Homomorphic Encryption with Authority

Joohee Lee[1] and Joon-Woo Lee[2]

[1] Department of Convergence Security Engineering, Sungshin Women's University
jooheelee@sungshin.ac.kr
[2] Department of Computer Science and Engineering, Chung-Ang University
jwlee2815@cau.ac.kr

Abstract. Fully homomorphic encryption (FHE) enables computations over encrypted data, which allows privacy-preserving services to be held between a server and a client. However, real-world applications demand practical considerations, especially concerning public safety and legal investigations. Existing FHE schemes focus solely on privacy, neglecting the societal risks of criminal activities utilizing privacy-preserving services.

This paper introduces Homomorphic Encryption with Authority (HEwA), a novel framework that balances data privacy with public safety by incorporating an "authority" party. The proposed HEwA system operates in two phases: a normal phase, where client data privacy is protected, and an investigative phase, where the authority referring to a legally authorized entity, such as a government agency, exerts the right to recover suspicious client's data. We formalize the security model for HEwA, ensuring that client privacy is protected during the normal phase while enabling authorities to recover encrypted data in the investigative phase. As a concrete example, we design an efficient HEwA system solely based on the CKKS homomorphic encryption scheme, which supports approximate computations over real-number data, making it highly suitable for fruitful applications in AI, such as secure genomic analysis. We further provide rigorous security proofs. This new approach addresses the tension between privacy and public safety in cloud services, paving the way for the responsible use of homomorphic encryption in practice.

1 Introduction

Fully homomorphic encryption (FHE) is an encryption scheme that allows any circuit to be performed over encrypted data. Since Gentry proposed the first FHE scheme from the ideal lattice by developing the bootstrapping technique [14], numerous FHE constructions have been suggested based on lattice-based assumptions [5, 6, 4, 13, 12, 10]. Since FHE enables the delegation of computations while preserving the privacy of data, service providers with large-scale computing power can use FHE to provide a wide range of services without worrying about the privacy of the client's sensitive data.

⋆ All authors contributed equally to this work.

Although FHE is a promising primitive for protecting the privacy of data in use, its application in the real world requires many practical considerations in advance. For example, in real society, the government has a duty to perform national defense and public safety functions. Investigative authorities can exercise coercive power over suspects based on a court-issued warrant to fulfill this duty. In other words, while the privacy of citizens is generally respected in real society, the government, under legal authority, can enforce measures like search and seizure against specific suspicious individuals to maintain public order. In this regard, current FHE techniques focus solely on protecting client privacy without considering public safety. This raises the risk of significant negative societal consequences from the unchecked misuse of cloud services.

We consider the basic scenario with two parties, server and client, using FHE as follows. The client owns the private data, and the server has computation models for the desired services. The client generates a secret key and a corresponding public key, including evaluation keys, and sends them to the server. The client then encrypts their private data and sends it to the server. The server performs computations over the encrypted data and then sends the resulting ciphertexts to the client, which are then decrypted with the secret key by the client.

We now consider the case where the encrypted data in the hand of the server is decisive evidence of some criminal investigation for the client. For example, crimes using deepfake services have been rapidly increasing [19, 25], where fake voices, photos, or videos are easily generated using a specific person's voice or image for voice phishing or to create photos or videos that defame a particular individual. If AI services like deepfake could also be processed over encrypted data, criminals could take advantage of the privacy-preserving services without leaving any evidence of the crime. Specifically, they could upload stolen voices or images encrypted using FHE to a cloud server, receive the evaluation results as forged voices or photos for criminal use, and then completely destroy the private key to erase any evidence. In this case, due to the security guaranteed by homomorphic encryption, neither the government nor the server could access the data submitted by the client, preventing the government from fulfilling its role in crime prevention.

In this case, the security model for the FHE scheme should be changed to allow investigative authorities to access the data of a suspected individual with the authorization of the government. The traditional server-client two-party model is insufficient, and the other third party (or parties), which we call "authority" is required to decide whether they force certain clients to decrypt their data encrypted with FHE. In other words, we suggest the following key question about this issue.

How can the authority decrypt the client's FHE ciphertext without any help from the client?

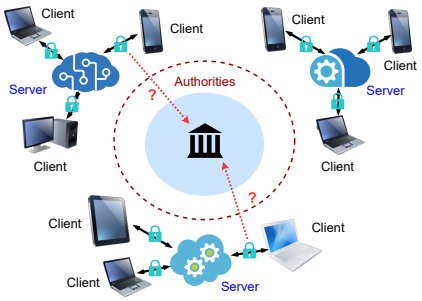

Fig. 1: Normal phase in HEwA

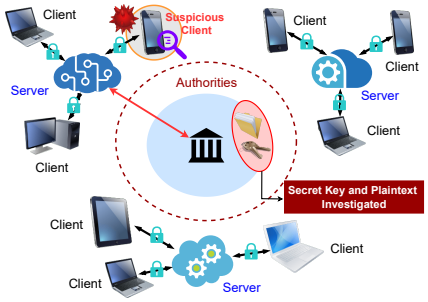

Fig. 2: Investigative phase in HEwA

1.1 Our Contribution

1.1.1 Homomorphic Encryption with Authority (HEwA)

We propose a new homomorphic encryption system, Homomorphic Encryption with Authority (HEwA), which not only ensures the data privacy of regular clients but also allows government agencies to fulfill their role in maintaining public safety. Instead of the conventional server-client two-party model, we propose a server-client-authority three-party model by introducing additional authority parties. Authority parties refer to entities such as investigative authorities or government agencies that possess the legal power to conduct investigations and enforce measures for public safety only under a warrant. There are two key considerations when incorporating these parties into the security model of FHE. First, when a warrant is issued, it should be possible to fully recover the data of a specific suspicious client. Second, without a warrant, the activities of regular clients and servers should remain unaffected. In other words, it is important to ensure that the authority of investigative bodies is not overly extended, maintaining a balance between security and public safety functions.

We first divide the HEwA protocol into two phases. The first phase is the *normal phase*, where the primary goal is to protect client data privacy. In this phase, the client's privacy should be guaranteed so that no other parties, including the server and authority, can gain any information about the client's sensitive data. Also, the client only interacts with the server in this phase. A key point to note is that the authority party does not interact with the client or the server during this phase. This design aims that the client or server can provide or receive services smoothly without the interference of the authority or any potential data leakage to the authority. The second phase is the *investigative phase*, in which the authority gains access to the ciphertexts submitted by the suspicious client to the server and can examine the message as evidence. In this phase, the authority enforces its legal power on the server, requiring the server to hand over the client's ciphertexts. The authority, through interaction with the server, obtains the client's secret key and decrypts the ciphertexts provided by the server.

We propose a new security model and definitions that ensure the objectives of both phases mentioned above are achieved. To achieve these goals, we need to define additional security properties. We have established three conditions that describe the role of the authority party to outline a new security model:

1. In the normal phase, there should be no interaction between the authority and the other two parties.
2. In the normal phase, no party should be able to obtain any secret information based on valid public information (e.g., the client's public key sent to the server, the authority's published public key, or the ciphertext sent by the client to the server) or through eavesdropping on public channels (e.g., the authority eavesdropping on the ciphertext sent by the client to the server).
3. In the investigative phase, the authority should be able to retrieve the client's secret key through communication with the server.

By formalizing the security model for the first time, this paper establishes clear security standards for future works that share the same objectives.

New Framework with HEwA. To be more specific, the authority party generates its secret key and public key and posts the public key in a public place. In the normal phase, suppose a client and a server want to execute the outsourcing computation protocol. In that case, the client generates its secret key and public key using the authority's public key, where the client's public key includes the evaluation keys that enable homomorphic operations. Note that the authority's public key can be downloaded without any interaction with the authority since it is posted in a public place. The client sends the public key and the ciphertext to the server via their secure channel, and the server performs a desired homomorphic circuit. Then, the server sends the resultant ciphertext to the client, which is decrypted by the client with its secret key.

In the investigative phase, we assume that the authority party issues the warrant for some suspicious client, and the client is reluctant to cooperate with the authority. Then, the server and the authority party help each other to recover the client's secret key. It is important to note that, in the normal phase, both the client and the server must be able to proceed with all protocols without any interaction with the authority. At the same time, in the investigative phase, the authority should be able to gain access to the client's secret key from communication only with the server and decrypt only the ciphertexts provided by the server. This requirement presents a nontrivial challenge in the design of the HEwA scheme.

Indistinguishability under Chosen-Plaintext Attack. With the addition of the authority party, new considerations regarding security have emerged. Specifically, there is a risk that the server could launch an attack to get information about the authority's secret key, which could then be used to gain access to the client's private data. Similarly, a client might attempt to obtain the authority's secret key to decrypt another client's ciphertext. These scenarios present

security challenges that were not previously considered in traditional homomorphic encryption schemes, making it necessary to extend and strengthen the definition of security to address these potential vulnerabilities. For these reasons, we propose a new security definition for the HEwA scheme, called $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$. This definition addresses all potential risks of information leakage that may arise between parties when the authority, a new party with a distinct role, is introduced. We will demonstrate that this definition effectively ensures security even with the inclusion of the authority.

Traceability of the Client's Evaluation Key. A malicious client may send a wrong public key or evaluation key to prevent the authority from extracting its secret key. The HEwA system does not explicitly verify that the client sends an honestly generated evaluation key with the authority's public key. Instead, we formalize the security condition, called the traceability of the client secret key, that if the homomorphic evaluation operated with the public key and the evaluation key is correct, then the authority should be able to recover the client secret key from the public key and the evaluation key with all but negligible probability. If an HEwA scheme with this property is used, it implies that if the client attempts to prevent their secret key from being recovered during the investigative phase, they will be unable to receive the desired services based on homomorphic computation. Therefore, for the client to receive homomorphic encryption services properly, they must accept that their secret key can be recovered during the investigative phase. If this condition is ensured, it serves as a safeguard to prevent the client from maliciously exploiting the services provided by the server.

1.1.2 Construction with the CKKS HE Scheme

We construct a HEwA scheme based on the approximate homomorphic encryption, also known as Cheon-Kim-Kim-Song (CKKS) scheme [10]. The CKKS scheme is a Ring-LWE-based FHE scheme that can encrypt real or complex number data. It is one of the most promising FHE schemes, which can be used in lots of applications with real number data, such as artificial intelligence, privacy-preserving cloud services, and secure genome analysis. Indeed, several researchers deal with performing privacy-preserving machine learning using the CKKS scheme, and it is also often used in secure genome analysis in iDASH competition. This enables many clients to take advantage of high performance computing while ensuring the privacy of their sensitive data. For this reason, the CKKS scheme is one of the most practical homomorphic encryption schemes.

One of the most important aspects in the design of the HEwA scheme is to allow only interaction between the client and the server during the normal phase, and in the investigative phase, to enable interaction solely between the authority and the server to retrieve the client's secret key. Since this is the most nontrivial part of the construction, devising a method to implement this is of utmost importance. We design the HEwA system based on the observation that

the CKKS homomorphic encryption scheme requires the client to share evaluation keys with the server. Evaluation keys are a type of public key needed for performing encrypted operations such as multiplication, rotation, and conjugation. They consist of bundles of Ring-LWE samples that encrypt transformed secret keys under the same secret key. Specifically, we will focus on rotation and conjugation evaluation keys, which are referred to as Galois keys. Note that the Galois key is composed of the several ciphertexts, and each ciphertext is the form of $\{(b_i, a_i)\}_{i=0,\cdots,\mathtt{dnum}-1} \in \mathcal{R}_{P\cdot Q}^{\mathtt{dnum}}$, where $\mathcal{R}_{P\cdot Q} = \mathbb{Z}_{P\cdot Q}[X]/(X^N+1)$, and $b_i = a_i \cdot s + e + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot h_i(s)$. Here, h_i's are automorphisms performed on the polynomials whose inverse can be easily computed.

We can design the scheme using any Galois key, but for explanation, we will focus on the conjugation key. The conjugation key refers to the evaluation key for the operation $h_i(s) = s(X^{-1})$. Here, we need the Ring-LWE-based public key of the authority. Suppose the authority has a secret key $s_{\mathsf{auth}} \in \mathcal{R}_{PQ}$ and issues the public key $(b_{\mathsf{auth}}, a_{\mathsf{auth}})$, which is posted in a public space. The public key is constructed as $b_{\mathsf{auth}} = -a_{\mathsf{auth}} \cdot s_{\mathsf{auth}} + e_{\mathsf{auth}}$ for a small error e_{auth}. Then, the client can obtain $(b_{\mathsf{auth}}, a_{\mathsf{auth}})$ without directly interacting with the authority and use it to construct the first and second elements of the conjugation key (b_0, a_0), (b_1, a_1) as follows:

$$a_0 = b_{\mathsf{auth}}, \quad b_0 = -b_{\mathsf{auth}} \cdot s + e + P \cdot \hat{Q}_0 \cdot [\hat{Q}_0^{-1}]_{Q_0} \cdot s(X^{-1}),$$

$$a_1 = a_{\mathsf{auth}}, \quad b_1 = -a_{\mathsf{auth}} \cdot s + e + P \cdot \hat{Q}_1 \cdot [\hat{Q}_1^{-1}]_{Q_1} \cdot s(X^{-1}),$$

Note that a_0 and a_1 are the posted public key of the authority, and the server can access these elements. Thus, the conjugation key of the client only includes b_0 and b_1, and the server uses the public key of the authority as elements of the conjugation key in the conjugation operation. The other elements beyond the first and second elements are constructed in the usual manner to form $\{(b_i, a_i)\}_{i=0,\cdots,\mathtt{dnum}-1} \in \mathcal{R}_{P\cdot Q}^{\mathtt{dnum}}$. In summary, the conjugation key is $\{b_0, b_1, \{(b_i, a_i)\}_{i=2,\cdots,\mathtt{dnum}-1}\}$, and when mixed with the public key of the authority $(b_{\mathsf{auth}}, a_{\mathsf{auth}})$, it acts as a valid conjugation key. In the investigative phase, to retrieve the client's secret key with the cooperation of the authority and the server, the following computation is performed:

$$b_0 + b_1 \cdot s_{\mathsf{auth}} = P \cdot (\hat{Q}_0 \cdot [\hat{Q}_0^{-1}]_{Q_0} + \hat{Q}_1 \cdot [\hat{Q}_1^{-1}]_{Q_1} \cdot s_{\mathsf{auth}}) \cdot s(X^{-1}) + e'',$$

where e'' is a small error. After performing this computation, removing the known multiplicand and errors allows us to obtain $s(X^{-1})$. By applying the inverse automorphism, we can retrieve s. Once the authority recovers s, it can decrypt the ciphertext obtained by cooperation of the server and investigate the client's data.

From a security perspective, two key aspects need to be proven. By proving these two aspects, we establish both the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security of the scheme and the traceability of the client's secret key, thus reinforcing the overall security and robustness of the system.

Proof of $\mathsf{IND\text{-}CPA}_{\mathbf{HEwA}}$ Security. We demonstrate the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security of the system. Since the authority generates its public and secret keys based on the Ring-LWE problem, the public key is indistinguishable from a uniform distribution for parties that do not know the secret key. By leveraging this fact and assuming that the underlying RNS-CKKS homomorphic encryption scheme is already proven to be $\mathsf{IND\text{-}CPA}$ secure, we can extend the security guarantee to show that our new scheme also satisfies $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security. Since RNS-CKKS homomorphic encryption is known to provide $\mathsf{IND\text{-}CPA}$ security, the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security of our scheme is ensured by our proof.

Proof of Traceability of the Client Secret Key. To establish traceability, we need to prove that if the homomorphic conjugation operation is performed accurately, the conjugation key must have been generated through a valid key generation process. The key point here is that, while previous research in homomorphic encryption focused on showing the correctness of key generation and homomorphic operations, proving traceability requires demonstrating the reverse implication. Specifically, we should show that accurate computation of the conjugation operation necessitates the correct generation of the conjugation key. To achieve this, we carefully examine each step of the conjugation operation and rigorously prove the necessary conditions for ensuring the correctness of the computation. We also compare these conditions to the practical parameters used in the RNS-CKKS scheme to ensure they hold under realistic circumstances.

1.2 Related Works

After the blueprint of Gentry's FHE proposal [15], the various FHE schemes [5, 6, 4, 13] has been proposed. Among them, the CKKS scheme and its RNS variant [12, 10] stands out as a promising candidate for privacy-preserving machine learning research [17, 18, 2, 23, 21, 22, 7], as it is suitable for dealing with a large-size of real valued data and it provides approximate computation in a Single Instruction Multiple Data (SIMD) manner. So far, FHE has historically been developed for the two-party scenario engaging a client and a server.

There are two types of research that extend homomorphic encryption to include more than two parties by introducing new models. These are threshold homomorphic encryption [16, 3, 26] and multi-key homomorphic encryption [8, 1, 9]. Both approaches assume a model with one server and multiple clients, but they differ in the key generation process and decryption method.

In threshold homomorphic encryption, the client parties with decryption authority must be fixed before the actual computation takes place. During the creation of the public key, multiple secret keys are combined to generate a single public key and evaluation key that are used in the computations. Decryption can only occur if a certain threshold number of clients participate in the process. On the other hand, in multi-key homomorphic encryption, computations can be carried out with ciphertexts encrypted under different keys at any point during the process, meaning the client parties do not need to be fixed before computation

begins. However, all client parties must participate in the decryption process to decrypt the result.

These homomorphic encryption models differ in their goals from our HEwA model. In both of the above models, even with multiple client parties, no party can inherently access the secret information of other clients. This makes it challenging to prevent a malicious client from exploiting the server's services. In contrast, the authority party in our HEwA model does not participate in the computations and only becomes involved when a warrant is issued. The HEwA model is designed to prevent abuse of the server's services by clients. The HEwA model is compatible with both threshold homomorphic encryption and multi-key homomorphic encryption, allowing them to be used together. The HEwA scheme complements these models by introducing an authority party that can ensure compliance in investigative scenarios, without altering the core computation or encryption processes.

1.3 Outlines

In Section 2, we cover the preliminaries of this paper, introducing the definitions and security of homomorphic encryption, along with the RNS-CKKS homomorphic encryption scheme. Section 3 defines the HEwA scheme suggesting the new properties and security definitions that extend conventional homomorphic encryption. In Section 4, we construct the HEwA scheme from the RNS-CKKS scheme based on the definitions from the previous section and prove the corresponding properties. Section 5 evaluates whether the conditions required for the scheme's security can be met using practical parameters in the RNS-CKKS scheme and verifies that the scheme demonstrates practical performance. Finally, Section 6 summarizes the results of this paper and suggests potential directions for future research.

2 Preliminaries

2.1 Notation

$\mathbb{Z}$ refers to the set of integers. For positive integers N and q, $\mathcal{R}$ and $\mathcal{R}_q$ are defined as $\mathbb{Z}[X]/(X^N+1)$ and $\mathbb{Z}_q[X]/(X^N+1)$, respectively. $\mathsf{negl}(n)$ refers to a negligible function in terms of n, meaning that for every positive polynomial $p(\,\cdot\,)$, there exists some $N_p > 0$ such that, for all $x > N_p$, the value of the function is less than $\frac{1}{p(x)}$. Let $[x]_q$ denote the remainder when x is divided by q, expressed as an element in the range $(-q/2, q/2]$. For $r = \sum_{i=0}^{N-1} r_i X^i \in \mathcal{R}$, the ℓ_1 norm $\|r\|_1$ and the ℓ_∞ norm $\|r\|_\infty$ are defined as $\sum_{i=0}^{N-1} |r_i|$ and $\max_{i=0}^{N-1} |r_i|$, respectively. ζ_N denotes $e^{\frac{2\pi i}{2N}}$ and if N is clear from the context, we can represent it as ζ.

2.2 LWE and its Ring Variant

In this subsection, we introduce the Learning with Errors (LWE) [24] problem and its ring variant [20] of which hardness assumption is the basis of the security of our HEwA scheme.

Let m, n, and q be positive integers. For distributions χ_e over $\mathbb{Z}$ and χ_s over $\mathbb{Z}^n$, the LWE distribution $A^{LWE}_{m,n,q,\chi_e}(\chi_s)$ consisting of m samples $\{(\vec{a}_i, b_i)\}_{i=1}^m$ is defined by computing the equation

$$b_i = \langle \vec{a}_i, \vec{s} \rangle + e_i \mod q,$$

where $\vec{a}_i \in \mathbb{Z}_q^n$ is a uniform random vector, $\vec{e}_i \leftarrow \chi_e \in \mathbb{Z}$, and $\vec{s} \leftarrow \chi_s \in \mathbb{Z}^n$ is a secret vector.

The LWE problem has two versions: The decision LWE problem denoted as $\mathsf{LWE}_{n,q,\chi_e}(\chi_s)$ asks to distinguish $\{(\vec{a}_i, b_i)\}_{i=1}^m$ either from $A^{LWE}_{m,n,q,\chi_e}(\chi_s)$ or from the uniform distribution, while the search LWE problem asks to find $\vec{s} \in \mathbb{Z}_q^n$ sampled from $A^{LWE}_{m,n,q,\chi}(s)$.

The Ring LWE (RLWE) problem is a ring variant of the LWE problem. Let n and q be positive integers, χ_s and χ_e be distributions over $\mathcal{R}$. The (decision) RLWE problem over $\mathcal{R}_q$ denoted as $\mathsf{RLWE}_{n,q,\chi_e}(\chi_s)$ is to distinguish between uniform distribution over $\mathcal{R}_q^2$ and the distribution of $(a, a \cdot s + e) \in \mathcal{R}_q^2$, where a is a uniform random element in $\mathcal{R}_q$, $s \leftarrow \chi_s$ is the secret polynomial, and $e \leftarrow \chi_e$.

2.3 Homomorphic Encryption

Homomorphic Encryption (HE) is an encryption scheme that allows for the execution of arithmetic operations on encrypted data. In addition to the traditional public-key encryption algorithms, HE introduces the Eval algorithm. The Eval algorithm defines the basic operations that the homomorphic encryption scheme supports. The security definition of homomorphic encryption is the same as that of traditional public-key encryption, with the added requirement that the homomorphism property is satisfied. Below, we provide the formal definition of HE along with the IND-CPA security definition and the homomorphism property. The HEwA scheme we propose incorporates these definitions.

Definition 1 (Homomorphic Encryption). *A homomorphic encryption scheme (HE) for a message space $\mathcal{M}$ is a tuple of PPT algorithms* $\mathsf{HE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *defined as follows:*

- $\mathsf{Setup}(1^\lambda) \rightarrow \mathsf{pp}$*: Given the security parameter λ as input, the setup algorithm produces the public parameters* pp*.*
- $\mathsf{KeyGen}(\mathsf{pp}) \rightarrow (\mathsf{pk}, \mathsf{evk}, \mathsf{sk})$*: Given the public parameters* pp *as input, the key generation algorithm produces a public key* pk*, an evaluation key* evk *(or a collection of evaluation keys), and a secret key* sk*.*
- $\mathsf{Enc}(\mathsf{pk}, m) \rightarrow \mathsf{ct}$*: Given the public key* pk *and a message $m \in \mathcal{M}$, the encryption algorithm produces a ciphertext* ct*.*

– $\mathsf{Eval}(\mathsf{pk}, \mathsf{evk}, C, \mathsf{ct}_1, \cdots, \mathsf{ct}_k) \rightarrow \mathsf{ct}$*: Given the public key* pk*, a circuit* $C : \mathcal{M}^k \rightarrow \mathcal{M}$ *and a set of ciphertexts* $\mathsf{ct}_1, \cdots, \mathsf{ct}_k$*, the evaluation algorithm produces a ciphertext* ct*.*
– $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \rightarrow m'$*: Given the secret key and a ciphertext* ct*, the decryption algorithm produces a message* m'*.*

Table 1: IND-CPA game

Game IND-CPA
1: $(\mathsf{pk}, \mathsf{evk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2: $(m_0, m_1) \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{evk})$
3: $b \leftarrow \{0, 1\}$
4: $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$
5: $b' \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{evk}, \mathsf{ct}^*, m_0, m_1)$
6: **return** $[[b = b']]$

Definition 2 (IND-CPA security). *Let* $\mathsf{HE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *be a homomorphic encryption scheme.* IND-CPA *(INDistinguishability under Chosen Plaintext Attack) security is defined via the* IND-CPA *game in Table 1 and the advantage of adversary* $\mathcal{A}$ *is defined by*

$$\mathcal{A}dv_{\mathsf{HE}}^{\mathsf{IND\text{-}CPA}}(\mathcal{A}) := \Pr[\mathsf{IND\text{-}CPA}_{\mathsf{HE}}^{\mathcal{A}} = 1].$$

HE *is* IND-CPA *secure if* $\mathcal{A}dv_{\mathsf{HE}}^{\mathsf{IND\text{-}CPA}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$ *for an arbitrary probabilistic polynomial-time adversary* $\mathcal{A}$*.*

Definition 3. ($\mathcal{C}$-homomorphism)*. Let* $\mathcal{C}$ *be a circuit class. A scheme* HE *is* $\mathcal{C}$*-homomorphic if for any circuit* $C \in \mathcal{C}$ *and any inputs* $m_1, \cdots, m_\ell \in \mathcal{M}$*,*

$$\Pr[\mathsf{HE.Dec}(\mathsf{sk}, \mathsf{HE.Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \cdots, \mathsf{ct}_\ell)) \neq C(m_1, \cdots, m_\ell)] = \mathsf{negl}(\lambda),$$

where $(\mathsf{pk}, \mathsf{evk}, \mathsf{sk}) \leftarrow \mathsf{HE.KeyGen}(1^\lambda)$ *and* $\mathsf{ct}_i \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, m_i)$*. If a scheme is* $\mathcal{C}$*-homomorphic such that* $\mathcal{C}$ *includes all arithmetic circuits, then the scheme is defined to be fully homomorphic.*

2.4 RNS-CKKS Scheme

To instantiate our HEwA system, we utilize the RNS-CKKS scheme [11] as an underlying primitive. We describe the basic operations for the RNS-CKKS scheme as follows. The RNS-CKKS scheme is proven to satisfy the IND-CPA security, which is used in the security proof of our HEwA scheme.

- $\mathsf{Setup}(1^\lambda)$: Set a power-of-two ring degree N, a secret distribution χ_{sk} over $\mathcal{R}$, an error distribution χ_{err} over $\mathcal{R}$, an ephemeral secret distribution χ_{enc} over $\mathcal{R}$, moduli chains $\{q_0, \cdots, q_L\}$ and $\{p_0, \cdots, p_{k-1}\}$ which contain different primes, $Q = \prod_{i=0}^{L} q_i$ and $P = \prod_{j=0}^{k-1} p_j$ with respect to the security parameter λ.
- $\mathsf{KeyGen}(\mathsf{pp})$:
  - $\mathsf{SecretKeyGen}()$: Sample $s \leftarrow \chi_{\mathsf{sk}}$ and return the secret key $\mathsf{sk} := s$.
  - $\mathsf{SwitchKeyGen}(s, s')$: Let $Q = \prod_{i=0}^{\mathtt{dnum}-1} Q_i$, where $\mathtt{dnum} = \lceil (L+1)/\alpha \rceil$ and $Q_i = \prod_{j=\alpha i}^{\min(\alpha(i+1)-1, L)} q_j$ for $i = 0, \cdots, \mathtt{dnum} - 1$. We set $\hat{Q}_i = Q/Q_i$. Output a key-switching key

    $$\mathsf{swk}_{s \to s'} = \{\mathsf{swk}_{s \to s'}^{(i)}\}_{i=0}^{\mathtt{dnum}-1} := \{([-a_i \cdot s' + e_i + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s]_{PQ}, a_i)\}_{i=0}^{\mathtt{dnum}-1},$$

    where $a_i \leftarrow \mathcal{R}_{PQ}$ and $e_i \leftarrow \chi_{\mathsf{err}}$.
  - $\mathsf{PublicKeyGen}(\mathsf{sk})$: For the secret key $\mathsf{sk} = s$, let $\mathsf{pk} = ([-as + e]_Q, a)$ be a public encryption key, where $a \leftarrow \mathcal{R}_Q$, and $e \leftarrow \chi_{\mathsf{err}}$. Sample the relinearization key $\mathsf{rlk} \leftarrow \mathsf{SwitchKeyGen}(s^2, s)$, the rotation key $\mathsf{rot}_r \leftarrow \mathsf{SwitchKeyGen}(s^{5^r}, s)$ for each r, and the conjugation key $\mathsf{conj} \leftarrow \mathsf{SwitchKeyGen}(s^{-1}, s)$. Output $(\mathsf{pk}, \mathsf{evk} = (\mathsf{rlk}, \{\mathsf{rot}_\mathsf{r}\}, \mathsf{conj}))$.
- $\mathsf{Enc}(\mathsf{pk}, m \in \mathcal{R})$: Sample $v \leftarrow \chi_{\mathsf{enc}}$ and $e_0, e_1 \leftarrow \chi_{\mathsf{err}}$. Return $\mathsf{ct} \leftarrow [v \cdot \mathsf{pk} + (m + e_0, e_1)]_Q$.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct} = (c_0, c_1))$: For the secret key $\mathsf{sk} = s$, return $m = c_0 + c_1 s$.
- $\mathsf{Add}(\mathsf{ct}, \mathsf{ct}')$: Given two ciphertexts ct, $\mathsf{ct}' \in \mathcal{R}_q$, return a ciphertext $\mathsf{ct}_{\mathsf{add}} \leftarrow \mathsf{ct} + \mathsf{ct}' \mod q$.
- $\mathsf{Mult}(\mathsf{evk}, \mathsf{ct}, \mathsf{ct}')$: Given two ciphertexts $\mathsf{ct} = (c_0, c_1), \mathsf{ct} = (c_0', c_1') \in \mathcal{R}_q^2$ and an evaluation key $\mathsf{evk} = (\mathsf{rlk}, \{\mathsf{rot}_\mathsf{r}\}, \mathsf{conj}))$, compute an output $\mathsf{ct}_{\mathsf{mult}}$ as follows:
  1. compute $(d_0, d_1, d_2) \leftarrow [(c_0 \cdot c_0', c_0 \cdot c_1' + c_1 \cdot c_0', c_1 \cdot c_1')]_q$.
  2. $\mathsf{ct}_{\mathsf{mult}} \leftarrow [(d_0, d_1) + \lfloor P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [d_2]_{Q_i} \cdot \mathsf{rlk}^{(i)} \rceil]_q$
- $\mathsf{Rot}(\mathsf{evk}, \mathsf{ct}, r)$: Given a ciphertext $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_q^2$ and an evaluation key $\mathsf{evk} = (\mathsf{rlk}, \{\mathsf{rot}_\mathsf{r}\}, \mathsf{conj}))$, compute an left r-step rotation output $\mathsf{ct}_{\mathsf{rot}}$ as follows:
  1. compute $(d_0, d_1) \leftarrow (c_0(X^{5^r}), c_1(X^{5^r}))$.
  2. $\mathsf{ct}_{\mathsf{rot}} \leftarrow [(d_0, 0) + \lfloor P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [d_1]_{Q_i} \cdot \mathsf{rot_r}^{(i)} \rceil]_q$
- $\mathsf{Conj}(\mathsf{evk}, \mathsf{ct})$: Given a ciphertext $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_q^2$ and an evaluation key $\mathsf{evk} = (\mathsf{rlk}, \{\mathsf{rot}_\mathsf{r}\}, \mathsf{conj}))$, compute a conjugation output $\mathsf{ct}_{\mathsf{conj}}$ as follows:
  1. compute $(d_0, d_1) \leftarrow (c_0(X^{-1}), c_1(X^{-1}))$.
  2. $\mathsf{ct}_{\mathsf{conj}} \leftarrow [(d_0, 0) + \lfloor P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [d_1]_{Q_i} \cdot \mathsf{conj}^{(i)} \rceil]_q$

3 Model

3.1 System Model

In this subsection, we present a detailed formalization of the system model for the HEwA scheme. First, we provide definitions that specify the algorithms constituting the HEwA scheme, followed by a discussion of the additional properties

that are not present in standard homomorphic encryption. Then, we describe how the defined HEwA scheme can be used by three parties through specific protocols in each phase.

The following presents the definition of the HEwA scheme. In this definition, we introduce newly defined operations to provide additional functionalities beyond the homomorphic encryption scheme defined in Section 2.3. First, the operation $\mathsf{AuthKeyGen}$ is an algorithm that generates a pair $(\mathsf{sk}_a, \mathsf{pk}_a)$ of secret and public keys of the authority, allowing the authority to eventually obtain the client's secret key in the investigative situation. The operation $\mathsf{ClientKeyGen}$ generates the client's secret key sk_c, public key pk_c, and evaluation key evk_c for homomorphic operations, after downloading the authority's public key. The definitions of Enc, Eval, and Dec are the same as those in standard homomorphic encryption, where only the client's keys are used to ensure the protocol operates solely between the client and the server. Additionally, the SecRes operation is introduced, allowing the authority to derive the client's secret key sk_c from $(\mathsf{pk}_c, \mathsf{evk}_c)$ using its own secret key, sk_a.

Definition 4 (Homomorphic Encryption with Authority). *A Homomorphic-Encryption-with-Authority scheme (HEwA) is a tuple of PPT algorithms* $\mathsf{HEwA} = (\mathsf{Setup}, \mathsf{AuthKeyGen}, \mathsf{ClientKeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{SecRec})$ *defined as follows:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$*: Given the security parameter* λ *as input, the setup algorithm produces the public parameters* pp*.*
- $\mathsf{AuthKeyGen}(\mathsf{pp}) \to \{(\mathsf{pk}_a, \mathsf{sk}_a)\}$*: Given the public parameters* pp *as input, the authority key generation algorithm produces a pair of public key and secret key for the authority* $(\mathsf{pk}_a, \mathsf{sk}_a)$*.*
- $\mathsf{ClientKeyGen}(\mathsf{pp}, \mathsf{pk}_a) \to (\mathsf{pk}_c, \mathsf{sk}_c, \mathsf{evk}_c)$*: Given the public parameters* pp *and the authority public key as input, the client key generation algorithm produces a tuple of public key, secret key, and evaluation key for the client* $(\mathsf{pk}_c, \mathsf{sk}_c, \mathsf{evk}_c)$
- $\mathsf{Enc}(\mathsf{pk}_c, m) \to \mathsf{ct}$*: Given the client public key* pk_c *and a message* $m \in \mathcal{M}$*, the encryption algorithm produces a ciphertext* ct*.*
- $\mathsf{Eval}(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c, C, \mathsf{ct}_1, \cdots, \mathsf{ct}_k) \to \mathsf{ct}$*: Given the authority public key* pk_a*, the client public key* pk_c*, the client evaluation key* evk_c*, a circuit* $C : \mathcal{M}^k \to \mathcal{M}$ *and a set of ciphertexts* $\mathsf{ct}_1, \cdots, \mathsf{ct}_k$*, the evaluation algorithm produces a ciphertext* ct*.*
- $\mathsf{Dec}(\mathsf{sk}_c, \mathsf{ct}) \to m'$*: Given the secret key and a ciphertext* ct*, the decryption algorithm produces a message* m'*.*
- $\mathsf{SecRes}(\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_a) \to \mathsf{sk}'$*: Given the client public key, the client evaluation key, and the authority secret key, the secret key restoration algorithm produces the client's secret key.*

The correctness of this definition includes the correctness defined in Section 2.3. In this section, we focus only on the correctness that is not dealt in the standard homomorphic encryption, specifically the correctness of the SecRec algorithm. This definition ensures that when SecRec is performed using pk_c and evk_c obtained through $\mathsf{ClientKeyGen}$ along with the authority's secret key, the resulting key is equal to sk_c with overwhelming probability.

Definition 5 (Correctness of Restoration). *Assume that after deriving the public parameter* pp *through* $\mathsf{Setup}(1^\lambda)$, $\mathsf{AuthKeyGen}(\mathsf{pp})$ *is performed to obtain* pk_a *and* sk_a*, and* $\mathsf{ClientKeyGen}(\mathsf{pp}, \mathsf{pk}_a)$ *is performed to obtain* pk_c*,* sk_c*, and* evk_c*. Then, the following holds.*

$$\Pr[\mathsf{SecRes}(\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_a) \neq \mathsf{sk}_c] \leq \mathsf{negl}(\lambda).$$

Using the HEwA scheme that satisfies the above definition, we assume the following system model. We consider a large society where numerous independent servers provide cloud services using homomorphic encryption. Each server operates independently of other servers, offering services to a large number of clients while ensuring the privacy of client data. We also assume the existence of an investigative authority responsible for maintaining public safety. This investigative authority is not allowed to interact with any servers or clients, nor to eavesdrop on any information under normal circumstances. However, when a warrant is issued for a suspicious client, the authority must be able to fully recover information about the client's data. The former scenario is referred to as the *normal phase*, and the latter as the *investigative phase*.

Normal Phase

- **Authority Setup:**
  1. The authority generates pk_a and sk_a using $\mathsf{AuthKeyGen}(\mathsf{pp})$
  2. The authority announces pk_a. That is, the authority makes pk_a accessible to anyone without further interaction with the authority, and signed with the authority's signing key.
- **Client-Server Setup:**
  1. The server and client establish a secure channel between them.
  2. After receiving pk_a, the client performs $\mathsf{ClientKeyGen}(\mathsf{pp}, \mathsf{pk}_a)$ to generate pk_c, sk_c, and evk_c. Then, the client sends pk_c and evk_c to the server via secure channel between the client and server.
- **Homomorphic Evaluations:**
  1. For its input messages $m_1, \cdots, m_k \in \mathcal{M}$, the client computes and sends the ciphertexts

     $$\mathsf{ct}_1 = \mathsf{Enc}(\mathsf{pk}_c, m_1), \cdots, \mathsf{ct}_k = \mathsf{Enc}(\mathsf{pk}_c, m_k)$$

     to the server via secure channel.
  2. After receiving pk_a, the server performs homomorphic operations on the circuit $C : \mathcal{M}^k \to \mathcal{M}$ to obtain $\mathsf{ct} \leftarrow \mathsf{Eval}(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c, C, \mathsf{ct}_1, \cdots, \mathsf{ct}_k)$, then sends it to the client via secure channel between the client and server.
  3. The client decrypts the received ct and obtains $m \leftarrow \mathsf{Dec}(\mathsf{sk}_c, \mathsf{ct})$.

In the normal phase, the process begins with the authority generating its public and secret key pair and announcing the public key in a publicly accessible

location. In the description of the normal phase, this is referred to as Authority Setup, which can be considered as a preprocessing step. This step can be held even before the client agrees to get services from the server. Therefore, it is reasonable to assume that by the time the client requests cloud computations from the server, Authority Setup has already been completed. Additionally, it is assumed that pk_a is posted in a public location accessible to anyone and signed with the authority's signing key, and since it can be downloaded without interacting with the authority, accessing this key is not considered interaction with the authority. Step 1 in Client-Server Setup is the process where the server and client invoke a secure channel to encrypt the further interactions with symmetric encryption such as AES. Since the rest of the communication between the server and client is encrypted, the authority cannot obtain pk_c and evk_c to run the SecRec algorithm, which prevents the authority from recovering the client's secret key in the normal phase. Step 2 in Client-Server Setup is the procedure where the client generates its key tuple when it wishes to request cloud computations from the server. At this point, the client uses the authority's public key to generate the key tuple. Among the generated keys, the client sends the public key and evaluation key to the server. We note that the Authority Setup is held only once before multiple client-server sessions, and Client-Server Setup is also established only once before multiple requests of homomorphic evaluations.

Steps 1-3 in the homomorphic evaluations follow the standard process of homomorphic encryption-based cloud services between the client and the server. The client sends encrypted data to the server, which then performs homomorphic computations on the model C using the client's public key and evaluation key. The server sends the resulting ciphertext to the client, who decrypts it with their secret key to obtain $C(m_1, \cdots, m_k)$. Note that the step 2 in the homomorphic evaluations requires the server to receive pk_a from the public place, and this enables the scheme to satisfy the traceability condition for HEwA, which will be introduced in the next subsection.

Investigative Phase

1. The server sends pk_c and evk_c to the authority.
2. The authority performs $\mathsf{SecRes}(\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_a)$ using pk_c, evk_c, and sk_a to compute sk_c.
3. The server sends the target ciphertexts $\mathsf{ct}_1, \cdots, \mathsf{ct}_k$ to the authority.
4. The authority obtains the client's messages by decrypting $\mathsf{ct}_1, \cdots, \mathsf{ct}_k$ using sk_c, i.e., $\mathsf{Dec}(\mathsf{sk}_c, \mathsf{ct}_i)$ for $1 \leq i \leq k$.

In the investigative phase, the process involves receiving data from the server that previously interacted with the suspicious client in order to conduct an investigation. In traditional homomorphic encryption schemes, even if the server provides the client's encrypted data, it cannot be decrypted without the client's secret key. However, in the HEwA scheme, SecRec operation allows the recovery

of the client's secret key in the investigative phase. When the server provides the client's public key or evaluation key to the authority, the authority can use its own secret key to recover the client's secret key. The server then sends the client's encrypted data to the authority, which can decrypt the ciphertexts using the recovered secret key to proceed with the investigation.

One might argue that the authority has excessive power since it could potentially recover the decryption key. However, the authority cannot access the ciphertext without the server's cooperation in the investigative phase. Ciphertexts exchanged between the client and the server are transmitted through a point-to-point secure channel, making it impossible for the authority to eavesdrop on them. If access permissions need to be set differently for each service unit, separate public and private key pairs can be generated for each service. This can be implemented without requiring any modifications to the algorithm.

3.2 Security Model

In this subsection, we propose the security model of the HEwA scheme. More precisely, we model two security properties that HEwA should meet in the normal phase and investigative phase, respectively. First, in the normal phase, an adversary should not be able to learn any private information from the public information generated from each party. We extend the $\mathsf{IND\text{-}CPA}$ security of HE scheme modeling the normal phase, and name this property as $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security. The second type of security dubbed as "traceability" ensures that the client cannot maliciously alter the public key or evaluation keys to prevent the authority from recovering its secret key as far as the homomorphic evaluation on the server returns a correct result.

3.2.1 Indistinguishability under Chosen-Plaintext Attack for HEwA

We propose the following $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ (Indistinguishability under Chosen-Plaintext Attack for HEwA) security, extending the standard $\mathsf{IND\text{-}CPA}$ security of HE scheme. In this security game, an adversary models a server or an eavesdropper in our scenario which has an access to the public keys of authority and client and a challenge ciphertext encrypted with the client's public key.

Definition 6 ($\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ **security).** *Let Π be an HEwA scheme.* $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ *security is defined via the* $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ *game in Table 2 and the advantage of adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is defined by*

$$\mathcal{A}dv^{\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}}_{\mathsf{HEwA}}(\mathcal{A}) := \Pr[\mathsf{IND\text{-}CPA}^{\mathcal{A}}_{\mathrm{HEwA}} = 1].$$

Π is IND-CPA secure if $\mathcal{A}dv^{\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}}_{\mathsf{HEwA}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$ for an arbitrary probabilistic polynomial-time adversary $\mathcal{A}$.

We explain that, using the HEwA scheme satisfying $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security following the protocol specified in Section 3.1 in the normal phase, three parties, authority, server, and client, in our scenario cannot learn any meaningful information about each other's private data under the assumption that any two of three parties do not collude and either the server or the authority is honest.

Table 2: $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ game

Game $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$
1: $(\mathsf{pk}_a, \mathsf{sk}_a) \leftarrow \mathsf{AuthKeyGen}(1^\lambda)$
2: $(\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_c) \leftarrow \mathsf{ClientKeyGen}(1^\lambda, \mathsf{pk}_a)$
3: $(m_0, m_1) \leftarrow \mathcal{A}_1(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c)$
4: $b \leftarrow \{0, 1\}$
5: $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_c, m_b)$
6: $b' \leftarrow \mathcal{A}_2(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c, \mathsf{ct}^*, m_0, m_1)$
7: **return** $[[b = b']]$

Adversarial Server. Ensuring the server cannot access the client's private data is one of the most critical aspects in the security definition. In our new $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ game, the challenger simulates the view of the server to an adversary, forwarding the client's public key pk_c, evaluation key evk_c, and the authority's public key pk_a. The security game then asks the adversary to distinguish whether the challenge ciphertext is an encryption of m_0 or m_1 with non-negligible advantages. This definition guarantees that the server cannot gain any information from the client's ciphertext.

Also, since HEwA introduces the authority as a participant in the protocol, the server should not be able to learn any useful information from the authority as well. For example, if an adversary can somehow get the authority's secret key sk_a using the public keys and evaluation keys, then it may run $\mathsf{SecRec}(\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_a)$ and retrieve the client's secret key sk_c all but negligible probability by the correctness of restoration so that the adversary wins the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ game. Hence, the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security aims to ensure that the adversary modeling the server, given pk_a, pk_c, and evk_c, cannot obtain enough hints about the authority's secret key sk_a to attack the challenge ciphertext encrypted with pk_c. More precisely, suppose there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can derive a hint about sk_a using only pk_a, pk_c, and evk_c. In this case, the adversary $\mathcal{A}_1$ and $\mathcal{A}_2$ in the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ game could invoke $\mathcal{B}$ as many times as needed. However, the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security definition guarantees that any probabilistic polynomial-time algorithm will have only a negligible advantage of success. As a result, any information about sk_a obtained using $\mathcal{B}$ cannot be used in a meaningful way to attack the challenge ciphertext.

Adversarial Client. In the conventional client-server scenario using FHE, it was unnecessary to consider the client who owns the decryption key as an adversary, as the server does not have any private information that could be exploited. However, in the HEwA scheme, the presence of the authority party introduces the possibility that the client may attempt to access the authority's secret key to attack the other client's challenge ciphertext. However, given only the public key of authority pk_a, public key pk_c and evaluation key evk_c of the other client, the client cannot derive a meaningful information about the authority's secret

key sk_a to attack the other client's challenge ciphertext. This can be shown as in the case that considers an adversarial server.

Adversarial Authority. The authority may attempt to gain information about the client's message through the protocol between the client and server during the normal phase. However, as specified in the normal phase protocol, the public key, evaluation key, and the ciphertext of the client are trasmitted via an encrypted channel between the client and server. As a result, the authority cannot access the client's data during the normal phase. We note that we can also consider a security game for a malicious authority that an adversary generates pk_a in the public key space as well as m_0 and m_1 in the message space, and then guess a random bit b without receiving anything. The advantage of the game would be 0 since the adversary gets no further input. Since this security game would be rather trivial, we do not define such game explicitly.

3.2.2 Traceability of Client Secret Key

A malicious client may attempt to prevent its secret key from being recovered by the authority during the investigative phase. To achieve this, the client could manipulate the generation procedure of the public key and evaluation key that makes it impossible to recover the secret key. One of natural approaches to address this issue could be using a zero-knowledge proof (ZKP), where the client would prove that it has indeed generated the public key and evaluation key honestly from the secret key following the $\mathsf{ClientKeyGen}$ algorithm. However, since homomorphic encryption generally involves large parameter sizes, generating such a proof would result in a very large proof size, and proving and verifying it would be computationally expensive and time-consuming. Therefore, we formulate a security definition called traceability that fits in the investigative phase and directly design a practical HEwA system satisfying the traceability, rather than relying on the heavy tools such as ZKP.

We note that the public key and the evaluation keys are essential for the server to provide services to the client. In particular, we focus on the evaluation keys that enable the server to proceed with homomorphic computations over encrypted data. Our idea is to make the client and the server agree on the authority's public key, which conceives a part of the client's evaluation keys. In this way, we can make it possible that, if a malicious client focuses solely on preventing the recovery of its secret key during the investigative phase, it may fail to enable the server to perform the desired computations for the client during the normal phase with overwhelming probability. This would mean that the client cannot receive the data processing service it seeks, rendering the goal of preventing secret key recovery meaningless. Therefore, we define the 'traceability of a client's secret key' as a property that the secret key can be recovered through SecRec algorithm in the investigative phase as far as the homomorphic evaluation can be held correctly. We formalize it in the following definitions. In these definitions, unit homomorphic operations refer to the basic operations supported by the homomorphic encryption scheme. For example, in the BFV

scheme, the unit operations include addition, multiplication, row rotation, and column rotation. In the CKKS scheme, they include addition, multiplication, rotation, and conjugation. In Definition 7, we define a notion for unit-preserving public keys of homomorphic encryption scheme to design the traceability game in Definition 8.

Definition 7 ($(\mathcal{C},\ell)$-preserving, ℓ-unit-preserving). *Let $\mathcal{PK}$ and $\mathcal{EVK}$ be the client public key space and the evaluation key space for the HEwA scheme $\Pi = (\mathsf{Setup}, \mathsf{AuthKeyGen}, \mathsf{ClientKeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{SecRec})$, and $\mathcal{C}$ be a circuit class. Let $(\mathsf{pk}_a, \mathsf{sk}_a) \leftarrow \mathsf{AuthKeyGen}(\mathsf{pp})$. For $\ell > 0$ and a positive integer k, a pair of public key and evaluation key $(\mathsf{pk}', \mathsf{evk}') \in \mathcal{PK} \times \mathcal{EVK}$ is* $(\mathcal{C},\ell)$-preserving *if for any circuit $C \in \mathcal{C}$ and any inputs $m_1, \cdots, m_k \in \mathcal{M}$,*

$$\Pr[\|\mathsf{HE.Dec}(\mathsf{sk}, \mathsf{HE.Eval}(\mathsf{pk}_a, \mathsf{pk}', \mathsf{evk}', C, \mathsf{ct}_1, \cdots, \mathsf{ct}_k)) - C(m_1, \cdots, m_k)\|_\infty \geq 2^{-\ell}] \leq \mathsf{negl}(\lambda),$$

where $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}', m_i)$. If a pair of public key and evaluation key $(\mathsf{pk}', \mathsf{evk}')$ is $(\mathcal{C},\ell)$-preserving such that $\mathcal{C}$ includes all unit homomorphic operations, then the pair is defined to be ℓ-unit-preserving.

Intuitively, a pair $(\mathsf{pk}', \mathsf{evk}')$ is ℓ-unit-preserving if the unit homomorphic operations are correctly held within small errors when using $(\mathsf{pk}', \mathsf{evk}')$ with overwhelming probability. The following definition presents the traceability notion which introduces an adversary who generates ℓ-unit-preserving pair of public key and evaluation key and asks whether the secret key of the target client can be recovered using the key pair generated by the adversary.

Definition 8 (Traceability). *Let Π be an HEwA scheme and $\ell > 0$. The ℓ-traceability of the client secret key is defined via the* Traceability *game in Table 3 and the advantage of adversary $\mathcal{A}$ is defined by*

$$\mathcal{A}dv^{\mathsf{Tr},\ell}_{\mathsf{HEwA}}(\mathcal{A}) := \Pr[\mathsf{Traceability} = 1 | (\mathsf{pk}', \mathsf{evk}') \text{ is } \ell\text{-unit-preserving}], \quad (1)$$

where $\mathsf{pk}', \mathsf{evk}'$ are defined in the Traceability *game. Π is secret ℓ-traceable if $\mathcal{A}dv^{\mathsf{Tr},\ell}_{\mathsf{HEwA}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$ for an arbitrary probabilistic polynomial-time adversary $\mathcal{A}$.*

4 Construction from RNS-CKKS Scheme

In this section, we describe our construction for HEwA instantiated with the RNS-CKKS scheme. More specifically, we explain the intuition behind the construction, and present the algorithms for Setup, AuthKeyGen, ClientKeyGen, and SecRes, as remaining algorithms such as Enc, Eval, and Dec are the same as in the original RNS-CKKS scheme.

Table 3: Traceability game

Game Traceability
1: $(\mathsf{pk}_a, \mathsf{sk}_a) \leftarrow \mathsf{AuthKeyGen}(\mathsf{pp})$
2: $(\mathsf{pk}_c, \mathsf{sk}_c, \mathsf{evk}_c) \leftarrow \mathsf{ClientKeyGen}(\mathsf{pp}, \mathsf{pk}_a)$
3: $(\mathsf{pk}', \mathsf{evk}') \leftarrow \mathcal{A}(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c)$
4: $\mathsf{sk}' \leftarrow \mathsf{SecRec}(\mathsf{pk}', \mathsf{evk}', \mathsf{sk}_a)$
5: **return** $[[\mathsf{sk}_c \neq \mathsf{sk}']]$

4.1 Intuition and Key Idea

The key idea to construct our HEwA scheme is to enforce the client to encrypt the client's secret key with the authority's public key and engrave it as a part of the evaluation keys of the RNS-CKKS scheme.

In the original RNS-CKKS scheme, $s(X^{-1})$ is encrypted in the conjugation keys with the secret key $s \in \mathcal{R}$ to allow key switching from $s(X^{-1})$ to s. More precisely, a conjugation key is of the form $\{(b_i, a_i)\}_{i=0,\cdots,\mathtt{dnum}-1}$, where

$$a_i \leftarrow \mathcal{R}_{PQ},$$
$$b_i \leftarrow [-a_i \cdot s + e_i + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s(X^{-1})]_{PQ}.$$

In our setting, the authority announces its public key $\mathsf{pk}_a = (b_{\mathsf{auth}} = -a_{\mathsf{auth}} \cdot s_{\mathsf{auth}} + e_{\mathsf{auth}}, a_{\mathsf{auth}})$, where $a_{\mathsf{auth}} \leftarrow \mathcal{R}_{PQ}$, $s_{\mathsf{auth}} \leftarrow \chi_{\mathsf{sk}}$, and $e_{\mathsf{auth}} \leftarrow \chi_{\mathsf{err}}$ and stores the corresponding secret key s_{auth}. Using the public key of the authority, the client samples a conjugation key $\{(b_i, a_i)\}_{i=0,\cdots,\mathtt{dnum}-1}$, where (a_0, a_1) is set to be $(b_{\mathsf{auth}}, a_{\mathsf{auth}})$ instead of sampled them from the uniform distribution over $\mathcal{R}_{PQ}$, and the rests are computed as same as the original conjugation key. In this way, if the authority retrieves a part of the conjugation key $\{(b_0, a_0), (b_1, a_1)\}$, the authority can recover $s(X^{-1})$ by computing

$$b_0 + b_1 \cdot s_{\mathsf{auth}} = P \cdot (\hat{Q}_0 \cdot [\hat{Q}_0{}^{-1}]_{Q_0} - \hat{Q}_1 \cdot [\hat{Q}_1{}^{-1}]_{Q_1} \cdot s_{\mathsf{auth}}) \cdot s(X^{-1}) + (e_{\mathsf{auth}} \cdot s - e_1 \cdot s_{\mathsf{auth}} + e_0),$$

discarding the error terms and dividing the known value multiplied to it.

Moreover, it can be shown that the proposed HEwA scheme satisfies the traceability condition defined in Definition 8. That is, if the client requests a service that includes unit operations, especially the conjugation in our construction, and wants the service to be held correctly, then it should honestly generate the conjugation key allowing the authority to recover the client's secret key in the investigative phase with an overwhelming probability under certain parameter conditions. The parameter conditions for traceability are specified in Theorem 3 in the following subsection. The beauty of this procedure is that it solely relies on the FHE without the help of any other heavy tools such as ZKP, to guarantee that the conjugation key is generated honestly using the given authority's public key except for the cryptographically negligible probability.

We remark that this idea applies not only to conjugation keys, but also to rotation keys in a similar manner. We embed the authority's public key into the client's conjugation key for the following reasons.

1. **Conjugation operation is essential for bootstrapping in homomorphic encryption.** Conjugation operation is a fundamental component of the bootstrapping process in homomorphic encryption. Therefore, if a sufficiently complex computation requires bootstrapping, conjugation becomes necessary even if the computation itself does not inherently involve conjugation. For sufficiently complex computational models, conjugation operations are indispensable. Typically, computations requested from the cloud tend to involve complex models that are challenging for clients to compute independently, making bootstrapping—and consequently, conjugation—frequently necessary.
2. **Redundant inclusion of conjugation in simpler calculation is possible.** Even if a model is simple enough to not require bootstrapping or conjugation operations, the server can redundantly incorporate conjugation into the computation model. For example, the server could include a step where conjugation is performed twice on a specific intermediate value. In such cases, if the client issues a valid conjugation key honestly, the computations yield the correct result. However, if the client intentionally provides an invalid conjugation key, the result will be incorrect. Since the client has no authority to constrain the server's computation procedures, the server is free to include such operations.

4.2 Algorithms

In this subsection, we present the algorithms for HEwA instantiated with the RNS-CKKS scheme. Let $\Pi = (\mathsf{Setup}, \mathsf{AuthKeyGen}, \mathsf{ClientKeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{SecRec})$ for which Setup, $\mathsf{AuthKeyGen}$, $\mathsf{ClientKeyGen}$, and SecRes are defined as follows. Then the correctness of restoration algorithm SecRec is proven.

- $\mathsf{Setup}(1^\lambda)$: Set a power-of-two ring degree N, a secret distribution χ_{sk} over $\mathcal{R}$, an error distribution χ_{err} over $\mathcal{R}$, an ephemeral secret distribution χ_{enc} over $\mathcal{R}$, moduli chains $\{q_0, \cdots, q_L\}$ and $\{p_0, \cdots, p_{k-1}\}$ which contain different NTT-friendly primes, $Q = \prod_{i=0}^{L} q_i$ and $P = \prod_{j=0}^{k-1} p_j$ with respect to the security parameter λ. Let α be a positive integer, $\mathtt{dnum} := \lceil (L+1)/\alpha \rceil$ and $Q_i := \prod_{j=\alpha i}^{\min(\alpha(i+1)-1, L)} q_j$ for $i = 0, \cdots, \mathtt{dnum} - 1$ so that $Q = \prod_{i=0}^{\mathtt{dnum}-1} Q_i$. We set $\hat{Q}_i = Q/Q_i$.
- $\mathsf{AuthKeyGen}(\cdot)$: Sample $a_{\mathsf{auth}} \leftarrow \mathcal{R}_{PQ}$, $s_{\mathsf{auth}} \leftarrow \chi_{\mathsf{sk}}$, and $e_{\mathsf{auth}} \leftarrow \chi_{\mathsf{err}}$. Return $\mathsf{pk}_a = (b_{\mathsf{auth}} := -a_{\mathsf{auth}} \cdot s_{\mathsf{auth}} + e_{\mathsf{auth}}, a_{\mathsf{auth}})$ and $\mathsf{sk}_a = s_{\mathsf{auth}}$.
- $\mathsf{ClientKeyGen}(\mathsf{pk}_a)$: Parse $\mathsf{pk}_a := (b_{\mathsf{auth}}, a_{\mathsf{auth}}) \in \mathcal{R}_{PQ}^2$.
  - $\mathsf{SecretKeyGen}()$: Sample $s \leftarrow \chi_{\mathsf{sk}}$ and return the secret key $\mathsf{sk} := s$.
  - $\mathsf{SwitchKeyGen}(s, s')$: Output a key-switching key

    $$\mathsf{swk}_{s \to s'} = \{\mathsf{swk}_{s\to s'}^{(i)}\}_{i=0}^{\mathtt{dnum}-1} := \{([-a_i \cdot s' + e_i + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s]_{PQ}, a_i)\}_{i=0}^{\mathtt{dnum}-1},$$

    where $a_i \leftarrow \mathcal{R}_{PQ}$ and $e_i \leftarrow \chi_{\mathsf{err}}$.

- $\mathsf{PublicKeyGen}(\mathsf{sk}, \mathsf{pk}_a = (b_{\mathsf{auth}}, a_{\mathsf{auth}}))$: For the secret key $\mathsf{sk} = s$, let $\mathsf{pk} = ([-as+e]_Q, a)$ be a public encryption key, where $a \leftarrow \mathcal{R}_Q$, and $e \leftarrow \chi_{\mathsf{err}}$. Sample the relinearization key $\mathsf{rlk} \leftarrow \mathsf{SwitchKeyGen}(s^2, s)$, the rotation key $\mathsf{rot}_r \leftarrow \mathsf{SwitchKeyGen}(s^{5^r}, s)$ for each r. Sample the conjugation key $\mathsf{conj} \leftarrow (b_0, b_1, \{(b_i, a_i)\}_{i=2,\cdots,\mathtt{dnum}-1})$ as follows. Sample $a_i \leftarrow \mathcal{R}_{PQ}$ for $i = 2, \cdots, \mathtt{dnum} - 1$ and $e_i \leftarrow \chi_{\mathsf{err}}$ for $i = 0, \cdots, \mathtt{dnum} - 1$, and compute $b_0 = -b_{\mathsf{auth}} \cdot s + e + P \cdot \hat{Q}_0 \cdot [\hat{Q}_0^{-1}]_{Q_0} \cdot s(X^{-1}), \quad b_1 = -a_{\mathsf{auth}} \cdot s + e + P \cdot \hat{Q}_1 \cdot [\hat{Q}_1^{-1}]_{Q_1} \cdot s(X^{-1})$, and $b_i \leftarrow [-a_i \cdot s + e_i + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s(X^{-1})]_{PQ}$ for $i = 2, \cdots, \mathtt{dnum} - 1$.

– $\mathsf{SecRes}(\mathsf{evk}_c, \mathsf{sk}_a)$: Let $\mathsf{sk}_a = s_{\mathsf{auth}}$. From evk_c, parse the conjugation key $\mathsf{conj} = (b_0, b_1, \{(b_i, a_i)\}_{i=2,\cdots,\mathtt{dnum}-1})$. Compute

$$s' \leftarrow \left[\left\lfloor \frac{1}{P}(b_0 + b_1 \cdot s_{\mathsf{auth}}) \right\rceil \cdot \frac{Q_0 Q_1}{Q} \cdot (Q_1[\hat{Q}_0^{-1}]_{Q_0} + Q_0[\hat{Q}_1^{-1}]_{Q_1} \cdot s_{\mathsf{auth}})^{-1}\right]_{Q_0 Q_1}$$

and output $\mathsf{sk}' := s'(X^{-1})$.

Theorem 1 (Correctness of Restoration). *The secret key of Π is recovered from $\mathsf{SecRes}(\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_a)$ with all but negligible probability as long as the following inequality holds:*

$$\Pr\left[\|-e_{\mathsf{auth}} \cdot s + e_1 \cdot s_{\mathsf{auth}} + e_0\|_\infty < \frac{P}{2} : s \leftarrow \chi_{\mathsf{sk}}, e_{\mathsf{auth}}, e_0, e_1 \leftarrow \chi_{\mathsf{err}}\right] > 1 - \mathsf{negl}(\lambda).$$

Proof. Let $(\mathsf{pk}_a = (b_{\mathsf{auth}} = -a_{\mathsf{auth}} \cdot s_{\mathsf{auth}} + e_{\mathsf{auth}}), \mathsf{sk}_a := s_{\mathsf{auth}})$ sampled from $\mathsf{AuthKeyGen}$, and $\{(b_i = [-a_i \cdot s + e_i + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s(X^{-1})]_{PQ}, a_i)\}_{i=0,1}$ be a part of the conjugation key in evk_c. Then,

$$b_0 = [-(-a_{\mathsf{auth}} \cdot s_{\mathsf{auth}} + e_{\mathsf{auth}}) \cdot s + e_0 + P \cdot \hat{Q}_0 \cdot [\hat{Q}_0^{-1}]_{Q_0} \cdot s(X^{-1})]_{PQ}, \text{ and}$$
$$b_1 \cdot s_{\mathsf{auth}} = [-a_{\mathsf{auth}} \cdot s \cdot s_{\mathsf{auth}} + e_1 \cdot s_{\mathsf{auth}} + P \cdot \hat{Q}_1 \cdot [\hat{Q}_1^{-1}]_{Q_1} \cdot s(X^{-1}) \cdot s_{\mathsf{auth}}]_{PQ}.$$

Hence,

$$b_0 + b_1 \cdot s_{\mathsf{auth}} = P \cdot (\hat{Q}_0 \cdot [\hat{Q}_0^{-1}]_{Q_0} + \hat{Q}_1 \cdot [\hat{Q}_1^{-1}]_{Q_1} \cdot s_{\mathsf{auth}}) \cdot s(X^{-1}) + (-e_{\mathsf{auth}} \cdot s + e_1 \cdot s_{\mathsf{auth}} + e_0),$$

where the operations are held modulo PQ. If $\|-e_{\mathsf{auth}} \cdot s + e_1 \cdot s_{\mathsf{auth}} + e_0\|_\infty < \frac{P}{2}$, then $(\hat{Q}_0 \cdot [\hat{Q}_0^{-1}]_{Q_0} + \hat{Q}_1 \cdot [\hat{Q}_1^{-1}]_{Q_1} \cdot s_{\mathsf{auth}}) \cdot s(X^{-1}) \mod Q$ is recovered by the calculation $\left\lfloor \frac{1}{P}(b_0 + b_1 \cdot s_{\mathsf{auth}}) \right\rceil$. After that, $s(X^{-1}) \mod Q_0 Q_1$ can be recovered by dividing it and the modulus Q by $Q/Q_0 Q_1$ first, and multiplying an inverse of $Q_1[\hat{Q}_0^{-1}]_{Q_0} + Q_0[\hat{Q}_1^{-1}]_{Q_1} \cdot s_{\mathsf{auth}} \mod Q_0 Q_1$ to the result. This concludes the proof of the correctness of restoration. □

4.3 Security Proof

In this section, we prove that the HEwA scheme defined above satisfies all security conditions. The first theorem demonstrates that the scheme satisfies the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ security definition. This proof leverages the fact that the underlying RNS-CKKS scheme already satisfies $\mathsf{IND\text{-}CPA}$ security.

Theorem 2 ($\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ Security). *Given that RNS-CKKS is* IND-CPA *secure, the HEwA scheme Π instantiated with RNS-CKKS satisfies the* $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ *security under the hardness assumption of* $\mathsf{RLWE}_{n,PQ,\chi_{\mathsf{err}}}(\chi_{\mathsf{sk}})$.

Proof. For a message $m \in \mathcal{M}$, We define distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ as follows.

$$\mathcal{D}_0 := \{(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c, \mathsf{ct}) : (\mathsf{pk}_a, \mathsf{sk}_a) \leftarrow \mathsf{AuthKeyGen}(\cdot), (\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_c) \leftarrow \mathsf{ClientKeyGen}(\mathsf{pk}_a), \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}_c, m)\},$$

$$\mathcal{D}_1 := \{(\mathsf{pk}_a, \mathsf{pk}_c, \mathsf{evk}_c, \mathsf{ct}) : \mathsf{pk}_a \leftarrow \mathcal{R}_{PQ}, (\mathsf{pk}_c, \mathsf{evk}_c, \mathsf{sk}_c) \leftarrow \mathsf{ClientKeyGen}(\mathsf{pk}_a), \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}_c, m)\}.$$

Then, the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ are computationally indistinguishable under the hardness assumption of $\mathsf{RLWE}_{n,PQ,\chi_{\mathsf{err}}}(\chi_{\mathsf{sk}})$.

Suppose that the HEwA scheme Π is not IND-CPA secure so that there exists an adversary $\mathcal{A}$ winning the $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ game with a non-negligible advantage. We can construct an IND-CPA adversary $\mathcal{B}$ for the RNS-CKKS scheme with a non-negligible advantage by simulating a $\mathsf{IND\text{-}CPA}_{\mathrm{HEwA}}$ game to $\mathcal{A}$ as follows.

1. The IND-CPA challenger $\mathcal{C}$ sends the public key and evaluation key $(\mathsf{pk}, \mathsf{evk})$ of RNS-CKKS to $\mathcal{B}$.
2. $\mathcal{B}$ samples $\mathsf{pk}_a \leftarrow \mathcal{R}_{PQ}$, and sends $(\mathsf{pk}_a, \mathsf{pk}, \mathsf{evk})$ to $\mathcal{A}$.
3. $\mathcal{A}$ sends (m_0, m_1) to $\mathcal{B}$ so that $\mathcal{B}$ forwards it to $\mathcal{C}$.
4. As $\mathcal{C}$ sends ct^* to $\mathcal{B}$, $\mathcal{B}$ forwards it to $\mathcal{A}$.
5. Finally, when $\mathcal{A}$ sends a bit b' back, $\mathcal{B}$ forwards it to $\mathcal{C}$.

Since $\mathcal{D}_0$ and $\mathcal{D}_1$ are computationally indistinguishable, the security game simulated by $\mathcal{B}$ defined as above is indistinguishable from the real IND-CPA security game for $\mathcal{A}$. Hence, $\mathcal{B}$ wins the IND-CPA game with a non-negligible advantage by forwarding $\mathcal{A}$'s answers, and this contradicts to the assumption that RNS-CKKS is IND-CPA secure. Therefore, the HEwA scheme Π instantiated with RNS-CKKS is IND-CPA secure under the assumption that RNS-CKKS is IND-CPA secure and $\mathsf{RLWE}_{n,PQ,\chi_{\mathsf{err}}}(\chi_{\mathsf{sk}})$ is hard. □

Next, we prove the traceability of the client's secret key. We need to show that for the conjugation operation to produce the desired result the conjugation key should follow the known structure of the conjugation key. This proof is more complex than merely demonstrating the correctness of the homomorphic operation, as it requires proving the reverse direction, making the argument more challenging. We will use the following useful lemma in the main proof. After proving the following lemma, we will proceed to prove Theorem 3.

Lemma 1. *Let X_i be independent integer-valued random variables with uniform distribution from $(-t_i, t_i] \cap \mathbb{Z}$ for $i = 0, \cdots, N-1$ and integer $t_i \geq 0$, and $t_{\min}$ is the minimum value among the nonzero t_i's. If $N \geq 2$, $\sum_{i=0}^{N-1} t_i > 0$, and $t_{\min} \geq 5$, then the following formula holds.*

$$\Pr\left[\left|\sum_{i=0}^{N-1} X_i\right| \geq \frac{1}{N}\sum_{i=0}^{N-1} t_i\right] > \frac{1}{4}$$

Proof. Let the number of non-zero t_i's be n. Since $\sum_{i=0}^{N} t_i > 0$, we have $n \geq 1$.

1) The case $n = 1$: Without loss of generality, assume that $t_0 > 0$. Then, we have

$$\Pr\left[\left|\sum_{i=0}^{N-1} X_i\right| \geq \frac{1}{N}\sum_{i=0}^{N-1} t_i\right] = \Pr\left[|X_0| \geq \frac{t_0}{N}\right] \geq \frac{2(t_0(1-1/N)-1)}{2t_0}$$
$$= 1 - \frac{1}{N} - \frac{1}{t_0} > \frac{1}{4},$$

2) The case $n \geq 2$: Without loss of generality, assume that $t_i > 0$ for $i = 0, \cdots, n$. Since $\Pr\left[X_i \in (-t_i\left(1-\frac{2}{n}\right), t_i]\right] \geq 1 - \frac{1}{n}$ and X_i's are mutually independent, we have

$$\Pr\left[X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right], i = 0, \cdots, n-1\right] \geq \left(1-\frac{1}{n}\right)^n$$

The conditional distribution of X_i given the event that $X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right]$ is also the uniform distribution from $\left[[-t_i\left(1-\frac{2}{n}\right)]+1, t_i\right]$, where the inner square bracket is the gauss function. Note that the center of the interval $[[-t_i\left(1-\frac{2}{n}\right)]+1, t_i]$ is $\beta_i = ([-t_i\left(1-\frac{2}{n}\right)]+1+t_i)/2$, which is larger than t_i/n. Since each conditional distribution of X_i has a symmetric distribution with mean β_i, the sum of these conditional distribution $\sum_{i=0}^{n-1} X_i$ has also a symmetric distribution with mean $\sum_{i=0}^{n-1}\beta_i$, which is larger than $\sum_{i=0}^{n-1}\frac{t_i}{n}$. Thus, we have

$$\Pr\left[\sum_{i=0}^{n-1} X_i \geq \frac{1}{n}\sum_{i=0}^{n-1} t_i \middle| X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right], i = 0, \cdots, n-1\right]$$
$$\geq \Pr\left[\sum_{i=0}^{n-1} X_i \geq \sum_{i=0}^{n-1}\beta_i \middle| X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right], i = 0, \cdots, n-1\right] \geq \frac{1}{2}$$

We know that $(1-\frac{1}{x})^x$ is increasing function when $x > 0$, and thus $(1-\frac{1}{n})^n \geq (1-\frac{1}{2})^2 = \frac{1}{4}$. Thus, we have

$$\Pr\left[\sum_{i=0}^{N-1} X_i \geq \frac{1}{N}\sum_{i=0}^{N-1} t_i\right] \geq \Pr\left[\sum_{i=0}^{n-1} X_i \geq \frac{1}{n}\sum_{i=0}^{n-1} t_i\right]$$
$$> \Pr\left[\sum_{i=0}^{n-1} X_i \geq \frac{1}{n}\sum_{i=0}^{n-1} t_i \bigwedge X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right], i = 0, \cdots, n-1\right]$$
$$= \Pr\left[\sum_{i=0}^{n-1} X_i \geq \frac{1}{n}\sum_{i=0}^{n-1} t_i \middle| X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right], i = 0, \cdots, n-1\right]$$
$$\cdot \Pr\left[X_i \in \left(-t_i\left(1-\frac{2}{n}\right), t_i\right], i = 0, \cdots, n-1\right]$$
$$\geq \frac{1}{2}\cdot\left(1-\frac{1}{n}\right)^n \geq \frac{1}{8}$$

Similarly, we can prove that $\Pr\left[\sum_{i=0}^{N-1} X_i \leq -\frac{1}{N}\sum_{i=0}^{N-1} t_i\right] > \frac{1}{8}$, and thus we have

$$\Pr\left[\left|\sum_{i=0}^{n-1} X_i\right| \geq \frac{1}{n}\sum_{i=0}^{n-1} t_i\right] > \frac{1}{4}.$$

□

Theorem 3 (Traceability of Client Secret Key). *If there exist B_s and B_e such that*

1. $\Pr[\|s\|_\infty > B_s \mid s \leftarrow \chi_{\mathsf{sk}}]$ *and* $\Pr[\|e\|_\infty > B_e \mid e \leftarrow \chi_{\mathsf{err}}]$ *are negligible,*
2. $Q_i \geq 5$ *for all i's, and*
3. *the following inequality holds,*

$$N \cdot B_e \cdot B_s + \frac{P}{Q_1} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} \cdot B_s + \frac{P}{Q_0} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} < \frac{P}{2},$$

then the HEwA scheme Π is secret ℓ-traceable.

Proof. We will prove that if the unit homomorphic operations of Π—addition, multiplication, rotation, and conjugation—are performed with sufficient accuracy, the SecRec algorithm should necessarily recover sk_c with overwhelming probability. To do this, we need to show that no pk', evk' exist such that the unit homomorphic operations can be performed with sufficient accuracy but the probability that SecRec fails to recover sk_c is non-negligible. Since the SecRec algorithm in Π uses only the conjugation key to recover the secret key, it is sufficient to consider only the conjugation operation to consider the result of SecRec. Therefore, we should prove that no conj' exists such that the Conj operation is performed correctly, but the probability that the result of SecRec is not sk_c remains non-negligible.

For conj' to be used as a conjugation key in the Conj operation, it should consist of $\tilde{b}_0$, $\tilde{b}_1$, $\{(\tilde{b}_i, \tilde{a}_i)\}_{i=2,\cdots,\mathtt{dnum}-1}$, each of which is in $\mathcal{R}_{PQ}$. For the conjugation operation, we set $\tilde{a}_0$ and $\tilde{a}_1$ based on the authority's public key $\mathsf{pk}_a = (b_{\mathsf{auth}}, a_{\mathsf{auth}}) \in \mathcal{R}_{PQ}^2$, where $\tilde{a}_0 = b_{\mathsf{auth}}$ and $\tilde{a}_1 = a_{\mathsf{auth}}$, and prepares $\{(\tilde{b}_i, \tilde{a}_i)\}_{i=0,\cdots,\mathtt{dnum}-1}$ for the conjugation operation. Let $\tilde{e}_i := \tilde{b}_i + \tilde{a}_i \cdot s - P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s(X^{-1})$ for $i = 0, \cdots, \mathtt{dnum} - 1$. If $\|-e_{\mathsf{auth}} \cdot s + \tilde{e}_1 \cdot s_{\mathsf{auth}} + \tilde{e}_0\|_\infty < \frac{P}{2}$ holds, then for the same reasons as in the proof of Theorem 2, the result of SecRec will correctly derive sk_c. Therefore, we will prove that with overwhelming probability, $\|-e_{\mathsf{auth}} \cdot s + \tilde{e}_1 \cdot s_{\mathsf{auth}} + \tilde{e}_0\|_\infty < \frac{P}{2}$ holds.

Let $(b, a) \in \mathcal{R}_Q^2$ represent the ciphertext that encrypts $\Delta \cdot m$ through Enc, such that $b + a \cdot s = \Delta \cdot m$. (The LWE error is now regarded as inserted in the message m, and it does not harm our proof.) The Conj algorithm starts by computing $(b(X^{-1}), a(X^{-1}))$, followed by performing a key-switching operation on $a(X^{-1})$ using conj. After the key-switching operation, let the resulting ciphertext be (b', a'), and let e' represent the error related to the computational accuracy, satisfying the following:

$$b' + a' \cdot s = a(X^{-1}) \cdot s(X^{-1}) + e'.$$

To ensure that the error is within $2^{-\ell}$ for each message, we require that $|e'(\zeta^{5^j})| < \Delta \cdot 2^{-\ell}$ for all $j = 0, \cdots, N/2$, with overwhelming probability. Defining $z_i := e'(\zeta^{5^j})$, we have

$$\mathbf{z}' = \begin{bmatrix} \mathbf{z} \\ \overline{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} z_0 \\ \vdots \\ z_{N/2-1} \\ \bar{z}_0 \\ \vdots \\ \bar{z}_{N/2-1} \end{bmatrix} = \begin{bmatrix} 1 \; \zeta^{5^0} \; \zeta^{5^0 \cdot 2} \cdots \zeta^{5^0 \cdot N-1} \\ \vdots \\ 1 \; \zeta^{5^1} \; \zeta^{5^1 \cdot 2} \cdots \zeta^{5^1 \cdot N-1} \\ \hline 1 \; \bar{\zeta}^{5^0} \; \zeta^{5^0 \cdot 2} \cdots \zeta^{5^0 \cdot N-1} \\ \vdots \\ 1 \; \zeta^{5^1} \; \zeta^{5^1 \cdot 2} \cdots \zeta^{5^1 \cdot N-1} \end{bmatrix} \cdot \begin{bmatrix} e'_0 \\ e'_1 \\ e'_2 \\ \vdots \\ e'_{N-1} \end{bmatrix} = U \cdot \mathbf{e}'.$$

Given the matrix U defined as above, $\frac{1}{\sqrt{N}}U$ is a unitary matrix, so it satisfies $U^{-1} = \frac{1}{N}\bar{U}^T$. Therefore, we have $\mathbf{e}' = \frac{1}{N}\bar{U}^T \cdot \mathbf{z}'$. Since $|\zeta| = 1$, we have $\|\mathbf{e}'\|_\infty \leq \frac{1}{N}\|\bar{U}^T\|_\infty \cdot \|\mathbf{z}'\|_1 < \frac{1}{N} \cdot N \cdot \Delta \cdot 2^{-\ell} = \Delta \cdot 2^{-\ell}$. Thus, the condition $\|e'\|_\infty < \Delta \cdot 2^{-\ell}$ with overwhelming probability is a necessary condition for the precise evaluation of the conjugation operation.

The ciphertext (b', a') after the key-switching operation is computed as follows:

$$(b', a') = P^{-1} \cdot \left(\sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot (\tilde{b}_i, \tilde{a}_i) \right).$$

When calculating $b' + a' \cdot s$, the expression becomes

$$\begin{aligned} b' + a' \cdot s &= P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot (\tilde{b}_i + \tilde{a}_i \cdot s) \\ &= P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot (\tilde{e}_i + P \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s(X^{-1})) \\ &= P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot \tilde{e}_i + \sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot \hat{Q}_i \cdot [\hat{Q}_i^{-1}]_{Q_i} \cdot s(X^{-1})) \\ &= P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot \tilde{e}_i + a(X^{-1}) \cdot s(X^{-1}). \end{aligned}$$

Therefore, for precise evaluation to be possible, the following condition must be satisfied:

$$\|e'\|_\infty = \|P^{-1} \cdot \sum_{i=0}^{\mathtt{dnum}-1} [a(X^{-1})]_{Q_i} \cdot \tilde{e}_i\|_\infty < \Delta \cdot 2^{-\ell}.$$

Here, a is a component of the ciphertext that is extracted independently of conj, and it is computed as $a = v \cdot a_{\mathsf{pk}} + e_0$ for some ephemeral v. Since a is derived from a secret key independent of s, it can be viewed as an LWE sample. From the client's perspective, who knows only s, a is indistinguishable from an

element uniformly sampled from $\mathcal{R}_Q$. Therefore, it is valid to assume that a is sampled from a uniform distribution for the sake of analysis. Given that a follows a uniform distribution, $a(X^{-1})$ will also follow a uniform distribution due to the permutation of coefficients. Thus, for $u_i \leftarrow \mathcal{R}_{Q_i}$, we should have

$$\left\| \sum_{i=0}^{\mathtt{dnum}-1} u_i \cdot \tilde{e}_i \right\|_\infty < P \cdot \Delta \cdot 2^{-\ell} \tag{2}$$

with overwhelming probability.

We aim to prove that the condition $\sum_{i=0}^{\mathtt{dnum}-1} Q_i \|\tilde{e}_i\|_1 \leq P \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell}$ is a necessary condition for Equation (2) with overwhelming probability. To do so, we need to show that if $\sum_{i=0}^{\mathtt{dnum}-1} Q_i \|\tilde{e}_i\|_1 > P \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell}$, then $\Pr\left[\left\|\sum_{i=0}^{\mathtt{dnum}-1} u_i \cdot \tilde{e}_i\right\|_\infty \geq P \cdot \Delta \cdot 2^{-\ell}\right]$ is non-negligible. Note that $\tilde{e}_i$ are fixed values in the current situation. Let $u_i = u_{i,0} + u_{i,1}X + \cdots + u_{i,N-1}X^{N-1}$ and $\tilde{e}_i = \tilde{e}_{i,0} + \tilde{e}_{i,1}X + \cdots + \tilde{e}_{i,N-1}X^{N-1}$. Define $T_{i,j} = u_{i,j} \cdot e_{i,-j}$, where the negative index in $e_{i,-j}$ is considered modulo N. $T_{i,j}$ follows a uniform distribution in the range $\left[-\frac{Q_i}{2} \cdot e_{i,-j}, \frac{Q_i}{2} \cdot e_{i,-j}\right]$, and since the $u_{i,j}$ are independent for different i, j, the $T_{i,j}$ are also independent. We can express the constant term v_0 of $v = \sum_{i=0}^{\mathtt{dnum}-1} u_i \cdot \tilde{e}_i$ as $\sum_{i=0}^{\mathtt{dnum}-1} \sum_{j=0}^{N-1} T_{i,j}$. By applying Lemma 1, we obtain

$$\Pr\left[|v_0| > \frac{1}{N \cdot \mathtt{dnum}} \sum_{i=0}^{\mathtt{dnum}-1} \frac{Q_i}{2} \cdot \|\tilde{e}_i\|_1\right] > \frac{1}{4},$$

Thus, with a probability greater than $\frac{1}{4}$, the following holds:

$$\begin{aligned}\left\| \sum_{i=0}^{\mathtt{dnum}-1} u_i \cdot \tilde{e}_i \right\|_\infty &= \|v\|_\infty \geq |v_0| \\ &> \frac{1}{N \cdot \mathtt{dnum}} \sum_{i=0}^{\mathtt{dnum}-1} \frac{Q_i}{2} \cdot \|\tilde{e}_i\|_1 \\ &> P \cdot \Delta \cdot 2^{-\ell}\end{aligned}$$

Through this, we have proven that the condition $\sum_{i=0}^{\mathtt{dnum}-1} Q_i \|\tilde{e}_i\|_1 \leq P \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell}$ is a necessary condition for Equation 2 with overwhelming probability.

As a result, we can show that $\|\tilde{e}_i\|_1 \leq \frac{P}{Q_i} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell}$. Therefore, with overwhelming probability, we obtain the following inequality,

$$\begin{aligned}
\|-e_{\mathsf{auth}} \cdot s + \tilde{e}_1 \cdot s_{\mathsf{auth}} + \tilde{e}_0\|_\infty \leq & \|e_{\mathsf{auth}} \cdot s\|_\infty + \|\tilde{e}_1 \cdot s_{\mathsf{auth}}\|_\infty + \|\tilde{e}_0\|_\infty \\
\leq & \|e_{\mathsf{auth}}\|_1 \cdot \|s\|_\infty + \|\tilde{e}_1\|_1 \cdot \|s_{\mathsf{auth}}\|_\infty + \|\tilde{e}_0\|_1 \\
\leq & N \cdot \|e_{\mathsf{auth}}\|_\infty \cdot \|s\|_\infty \\
& + \frac{P}{Q_1} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} \cdot \|s_{\mathsf{auth}}\|_\infty \\
& + \frac{P}{Q_0} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} \\
\leq & N \cdot B_e \cdot B_s + \frac{P}{Q_1} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} \cdot B_s \\
& + \frac{P}{Q_0} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} < \frac{P}{2},
\end{aligned}$$

Therefore, by Theorem 1, with overwhelming probability, the SecRec algorithm can successfully recover sk_c. □

5 Feasibility and Performance

The two remaining issues are to verify whether the RNS-CKKS-based HEwA scheme defined above is practical for implementation and to assess its performance compared to the existing RNS-CKKS scheme. In this section, we will confirm that our scheme can be designed to guarantee correctness and security using the current RNS-CKKS scheme. Additionally, we will show that the parameter regime that meets the conditions in the correctness and security proofs for the RNS-CKKS-based HEwA is compatible with that of the original RNS-CKKS setting so that the time performance of the HEwA scheme does not differ significantly from the homomorphic encryption schemes.

5.1 Feasibility of Secure HEwA

The conditions required to ensure correctness in Theorem 1 are as follows,

$$\Pr\left[\|-e_{\mathsf{auth}} \cdot s + e_1 \cdot s_{\mathsf{auth}} + e_0\|_\infty < \frac{P}{2} : s \leftarrow \chi_{\mathsf{sk}}, e_{\mathsf{auth}}, e_0, e_1 \leftarrow \chi_{\mathsf{err}}\right] > 1 - \mathsf{negl}(\lambda).$$

This condition can easily be satisfied. In the case of χ_{err}, it refers to the error distribution used in ring-LWE sampling, typically a normal distribution with $\sigma = 3.2$. For χ_{sk}, a ternary distribution with support $\{-1, 0, 1\}$ is commonly used in homomorphic encryption, meaning that $\|s\|_\infty = 1$ in most cases. Therefore, in a homomorphic encryption system that uses integers with hundreds of bits for P, these conditions are met without issue.

The conditions required to ensure traceability in Theorem 3 are as follows. $\Pr[\|s\|_\infty > B_s \mid s \leftarrow \chi_{\mathsf{sk}}]$ and $\Pr[\|e\|_\infty > B_e \mid e \leftarrow \chi_{\mathsf{err}}]$ are negligible, Q_i's are larger than or equal to 5, and

$$N \cdot B_e \cdot B_s + \frac{P}{Q_1} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} \cdot B_s + \frac{P}{Q_0} \cdot 2N \cdot \mathtt{dnum} \cdot \Delta \cdot 2^{-\ell} < \frac{P}{2}.$$

By setting $B_e = 2^7$ and $B_s = 2$, the two probabilities mentioned above become negligible. Trivially, the modulus q_i's used in CKKS schemes are larger than or equal to 5. Additionally, if $N = 2^{16}$, $\mathtt{dnum} < 2^5$, $\Delta < 2^{60}$, and $\ell > 20$, these values fall within the parameter ranges currently used in most schemes. Since we set the parameters for key-switching correctness with $P \approx Q_i$, we can safely assume $\frac{P}{Q_i} < 2^{10}$. In this case, the left-hand side of the inequality becomes less than 2^{64}. Given that P is an integer with hundreds of bits, this inequality is easily satisfied, ensuring that the condition holds.

5.2 Runtime Performance

The most important aspect of runtime performance is the efficiency of encryption, decryption, and homomorphic operations. Since the computations in these algorithms are identical to those in the existing RNS-CKKS homomorphic encryption scheme, there is no difference in runtime performance. Therefore, there is no runtime trade-off resulting from the new security conditions, as the performance remains unchanged.

6 Conclusion

We extended the traditional client-server model for homomorphic encryption by introducing a new definition of Homomorphic Encryption with Authority (HEwA) based on a client-server-authority model. We proposed the corresponding definitions for correctness and security. Furthermore, we designed a practical HEwA scheme based on the RNS-CKKS scheme that satisfies these definitions. We demonstrated that the parameter conditions required by this scheme are met by the parameters currently used in practical implementations. Additionally, we confirmed that the performance of the HEwA scheme is equivalent to that of the original RNS-CKKS scheme. Future work will focus on applying this model to more complex scenarios, such as multi-key homomorphic encryption and other advanced use cases.

Acknowledgements

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2024-00399491,Development of Privacy-Preserving Multiparty Computation Techniques for Secure Multiparty Data Integration).

References

[1] Ananth, P., Jain, A., Jin, Z., Malavolta, G.: Multi-key fully-homomorphic encryption in the plain model. In: Theory of Cryptography: 18th International Conference, TCC 2020, Durham, NC, USA, November 16–19, 2020, Proceedings, Part I 18. pp. 28–57. Springer (2020)

[2] Boemer, F., Lao, Y., Cammarota, R., Wierzynski, C.: Ngraph-he: A graph compiler for deep learning on homomorphically encrypted data. In: Proceedings of the 16th ACM International Conference on Computing Frontiers. p. 3–13. CF '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3310273.3323047, https://doi.org/10.1145/3310273.3323047

[3] Boudgoust, K., Scholl, P.: Simple threshold (fully homomorphic) encryption from lwe with polynomial modulus. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 371–404. Springer (2023)

[4] Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: Annual Cryptology Conference. pp. 868–886. Springer (2012)

[5] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. Cryptology ePrint Archive, Report 2011/277 (2011), https://eprint.iacr.org/2011/277

[6] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT) **6**(3), 1–36 (2014)

[7] Byun, J., Park, S., Choi, Y., Lee, J.: Efficient homomorphic encryption framework for privacy-preserving regression. Applied Intelligence **53**(9), 10114–10129 (2023)

[8] Chen, H., Chillotti, I., Song, Y.: Multi-key homomorphic encryption from tfhe. In: Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part II 25. pp. 446–472. Springer (2019)

[9] Chen, H., Dai, W., Kim, M., Song, Y.: Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 395–412 (2019)

[10] Cheon, J.H., Han, K., Kim, A., Kim, M., Song, Y.: A full rns variant of approximate homomorphic encryption. Cryptology ePrint Archive, Paper 2018/931 (2018), https://eprint.iacr.org/2018/931, https://eprint.iacr.org/2018/931

[11] Cheon, J.H., Han, K., Kim, A., Kim, M., Song, Y.: A full rns variant of approximate homomorphic encryption. In: Selected Areas in Cryptography – SAC 2018: 25th International Conference, Calgary, AB, Canada, August 15–17, 2018, Revised Selected Papers. p. 347–368. Springer-Verlag, Berlin, Heidelberg (2019). https://doi.org/10.1007/978-3-030-10970-$7_1$6, https://doi.org/10.1007/978-3-030-10970-7_16

[12] Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 409–437. Springer (2017)

[13] Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144 (2012), https://eprint.iacr.org/2012/144

[14] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing. p. 169–178. STOC '09, Association for Computing Machinery, New York, NY, USA (2009). https://doi.org/10.1145/1536414.1536440, https://doi.org/10.1145/1536414.1536440

[15] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 169–178 (2009)

[16] Jain, A., Rasmussen, P.M., Sahai, A.: Threshold fully homomorphic encryption. Cryptology ePrint Archive (2017)

[17] Kim, A., Song, Y., Kim, M., Lee, K., Cheon, J.H.: Logistic regression model training based on the approximate homomorphic encryption. BMC Med Genomics **11**(83) (2018)

[18] Kim, M., Song, Y., Li, B., Micciancio, D.: Semi-parallel logistic regression for gwas on encrypted data. BMC Medical Genomics **13**(7), 99 (2020). https://doi.org/10.1186/s12920-020-0724-z, https://doi.org/10.1186/s12920-020-0724-z

[19] Kirchengast, T.: Deepfakes and image manipulation: criminalisation and control. Information & Communications Technology Law **29**(3), 308–323 (2020)

[20] Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. Journal of the ACM (JACM) **60**(6), 1–35 (2013)

[21] Park, S., Byun, J., Lee, J.: Privacy-preserving fair learning of support vector machine with homomorphic encryption. In: Proceedings of the ACM Web Conference 2022. pp. 3572–3583 (2022)

[22] Park, S., Byun, J., Lee, J., Cheon, J.H., Lee, J.: He-friendly algorithm for privacy-preserving svm training. IEEE Access **8**, 57414–57425 (2020)

[23] Park, S., Kim, J., Lee, J., Byun, J., Cheon, J.H., Lee, J.: Security-preserving support vector machine with fully homomorphic encryption. SafeAI@ AAAI **2301** (2019)

[24] Regev, O.: The learning with errors problem. Invited survey in CCC **7**(30), 11 (2010)

[25] Riehle, C.: Europol report criminal use of deepfake technology. Eucrim. Retrieved June **27**, 2023 (2022)

[26] Sugizaki, Y., Tsuchida, H., Hayashi, T., Nuida, K., Nakashima, A., Isshiki, T., Mori, K.: Threshold fully homomorphic encryption over the torus. In: Tsudik, G., Conti, M., Liang, K., Smaragdakis, G. (eds.) Computer Security – ESORICS 2023. pp. 45–65. Springer Nature Switzerland, Cham (2024)