Received July 30, 2020, accepted September 4, 2020, date of publication October 14, 2020, date of current version October 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3030970

How to Securely Collaborate on Data: Decentralized Threshold HE and Secure Key Update

EUNKYUNG KIM[1], JINHYUCK JEONG[1], HYOJIN YOON[1], YOUNGHYUN KIM[1], JIHOON CHO[1], AND JUNG HEE CHEON[2], (Associate Member, IEEE)
[1]Security Research Center, Samsung SDS, Seoul 06765, South Korea
[2]Department of Mathematical Sciences, Seoul National University, Seoul 08826, South Korea

Corresponding authors: Eunkyung Kim (ek41.kim@samsung.com) and Jinhyuck Jeong (jhyuck.jeong@samsung.com)

This work was supported by the Institute for Information and Communications Technology Promotion (IITP) grant through the Korean Government (MSIT), Development and Library Implementation of Fully Homomorphic Machine Learning Algorithms Supporting Neural Network Learning Over Encrypted Data, under Grant 2020-0-00840.

ABSTRACT Threshold homomorphic encryption (Threshold HE) schemes are modified homomorphic encryption schemes to be suitable for privacy-preserving data integration and analysis. In actual usage of it, one should take it care into consideration who manages secret keys. In Eurocrypt 2012, Asharov *et al.* proposed decentralized (n, n)-threshold HE schemes in bottom-up approach for which all n parties must allow by doing *a partial decryption* to decrypt successfully a ciphertext. To support more general threshold structure for HE, Boneh *et al.* presented (t, n)-threshold HE schemes using secret sharing schemes in top-down approach with a central key dealer. In this article, decentralized (t, n)-threshold HE schemes in bottom-up approach will be constructed. The decentralized (n, n)-threshold HE scheme is fisrt modified to reduce the error contained in the common evaluation key which affects to the entire parameter size. Then by applying (t, n)-threshold secret sharing scheme, (n, n)-threshold HE scheme is converted to (t, n)-threshold HE scheme. Moreover, proactive secret sharing scheme is applied to update secret key share of the constructed (t, n)-threshold HE scheme whenever needed.

INDEX TERMS Fully homomorphic encryption, threshold decryption, proactive secret sharing.

I. INTRODUCTION

Data collection and analysis have become essential parts of the modern industry and companies want to collect user data to make more profit. However, data privacy regulations, such as EU General Data Protection Regulation (GDPR), force organizations to use data only for a pre-defined purposes and periods of time, and demand de-identification process before use, which may degrade the quality of the analysis. Thus, we live in the world where protecting data privacy is as important as accuracy of analysis. Data privacy is particularly fatal to organizations such as financial and healthcare sectors that deal with sensitive customer data. Obviously, accuracy has a big impact on these organizations. On the other hand, the demand to combine data with other organizations also has increased in order to obtain greater insight from the aggregated data by using advanced analysis and AI capabilities. The aggregated data must be further processed to ensure that there is no risk of re-identification because combining multiple datasets may lead to serious infringement of individual data privacy. The analysis results, however, may not be valuable as some important features in the dataset can be omitted or replaced with inaccurate values during the process.

This is where privacy-enhancing analysis technologies such as homomorphic encryption (HE) come into play to solve both the problem of degradation in analysis quality during the de-identification process. HE is one of the most promising cryptographic technologies for data usage in untrusted environments. HE supports computation on encrypted data, so that the original data can be completely protected during the encrypted analysis. Since the first construction of HE of Gentry [1], the efficiency has been improved significantly over the decade ([2]–[14]). Especially, CKKS scheme [15] that supports encrypted approximate

The associate editor coordinating the review of this manuscript and approving it for publication was Xiao Liu.

arithmetic of real numbers made great strides as it demonstrates the applicability of HE in real use cases ([16]–[21]).

By using HE, organizations can benefit from collaboration without exposing original data, compromising privacy, or concern about re-identification through data integration with other sources. All the computations are performed on the encrypted data, and only the result is decrypted by the person holding the secret key. To prevent unauthorized decryption, organizations can delegate the secret key management to a trusted third party (TTP) such as government agencies or courts. Unfortunately, such a TTP is not always available, and it may not be desirable to rely on a single party for maintaining the secret key.

Shamir [22] proposed the notion of (t, n)-threshold cryptography, where the secret shares are distributed among n parties and more than t shares allow a successful construction of the master secret. Following the idea, the notion of threshold decryption for HE was introduced [23], [24] in that a secret key is distributed among all the parties and decryption can be done when all agree. Two types of HE with threshold decryption – threshold HE and multi-key HE – have been proposed, both of which essentially share the same idea. The difference between threshold HE and multi-key HE can be understood as when the common public keys are generated, before or after the integration, respectively. It is reasonable to assume that organizations have a contract to collaborate on data before exchanging data. So it is more realistic to generate the common key before the encrypted data integration, and in real use cases threshold HE seems to be a better choice. Moreover, organizations typically hold their customer data, so they do not have full data ownership. Data privacy must be guaranteed for data owners (customers), not for data users (organizations), and data users may delegate secret key management to independent multiple TTPs. As the focus of the paper is secret key management of HE for real use cases that multiple TTPs manages secret key on behalf of data owners and data users, decentralized (t, n)-threshold HE scheme in bottom-up approach is studied.

In Eurocrypt 2012, Asharov *et al.* proposed decentralized (n, n)-threshold HE schemes in bottom-up approach from an HE scheme based on learning with errors (LWE) problem and ring LWE problem, respectively ([24], [25]). In the scheme, n parties establish common public keys collaboratively using the linearity of public keys while no one actually has the common secret key. The common secret key is implicitly defined by the summation of partial secret keys generated by each party, and all n parties must cooperate to decrypt a ciphertext. On the other hand, Boneh *et al.* proposed (t, n)-threshold HE schemes in top-down approach using $\{0, 1\}$-Linear secret sharing scheme and Shamir secret sharing scheme. Roughly speaking, a trusted third party or someone who can be one of n parties first generates a common key tuple $(\mathsf{pk}, \mathsf{evk}, \mathsf{sk})$ and then distributes this secret key sk into n shares $\{\mathsf{sk}_i\}_i$ using the secret sharing schemes. They also adopt an auxiliary public key encryption scheme to propose a decentralized threshold HE scheme, but this can be thought as a top-down approach in a nutshell.

The proactive secret sharing schemes refers to the secret sharing schemes that are secure against *mobile adversary* who can keep an eye on secret share holders over time but have a limitation for the number of accessible holders at a time unit. In order to protect the shared secret from the adversary, the shares should be periodically updated so that the shared secret remains the same and the previous shares are no longer useful. Since the first proactive secret sharing scheme [26], many proactive secret sharing schemes have been designed for various purposes; mobile proactive secret sharing schemes ([27], [28]), verifiable proactive secret sharing schemes ([29]–[31]).

Our Contribution. We first modify the (n, n)-threshold HE construction of [24]: From an HE scheme with key-homomorphic property, we construct (n, n)-threshold HE scheme by generating a key pair of the threshold version as a sum of key pairs of original HE scheme. We then use each secret key share, instead of the common public key, to encrypt itself for evaluation key generation. The evaluation key is constructed by aggregating encrypted secret key share of each party. As the construction of [24] uses the common public key to encrypt individual secret key, the evaluation key contains error proportional to n^2 where n is the number of participants. On the other hand, our approach allows error proportional to n which results in reducing the entire parameter size. Since the distribution of the combined key is different from the original key distribution, we show joint key security as in [24]. Joint key security assures that ciphertexts of our threshold HE scheme is indistinguishable from uniformly random one.

We then apply (t, n)-threshold secret sharing scheme to convert our (n, n)-threshold HE scheme to (t, n)-threshold HE scheme. We use a generalized Shamir secret sharing scheme and key-homomorphic property of the underlying HE scheme. Moreover, we introduce proactive secret sharing schemes to update secret key share of our (t, n)-threshold HE scheme whenever needed. In particular, we provide a simpler method for updating secret key shares on the (n, n)-threshold case. As a result, this scheme can get immune to the errors relative to the general case.

II. PRELIMINARIES

We use bold lowercase letters to denote vectors and bold uppercase letters to denote matrices. For a positive integer q, $[q]$ is defined by $\{1, 2, \ldots, q\}$ and $\mathbb{Z}_q = [-q/2, q/2) \cap \mathbb{Z}$ is a set of representatives of residues modulo q. Given an integer m and a modulus q, $[m]_q$ means an element $a \in \mathbb{Z}_q$ such that $m \equiv a \pmod{q}$. For a finite set, $a \leftarrow A$ means a is uniformly chosen from A. Through this article, n is the number of parties with a threshold $t \leq n$ and $P_1, \ldots, P_n$ means a set of parties.

For a power-of-two N, we use $\mathcal{R} = \mathbb{Z}[X]/(X^N + 1)$ to denote the ring of integers of a number field $\mathbb{Q}[X]/(X^N + 1)$. Given a modulus q, $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$ is the residue ring of $\mathcal{R}$ modulo q. An element $a \in \mathbb{R}[X]/(X^N + 1)$ represented by $a(X) = \sum_{j=0}^{N-1} a_j X^j$ of degree $< N$ will be identified with its coefficient vector $(a_0, \ldots, a_{N-1}) \in \mathbb{R}^N$. We use the notation $\|a\|_\infty$ to denote the usual ℓ_∞-norm of a. Let $M = 2N$ and $\mathbb{Z}_M^* = \{x \in \mathbb{Z}_M \mid \gcd(x, M) = 1\}$. Then, the canonical embedding τ for $a \in \mathbb{R}[X]/(X^N + 1)$ is the vector of evaluation value of a at each primitive M-th roots of unity. That is, each component of the vector $\tau(a)$ is of the form $a(\zeta^j)$ where $\zeta = \exp(-2\pi i)$ is a primitive M-th root of unity and $j \in \mathbb{Z}_M^*$. We define the canonical embedding norm $\|a\|_\infty^{\mathsf{can}}$ of a as the ℓ_∞-norm of $\tau(a)$, $\|a\|_\infty^{\mathsf{can}} = \|\tau(a)\|_\infty$.

For two distributions X and Y over a finite domain Ω, the *statistical distance* between X and Y is defined by $\Delta(X, Y) := \frac{1}{2}\sum_{\omega \in \Omega} |X(\omega) - Y(\omega)|$. Let X and Y be distribution ensembles parameterized by the security parameter λ, we say that X and Y are *statistically indistinguishable*, denoted by $X \overset{stat}{\approx} Y$, if the distance $\Delta(X, Y)$ is negligible.

Lemma 1 (Smudging Lemma [24], [32]): For the security parameter λ, let $B_1 = B_1(\lambda)$ and $B_2 = B_2(\lambda)$ be positive integers and let $e_1 \in [-B_1, B_1]$ be a fixed integer. Let $e_2 \leftarrow [-B_2, B_2]$ 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 is negligible in λ.

For positive integers B_1 and B_2, let $e_1 \in \mathcal{R}$ be a fixed polynomial with $\|e_1\|_\infty \le B_1$, and $e_2 \in \mathcal{R}$ be another polynomial whose coefficients are chosen independently and uniformly at random from $[-B_2, B_2]$. Then we can show that the distribution of e_2 and that of $e_2 + e_1$ are statistically indistinguishable as long as B_1/B_2 is negligible because each coefficient was chosen independently.[1]

Throughout the paper, we denote by $\chi_{\mathsf{sm}}(B_2)$ the distribution of polynomials in $\mathcal{R}$ whose coefficients are chosen independently and uniformly at random from $[-B_2, B_2]$ for a positive integer B_2.

Definition 1 (Homomorphic Encryption): A (leveled) homomorphic encryption consists of a tuple of algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$:

- *$\mathsf{KeyGen}(1^\lambda, 1^L) \to (\mathsf{pk}, \mathsf{evk}, \mathsf{sk})$: Key generation algorithm KeyGen takes a security parameter λ and a depth bound L, and outputs public encryption key pk, public evaluation key evk, and secret decryption key sk.*
- *$\mathsf{Enc}(\mathsf{pk}, m) \to \mathbf{c}$: Encryption algorithm Enc takes pk and a message m, and outputs a ciphertext $\mathbf{c}$.*
- *$\mathsf{Eval}(\mathsf{evk}, f, \mathbf{c}_1, \cdots, \mathbf{c}_k) \to \mathbf{c}^*$: Evaluation algorithm Eval takes a k-input function f that can be computed by a circuit of depth at most L, and outputs a new ciphertext $\mathbf{c}^*$ that is an encryption of $f(m_1, \cdots, m_k)$ where $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i)$ for $i = 1, \cdots, k$.*
- *$\mathsf{Dec}(\mathsf{sk}, \mathbf{c}) \to m$ or $\perp$: Decryption algorithm Dec takes sk and a ciphertext $\mathbf{c}$, and outputs a message m or $\perp$.*

Definition 2 (Threshold Homomorphic Encryption): Let $P = \{P_1, \cdots, P_n\}$ be a set of parties and $t \le n$ be a threshold number. A (t, n)-threshold homomorphic encryption (Threshold HE) scheme is a homomorphic encryption scheme, with the difference that KeyGen and Dec are now n-party protocols instead of algorithms, and Dec works if at least t parties participate in the protocol.

- *$\mathsf{THE.KeyGen}(1^\lambda, 1^L; \mathsf{pp}) \to (\mathsf{pk}, \mathsf{evk}, \{\mathsf{sk}_i\}_{i\in[n]})$: Given the security parameter λ and a depth bound L, each party P_i outputs a common encryption key pk, a common evaluation key evk, and a secret key share sk_i of the implicitly defined secret key sk under a public parameter pp after a key generation protocol.*
- *$\mathsf{THE.Enc}(\mathsf{pk}, m) \to \mathbf{c}$: Given a public key pk and a message m, the encryption algorithm outputs a ciphertext $\mathbf{c}$.*
- *$\mathsf{THE.Eval}(\mathsf{evk}, f, \{\mathbf{c}_i\}_{i\in[k]}) \to \mathbf{c}^*$: Given an evaluation key evk, a function f with a depth smaller than some parameter L, and ciphertexts $\mathbf{c}_i$, the evaluation algorithm outputs a new ciphertext $\mathbf{c}^*$.*
- *$\mathsf{THE.Dec}(\{\mathsf{sk}_i\}_{i\in I}, \mathbf{c}) \to m$ or $\perp$: For an index set $I \subseteq [n]$ with $|I| \ge t$, each party P_i, $i \in I$, holds an individual secret shares sk_i of the secret key sk. After a decryption protocol, one who wants to decrypt a ciphertext $\mathbf{c}$ gets a result if more than t parties give back their partial decryptions.*

Moreover, we say that a (t, n)-threshold FHE is decentralized if all the parties have the same level of information and play the same role in the KeyGen and Dec protocols.

Definition 3 (CKKS scheme): We recall a leveled homomorphic encryption scheme [15] that supports fixed point arithmetic of real numbers.

- *$\mathsf{Setup}(1^\lambda, 1^L)$: Given the security parameter λ and a depth bound L, do the following:*
  - *Choose a power-of-two integer $N > 0$, a base $p > 0$, a modulus q_0, and a special modulus P, and let $q_L = q_0 \cdot p^L$ so that N and $P \cdot q_L$ satisfies the security level λ.*
  - *Choose a secret key distribution χ_{sk}, an error distribution χ_{err}, and a random distribution χ_{Enc} for encryption.*
  - *Output public parameters*

$$\mathsf{pp} = (1^\lambda, 1^L, N, p, q_0, q_L, P, \chi_{\mathsf{sk}}, \chi_{\mathsf{err}}, \chi_{\mathsf{Enc}}).$$

- *KeyGen:*
  - *Sample $s \leftarrow \chi_{\mathsf{sk}}$, $a \leftarrow \mathcal{R}_{q_L}$ and $e \leftarrow \chi_{\mathsf{err}}$. Set the secret key as $\mathsf{sk} \leftarrow (1, s)$ and the public key as $\mathsf{pk} \leftarrow (b, a) \in \mathcal{R}_{q_L}^2$ where*

$$b \leftarrow -a \cdot s + e \pmod{q_L}.$$

  - *Sample $a' \leftarrow \mathcal{R}_{P \cdot q_L}$ and $e' \leftarrow \chi_{\mathsf{err}}$. Set the evaluation key as $\mathsf{evk} \leftarrow (b', a') \in \mathcal{R}_{P\cdot q_L}^2$ where*

$$b' \leftarrow -a' \cdot s + e' + P \cdot s^2 \pmod{P \cdot q_L}.$$

- *$\mathsf{Enc}(\mathsf{pk}, m)$: Sample $v \leftarrow \chi_{\mathsf{Enc}}$ and $e_0, e_1 \leftarrow \chi_{\mathsf{err}}$. Output $\mathbf{c} \leftarrow v \cdot \mathsf{pk} + (m + e_0, e_1) \pmod{q_L}$.*
- *$\mathsf{Add}(\mathbf{c}, \mathbf{c}')$: Output $\mathbf{c}_{\mathsf{add}} \leftarrow \mathbf{c} + \mathbf{c}'$.*
- *$\mathsf{Mult}(\mathsf{evk}, \mathbf{c}, \mathbf{c}')$: For $\mathbf{c} = (c_0, c_1)$, $\mathbf{c}' = (c_0', c_1') \in \mathcal{R}_{q_\ell}^2$, let $(d_0, d_1, d_2) = (c_0c_0', c_0c_1' + c_0'c_1, c_1c_1') \pmod{q_\ell}$. Output $\mathbf{c}_{\mathsf{mult}} \leftarrow (d_0, d_1) + \lfloor P^{-1} \cdot d_2 \cdot \mathsf{evk} \rceil \pmod{q_\ell}$*

[1]If $\|e_1\|_\infty$ is less than B_1 with high probability, we can say that the two distributions are statistically indistinguishable with high probability.

- *Rescale*$_{\ell\to\ell'}(\mathbf{c})$*: Output* $\mathbf{c}' \leftarrow \lfloor \frac{q_{\ell'}}{q_\ell}\mathbf{c}\rceil \pmod{q_{\ell'}}$.
- *Dec*$(sk, \mathbf{c})$*: Output* $\langle \mathbf{c}, sk\rangle \pmod{q_\ell}$.

Note that if the output $\langle \mathbf{c}, \mathsf{sk}\rangle \pmod{q_\ell}$ of Dec is sufficiently smaller than q_0 for all valid ciphertext $\mathbf{c}$, we can take q_0 instead of q_ℓ in Dec algorithm. Thus, we assume the message space is limited to be smaller than a proper bound that $\langle \mathbf{c}, \mathsf{sk}\rangle \pmod{q_\ell} = \langle \mathbf{c}, \mathsf{sk}\rangle \pmod{q_0}$ holds. For example, we can restrict the message space for $\langle \mathbf{c}, \mathsf{sk}\rangle$ to be smaller than $q_0/4$.

III. THRESHOLD HOMOMORPHIC ENCRYPTION WITH NEW EVALUATION KEY GENERATION

Our threshold HE scheme starts from the (n, n)-threshold HE scheme of [24]. After n parties $P_1, \cdots, P_n$ have generated its secret key share $\mathsf{sk}_i = (1, s_i)$ and the corresponding public key share pk_i, and parties collaboratively generate a common encryption key pk and a common evaluation key evk following the protocol in [24]. More specifically, after gathering all the public key shares $\{\mathsf{pk}_1, \cdots, \mathsf{pk}_n\}$, P_i computes the common public key pk and then broadcasts encrypted secret key share under pk to other parties. Then from encrypted secret key shares, P_i generates the common evaluation key evk which can be seen as an encryption of $(\sum_{i=1}^n s_i)^2$ under pk.

In this article, instead of pk, each party P_i uses its secret key sk_i to encrypt sk_i for evaluation key generation. That is, the party P_i holding sk_i broadcasts $\mathsf{evk}_{i,0} \leftarrow (-a' \cdot s_i + e_i + s_i, a')$ for some small error e_i which is an encryption of s_i under sk_i. From $\{\mathsf{evk}_{1,0}, \cdots, \mathsf{evk}_{n,0}\}$, P_i can compute an encryption evk_0 of $s := \sum_{i=1}^n s_i$ under sk by adding the first components of the given other $\mathsf{evk}_{k,0}$'s:

$$\begin{aligned}\mathsf{evk}_0 &= \left(\sum_{i=1}^n \left(-a' \cdot s_i + e_i + s_i\right), a'\right)\\ &= \left(-a' \cdot s + e + s, a'\right),\end{aligned}$$

where $e = \sum_{i=1}^n e_i$. Then, P_i can compute an encryption of $s_i \cdot s$ under $\mathsf{sk} = (1, \sum_{i=1}^n s_i)$ by multiplying $s_i \cdot \mathsf{evk}_0$:

$$\begin{aligned}s_i \cdot \mathsf{evk}_0 &= (s_i \cdot (-a' \cdot s + e + s), s_i \cdot a')\\ &= \left(-s_i \cdot (a' \cdot s) + s_i \cdot e + s_i \cdot s, s_i \cdot a'\right).\end{aligned}$$

Finally a new encryption of $s^2 = (\sum_{i=1}^n s_i)^2$ can be computed by adding them for all i:

$$\begin{aligned}\sum_{i=1}^n s_i \cdot \mathsf{evk}_0 &= \left((\sum_{i=1}^n s_i) \cdot (-a' \cdot s + e + s), \sum_{i=1}^n s_i \cdot a'\right)\\ &= \left(-(a' \cdot s) \cdot s + e \cdot s + s^2, a' \cdot s\right).\end{aligned}$$

The above outlines our main construction, but we have to add appropriate smudging errors when broadcasting small polynomials to hide secret elements properly.

A. CONSTRUCTION

In this section we describe our (n, n)-threshold HE construction based on CKKS scheme [15] to fix an idea. Note that it can be easily generalized by using other HE scheme with key-homomorphic property such as [4].

- $\mathsf{Setup}(1^\lambda, 1^L, 1^n)$: Given the security parameter λ, a depth bound L and the number of parties n, do the following:
  - Choose a power-of-two integer $N > 0$, a base $p > 0$, a modulus q_0, and a special modulus P, and let $q_L = q_0 \cdot p^L$ such that N and $P \cdot q_L$ satisfies the security level λ.
  - Choose a secret key distribution χ_{sk}, an error distribution χ_{err}, a random distribution χ_{Enc} for encryption, and a set B of smudging error bounds $B = \{B_{\mathsf{sm}}^{\mathsf{Eval}}, B_{\mathsf{sm}}^{\mathsf{Enc}}, B_{\mathsf{sm}}^{\mathsf{Dec}}\}$.
  - Choose common random elements $a, a' \leftarrow \mathcal{R}_{P\cdot q_L}$.
  - Output public parameters include
$$\mathsf{pp} = (N, p, q_0, q_L, P, \chi_{\mathsf{sk}}, \chi_{\mathsf{err}}, \chi_{\mathsf{Enc}}, B).$$
- $\mathsf{THE.KeyGen}$: Our key generation protocol KeyGen is a two-round, n-party protocol among $P_1, \cdots, P_n$.
  - **Round I.** For $i \in [n]$, each party P_i executes the following.
    1) Sample $s_i \leftarrow \chi_{\mathsf{sk}}$ and $e_i \leftarrow \chi_{\mathsf{err}}$, and compute $b_i \leftarrow -a \cdot s_i + e_i \pmod{P \cdot q_L}$.
    2) Sample $e'_{i,0} \leftarrow \chi_{\mathsf{err}}$, and compute $b'_{i,0} \leftarrow -a' \cdot s_i + e'_{i,0} + P \cdot s_i \pmod{P \cdot q_L}$.
    3) Set its secret key share as $\mathsf{sk}_i \leftarrow (1, s_i)$ and broadcast $b_i, b'_{i,0}$ to other parties.
  - **Round II.** Upon receiving $\{b_j, b'_{j,0}\}_{j\neq i}$, P_i sets the common public key as $\mathsf{pk} \leftarrow (\sum_{j=1}^n b_j, a) \in \mathcal{R}^2_{P\cdot q_L}$.
    1) Compute $b'_0 \leftarrow \sum_{i=1}^n b'_{i,0} \pmod{P \cdot q_L}$.
    2) Sample smudging errors $e^{\mathsf{sm}}_{i,0}, e^{\mathsf{sm}}_{i,1} \leftarrow \chi_{\mathsf{sm}}(B^{\mathsf{Eval}}_{\mathsf{sm}})$ and compute an intermediate evaluation key $\mathsf{evk}_i \leftarrow (s_i \cdot b'_0 + e^{\mathsf{sm}}_{i,0}, s_i \cdot a' + e^{\mathsf{sm}}_{i,1}) \in \mathcal{R}^2_{P\cdot q_L}$.
    3) Broadcast evk_i to other parties.

    Upon receiving $\{\mathsf{evk}_j\}_{j\neq i}$, P_i sets the common evaluation key as $\mathsf{evk} \leftarrow \sum_{j=1}^n \mathsf{evk}_j \in \mathcal{R}^2_{P\cdot q_L}$.
- $\mathsf{THE.Enc}(\mathsf{pk}, m)$:
  1) Sample a random element $v \leftarrow \chi_{\mathsf{Enc}}$ and smudging errors $e^{\mathsf{sm}}_0, e^{\mathsf{sm}}_1 \leftarrow \chi_{\mathsf{sm}}(B^{\mathsf{Enc}}_{\mathsf{sm}})$.
  2) Output $\mathbf{c} \leftarrow v \cdot \mathsf{pk} + (m + e^{\mathsf{sm}}_0, e^{\mathsf{sm}}_1) \pmod{q_L}$.
- $\mathsf{THE.Add}(\mathbf{c}, \mathbf{c}')$: Output $\mathbf{c}_{\mathsf{add}} \leftarrow \mathbf{c} + \mathbf{c}'$.
- $\mathsf{THE.Mult}(\mathsf{evk}, \mathbf{c}, \mathbf{c}')$: For $\mathbf{c} = (c_0, c_1)$, $\mathbf{c}' = (c'_0, c'_1) \in \mathcal{R}^2_{q_\ell}$, let $(d_0, d_1, d_2) = (c_0c'_0, c_0c'_1 + c'_0c_1, c_1c'_1) \pmod{q_\ell}$. Output $\mathbf{c}_{\mathsf{mult}} \leftarrow (d_0, d_1) + \lfloor P^{-1} \cdot d_2 \cdot \mathsf{evk}\rceil \pmod{q_\ell}$
- $\mathsf{THE.Rescale}_{\ell\to\ell'}(\mathbf{c})$: Output $\mathbf{c}' \leftarrow \lfloor \frac{q_{\ell'}}{q_\ell}\mathbf{c}\rceil \pmod{q_{\ell'}}$.
- $\mathsf{THE.Dec}(\{\mathsf{sk}_1, \cdots, \mathsf{sk}_n\}, \mathbf{c})$: Our threshold decryption protocol Dec is an one round, n-party protocol among $P_1, \cdots, P_n$.
  - **Round I.** Given the ciphertext $\mathbf{c} = (c_0, c_1)$ at level ℓ, for $i \in [n]$ each party P_i executes the following.
    1) Sample a smudging error $e^{\mathsf{sm}}_i \leftarrow \chi_{\mathsf{sm}}(B^{\mathsf{Dec}}_{\mathsf{sm}})$.
    2) Compute a partial decryption $p_i \leftarrow c_1 \cdot s_i + e^{\mathsf{sm}}_i \pmod{q_i}$, and broadcast it to other parties.

Upon receiving $\{p_j\}_{j\neq i}$, P_i computes the final decryption

$$c_0 + \sum_{i=1}^{n} p_i \pmod{q_\ell}.$$

B. SECURITY OF JOINT KEYS

In [24], they proved the security of joint keys by showing the ciphertexts under the joint key is distributed uniformly random after adding *smudging errors*. Following their idea, we can also show that the joint key security of our construction: Assume a public encryption key (b, a) is constructed honestly and an attacker can then adaptively choose $(b' = -a \cdot s' + e', a)$. Then, it suffices to show that the attacker should not be able to distinguish some ciphertexts under the combined key $(b + b', a)$ from uniformly random ones. Let $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ denote the CKKS scheme [15] which satisfies key-homomorphic properties and let us consider an experiment $\mathsf{JointKey}_{\mathcal{A}}(\mathsf{pp}, B_1, B_2)$ between an attacker $\mathcal{A}$ and a challenger defined as:

1) Challenger chooses $s \leftarrow \chi_{\mathsf{sk}}$ and $e \leftarrow \chi_{\mathsf{err}}$, and gives $(b = -a \cdot s + e, a)$ to $\mathcal{A}$.
2) $\mathcal{A}$ adaptively chooses $s' \leftarrow \chi_{\mathsf{sk}}$, $e' \leftarrow \chi_{\mathsf{err}}$ and a random message $m \in \mathcal{R}$. It gives $(b' = -a \cdot s' + e', s', e', m)$ to the challenger.
3) The challenger sets the public encryption key $\mathsf{pk} = (b + b', a)$ and chooses a random bit $\beta \leftarrow \{0, 1\}$.
   - If $\beta = 0$, it chooses $\mathbf{c}^* = (c_0^*, c_1^*) \leftarrow \mathcal{R}_{q_L}^2$ uniformly.
   - If $\beta = 1$, it generates a valid ciphertext $\mathbf{c}^* = (c_0, c_1^*) \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$, $e^* \leftarrow \chi_{\mathsf{sm}}(B_2)$ and set $c_0^* = c_0 + e^*$.
4) $\mathcal{A}$ gets $\mathbf{c}^* = (c_0^*, c_1^*)$ and outputs a bit β'.

The output of the experiment is 1 if $\beta = \beta'$ ($\mathcal{A}$ wins), and 0 otherwise ($\mathcal{A}$ loses).

Note that if χ_{err} is a discrete Gaussian distribution, then there exists a high probability bound B_1 which satisfies $\|e\|_\infty \leq B_1$ for $e \leftarrow \chi_{\mathsf{err}}$ with high probability.

Lemma 2: Assume the CKKS scheme has pseudorandom ciphertexts and χ_{err} is a discrete Gaussian distribution with high probability bound B_1. If $B_1/B_2 = \mathsf{negl}(\lambda)$, then for any probabilistic polynomial time adversary $\mathcal{A}$, we have

$$\left|\Pr[\mathit{JointKey}_{\mathcal{A}}(\mathit{params}) = 1] - \frac{1}{2}\right| = \mathsf{negl}(\lambda).$$

Proof: We will construct an adversary $\mathcal{A}'$ distinguishing ciphertexts from uniform distribution using the given adversary $\mathcal{A}$. Assume a challenger for pseudorandom ciphertexts property of CKKS scheme gives $\mathsf{pk} = (b = -a \cdot s + e, a)$ and a challenge $(\tilde{c_0}, \tilde{c_1})$ to the adversary $\mathcal{A}'$. Then, $\mathcal{A}'$ gives it to $\mathcal{A}$ and get back $(b' = -a \cdot s' + e', s', e', m)$ from $\mathcal{A}$. Lastly, $\mathcal{A}'$ sets $(c_0^*, c_1^*) = (\tilde{c_0}, \tilde{c_1}) - (\tilde{c_1} \cdot s', 0)$, sends it to $\mathcal{A}$, and outputs the bit β' obtained from $\mathcal{A}$.

Note that (c_0^*, c_1^*) is uniformly random if $(\tilde{c_0}, \tilde{c_1})$ so is. On the other hand, if $(\tilde{c_0}, \tilde{c_1})$ is of the form $(-\tilde{c_1} \cdot s + \tilde{e}, \tilde{c_1})$, then

$$(c_0^*, c_1^*) = (\tilde{c_0}, \tilde{c_1}) - (\tilde{c_1} \cdot s', 0) = (-\tilde{c_1} \cdot (s + s') + \tilde{e}, \tilde{c_1})$$

which is an instance of (R)LWE problem. Thus, $\mathcal{A}'$ breaks the pseudorandomness of ciphertexts with the same advantage as $\mathcal{A}$ by smudging lemma. □

C. NOISE ESTIMATION

In this section, we estimate the growth of errors after various operations assuming specific distributions for χ_{sk}, χ_{err} and χ_{Enc}, following the heuristic approach in [11], [15], [33]. One can set specific parameters for our threshold homomorphic encryption schemes using these estimated errors. To do this, we first examine each distribution and give high probability upper bounds for the size of error polynomials. Recall that we measure the size of a polynomial $a \in \mathcal{R}$ by the canonical embedding norm

$$\|a\|_\infty^{\mathsf{can}} = \|\tau(a)\|_\infty = \max_{j \in \mathbb{Z}_M^*} |a(\zeta^j)|$$

where τ is the canonical embedding into $\mathbb{C}^N$. For $j \in \mathbb{Z}_M^*$, each component $a(\zeta^j)$ of $\tau(a)$ is the inner product of the coefficient vector of a and a vector $(1, \zeta^j, \cdots, \zeta^{j(N-1)})$, having the Euclidean norm $\sqrt{N}$, where ζ is a primitive M-th root of unity. If each coefficient of a is sampled independently from the identical distribution χ with the variance δ^2, then we may assume that the random variable $a(\zeta^j)$ is distributed similarly to a Gaussian random variable over the complex plane having the variance $\delta^2 N$. If we assume $a(\zeta^j)$ is from a Gaussian distribution with the variance $\delta^2 N$, we can use a high probability upper bound $6\sqrt{\delta^2 N} = 6\delta\sqrt{N}$ for $\|a\|_\infty^{\mathsf{can}}$. Furthermore, for two polynomials $a, b \in \mathcal{R}$, since we have $(a \cdot b)(\zeta) = a(\zeta)b(\zeta)$, we use a high probability upper bound $\|a \cdot b\|_\infty^{\mathsf{can}} \leq 16\delta_1\delta_2$ when random variables $a(\zeta^j)$ and $b(\zeta^j)$ have variance δ_1^2 and δ_2^2, respectively. We will also use the fact that the random variable $c(\zeta^j)$ has variance δ_3^2, then it holds that $\|a \cdot b \cdot c\|_\infty^{\mathsf{can}} \leq 32\delta_1\delta_2\delta_3$ with high probability. See [11], [15], [33] for more details.

To precisely estimate the size of errors, we fix some specific distributions for various polynomials. Let us start from the error distribution χ_{err}. For the error distribution, many homomorphic encryption schemes [15] based on RLWE use the distribution $\mathcal{DG}(\sigma^2)$ which draws a polynomial by sampling each coefficient independently from the discrete Gaussian distribution with the variance σ^2. Following the above argument, we use a high probability upper bound $\|e\|_\infty^{\mathsf{can}} \leq 6\sigma\sqrt{N}$ for $e \leftarrow \mathcal{DG}(\sigma^2)$. Let denote by $\mathcal{ZO}$ the distribution over $\mathcal{R}$ which will be used for χ_{sk} and χ_{Enc} where each coefficient is sampled independently from $\{0, \pm 1\}$, with probability $1/4$ for each of -1 and $+1$, and probability being zero $1/2$. Again, since the coefficients are sampled independently from an identical distribution with the variance $1/2$, we think $a \leftarrow \mathcal{ZO}$ has an upper bound $\|a\|_\infty^{\mathsf{can}} \leq 3\sqrt{2N}$ with high probability. In addition, we consider a distribution $\chi_{\mathsf{sm}}(B)$ for smudging errors. For a given $B > 0$, the distribution $\chi_{\mathsf{sm}}(B)$ draws each coefficient from the uniform distribution $[-B, B]$ with the variance $B^2/3$, and hence we use a high-probability upper bound $\|a\|_\infty^{\mathsf{can}} \leq 2\sqrt{3N}B$ for $a \leftarrow \chi_{\mathsf{sm}}(B)$.

TABLE 1. Assumption for distributions: If a polynomial $a \in \mathcal{R}$ is sampled from a distribution χ, then we can compute the variance of $a(\zeta^j)$ to give a high-probability upper bound for $\|a\|_\infty^{\mathsf{can}}$.

Distribution χ	Assumption for χ	Variance of $a(\zeta^j)$
χ_{err}	$\mathcal{DG}(\sigma^2)$	$\sigma^2 N$
χ_{Enc}	$\mathcal{ZO}(0.5)$	$N/2$
χ_{sk}	$\mathcal{ZO}(0.5)$	$N/2$
$\chi_{\mathsf{sm}}(B)$	$-$	$B^2N/3$

For a common public key $\mathsf{pk} = (b, a)$ where $b = \sum_{i=1}^n b_i = a \cdot \left(\sum_{i=1}^n s_i\right) + \left(\sum_{i=1}^n e_i\right)$ generated by $\mathsf{THE.KeyGen}$ protocol, let denote the corresponding secret key $\mathsf{sk} = (1, s)$ where $s = \sum_{i=1}^n s_i$ which is never computed by any party. Note that for two polynomials $a, b \in \mathcal{R}$ chosen independently from the distributions with the variance δ_1^2 and δ_2^2, respectively, we can conclude that the random variable $(a+b)(\zeta) = a(\zeta)+b(\zeta)$ is similarly distributed to a Gaussian distribution with the variance $\delta_1^2 + \delta_2^2$.

Lemma 3 (Public encryption key): Let pk be a common public encryption key generated by $\mathsf{THE.KeyGen}$ protocol, and sk be the corresponding secret key. Then it holds that

$$\|\langle \mathsf{pk}, \mathsf{sk}\rangle \pmod{P \cdot q_L}\|_\infty^{can} \leq 6\sigma n^{\frac{1}{2}} N^{\frac{1}{2}}.$$

Proof: In $\mathsf{THE.KeyGen}$ protocol, each party generates its secret key share $\mathsf{sk}_i = (1, s_i)$ and compute the individual public key $\mathsf{pk}_i = (b_i, a)$ where $b_i = -a \cdot s_i + e_i \pmod{P \cdot q_L}$. Then the common public key $\mathsf{pk} = (b, a)$ is computed by $b = \sum_{i=1}^n b_i = -a \cdot \sum_{i=1}^n s_i + \sum_{i=1}^n e_i \pmod{P \cdot q_L}$, and we have $\langle \mathsf{pk}, \mathsf{sk}\rangle = b + a \cdot s = \sum_{i=1}^n e_i \pmod{P \cdot q_L}$. Since the noise term e_i is sampled independently from $\mathcal{DG}(\sigma^2)$, $e = \sum_{i=1}^n e_i$ can be seen as a sample from the distribution $\mathcal{DG}(\sigma^2) + \cdots + \mathcal{DG}(\sigma^2) \overset{stat}{\approx} \mathcal{DG}(n\sigma^2)$ with the variance $n\sigma^2$. Thus, we have $\|e\|_\infty^{\mathsf{can}} \leq 6\sqrt{\sigma^2\ nN} = 6\sigma n^{\frac{1}{2}} N^{\frac{1}{2}}$ as desired. □

Lemma 4 (Evaluation key): Let evk be an evaluation key generated by $\mathsf{THE.KeyGen}$ protocol, then it holds that

$$\|\langle \mathsf{evk}, \mathsf{sk}\rangle - P \cdot s^2 \pmod{P \cdot q_L}\|_\infty^{can} \leq B_{multkey}$$

where

$$B_{multKey} = 8\sqrt{2}\sigma nN + 2\sqrt{3}B_{sm}^{Eval}\left(n^{\frac{1}{2}}N^{\frac{1}{2}} + \frac{4\sqrt{2}}{3}nN\right).$$

Proof: In this proof, all equations are defined modulo $P \cdot q_L$. To generate evk, each party generates $b'_{i,0} = -a' \cdot s_i + e'_{i,0} + P \cdot s_i$ and broadcasts it to compute $\mathsf{evk}_i = (s_i \cdot b'_0 + e_{i,0}^{\mathsf{sm}}, s_i \cdot a' + e_{i,1}^{\mathsf{sm}})$ where $b'_0 = \sum_{i=1}^n b'_{i,0}$. For $e'_{i,0} \leftarrow \mathcal{DG}(\sigma^2)$, $e_{i,0}^{\mathsf{sm}}, e_{i,1}^{\mathsf{sm}} \leftarrow \chi_{\mathsf{sm}}(B_{\mathsf{sm}}^{\mathsf{Eval}})$, let $e' = \sum_{i=1}^n e'_{i,0}$, $e_0^{\mathsf{sm}} = \sum_{i=1}^n e_{i,0}^{\mathsf{sm}}$ and $e_1^{\mathsf{sm}} = \sum_{i=1}^n e_{i,1}^{\mathsf{sm}}$. Then we have $\mathsf{Var}(e'(\zeta)) = \sum_{i=1}^n \mathsf{Var}(e'_{i,0}(\zeta)) = n\sigma^2 N$, and similarly $\mathsf{Var}(e_0^{\mathsf{sm}}(\zeta)) = \mathsf{Var}(e_1^{\mathsf{sm}}(\zeta)) = n(B_{\mathsf{sm}}^{\mathsf{Eval}})^2 N/3$ and $\mathsf{Var}(s(\zeta)) = nN/2$. Therefore, it holds that

$$\begin{aligned}
&\|\langle \mathsf{evk}, \mathsf{sk}\rangle - P \cdot s^2\|_\infty^{\mathsf{can}} \\
&\leq \|e' \cdot s + e_0^{\mathsf{sm}} + e_1^{\mathsf{sm}} \cdot s\|_\infty^{\mathsf{can}} \\
&\leq 16\sigma\sqrt{nN}\sqrt{\frac{nN}{2}} + 6\sqrt{nN/3}B_{\mathsf{sm}}^{\mathsf{Eval}} + 16\sqrt{\frac{nN}{3}}B_{\mathsf{sm}}^{\mathsf{Eval}}\sqrt{\frac{nN}{2}} \\
&= 8\sqrt{2}\sigma nN + 2\sqrt{3}n^{\frac{1}{2}}N^{\frac{1}{2}}B_{\mathsf{sm}}^{\mathsf{Eval}} + \frac{8}{3}\sqrt{6}nNB_{\mathsf{sm}}^{\mathsf{Eval}}.
\end{aligned}$$

□

Lemma 5 (Encryption): Let $\mathbf{c} \leftarrow \mathsf{THE.Enc}(\mathsf{pk}, m)$ be a ciphertext of m under pk. Then, it holds that

$$\|\langle \mathbf{c}, \mathsf{sk}\rangle - m \pmod{q_L}\|_\infty^{can} \leq B_{clean}$$

where $B_{clean} = 8\sqrt{2}\sigma n^{\frac{1}{2}}N + 2\sqrt{3}B_{\mathsf{sm}}^{\mathsf{Enc}}(N^{\frac{1}{2}} + \frac{4\sqrt{2}}{3}n^{\frac{1}{2}}N)$.

Proof: The ciphertext $\mathbf{c}$ is computed by $v \cdot \mathsf{pk} + (m + e_0^{\mathsf{sm}}, e_1^{\mathsf{sm}}) \pmod{q_L}$ where $v \leftarrow \mathcal{ZO}$ and $e_0^{\mathsf{sm}}, e_1^{\mathsf{sm}} \leftarrow \chi_{\mathsf{sm}}(B_{\mathsf{sm}}^{\mathsf{Enc}})$. Thus, we have $\langle \mathbf{c}, \mathsf{sk}\rangle = v \cdot \langle \mathsf{pk}, \mathsf{sk}\rangle + m + e_0^{\mathsf{sm}} + e_1^{\mathsf{sm}} \cdot s = v \cdot e + m + e_0^{\mathsf{sm}} + e_1^{\mathsf{sm}} \cdot s \pmod{q_L}$, and

$$\begin{aligned}
&\|\langle \mathbf{c}, \mathsf{sk}\rangle - m \pmod{q_L}\|_\infty^{\mathsf{can}} \\
&\leq \|v \cdot e + e_0^{\mathsf{sm}} + e_1^{\mathsf{sm}} \cdot s\|_\infty^{\mathsf{can}} \\
&\leq 16\sqrt{\frac{N}{2}}\sigma\sqrt{nN} + 2\sqrt{3N}B_{\mathsf{sm}}^{\mathsf{Enc}} + 16B_{\mathsf{sm}}^{\mathsf{Enc}}\sqrt{\frac{N}{3}}\sqrt{\frac{nN}{2}} \\
&= 8\sqrt{2}\sigma n^{\frac{1}{2}}N + 2\sqrt{3}N^{\frac{1}{2}}B_{\mathsf{sm}}^{\mathsf{Enc}} + \frac{8\sqrt{6}}{3}n^{\frac{1}{2}}NB_{\mathsf{sm}}^{\mathsf{Enc}}.
\end{aligned}$$

□

Lemma 6 (Rescaling): Let $\mathbf{c}$ be a level-ℓ ciphertext of m under pk satisfying

$$\|\langle \mathbf{c}, \mathsf{sk}\rangle - m \pmod{q_\ell}\|_\infty^{can} \leq B$$

for some $B > 0$. Then, for $\mathbf{c}' \leftarrow \mathsf{Rescale}_{\ell \to \ell'}(\mathbf{c})$, it holds that

$$\|\langle \mathbf{c}', \mathsf{sk}\rangle - p^{\ell' - \ell} \cdot m\|_\infty^{can} \leq p^{\ell' - \ell} \cdot B + B_{\mathsf{scale}}$$

where $B_{\mathsf{scale}} = \sqrt{3}N^{\frac{1}{2}} + \frac{4\sqrt{6}}{3}n^{\frac{1}{2}}N$.

Proof: Let $\langle \mathbf{c}, \mathsf{sk}\rangle = m + e \pmod{q_\ell}$ where $\|e\|_\infty^{\mathsf{can}} \leq B$. Then for $\mathbf{c}' \leftarrow \mathsf{Rescale}_{\ell \to \ell'}(\mathbf{c})$ it is satisfied that $\langle \mathbf{c}', \mathsf{sk}\rangle = \frac{q_{\ell'}}{q_\ell}(m + e) + e_{scale} \pmod{q_{\ell'}}$ for the rounding error vector $(\tau_0, \tau_1) = \mathbf{c}' - \frac{q_{\ell'}}{q_\ell}\mathbf{c}$ and $e_{scale} = \tau_0 + \tau_1 \cdot s$. Note that our s is a summation of s_i's where $s_i \leftarrow \mathcal{ZO}$, and hence we have $\mathsf{Var}(s(\zeta)) = nN/2$. Thus, if we assume the each coefficient of τ_0 and τ_1 is from a distribution with the variance $\sqrt{\frac{1}{12}}$, as in Lemma 2 of [15], we have

$$\begin{aligned}
\|e_{scale}\|_\infty^{\mathsf{can}} &\leq \|\tau_0\|_\infty^{\mathsf{can}} + \|\tau_1 \cdot s\|_\infty^{\mathsf{can}} \\
&\leq 6\sqrt{\frac{N}{12}} + 16\sqrt{\frac{N}{12}}\sqrt{\frac{nN}{2}} \leq \sqrt{3}N^{\frac{1}{2}} + \frac{4\sqrt{6}}{3}n^{\frac{1}{2}}N.
\end{aligned}$$

□

Lemma 7 (Addition/Multiplication): Let $\mathbf{c}_i$ be a level-ℓ ciphertext of m_i with $\|m_i\|_\infty^{can} \leq \nu_i$ under pk satisfying

$$\|\langle \mathbf{c}_i, \mathsf{sk}\rangle - m_i \pmod{q_\ell}\|_\infty^{can} \leq B_i$$

for some $B_i > 0$ and $i = 1, 2$. Then, for $\mathbf{c}_{add} \leftarrow \mathsf{add}(\mathbf{c}_1, \mathbf{c}_2)$ and $\mathbf{c}_{mult} \leftarrow \mathsf{mult}(\mathsf{evk}, \mathbf{c}_1, \mathbf{c}_2)$, it holds that

$$\|\langle \mathbf{c}_{add}, \mathsf{sk}\rangle - (m_1 + m_2) \pmod{q_\ell}\|_\infty^{can} \leq B_1 + B_2$$

and

$$\|\langle \mathbf{c}_{mult}, \mathsf{sk}\rangle - m_1 \cdot m_2 \pmod{q_\ell}\|_\infty^{can} \leq B_{mult}(\ell)$$

where $B_{mult}(\ell) = \nu_1 B_2 + \nu_2 B_1 + B_1 B_2 + P^{-1} \cdot q_\ell \cdot \sqrt{N/3} + B_{\mathsf{scale}}$.

Proof: Let $\mathbf{c}_i = (b_i, a_i)$ with $\langle \mathbf{c}_i, sk \rangle = m_i + e_i \pmod{q_\ell}$ for some polynomials m_i, e_i satisfying $\|e_i\|_\infty^{\mathsf{can}} \leq B_i$, $(i = 1, 2)$. Then, by Lemma 4, $\mathbf{c}_{\mathsf{mult}}$ contains three errors; an error $m_1 e_2 + m_2 e_1 + e_1 e_1$, an additional error $e'' = P^{-1} \cdot d_2\, e'$ and a rounding error bounded by B_{scale} where $d_2 = a_1 a_2$ and e' is the noise in the evaluation key evk. Assuming d_2 behaves as a uniform random variable on $\mathcal{R}_{q^\ell}$, $P\|e''\|_\infty^{\mathsf{can}}$ is bounded by $2\sqrt{Nq_\ell^2/12B_{multKey}} = B_{ks} \cdot q_\ell$. Thus, $\mathbf{c}_{\mathsf{mult}}$ is an encryption of $m_1 m_2$ with an error bounded by

$$\begin{aligned}\|m_1 e_2 + m_2 e_1 + e_1 e_2 + e''\|_\infty^{\mathsf{can}} + B_{scale} \\ \leq \nu_1 B_2 + \nu_2 B_1 + B_1 B_2 + P^{-1} \cdot q_\ell \cdot B_{ks} + B_{scale}.\end{aligned}$$

□

Lemma 8 (Decryption): Let $\mathbf{c} = (c_0, c_1)$ be a level-ℓ ciphertext of m under pk with $\|\langle \mathbf{c}, \mathsf{sk}\rangle - m\|_\infty^{\mathsf{can}} \leq B$, and let $m' \leftarrow \mathsf{THE.Dec}(\{\mathsf{sk}_1, \cdots, \mathsf{sk}_n\}, \mathbf{c})$ be the final decryption from the decryption protocol. Then, it holds that

$$\|m' - m \pmod{q_\ell}\|_\infty^{\mathsf{can}} \leq B + B_{Dec}$$

where $B_{Dec} = 2\sqrt{3}nN^{\frac{1}{2}}B_{\mathsf{sm}}^{\mathsf{Dec}}$.

Proof: The final decryption m' is computed by $m' \leftarrow c_0 + \sum_{i=1}^n p_i \pmod{q_0}$ and $p_i \leftarrow c_1 \cdot s_i + e_i^{\mathsf{sm}}$ where $e_i^{\mathsf{sm}} \leftarrow \chi_{\mathsf{sm}}(B_{\mathsf{sm}}^{\mathsf{Dec}})$. Thus, we have $m' = c_0 + c_1 \cdot \sum_{i=1}^n s_i + \sum_{i=1}^n e_i^{\mathsf{sm}}$.

Therefore, it holds that

$$\begin{aligned}\|m' - m \pmod{q_\ell}\|_\infty^{\mathsf{can}} \\ \leq \|m' - \langle \mathbf{c}, \mathsf{sk}\rangle\|_\infty^{\mathsf{can}} + \|\langle \mathbf{c}, \mathsf{sk}\rangle - m\|_\infty^{\mathsf{can}} \\ \leq \sum_{i=1}^n \|e_i^{\mathsf{sm}}\|_\infty^{\mathsf{can}} + B = B + 2\sqrt{3}nN^{\frac{1}{2}}B_{\mathsf{sm}}^{\mathsf{Dec}}.\end{aligned}$$

□

IV. SECRET KEY UPDATE

A. EXTENDED SHAMIR SECRET SHARING SCHEME

First, we revisit the Shamir secret sharing scheme to extend it for threshold FHE. Let q be a prime and assume we will share a secret $s \in \mathbb{Z}_q$ to n parties so that any more than t parties can reconstruct the secret s successfully. Then, the sharing algorithm of Shamir secret sharing scheme is following: a dealer randomly chooses $t-1$ elements $a_1, \ldots, a_{t-1}$ from $\mathbb{Z}_q$ and sets a polynomial

$$f(x) = a_0 + a_1\, x + \ldots + a_{t-1}x^{t-1} \in \mathbb{Z}_q[x]$$

whose constant term a_0 equals to the secret s. After computing each share $s_i = f(i)$, the dealer sends it to the party P_i for $i = 1, \ldots, n$. Later, if more than t parties $\{P_i\}_{i\in I}$ gather, the shared secret is reconstructed by computing

$$s' = \sum_{i\in I} \lambda_{I,i} s_i$$

where $\lambda_{I,i} = \prod_{j\in I\setminus\{i\}} \frac{j}{j-i}$ called as *Lagrange coefficients*. Note that $\lambda_{i,I}$ is well-defined since $\mathbb{Z}_q$ is a field and s' equals to s if the $\{s_i\}$'s are legitimate shares.

Definition 4 (Shamir secret sharing scheme on $\mathbb{Z}_q$): Let $s \in \mathbb{Z}_q$ be a secret to be shared with a prime q. A Shamir secret sharing scheme Shamir is a duple of algorithms $(\mathsf{Share}, \mathsf{Combine})$ as defined follows:

- *$\mathsf{Share}(s, t, n) \rightarrow (s_1, \ldots, s_n)$: The share algorithm takes a secret s and a threshold number t with n parties as input. After randomly choosing a polynomial $f(x) = a_0 + a_1\, x + \cdots + a_{t-1}x^{t-1} \in \mathbb{Z}_q[x]$ such that $a_0 = s$, it outputs the secret shares $s_i = f(i)$ for $i \in [n]$*
- *$\mathsf{Combine}(\{s_i\}_{i\in I}) \rightarrow s'$: The combine algorithm takes a set $\{s_i\}_{i\in I}$ of secret shares for some index set $I \subseteq [n]$. Then, it outputs a reconstructed secret $s' = \sum_{i\in I}\lambda_{I,i}s_i$ where $\lambda_{I,i} = \prod_{j\in I\setminus\{i\}}\frac{j}{j-i}$. For a convenience, we sometimes omit I in the notation $\lambda_{I,i}$.*

Now, let us consider the case we want to share a secret $s(X)$ lies in $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$ instead of $\mathbb{Z}_q$ We can extend the sharing algorithm by choosing $t-1$ elements $a_1(X), \ldots, a_{t-1}(X)$ in $\mathcal{R}_q$ and setting

$$f(x) = a_0(X) + a_1(X)x + \ldots + a_{t-1}(X)x^{t-1} \in \mathcal{R}_q[x]$$

whose constant term $a_0(X)$ equals to $s(X)$. Then, shares

$$\begin{aligned} s_i(X) &\text{ for } i = 1, \ldots, n \\ &= f(i) \\ &= a_0(X) + a_1(X)\cdot i + \cdots + a_{t-1}(X)\cdot i^{t-1} \\ &= \sum_{\ell=0}^{N-1} a_{0,\ell}X^\ell + \sum_{\ell=0}^{N-1} a_{1,\ell}X^\ell \cdot i + \cdots + \sum_{\ell=0}^{N-1} a_{t-1,\ell}X^\ell \cdot i^{t-1} \\ &= \sum_{\ell=0}^{N-1}(a_{0,\ell} + a_{1,\ell}\cdot i + \cdots + a_{t-1,\ell}\cdot i^{t-1})X^\ell \end{aligned}$$

can be considered as independent applications of the standard Shamir secret sharing for each coefficients of $s(X)$. Thus, if the coefficients of $s(X)$ are independently sampled, one can easily get the correctness and security of Shamir secret sharing over $\mathcal{R}_q$.

In some homomorphic encryption schemes, however, they use *sparse binary* (or *sparse ternary*) secret polynomials whose coefficients lies in $\{0, 1\}$ (or $\{0, \pm 1\}$) with a fixed Hamming distance h for their efficiency. For example, the HEAAN library[2] uses ternary secrets with $h = 64$. This relation from the sparsity may have an impact on the guarantee of security for this extended Shamir secret sharing scheme breaking the independence of coefficients of secret polynomials. Therefore, we recommend using some distribution for secret in which the coefficients are sampled independently instead of sparse secret distributions.

B. THRESHOLD STRUCTURE AND PROACTIVE SECRET SHARING

1) KEY SHARE RE-DISTRIBUTION PROTOCOLS

In this paragraph, we will introduce some key share renewal methods to be used for giving a threshold structure and proactive secret sharing to our scheme. Since the first proactive secret sharing was proposed in 1995 by

[2]https://github.com/snucrypto/HEAAN

Herzberg *et al.* [26], many proactive secret sharing schemes have been designed with each purpose. These proactive secret sharing schemes can be classified into two cases upon the method of re-distributing shares as following:

- **By adding shares of 0.** This method is mostly chosen for proactive secret schemes including the first one ([26]–[28]). Note that Shamir secret sharing has a linearity between the secrets and their shares. More precisely, if $(s_1, \ldots, s_n) \leftarrow \mathsf{Share}(s, t, n)$ and $(\bar{s}_1, \ldots, \bar{s}_n) \leftarrow \mathsf{Share}(\bar{s}, t, n)$, then we have $\mathsf{Combine}(\{s_i + \bar{s}_i\}_{i\in I}) \rightarrow s' = s + \bar{s}$ for any subset $I \subseteq [n]$ of size more than or equals to t. Thus, the key point of re-distributing algorithm in this class is generating new shares whose shared secret is actually 0 and adding these to the old shares like masking.
- **By re-sharing each old shares.** This method chosen by [29]–[31] is more complicate than the former and comes from the fact that the reconstruction algorithm is performed by linear operation on the shares s_i and their coefficients $\lambda_{I,i}$ are determined only by the indices I of gathered parties not by the collected shares themselves. For more detailed, let us assume that n parties shared a secret s by $(s_1, \ldots, s_n) \leftarrow \mathsf{Share}(s, t, n)$ and that t parties $\{P_i\}_{i\in I}$ gathered to re-distribute the shares with new (t', n')-threshold structure. First, each party $P_i (i \in I)$ runs $(s_{i,1}, \ldots, s_{i,n'}) \leftarrow \mathsf{Share}(s_i, t', n')$ and sends $s_{i,j}$ to the party P_j. Then, we have $s_i = \sum_{j\in J} \lambda_{J,j} s_{i,j}$ for all i and all $J \subseteq [n']$ with $|j| \geq t'$ and so if we set new shares $s'_j = \sum_{i\in I} \lambda_{I,i} s_{i,j}$ for all $j \in [n']$, the shared secret s can be reconstructed by computing
$$\begin{aligned}\sum_{j\in J} \lambda_{J,j} s'_j &= \sum_{j\in J} \lambda_{J,j} \cdot (\sum_{i\in I} \lambda_{I,i} s_{i,j}) \\ &= \sum_{j\in J} \sum_{i\in I} \lambda_{J,j} \cdot \lambda_{I,i} s_{i,j} \\ &= \sum_{i\in I} \lambda_{I,i} \sum_{j\in J} \lambda_{J,j} s_{i,j} \\ &= \sum_{i\in I} \lambda_{I,i} s_i = s\end{aligned}$$
for any $J \subseteq [n']$ with $|j| \geq t'$.

However, Nikov and Nikova showed the re-distributing method by adding shares of 0 are vulnerable to a specific attack even for a passive mobile adversary [34]. Thus, we focus on the re-sharing methods from now on.

2) DECENTRALIZED THRESHOLD CKKS SCHEME

Adapting a key share re-distribution protocol after generation of public encryption key and evaluation keys, one can give a threshold structure into our decentralized CKKS scheme. Here, we provide some protocols for this with its modified threshold decryption. Remind that the shared secret is of the form $s = s_1 + \cdots + s_n$ in the previous section.

- $\mathsf{THE.Threshold}(\{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}, t)$: This key update protocol $\mathsf{Threshold}$ is an one round, n-party protocol among $P_1, \ldots, P_n$.
  - **Round I.** For $i \in [n]$, each party P_i executes the following:
    1) Sample independently $a_{i,1}, \ldots, a_{i,t-1} \leftarrow \mathcal{R}_q$ and set $f_i(x) = s_i + a_{i,1}x + \cdots + a_{i,t-1}x^{t-1}$ where $\mathsf{sk}_i = (1, s_i)$.
    2) Compute shares $s_{i,j} = f_i(j)$ and send it to P_j ($j \in [n]$).

    Upon receiving $\{s_{i,j}\}_{i\in[n]}$, each party P_j for $j \in [n]$ computes new share $s'_j = \sum_{i\in I} s_{i,j}$ and sets $\mathsf{sk}_j \leftarrow (1, s'_j)$.

Since the relation between shares to bring back the common shared secret is changed from simple additions into a linear operation with Lagrange coefficients after this key update algorithm, the decryption protocol should also be modified as in [35].

- $\mathsf{THE.Dec1}(\{\mathsf{sk}_1, \cdots, \mathsf{sk}_n\}_{i\in I}, \mathbf{c})$: Our (t, n)-threshold decryption protocol based on Shamir secret sharing $\mathsf{Dec1}$ is an one round, t-party protocol among $P_1, \ldots, P_n$.
  - **Round I.** Given the ciphertext $\mathbf{c} = (c_0, c_1)$, for $i \in I$ each party P_i executes the following.
    1) Sample a smudging error $e_i^{\mathsf{sm}} \leftarrow \chi_{\mathsf{sm}}(B_{\mathsf{sm}}^{\mathsf{Dec}})$.
    2) Compute a partial decryption $p_i \leftarrow c_1 \cdot s_i + (n!)^2\ e_i^{\mathsf{sm}} \pmod{q_0}$, and broadcast it to other parties.

    Upon receiving $\{p_j\}_{j\neq i}$, P_i computes the final decryption
$$c_0 + \sum_{i\in I} \lambda_{I,i} p_i \pmod{q_0}.$$

Theorem 1 (Thresholdize/Decryption): Let $\mathbf{c} = (c_0, c_1)$ *be a level-*ℓ *ciphertext of* m *under* pk *satisfying*
$$\|\langle \mathbf{c}, \mathsf{sk}\rangle - m \pmod{q_0}\|_\infty^{\mathsf{can}} \leq B$$
for some $B > 0$ *and let* $(n!)^3 \leq q_0$.

If $\{\mathsf{sk}'_i\}_{i\in[n]} \leftarrow \mathsf{THE.Threshold}(\{\mathsf{sk}_i\}_{i\in[n]}, t)$ *and* $m' \leftarrow \mathsf{THE.Dec1}(\{\mathsf{sk}'_1, \cdots, \mathsf{sk}'_n\}, \mathbf{c})$*, then it holds that*
$$\|m' - m \pmod{q_0}\|_\infty^{\mathsf{can}} \leq B + B_{\mathsf{Dec}}$$
where $B_{\mathsf{Dec}} = n(n!)^3\ B_{\mathsf{sm}}^{\mathsf{Dec}}$.

Proof: Since m' is computed by $c_0 + \sum_{i\in I} \lambda_{I,i} p_i \pmod{q_0}$ and $p_i \leftarrow c_1 \cdot s_i + (n!)^2\ e_i^{\mathsf{sm}} \pmod{q_0}$ where $e_i^{\mathsf{sm}} \leftarrow \chi_{\mathsf{sm}}(B_{\mathsf{sm}}^{\mathsf{Dec}})$, $m' = c_0 + c_1 \cdot \sum_{i\in I} \lambda_{I,i} s_i + \sum_{i\in I} \lambda_{I,i} (n!)^2\ e_i^{\mathsf{sm}}$ Therefore, it holds that
$$\begin{aligned}&|m' - m \pmod{q_\ell}\|_\infty^{\mathsf{can}} \\ &\leq \|m' - \langle \mathbf{c}, \mathsf{sk}\rangle\|_\infty^{\mathsf{can}} + \|\langle \mathbf{c}, \mathsf{sk}\rangle - m\|_\infty^{\mathsf{can}} \\ &\leq \|\sum_{i\in I} \lambda_{I,i} (n!)^2\ e_i^{\mathsf{sm}}\|_\infty^{\mathsf{can}} + B \\ &\leq n(n!)^3\ B_{\mathsf{sm}}^{\mathsf{Dec}} + B.\end{aligned}$$
The last inequality comes from $|(n!)^2 \cdot \lambda_{i,I}| \leq (n!)^3$ as an integer [36]. □

3) PROACTIVE SECURITY

- $\mathsf{THE.KeyUpdate1}(\{\mathsf{sk}_i\}_{i\in I})$: Our protocol for key update $\mathsf{KeyUpdate1}$ based on Shamir secret sharing scheme is an one round, n-party protocol among $P_1, \ldots, P_n$.
  - **Round I.** For $i \in I$, each party P_i executes the following:
    1) Sample independently $a_{i,1}, \ldots, a_{i,t-1} \leftarrow \mathcal{R}_q$ and set $f_i(x) = s_i + a_{i,1}x + \cdots + a_{i,t-1}x^{t-1}$ where $\mathsf{sk}_i = (1, s_i)$.
    2) Compute shares $s_{i,j} = f_i(j)$ and send it to P_j ($j \in [n]$).

    Upon receiving $\{s_{i,j}\}_{i\in I}$, each party P_j for $j \in [n]$ computes new share $s'_j = \sum_{i\in I} \lambda_{I,i} s_{i,j}$ and sets $\mathsf{sk}_j \leftarrow (1, s'_j)$.

In fact, when we restrict the case on (n,n)-threshold, there are simple methods to re-distribute secret shares in which decryption errors are smaller than the previous methods. Let a secret s be distributed to shares s_i so that $s = s_1 + \cdots + s_n$.

- **By adding shares of 0.** Each party P_i generate $s_{i,j}$ ($j = 1, \ldots, n$) such that $0 = s_{i,1} + \cdots + s_{i,n}$ and send $s_{i,j}$ to the party P_j. Then, setting the new shares $s'_j = s_j + \sum_{i=1}^{n} s_{i,j}$ for all $j \in [n]$, the shared secret s can be reconstructed by $s = s'_1 + \cdots + s'_n$.
- **By re-sharing each old shares.** Each party P_i can re-share s_i into $s_{i,j}$ so that $s_i = s_{i,1} + \cdots + s_{i,n}$ and send $s_{i,j}$ to the party P_j. Then, setting the new shares $s'_j = s_{1,j} + \cdots + s_{n,j}$ for all $j \in [n]$, the shared secret s can be reconstructed by $s = s'_1 + \cdots + s'_n$.

Applying this to our key update protocol, we can get another protocol as following which is usable only for (n, n)-threshold structure.

- $\mathsf{THE.KeyUpdate2}(\{\mathsf{sk}_1 \ldots, \mathsf{sk}_n\})$: Our protocol for key update $\mathsf{KeyUpdate2}$ supporting (n, n)-threshold structure is an one round, n-party protocol among $P_1, \ldots, P_n$.
  - **Round I.** For $i \in [n]$, each party P_i executes the following:
    1) Sample independently $a_{i,1}, \ldots, a_{i,t-1} \leftarrow \mathcal{R}_q$ and set $a_{i,t} = s_i - \sum_{j=1}^{t-1} a_{i,j}$ where $\mathsf{sk}_i = (1, s_i)$.
    2) Send $a_{i,j}$ to P_j ($j \in [n]$).

    Upon receiving $\{a_{i,j}\}_{i\in[n]}$, each party P_j for $j \in [n]$ computes new share $s'_j = \sum_{i=1}^{n} a_{i,j}$ and sets $\mathsf{sk}_j \leftarrow (1, s'_j)$.

V. CONCLUSION

Using our (n, n)-threshold HE scheme, we have obtained (t, n)-threshold HE scheme based on CKKS HE scheme. Then we have introduced proactive secret sharing scheme to update secret key share of our (t, n)-threshold HE scheme whenever needed.

As a real use case of our (t, n)-threshold HE scheme, consider multiple organizations who want to analyze the aggregated data without concerning data privacy. In that case, organizations can delegate the secret key management to n TTPs who can decrypt the encrypted result when more than t of them agree, using our decentralized (t, n)-threshold HE scheme. It is plausible to assume that there is no hierarchy among n TTPs, so bottom-up approach is appropriate in key generation protocol.

To ensure the joint key security, smudging errors are required which inevitably enlarge the entire parameter size. An interesting open problem is to eliminate the smudging errors in our scheme. One can further reduce the parameter size of our threshold HE scheme using well-known optimization techniques for CKKS scheme, such as RNS decomposition method [37], [38].

APPENDIX A
THE EVALUATION KEY GENERATION PROTOCOL IN [24] ON RING LWE PROBLEMS

The basic concept of [24] is the same as ours except for the generation of public evaluation keys. To generate an evaluation key, the common public encryption key pk should be built first by a protocol because the individual secret key share s_i has to be encrypted under pk. Thus, their protocol for evaluation key generation is performed by following:

- $\mathsf{THE.KeyGen}$: This generation protocol KeyGen is a three round, n-party protocol among $P_1, \cdots, P_n$.
  - **Round I.** For $i \in [n]$, each party P_i executes the following.
    1) Sample $s_i \leftarrow \chi_{\mathsf{sk}}$ and $e_i \leftarrow \chi_{\mathsf{err}}$, and compute $b_i \leftarrow -a \cdot s_i + e_i \pmod{P \cdot q_L}$.
    2) Set its secret key share as $\mathsf{sk}_i \leftarrow (1, s_i)$ and broadcast b_i to other parties.
  - **Round II.** Upon receiving $\{b_j\}_{j\neq i}$, P_i sets the common public key as $\mathsf{pk} \leftarrow (\sum_{j=1}^{n} b_j, a) \in \mathcal{R}^2_{P\cdot q_L}$.
    1) Compute an encryption $\mathbf{c}_i \leftarrow \mathsf{THE.Enc}(\mathsf{pk}, Ps_i)$.
    2) Broadcast $\mathbf{c}_i$ to other parties.
  - **Round III.** Upon receiving $\{\mathbf{c}_j\}_{j\neq i}$, P_i executes the following.
    1) Compute $\mathsf{evk}_{i,j} \leftarrow s_i \cdot \mathbf{c}_j$ which can be considered as an encryption of Ps_is_j under pk for all $j \neq i$.
    2) Get a new ciphertext $\mathsf{evk}_{i,i} \leftarrow v_i \cdot \mathsf{pk} + (Ps_i^2 + \tilde{e}^{\mathsf{sm}}_{i,0}, \tilde{e}^{\mathsf{sm}}_{i,1}) \in \mathcal{R}^2_{P\cdot q_L}$ where $v_i \leftarrow \chi_{\mathsf{Enc}}$ and $\tilde{e}^{\mathsf{sm}}_{i,0}, \tilde{e}^{\mathsf{sm}}_{i,1} \leftarrow \chi_{\mathsf{sm}}(B^{\mathsf{Eval}}_{\mathsf{sm}})$.
    3) Compute $\mathsf{evk}_i \leftarrow \sum_{j=1}^{n} \mathsf{evk}_{i,j}$ which can be considered as an encryption of Pss_i under pk.
    4) Broadcast evk_i.

    Upon receiving $\{\mathsf{evk}_j\}_{j\neq i}$, P_i can finally set the common evaluation key $\mathsf{evk} \leftarrow \sum_{j=1}^{n} \mathsf{evk}_j \in \mathcal{R}^2_{P\cdot q_L}$

Lemma 9 (Evaluation key): Let evk *be an evaluation key generated by the previous protocol, then it holds that*

$$\|\langle \mathsf{evk}, \mathsf{sk}\rangle - P \cdot s^2 \pmod{P \cdot q_L}\| \leq B$$

where $B = O(n^2 N^{\frac{3}{2}} B^{\mathsf{Enc}}_{\mathsf{sm}} + nNB^{\mathsf{Eval}}_{\mathsf{sm}})$.

Proof: Note that we have if $j \neq i$,

$$\langle \mathsf{evk}_{i,j}, \mathsf{sk}\rangle - P \cdot s_i s_j = s_i \cdot (\langle \mathbf{c}_j, \mathsf{sk}\rangle - P \cdot s_j)$$

$$= s_i \cdot (v'_j \cdot e + e^{\mathsf{sm}}_{j,0} + s e^{\mathsf{sm}}_{j,1})$$

for some $v'_j \leftarrow \mathcal{ZO}$ and $e^{\mathsf{sm}}_{j,0}, e^{\mathsf{sm}}_{j,1} \leftarrow \chi_{\mathsf{sm}}(B^{\mathsf{Enc}}_{\mathsf{sm}})$, and

$$\begin{aligned}\langle \mathsf{evk}_{i,i}, \mathsf{sk}\rangle - P \cdot s_i^2 &= v_i \langle \mathsf{pk}, \mathsf{sk}\rangle + \tilde{e}^{\mathsf{sm}}_{i,0} + s \cdot \tilde{e}^{\mathsf{sm}}_{i,1} \\ &= v_i \cdot e + \tilde{e}^{\mathsf{sm}}_{i,0} + s \cdot \tilde{e}^{\mathsf{sm}}_{i,1}\end{aligned}$$

where $e = \sum_{i=1}^{n} e_i$ and $\tilde{e}^{\mathsf{sm}}_{i,0}, \tilde{e}^{\mathsf{sm}}_{i,1} \leftarrow \chi_{\mathsf{sm}}(B^{\mathsf{Eval}}_{\mathsf{sm}})$. Thus,

$$\begin{aligned}&\|\langle \mathsf{evk}, \mathsf{sk}\rangle - P \cdot s^2\| \\ &= \|\sum_{i,j}(\langle \mathsf{evk}_{i,j}, \mathsf{sk}\rangle - P \cdot s_i s_j)\| \\ &= \|\sum_{i}(s_i \sum_{j\neq i}(v'_j e + e^{\mathsf{sm}}_{j,0} + s e^{\mathsf{sm}}_{j,1}) + v_i e + \tilde{e}^{\mathsf{sm}}_{i,0} + s\tilde{e}^{\mathsf{sm}}_{i,1})\| \\ &\leq 32\, n\sqrt{\frac{N}{2}}\sqrt{\frac{(n-1)N}{2}}\sigma\sqrt{nN} + 16\, n\sqrt{\frac{N}{2}}B^{\mathsf{Enc}}_{\mathsf{sm}}\sqrt{\frac{(n-1)N}{3}} \\ &\quad + 32\, n\sqrt{\frac{N}{2}}\sqrt{\frac{nN}{2}}B^{\mathsf{Enc}}_{\mathsf{sm}}\sqrt{\frac{(n-1)N}{3}} + 16\sqrt{\frac{nN}{2}}\sigma\sqrt{nN} \\ &\quad + 6\, B^{\mathsf{Eval}}_{\mathsf{sm}}\sqrt{\frac{nN}{3}} + 6\sqrt{\frac{nN}{2}}B^{\mathsf{Eval}}_{\mathsf{sm}}\sqrt{\frac{nN}{3}} \leq B.\end{aligned}$$

□

REFERENCES

[1] C. Gentry, *A Fully Homomorphic Encryption Scheme*, vol. 20, no. 9. Stanford, CA, USA: Stanford Univ. Stanford, 2009.

[2] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *Proc. IMA Int. Conf. Cryptogr. Coding*. New York, NY, USA: Springer, 2013, pp. 45–64.

[3] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical gapsvp," in *Proc. Annu. Cryptol. Conf.* New York, NY, USA: Springer, 2012, pp. 868–886.

[4] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.

[5] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Proc. Annu. Cryptol. Conf.*, New York, NY, USA: Springer, 2011, pp. 505–524.

[6] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM J. Comput.*, vol. 43, no. 2, pp. 831–871, 2014.

[7] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* New York, NY, USA: Springer, 2010, pp. 24–43.

[8] J.-S. Coron, T. Lepoint, and M. Tibouchi, "Scale-invariant fully homomorphic encryption over the integers," in *Proc. Int. Workshop Public Key Cryptogr.* New York, NY, USA: Springer, 2014, pp. 311–328.

[9] J. H. Cheon and D. Stehlé, "Fully homomophic encryption over the integers revisited," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* New York, NY, USA: Springer, 2015, pp. 513–536.

[10] L. Ducas and D. Micciancio, "Fhew: Bootstrapping homomorphic encryption in less than a second," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* New York, NY, USA: Springer, 2015, pp. 617–640.

[11] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic evaluation of the aes circuit," in *Proc. Annu. Cryptol. Conf.* New York, NY, USA: Springer, 2012, pp. 850–867.

[12] Y. Doröz, Y. Hu, and B. Sunar, "Homomorphic AES evaluation using the modified LTV scheme," *Des., Codes Cryptogr.*, vol. 80, no. 2, pp. 333–358, Aug. 2016.

[13] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Advances in Cryptology—CRYPTO*. New York, NY, USA: Springer, 2013, pp. 75–92.

[14] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proc. 44th Symp. Theory Comput. STOC*, 2012, pp. 1219–1234.

[15] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* New York, NY, USA: Springer, 2017, pp. 409–437.

[16] S. Park, J. Byun, J. Lee, J. H. Cheon, and J. Lee, "HE-friendly algorithm for privacy-preserving SVM training," *IEEE Access*, vol. 8, pp. 57414–57425, 2020.

[17] C. Boura, N. Gama, M. Georgieva, and D. Jetchev, "Simulating homomorphic evaluation of deep learning predictions," in *Cyber Security Cryptography and Machine Learning*, S. Dolev, D. Hendler, S. Lodha, and M. Yung, Eds. Cham, Switzerland: Springer, 2019, pp. 212–230.

[18] J. H. Cheon, D. Kim, Y. Kim, and Y. Song, "Ensemble method for privacy-preserving logistic regression based on homomorphic encryption," *IEEE Access*, vol. 6, pp. 46938–46948, 2018.

[19] H. Chen, W. Dai, M. Kim, and Y. Song, "Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 395–412.

[20] A. Kim, Y. Song, M. Kim, K. Lee, and J. H. Cheon, "Logistic regression model training based on the approximate homomorphic encryption," *BMC Med. Genomics*, vol. 11, no. S4, p. 83, Oct. 2018.

[21] K. Han, S. Hong, J. H. Cheon, and D. Park, "Efficient logistic regression on large encrypted data," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 662, Jul. 2018.

[22] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[23] G. Asharov, A. Jain, and D. Wichs, "Multiparty computation with low communication, computation and interaction via threshold fhe," IACR Cryptol. ePrint Arch., Tech. Rep. 2011/613, 2011. [Online]. Available: https://eprint.iacr.org/2011/613

[24] G. Asharov, A. Jain, A. López-Alt, E. Tromer, V. Vaikuntanathan, and D. Wichs, "Multiparty computation with low communication, computation and interaction via threshold fhe," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* New York, NY, USA: Springer, 2012, pp. 483–501.

[25] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "Cloud-assisted multiparty computation from fully homomorphic encryption," *IACR Cryptol. ePrint Arch.*, vol. 2011, p. 663, Dec. 2011.

[26] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," in *Proc. Annu. Int. Cryptol. Conf.* New York, NY, USA: Springer, 1995, pp. 339–352.

[27] D. Schultz, B. Liskov, and M. Liskov, "MPSS: Mobile proactive secret sharing," *ACM Trans. Inf. Syst. Secur.*, vol. 13, no. 4, pp. 1–32, Dec. 2010.

[28] S.-J. Wang, Y.-R. Tsai, and P.-Y. Chen, "Strategies of proactive (k, n) threshold secret sharing and application in a secure message exchange system," *J. Comput.*, vol. 19, no. 1, pp. 29–38, 2008.

[29] T. M. Wong, C. Wang, and J. M. Wing, "Verifiable secret redistribution for archive systems," in *Proc. 1st Int. IEEE Secur. Storage Workshop*, Dec. 2002, pp. 94–105.

[30] V. H. Gupta and K. Gopinath, "An extended verifiable secret redistribution protocol for archival systems," in *Proc. 1st Int. Conf. Availability, Rel. Secur. (ARES)*, Apr. 2006, p. 8.

[31] V. H. Gupta and K. Gopinath, "$G_{its}{}^2$ VSR: An information theoretical secure verifiable secret redistribution protocol for long-term archival storage," in *Proc. 4th Int. IEEE Secur. Storage Workshop*, Sep. 2007, pp. 22–33.

[32] P. Mukherjee and D. Wichs, "Two round multiparty computation via multikey fhe," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* New York, NY, USA: Springer, 2016, pp. 735–763.

[33] A. Costache and P. Nigel, "Which ring based somewhat homomorphic encryption scheme is best," in *Topics in Cryptology—CT-RSA*, vol. 9610, San Francisco, CA, USA: Springer, Feb. 2016, 2016, p. 325.

[34] V. Nikov and S. Nikova, "On proactive secret sharing schemes," in *Proc. Int. Workshop Sel. Areas Cryptogr.* New York, NY, USA: Springer, 2004, pp. 308–325.

[35] D. Boneh, R. Gennaro, S. Goldfeder, A. Jain, S. Kim, P. M. Rasmussen, and A. Sahai, "Threshold cryptosystems from threshold fully homomorphic encryption," in *Proc. Annu. Int. Cryptol. Conf.* New York, NY, USA: Springer, 2018, pp. 565–596.

[36] S. Agrawal, X. Boyen, V. Vaikuntanathan, P. Voulgaris, and H. Wee, "Functional encryption for threshold functions (or fuzzy ibe) from lattices," in *Proc. Int. Workshop Public Key Cryptogr.* New York, NY, USA: Springer, 2012, pp. 280–297.

[37] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "A full RNS variant of approximate homomorphic encryption," in *Proc. Int. Conf. Sel. Areas Cryptogr.* New York, NY, USA: Springer, 2018, pp. 347–368.
[38] K. Han and D. Ki, "Better bootstrapping for approximate homomorphic encryption," in *Proc. Cryptographers' Track RSA Conf.* New York, NY, USA: Springer, 2020, pp. 364–390.

EUNKYUNG KIM received the B.S. degree in mathematics and statistics and the M.S. and Ph.D. degrees in mathematics from Ewha Womans University, Seoul, South Korea, in 2011, 2013, and 2018, respectively.

She has been doing research and development with Samsung SDS, Seoul, since 2018. Her current research interests include efficient applications and secure key management for homomorphic encryption.

JINHYUCK JEONG received the B.S. degree in education of mathematics and the Ph.D. degree in mathematical sciences from Seoul National University, Seoul, South Korea, in 2012 and 2019, respectively.

He has been doing research and development with Samsung SDS, South Korea, since 2019. His current research interests include biometrics authentication, homomorphic encryption with its applications, and multiparty computations.

HYOJIN YOON received the master's and Ph.D. degrees in mathematics and cryptography from Seoul National University. She is currently the Team Leader of the Security Algorithm Team, Samsung SDS.

YOUNGHYUN KIM received the B.S. degree in mathematical science and computer engineering from Seoul National University, Seoul, South Korea, in 2018.

He has been doing research and development with the Security Algorithm Team, Samsung SDS, South Korea, since 2018. His current research interests include generative adversarial networks (GANs) with differential privacy, and homomorphic encryption.

JIHOON CHO received the M.Math. degree in cryptography from the University of Waterloo and the Ph.D. degree in information security from the Royal Holloway University of London.

He has worked as a Security Architect for mobile devices with LG Electronics, South Korea. He is currently the Director of the security research with Samsung SDS, South Korea.

JUNG HEE CHEON (Associate Member, IEEE) received the B.S. and Ph.D. degrees in mathematics from KAIST, in 1991 and 1997, respectively.

He is currently a Professor with the Department of Mathematical Sciences and the Director of the Industrial and Mathematical Data Analytics Research Center (IMDARC), Seoul National University (SNU). Before joining SNU, he has worked as a Senior Researcher with ETRI, the Visiting Scientist of Brown University, and an Assistant Professor with ICU. He has been working on computational number theory, cryptology, and information security. In those areas, he has published more than 50 journal articles, including *Journal of Cryptology*, *Mathematics of Computation*, IEEE Transactions on Information Theory, IEEE Transactions on Computers, ACM TISSEC, and *SIAM Journal on Discrete Mathematics*; and many conference papers, including 13 Crypto/Eurocrypt, which are the most prestigious conferences for Cryptology. He is one of the co-inventors of braid cryptography and (independently) Xedni calculus on elliptic curve discrete logarithms. As one of the co-inventors of approximate homomorphic encryption HEAAN, he is actively working on homomorphic encryptions and their applications, including machine learning on encrypted data, homomorphic control systems, and DNA computation on encrypted data.

Dr. Cheon received the Best Paper Award in Eurocrypt 2015 for this joint work on cryptanalysis of multilinear maps. His works on improved Pollard rho algorithm and an efficient algorithm on strong DH problem drew many attentions, the former of which won the Best Paper Award in Asiacrypt 2008. He was selected as the Scientist of the month by the Korean Government in 2018 and won the POSCO Science Prize in 2019. He has co-chaired ANTS-XI, Asiacrypt 2015/2016, and MathCrypt 2018/2019. He served on program committee for numerous crypto conferences, including Crypto, Eurocrypt, Asiacrypt, and so on. He is an Associate Editor of *Designs, Codes and Cryptography*, *Journal of Communication Networks*, and *Journal of Cryptology*, that is the plagship journal in cryptology.