# On the Security of Multikey Homomorphic Encryption

Hyang-Sook Lee and Jeongeun Park

Department of Mathematics, Ewha Womans University, Republic of Korea
hsl@ewha.ac.kr, jungeun7430@ewhain.net

**Abstract.** Multikey fully homomorphic encryption (MFHE) scheme enables homomorphic computation on data encrypted under different keys. To decrypt a result ciphertext, all the involved secret keys are required. For multi decryptor setting, decryption is a protocol with minimal interaction among parties. However, all prior schemes supporting the protocol are not secure in public channel against a passive external adversary who can see any public information not joining the protocol. Furthermore, the possible adversaries have not been defined clearly.
In this paper, we revisit the security of MFHE and present a secure one-round decryption protocol. We apply it to one of existing schemes and prove the scheme is secure against possible static adversaries. As an application, we construct a two round multiparty computation without common random string.

**Keywords:** security of MFHE · MPC without CRS · Multikey homomorphic encryption.

## 1 Introduction

### 1.1 Multikey fully Homomorphic Encryption schemes

Fully homomorphic encryption (FHE) supports arbitrary computation on encrypted data under the same key. Multikey fully homomorphic encryption (MFHE) is a generalization of FHE, which allows arbitrary computation on encrypted data under *different* keys. The important thing is that all relevant secret keys are required to decrypt a ciphertext. This concept was first proposed by Lopez, Tromer, and Vaikuntanathan [17] in 2012, which is intended to apply to on-the-fly multiparty computation. In fact, MFHE has been an interesting topic for round efficient secure computation with minimal communication cost [5,9,16,18]. There are several results on MFHE [4,9,14,15,19] based on LWE problem, all of which do not allow any interaction among associated parties before decryption protocol is started. All their schemes assume a common random string (CRS) model, which additionally requires a trusted party who distributes the CRS to every party and can be viewed as an ideal version of multikey homomorphic encryption. A CRS plays a role of linking all the parties' ciphertexts under different keys to do correct computation on them. Kim, Lee and Park introduce another scheme to get rid of a role of CRS, which is defined in [10] and also [16] implies the same scheme. In their scheme, the parties share their public keys after key generation step to relate their keys for their own sake like threshold fully homomorphic encryption scheme [1], but keys are generated independently by each user. On the other hand, each user generates key pair and encrypts its own message using single key encryption then publishes together at once without any interaction in the previous other schemes. Their scheme is for the fixed number of users so that it might loose some dynamic property which lets users join and leave the computation freely.

So far, there are at least two types of MFHE schemes: (1) non-interactive MFHE which does not allow any interaction among parties before decryption (note that the computation is done by a server), (2) interactive MFHE to share public keys after key generation. We call a non-interactive scheme and an interactive scheme to distinguish the two schemes in this paper. The interactive scheme might loose a round efficiency for applying to multiparty computation (MPC) since there is at least one interaction by default. On the other hand, it is possible to remove CRS. The interactive schemes are defined in [3] and [10,16] as designated multikey homomorphic encryption. However, regardless of schemes, we can think of that the decryption procedure can be divided into two cases. The first case is a single decryptor setting like in [17], where a trusted decryptor exists and holds all parties' secret keys so that no interaction among users is required. Here, only the decryptor gets the output value using all the secret keys. Then, each user just needs to keep its input privacy against one another. The second one is a multi decryptor setting, where all users jointly decrypt a common message with minimal interaction among them. Here, they have to make sure that any information about the function value should not be revealed until the joint decryption protocol is completed. Therefore, Kim, Lee and Park define a new security notion for MFHE, which is called multikey IND-CPA security, for the first time. This is a security for a multikey ciphertext (associated with different keys) against one of users whose key is involved in the ciphertext. It is a reasonable security since the ciphertext may leak some information before decrypted with all relevant secret keys. However, it is not sufficient for MFHE itself. All the existing MFHE schemes for multi decryptor setting employ a decryption protocol called distributed decryption, which is also widely used in threshold FHE [1,2]. More precisely, the distributed decryption protocol consists of two steps:

– **Partial Decryption**: Each user decrypts a common ciphertext partially with its own secret key. It outputs a partial decryption.
– **Final Decryption**: Given all users' partial decryptions, it outputs the plaintext(or evaluated plaintext).

In the protocol, the final decryption algorithm takes only partial decryptions without any key. In other words, the partial decryption is not a ciphertext any more. The final message is decrypted only adding all the partial decryptions. Therefore, anyone who is not joining the protocol but can get all partial decryptions from a transmission channel easily gets the evaluated message by running the final decryption algorithm *for free*. Indeed, this would cause a big problem in industry. For example, there are three companies A, B and C, all of which want to construct the best machine learning model using their clients' information as input. For clients' privacy, each company encrypts each input data with its own key. Then the three companies jointly compute a function to output the best machine learning model via an MFHE scheme with multi decryptor. There is a rival company D which does not join this computation, but also needs a good machine learning model for the same type of data as A, B and C have. If A, B and C use one of existing MFHE schemes, D can learn partial decryption shares from a transmission channel and just run the public algorithm(final decryption) to get the best machine learning model for free. If the above three companies handle a private technique deserving patents, this can be a more critical issue. As such, an MFHE scheme with the above decryption protocol is not secure in a public channel. Up to our best knowledge, all the existing schemes have not considered this situation since most of schemes are constructed for MPC hence they can assume a private channel. In other words, MFHE schemes in public channel have not been studied yet. In fact, it is worth consideration for MFHE itself since "encryption" must protect a plaintext from at least a static adversary in any public channel.

Likewise, an adversarial model and what should be protected from that adversaries are different depending on the decryptor setting in MFHE. However, the existing security definition of MFHE is somewhat ambiguous and not clear enough. Most of previous schemes just check their correctness and apply the IND-CPA security of their base single key homomorphic encryption scheme, even they do not clearly mention which decryptor setting they assume. That is, no particular security definition for MFHE itself has been fully discussed yet.

## 1.2 Our contribution

In this paper, we resolve the above problems as a main result. We revisit the security of multikey homomorphic encryption scheme and construct a secure decryption protocol based on an existing multikey homomorphic encryption scheme [14] based on TFHE [6–8]. To do this, we define a possible static adversary and semantic security for MFHE. Then we prove that a MFHE scheme with our protocol is semantically secure. Our idea is that a partial decryption remains a multikey ciphertext still encrypted under the other users' keys even if it is partially decrypted by a user. Therefore, it is still secure against an adversary not holding any key.

As an additional result, we obtain a round optimal multiparty computation protocol without CRS. To do this, we convert the non-interactive MFHE with CRS to an interactive MFHE *without CRS* combining two MFHE schemes to get a hybrid scheme. In the hybrid scheme, we use the original (leveled) MFHE for an encryption of message, and we use the converted interactive scheme for bootstrapping part (encryption of secret key). As a result, we construct a 2 round multiparty computation without CRS via the hybrid scheme. We show that it is possible to construct two round MPC protocol via MFHE *without CRS*, but with limited operation only in the first computation. Any MPC construction from MFHE without CRS [3,10] has not been achieved two round even against honest but curious adversaries. At least three rounds are required for even limited operation. Therefore, we contribute the first construction of optimal round of MPC from MFHE without CRS.

### 1.3 Organization

We review some important notions and pre-results in Section 2 and introduce a possible adversary and define the semantic security of MFHE in Section 3. In Section 4, we present our distributed decryption protocol and apply it to a multikey TFHE scheme. As an application of MFHE, we first modify a multikey TFHE to remove CRS, allowing interaction among parties, then combine the two multikey TFHE schemes to construct a 2 round MPC without CRS in Section 5.

## 2 Preliminaries

**Notation:** We denote $\lambda$ as the security parameter. We define vectors and matrices in lowercase bold and uppercase bold, respectively. Dot product of two vectors $\mathbf{v}, \mathbf{w}$ is denoted by $<\mathbf{v}, \mathbf{w}>$. For a vector $\mathbf{x}$, $\mathbf{x}[i]$ denotes the $i$-th component scalar. We denote that $\mathbb{B}$ as the set $\{0, 1\}$ and $\mathbb{T}$ as the real torus $\mathbb{R}/\mathbb{Z}$, the set of real number modulo 1. We denote $\mathbb{Z}_N[X]$ and $\mathbb{T}_N[X]$ by $\mathbb{Z}[X]/(X^N + 1)$ and $\mathbb{R}[X]/(X^N + 1)$ mod 1, respectively. $\mathbb{B}_N[X]$ denotes the polynomials in $\mathbb{Z}_N[X]$ with binary coefficients. For a real $\alpha > 0$, $D_\alpha$ denotes the Gaussian distribution of standard deviation $\alpha$. In this paper, we use the same notation with [7] for better understanding.

### 2.1 TFHE scheme

We describe our base FHE scheme TFHE [7] and its multikey version [14]. The multikey version of TFHE has smaller parameter and ciphertext size, leading to better performance than previous GSW [13]-based multikey schemes [4,5,9,18,19]. The TFHE scheme [7] is working entirely on real torus $\mathbb{T}$ and $\mathbb{T}_N[X]$ based on $\mathsf{TLWE}$ problem and $\mathsf{TRLWE}$ problem which are torus variant of $\mathsf{LWE}$ problem and $\mathsf{RLWE}$ problem respectively, where $N$ is a power of two. It is easy to see that $(\mathbb{T}, +, \cdot)$(resp. $(\mathbb{T}_N[X], +, \cdot)$) is $\mathbb{Z}$(resp. $\mathbb{Z}_N[X]$) module.

A $\mathsf{TLWE}$ (resp. $\mathsf{TRLWE}$)sample is defined as $(\mathbf{a}, b) \in \mathbb{T}^{kn+1}$ (resp. $\mathbb{T}_N[X]^{k+1}$) for any $k > 0$, where $\mathbf{a}$ is chosen uniformly over $\mathbb{T}^{kn}$(resp. $\mathbb{T}_N[X]^k$) and $b =< \mathbf{a}, \mathbf{s} > +e$. The vector $\mathbf{s}$ is a secret key which is chosen uniformly from $\mathbb{B}^{kn}$(resp. $\mathbb{B}_N[X]^k$) and the error $e$ is chosen from Gaussian distribution with standard deviation $\alpha \in \mathbb{R} > 0$. Furthermore, we follow the [7]'s definition of trivial sample as having $\mathbf{a} = \mathbf{0}$ and noiseless sample as having the standard deviation $\alpha$=0. Here, we denote the message space to $\mathcal{M} \subseteq \mathbb{T}$. A $\mathsf{TLWE}$ ciphertext of $\mu \in \mathcal{M}$ is constructed by adding a trivial noiseless $\mathsf{TLWE}$ message sample $(0,\dots,\mu) \in \mathbb{T}^{kn+1}$ to a non-trivial $\mathsf{TLWE}$ sample. Therefore, the $\mathsf{TLWE}$ ciphertext of $\mu$, say $\mathfrak{c}$, which we will interpret as a $\mathsf{TLWE}$ sample (of $\mu$) is $(\mathbf{a}, b) \in \mathbb{T}^{k+1}$, where $b = <\mathbf{a}, \mathbf{s}> + e + \mu$. To decrypt it correctly, we use a linear function $\varphi_{\mathbf{s}}$ called *phase*, which results in $\varphi_{\mathbf{s}}(\mathfrak{c}) = b - <\mathbf{a}, \mathbf{s}> = \mu + e$ and we round it to the nearest element in $\mathcal{M}$. We denote the error as $\mathsf{Err}(\mathfrak{c})$, which is equal to $\varphi_s(\mathfrak{c}) - \mu$. For a $\mathsf{TRLWE}$ encryption, it follows the same way over $\mathbb{T}_N[X]$ but a message $\mu$ is a polynomial of degree $N$ with coefficients $\in \mathcal{M}$.

From the above definition, we can define the decisional $\mathsf{TLWE}$(resp. $\mathsf{TRLWE}$) problem which is parametrized by an error distribution on $\mathbb{T}_N[X]$ and a function $\varphi_s$.

- Decision Problem: distinguish the uniform distribution on $\mathbb{T}^{kn+1}$ (resp. $\mathbb{T}_N^{k+1}$) from $\mathsf{TLWE}$ (resp.$\mathsf{TRLWE}$) samples for a fixed $\mathsf{TLWE}$ (resp. $\mathsf{TRLWE}$) secret $\mathbf{s}$.

The $\mathsf{TLWE}$ (resp. $\mathsf{TRLWE}$) problem is a generalization of LWE (resp. RLWE) problem which is as hard as approximating the shortest vector problem.

## 2.2 TGSW and an external product

As we can see, $\mathsf{TLWE}$ and $\mathsf{TRLWE}$ samples have additive homomorphic property. In order to have FHE scheme, [7] defined $\mathsf{TGSW}$ ciphertext which supports external product with $\mathsf{TLWE}$ ciphertext to get a $\mathsf{TLWE}$ ciphertext encrypting multiplication of messages. For $\mathsf{TGSW}$ samples in the ring mode, we use the notation $\mathsf{TRGSW}$ which is working as $\mathsf{TRLWE}$ and also give the definition of a $\mathsf{TRGSW}$ sample only. (for $N = 1$, we can think of $\mathsf{TGSW}$ sample).

For any positive integer $B_g(\geq 2), \ell, k$, a $\mathsf{TRGSW}$ sample is a matrix $\mathbf{C} = \mathbf{Z} + \mu \cdot \mathbf{H} \in \mathbb{T}_N[X]^{(k+1)\ell \times (k+1)}$, where each row of $\mathbf{Z}$ is a $\mathsf{TRLWE}$ sample of zero and $\mathbf{H}$ is a gadget matrix which is defined by

$$\mathbf{H} := \left[\begin{array}{c|c|c} 1/B_g & \dots & 0 \\ \vdots & \ddots & \vdots \\ 1/B_g^\ell & \dots & 0 \\ \hline \vdots & \ddots & \vdots \\ \hline 0 & \dots & 1/B_g \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1/B_g^\ell \end{array}\right] \in \mathbb{T}_N[X]^{(k+1)\ell \times (k+1)}.$$

i.e. $\mathbf{H} = \mathbf{I}_{k+1} \otimes \mathbf{g}$, where $\mathbf{g} = (1/B_g, \dots, 1/B_g^\ell)$. There is a decomposition algorithm $\mathbf{g}^{-1}(\cdot)$, which outputs the $\ell$-dimensional vector in $\mathbb{Z} \cap (-B_g/2, B_g/2]$, satisfying $< \mathbf{g}^{-1}(a), \mathbf{g} > \approx a$ for $a \in \mathbb{R}$. The message $\mu$ is in $\mathbb{Z}_N[X]$. We denote $\mathsf{TLWE}(\mu), \mathsf{TRLWE}(\mu)$, and $\mathsf{TRGSW}(\mu)$ as a ciphertext of each proper message $\mu$ of $\mathsf{TLWE}, \mathsf{TRLWE}$, and $\mathsf{TRGSW}$, respectively. We also denote a trivial $\mathsf{TRGSW}(1)$(resp. $\mathsf{TGSW}(1)$) as $\mathbf{Z}_t + \mathbf{H}$, where each row of $\mathbf{Z}_t$ is a trivial $\mathsf{TRLWE}$(resp. $\mathsf{TLWE}$) sample. An external product between a $\mathsf{TGSW}$ ciphertext and a $\mathsf{TLWE}$ ciphertext, denoted as $\boxdot$, is defined as $\mathbf{A} \boxdot \mathbf{b} = \mathbf{H}^{-1}(\mathbf{b}) \cdot \mathbf{A}$, where $\mathbf{A}$ is a $\mathsf{TGSW}$ sample of $\mu_A$, $\mathbf{b}$ is a $\mathsf{TLWE}$ sample of $\mu_b$ and $\mathbf{H}^{-1}(\cdot)$ is the gadget decomposition function $Dec_{\mathbf{H},\beta,\epsilon}$ of [7] with different notation. Then the output of the product is a $\mathsf{TLWE}(\mu_A \cdot \mu_b)$. We denote the error as $\mathsf{Err}(\mathbf{A})$, which is a list of the $(k+1)\ell$ $\mathsf{TLWE}$ errors of each line of $\mathbf{A}$. Then the error growth of the external product between $\mathbf{A}$ and $\mathbf{b}$ is following:

$$\|\mathsf{Err}(\mathbf{A} \boxdot \mathbf{b})\|_\infty \le (k+1)\ell N\beta \|\mathsf{Err}(\mathbf{A})\|_\infty + \|\mu_A\|_1 (1+kN)\epsilon + \|\mu_A\|_1 \|\mathsf{Err}(\mathbf{b})\|_\infty$$

## 2.3 Multikey version of TFHE

We only describe the *leveled* fully homomorphic mode of Chen, Chillotti and Song's scheme since we focus on decryption algorithm so that we refer [14] for more detail. Their scheme assumes a common random string (CRS) among users and the CRS is used for generating a public key and evaluation keys so we do not care of it in detail here. Their scheme is non-interactive before decryption.

- $\mathsf{MTFHE1.Setup}(1^\lambda)$: It takes security parameter and outputs $\mathsf{TLWE}$ parameter $\mathsf{params}$ which consists of $\mathsf{TLWE}$ dimension $n$, key distribution $\chi$, error parameter $\alpha$, and evaluation parameters $\mathsf{evparam}$.
- $\mathsf{MTFHE1.KeyGen}(\mathsf{params})$:
  - Sample $\mathbf{s} = (s_0, \ldots, s_{n-1}) \leftarrow \chi$ and set it as a $\mathsf{TLWE}$ secret key $\mathsf{sk}$.
  - construct a public key $\mathsf{pk}$, evaluation keys $\mathsf{evk}$ from proper algorithms in [14] with $\mathsf{params}$ and $\mathsf{evparam}$.
- $\mathsf{MTFHE1.Enc}(m, \mathbf{s})$: Sample $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1})$ from $\mathbb{T}^n$ uniformly at random. Then take a message bit $m \in \{0,1\}$ and construct a $\mathsf{TLWE}$ sample $(\mathbf{a}, b) \in \mathbb{T}^{n+1}$, where $b = \frac{1}{4}m - < \mathbf{a}, \mathbf{s} > +e \pmod 1$, $e$ is chosen from the Gaussian distribution $D_\alpha$. Returns a ciphertext $\mathsf{ct} = (\mathbf{a}, b)$.
- $\mathsf{MTFHE1.Dec}(\hat{\mathsf{ct}}, \{sk_j\}_{j\in[k]})$: Taking a (evaluated)ciphertext $\hat{\mathsf{ct}} = (\mathbf{a}_1, \ldots, \mathbf{a}_k, b) \in \mathbb{T}^{kn+1}$ and a concatenation of secret key vectors $(\mathbf{s}_1, \ldots, \mathbf{s}_k, 1) \in \mathbb{T}^{kn+1}$ as input and return the message bit $m \in \{0,1\}$, which satisfies $b + \sum_{j=1}^{k} < \mathbf{a}_j, \mathbf{s}_j > \approx \frac{1}{4}m \pmod 1$.
- $\mathsf{MTFHE1.Eval}(\hat{\mathsf{ct}}_1, \hat{\mathsf{ct}}_2)$: It takes two ciphertexts $\hat{\mathsf{ct}}_1 \in \mathbb{T}^{k_1 n+1}, \hat{\mathsf{ct}}_2 \in \mathbb{T}^{k_2 n+1}$, where $k_1$, $k_2$ are the number of parties who joined the previous evaluations to construct $\hat{\mathsf{ct}}_1, \hat{\mathsf{ct}}_2$ ,respectively, and the set $[k]$ is the indices of parties who are associated either $\hat{\mathsf{ct}}_1$ or $\hat{\mathsf{ct}}_2$. (we only consider $\mathsf{NAND}$ gate here.)
  - Extend $\hat{\mathsf{ct}}_1$ and $\hat{\mathsf{ct}}_2$ to make them the same dimensional vectors $\hat{\mathsf{ct}}_1', \hat{\mathsf{ct}}_2'$ $\in \mathbb{T}^{kn+1}$ encrypted under the concatenated secret key $\hat{\mathbf{s}} = (\mathbf{s}_1, \ldots, \mathbf{s}_k) \in \mathbb{Z}^{kn}$. Rearrange $\mathbf{a}_j$s giving each index to each user and putting zero in the empty slots, for $j \in [k]$.
  - Return an evaluated ciphertext $\hat{\mathsf{ct}}' = (\mathbf{0}, \ldots, \mathbf{0}, \frac{5}{8}) - \hat{\mathsf{ct}}_1' - \hat{\mathsf{ct}}_2' \pmod 1$.

  For the bootstrapping part with $\{\mathsf{evk}_j\}_{j\in[k]}$, we do not consider here, so we refer the original paper. We call an evaluated ciphertext multikey ciphertext in this paper. The dimension of a multikey ciphertext increases as a number of homomorphic evaluation increases.

### 2.4 Distributed decryption

A multikey homomorphic encryption scheme for multi decryptor setting includes a decryption protocol in which all users jointly decrypt a common evaluated message. The most round efficient distributed decryption [18] has been widely adopted to recent schemes, which is following:

– $\mathsf{PartDec}(\hat{\mathsf{ct}}, \mathsf{sk}_i)$ On inputs a multikey ciphertext $\hat{\mathsf{ct}}$ under a sequence of $k$ users' keys and the $i$-th secret key $\mathsf{sk}_i$, outputs a partial decryption $p_i$.
– $\mathsf{FinDec}(p_1, \ldots, p_k)$ Given all parties' partial decryptions $\{p_i\}_{i \in [k]}$, outputs a plaintext(or evaluated plaintext).

The decryption algorithm of $\mathsf{MTFHE1}$ is defined for a single decryptor case. But, they also suggest a distributed decryption protocol following the above for multi decryptor case, hence it is not secure in a public channel.

## 3 Multikey fully homomorphic encryption security

Before we define the semantic security of MFHE, we observe possible static(passive) adversaries first. The goal of MFHE for multi decryptor is to protect each user's individual message and the common evaluated message. Then we can see that there are at least two types of static adversaries, one of which is an internal adversary and the other is an external adversary. The internal adversary is one of participants of computation but an external adversary is not. Both can just see messages transmitted over any public channel but hope to learn any information about each user's message. However, the external adversary wants to learn the evaluated message as well. Therefore, a MFHE with multi decryptor has to consider a security for multikey ciphertext against both adversaries and partial decryption against an external adversary. Thanks to the multikey IND-CPA security [10], the multikey ciphertext is guaranteed to be secure against both adversaries. Even if they only care of one of joint users, it is obvious that if a multikey ciphertext is secure against one of secret key owners, it is secure against one not holding any key. So we call it the internal security (of MFHE) in this paper. Then we now define the external security (of MFHE).

For a probabilistic multikey fully homomorphic encryption algorithm, we naturally extend the original indistinguishability under chosen plaintext attack (IND-CPA) to any multikey FHE scheme by the following game between a PPT static external adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. For any multikey FHE encryption scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$, any static external adversary $\mathcal{A}$, and any value $\lambda$ for the security parameter, where $\mathsf{Dec} = (\mathsf{MFHE.PartDec}, \mathsf{MFHE.FinDec})$ is a distributed decryption protocol, MFHE security game is defined as:

1 $\mathcal{A}$ chooses a positive integer $k$ and gives it to the challenger $\mathcal{C}$.
2 $\mathcal{C}$ runs $\mathsf{KeyGen}(1^\lambda)$ to generate $k$ random key pairs $\{(\mathsf{sk}_i, \mathsf{pk}_i)\}_{i \in [k]}$ and $k$ evaluation keys $\{\mathsf{evk}_i\}_{i \in [k]}$. Then it publishes all the public keys $\{\mathsf{pk}_i\}_{i \in [k]}$ and $\{\mathsf{evk}_i\}_{i \in [k]}$ to $\mathcal{A}$ and keeps all the secret keys $\{\mathsf{sk}_i\}_{i \in [k]}$ in secret.

3 The adversary $\mathcal{A}$ is given input $1^\lambda$ and oracle access to $\mathsf{Enc}()$ with all public keys and its chosen messages. Then it chooses an index $j \in [k]$ and outputs a pair of message vectors $\mathbf{m_0}, \mathbf{m_1}$ of the same length and a function $f$ (here, each component of a vector is viewed as each user's message). The message vectors are $\mathbf{m_0} = (0, 0, \dots, 0) \in \{0,1\}^k$ and $\mathbf{m_1} = (1, 0, \dots, 0) \in \{0,1\}^k$ and a funtion $f$ is defined as $f : \{0,1\}^k \to \{0,1\}$ which outputs the first component of input vector. Then it gives $\mathbf{m_0}, \mathbf{m_1}$ , $f$, and the index $j$ to $\mathcal{C}$.
4 $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$, computes a multikey ciphertext which is an encryption of $f(\mathbf{m_b})$ under $k$ public keys, running $\mathsf{Eval}$. Then it partially decrypts the ciphertexts using $\{\mathsf{sk}_i\}_{i\in[k]\setminus\{j\}}$. It sends the partial decryption messages to $\mathcal{A}$.
5 The adversary is free to perform any number of additional computations, encryptions by given keys (free access to $\mathsf{Enc}, \mathsf{Eval}$). Finally, it outputs a guess for the value of $b'$. If $b' = b$, $\mathcal{A}$ wins.

We define that for any multikey homomorphic encryption scheme, if the advantage of $\mathcal{A}$ is negligible, then the scheme achieves the external security. As a result, we define semantic security for a MFHE.

**Definition 1.** *For a multikey homomorphic encryption scheme(*$\mathsf{KeyGen}$*,* $\mathsf{Enc}$*,* $\mathsf{Eval}$*,* $\mathsf{Dec}$*), where* $\mathsf{Dec}$ *is run by a single decryptor, it is semantically secure if it achieves the internal security. For a multikey homomorphic encryption scheme for multi decryptor, where* $\mathsf{Dec}$ *is a protocol among users, it is semantically secure if it achieves both internal and external security.*

It is possible for a single decryptor who holds every associated secret keys to decrypt a multikey ciphertext by itself. Therefore, achieving the only internal security is enough for its semantic security.

## 4 Distributed decryption for only joint users

### 4.1 Distributed decryption protocol

We first formalize the distributed decryption protocol with general algorithms for multikey homomorphic encryption scheme and then construct a specific protocol applying TFHE scheme based on LWE problem. Such a protocol consists of two steps: (1) each user first decrypts a common evaluated ciphertext partially with its secret key and broadcast the partial information, (2) after gathering all the partial decryption from all users, each user decrypts the correct evaluated message finally with its *secret key*, independently. Let $k$ be the number of users.

**Definition 2.** *A distributed decryption for multikey homomorphic encryption consists of two algorithms:*

– $\mathsf{MFHE}.\mathsf{PartDec}(\hat{\mathsf{ct}}, \mathsf{sk}_i)$*: It takes a common evaluated ciphertext* $\hat{\mathsf{ct}}$ *and* $i$*-th user's secret key* $\mathsf{sk}_i$ *for* $i \in [k]$ *on input. It returns partial decryptions* $p_{i,j}$ *for* $j \in [k]$*. The user keeps* $p_{i,i}$ *secret and broadcasts* $p_{i,j}$ *for* $j \in [k]\setminus\{i\}$*.*

– $\mathsf{MFHE}.\mathsf{FinDec}(\{p_{j,i}\}_{j\in[k]}, \mathsf{sk}_i)$*: It takes all the partial decrypted messages* $p_{j,i}$ *for* $j \in [k]\backslash\{i\}$ *which are given to the* $i$*-th user and its own partial decryption message* $p_{i,i}$ *and its secret key* $\mathsf{sk}_i$ *on input. It outputs the correct evaluated message.*

Comparing to previous protocol, the number of partial decryption increases linearly on $k$ while the previous one is constant. However, it seems inevitable for keeping privacy. If a scheme is interactive before decryption protocol, it is easy to think of an encryption of a partial decrypted message with other users' public keys since users share their public keys before the computation. Hence the $\mathsf{MFHE}.\mathsf{PartDec}$ might have another input $\mathsf{pk}_j$ for $j \in [k]\backslash\{i\}$. For a non-interactive scheme, server can give all required inputs for distributed decryption such as joint users' public keys and evaluation keys for users' own sake, however, it would not be straightforward to agree on sharing keys among users. Therefore, we introduce a naive protocol preserving the optimal round without having other users' information.

## 4.2 Specific protocol with multikey TFHE scheme

The multikey ciphertext of [14] scheme is $\hat{\mathsf{ct}} = (\mathbf{a}_1, \dots, \mathbf{a}_k, b) \in \mathbb{T}^{kn+1}$, which satisfies $b = \frac{1}{4}m - \sum_{j=1}^{k} < \mathbf{a}_j, \mathbf{s}_j > +e \pmod 1$, where $k$ is the number of joint users, $m \in \{0, 1\}$. Then the distributed decryption protocol (of the i-th user) is following:

– $\mathsf{MTFHE1}.\mathsf{PartDec}(\hat{\mathsf{ct}}, \mathbf{s}_i)$:
  - compute $p_{i,i} = b+ < \mathbf{a}_i, \mathbf{s}_i >= \frac{1}{4}m - \sum_{j\neq i}^{k} < \mathbf{a}_j, \mathbf{s}_j > +e \pmod 1 \in \mathbb{T}$
  - $p_{i,i}$ can be viewed as a one component of $\mathsf{TLWE}$ ciphertext of j-th user for $j \in [k]\backslash\{i\}$, i.e $p_{i,i} =< \mathbf{a}_j, \mathbf{s}_j > + \mathsf{mess} + e$, where $\mathsf{mess} = \frac{1}{4}m - \sum_{t\neq i,j}^{k} < \mathbf{a}_t, \mathbf{s}_t >$ for $t \in [k]\backslash\{i, j\}$. Then the user does external product between a $\mathsf{TLWE}$ sample $(\mathbf{a}_j, p_{i,i})$ and a trivial $\mathsf{TGSW}(1)$ which is denoted as $\mathbf{A}_j$ with noise $e_j$ from $D_\alpha$ by the user $i$.
  - The output of the external product is a $\mathsf{TLWE}$ ciphertext of the same message $\mathsf{mess}$, which is $p_{i,j} = (\mathbf{a}_{i,j}, b_{i,j}) \in \mathbb{T}^{n+1}$ and is given to each user $j$ for $j \in [k]\backslash i$.

– $\mathsf{MTFHE1}.\mathsf{FinDec}(\{p_{j,i}\}_{j\in[k]}, \mathbf{s}_i)$:
  - It parses $p_{j,i}$ into $\mathbf{a}_{j,i}$ and $b_{j,i}$ and compute $b'_{j,i} = b_{j,i}+ < \mathbf{a}_i, \mathbf{s}_i >$ for every $j \in [k]\backslash\{i\}$.
  - It computes $\sum_{j\neq i}^{k} b'_{j,i} - (k - 2)p_{i,i} = \frac{1}{4}m + \bar{e}$.
  - if the output is close to $\frac{1}{4}$ the evaluate message 1, otherwise 0.

Note that for $k = 2$ (two users), $b'_{j,i} = b_{j,i}+ < \mathbf{a}_i, \mathbf{s}_i >$ itself gives the result for $i \neq j \in [2]$.

**Correctness of decryption** Fix a user $i$ for $i \in [k]$. Then the correctness for the user follows:

$$\begin{aligned}
&\sum_{j\neq i}^{k} b'_{j,i} - (k-2)p_{i,i} \\
&= \sum_{j\neq i}^{k} (\frac{1}{4}m - \sum_{t\neq i,j} < \mathbf{a}_t, \mathbf{s}_t > + \tilde{e}_j) - (k-2)p_{i,i} \\
&= (k-1)\frac{1}{4}m - \sum_{j\neq i}^{k} (\sum_{t\neq i,j}^{k} < \mathbf{a}_t, \mathbf{s}_t > + \tilde{e}_j) - (k-2)(\frac{1}{4}m - \sum_{j\neq i}^{k} < \mathbf{a}_j, \mathbf{s}_j > + e) \\
&= \frac{1}{4}m - (k-2)\sum_{j\neq i}^{k} < \mathbf{a}_j, \mathbf{s}_j > + (k-2)\sum_{j\neq i}^{k} < \mathbf{a}_j, \mathbf{s}_j > + \sum_{j\neq i}^{k} \tilde{e}_j - (k-2)e \\
&= \frac{1}{4}m + \bar{e}
\end{aligned}$$

If the magnitude of the error term $\bar{e}$ is less than $\frac{1}{8}$, the decryption works correctly.

**Error growth estimation** Note that $e$ is $\mathsf{Err}(b)$ and also $\mathsf{Err}(p_{i,i})$. After the external product, $e$ becomes $\tilde{e_j}$ for each $j \in [k]\backslash\{i\}$ then we can say $\tilde{e}_j = e + e_{add_j}$. We can estimate the magnitude of the growth $e_{add_j}$ from the external product noise propagation formula. Finally,

$$\bar{e} = \sum_{j\neq i}^{k} \tilde{e}_j - (k-2)e = (k-1)e + \sum_{j\neq i}^{k} e_{add_j} - (k-2)e = e + \sum_{j\neq i}^{k} e_{add_j},$$

$$\|\bar{e}\|_\infty \leq \|e\|_\infty + (k-1)max_j(\|e_{add_j}\|_\infty) = \|e\|_\infty + 2(k-1)\ell\beta max_j(\|\mathsf{Err}(\mathbf{A}_j)\|_\infty) + 2\epsilon.$$

Therefore, the noise growth after partial decryption procedure is quite small since $\mathsf{Err}(\mathbf{A}_j)$ is the error of a fresh $\mathsf{TGSW}$ ciphertext $\mathbf{A}_j$ for $j \in [k]\backslash\{i\}$.

**Semantic security of** $\mathsf{MTFHE1}$ We prove the semantic security of the above multikey homomorphic encryption scheme.

**Theorem 3.** $\mathsf{MTFHE1}$ *scheme with the distributed decryption protocol is semantically secure assuming the hardness of the underlying* $\mathsf{TLWE}$ *problem.*

The security against an internal adversary is trivial. Since the evaluated multikey ciphertext is a $\mathsf{TLWE}$ ciphertext $(\mathbf{a}_1, \ldots, \mathbf{a}_k, b)$, where $b = \frac{1}{4}m - \sum_{i=1}^{k} < \mathbf{a}_i, \mathbf{s}_i > + e$ which is semantically secure itself by TLWE assumption even if any user partially decrypts it with its own secret key (i.e. $b + < \mathbf{a}_i, \mathbf{s}_i >$ is also a $\mathsf{TLWE}$ ciphertext for $i \in [k]$). We now prove the security of the above scheme against external adversary.

*Proof.* The security game defined in Section 3 follows: After the step 4, all $\mathcal{A}$ has got is a multikey ciphertext $\hat{\mathsf{ct}} = (\mathbf{a}_1, \ldots, \mathbf{a}_k, b)$, $p_{i,j} = (\mathbf{a}_{i,j}, b_{i,j})$ for $i \in [k] \backslash \{j\}$. What $\mathcal{A}$ may perform is to do $\sum_{i \neq j}^{k} p_{i,j} - (k-2)(\mathbf{a}_j, b)$. The result is $(\mathbf{a}'_j, \frac{1}{4}m - <\mathbf{a}'_j, \mathbf{s}_j> + error)$, which is a $\mathsf{TLWE}$ ciphertext under $\mathbf{s}_j$. Then $\mathcal{A}$ gives the result to a $\mathsf{TLWE}$ distinguisher $\mathcal{D}$ and $\mathcal{A}$ outputs whatever $\mathcal{D}$ outputs. By the $\mathsf{TLWE}$ assumption, the advantage of $\mathcal{D}$ is negligible, so is $\mathcal{A}$'s. $\square$

We note that the same technique can be applied to the recent MFHE schemes for batched ciphertext [15] in which the external product would be replaced with a tensor product.

## 5 Round optimal MPC protocol without a CRS via two MFHE schemes

Multikey fully homomorphic encryption (MFHE) scheme is known as achieving a round efficient multi party compuation (MPC) [1, 16, 18]. Mukherjee and Wichs [18] constructed a round optimal(2 round) MPC with CRS and Kim, Lee, and Park [16] achieved a three round MPC without a CRS via interactive MFHE scheme (against semi-malicious adversaries). There is a 2 round semi malicious secure MPC protocol without CRS assuming the existence of two round oblivious transfer (OT) [12]. Also, if there is a two round MPC protocol without CRS via MFHE, and the ciphertexts and public keys are stored to reuse, it can be done to get a correct output for one round assuming the adversaries are static. Such scenario can be achieved in a hospital. For instance, authorized doctors want to experiment using several patients' data encrypted under individual keys. Once their public keys and encrypted data are registered, doctors can do any computation on them to get a result executing one round MPC via MFHE. However, achieving the full security without CRS takes at least 4 rounds, which is proved in [11].

Assuming CRS in any protocol for multi parties might be a quite strong assumption and does not fit in real situation. Like Kim, Lee and Park's scheme, it seems that alternating a role of CRS can be achieved sharing public keys allowing an interaction among parties as a trade off. We construct an optimal round(two round) MPC protocol without CRS against honest-but curious adversaries, combining non-interactive MFHE scheme with CRS and interactive MFHE scheme without CRS. First, we convert MTFHE1 to an interactive version MTFHE2 then construct a MPC protocol.

### 5.1 MTFHE2 scheme without common random string

We convert the non-interactive scheme for bootstrapping part of MTFHE1 to have an interaction before decryption. In particular, we convert it into designated multikey homomorphic encryption scheme as [10]. It means that all parties share their public keys and relate keys to alter the role of common random string. In MTFHE1, a common random string is the common random parameter $\mathbf{a}$. Instead, each party generates its public key independently, then publishes it. It suffices to show how multikey TRGSW ciphertexts (Section 3.2 of [14]) are correctly constructed substituting the common random parameter $\mathbf{a}$ to $\mathbf{a}_i$ for $i \in [k]$, where $k$ is the fixed number of parties, since the common random parameter only has an effect on those procedure. Other algorithms are compatible with our modification. After showing that, we construct MTFHE2 with multi decryptor.

- $\mathsf{mTRGSW.Setup}(1^\lambda)$: It outputs $\mathsf{evparam} = (N, \psi, \alpha, \mathbf{g}, \ell)$, where $N$ is TRLWE dimension, $\psi$ is a key distribution over $\mathbb{Z}_N[X]$, $\alpha$ is an error parameter, and $\mathbf{g}$ and $\ell$ are TRGSW parameter.
- $\mathsf{mTRGSW.KeyGen}(\mathsf{evparam})$: Sample a secret $z \leftarrow \psi$ and set a vector $\mathbf{z} = (z, 1)$. Sample an error vector $\mathbf{e} \leftarrow D_\alpha^\ell$ over $\mathbb{T}_N[X]$ and a random vector $\mathbf{a} \leftarrow \mathbb{T}_N[X]^\ell$. Set the public key as $\mathbf{P} \leftarrow \mathsf{pk} = [\mathbf{a}, \mathbf{b}] \in \mathbb{T}_N[X]^{\ell \times 2}$, where $\mathbf{b} = -\mathbf{a} \cdot z + \mathbf{e} \pmod 1$. It returns $(z, \mathsf{pk})$.
- $\mathsf{mTRGSW.Enc}(\mu, z, \{\mathsf{pk}_j\}_{j\in[k]}, i)$: It takes a plaintext $\mu \in \mathbb{Z}_N[X]$, a secret $\mathbf{z}$, and all involved parties' public keys, (it is run by a party $i$) it returns a multikey ciphertext $\hat{\mathbf{C}}_i \in \mathbb{T}_N[X]^{\ell(k+1)\times(k+1)}$. And the procedure is following:
  **(1)** Sample $\mathbf{c}_0 \leftarrow \mathbb{T}_N[X]^\ell$ and $\mathbf{e}_c \leftarrow D_\alpha^\ell$ uniformly at random. Set $\mathbf{C}_i = [\mathbf{c}_{0,i}|\mathbf{c}_{1,i}]$, where $\mathbf{c}_{1,i} = -z_i \cdot \mathbf{c}_{0,i} + \mathbf{e}_c + \mu\mathbf{g} \pmod 1$.
  **(2)** Sample a randomness $r_j \leftarrow \psi$ and an error matrix $\mathbf{E}_j \leftarrow D_\alpha^{\ell\times 2}$ for $j \in [k]$. Output $\mathbf{D}_j = [\mathbf{d}_{0,j}|\mathbf{d}_{1,j}] = r_j\mathbf{P}_j + \mathbf{E}_j + [\mu \cdot \mathbf{g}|\mathbf{0}] \pmod 1 \in \mathbb{T}_N[X]^{\ell\times 2}$ for $j \in [k]$. And for $j \in [k]\backslash\{i\}$, set $\bar{\mathbf{D}}_j = [\bar{\mathbf{d}}_{0,j}|\bar{\mathbf{d}}_{1,j}] = r_j \cdot \mathbf{P}_i + \bar{\mathbf{E}}_j$, where $\bar{\mathbf{E}}_j \leftarrow D_\alpha^{\ell\times 2}$ uniformly at random.
  **(3)** Sample $\mathbf{f}_0 \leftarrow \mathbb{T}_N[X]^\ell$, $\mathbf{e}_f \leftarrow D_\alpha^\ell$ uniformly at random. Set a ciphertext $\mathbf{F} = [\mathbf{f}_0|\mathbf{f}_1] \in \mathbb{T}_N[X]^{\ell\times 2}$ where $\mathbf{f}_1 = -z_i \cdot \mathbf{f}_0 + \mathbf{e}_f \pmod 1$.

$$\hat{\mathbf{C}}_i := \begin{bmatrix} \mathbf{d}_{0,1} & \cdots & \bar{\mathbf{d}}_{0,1} + \mathbf{f}_0 & \cdots & \mathbf{0} & \mathbf{f}_1 + \mathbf{d}_{1,1} + \bar{\mathbf{d}}_{1,1} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{d}_{0,i} & \cdots & \mathbf{0} & \mathbf{d}_{1,i} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \bar{\mathbf{d}}_{0,k} + \mathbf{f}_0 & \cdots & \mathbf{d}_{0,k} & \mathbf{f}_1 + \mathbf{d}_{1,k} + \bar{\mathbf{d}}_{1,k} \\ \mathbf{0} & \cdots & \mathbf{c}_{0,i} & \cdots & \mathbf{0} & \mathbf{c}_{1,i} \end{bmatrix} \in \mathbb{T}_N[X]^{\ell(k+1)\times(k+1)}$$

  Note that the $(j, j)$-th component of $\hat{\mathbf{C}}_i$ is $\mathbf{d}_{0,j}$ for $j \in [k]$. The elements other than the diagonal, the $i$-th column and $k + 1$ th column are zero vectors.
- $\mathsf{mTRGSW.Dec}(\{\mathsf{sk}_j\}_{j\in[k]}, \hat{\mathbf{C}})$: Given all the involved secret keys $\{\mathsf{sk}_j\}$ and a multikey ciphertext $\hat{\mathbf{C}}$, it returns a message $\mu$.

**Correctness** Then we check the correctness i.e. $\hat{\mathbf{C}}_i\hat{\mathbf{z}} \approx \mu\mathbf{H}\hat{\mathbf{z}} \pmod 1$, where $\hat{\mathbf{z}}$ is a concatenation of each party's secret vector $(\mathbf{z}_1, \ldots, \mathbf{z}_k, 1)$. The correctness is done with the following equation:

- $z_i \cdot \mathbf{c}_0 + \mathbf{c}_1 + = \mathbf{C}_i\mathbf{z}_i \approx \mu \cdot \mathbf{g} \pmod 1$.
- $z_i \cdot \bar{\mathbf{d}}_{0,j} + \bar{\mathbf{d}}_{1,j} = \bar{\mathbf{D}}_j\mathbf{z}_i \approx 0 \pmod 1$.
- $z_i \cdot \mathbf{f}_0 + \mathbf{f}_1 = \mathbf{F}\mathbf{z}_i \approx 0 \pmod 1$.

– $z_j \cdot \mathbf{d}_{0,j} + \mathbf{d}_{1,j} = \mathbf{D}_j \mathbf{z}_j \approx \mu z_j \mathbf{g} \pmod 1$.

**Theorem 4.** *The above MFHE scheme* $\mathsf{mTRGSW}$ *with a single decryptor is semantically secure by the underlying* $\mathsf{TRLWE}$ *assumption.*

*Proof.* For a single key ciphertext, the two distributions $\{(\mathbf{P}_i, \mathbf{C}_i, \mathbf{D}_i = [\mathbf{d}_{0,i}|\mathbf{d}_{1,i}])\}$ and $\{(\mathbf{P}_u, \mathbf{C}_u, \mathbf{D}_u),\ \mathbf{P}_u, \mathbf{C}_u, \mathbf{D}_u \leftarrow \mathbb{T}_N[X]^{\ell\times 2}$ uniformly at random $\}$ are computationally indistinguishable by the underlying $\mathsf{TRLWE}$ problem. Therefor the single key ciphertext for $i$-th party is semantically secure. Then we consider the internal security for the other information related with other keys. For the $j$-th user holding $z_j$ and $j \in [k]\backslash\{i\}$, what she can do is $z_j \cdot \mathbf{d}_{0,j} + \mathbf{f}_1 + \mathbf{d}_{1,j} + \bar{\mathbf{d}}_{1,j}$. Then the result is $\mu_i z_j \cdot \mathbf{g} + \mathbf{f}_1 \bar{\mathbf{d}}_{1,j}$, which looks totally uniform random element for her. The reason follows:
$(\mathbf{a}_i, \bar{\mathbf{d}}_{0,j} = r_j \cdot \mathbf{a}_i + \bar{\mathbf{e}}_{0,j})$ is computationally indistinguishable with $(\mathbf{a}_i, \mathbf{u})$ by the underlying $\mathsf{TRLWE}$ problem, where $\mathbf{u} \leftarrow \mathbb{T}_N[X]^{\ell}$ uniformly at random, since $r_j$ is a secret chosen by the $i$-th user. This implies that $\bar{\mathbf{d}}_{0,j} + \mathbf{f}_0$ looks also uniformly random. Thanks to this, $\mathbf{f}_0$ is not revealed at all so that $\mathbf{f}_1$ itself is considered as a uniform element. Then it makes $\mathbf{f}_1 + \bar{\mathbf{d}}_{1,j}$ looks uniformly random. As a result, no internal adversary can distinguish if the given multikey ciphertext encrypts 0 or 1. In other words, the $\mathsf{mTRGSW}$ scheme achieves the internal security so that it is semantically secure by the underlying $\mathsf{TRLWE}$ assumption. □

Now, we construct an interactive MFHE scheme $\mathsf{MTFHE2}$ from $\mathsf{MTFHE1}$ and $\mathsf{mTRGSW}$ scheme.

– $\mathsf{MTFHE2}.\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{params})$: It runs $\mathsf{MTFHE1}.\mathsf{Setup}(1^\lambda)$ and we now specify the $\mathsf{evparam}$ is included in $\mathsf{params}$. $\mathsf{evparam} \leftarrow \mathsf{mTRGSW}.\mathsf{Setup}(1^\lambda)$.
– $\mathsf{MTFHE2}.\mathsf{KeyGen}(\mathsf{params}) \rightarrow (z, \mathsf{pk}, \mathsf{sk})$ : It runs $\mathsf{MTFHE1}.\mathsf{KeyGen}(\mathsf{params})$ to get $\mathsf{sk}$ and runs $\mathsf{mTRGSW}.\mathsf{KeyGen}(\mathsf{evparam})$ to get $(z, \mathsf{pk})$
– $\mathsf{MTFHE2}.\mathsf{Enc}(\mathsf{pk}_1, \ldots, \mathsf{pk}_k, \mathsf{sk}_i, z, \mu)$: It runs $\mathsf{MTFHE1}.\mathsf{Enc}(\mu, \mathsf{sk}_i)$ to get a ciphertext $\mathsf{ct}$ and runs $\mathsf{mTRGSW}(\mathsf{sk}_i[t], z, \mathsf{pk}_1, \ldots, \mathsf{pk}_k, i)$ to get $\hat{\mathbf{C}}_{i,t}$ for $t \in [n]$. Set $\{\mathbf{C}_{i,t}\}_{t\in[n]}$ as $\mathsf{evk}_i$. It outputs $\mathsf{ct}$ and $\mathsf{evk}$
– $\mathsf{MTFHE2}.\mathsf{Dec}(\hat{\mathsf{ct}}, \mathsf{sk}_1, \ldots, \mathsf{sk}_k)$ : It runs $\mathsf{MTFHE1}.\mathsf{Dec}(\hat{\mathsf{ct}}, \mathsf{sk}_1, \ldots, \mathsf{sk}_k)$ and outputs the message $\mu$.
– $\mathsf{MTFHE2}.\mathsf{Eval}(\hat{\mathsf{ct}}_1, \hat{\mathsf{ct}}_2, \{\mathsf{evk}_j\}_{j\in[k]})$ : It runs $\mathsf{MTFHE1}.\mathsf{Eval}(\hat{\mathsf{ct}}_1, \hat{\mathsf{ct}}_2)$ and bootstrapping algorithm of [14] with $\{\mathsf{evk}_j\}_{j\in[k]}$ then outputs the evaluated ciphertext $\hat{\mathsf{ct}}'$.

Note that the decryption protocol is done by $\mathsf{MTFHE1}.\mathsf{PartDec}$ and $\mathsf{MTFHE1}.\mathsf{FinDec}$ having an interaction with other parties defined in Section 4. We do not cover how bootstrapping procedure works with $\mathsf{evk}_j$ since it is exactly the same procedure as the original paper [14]. We have already checked the correctness of $\mathsf{mTRGSW}$ ciphertext so that all other algorithms work correctly. This modification allows to get rid of the assumption of the common random parameter (CRS) among all users. However, the number of users should be fixed before the computation, which might be a negative point for some computation.

### 5.2 2 round MPC without a CRS via $\mathsf{MTFHE2}$

We give an optimal round MPC protocol without CRS below. It is not necessary to use our distributed decryption protocol in a MPC protocol for the last step since traditional MPC only cares of input security. Applying what decryption protocol depends on a purpose of a computation.

Let $f : \{0,1\}^k \to \{0,1\}$ be the function to compute.

**Setup.** Run $\mathsf{params} \leftarrow \mathsf{MTFHE2.Setup}(1^\lambda)$. Make sure that all the parties have $\mathsf{params}$.

**Input:** For $i \in [k]$, each party $U_i$ holds input $m_i \in \{0,1\}$, and wants to compute $f(m_1, \cdots, m_k)$.

**Round I.** Each party $U_i$ executes the following steps:

- Generates its public key $\mathsf{pk}_i$ and secret keys by running $\mathsf{MTFHE2.KeyGen}(\mathsf{params})$.
- Encrypts a message $m_i$ running $\mathsf{MTFHE1.Enc}$ to get a $\mathsf{TLWE}$ sample $\mathsf{ct}_i = (\mathbf{a}_i, b_i)$.
- Broadcasts the public key $\mathsf{pk}_i, \mathsf{ct}_i$.

**Round II.** Once receiving public keys $\{\mathsf{pk}_k\}_{k \neq i}$, each party $U_i$ for $i \in [k]$ executes evaluation procedure with the following steps:

- Runs $\mathsf{mTRGSW}, \mathsf{Enc}(\mathsf{sk}_i[t], z, \mathsf{pk}_1, \ldots, \mathsf{pk}_k, i)$ to get $\mathsf{evk}_i$.
- Extends $\{\mathsf{ct}_j\}_{j\in[k]}$ to get $\{\hat{\mathsf{ct}}_j\}_{j\in[k]}$.
  - If the protocol is run for the first time, each party runs $\mathsf{MTFHE1.Eval}(\hat{\mathsf{ct}}_{j1}, \hat{\mathsf{ct}}_{j2})$ to get an evaluted ciphertext for $j1, j2 \in [k]$.
  - else, runs $\mathsf{MTFHE2.Eval}(\hat{\mathsf{ct}}_{j1}, \hat{\mathsf{ct}}_{j2}, \{\mathsf{evk}_j\}_{j\in[k]})$ to get an evaluted ciphertext for $j1, j2 \in [k]$.
- Runs $\mathsf{MTFHE2.PartDec}(\hat{\mathsf{ct}}, \mathsf{sk}_i)$ to get a partial decryption $p_{i,j}$ for $j \in [k]$.
- broadcasts $\mathsf{evk}_i, p_{i,j}$.

**Output:** On receiving all the values $\{p_{j,i}\}_{j\in[k]}$, each party $U_i$ runs the final decryption algorithm to obtain the function value $f(m_1, \cdots, m_k)$:

$$y \leftarrow \mathsf{MTFHE2.FinDec}(\{p_{j,i}\}_{j\in[k]}, \mathsf{sk}_i),$$

and output $y = f(m_1, \cdots, m_k)$.

The above protocol is limited to run a function which has a depth which guarantees that the decryption never fails only for the first time. In fact, very limited operation is possible (for instance, gate (NOT gate only) for TFHE, addition for BFV scheme). After that, from the second time, all parties can run the protocol with arbitrary function without predefined depth since the parties share bootstrapping key$\{\mathsf{evk}_j\}_{j\in[k]}$ in the second round. Therefore, once the keys $\{\mathsf{pk}_j\}_{j\in[k]}, \{\mathsf{evk}_j\}_{j\in[k]}$ are published, the protocol does not generate keys for the same number of parties and the parties reuse the keys.

**Security** This 2 round MPC protocol is secure against honest but curious adversaries. Honest but curious adversary is a legitimate party in a communication protocol who does not deviate from the defined protocol but will attempt to learn all possible information from legitimately received messages. Then we can see that the security is guaranteed by the semantic security of $\mathsf{MTFHE2}$.

**Acknowledgement.** Hyang-Sook Lee and Jeongeun Park were supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2018R1A2A1A05079095).

## References

1. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. Lecture Notes in Computer Science, vol. 7237, pp. 483–501. Springer, Heidelberg, Germany, Cambridge, UK (Apr 15–19, 2012). https://doi.org/10.1007/978-3-642-29011-4$_2$9
2. Boneh, D., Gennaro, R., Goldfeder, S., Jain, A., Kim, S., Rasmussen, P.M.R., Sahai, A.: Threshold cryptosystems from threshold fully homomorphic encryption. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part I. Lecture Notes in Computer Science, vol. 10991, pp. 565–596. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2018). https://doi.org/10.1007/978-3-319-96884-1$_1$9
3. Brakerski, Z., Halevi, S., Polychroniadou, A.: Four round secure computation without setup. In: Theory of Cryptography Conference. pp. 645–677. Springer (2017)
4. Brakerski, Z., Perlman, R.: Lattice-based fully dynamic multi-key FHE with short ciphertexts. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part I. Lecture Notes in Computer Science, vol. 9814, pp. 190–213. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2016). https://doi.org/10.1007/978-3-662-53018-4$_8$
5. Chen, L., Zhang, Z., Wang, X.: Batched multi-hop multi-key FHE from ring-LWE with compact ciphertext extension. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017: 15th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 10678, pp. 597–627. Springer, Heidelberg, Germany, Baltimore, MD, USA (Nov 12–15, 2017). https://doi.org/10.1007/978-3-319-70503-3$_2$0
6. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology – ASIACRYPT 2016, Part I. Lecture Notes in Computer Science, vol. 10031, pp. 3–33. Springer, Heidelberg, Germany, Hanoi, Vietnam (Dec 4–8, 2016). https://doi.org/10.1007/978-3-662-53887-6$_1$
7. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology – ASIACRYPT 2017, Part I. Lecture Notes in Computer Science, vol. 10624, pp. 377–408. Springer, Heidelberg, Germany, Hong Kong, China (Dec 3–7, 2017). https://doi.org/10.1007/978-3-319-70694-8$_1$4
8. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption over the torus. Cryptology ePrint Archive, Report 2018/421 (2018), https://eprint.iacr.org/2018/421
9. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part II. Lecture Notes in Computer Science, vol. 9216, pp. 630–656. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015). https://doi.org/10.1007/978-3-662-48000-7$_3$1

10. Eunkyung Kim, Hyang-Sook Lee, J.P.: Towards round-optimal secure multiparty computations: Multikey fhe without a crs. Cryptology ePrint Archive, Report 2018/1156 (2018), https://eprint.iacr.org/2018/1156
11. Garg, S., Mukherjee, P., Pandey, O., Polychroniadou, A.: The exact round complexity of secure computation. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part II. Lecture Notes in Computer Science, vol. 9666, pp. 448–476. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016). https://doi.org/10.1007/978-3-662-49896-5$_1$6
12. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part II. Lecture Notes in Computer Science, vol. 10821, pp. 468–499. Springer, Heidelberg, Germany, Tel Aviv, Israel (Apr 29 – May 3, 2018). https://doi.org/10.1007/978-3-319-78375-8$_1$6
13. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 75–92. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 18–22, 2013). https://doi.org/10.1007/978-3-642-40041-4$_5$
14. Hao Chen, I.C., Song, Y.: Multi-key homomophic encryption from tfhe. Cryptology ePrint Archive, Report 2019/116 (2019), https://eprint.iacr.org/2019/116
15. Hao Chen, Wei Dai, M.K., Song, Y.: Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference. Cryptology ePrint Archive, Report 2019/524 (2019), https://eprint.iacr.org/2019/524
16. Kim, E., Lee, H.S., Park, J.: Towards round-optimal secure multiparty computations: Multikey FHE without a CRS. In: Susilo, W., Yang, G. (eds.) ACISP 18: 23rd Australasian Conference on Information Security and Privacy. Lecture Notes in Computer Science, vol. 10946, pp. 101–113. Springer, Heidelberg, Germany, Wollongong, NSW, Australia (Jul 11–13, 2018). https://doi.org/10.1007/978-3-319-93638-3$_7$
17. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Karloff, H.J., Pitassi, T. (eds.) 44th Annual ACM Symposium on Theory of Computing. pp. 1219–1234. ACM Press, New York, NY, USA (May 19–22, 2012). https://doi.org/10.1145/2213977.2214086
18. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part II. Lecture Notes in Computer Science, vol. 9666, pp. 735–763. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016). https://doi.org/10.1007/978-3-662-49896-5$_2$6
19. Peikert, C., Shiehian, S.: Multi-key FHE from LWE, revisited. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B: 14th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 9986, pp. 217–238. Springer, Heidelberg, Germany, Beijing, China (Oct 31 – Nov 3, 2016). https://doi.org/10.1007/978-3-662-53644-5$_9$