TMKFHE) is a MKFHE scheme with two additional algorithms MFHE.PartDec, MFHE.FinDec described as follows:*

- $\rho_i \leftarrow$ *MFHE.PartDec*$(ct, (pk_1, \ldots, pk_K), i, sk_i)$*: On input an expanded ciphertext under a sequence of K keys and the i-th secret key, output a partial decryption ρ_i.*
- $\mu \leftarrow$ *MFHE.FinDec*$(\rho_1, \ldots, \rho_K)$*: On input K partial decryption, output the plaintext μ.*

Along with the properties of multi-key FHE we require the scheme to satisfy the following properties.

Correctness. The following holds with probability 1:

$$\mathsf{MKFHE.FinDec}(\rho_1, \ldots, \rho_N) = \mathcal{C}(\mu_1, \ldots, \mu_h)$$

where $\{\rho_i \leftarrow \mathsf{MKFHE.PartDec}(ct, (pk_1, \ldots, pk_K), i, sk_i)\}_{i\in[K]}$ are the partial decryptions and ct is the final output ciphertext by the evaluation algorithm for the circuit $\mathcal{C}$.

Simulatability. There exists a PPT simulator $\mathcal{S}^{thr}$ which, on input index $i \in [K]$, all but the i-th keys $\{sk_j\}_{j\in[K]/\{i\}}$, the evaluated ciphertext ct and the output message $\mu := \mathcal{C}(\mu_1, \ldots, \mu_h)$, produces a simulated partial decryption $\rho_i' \leftarrow \mathcal{S}^{thr}\left(\mu, ct, i, \{sk_j\}_{j\in[K]/\{i\}}\right)$ such that

$$\rho_i \approx \rho_i'$$

where $\rho_i \leftarrow \mathsf{MFHE.PartDec}\left(ct, (pk_1, \ldots, pk_N), i, sk_i\right)$. Note that the randomness is only over the random coins of the simulator and the $\mathsf{MFHE.PartDec}$ procedure, and all other values are assumed to be fixed (and known).

Theorem 2 ([MW16]). *Given any threshold multi-key fully homomorphic scheme defined as above, one can construct a two-round MPC protocol for any circuit which achieves honest-but-curious security in the CRS model. Additionally assuming the existence of NIZKs, then one can construct a two-round MPC protocol for any circuit which achieves fully malicious security in the UC framework in the CRS model.*

5.2 Construction

We now show how to implement a threshold decryption for the MKFHE construction presented in the previous section. Since Smudging Lemma 1 is involved to ensure the simulatability, we should choose the modulus q_L as large as $2^{O(K,\lambda,L)}$, which implies the approximate factor for the underlying problem to be exponentially large. Note that the same problem exists in [MW16] as well.

MKFHE.PartDec($\bar{\mathbf{c}}, (pk_1, \ldots, pk_k), i, sk_i$*):* On input an expanded ciphertext $\bar{\mathbf{c}} \in R_q^{2k}$ under a sequence of keys $(pk_1, \ldots, pk_k)$ and the ith secret key at level-l $\mathbf{s}_{l,i} \in R_q^2$, do the following:

– Parse $\bar{\mathbf{c}}$ as a concatenation of k sub-vectors $\mathbf{c}_i \in R_q^2$ such that $\bar{\mathbf{c}} = (\mathbf{c}_1 | \ldots | \mathbf{c}_k)$.
– Then compute $\gamma_i = \langle \mathbf{s}_i, \mathbf{c}_i \rangle \in R_q$ and output $\rho_i = \gamma_i + e_i^{sm} \in R_q$, where each coefficient of the random "smudging noise" e_i^{sm} is uniformly sampled from $[-B_{smdg}^{dec}, B_{smdg}^{dec}]$ for $B_{smdg}^{dec} = 2^{\lambda} B_{max}$ and $B_{max} = \tilde{O}(\lambda K)$.

MFHE:FinDec($p_1, \ldots, p_k$*)* : Given $\rho_1, \ldots, \rho_k$, compute the sum $\rho := \sum_{i=1}^{k} \rho_i$. Output $\mu := \rho \mod p$.

5.3 Correctness and Simulation Security

Theorem 3. *The above threshold decryption procedures for MKFHE satisfy the correctness and the (statistical) simulation security.*

Correctness. The entire scheme is the same as MKFHE except the decryption. If $\mathcal{C}$ is an evaluated ciphertext encrypting a bit μ and the secret keys are $\bar{\mathbf{s}}_l = (s_{l,1}, \ldots, s_{l,k})$, by the correctness analysis of the non-threshold MKFHE, we have

$$\langle \bar{\mathbf{s}}_l, \bar{\mathbf{c}} \rangle = \sum_{i \in [k]} \langle \mathbf{s}_{l,i}, \mathbf{c}_i \rangle = \mu + pe,$$

where $\|e\|_\infty \le K \cdot B_0$. Therefore, if the partial decryptions ρ_i are computed as above, we have

$$\begin{aligned} \sum_{i\in[k]} \rho_i &= \sum_{i\in[k]} \gamma_i + p \sum_{i\in[k]} e_i^{sm} \\ &= \sum_{i\in[k]} \langle \mathbf{s}_{l,i}, \mathbf{c}_i \rangle + pe^{sm} \\ &= \mu + pe + pe^{sm}, \end{aligned} \tag{14}$$

where $e^{sm} = \sum_{i\in[k]} e_i^{sm}$ has norm $\|e^{sm}\|_\infty \le K \cdot B_{smdg}^{dec} \le K \cdot 2^{O(\lambda)} B_{max}$ and e has norm $\|e\|_\infty \le B_{max}$. If we set $q_0 = 4K \cdot 2^\lambda B_{max}$, then $\|e_0 + e_{sm}\| < q/4$ and the correctness holds immediately.

Simulatability. The simulator $\mathcal{S}^{thr}\left(\mu, \hat{\mathbf{c}}, i, \{s_{l,j}\}_{j\in[k]/\{i\}}\right)$ takes as inputs the secrets keys $\{s_{l,j}\}_{j\in[k]/\{i\}}$, the evaluated ciphertext $\hat{\mathbf{c}} \in R_q^{2k}$ and the output value $\mu = \mathcal{C}(\mu_1, \ldots, \mu_k)$ encrypted in $\hat{\mathbf{c}}$. It outputs the simulated partial decryption as

$$\rho_i' = \mu + pe_i^{sm} - p \sum_{i\ne j} \gamma_i$$

for $e^{sm} \in [-B_{smdg}^{dec}, B_{smdg}^{dec}]$ where $\gamma_i = \langle \mathbf{s}_{l,i}, \mathbf{c}_i \rangle$. To see the indistinguishability, note that if $\rho_i = \gamma_i + e_i^{sm}$ is the real partial decryption then according to (14)

$$\rho_i = \mu + pe + pe_i^{sm} - p \sum_{i\ne j} \gamma_i.$$

The difference between the real value ρ_i and the simulated value ρ_i' is the noise e of norm $\|e\|_\infty \le B_{max}$. By Lemma 1, the distributions of e_i^{sm} and $e_i^{sm} + e$ are statistically close since each coefficient of e_i^{sm} is uniformly sampled from $[-B_{smdg}^{dec}, B_{smdg}^{dec}]$ where $B_{smdg}^{dec} = 2^\lambda B_{max}$, so that $B_{smdg}^{dec}/\|e\|_\infty \ge 2^\lambda$. Therefore, the simulated partial decryption and the real one are statistically indistinguishable.

6 Conclusion

In this paper, we show the multi-hop multi-key FHE can be achieved from the BGV scheme. Therefore, the scheme inherits the advantages of the BGV scheme, for example, it can encrypt a ring element as the plaintext and support the CRT-based packed ciphertexts technique. Moreover, the complexity of the ciphertext extension procedure in out scheme is dependent only on the number of involved secret keys but not on the number of ciphertexts.

Acknowledgement. We would like to thank Jiang Zhang and Qiang Tang for thoughtful discussions. Also we would like to thank the anonymous reviewers for their valuable comments. The work is supported by the National Key Research and Development Program of China (No.2017YFB0802005,2017YFB0802504) the National Natural Science Foundation of China (No.U1536205) and the National Basic Research Program of China (No.2013CB338003).

A The BGV Cryptosystem

In this section, we revisit the BGV scheme from [BGV12]. As explained in the introduction, our MKFHE is based on the BGV FHE scheme.

A.1 Modulus Switching

In the BGV LFHE scheme, since the noise term grows with homomorphic operations of the cryptosystem, switching modulus from q_{i+1} to q_i is used to decrease the noise term roughly by the ratio q_{i+1}/q_i.

– $\mathsf{ModulusSwitch}(\mathbf{c}, i)$: The operation takes a ciphertext $\mathbf{c} = (c_0, c_1)$ defined modulo q_i as input, and produces a ciphertext $\mathbf{c}' = (c'_0, c'_1)$ defined modulus q_{i-1}, such that $[c_0 - z \cdot c_1]_{q_i} \equiv [c'_0 - z \cdot c'_1]_{q_{i-1}} (\bmod p)$. Then change the level tag from i to $i-1$.

The Modulus Switching procedure makes use of the function $\mathsf{Scale}(x, q, q')$ that takes an element $x \in R_q$ as input and returns an element $y \in R_{q'}$ such that in coefficient representation it holds that $y \equiv x(\bmod p)$, and y is the closest element to $(q'/q) \cdot x$ that satisfies this mod-p condition for $p \ll q$. The details are available in [BGV12,GHS12c]. Once we have a level-0 ciphertext ct, we can no longer use modulus switching technique to reduce the noise. Then the bootstrapping technique is needed to regain a fresh cipher.

Lemma 6 ([BGV12,GHS12c]). *Let $q_i > q_{i-1} > p$ be positive integers satisfying $q_i = q_{i-1} = 1(\mod p)$. Let c,s be two ring elements over $R = \mathbb{Z}[X]/\Phi_m(X)$ such that*

$$\|c \cdot s\|^{can}_{q_i} < q_i/2 - \frac{q_i}{q_{i-1}} pn \cdot \phi(m) \|s\|^{can},$$

and let $c' = \mathit{Scale}(c, q_i, q_{i-1}, p)$. Denoting $e = cs \mod \Phi_m(X)$ and $e' = c's \mod \Phi_m(X)$ (arithmetic in $\mathbb{Z}[X] = \Phi_m(X)$), it holds that $e \bmod q_{i-1} \bmod p \equiv e' \bmod q_i \bmod p$ in coefficient representation, and

$$\|e'\|^{can}_{q_{i-1}} < \frac{q_{i-1}}{q_i} \cdot \|e\|^{can}_{q_i} + pn \cdot \phi(m) \cdot \|s\|^{can}.$$

A.2 Key Switching

After some homomorphic evaluation operations, we have on our hands not a "normal" ciphertext which is valid relative to a "normal" secret key, but an "extended ciphertext" which is valid with respect to an "extended secret key". Let $\beta = \lfloor \log q \rfloor + 1$. The key switching approach consists of two procedures, i.e.,

– $\mathsf{SwitchKeyGen}(\mathbf{s}_1 \in R^k_q, \mathbf{s}_2 = (1, -z_2)^T \in R^2_q)$: Compute $\overline{\mathbf{s}} = \mathsf{Powersof2}(\mathbf{s}_1) \in R^{k\beta}_q$, sample $k \cdot \beta$ ring-LWE instances $(a_i, a_i z_2 + pe_i), i = 1, \cdots, k\beta$, and output

$$\tau_{\mathbf{s}_1 \to \mathbf{s}_2} := \{\mathcal{K}_i = (a_i z_2 + pe_i + \overline{\mathbf{s}}[i], a_i) \in R^2_q\}_{i=1,\cdots,k\beta}.$$

– $\mathsf{SwitchKey}(\tau_{\mathbf{s}_1 \to \mathbf{s}_2}, \mathbf{c} \in R_q^k)$: Since $\bar{\mathbf{c}} = \mathsf{BitDecomp}(\mathbf{c})$, output

$$\mathbf{c}' = \sum_i \bar{\mathbf{c}}[i]\mathcal{K}_i$$

as the new ciphertext under the secret key $\mathbf{s}_2$. The correctness requires that $\langle \mathcal{K}_i, \mathbf{s}_2 \rangle = pe + \mathsf{Powersof2}(\mathbf{s}_1)[i]$ for a small norm e.

A.3 BGV LFHE Scheme

Following we list the basic algorithms of BGV schemes. See [BGV12,GHS12c] for details. Specifically, the BGV scheme is parameterized by a sequence of decreasing module $q_L \gg q_{L-1} \gg \cdots \gg q_0$, and an " level-$l$ ciphertext" in the scheme is $\mathbf{c} = (c^0, c^1) \in R_{q_l}^2$. Let $\beta_l = \lfloor \log q_l \rfloor + 1$ for $l = L, \ldots, 0$. After each homomorphic operation, modulus q_l at level-l is switched to q_{l-1} at level-$l-1$. Also, the corresponding secret key is switched.

BGV.KeyGen($1^\lambda, 1^L$) : Given the security parameter λ and L, choose the noise distribution $\chi = \chi(\lambda, L)$ which is a B-bounded distribution over R, L decreasing module $q_L \gg q_{L-1} \gg \cdots \gg q_0$ for each level, and a small integer p coprime with all q_l's. For l from L down to 0, do the following:

1. Choose a vector $z_l \leftarrow \chi$, and set $\mathbf{s}_l := (1, -z_l)^T \in R_{q_l}^2$.
2. Generate ring-LWE instances $pt_l := (b_l = a_l \cdot z_l + pe_l \bmod q_l, a_l) \in R_{q_l}^2$ for $a_l \in R_{q_l}$, set pt_l as the level-l public key relative to the secret key $\mathbf{s}_l$.
3. Set $\mathbf{s}'_l = \mathbf{s}_l \otimes \mathbf{s}_l \in R_{q_l}^4$, run $\tau_{\mathbf{s}'_l \to \mathbf{s}_{l-1}} \leftarrow \mathsf{SwitchKeyGen}(\mathbf{s}'_l, \mathbf{s}_{l-1})$ (omit this step when $l = 0$).

The public key is $pk = \{pt_l\}_{l \in \{L,\ldots,0\}}$, the evaluation key is $evk = \{\tau_{\mathbf{s}'_l \to \mathbf{s}_{l-1}}\}_{l \in [L]}$ and the secret key is $sk = \{\mathbf{s}_l\}_{l \in \{L,\ldots,0\}}$.

BGV.Enc(pk, μ) : To encrypt an element $\mu \in R_p$, choose two random elements $r, e \leftarrow \chi$ and output level-L ciphertext $\mathbf{c} = (c^0, c^1) \in R_{q_L}^2$ where

$$c^0 = rb_L + pe + \mu \in R_{q_L} \text{ and } c^1 = ra_L \in R_{q_L}.$$

BGV.Dec($sk, \mathbf{c}, l$): Given a level-l ciphertext $\mathbf{c} = (c^0, c^1) \in R_{q_l}^2$, compute

$$\mu = \langle \mathbf{c}_1, \mathbf{s}_l \rangle \bmod q_l \bmod p.$$

BGV.HomAdd($evk, \mathbf{c}_1, \mathbf{c}_2$): Take two ciphertexts $\mathbf{c}_1$ and $\mathbf{c}_2$ at the same level-l under the same $\mathbf{s}_l$ as inputs (If needed, use $\mathsf{SwitchKey}$ and $\mathsf{ModulusSwitch}$ to make it so). First, compute $\mathbf{c}_1 + \mathbf{c}_2 \bmod q_l$ and pad zeros to get $\mathbf{c}'_3 \in R_{q_l}^4$ under the key $\mathbf{s}'_l := \mathbf{s}_l \otimes \mathbf{s}_l$. Second, use $\mathsf{SwitchKey}(\tau_{\mathbf{s}'_l \to \mathbf{s}_{l-1}}, \mathbf{c}'_3)$ to generate a ciphertext $\bar{\mathbf{c}}_3$ under the secret key $\mathbf{s}_{l-1}$ ($\mathbf{s}'_l$'s coefficients include all of $\mathbf{s}_l$s since $\mathbf{s}'_l = \mathbf{s}_l \otimes \mathbf{s}_l$ and $\mathbf{s}_l$'s first coefficient is 1). Third, compute $\mathbf{c}_3 = \mathsf{ModulusSwitch}(\bar{\mathbf{c}}_3, l)$.

BGV.HomMult$(evk, \mathbf{c}_1, \mathbf{c}_2)$: Take two ciphertexts $\mathbf{c}_1$ and $\mathbf{c}_2$ at same level-l under the same $\mathbf{s}_l$ as inputs (If needed, use SwitchKey and ModulusSwitch to make it so). First, compute $\tilde{\mathbf{c}}_3 = \mathbf{c}_1 \otimes \mathbf{c}_2$ under the secret key $\mathbf{s}_l' = \mathbf{s}_l \otimes \mathbf{s}_l$. Second, use $\mathsf{SwitchKey}(\tilde{\mathbf{c}}_3, \tau_{\mathbf{s}_l' \to \mathbf{s}_{l-1}}, q_l)$ to generate a ciphertext $\mathbf{c}_3'$ under the secret key $\mathbf{s}_{l-1}$. Third, compute $\mathbf{c}_3 = \mathsf{ModulusSwitch}(\mathbf{c}_3', l)$.

A.4 Packing Ciphertexts

Let p be a prime integer, coprime to m, and R_p be the localisation of R at p. The polynomial $\Phi_m(X)$ factors modulo p into $k^{(R)}$ irreducible factors, i.e., $\Phi_m(X) \equiv \prod_{i=1}^{k^{(R)}} F_i(X) (\bmod\, p)$. Each $F_i(X)$ has degree $d^{(R)} = \phi(m)/k^{(R)}$, where $d^{(R)}$ is the multiplicative order of p in $\mathbb{Z}_m^*$. In the packed ciphertext scheme, each of these $k^{(R)}$ factors corresponds to a "plaintext slot", i.e.

$$R_p \cong \mathbb{Z}_p[X]/F_1(X) \times \cdots \times \mathbb{Z}_p[X]/F_{k^{(R)}}(X) \cong (\mathbb{F}_{p^{d^{(R)}}})^{k^{(R)}}.$$

More precisely, we have $k^{(R)} = |\mathbb{Z}_m^* / \langle p \rangle|$ isomorphisms

$$\psi_i : \mathbb{Z}_p[X]/F_i(X) \to \mathbb{F}_{p^{d^{(R)}}}, i = 1, \ldots, k^{(R)},$$

that allow to represent $k^{(R)}$ plaintext elements of $\mathbb{F}_{p^d}$ as a single element in R_p. By the Chinese Remainder Theorem, addition and multiplication correspond to the SIMD operations on the slots, which allows us to process $k^{(R)}$ input values at once.

Beyond addition and multiplications, we can also manipulate elements in R_p using a set of automorphisms on R_p of the form $a(X) \mapsto a(X^j)$, or in more detail

$$\rho_j : R_p \to R_p,\, a(X) + (p, \Phi_m(X)) \mapsto a\left(X^j\right) + (p, \Phi_m(X))\ (j \in \mathbb{Z}_m^*). \quad (15)$$

Actually, the Galois group $\mathcal{G}al(\mathbb{Q}[X]/\Phi_m(X))$ consists of all the transformations $X \mapsto X^i$ for $i \in \mathbb{Z}_m^*$, hence there are exactly $\phi(m)$ of them. Specifically, $\mathcal{G}al(Q[X]/\Phi_m(X))$ contains a subgroup $\mathcal{G} = \{(X \mapsto X^{p^i}) : j = 0, 1, \ldots, d-1\}$ corresponding to the Frobenius automorphisms modulo p. This subgroup does not permute the slots at all, but the quotient group $H = \mathcal{G}al/\mathcal{G}$ does. Clearly, $\mathcal{G}$ has order d and H has order $\phi(m)/d = k$. We can homomorphically evaluate these automorphisms by applying them to the batched BGV ciphertext elements and then preforming a "key switching". As discussed in [GHS12b], the combinations of automorphisms in H can induce any permutations on the plaintext slots.

Theorem 4 ([GHS12b]). *Let l,t,ω and W be parameters. Then any t-gate fan-in-2 arithmetic circuit $\mathcal{C}$ with average width ω and maximum width W, can be evaluated using a network of $O\left(\lceil t/l \rceil \cdot \lceil l/w \rceil \cdot \log W \cdot poly \log(l)\right)$ l-fold gates of types l-Add, l-Mult, and l-Permute. The depth of this network of l-fold gates is at most $O(\log W)$ times that of the original circuit $\mathcal{C}$, and the description of the network can be computed in time $\tilde{O}(t)$ given the description of $\mathcal{C}$.*

Using this theorem, Gentry et.al showed, as the batched BGV scheme with bootstrapping [BGV12], the total overhead is polylogarithmic in the security parameter.

References

[ABD16] Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In *Advances in Cryptology - CRYPTO 2016, Proceedings, Part I*, pages 153–178, 2016.

[AJL$^+$12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In *Advances in Cryptology - EUROCRYPT 2012*, pages 483–501, 2012.

[ASP13] Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasilinear time. In *Advances in Cryptology–CRYPTO 2013*, pages 1–20. Springer, 2013.

[ASP14] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In *Advances in Cryptology–CRYPTO 2014*, pages 297–314. Springer, 2014.

[BGH13] Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in LWE-based homomorphic encryption. In *Public-Key Cryptography–PKC 2013*, pages 1–13. Springer, 2013.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 309–325. ACM, 2012.

[BP16] Zvika Brakerski and Renen Perlman. Lattice-based fully dynamic multi-key FHE with short ciphertexts. In *Advances in Cryptology - CRYPTO 2016, Proceedings, Part I*, pages 190–213, 2016.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In *Advances in Cryptology–CRYPTO 2012*, pages 868–886. Springer, 2012.

[BV11a] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *Foundations of Computer Science Annual Symposium on*, 2011(2):97–106, 2011.

[BV11b] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Advances in Cryptology–CRYPTO 2011*, pages 505–524. Springer, 2011.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 1–12. ACM, 2014.

[CJL16] Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee. An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low-level encoding of zero. *LMS Journal of Computation and Mathematics*, 19(A):255–266, 001 2016.

[CM15] Michael Clear and Ciaran McGoldrick. Multi-identity and multi-key leveled FHE from learning with errors. In *Advances in Cryptology - CRYPTO 2015, Proceedings, Part II*, pages 630–656, 2015.

[CP16] Eric Crockett and Chris Peikert. $\Lambda o\lambda$: Functional lattice cryptography. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016*, pages 993–1005, 2016.

[DHS16] Yarkin Doröz, Yin Hu, and Berk Sunar. Homomorphic AES evaluation using the modified LTV scheme. *Des. Codes Cryptography*, 80(2):333–358, 2016.

[DM15] Léo Ducas and Daniele Micciancio. Fhew: Bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology–EUROCRYPT 2015*, pages 617–640. Springer, 2015.

[DPSZ12] Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology–CRYPTO 2012*, pages 643–662. Springer, 2012.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.

[GHPS13] Craig Gentry, Shai Halevi, Chris Peikert, and Nigel P Smart. Field switching in bgv-style homomorphic encryption. *Journal of Computer Security*, 21(5):663–684, 2013.

[GHS12a] Craig Gentry, Shai Halevi, and Nigel P Smart. Better bootstrapping in fully homomorphic encryption. In *Public Key Cryptography–PKC 2012*, pages 1–16. Springer, 2012.

[GHS12b] Craig Gentry, Shai Halevi, and Nigel P Smart. Fully homomorphic encryption with polylog overhead. In *Advances in Cryptology–EUROCRYPT 2012*, pages 465–482. Springer, 2012.

[GHS12c] Craig Gentry, Shai Halevi, and Nigel P Smart. Homomorphic evaluation of the AES circuit. In *Advances in Cryptology–CRYPTO 2012*, pages 850–867. Springer, 2012.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology–CRYPTO 2013*, pages 75–92. Springer, 2013.

[HAO15] Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. Packing messages and optimizing bootstrapping in GSW-FHE. In *Public-Key Cryptography–PKC 2015*, pages 699–715. Springer, 2015.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In *International Symposium on Algorithmic Number Theory*, pages 267–288, 1998.

[HS14] Shai Halevi and Victor Shoup. Algorithms in HElib. In *Advances in Cryptology - CRYPTO 2014, Proceedings, Part I*, pages 554–571, 2014.

[HS15] Shai Halevi and Victor Shoup. Bootstrapping for HElib. In *Advances in Cryptology–EUROCRYPT 2015*, pages 641–670. Springer, 2015.

[LATV12] Adriana López-Alt, Eran Tromer, and Vinod 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*, pages 1219–1234. ACM, 2012.

[Lin] Yehuda Lindell. Tutorials on the foundations of cryptography.

[LPR13a] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6):43, 2013.

[LPR13b] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. In *Advances in Cryptology–EUROCRYPT 2013*, pages 35–54. Springer, 2013.

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In *Advances in Cryptology - CRYPTO 2016 Proceedings, Part II*, pages 629–658, 2016.

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In *Advances in Cryptology - EUROCRYPT 2016 , Proceedings, Part II*, pages 735–763, 2016.

[PS16] Chris Peikert and Sina Shiehian. Multi-key FHE from lwe, revisited. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Proceedings, Part II*, pages 217–238, 2016.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[SV14] Nigel P Smart and Frederik Vercauteren. Fully homomorphic simd operations. *Designs, codes and cryptography*, 71(1):57–81, 2014.

[WV12] Williams and Virginia Vassilevska. Multiplying matrices faster than coppersmith-winograd. *Proceedings of the Annual Acm Symposium on Theory of Computing*, 129(8):887–898, 2012.