Multi-key Fully Homomorphic Encryption with Non-Interactive Setup in the Plain Model

Seonhong Min[1], Jeongeun Park[2], Yongsoo Song[1]

Seoul National University[1]
Norwegian University of Science and Technology[2]

Fully Homomorphic Encryption (FHE)

- A cryptographic primitive for **private computation**.
- Private circuit evaluation **without** leaking information.
- Used in many cryptographic protocols:
  - Private Set Intersection (PSI)
  - Private Information Retrieval (PIR)
  - Privacy-Preserving Machine Learning (PPML)

2PC and FHE

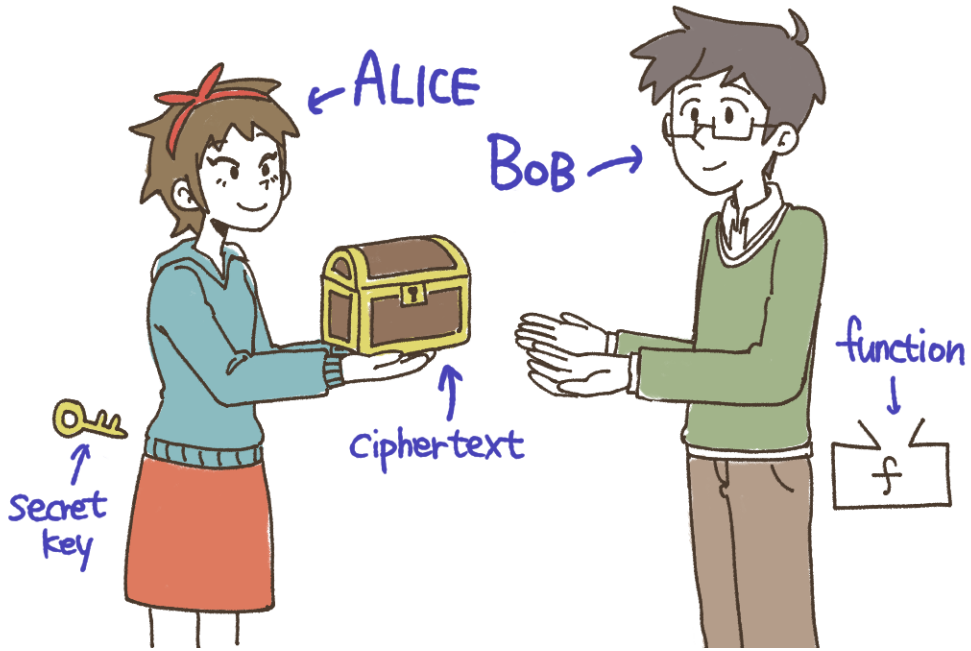
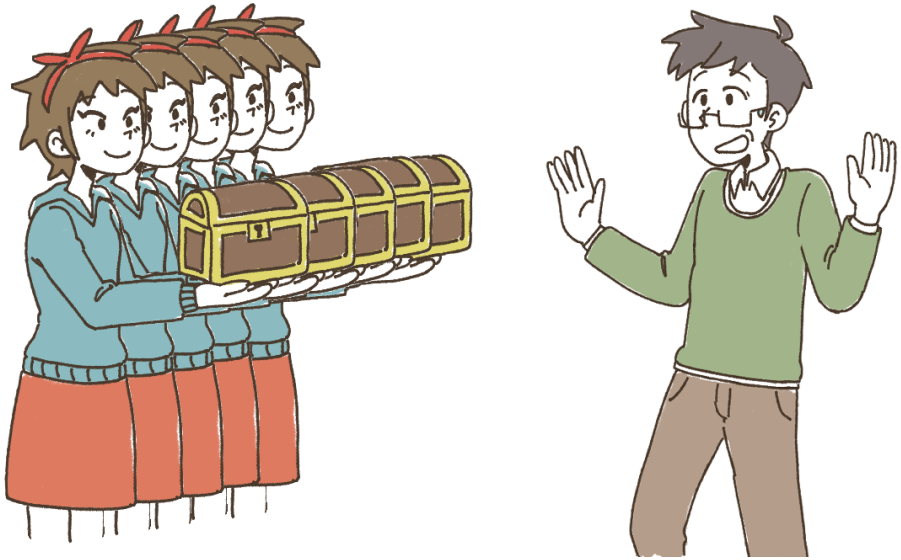


Figure: Basic **two-party computation (2PC)** protocol from FHE

- 1. **Alice** generates *evaluation keys* and *ciphertexts*.
- 2. **Bob** evaluates *function* over ciphertexts.
- 3. Alice decrypts the output ciphertext.

FHE for MPC



– What if there are **multiple participants**?

– **Multi-key FHE (MKFHE)**

- Ciphertexts encrypted under **different keys**
- Non-interactive setup, Dynamism

MPC from MKFHE

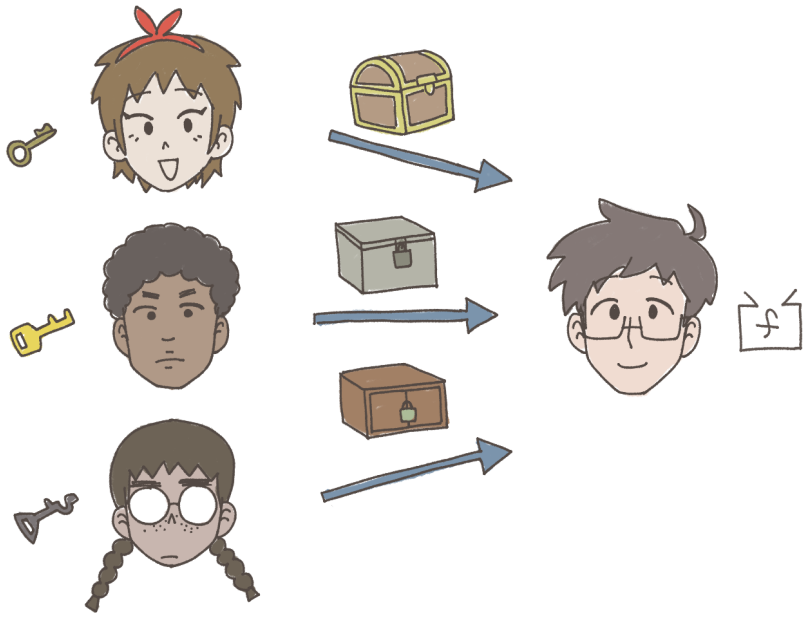


Figure: Basic **Multi-Party Computation (MPC)** protocol from MKFHE.

- Each party outputs a single-key ciphertext.
- The evaluator then computes the circuit using ciphertexts in a private manner.
- The parties collaborate to recover the underlying message.

Limitations of MKFHE

Definition (Common Random String (CRS) Assumption)

- CRS is a **random vector** of finite ring elements.
- A CRS is distributed to the parties *before* setup.
- Hard to instantiate in the real-world protocols.

Most MKFHE schemes rely on the CRS assumption.

- LTV [1] scheme **does not** require CRS assumption.
- LTV relies on the DSPR problem with **insecure** parameters.
- Most MKFHE schemes relies on the **(R)LWE problem**.

[1] On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption, STOC '12

Limitations of MKFHE

It would be interesting to get a 2 round protocol in the honest-but-curious model without a CRS. One way to achieve this would be to build an MKFHE without any common public parameters.

Mukherjee and Wichs, 2016

Is it possible to construct an MKFHE under standard assumptions such as LWE in the plain model?

Chongchitmate and Ostrovsky, 2017

The most compelling problem is to construct an MKFHE with one-round decryption assuming only the hardness of the (plain) LWE problem.

Ananth, Jain, Jin and Malavolta, 2020

Main Contributions

- In this work, we resolve the aforementioned *open problem*!
- Concretely, we provide
  - A new MK somewhat HE for poly eval without CRS
  - A new **(R)LWE-based MK-GSW without CRS**
  - A semi-honest **2-round** MPC protocol in the **plain model**
  - MK variants of...
    - BGV/BFV/CKKS
    - FHEW/TFHE/LMKCDEY/Carousel

Notations

- $R = \mathbb{Z}[X]/\Phi_M(X)$ and $R_Q = R/QR$.
- $\mathcal{U}(\mathcal{S})$ is a uniform distribution over a finite set $\mathcal{S}$.
- $\otimes$ is tensor product of vectors/matrices.
- $\vec{g} \in \mathbb{Z}^d$ and $g^{-1} : R_Q \to R^d$ satisfy
  - For any $a \in R_Q$,
  - 1. $g^{-1}(a)$ is small

    2. $\vec{g}^\top \cdot g^{-1}(a) = a \pmod{Q}$.
  - Typically, $\vec{g} = (1, 2^1, \ldots, 2^{\lfloor \log Q \rfloor})$ and g^{-1} is binary decomposition.

Intuition

- The main challenge: homomorphic **multiplication**.
- Quick review:
  - Let $\mathbf{s} = (s_1, \ldots, s_k)$ be secret.
  - Given $(b, a_1, \ldots, a_k)$ s.t. $b + \sum_{1 \le i \le k} a_i s_i \approx m$
  - To multiply μ, homomorphically compute

$$\mu \cdot b + \sum_{1 \le i \le k} a_i \cdot (\mu s_i) \approx \mu \cdot m$$

    from encryptions of μ and $\mu \cdot s_i$.
  - How do we generate encryption of s_i?
    $\rightarrow$ Use **CRS** to generate $\mu \cdot s_i$ on-the-fly.

Intuition

- Instead,
  - Given $(b, a_1, \ldots, a_k)$,

    generate an encryption of $(\mu b, \mu a_1, \ldots, \mu a_k)$.
  - Naturally $\mu b + \sum_{1 \le i \le k} \mu a_i s_i \approx \mu \cdot m$.
- An encryption scheme for vectors is required!

  $\rightarrow$ **Shared-A RLWE**.
- Does not require the **CRS** for multiplication.

Shared-A RLWE

Definition (Shared-A RLWE)

- Secret $\mathbf{s} = (s_i)_{1 \leq i \leq k}$.
- $a \leftarrow \mathcal{U}(R_Q)$, and $e_i \leftarrow \psi$ for $1 \leq i \leq k$.
- $(b_1 = -as_1 + e_1, \ldots, b_k = -as_k + e_k, a) \approx_c \mathcal{U}(R_Q)^{k+1}$.

– The **security** follows from conventional RLWE.

– Used in many works, including

  ▶ Lattice reduction, lossy trapdoor function, FHE, ...

– Allows for encrypting a **vector v** of ring elements.

Shared-A RLWE

- **Encryption**
  - $a \leftarrow \mathcal{U}(R_Q)$, $\mathbf{e} \leftarrow \psi^k$
  - $$a \cdot \begin{bmatrix} -\mathbf{s} \\ 1 \end{bmatrix} + \begin{bmatrix} \mathbf{v} + \mathbf{e} \\ 0 \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_k \\ a \end{bmatrix} \pmod{Q}$$
- **Decryption**
  - $$\left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \begin{bmatrix} b_1 \\ \vdots \\ b_k \\ a \end{bmatrix} \approx \mathbf{v} \pmod{Q}$$

cLEV and cGSW

- **compact Leveled Encryption (cLEV)**
  - shared-A encryptions of each column of $\mu \cdot \vec{g}^\top \otimes \mathbf{I}_k$.
  - i.e., $\left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \mathbf{C} \approx \mu \cdot \vec{g}^\top \otimes \mathbf{I}_k$
- **Intuitively...**
  - $$\begin{bmatrix} \mathsf{sARLWE}(\vec{g} \cdot \mu, 0, \ldots, 0) \\ \mathsf{sARLWE}(0, \vec{g} \cdot \mu, \ldots, 0) \\ \vdots \\ \mathsf{sARLWE}(0, 0, \ldots, \vec{g} \cdot \mu) \end{bmatrix}$$

cLEV and cGSW

- **Vector gadget product**
  - Input: a vector $\mathbf{v} \in R_Q^k$, and a cLEV enc $\mathbf{C}$ of μ
  - Compute $\mathbf{v} \odot_v \mathbf{C} = \mathbf{C} \cdot g^{-1}(\mathbf{v})$.
  - Output: shared-A encryption of $\mu \cdot \mathbf{v}$.

$$\left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot (\mathbf{v} \odot_v \mathbf{C}) = \left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \mathbf{C} \cdot g^{-1}(\mathbf{v})$$

$$\approx \mu \cdot \vec{g}^{\top} \otimes \mathbf{I}_k \cdot g^{-1}(\mathbf{v}) = \mu \cdot \mathbf{v}$$

cLEV and cGSW

- **Intuitively...**

$$
\begin{aligned}
& g^{-1}(v_1) \cdot \mathsf{sARLWE}(\vec{g} \cdot \mu, 0, \ldots, 0) \\
+\ & g^{-1}(v_2) \cdot \mathsf{sARLWE}(0, \vec{g} \cdot \mu, \ldots, 0) \\
+\ & \ldots \\
+\ & g^{-1}(v_k) \cdot \mathsf{sARLWE}(0, 0, \ldots, \vec{g} \cdot \mu) \\
=\ & \mathsf{sARLWE}(\mu v_1,\ \mu v_2,\ \ldots,\ \mu v_k) \\
=\ & \mathsf{sARLWE}(\mu \cdot \mathbf{v}).
\end{aligned}
$$

cLEV and cGSW

- **compact GSW (cGSW)**
  - shared-A encryptions of each column of $\mu \cdot \vec{g}^\top \otimes \left[\mathbf{I}_k \mid \mathbf{s}\right]$.
  - i.e., $\left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \overline{\mathbf{C}} \approx \mu \cdot \vec{g}^\top \otimes \left[\mathbf{I}_k \mid \mathbf{s}\right]$
- **Vector external product**
  - Input: a shared-A enc $\mathbf{c} \in R_Q^{k+1}$, and a cGSW enc $\overline{\mathbf{C}}$ of μ
  - Compute $\mathbf{c} \boxdot_v \overline{\mathbf{C}} = \overline{\mathbf{C}} \cdot g^{-1}(\mathbf{c})$.
  - Output: a shared-A enc of $\mu \cdot \left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \mathbf{c}$

$$\left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \left(\overline{\mathbf{C}} \odot_v \mathbf{c}\right) = \left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \overline{\mathbf{C}} \cdot g^{-1}(\mathbf{c})$$
$$\approx \mu \cdot \vec{g}^\top \otimes \left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot g^{-1}(\mathbf{c}) = \mu \cdot \left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \mathbf{c}$$

cLEV and cGSW

- cLEV and cGSW allow addition/multiplication of vectors.
- Concretely...
  - 1. a vector $\mathbf{v} \in R_Q^k$,
    2. cLEV enc $\mathbf{C}$ of μ_0, and
    3. cGSW enc $\overline{\mathbf{C}}_i$ of μ_i for $1 \leq i \leq n$,
  - $\overline{\mathbf{C}}_n \boxdot_v \left(\overline{\mathbf{C}}_{n-1} \boxdot_v \left(\dots \left(\overline{\mathbf{C}}_1 \boxdot_v (\mathbf{C} \odot_v \mathbf{v})\right)\right)\right)$
  - $\left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \overline{\mathbf{C}}_n \boxdot_v \left(\overline{\mathbf{C}}_{n-1} \boxdot_v \left(\dots \left(\overline{\mathbf{C}}_1 \boxdot_v (\mathbf{C} \odot_v \mathbf{v})\right)\right)\right)$

    $\approx \mu_n \cdot \left[\mathbf{I}_k \mid \mathbf{s}\right] \cdot \overline{\mathbf{C}}_{n-1} \boxdot_v \left(\dots \left(\overline{\mathbf{C}}_1 \boxdot_v (\mathbf{C} \odot_v \mathbf{v})\right)\right)$

    $\approx \dots \approx \mu_n \dots \mu_1 \mu_0 \mathbf{v}$

MKFHE from cLEV/cGSW

- Each i-th party outputs cLEV encryptions of dimension i.
  - Let $\mathbf{C}_i$ be a cLEV enc of μ_i under $\mathbf{s}_i$.
- For some $m \in R_Q$,
- First party computes $m \odot_v \mathbf{C}_1 \in R_Q^2$.
  - Encryption of $\mu_1 \cdot m$ under $\mathbf{s}_1$,
  - but also a **vector** of length 2!
- Second party then computes $(m \odot_v \mathbf{C}_1) \odot_v \mathbf{C}_2 \in R_Q^3$.
  - Encryption of $\mu_2 \cdot (m \odot_v \mathbf{C}_1)$ under $\mathbf{s}_2$,
  - which is an encryption of $\mu_2 \cdot \mu_1 \cdot m$ under
  - $\left[\mathbf{I}_1 \mid \mathbf{s}_1\right] \cdot \left[\mathbf{I}_2 \mid \mathbf{s}_2\right]$.

MKFHE from cLEV/cGSW

- For some $m \in R_Q$, $(((m \odot_v \mathbf{C}_1) \dots) \odot_v \mathbf{C}_{k-1}) \odot_v \mathbf{C}_k$
  is an encryption of $m \cdot \prod_{1 \le i \le k} \mu_i$
  under secret $\overline{\mathsf{sk}} = \left[\mathbf{I}_1 \mid \mathbf{s}_1\right] \cdot \left[\mathbf{I}_1 \mid \mathbf{s}_2\right] \dots \left[\mathbf{I}_k \mid \mathbf{s}_k\right]$.

$$
\begin{aligned}
&\left[\mathbf{I}_1 \mid \mathbf{s}_1\right] \dots \left[\mathbf{I}_k \mid \mathbf{s}_k\right] \cdot (((m \odot_v \mathbf{C}_1) \dots) \odot_v \mathbf{C}_{k-1}) \odot_v \mathbf{C}_k \\
&\approx \left[\mathbf{I}_1 \mid \mathbf{s}_1\right] \dots \left[\mathbf{I}_{k-1} \mid \mathbf{s}_{k-1}\right] \cdot \mu_k \cdot ((m \odot_v \mathbf{C}_1) \dots) \odot_v \mathbf{C}_{k-1} \\
&\approx \dots \approx \mu_k \cdot \mu_{k-1} \dots \mu_2 \cdot \left[\mathbf{I}_1 \mid \mathbf{s}_1\right] \cdot m \odot_v \mathbf{C}_1 \\
&\approx \mu_k \cdot \mu_{k-1} \dots \mu_1 \cdot m
\end{aligned}
$$

MKFHE from cLEV/cGSW

- **Somewhat MKHE** from cLEV/cGSW
  - cLEV/cGSW support **multi-key multiplication**!
  - Addition is supported by the **linearity** of RLWE

  $\Rightarrow$ **Evaluation of polynomials** with degree chosen a-priori.

What about **full homomorphism***?*

MK-RGSW

- Our Idea: Extend cGSW to **MK-RGSW**!
- **Quick Review**
  - (multi-hop) **Circuit evaluation** on binary data
  - **Fully-homomorphic**.
  - For secret key $\overline{\mathsf{sk}} = (1, \overline{s}_1, \ldots, \overline{s}_k)$,
  - MLWE encryptions of $\mu g_j \overline{s}_i$ for message μ
- Recall $\overline{\mathsf{sk}} = \left[\mathbf{I}_1 \mid \mathbf{s}_1\right] \cdot \left[\mathbf{I}_1 \mid \mathbf{s}_2\right] \ldots \left[\mathbf{I}_k \mid \mathbf{s}_k\right] = (1, \overline{s}_1, \ldots, \overline{s}_k)$
- Each term $\overline{s}_i$ is **multivariate poly** of $\mathbf{s}_i$.

$\rightarrow$ Use cLEV/cGSW to generate these encryptions!

MK-RGSW

- Roughly...
- Each i-th party outputs
  - cGSW encryption of message
  - cLEV encryptions of $\mathbf{s}_i$
- Compute encryptions of $\mu g_j \overline{s}_i$ using cLEV/cGSW.
- **Circular security assumption** of cLEV required.
- An explicit iterative algorithm is given in the paper.

Formalisation

- Ciphertext Shrink
  - ▶ cLEV/cGSW dimension = **multiplication order**
  - $\rightarrow$ Parties need to know the order **beforehand**
  - ▶ Output ciphertext of the **maximum size** K
  - ▶ Shrink the ciphertext on-the-fly
  - ▶ Evaluator can **choose** the multiplication order
- Key-switching
  - ▶ Computation between ctxts with **different mult order**
  - ▶ Or, use the **lexicographical** order of evk

2-round MPC

- Our MK-RGSW does not have **linear decryption**.
  $\rightarrow$ Multi-round distributed decryption with MPC.
- **Ananth et al. (TCC '20)**
  1. 2-round semi-malicious OT in the plain model
  2. MKFHE with CRS and 1-round decryption
  3. **MKFHE in the plain model**

  $\rightarrow$ MKFHE in the plain model with 1-round decryption
  $\rightarrow$ **2-round semi-honest MPC** in the plain model
- Our scheme fills the missing spot!

Generic MKFHE construction

- BGV/BFV/CKKS
  - ATK : Encryptions of $\psi(\bar{\mathbf{s}})$
  - RLK : RGSW encryption of $\bar{\mathbf{s}}$
- TFHE/Carousel/LMKCDEY
  - BRK : Encryptions of $\delta_\ell(\bar{s}_{i,j})$
  - ATK
- Other schemes
  - If they rely on **RGSW**, we can construct them.

Performance

- We implement our scheme in Julia, on top of HIENAA.jl
- https://github.com/SNUCP/crsless

K	Vector Gadget Product	RGSW Extension	Internal Product	Ciphertext Size	Public Key Size
2	11ms	177ms	178ms	10.6MB	21.3MB
4	66ms	962ms	756ms	41.0MB	164.0MB
8	425ms	10897ms	5718ms	191.28MB	1.530GB

Table: Performance of vector gadget product, RGSW extension, and internal product algorithms, respectively, and the size of the ciphertexts and public key, with respect to the number of the parties K.

Q&A and Feedback

Any Questions? [2]

[2] All contents in this presentation are human-generated.