sensors

*Article*

# A Lattice-Based Homomorphic Proxy Re-Encryption Scheme with Strong Anti-Collusion for Cloud Computing

**Juyan Li [1,2], Zhiqi Qiao [1], Kejia Zhang [1] and Chen Cui [1,*]**

[1] College of Data Science and Technology, Heilongjiang University, Harbin 150080, China; 2018043@hlju.edu.cn (J.L.); qiaozhiqi98@163.com (Z.Q.); zhangkejia@hlju.edu.cn (K.Z.)

[2] Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China

[*] Correspondence: 2018012@hlju.edu.cn

**Abstract:** The homomorphic proxy re-encryption scheme combines the characteristics of a homomorphic encryption scheme and proxy re-encryption scheme. The proxy can not only convert a ciphertext of the delegator into a ciphertext of the delegatee, but also can homomorphically calculate the original ciphertext and re-encryption ciphertext belonging to the same user, so it is especially suitable for cloud computing. Yin et al. put forward the concept of a strong collusion attack on a proxy re-encryption scheme, and carried out a strong collusion attack on the scheme through an example. The existing homomorphic proxy re-encryption schemes use key switching algorithms to generate re-encryption keys, so it can not resist strong collusion attack. In this paper, we construct the first lattice-based homomorphic proxy re-encryption scheme with strong anti-collusion (HPRE-SAC). Firstly, algorithm TrapGen is used to generate an encryption key and trapdoor, then trapdoor sampling is used to generate a decryption key and re-encryption key, respectively. Finally, in order to ensure the homomorphism of ciphertext, a key switching algorithm is only used to generate the evaluation key. Compared with the existing homomorphic proxy re-encryption schemes, our HPRE-SAC scheme not only can resist strong collusion attacks, but also has smaller parameters.

**Keywords:** LWE; homomorphic proxy re-encryption; strong anti-collusion; key switching; trapdoor sampling; cloud computing

**Citation:** Li, J.; Qiao, Z.; Zhang, K.; Cui, C. A Lattice-Based Homomorphic Proxy Re-Encryption Scheme with Strong Anti-Collusion for Cloud Computing. *Sensors* **2021**, *21*, 288. https://doi.org/10.3390/s21010288

Received: 9 December 2020
Accepted: 1 January 2021
Published: 4 January 2021

## 1. Introduction

Lattice-based cryptography is a kind of public key cryptosystem, which is widely believed to resist quantum computer attacks. The lattice-based cryptographic systems have attracted the attention of many scholars, on the one hand, because of the simper linear operation than the power operation that is needed in the traditional theory-based cryptosystems (such as RSA); on the other hand, because their security can be based on worst-case hard problems (such as SIVP, GapSVP). There are two basic average-case problems that had been shown to enjoy worst-case hardness guarantee. One is the learning with error (LWE) problem [1,2] the other one is the small integer solution (SIS) problem [3].

Public-key encryption (PKE) is one of the most fundamental primitives in cryptography. In recent years, some lattice-based PKE schemes were constructed based on LWE and SIS [4–6]. Fully-homomorphic encryption (FHE) is a kind of PKE, but the FHE scheme allows one to compute arbitrary functions over encrypted data without the decryption key. In an FHE scheme, the data owner can obtain ciphertexts $E(m_1), \cdots, E(m_n)$ that encrypts data $m_1, \cdots, m_n$ by encryption key $pk$ (the corresponding decryption key is $sk$), respectively. Anyone can efficiently compute compact ciphertext that encrypts $f(m_1, \cdots, m_n)$ for any efficiently computable function $f$, but only the owner of decryption key $sk$ can get $f(m_1, \cdots, m_n)$ by decrypting the compact ciphertext [7,8]. The interesting property makes FHE more applicable in many scenarios, such as cloud computing [9,10].

With the emerging of the cloud computing, the situation has transformed from a single user to multiple users on one of both communication ends. Most of the existing FHE schemes only allow the user to homomorphically compute ciphertexts that are encrypted by himself. Proxy re-encryption (PRE) [11] is an extension of public key encryption. In a PRE scheme, with the help of the re-encryption key, the proxy can convert the ciphertext of a delegator into the ciphertext of a delegatee. In this process, there is no need to decrypt the ciphertext of the delegator, and the proxy can not get the plaintext. It is very interesting to construct a homomorphic proxy re-encryption (HPRE) scheme, which allows users to homomorphically compute ciphertexts of different users. As shown in Figure 1. After getting the ciphertext $ct_i = E(pk_i, m_i)$ and the re-encryption key $rk_{A_i \to B}$ of $A_i, i = 1, 2, \cdots, n$, the proxy can convert the ciphertext $ct_i$ into the ciphertext $ct_{A_i \to B}$ of B, and guarantee the homomorphism of these re-encryption ciphertexts. That is, if $ct = f(ct_{A_1 \to B}, ct_{A_2 \to B}, \cdots, ct_{A_n \to B})$, then $D(sk_B, ct) = f(m_1, m_2, \cdots, m_n)$, where $pk_i$ is the encryption key of $A_i$, $sk_B$ is the decryption key of $B$, $f$ is an efficiently computable function.

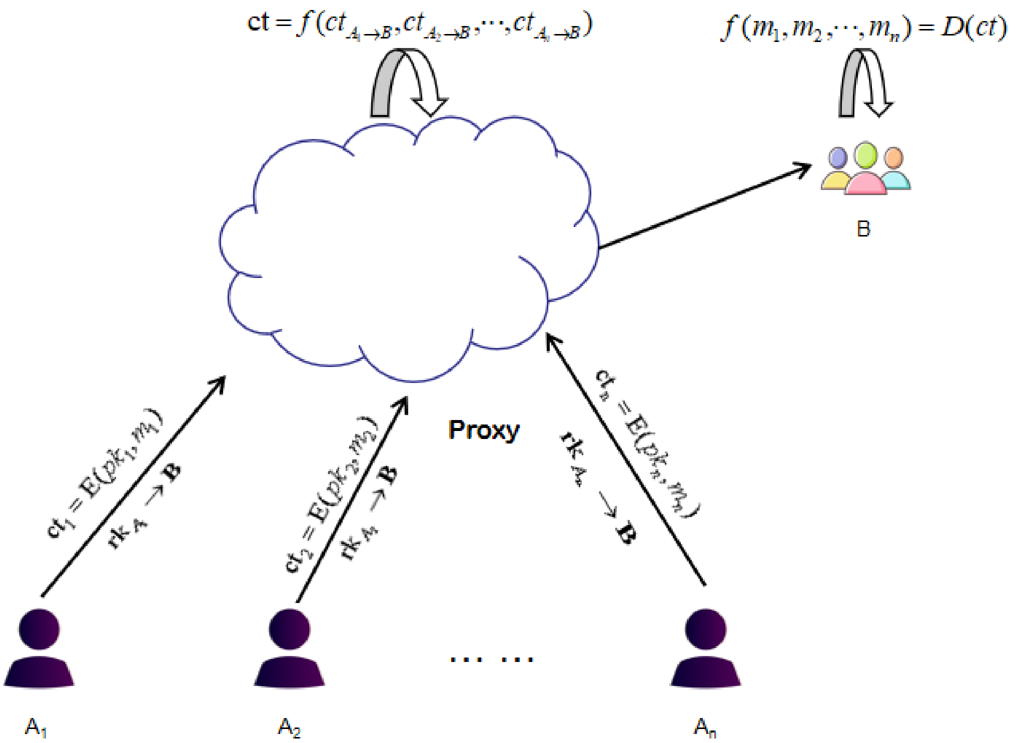

**Figure 1.** The homomorphic proxy re-encryption scheme.

### 1.1. *Related Work*

Proxy Re-Encryption (PRE) was introduced by Bleumer et al. [11], which can be applied in many scenarios, such as encrypted email forwarding, vehicular ad hoc network, the distributed file system [12], and the cloud sharing [13–17]. Many PRE schemes with special properties have been constructed to meet the increasingly complex cloud sharing environment. For example, conditional proxy re-encryption [18,19], which allows only the ciphertexts satisfying a condition to be converted by the proxy; attribute-based proxy re-encryption [20,21], which transforms a ciphertext under an access policy to a ciphertext under another access policy; broadcast proxy re-encryption [22,23], which converts a ciphertext to a set of ciphertexts under different users at a time; unidirectional proxy re-encryption [24,25], in which the proxy can use the re-encryption key to convert the delegator's ciphertext to the delegatee's ciphertext, but cannot reverse the conversion, otherwise it becomes bidirectional; multi-hop proxy re-encryption [26,27], in which the proxy can convert a re-encryption ciphertext into a re-encryption ciphertext of other users, otherwise it becomes single-hop; homomorphic proxy re-encryption (HPRE) scheme [19,28], and so on.

Security is an important index of the practicability of a PRE scheme. At present, the security of a PRE scheme mainly involves post quantum security, semantic security, key privacy, anti-collusion and so on. The construction of PRE can be based on the Diffie–Hellman assumption, but the Diffie-Hellman assumption is not considered post quantum secure. Therefore, it is necessary to construct a PRE based on LWE, because the LWE assumption is generally considered to be able to resist quantum computing attacks. Xagawa [29] constructs the first PRE based on LWE, but the scheme lacks a complete security analysis, and it is bidirectional and can not resist collusion attack. Compared with bidirectional PRE, unidirectional PRE is more in line with the security requirements of cloud sharing. Collusion attack means that the delegatee and the proxy can conspire to compute the decryption key of the delegator.

Aono et al. [30] constructed a unidirectional re-encryption scheme based on LWE and proved that the scheme has key privacy. Key privacy [31] means that even if an active proxy colludes with a set of malicious users in the system, it can not know the identity of the participants involved or the content of their encrypted messages from the re-encryption key. Singh et al. [32] pointed out that the scheme of Aono et al. [30] could not resist collusion attack, and constructed a PRE scheme against collusion attack based on [30]. Kirshanova [33] constructed the first chosen ciphertext attack (CCA) secure lattice-based PRE scheme. Nishimaki et al. [34] constructed two unidirectional single-hop key privacy PRE schemes based on LWE and proved the two schemes are chosen plaintext attack (CPA) secure. Hou et al. [35] constructed an efficient identity-based PRE over lattice and proved that the scheme is CPA secure in the standard model, but the scheme is bidirectional and cannot resist collusion attack. Yin et al. [36] constructed a unidirectional identity based PRE under LWE, and proved that the scheme is CPA secure in the standard model. Yin et al. [37] put forward the concept of a strong collusion attack (the strong collusion attack will be shown in Definition 7) relative to a traditional collusion attack, and called it a traditional collusion attack as weak collusion attack. Yin et al. pointed out through examples that if the adversary can not collude to attack the decryption key of the delegator, but can obtain an approximate value of the decryption key of the delegator, then it can also launch a strong collusion attack on the scheme of Aono et al. [30] and correctly decrypt the ciphertext of the delegator.

Zhong et al. [38] constructed a many-to-one homomorphic encryption scheme based on an approximate GCD problem, which can apply homomorphic addition and homomorphic multiplication to multi-party ciphertexts. However, the scheme is not a lattice-based scheme. Since its introduction, FHE [7,8] has attracted much attention and some FHE schemes have been constructed based on LWE. Since the noise is added at encryption for security, the noise will increase with every homomorphic operation in the FHE scheme based on LWE. For correct decryption, the magnitude of final noise must be less than some bound. How to control noise is an important issue. A number of techniques are proposed and used to control noise growth for building an FHE scheme based on LWE , for example, Brakerski et al. [39] proposed the re-linearization technique and the dimension modulus reduction technique; Brakerski et al. [40] proposed the modulus switching algorithm, Brakerski [41] proposed the scale-invariant technique; Gentry et al. [42] proposed the approximate eigenvector method. In addition, these techniques are also the main techniques for constructing homomorphic proxy re-encryption schemes to control noise growth.

Jiang et al. [26] based on [43] constructed a multi-hop unidirectional lattice-based proxy re-encryption. The scheme can only support one multiplicative homomorphic operation. Ma et al. [19,28] based on [42] constructed a single-hop homomorphic proxy re-encryption from lattices, which allows a user to homomorphically evaluate the original ciphertexts and the re-encrypted ciphertexts, which can come from different users. Li et al. [44,45] constructed a single-hop homomorphic proxy re-encryption via key homomorphic computation and obtained a multi-hop proxy re-encryption using a branching program. Li et al. [46] based on [47] constructed a homomorphic proxy re-encryption from a lattice, which is more flexible than [19,28]. All of these HPR schemes are CPA secure and can not resist strong collusion attack. For the sake of comparison, the comparison results are given in Table 1, which shows the comparison of these PRE schemes in LWE assumption, semantic security, multi-hop, unidirectional-direction (uni-direction), homomorphic encryption (HE) and strong anti-collusion. In this paper, we will construct a lattice-based homomorphic proxy re-encryption scheme with strong anti-collusion. Table 1 shows that our scheme meets all the above performance.

**Table 1.** Comparison of lattice-based proxy re-encryption (PRE) schemes.

| Scheme | LWE | Security | Hop | Direction | HE | Anti-Collusion |
|---|---|---|---|---|---|---|
| Li et al.[20] | Y | CPA | Single | Uni- | N | weak |
| Singh et al. [25] | Y | CPA | Single | Uni- | N | weak |
| Xagawa[29] | Y | N | Single | Uni- | N | weak |
| Kirshanova[33] | Y | CCA | Single | Uni- | N | strong |
| Nishimaki et al.[34] | Y | CPA | Single | Uni- | N | weak |
| Hou et al.[35] | Y | CPA | Multi | Bi- | N | weak |
| Yin et al.[36] | Y | CPA | Single | Uni- | N | strong |
| Yin et al. [37] | Y | CPA | Single | Uni- | N | strong |
| Jiang et al. [26] | Y | CPA | Multi | Uni- | N | strong |
| Ma et al.[28] | Y | CPA | Single | Uni- | Y | weak |
| Li et al. [44] | Y | CPA | Multi | Uni- | Y | weak |
| Li et al. [45] | Y | CPA | Multi | Uni- | Y | weak |
| Li et al. [46] | Y | CPA | Single | Uni- | Y | weak |
| Our Scheme FHPR-SAC | Y | CPA | Multi | Uni- | Y | strong |

Y indicates that the scheme has been achieved and N indicates that the scheme has not been achieved.

### *1.2. Our Contribution*

At present, there are two main methods to construct the re-encryption key in the lattice-based proxy re-encryption scheme. One is to use the key switching algorithm (see Lemma 6) and the other is to use trapdoor sampling technology (see Lemma 3). In fact, the key switching algorithm uses the delegatee's encryption key to encrypt the delegator's decryption key and hides the decryption key by noise. Therefore, when the delegatee colludes with the proxy, an approximate value of the delegator's decryption key can be recovered, that is, the sum of the decryption key and the decryption noise. Thus, the re-encryption key constructed by the key switching algorithm can only resist weak collusion attack, but not strong collusion attack. However, trapdoor sampling technology does not allow inverse operation, that is, we can not get $T_A$ or approximate value of $T_A$ by $\vec{x}, A, \vec{u}, \sigma, \vec{c}$, where $\vec{x} \leftarrow \text{SamplePre}(A, T_A, \vec{u}, \sigma, \vec{c})$, so it can resist strong collusion attack.

Because HPRE schemes need to be constructed based on basic homomorphic encryption schemes, and lattice based on homomorphic encryption schemes mostly use a key switching algorithm, modulus switching technique and approximate eigenvector method to control the growth of homomorphic multiplication ciphertext noise, so the current HPRE [28,44–46] schemes are constructed based on a key switching algorithm to generate a re-encryption key. The key switching algorithm can not only generate a re-encryption key, but also ensure the homomorphism of ciphertext. However, the re-encryption key generated by key switching algorithm can only resist weak anti-collusion, but not strong anti-collusion. However, the re-encryption key generated by trapdoor sampling technology can resist strong anti-collusion, but it cannot satisfy the homomorphism of ciphertext. This is the difficulty of the HPRE scheme with strong anti-collusion constructed in this paper. Therefore, it is necessary to use trapdoor sampling technology to generate a re-encryption key satisfying the homomorphism of the ciphertext.

In this paper, the ciphertext is divided into two parts, one of which is used to encrypt the plaintext, while ensuring the homomorphism of the ciphertext. In the other part of ciphertext, trapdoor sampling technology can be used to generate the re-encryption key.

Therefore, it is necessary to modify the existing homomorphic encryption scheme to make the ciphertext meet the above two requirements.

(1) Firstly, we use the trapdoor technology of [48] to modify the scheme of [1] and construct an L-homomorphic encryption scheme.
(2) Then, based on the L-homomorphic encryption scheme proposed in this paper, we construct an HPRE-SAC scheme by using trapdoor sampling technology and a key switching algorithm.
(3) Finally, a direct application of the HPRE-SAC scheme is given, that is, secure computing of personal health records (PHRs) in the cloud.

Compared with the existing HPRE schemes [28,44–46], our HPRE-SAC scheme not only can resist the strong collusion attack, but also has smaller parameters. Therefore, it is more suitable for cloud computing scenarios.

### *1.3. Paper Organization*

The rest of this paper is organized as follows. Section 2 is preliminaries. Section 3 describes the building blocks. Section 4 describes a $L$- Homomorphic Encryption Scheme. Section 5 describes the HPRE-SAC Scheme. Lastly, our work is concluded in Section 6.

## 2. Preliminaries

We employ some initial notations listed in Table 2 and let $\mathbb{Z}_q = [-q/2, q/2) \cap \mathbb{Z}$. When $A$ is a matrix, let $P2(A)$ be the matrix formed by applying the operation to each column of $A$.

**Table 2.** Notation.

| | |
|---|---|
| $x$ | scalar |
| $\lfloor x \rceil$ | rounding $x$ to the nearest integer |
| $\lfloor x \rfloor (\lceil x \rceil)$ | rounding down (up) |
| $\vec{x}$ | vector |
| $A$ | matrix or set |
| $\|\|\vec{x}\|\|_p$ | $l_p$ norm of $\vec{x}$ |
| $P2(\vec{x})$ | $\left(1\vec{x}; 2\vec{x}; \cdots; 2^{\lceil logq \rceil - 1}\vec{x}\right) \in \mathbb{Z}_q^{n\lceil logq \rceil}$, where $\vec{x} \in \mathbb{Z}_q^n$ |
| $BD(\vec{x})$ | $\left(\vec{u}_1, \cdots, \vec{u}_{\lceil logq \rceil}\right) \in \{0,1\}^{n\lceil logq \rceil}$, where $\vec{x} = \sum_{k=1}^{\lceil logq \rceil} 2^{k-1}\vec{u}_k$ |
| $(X \mid Y)$ | the concatenation of the columns of $X, Y$ |
| $(X; Y)$ | the concatenation of the rows of $X, Y$ |
| $x \leftarrow \chi$ | $x$ is sampled according to a probability distribution $\chi$ |
| $x \leftarrow S$ | $x$ is sampled uniformly from a set S |
| $X \approx_c Y$ | $X$ and $Y$ are computationally indistinguishable |
| $X \approx_s Y$ | $X$ and $Y$ are statistically indistinguishable |

### *2.1. Lattice and Gaussian Distributions*

In this section, we introduce the lattice, Gaussian distribution and some properties needed to construct the scheme.

**Definition 1.** *Let $q$ be a prime, $A \in \mathbb{Z}_q^{n \times m}$, $\vec{u} \in \mathbb{Z}_q^n$, define:*

$$\Lambda_q^{\perp}(A) = \{\vec{e} \in \mathbb{Z}^m, s.t. A\vec{e} = 0 \bmod q\}$$

$$\Lambda_q^{\vec{u}}(A) = \{\vec{e} \in \mathbb{Z}^m, s.t. A\vec{e} = \vec{u} \bmod q\}$$

**Lemma 1** ([49])**.** *Let $q \geq 2$ and $m \geq 6nlogq > 0$. There is a probabilistic polynomial-time (PPT) algorithm $TrapGen(q, n, m)$ that outputs matrixes $A \in \mathbb{Z}_q^{n \times m}$ which is statistically close to uniform in $\mathbb{Z}_q^{n \times m}$ and $T \in \mathbb{Z}^{m \times m}$ which is a basis for $\Lambda_q^{\perp}(A)$ with $\|T\| \leq O(nlogq)$ and*

$\left\|\widetilde{T}\right\| \leq O(\sqrt{n \log q})$ *(Alwen et al. asserted that the constant hidden in the first $O(\cdot)$ is no more than 20).*

For any positive parameter $\sigma > 0$, define the Gaussian function on $\mathbb{R}^m$, centered at $\vec{c}$: $\forall \vec{x} \in \mathbb{R}^m$,

$$\rho_{\sigma,\vec{c}} = \exp\left(-\pi\|\vec{x} - \vec{c}\|^2 \big/ \sigma^2\right).$$

Let $\Lambda$ be a discrete subset of $\mathbb{Z}^m$. Define the discrete Gaussian distribution over $\Lambda$ as: $\forall \vec{x} \in \mathbb{R}^m$,

$$D_{\Lambda,\sigma,\vec{c}}(\vec{x}) = \frac{\rho_{s,\vec{c}}(\vec{x})}{\rho_{\sigma,\vec{c}}(\Lambda)},$$

where $\rho_{\sigma,\vec{c}}(\Lambda) = \sum_{\vec{x}\in\Lambda} \rho_{\sigma,\vec{c}}(\vec{x})$.

**Lemma 2** ([50])**.** *Let $\vec{x} \leftarrow D_{\mathbb{Z}^m,\sigma}$, then with overwhelming probability $\|\vec{x}\|_2 < \sigma\sqrt{m}$, where $\sigma > 0$,.*

**Lemma 3** ([48])**.** *Let $q \geq 2$, $T_A$ be a basis for $\Lambda_q^{\perp}(A)$, where $\sigma \geq \left\|\widetilde{T}\right\|\omega(\sqrt{\log m})$, $A \in \mathbb{Z}_q^{n\times m}$. Then for any $\vec{c} \in \mathbb{R}^m$ and $\vec{u} \in \mathbb{Z}_q^n$, there is a PPT algorithm SamplePre($A, T_A, \vec{u}, \sigma, \vec{c}$) that outputs $\vec{x} \in \Lambda_q^{\vec{u}}(A)$ which is statistically close to $D_{\Lambda_q^{\vec{u}}(A),\sigma,\vec{c}}$.*

**Lemma 4** ([48])**.** *The algorithm SamplePre($A, T_A, \vec{u}, \sigma, \vec{c}$) gives a collection of trapdoor one-way functions with preimage sampling, if $ISIS_{q,m,\sigma\sqrt{m}}$ is hard on average. Furthermore, it gives a collection of trapdoor collision-resistant hash functions with preimage sampling, if $ISIS_{q,m,\sigma\sqrt{m}}$ is hard on the average.*

**Definition 2** ([1])**.** *Let $k$ be the security parameter, and $\chi = \chi(k)$ be a distribution over $\mathbb{Z}_q$. The* LWE$_{n,m,q,\chi}$ *assumption shows that, if $A \leftarrow \mathbb{Z}_q^{m\times n}$, $\vec{s} \leftarrow \mathbb{Z}_q^n$, $\vec{e} \leftarrow \chi^m$, $\vec{u} \leftarrow \mathbb{Z}_q^m$, then*

$$(A, A\vec{s} + \vec{e}) \approx_c (A, \vec{u}).$$

It is well known that if $\chi^m = D_{\mathbb{Z}^m,\alpha q}$, then when $\alpha q \geq 2\sqrt{n}$, this decision LWE problem is at least as hard as approximating several problems on $n$-dimensional lattices $\Lambda$ in the worst-case to within $\widetilde{O}(n/\alpha)$ factors with a quantum computer.

### 2.2. HE: Definition and Security

In this section, we show the definition and security model of the homomorphic encryption (HE) scheme based on [41].

**Definition 3.** *(Homomorphic encryption scheme)*

*A homomorphic encryption scheme consists of the following five algorithms:*

1. *$HE.Setup(1^k) \rightarrow pp$ : Input the security parameter k. Output the public parameters pp.*
2. *$HE.KeyGen(pp) \rightarrow (pk, sk, evk)$ : Input the public parameters pp. Output the encryption key pk , the public evaluation key evk and the decryption key sk.*
3. *$HE.Enc(pp, pk, \mu) \rightarrow ct$ : Input pp, pk, and a message $\mu \in \{0, 1\}$. Output a ciphertext ct.*
4. *$HE.Eval(pp, f, ct_1, \cdots, ct_l, evk) \rightarrow ct_f$: Input $pp, ct_1, \cdots, ct_l, evk$ and a function $f$: $\{0, 1\}^l \rightarrow \{0, 1\}$. Output a ciphertext $ct_f$. (We consider homomorphic addition $-Add(ct_1, ct_2, evk) \rightarrow ct_{add}$ and multiplication $-Mult(ct_1, ct_2, evk) \rightarrow ct_{mult}$ of depth L arithmetic circuits f over $GF(2)$ in a gate-to-gate manner.)*
5. *$HE.Dec(pp, sk, ct) \rightarrow \mu$ : Input pp, sk and ciphertext ct under secret key sk. Output the message $\mu$.*

Compared with the public key encryption scheme, the adversary obtains not only $pk$ but also $evk$ in the HE scheme. If the homomorphic encryption scheme is still semantically secure when the adversary obtains $pk$ and $evk$, it is said that the HE scheme is secure. The security model of HE scheme is omitted here.

**Definition 4.** *(L-homomorphism) If for any depth $L = L(k)$ arithmetic circuit $f$ over $GF(2)$ and any set of inputs $\mu_1, \cdots, \mu_l \in \{0,1\}$, it holds that*

$$HE.Dec\Big(HE.Eval(pp, f, ct_1, \cdots, ct_l)\Big) = f(\mu_1, \cdots, \mu_l)$$

*with overwhelming probability of $k$, where $(pk, sk, evk) \leftarrow HE.KeyGen(pp)$, $ct_i \leftarrow HE.Enc(pp, pk, \mu_i)$. Then the HE scheme is L-homomorphic.*

### 2.3. HPRE: Definition and Security Model

In this subsection, we recall the definition and the security model of the homomorphic proxy re-encryption (HPRE) scheme. There are four participants in the unidirectional HPRE scheme for cloud sharing, as shown in Figure 2.

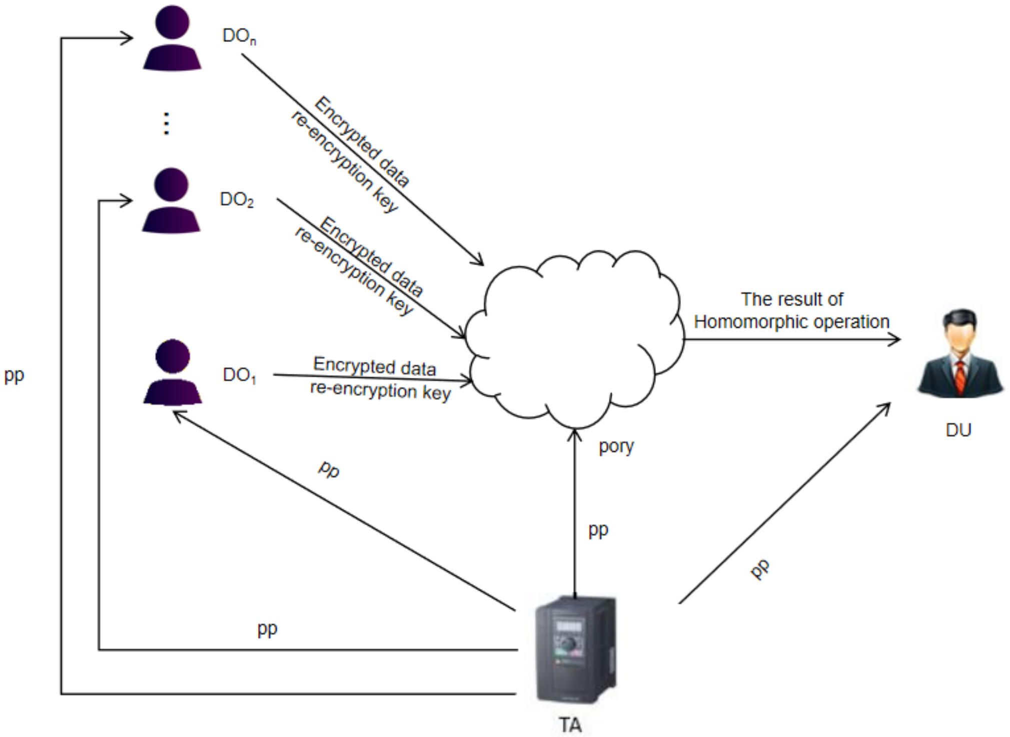

**Figure 2.** System model of the homomorphic proxy re-encryption (HPRE) scheme.

(1) Trusted authority (TA). The TA is trusted by all participants. TA generates the public parameters $pp$.

(2) Proxy. The proxy is semi-trusted by all participants. Proxy is generally a cloud service provider. Users use the cloud service provider to store and calculate data.

(3) Data owner (DO). The DO encrypts the data and stores the encrypted data in the cloud, and generates a proxy re-encryption key for data users.

(4) Data user (DU). The DU downloads the result of the homomorphic operation from the cloud service provider.

**Definition 5.** *(Unidirectional homomorphic proxy re-encryption scheme )*

*A unidirectional HPRE scheme consists of the following seven algorithms:*

1. *HPRE.Setup*$(1^k, 1^L) \to pp$: *For the security parameter k, the upper bound on the maximal multiplicative depth* $L = L(k)$ *that the scheme can homomorphically evaluate, the TA outputs the public parameters* $pp$.
2. *HPRE.KeyGen*$(pp, L) \to (pk^i, sk^i, evk^i)$: *For* $pp, L$*, user i (DO or DU) outputs an encryption/decryption key pair* $(pk^i, sk^i)$*, and public evaluation key* $evk^i$ .
3. *HPRE.Enc*$(pp, pk, \mu) \to ct$: *For* $pp$*, pk and a message* $\mu$*, user (DO or DU) outputs an original ciphertext ct.*
4. *HPRE.Rekey*$(pp, sk^i, pk^i, pk^j) \to rk^{i \to j}$: *For pp, an encryption/decryption key pair* $(pk^i, sk^i)$ *of user i, and an encryption key* $pk^j$ *of user j, user i outputs a re-encryption key* $rk^{i \to j}$.
5. *HPRE.ReEnc*$(pp, rk^{i \to j}, ct^i) \to ct^j$ : *For* $pp$*, a re-encryption key* $rk^{i \to j}$*, and an original ciphertext* $ct^i$ *of user i, the proxy outputs a re-encryption ciphertext* $ct^j$ *for the user j.*
6. *HPRE.Eval*$(pp, f, ct_1, \cdots, ct_l, evk) \to ct_f$: *For* $pp, ct_1, \cdots, ct_l, evk$ *and a function f:* $\{0,1\}^l \to \{0,1\}$*, the proxy outputs a ciphertext* $ct_f$*. (We consider homomorphic addition* $-Add(ct_1, ct_2, evk) \to ct_{add}$ *and multiplication* $-Mult(ct_1, ct_2, evk) \to ct_{mult}$ *of depth L arithmetic circuits f over* $GF(2)$ *in a gate-to-gate manner. In addition, it should be noted that the ciphertexts* $ct_1, \cdots, ct_l$ *belonging to a user can be original ciphertext or re-encryption ciphertext.)*
7. *HPRE.Dec*$(pp, sk, ct) \to \mu$: *For* $pp, sk$ *and a ciphertext ct under sk, user outputs the message* $\mu$.

Now we define the security model of the HPRE scheme.

**Definition 6.** *Let HPRE=(HPRE.Setup, HPRE.KeyGen, HPRE.Enc, HPRE.Rekey, HPRE.ReEn, HPRE.Eval, HPRE.Dec) be a unidirectional HPRE scheme , k be a security parameter. Consider the following games* $Expt^{IND-CPA}_{HPRE,\mathcal{A}}(k)$ *between challenger and adversary.*

*Setup Phase 1: Given a security parameter k, the challenger obtains public parameters pp by running HPRE.Setup*$(1^k, 1^L)$ *and sends pp to adversary.*

*Learning Phase: In this phase, the adversary can issue the queries to the following oracles polynomially many times, and the challenger needs to answer these oracles.*

*Encryption key generation oracle* $\mathcal{O}_{pk}$*: Given a user index i, the challenger obtains* $(pk^i, sk^i, evk^i)$ *of user i by running HPRE.KeyGen* $(pp, L)$ *which are recorded in a table, and returns* $pk^i$ *to the adversary.*

*Evaluation key generation oracle* $\mathcal{O}_{evk}$*: Given a user index i, the challenger first looks for the table and returns* $evk^i$ *if there is an* $evk^i$ *in the table. Otherwise, the challenger obtains* $(pk^i, sk^i, evk^i)$ *of user i by running HPRE.KeyGen*$(pp, L)$*, returns* $evk^i$ *to the adversary, and records* $(pk^i, sk^i, evk^i)$ *in the table.*

*Decryption key generation oracle* $\mathcal{O}_{sk}$*: Given a user index i, if user i is an honest user, the challenger returns* $\perp$*. If user i is a corrupted user, the challenger first looks for the table and returns* $sk^i$ *if there is a* $sk^i$ *in the table. Otherwise, the challenger obtains* $(pk^i, sk^i, evk^i)$ *of user i by running HPRE.KeyGen*$(pp, L)$*, returns* $sk^i$ *to the adversary, and records* $(pk^i, sk^i, evk^i)$ *in the table.*

*Re-encryption key generation oracle* $\mathcal{O}_{rk}$*: Given two user indices* $(i, j)$*, if user i and user j are honest or corrupted, the challenger obtains* $rk^{i \to j}$ *by running HPRE.Rekey*$(pp, sk^i, pk^i, pk^j)$*, and returns the* $rk^{i \to j}$ *to the adversary, where* $i \neq j$*. Otherwise, the challenger returns* $\perp$.

*Re-encryption ciphertext generation oracle* $\mathcal{O}_{re}$*: Given two user indices* $(i, j)$ *and a ciphertext* $ct^i$ *of user i, if user i and user j are honest or corrupted, the challenger obtains a ciphertext* $ct^j$ *of user j by running HPRE.ReEnc*$(pp, rk^{i \to j}, ct^i)$ *and returns* $ct^j$ *to the adversary, where* $i \neq j$*,* $rk^{i \to j} \leftarrow$ *HPRE.Rekey*$(pp, sk^i, pk^i, pk^j)$*. Otherwise, the challenger returns* $\perp$.

*Challenge: The adversary gives a target honest user* $i^*$ *and a message* $\mu$ *after finishing all queries. The challenger chooses* $b \leftarrow \{0,1\}$*, computes* $ct_0^{i*} \leftarrow HPRE.Enc(pp, pk, \mu)$*, lets* $ct_1^{i*}$ *be a random ciphertext, and sends the challenge ciphertext* $ct_b^{i^*}$ *to the adversary.*

*Learning Phase 2: The adversary could ask extra queries that for decryption key query, re-encryption key query and re-encryption query on the* $i \neq i^*$*, the challenger responses are the same as in Learning Phase 1.*

*Finalization: Output 1 if* $b' = b$*. Otherwise, output 0.*

*We say a unidirectional HPRE scheme is IND-CPA secure if for any PPT adversary, the advantage*

$$Adv_{FHPRE,A}^{IND-CPA}(k) = \left| \begin{array}{r} \Pr\left[Expt_{FHPRE,A}^{IND-CPA}(k) \rightarrow 1 | b=1\right] \\ - \Pr\left[Expt_{FHPRE,A}^{IND-CPA}(k) \rightarrow 1 | b=0\right] \end{array} \right|$$

*of adversary is negligible in k.*

Yin et al. [37] put forward the concept of strong collusion attack relative to traditional collusion attack, and called traditional collusion attack as weak collusion attack. Yin et al. pointed out through examples that if the adversary can not collude to attack the decryption key of the delegator, but can obtain an approximate value of the decryption key of the delegator, then it can also launch a strong collusion attack on the scheme of Aono et al. [30] and correctly decrypt the ciphertext of the delegator. In fact, the approximate value of the decryption key obtained by the strong collusion attack is $P2(S) + X$, where $S$ is the decryption key of the delegator, and $X$ is an error distribution (generally Gaussian distribution). Therefore, an approximate value of $S$ can be obtained. Combined with the definition of a strong collusion attack of Yin et al. [37], we give a new definition of strong collusion attack.

**Definition 7.** *In a unidirectional proxy re-encryption scheme, if the proxy (cloud service provider) and the delegatee (data user) can not collude to obtain the decryption key S or an approximate value $P2(S) + X$ of the decryption key of the delegator (data owner), the scheme is called strong anti-collusion, where X is an error distribution. If the decryption key S can not be calculated by collusion, but the approximate value $P2(S) + X$ of the decryption key can be obtained, it is called weak anti-collusion, where X is an error distribution.*

## 3. Building Blocks

In this section, we construct a new encryption scheme based on [1,48]. Based on this new basic encryption scheme, we can construct a homomorphic proxy re-encryption (HPRE) scheme against strong collusion attack, which is named HPRE-SAC .

### 3.1. The Basic Encryption Scheme

The basic encryption scheme consists of the following four algorithms.

- $E.Setup(1^k)$ : Input the security parameter $k$, sample $\vec{u} \leftarrow \mathbb{Z}_q^n$. Output the public parameters $pp = (1^k, 1^n, q, \chi, \vec{u})$.
- $E.KeyGen(pp)$ : Input the public parameters $pp$, use algorithm $\mathrm{TrapGen}(q,n,m)$ to generate matrices $A \in \mathbb{Z}_q^{n\times m}$ with trapdoor basis $T$, where $m \geq 6n\log q$. Then use algorithm $\mathrm{SamplePre}(A, T, \vec{u})$ to sample a vector $\vec{s} \in \mathbb{Z}_q^m$, where $A\vec{s} = \vec{u}$. Output the encryption key $pk = (\vec{u}| - A)$ and the decryption key $sk = (\vec{s}, T)$. (Note that the decryption key $T$ is redundant here, we can instead just let $sk = \vec{s}$. The decryption key $T$ is needed to construct the PRE scheme, as described below.)
- $E.Enc(pp, pk, \mu)$: Input the public parameters $pp$, the encryption key $pk = (\vec{u}| - A)$ and a message $\mu \in \{0, 1\}$. Output a ciphertext $ct \in \mathbb{Z}_q^{1\times m+1}$,

$$ct = \vec{e}^t(\vec{u}| - A) + \vec{y}^t + \lfloor \frac{q}{2} \rfloor \vec{\mu}^t,$$

  where $\vec{\mu}^t = (\mu, 0, \cdots, 0)$, $\vec{e} \leftarrow \chi^{1\times n}$, $\vec{y} \leftarrow \chi^{1\times m+1}$.
- $E.Dec(pp, sk, ct)$: Input the public parameters $pp$, the decryption key $sk = (\vec{s}, T)$ and a ciphertext $ct$. Compute and output

$$\mu = \left[\left\lfloor \frac{2}{q} [ct(1; \vec{s})]_q \right\rceil\right]_2.$$

### 3.2. Correctness Analysis

We show the correctness in this subsection.

For a ciphertext a ciphertext $ct = \vec{e}^t(\vec{u}|A) + \vec{y}^t + \lfloor \frac{q}{2} \rfloor \vec{\mu}^t$, where $\vec{\mu} = (\mu, 0, \cdots, 0)$, $\vec{e} \leftarrow \chi^{1\times n}, \vec{y} \leftarrow \chi^{1\times m+1}$. We have

$$\begin{aligned} ct(1;\vec{s}) &= \left(\vec{e}^t(\vec{u}| - A) + \vec{y}^t + \lfloor \frac{q}{2} \rfloor \vec{\mu}^t\right)(1;\vec{s}) \\ &= \vec{e}^t(\vec{u}| - A)(1;\vec{s}) + \vec{y}^t(1;\vec{s}) + \lfloor \frac{q}{2} \rfloor \vec{\mu}^t(1;\vec{s}) \\ &= \underbrace{\vec{y}^t(1;\vec{s})}_{x} + \lfloor \frac{q}{2} \rfloor \mu. \end{aligned}$$

If $\|x\|_\infty < \lfloor \frac{q}{2} \rfloor / 2$, then the decryption is correct.

For the correctness of this scheme, it needs to satisfy the following conditions:

(1) $\|x\|_\infty = \|\vec{y}^t(1;\vec{s})\|_\infty < \lfloor \frac{q}{2} \rfloor / 2$
(2) Algorithm TrapGen requires $m \geq 6n \log q$.
(3) Algorithm SamplePre requires $\sigma \geq \left\|\widetilde{T}\right\| \omega(\sqrt{\log m})$.

Because $\|T\|<O(n \log q)$, $\|\vec{y}\|, \|\vec{s}\| \leq \sigma\sqrt{m}$, we set the parameters as follows: $n = k$, $q$=the prime nearest to $2^{n^\delta}$, $m = 6n\lceil \log q \rceil$, $\sigma = m\omega(\sqrt{\log m})$, where $\delta$ is constant between 0 and 1. So we have the following Lemma 5.

**Lemma 5.** *Let $q, k, m, n$ be parameters for the above basic encryption scheme, $\chi$ be B-bounded. Set $(\vec{s}, A) \leftarrow E.KeyGen(pp)$, and $ct \leftarrow E.Enc(pp, pk, \mu)$. Then*

$$ct(1;\vec{s}) = x + \lfloor \frac{q}{2} \rfloor \mu,$$

*where $\|x\|_\infty = \|\vec{y}^t(1;\vec{s})\|_\infty \leq (m+1)B^2$. If $(m+1)B^2 < \lfloor \frac{q}{2} \rfloor / 2$, then $\mu \leftarrow E.Dec(pp, sk, ct)$.*

### 3.3. Security Analysis

We now outline the proof of security to show that the scheme is CPA secure based on LWE assumption. Since $\vec{u} \leftarrow \mathbb{Z}_q^n$, and $A \leftarrow$ TrapGen$(q, n, m)$, we have $(\vec{u}|A)$ uniformly distributed by Lemma 1. From LWE, we know that $\vec{e}^t(\vec{u}| - A) + \vec{y}^t$ is uniformly distributed and $ct$ hides $\lfloor \frac{q}{2} \rfloor \vec{\mu}^t$. Therefore, the basic encryption scheme is IND-CPA secure.

### 3.4. Key Switching

Based on the technology of [41], and the basic encryption scheme, we construct a key switching algorithm, which can switch the ciphertext under the decryption key $\vec{s}_1 \in \mathbb{Z}_q^{n_1}$ into the ciphertext under the decryption key $(1;\vec{s}_2) \in \mathbb{Z}_q^{(n_2+1)}$.

- *SwitchKeyGen*$(\vec{s}_1, \vec{s}_2)$ : Input decryption keys $\vec{s}_1 \in \mathbb{Z}_q^{n_1}$, $\vec{s}_2 \in \mathbb{Z}_q^{n_2}$. Sample $A_{\vec{s}_1:\vec{s}_2} \leftarrow \mathbb{Z}_q^{n_1\lceil logq \rceil \times n_2}$ , $\vec{x}_{\vec{s}_1:\vec{s}_2} \leftarrow \chi^{n_1\lceil logq \rceil}$, compute

  $$\vec{b}_{\vec{s}_1:\vec{s}_2} = A_{\vec{s}_1:\vec{s}_2}\vec{s}_2 + \vec{x}_{\vec{s}_1:\vec{s}_2} + P2(\vec{s}_1).$$

  Output a matrix

  $$P_{\vec{s}_1:\vec{s}_2} = (\vec{b}_{\vec{s}_1:\vec{s}_2}| - A_{\vec{s}_1:\vec{s}_2}) \in \mathbb{Z}_q^{n_1\lceil logq \rceil \times (1+n_2)}.$$

- *SwitchKey*$(P_{\vec{s}_1:\vec{s}_2}, ct_{S_1})$ : Input a ciphertext $ct_{\vec{s}_1}$ under the decryption key $\vec{s}_1$, and $P_{\vec{s}_1:\vec{s}_2}$. Output a ciphertext

  $$ct_{\vec{s}_2} = P^t_{\vec{s}_1:\vec{s}_2} BD(ct_{\vec{s}_1}).$$

**Lemma 6.** *(correctness) Let* $\vec{s}_1 \in \mathbb{Z}_q^{n_1}, \vec{s}_2 \in \mathbb{Z}_q^{n_2}$*. Let* $P_{\vec{s}_1:\vec{s}_2} \leftarrow SwitchKeyGen(\vec{s}_1, \vec{s}_2)$ *and* $ct_{\vec{s}_2} \leftarrow SwitchKey(P_{\vec{s}_1:\vec{s}_2}, ct_{\vec{s}_1})$*. Then*

$$ct^t_{\vec{s}_1}\vec{s}_1 = ct^t_{\vec{s}_2}(1;\vec{s}_2) - BD(ct^t_{\vec{s}_1})\vec{x}_{\vec{s}_1:\vec{s}_2}.$$

**Lemma 7.** *(security) Let* $\vec{s}_1 \in \mathbb{Z}_q^{n_1}$ *be any vector, if* $\vec{s}_2 \in \mathbb{Z}_q^{n_2} \leftarrow E.KeyGen(pp)$*,* $P_{\vec{s}_1:\vec{s}_2} \leftarrow SwitchKeyGen(\vec{s}_1, \vec{s}_2)$*. Then* $P_{\vec{s}_1:\vec{s}_2}$ *is computationally indistinguishable from uniform over* $\mathbb{Z}_q^{n_1\lceil logq \rceil \times (1+n_2)}$ *based on LWE.*

## 4. An L- Homomorphic Encryption Scheme

In this section, we construct an L-homomorphic encryption scheme based on the basic encryption scheme with the help of the technology of [41,47].

### *4.1. Construction*

An L- homomorphic encryption scheme consists of the following five algorithms.

- $HE.Setup(1^L, 1^k)$: Input the security parameter $k$, sample $\vec{u} \leftarrow \mathbb{Z}_q^n$, and let $L$ be the maximum depth of arithmetic circuit supporting homomorphic evaluation. Output the public parameters $pp = (1^k, 1^n, 1^m, q, \chi, \vec{u}, L)$.
- $HE.KeyGen(pp)$: Input the public parameters $pp$, use algorithm TrapGen$(q, n, m)$ to generate matrices $A \in \mathbb{Z}_q^{n\times m}$ with trapdoor basis $T$, where $m \geq 6n\log q$, use algorithm SamplePre$(A, T, \vec{u})$ to sample a vector $\vec{s}_0 \in \mathbb{Z}_q^m$, where $A\vec{s}_0 = \vec{u}$, sample $\vec{s}_l \leftarrow \chi^m$ and compute

$$\vec{s}_l^* = BD(1;\vec{s}_l) \otimes BD(1;\vec{s}_l) \in \{0,1\}^{((m+1)\lceil logq \rceil)^2}, \quad (1)$$

$$P_{(l-1):l} \leftarrow SwitchKeyGen(\vec{s}^*_{l-1}, \vec{s}_l),$$

where $l = 1, 2, \cdots, L$. Output the encryption key $pk = (\vec{u}| - A)$, the decryption key $sk = (\vec{s}_L, T)$, $evk = \{P_{(l-1):l}\}_{l=1,2,\cdots,L}$. (Note that the decryption key $T$ is redundant here, we can instead just let $sk = \vec{s}_L$. The decryption key $T$ is needed to construct the PRE scheme, as described below.)
- $HE.Enc(pp, pk, \mu)$: Identical to the basic encryption scheme, output $ct \leftarrow E.Enc(pp, pk, \mu)$.
- $HE.Eval(.)$: As [41] and [47], We consider homomorphic addition and multiplication of depth $L$ arithmetic circuits over $GF(2)$ in a gate-to-gate manner. That is, the decryption key of the ciphertexts operated by the gate at level i of the circuit is $\vec{s}_{i-1}$, and the decryption key of the ciphertexts output by the homomorphic operation is $\vec{s}_i$.

$-Add(ct_1, ct_2)$: Input ciphertexts $ct_1, ct_2$ under secret key $S_{i-1}$, compute

$$\widetilde{ct}_{add} = P2(ct_1 + ct_2) \otimes P2(1, , 0, \cdots, 0), \quad (2)$$

and output

$$ct_{add} \leftarrow SwitchKey(P_{(l-1):l}, \widetilde{ct}_{add}).$$

$-Mult(ct_1, ct_2)$: Input ciphertexts $ct_1, ct_2$ under secret key $S_{i-1}$, compute

$$\widetilde{ct}_{mult} = \left\lfloor \frac{2}{q}(P2(ct_1) \otimes P2(ct_2)) \right\rceil, \quad (3)$$

and output

$$ct_{mult} \leftarrow SwitchKey(P_{(l-1):l}, \widetilde{ct}_{mult}).$$

- $HE.Dec(pp, sk, ct)$: Input ciphertexts $ct$ under secret key $\vec{s}_L$. Output $\mu \leftarrow E.Dec(pp, sk, ct)$.

### *4.2. Analysis for Homomorphism*

We next show the homomorphism of the above L- Homomorphic Encryption scheme.

**Lemma 8.** *Let $q,k,m,n,s,L,\chi$ be parameters for the above homomorphic encryption scheme, $\chi$ be B-bounded, and $(pk, sk, evk) \leftarrow HE.KeyGen(pp)$. Let $ct_1, ct_2$ be such that*

$$\begin{aligned} ct_1(1;\vec{s}_l) &= x_1 + \lfloor \frac{q}{2} \rfloor \mu_1, \\ ct_2(1;\vec{s}_l) &= x_2 + \lfloor \frac{q}{2} \rfloor \mu_2, \end{aligned} \tag{4}$$

*$\|x_1\|_\infty, \|x_2\|_\infty \le E < \lfloor \frac{q}{2} \rfloor /2$. Set $ct_{add} \leftarrow Add(ct_1, ct_2)$, $ct_{mult} \leftarrow Mult(ct_1, ct_2)$, then*

$$ct_{add}(1;\vec{s}_{l+1}) = x_{add} + \lfloor \frac{q}{2} \rfloor [\mu_1 + \mu_2]_2,$$

$$ct_{mult}(1;\vec{s}_{l+1}) = x_{mult} + \lfloor \frac{q}{2} \rfloor \mu_1 \mu_2,$$

*where $\|x_{add}\|_\infty, \|x_{mult}\|_\infty \le O(mlogq) \cdot max\{(mlog^2q)B, E\}$.*

**Theorem 1.** *Let $q,k,m,n,L$ be parameters for the above HE scheme, $\chi$ be B-bounded. If$\big(O(mlogq)\big)^{L+O(1)} \le q/B^2$, then the HE scheme is L homomorphic.*

**Proof.** Let $E_i$ be the bound of noise after evaluation on the $i-th$ level of gates in ciphertext. By Lemma 5, we have $E_0 \le (m+1)B^2 = O(m)B^2$. According to Lemma 8, when $mlog^2qB \le E$ holds at a certain point, then $E_{i+1} = O(mlogq) \cdot E_i$ and $E_L = \big(O(mlogq)\big)^{L+O(1)} \cdot B^2$. Therefore, the decryption is correct if $E_L < \lfloor \frac{q}{2} \rfloor /2$, that is $\big(O(mlogq)\big)^{L+O(1)} < q/B^2$. □

### 4.3. Security Analysis

We now outline the proof of security to show that the HE scheme is CPA secure based on LWE assumption. We show $(pk, evk, ct) = ((\vec{u}|A), \{P_{(l-1):l}\}_{l=1,2,\cdots,L}, ct)$ is indistinguishable from uniform by applying a hybrid argument. Since $\vec{s}_L$ is only used to generate $P_{(L-1):L}$, we can get $P_{(L-1):L}$ is indistinguishable from uniform by Lemma 7. Then we can proceed to replace all $P_{(l-1):l}$ with uniform in descending order. Finally, there is only $((\vec{u}|A), ct)$ left, which is indistinguishable from uniform by the security analysis of the basic encryption scheme.

## 5. The HPRE-SAC Scheme

In this section, we will use the above homomorphic encryption (HE) scheme to construct the HPRE-SAC scheme by using Trapdoor Sampling [27,48].

### 5.1. Construction

The HPRE-SAC scheme consists of the following seven algorithms.

- $HPRE.Setup(1^k, 1^L)$ : Identical to the HE scheme, output $pp \leftarrow HE.Setup(1^k, 1^L)$.
- $HPRE.KeyGen(pp)$ : Identical to the HE scheme, output $(sk, pk, evk) \leftarrow HE.KeyGen(pp)$.
- $HPRE.Enc(pp, pk, \mu)$: Identical to the HE scheme, output $ct \leftarrow HE.Enc(pp, pk, \mu)$
- $HPRE.ReKey(pp, sk^i, pk^i, pk^j)$: Input $pp$, the encryption key $pk^i = (\vec{u}| - A^i)$ and the decryption key $sk^i = (\vec{s}^i_L, T^i)$ of user $i$, the encryption key $pk^j = (\vec{u}| - A^j)$ of user $j$, sample $X^{i\rightarrow j} \leftarrow \chi^{n\times m}$, use algorithm SamplePre($A^i, T^i, A^j + X^{i\rightarrow j}$) to sample a matrix $R^{i\rightarrow j}$, where
$$A^i R^{i\rightarrow j} = A^j + X^{i\rightarrow j}, \tag{5}$$
output the re-encryption key $rk^{i\rightarrow j} = R^{i\rightarrow j}$.

- $HPRE.ReEnc(pp, rk^{i\to j}, ct^i)$ : Input $pp$, a original ciphertext $ct^i$ of user $i$, and a re-encryption key $rk^{i\to j} = R^{i\to j}$. Output a re-encryption ciphertext
$$ct^j = ct^i \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{i\to j} \end{bmatrix} + (\vec{z}^{i\to j})^t \tag{6}$$
for user $j$, where $\vec{z}^{i\to j} \leftarrow \chi^{1\times(m+1)}$.
- HPRE.Eval$(pp, f, ct_1, \cdots, ct_l, evk) \to ct_f$: Except for the ciphertexts $ct_1, \cdots, ct_l$ that belongs to a user can be the original ciphertext or re-encryption ciphertext, the rest are the same as HE scheme, $ct_f \leftarrow HE.Eval(pp, f, ct_1, \cdots, ct_l, evk)$.
- HPRE.Dec$(pp, sk, ct) \to \mu$: Identical to the HE scheme, output $\mu \leftarrow HE.Dec(pp, sk, ct)$.

### 5.2. Correctness Analysis

We show the correctness in this subsection.

For a original ciphertext, we know the decryption is correct by Lemma 5. For a re-encryption ciphertext $ct^j = ct^i \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{i\to j} \end{bmatrix} + (\vec{z}^{i\to j})^t$, where $ct^i = \vec{e}^{it}(\vec{u}| - A^i) + \vec{y}^{it} + \lfloor \frac{q}{2} \rfloor \vec{\mu}^{it}$, $\vec{\mu}^{it} = (\mu, 0, \cdots, 0)$, $\vec{e}^i \leftarrow \chi^{1\times n}$, $\vec{y}^i \leftarrow \chi^{1\times m+1}$, $\vec{z}^{i\to j} \leftarrow \chi^{1\times(m+1)}$, we have

$$\begin{aligned} ct^j &= ct^i \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{i\to j} \end{bmatrix} + (\vec{z}^{i\to j})^t \\ &= \left(\vec{e}^{it}(\vec{u}| - A^i) + \vec{y}^{it} + \lfloor \frac{q}{2} \rfloor \vec{\mu}^{it}\right) \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{i\to j} \end{bmatrix} + (\vec{z}^{i\to j})^t \\ &= \lfloor \frac{q}{2} \rfloor \vec{\mu}^{it} + \vec{e}^{it}(\vec{u}| - A^j - X^{i\to j}) + \vec{\sigma}^t \end{aligned}$$

where $\vec{\sigma}^t = \vec{y}^{it} \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{i\to j} \end{bmatrix} + (\vec{z}^{i\to j})^t$ by (5), (6). Thus,

$$\begin{aligned} ct^j(1; \vec{s}^j) &= \left(\lfloor \frac{q}{2} \rfloor \vec{\mu}^{it} + \vec{e}^{it}(\vec{u}| - A^j - X^{i\to j}) + \vec{\sigma}^t\right)(1; \vec{s}^j) \\ &= \lfloor \frac{q}{2} \rfloor \mu^i + \vec{e}^{it}\left(\vec{u} - A^j\vec{s}^j - X^{i\to j}\vec{s}^j\right) + \vec{\sigma}^t(1; \vec{s}^j) \\ &= \lfloor \frac{q}{2} \rfloor \mu^i + \underbrace{\vec{e}^{it}\left(-X^{i\to j}\vec{s}^j\right) + \vec{\sigma}^t(1; \vec{s}^j)}_{y} \end{aligned} \tag{7}$$

So we have the following Lemma 9.

**Lemma 9.** *Let $q, k, m, n$ be parameters for the above basic encryption scheme, $\chi$ be B-bounded. Set $(\vec{s}^j, A^j) \leftarrow HPRE.KeyGen(pp)$, $ct^i \leftarrow HPRE.Enc(pp, pk^i, \mu^i)$, $ct^j \leftarrow HPRE.ReEnc (pp, rk^{i\to j}, ct^i)$. Then*
$$ct^j(1; \vec{s}^j_0) = y + \lfloor \frac{q}{2} \rfloor \mu^i,$$
*where $y = \left(\vec{y}^{it} \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{i\to j} \end{bmatrix} + (\vec{z}^{i\to j})^t\right)(1; \vec{s}^j) + \vec{e}^{it}(-X^{i\to j}\vec{s}^j)$. Since $R^{i\to j} \leftarrow$SamplePre$(A^i, T^i, A^j + X^{i\to j})$, we have $\|R^{i\to j}\|_\infty \leq B$ by Lemma 3. If $\|y\| \leq (m+1)(mB+1)B^2 + nmB^3 < \lfloor \frac{q}{2} \rfloor / 2$, then $\mu \leftarrow E.Dec(pp, sk, ct)$.*

Next, we consider the homomorphic operations of ciphertexts (including original ciphertexts and re-encryption ciphertexts). According to Lemma 9, the decryption of re-encryption ciphertext has the same form as the original ciphertext. Therefore, Lemma 8 shows that the homomorphism operation is feasible, including the homomorphic operation over the original ciphertexts, the homomorphic operation over the original ciphertexts and the re-encryption ciphertexts, and the homomorphic operation over the re-encryption ciphertexts. In addition, it is noted that the re-encryption ciphertexts has a larger decryption noise magnitude. Therefore, in order to prove that the HPRE scheme is L homomorphic, we only need to control the decryption noise magnitude of the homomorphic operations over the re-encryption ciphertexts. So similar to Theorem 1, we have Theorem 2.

**Theorem 2.** *Let $q, k, m, n, L$ be parameters for the above HPRE-SAC scheme, $\chi$ be B-bounded. If $\big(O(mlogq)\big)^{L+O(1)} < q/B^3$, then the HPRE-SAC scheme is L homomorphic.*

**Proof.** Let $E_i$ be the bound of noise after evaluation on the $i-th$ level of gates in ciphertext. By Lemma 9, we have $E_0 \leq (m+1)(mB+1)B^2 + nmB^3 = O(m^2)B^3$. According to Lemma 8, when $mlog^2qB \leq E$ holds at a certain point, then $E_{i+1} = O\big(mlogq\big) \cdot E_i$ and $E_L = \big(O(mlogq)\big)^{L+O(1)} \cdot B^3$. Therefore, the decryption is correct if $E_L < \lfloor \frac{q}{2} \rfloor /2$, that is $\big(O(mlogq)\big)^{L+O(1)} < q/B^3$. □

Finally, we show that the HPRE-SAC scheme is multi-hop.

**Theorem 3.** *Let $q, k, m, n, L$ be parameters for the above HPRE-SAC scheme, $\chi$ be B-bounded, then the HPRE-SAC scheme is multi-hop.*

**Proof.** Let the public key of user $i$ be $pk^i = (\vec{u}|A^i)$, the re-encryption key from user $i$ to user $j$ be $rk^{i\to j} = R^{i\to j}$, $i = 1, 2, \cdots, l$, the ciphertext of user 1 be $ct^1 = \vec{e}^{1t}(\vec{u}| - A^1) + \vec{y}^{1t} + \lfloor \frac{q}{2} \rfloor \vec{\mu}^{1t}$, where $\vec{\mu}^{1t} = (\mu^1, 0, \cdots, 0)$, $\vec{e}^1 \leftarrow \chi^{1\times n}$, $\vec{y}^1 \leftarrow \chi^{1\times m+1}$. If $ct^{i+1} \leftarrow HPRE.ReEnc(pp, rk^{i\to i+1}, ct^i)$, then by (6), we have

$$ct^2 = ct^1 \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{1\to 2} \end{bmatrix} + (\vec{z}^{1\to 2})^t,$$

$$\begin{aligned} ct^3 &= ct^2 \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{2\to 3} \end{bmatrix} + (\vec{z}^{2\to 3})^t \\ &= ct^1 \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{1\to 2}R^{2\to 3} \end{bmatrix} + (\vec{z}^{1\to 2})^t \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & R^{2\to 3} \end{bmatrix} + \quad (\vec{z}^{2\to 3})^t \end{aligned}$$

$$\cdots$$

$$ct^l = ct^1 \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & \prod_{i=1}^{l-1} R^{i\to i+1} \end{bmatrix} + \underbrace{\sum_{j=1}^{l-2} (\vec{z}^{j\to j+1})^t \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & \prod_{i=j+1}^{l-1} R^{i\to i+1} \end{bmatrix} + (\vec{z}^{(l-1)\to l})^t}_{\vec{\rho}_1^t}$$

By (5), we get

$$\begin{aligned} ct^1 \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & \prod_{i=1}^{l-1} R^{i\to i+1} \end{bmatrix} &= \Big( \vec{e}^{1t}(\vec{u}| - A^1) + \vec{y}^{1t} + \lfloor \frac{q}{2} \rfloor \vec{\mu}^{1t} \Big) \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & \prod_{i=1}^{l-1} R^{i\to i+1} \end{bmatrix} \\ &= \vec{e}^{1t} \left( \vec{u}| - A^1 \prod_{i=1}^{l-1} R^{i\to i+1} \right) + \lfloor \frac{q}{2} \rfloor \vec{\mu}^{1t} + \underbrace{\vec{y}^{1t} \begin{bmatrix} 1 & 0_{1\times m} \\ 0 & \prod_{i=1}^{l-1} R^{i\to i+1} \end{bmatrix}}_{\vec{\rho}_2^t} \\ &= \vec{e}^{1t}(\vec{u}| - A^l - Y) + \lfloor \frac{q}{2} \rfloor \vec{\mu}^{1t} + \vec{\rho}_2^t, \end{aligned}$$

where $Y = X^{(l-1)\to l} + \sum_{j=1}^{l-1} X^{j\to(j+1)} \prod_{i=j+1}^{l-1} R^{i\to(i+1)}$. Therefore,

$$\begin{aligned} ct^l(1;\vec{s}^l) &= \left(\vec{e}^{1t}(\vec{u}| - A^l - Y) + \lfloor\frac{q}{2}\rfloor\vec{\mu}^{1t} + (\vec{\rho}_1 + \vec{\rho}_2)^t\right)(1;\vec{s}^l) \\ &= \lfloor\frac{q}{2}\rfloor\mu^1 \underbrace{-\vec{e}^{1t}Y\vec{s}^l + (\vec{\rho}_1 + \vec{\rho}_2)^t)(1;\vec{s}^l)}_{\rho} \end{aligned}$$

If $\|\rho\| = O(m^l)B^{l+1} < \lfloor\frac{q}{2}\rfloor/2$, the re-encryption ciphertext $ct^l$ can be correctly decrypted.

Similar to the proof of Lemma 9 and Theorem 2, we know that if $\big(O(mlogq)\big)^{l+L+O(1)} < q/B^{l+1}$, the HPRE-SAC scheme is multi-hop. □

### 5.3. Security Analysis

We show the security in this subsection.

**Theorem 4.** *Let $q, k, m, n, L$ be parameters for the above HPRE-SAC scheme, $\chi$ be B-bounded. If $\big(O(mlogq)\big)^{L+O(1)} < q/B^3$, then the HPRE-SAC scheme is IND-CPA secure based on LWE.*

**Proof.** We consider the following games.

Game $G_0^b$: This game is the original game $Expt_{HPRE,\mathcal{A}}^{CPA}(k)$ between challenger and adversary. Suppose that the index of target honest user is 0, the $pk^0 = (\vec{u}| - A^0)$, $sk^0 = (\vec{s}_L^0, T^0)$, $evk^0 = \{P_{(l-1):l}^0\}_{l=1,2,\cdots,L}$, where $P_{(i-1):i}^0 \leftarrow SwitchKeyGen(\vec{s}_{l-1}^{0*}, \vec{s}_l^0)$, $\vec{s}_l^{0*} = BD(1;\vec{s}_l^0) \otimes BD(1;\vec{s}_l^0)$, $\vec{s}_l^0 \leftarrow \chi^m$, $\vec{s}_0^0 \leftarrow$SamplePre$(A, T, \vec{u})$. The challenger computes the challenge ciphertext on query $\mu$ as follows:

- If $b = 0$, it returns $ct = \vec{e}^{0t}(\vec{u}| - A^0) + \vec{y}^{0t} + \lfloor\frac{q}{2}\rfloor\vec{\mu}^t$, where $\vec{\mu}^t = (\mu, 0, \cdots, 0)$, $\vec{e}^0 \leftarrow \chi^{1\times n}$, $\vec{y}^0 \leftarrow \chi^{1\times m+1}$.
- If $b = 1$, it returns a random ciphertext $ct \leftarrow \mathbb{Z}_q^{1\times m+1}$

Game $G_1^b$: We modify the encryption key generation oracle $\mathcal{O}_{pk}$. This game is identical to game $G_0$, except that the challenger replaces $A^i$ of user $i$ with $A_+^i$, where $(A_+^i, T_+^i) \leftarrow$ TrapGen$(q, n, m)$.

Because of $(A_+^i, T_+^i) \leftarrow$TrapGen$(q, n, m)$, $(A^i, T^i) \leftarrow$ TrapGen$(q, n, m)$, we have $A_+^i, A^i$ are statistically close to uniform by Lemma 1. Therefore, $A_i \approx_s A_i^+$. So $G_0^b \approx_s G_1^b$

Game $G_2^b$: We modify the evaluation key generation oracle $\mathcal{O}_{evk}$. The challenger computes $P_{(l-1):l,+}^i \leftarrow SwitchKeyGen(\vec{s}_{l-1+}^{i*}, \vec{s}_{l+}^i)$, where $\vec{s}_{l+}^{i*} = BD(1;\vec{s}_{l+}^i) \otimes BD(1;\vec{s}_{l+}^i)$, $\vec{s}_{0+}^i \leftarrow$SamplePre$(A_+^i, T_+^i, \vec{u})$, $\vec{s}_{l+}^i \leftarrow \chi^m$, and replaces $P_{(l-1):l}^i$ of user $i$ with $P_{(l-1):l,+}^i$, $l = 1, 2, \cdots, L$. The rest are the same as $G_1^b$.

Since $\vec{s}_L^i \leftarrow \chi^m(\vec{s}_{L+}^i \leftarrow \chi^m)$ is only used to generate $P_{(L-1):L}^i(P_{(L-1):L,+}^i)$, we can get $P_{(L-1):L}^i(P_{(L-1):L,+}^i)$ indistinguishable from uniform by Lemma 7. Therefore, $P_{(L-1):L}^i \approx_s P_{(L-1):L,+}^i$.Then we can get $P_{(l-1):l}^i \approx_s P_{(l-1):l,+}^i$ in descending order, $l = 1, 2, \cdots, L$. So $G_1^b \approx_s G_2^b$.

Game $G_3^b$: We modify the re-encryption key generation oracle $\mathcal{O}_{rk}$. the challenger samples $R_+^{i\to j} \leftarrow \chi^{m\times m}$ and replaces $R_{i\to j}$ with $R_+^{i\to j}$. The rest are the same as $G_2^b$.

Because of $A^iR^{i\to j} = A^j + X^{i\to j}$, we have $A^1R^{1\to 2}R^{2\to 3} = (A^2 + X^{1\to 2})R^{2\to 3} = A^3 + X^{2\to 3} + X^{1\to 2}R^{2\to 3}$. Therefore, the adversary cannot use $R^{1\to 2}, R^{2\to 3}$ to verify the relationship between $A^1$ $A^2$and $A^3$. So $R^{i\to j}$ is independent of each other. Since $R^{i\to j} \leftarrow$SamplePre $(A^i, T^i, A^j + X^{i\to j})$, we know $R^{i\to j}$ statistically close to $\chi^{m\times m}$ by Lemma 3.That is $R_{i\to j} \approx_s R_+^{i\to j}$. So $G_2^b \approx_s G_3^b$.

Game $G_4^b$: We modify re-encryption ciphertext generation oracle $\mathcal{O}_{re}$. The challenger replaces the re-encrypted ciphertext $ct^j$ with $ct_+^j \leftarrow$ HPRE.ReEnc$\big(pp, r^{i\to j}, ct^i\big)$. The rest are the same as $G_3^b$.

According to Lemma 3, we have the $R_{i\to j} \approx_s R^{+}_{i\to j}$. It follows that $G_3^b \approx_s G_4^b$, for efficient adversary.

Finally, we have that $G_4^1 \approx_c G_4^0$ from LWE. Combining the above indistinguishability, we have shown that $G_0^1 \approx_c G_0^0$. This completes the proof. □

It should be noted that our HPRE-SAC scheme uses trapdoor to generate re-encryption key and decryption key respectively, which not only ensures the homomorphism, but also ensures the resistance to strong collusion attack. By Lemma 4, we know that the trapdoor sampling algorithm is one-way and collision-resistant, so the delegatee and the proxy can not attack the decryption key of the delegator. In addition, the decryption key does not participate in the re-encryption key generation, and is only used for ciphertext decryption. Therefore, the adversary can not get any information of the decryption key, so the approximate value of the decryption key can not be obtained.

If the adversary obtains the approximate value $P2(\vec{s}) + \vec{x}$ of the decryption key $\vec{s}$, where $\vec{x}$ is an error distribution, then the adversary can decrypt the delegator's ciphertext. Let $ct = (ct_1, ct_2) = \left(\vec{e}^t\vec{u} + y + \lfloor\frac{q}{2}\rfloor\mu, \vec{e}^t(-A) + \vec{y}^t\right)$, then we have

$$\begin{aligned}(ct_1, BD(ct_2))(1; P2(\vec{s}) + \vec{x}) &= ct_1 + ct_2\vec{s} + BD(ct_2)\vec{x}\\ &= (ct_1, ct_2)(1;\vec{s}) + BD(ct_2)\vec{x}\\ &= \left(\vec{e}^t(\vec{u}| - A) + \vec{y}^t + \lfloor\frac{q}{2}\rfloor\vec{\mu}^t\right)(1;\vec{s}) + BD(ct_2)\vec{x}\\ &= \vec{e}^t(\vec{u}| - A)(1;\vec{s}) + \vec{y}^t(1;\vec{s}) + \lfloor\frac{q}{2}\rfloor\vec{\mu}^t(1;\vec{s}) + BD(ct_2)\vec{x}\\ &= \underbrace{\vec{y}^t(1;\vec{s}) + BD(ct_2)\vec{x}}_{x} + \lfloor\frac{q}{2}\rfloor\mu.\end{aligned}$$

If $\|x\|_\infty < \lfloor\frac{q}{2}\rfloor/2$, then the decryption is correct. Thus, the IND-CPA security of the HPRE-SAC scheme does not hold, which is in contradiction with Theorem 4. Therefore, the adversary can not obtain the approximate value $P2(\vec{s}) + \vec{x}$ of the decryption key $\vec{s}$.

In addition, although our HPRE-SAC scheme is single bit encryption, we can use homomorphic ciphertext packing technology [51] and trapdoor based multi bit proxy re-encryption scheme [27] to construct a multi bit homomorphic proxy re-encryption scheme against strong collusion attack.

### *5.4. Comparisons*

We compare the related works in this subsection.

At present, there are many PRE schemes. We only select some related works from the lattice based PRE and compare them with our schemes. It can be seen from Table 1 that Ma et al. [28], Li et al. [44,45], Li et al. [46] and our scheme are homomorphic proxy re-encryption schemes. The following comparison is made from the length of the encryption key, decryption key, re-encryption key and ciphertext (including original ciphertexts and re-encryption ciphertexts). The comparison results are shown in Table 3.

It can be seen from Table 3 that the public key length of Ma et al. [28] is $nlogq$, that of Li et al. [44] is $m(n+1)logq$, that of Li et al. [45] is the same as that of Li et al. [44], and that of Li et al. [46] is the longest, which is $(nlogn+2)logq$. The length of the public key of our HPRE-SAC scheme is $nm$, which is smaller than that of Li et al. [44] and only one constant times different from that of Ma et al. [28]. From the length of re-encryption key, we can find that the complexity of Ma et al. [28] is $O(n^3logq)$, that of Li et al. [46] is only $O(nlogq)$, and the rest is $O(n^2logq)$. However, by observing the length of the ciphertext (including original ciphertexts and re-encryption ciphertexts), we can find that the length of the ciphertext of Li et al. [46] is the largest, that is $O((nlogq)^2logq)$, while that of our scheme HPRE-SAC and [44,45] are the smallest, the complexity is only $O(nlogq)$. In conclusion, the comparison shows that our scheme HPRE-SAC has better parameters.

In addition, it should be noted from Table 1 that only our HPRE-SAC scheme can resist strong collusion attack.

**Table 3.** The parameters comparison of HPRE schemes.

| **Scheme** | $\|pk\|$ | $\|sk\|$ | $\|rk\|$ | $\|ct\|$ |
|---|---|---|---|---|
| Ma et al. [28] | $nlogq$ | $n$ | $n^3logq + n^2logq$ | $n^2logq$ |
| Li et al. [44] | $m(n+1)logq$ | $(2m+1)logq$ | $(2n+1)((2n+1)logq + m)$ | $(2n+1)logq$ |
| Li et al. [45] | $m(n+1)logq$ | $(n+1)logq$ | $(n+1)((n+1)logq + m)logq$ | $(n+1)logq$ |
| Li et al. [46] | $(nlogn+2)logq$ | $nlogn$ | $nlogq + 2$ | $(nlogq+2)^2log + nlog(n+2)$ |
| HPRE-SAC | $nm$ | $m$ | $(m+1)^2$ | $m+1$ |

$\|pk\|$, $\|sk\|$, $\|rk\|$, $\|ct\|$ represent the length of encryption key, decryption key re-encryption key and ciphertext (including original ciphertexts and re-encryption ciphertexts) respectively, $m = 6n\lceil logq \rceil$.

### 5.5. An Application

In this section, we present an application of our scheme HPRE-SAC: Secure computing of personal healthcare records (PHRs) in the cloud.

At present, there are many applications of PRE in the cloud [52–56], especially in cloud based PHRs [57,58]. The overall system architecture of cloud based PHRs computing using the proposed HPRE-SAC scheme is shown in Figure 3. It includes four entities: patient (data owner), E-Healthcare cloud service provider (CSP), trusted authority (TA) and doctor (data receiver). The following steps are required.

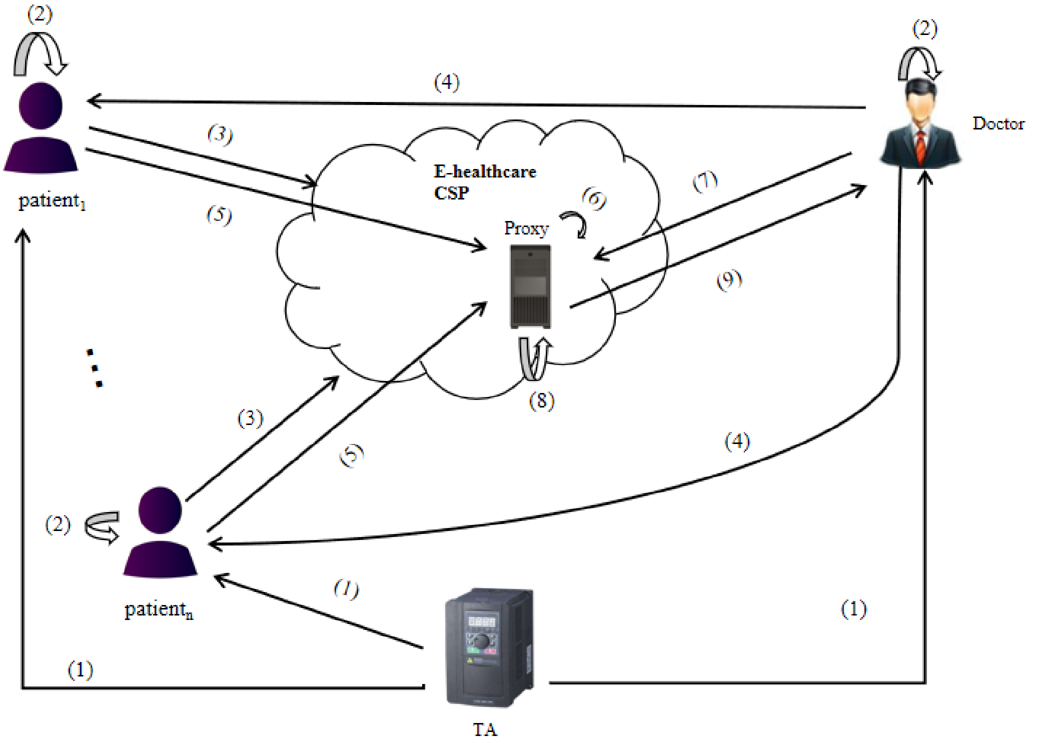

**Figure 3.** Secure computing of personal healthcare records (PHRs) using HPRE with strong anti-collusion (SAC) in the cloud.

(1) Patients and the doctor use the algorithm *HPRE.Setup* to register in TA to obtain the public parameters of the system.

(2) Patients and the doctor use the algorithm *HPRE.KeyGen* to generate their own encryption key, public evaluation key and decryption key.

(3) Patients use the algorithm $HPRE.Enc$ to encrypt their PHRs and upload them to the E-healthcare cloud service provider for storage. The PHRs here includes not only diagnostic information from doctors, but also personal health information collected by smart wearable devices. We assume that the E-healthcare cloud service provider is not trusted, so the patients need to encrypt the data.
(4) For a certain purpose (in addition to clinical purposes, it can also be for research purposes), the doctor asks patients for the right to decrypt their encrypted data.
(5) After the patient agrees with the doctor's request, the algorithm $HPRE.ReKey$ is used to generate the re-encryption key and send it to the proxy.
(6) Suppose that the proxy residing in the cloud is semi-trusted, that is to say, it follows the protocol, but can collect information to infer private information, or collude with the data user to attack the data owner. The proxy re-encrypts the patient's ciphertext to generate the doctor's ciphertext by using the algorithm $HPRE.ReEnc$.
(7) The doctor needs to analyze and calculate the PHRs of multiple patients for a certain purpose (in addition to clinical purpose, it can also be for research). In order to reduce the burden of local computation and communication, the doctor sends the function to the proxy.
(8) The proxy uses the algorithm $HPRE.Eval$ to perform homomorphic function operation on the re-encryption ciphertext belonging to the doctor.
(9) The doctor downloads the results of homomorphic operation and decrypts them locally by using the algorithm $HPRE.Dec$ to obtain the required data.

In this system architecture, it not only ensures the safety of the patient's data, but also meets the efficient needs of doctors for the statistical analysis of PHRs of multiple patients.

## 6. Conclusions

In order to adapt to efficient and secure cloud computing, this paper proposes a lattice based homomorphic proxy re-encryption scheme, namely HPRE-SAC, which can resist strong collusion attack. In particular, the HPRE-SAC scheme is unidirectional, multi-hop, and CPA secure under LWE. Compared with the existing HPRE scheme, the HPRE-SAC scheme has better parameters. However, the efficiency of the HPRE-SAC scheme is still low. The future work will be to construct a more efficient HPRE scheme based on the existing scheme, such as constructing an HPRE scheme on the ring LWE to meet the more comprehensive application requirements.

**Author Contributions:** All authors contributed to the paper. J.L. and Z.Q. wrote the manuscript with the supervision from K.Z. and C.C. is responsible for the design of the cryptosystem. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China (61702224, U1936112), the Heilongjiang Provincial Natural Science Foundation of China (LH2020F044), and the Guangxi Key Laboratory of Cryptography and Information Security (GCIS201904).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Regev, O. On lattices, learning with errors, random linear codes, and cryptography. In Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, 22–24 May 2005; pp. 84–93.
2. Regev, O. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM* **2009**, *56*, 1–40. [CrossRef]
3. Micciancio, D.; Regev, O. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.* **2007**, *37*, 267–302. [CrossRef]
4. Applebaum, B.; Cash, D.; Peikert, C.; Sahai, A. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology, Santa Barbara, CA, USA, 16–20 August 2009; pp. 595–618.
5. Lindner, R.; Peikert, C. Better key sizes (and attacks) for LWE-based encryption. In Proceedings of the 11th International Conference on Topics in Cryptology: CT-RSA 2011, San Francisco, CA, USA, 14–18 February 2011; pp. 319–339.
6. Orsini, E.; Smart, N.P. Bootstrapping BGV ciphertexts with a wider choice of $p$ and $q$. In Proceedings of the 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, 30 March–1 April 2015; pp. 673–698.

7. Gentry, C. A Fully Homomorphic Encryption Scheme. Ph.D. Thesis, Stanford University, Stanford, CA, USA, 2009.
8. Gentry, C. Fully homomorphic encryption using ideal lattices. In Proceedings of the 41st Annual ACM Symposium on Symposium on Theory of Computing, Bethesda, MD, USA, 31 May–2 June 2009; pp. 169–178.
9. Mai, V.; Khalil, I. Design and implementation of a secure cloud-based billing model for smart meters as an Internet of things using homomorphic cryptography. *Future Gener. Comput. Syst.* **2017**, *72*, 327–338. [CrossRef]
10. Ren, S.Q.; Tan, B.H.M.; Sundaram, S.; Wang, T.; Ng, Y.; Chang, V.; Aung, K.M.M. Secure searching on cloud storage enhanced by homomorphic indexing. *Future Gener. Comput. Syst.* **2016**, *65*, 102–110. [CrossRef]
11. Blaze, M.; Bleumer, G.; Strauss,M. Divertible protocols and atomic proxy cryptography. In Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, 31 May–4 June 1998; pp. 127–144.
12. Ateniese,G.; Fu, K.; Green, M.; Hohenberger, S. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur. (TISSEC)* **2006**, *9*, 1–30. [CrossRef]
13. Yang, Y.; Zhu, H.; Lu, H.; Weng, J.; Zhang, Y.; Choo, K.K.R. Cloud based data sharing with fine-grained proxy re-encryption. *Pervasive Mob. Comput.* **2016**, *28*, 122–134. [CrossRef]
14. Wang, D.; Li, W.; Wang, P. Measuring Two-Factor Authentication Schemes for Real-Time Data Access in Industrial Wireless Sensor Networks. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4081–4092. [CrossRef]
15. Jiang, L.; Guo, D. Dynamic Encrypted Data Sharing Scheme Based on Conditional Proxy Broadcast Re-Encryption for Cloud Storage. *IEEE Access* **2017**, *5*, 13336–13345. [CrossRef]
16. Zhou, Y.; Deng, H.; Wu, Q.; Qin, B.; Liu, J.; Ding, Y. Identity-based proxy re-encryption version 2: Making mobile access easy in cloud. *Future Gener. Comput. Syst.* **2016**, *62*, 128–139. [CrossRef]
17. Lu, Y.; Li, J. A pairing-free certificate-based proxy re-encryption scheme for secure data sharing in public clouds. *Future Gener. Comput. Syst.* **2016**, *62*, 140–147. [CrossRef]
18. Zeng, P.; Choo, K R. A New Kind of Conditional Proxy Re-Encryption for Secure Cloud Storage. *IEEE Access* **2018**, *6*, 70017–70024. [CrossRef]
19. Ma, C.; Li, J.; Ouyang, W. Lattice-based identity-based homomorphic conditional proxy re-encryption for secure big data computing in cloud environment. *Int. J. Found. Comput. Sci.* **2017**, *28*, 645–660. [CrossRef]
20. Li, J.; Ma, C.; Zhang, K. A Novel Lattice-Based CP-ABPRE Scheme for Cloud Sharing. *Symmetry* **2019**, *11*, 1262. [CrossRef]
21. Liang, K.; Au, M.H.J.; Liu, K.J.; Susilo, W.; Wong, D.S.; Yang, G.; Yu, Y.; Yang, A. A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. *Future Gener. Comput. Syst.* **2015**, *52*, 95–108. [CrossRef]
22. Sun, M.; Ge, C.; Fang, L.; Wang, J. A proxy broadcast re-encryption for cloud data sharing. *Multimed. Tools Appl.* **2018**, *77*, 10455–10469. [CrossRef]
23. Xu, P.; Jiao, T.; Wu, Q.; Wang, W.; Jin, H. Conditional Identity-Based Broadcast Proxy Re-Encryption and Its Application to Cloud Email. *IEEE Trans. Comput.* **2016**, *65*, 66–79. [CrossRef]
24. Ivan, A.; Dodis, Y. Proxy Cryptography Revisited. In Proceedings of the 10th Annual Network and Distributed System Security Symposium, 497 NDSS, DBLP, San Diego, CA, USA, 23–26 February 2003.
25. Singh, K.; Pandu Rangan, C.; Banerjee, A.K. Lattice based identity based proxy re-encryption scheme. *J. Internet Serv. Inf. Secur. (JISIS)* **2013**, *3*, 38–51.
26. Jiang, M.M.; Hu, Y.P.; Wang, B.C.; Wang, F.H.; Lai, Q.Q. Lattice-based multi-use unidirectional proxy re-encryption. *Secur. Commun. Netw.* **2015**, *8*, 3796–3803. [CrossRef]
27. Li, J.; Ma, C.; Gu, Z. Multi-use Deterministic Public Key Proxy Re-Encryption from Lattices in the Auxiliary-Input Setting. *Int. J. Found. Comput. Sci.* **2020**, *31*, 551–567. [CrossRef]
28. Ma, C.; Li, J.; Ouyang, W. A Homomorphic Proxy Re-encryption from Lattices. In Proceedings of the 10th International Conference, ProvSec 2016, Nanjing, China, 10–11 November 2016; pp. 353-372.
29. Xagawa, K. Cryptography with Lattices. Ph.D. Thesis, Tokyo Institute of Technology, Tokyo, Japan, 2010.
30. Aono, Y.; Boyen, X.; Wang, L. Key-private proxy re-encryption under LWE. In Proceedings of the International Conference on Cryptology in India, Mumbai, India, 7–10 December 2013; pp. 1–18.
31. Ateniese, G.; Benson, K.; Hohenberger, S. Key-private proxy re-encryption. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 24–28 February 2009; pp. 279–294.
32. Singh, K.; Rangan, C.P.; Banerjee, A.K. Cryptanalysis of unidirectional proxy re-encryption scheme. In Proceedings of the Information and Communication Technology-EurAsia Conference, Bali, Indonesia, 14–17 April 2014; pp. 564–575.
33. Kirshanova, E. Proxy re-encryption from lattices. In Proceedings of the International Workshop on Public Key Cryptography, Buenos Aires, Argentina, 26–28 March 2014; pp. 77–94.
34. Nishimaki, R.; Xagawa, K. Key-Private Proxy Re-Encryption from Lattices, Revisited, IEICE Transactions on Fundamentals of Electronics. *Commun. Comput. Sci.* **2015**, *98*, 100–116.
35. Hou, J.; Jiang, M.; Guo, Y.; Song, W. Efficient identity-based multi-bit proxy re-encryption over lattice in the standard model. *Inf. Secur. Tech. Rep.* **2019**, *47*, 329–334. [CrossRef]
36. Yin, W.; Wen, Q.; Li, W.; Zhang, H.; Jin, Z.P. Identity Based Proxy Re-encryption Scheme under LWE. *KSII Trans. Internet Inf. Syst.* **2017**, *11*, 6116–6132.

37. Yin, W.; Wen, Q.; Li, W.; Zhang, H.; Jin, Z. A New Insigh-Proxy Re-encryption Under LWE with Strong Anti-collusion. In Proceedings of the International Conference on Information Security Practice and Experience, Kuala Lumpur, Malaysia, 26–28 November 2019; Springer: Cham, Switzerland, 2018; pp. 559-577.
38. Zhong, H.; Cui, J.; Shi, R.; Xia, C. Many-to-one homomorphic encryption scheme. *Secur. Commun. Netw.* **2015**, *9*, 1007–1015. [CrossRef]
39. Brakerski, Z.; Vaikuntanathan, V. Efficient fully homomorphic encryption from (Standard) LWE. In Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, Palm Springs, CA, USA, 22–25 October 2011; pp. 97–106.
40. Brakerski, Z.; Gentry, C.; Vaikuntanathan, V. (leveled) Fully Homomorphic Encryption without Bootstrapping. In Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, Cambridge, MA, USA, 8–10 January 2012; pp. 309–325.
41. Brakerski, Z. Fully Homomorphic Encryption without Modulus Switching from Classical Gapsvp. In Proceedings of the 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2012; pp. 868–886.
42. Gentry, C.; Sahaiy, A.; Waters, B. Homomorphic Encryption From Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In Proceedings of the 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2013; pp. 75–92.
43. Gentry, C.; Halevi, S.; Vaikuntanathan, V. A simple BGN-type cryptosystem from LWE. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, French, 30 May–3 June 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 506–522.
44. Li, Z.; Ma, C.; Wang, D. Towards Multi-Hop Homomorphic Identity-Based Proxy Re-Encryption via Branching Program. *IEEE Access* **2017**, *5*, 16214–16228. [CrossRef]
45. Li, Z.; Ma, C.; Wang, D. Achieving Multi-Hop PRE via Branching Program. *IEEE Trans. Cloud Comput.* **2020**, *8*, 45–58. [CrossRef]
46. Li, J.; Ma, C.; Zhang, L.; Yuan, Q. Unidirectional FHPRE Scheme from Lattice for Cloud Computing. *Int. J. Netw. Secur.* **2019**, *21*, 592–600.
47. Ma, C.; Li, J.; Du, G. A Flexible Fully Homomorphic Encryption. *Wirel. Pers. Commun.* **2017**, *95*, 761–772. [CrossRef]
48. Gentry, C.; Peikert, C.; Vaikuntanathan, V. Trapdoors for hard lattices and new cryptographic constructions. In Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, BC, Canada, 17–20 May 2008; pp. 197–206.
49. Alwen, J.; Peikert, C. Generating shorter bases for hard random lattices. *Theory Comput. Syst.* **2011**, *48*, 535–553. [CrossRef]
50. Micciancio, D.; Peikert, C. Trapdoors for lattices: Simpler, tighter, faster, smaller. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, 15–19 April 2012; pp. 700–718.
51. Brakerski, Z.; Gentry, C.; Halevi, S. Packed Ciphertexts in LWE-Based Homomorphic Encryption. In Proceedings of the Public-Key Cryptography-PKC 2013, Lecture Notes in Computer Science, Nara, Japan, 26 February–1 March 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 1–13.
52. Lin, H.Y.; Hung, Y.M. An Improved Proxy Re-Encryption Scheme for IoT-Based Data Outsourcing Services in Clouds. *Sensors* **2021**, *21*, 67. [CrossRef]
53. Qin, Z.; Xiong, H.; Wu, S.; Batamuliza, J. A survey of proxy re-encryption for secure data sharing in cloud computing. *IEEE Trans. Serv. Comput.* **2016**. [CrossRef]
54. Gai, K.; Qiu, M.; Zhao, H. Security-Aware Efficient Mass Distributed Storage Approach for Cloud Systems in Big Data. In Proceedings of the 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), New York, NY, USA, 9–10 April 2016; pp. 140–145.
55. Qiu, M.; Xue, C.; Sha, H.M.; Jia, Z.; Shao, Z.; Sha, E.H.M. Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP. *J. VLSI Signal Process. Syst. Signal Image Video Technol.* **2017**, *46*, 55–73. [CrossRef]
56. Dai, W.; Qiu, L.; Wu, A.; Qiu, M. Cloud infrastructure resource allocation for big data applications. *IEEE Trans. Big Data* **2016**, *4*, 313–324. [CrossRef]
57. Bhatia, T.; Verma, A.K.; Sharma, G. Secure sharing of mobile personal healthcare records using certificateless proxy re-encryption in cloud. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3309. [CrossRef]
58. Bhatia, T.; Verma, A.K.; Sharma, G. Towards a secure incremental proxy re-encryption for e-healthcare data sharing in mobile cloud computing. *Concurr. Comput. Pract. Exp.* **2020**, *32*, e5520. [CrossRef]