Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 154 (2019) 73–83

Procedia Computer Science

www.elsevier.com/locate/procedia

8th International Congress of Information and Communication Technology, ICICT 2019

Homomorphic Encryption Technology for Cloud Computing

Min Zhao E*[1, 2], Yang Geng[2]

School of Computer, Nanjing University of Post and Telecommunications, Nanjing, 210003, China
Key Laboratory of Broadband Wireless Communication and Sensor Network Technology of Ministry of Education, Nanjing University of Post and Telecommunications, Nanjing, 210003, China

Abstract

Four kinds of single homomorphic encryption algorithm were summarized for the advantages of homomorphic encryption technology in the cloud environment. It analyzed the security characteristics of four kinds of encryption algorithms. On the basis of experiments, the paper compared the efficiency of four single homomorphic encryption algorithms, gave the application scenarios of various algorithms, and introduced the research and application of single homomorphic encryption algorithm in the cloud environment. We described five kinds of fully homomorphic encryption algorithms, summarized the research situation and its application in the cloud environment and the performance of some fully homomorphic encryption algorithms were analyzed. Finally, we proposed the research direction for the subsequent studies.

Keyword: Single homomorphic encryption; Fully homomorphic encryption (FHE); Cloud computing; Ciphertext computing

1. Introduction

With the widespread application of cloud computing, more and more sensitive information and private data are stored in the cloud by users. Cloud storage security is one of the important security issues in cloud computing. In order to protect the privacy of user data, cloud data should be stored in the form of ciphertext. But encryption adds computational cost. It is hoped that the confidentiality of the data will be guaranteed at the lowest possible cost. Since cloud service providers are unreliable third parties, how to keep data confidential to cloud service providers and allow cloud service providers to complete various operations on data, it is difficult for traditional encryption methods to solve the above problems. Homomorphic encryption technology supports the management of ciphertext

[1] * Corresponding author. Email: minzhaoe@njupt.edu.cn

1877-0509
10.1016/j.procs.2019.06.012

data under privacy protection. It can directly retrieve, calculate and count ciphertext in the cloud and return the results to users in the form of ciphertext. Compared with traditional encryption algorithms, homomorphic encryption technology does not require frequent encryption and decryption between the cloud and users, thus reducing the cost of communication and computing. Homomorphic encryption technology is the key technology to ensure the confidentiality of data in cloud environments. With the homomorphic features of homomorphic encryption technology, key security issues in cloud services can be solved, and the application of cloud computing can further promote the development of homomorphic encryption technology.

In this paper, single homomorphic encryption algorithms and fully homomorphic encryption algorithms were described and analyzed respectively. For the homomorphic algorithm, the common widely used Hill, RSA, Paillier, and ElGamal encryption algorithms were listed, which were analyzed from the aspects of algorithm description, homomorphism analysis, and performance. And the efficiency of encryption and decryption of these algorithms was compared. The research and application of these algorithms in the cloud environment were introduced. For the homomorphic encryption algorithm, several representative homomorphic algorithms and their specific application scenarios in the cloud environment were reviewed. The performance indicators were compared and analyzed, and the problems to be solved in the application of homomorphic encryption algorithms were pointed out.

2 Development of Homomorphic Encryption Technology

Homomorphic Encryption (HE) was first proposed by Rivest in 1978 with the concept of "privacy homomorphism" [1]. It is a kind of encryption scheme that can directly manipulate ciphertext. The basic idea is that the encryption of plaintext after addition or multiplication is equivalent to that of ciphertext after encryption.

Before the concept of homomorphic encryption was proposed, some encryption algorithms met homomorphism, for example, Hill encryption algorithm met the addition homomorphism [2]. With the introduction of the concept of privacy homomorphism, more homomorphic encryption schemes have been proposed. Rivest, Shamir, and Adleman proposed RSA algorithm based on number theory when the concept of homomorphic encryption was born[3], which satisfies any multiplication homomorphic operation. In 1984, Goldwasser and Micali used probability encryption to propose a GM algorithm based on trapdoor function and quadratic residue [4], which was the first homomorphic public-key encryption algorithm with semantic security. However, this scheme only satisfies the addition homomorphism of any submodule 2 and has low efficiency. In 1985, ElGamal, an asymmetric encryption algorithm based on public key cryptosystem and elliptic curve cryptosystem, was proposed[5]. The algorithm satisfies any multiplicative homomorphism and can be used for encryption and signature. In 1994, Benaloh proposed an improved probabilistic encryption algorithm [6], which can encrypt r bits at a time, but this algorithm can only perform a finite number of addition homomorphic operations.

The famous Paillier encryption scheme based on the quadratic residue was proposed in 1999 [7]. This scheme is a random encryption scheme, which can perform any addition homomorphic operation. In 2005, Boneh, Goh and Nissim proposed the BGN cryptosystem based on bilinear pairings [8]. This algorithm satisfies any addition omomorphism and one multiplication homomorphism. It is the nearest scheme to the idea of homomorphism. These algorithms either satisfy the homomorphism of addition, such as GM and Paillier, or the homomorphism of multiplication, such as RSA and ElGamal, and BGN, which satisfies the homomorphism of multiple addition and single multiplication. They all basically have a single homomorphism, so they are all called single homomorphic encryption algorithm.

Gentry proposed the first homomorphic encryption scheme based on ideal lattices in 2009[9], which can add and multiply ciphertext any number of times. Then homomorphic encryption technology entered a period of rapid development. Homomorphic encryption technology can be divided into three categories: the first is an ideal lattice-based fully homomorphic encryption scheme proposed by Gentry, which is to construct a SomeWhat Homomorphic Encryption (SWHE) on the ideal of various rings, then compress the decryption circuit to reduce polynomials, and finally complete the fully homomorphic encryption under the assumption of cyclic security through bootstrapping technology. The second is an integer-based homomorphic encryption scheme [10], which is based on Gentry's idea but does not require operations based on ideal lattices of the polynomial ring. All operations are based on integers. The third is a fully homomorphic encryption scheme based on LWE (Learning With Errors) or R-LWE (Learning With Errors over Ring). This scheme is based on fault-tolerant learning and constructs a fully homomorphic encryption scheme using non-linearization, such as BGV encryption scheme [11].

3 Single Homomorphic Encryption Algorithm

Before the concept of "privacy homomorphism" was put forward, some algorithms met some homomorphic requirements, such as Hill Cipher. After the concept of homomorphic encryption was put forward, more homomorphic encryption algorithms were proposed, such as RSA, Paillier, ElGamal and other encryption algorithms. With the development of cloud technology, many homomorphic encryption algorithms make full use of the parallel performance of the cloud environment to improve the efficiency of encryption and decryption. More research and application of homomorphic encryption technology also improve the security of data in the cloud environment.

3.1 The basic concepts

3.1.1 Hill cipher

The Hill Cipher is a polygraphic substitution cipher based on linear algebra, invented by Hill in 1929. The encryption is:

$$C = E_K(M) = KM \bmod 26 \qquad (1)$$

where K is a key matrix and M is an n-component vector, each letter is represented by a number modulo 26: A = 0, B = 1, C=2..., Z = 25. M (consisting of a string of letters) is multiplied by K (an n× n matrix), against modulus 26. The key matrix used for encryption must be reversible, otherwise, it is impossible to decrypt. The decryption is:

$$M = D_K(C) = K^{-1}C \bmod 26 \qquad (2)$$

Where K-1 is the inverse matrix of K and C is the ciphertext.

3.1.2 RSA

RAS is a block cipher algorithm [3], whose plaintext and ciphertext are integers between 0~n-1.

Generally, the size of n is 512 bits or 1024 bits of the binary number. Key generation: p, q are two large prime numbers p≠q and n=p • q. According to Euler's theorem Φ(n)=(p-1)(q-1), the integer e is randomly selected to make gcd(Φ(n),e)=1,1<e<Φ(n) and d≡e-1modΦ(n), where the public key is KP={n,e} and SK={d}. The encryption is:

$$C=E_{kp}(m)=m^e \bmod n \qquad (3)$$

where m is the plaintext, and e is the public key. The decryption is:

$$M = D_{sk}(C) = C^d \bmod n \qquad (4)$$

where C is the ciphertext and d is the sk. It can be seen from the encryption that the RSA algorithm satisfies the homomorphism of multiplication.

3.1.3 ElGamal

In 1985, ElGamal, an asymmetric encryption algorithm based on public key cryptosystem and elliptic curve cryptosystem, was proposed [5]. Key generation: select large prime numbers p, g (g<p) as the generator of the cyclic group Z_p, randomly select $x \in [0,1,2,\cdots,p-1]$ and calculate $y \equiv g^x \bmod p$, where $KP = \{y\}$ and SK={x}. The encryption process is to randomly select number k to be coprime to p-1. The encryption is:

$$E_y(m)=(a, b) \qquad (5)$$

where $a=g^k \bmod p$, $b=my^k \bmod p$.

The decryption is:

$$m=D_x(c)=b(a^x)^{-1}\bmod p \qquad (6)$$

where a, b are the ciphertexts, and x is the SK. ElGamal encryption algorithm the homomorphism of multiplication under certain conditions.

3.1.4 Paillier cryptosystem

Paillier proposed a probabilistic public-key cryptosystem based on higher-order residual classes, namely Paillier cryptosystem in 1999 [7]. Key generation: set $n=p \cdot q$, where p, q is 2 large prime numbers. $g \in \mathbf{Z}_{n^2}^*$ is randomly selected to make g meet $\gcd(L(g^\lambda \bmod n^2),n)=1$, where function L is defined as:

$$L(u) = (u-1)/n \qquad (7)$$

$$\lambda(n) = lcm(p-1)(q-1) \qquad (8)$$

where the public key is $KP=\{n,g\}$ and $SK=\lambda(n)$. The encryption process is to set an arbitrary plaintext $m \in \mathbf{Z}_N$ and randomly select number $r \in \mathbf{Z}_N^*$, then the ciphertext is:

$$c = E(m) = g^m r^n \bmod n^2 \qquad (9)$$

The decryption is:

$$m=D(c)=L(c^\lambda \bmod n^2)/L(g^\lambda \bmod n^2)\bmod n \qquad (10)$$

The analysis shows that Paillier cryptosystem satisfies homomorphism of addition and homomorphism of mixed multiplication.

3.2 Security analysis of single homomorphism algorithm

Because the Hill cipher algorithm has linear characteristics and directly matrices plaintext information, it is easier to be cracked by known plaintext. Subsequently, an improved key matrix based on chaotic theory is proposed. The matrix mainly uses chaotic dynamic equations with complete randomness to generate self-inverse matrix elements, and each element has good randomness. Usually, the inherent laws between them cannot be simply obtained, so that the key matrix generated based on the block matrix is more robust.

The RSA algorithm is a widely used public key encryption algorithm, which can be used for both data encryption and digital signature. The security of the RSA algorithm is based on the integer factorization. In order to enhance the security of RSA, (p, q) are not only large prime numbers but also strong prime numbers. And the difference between them should be very large, so that n is not easy to be factorized. The parameter e cannot be too small, and generally d is larger than $n^{1/4}$, to prevent attacks by known plaintext.

ElGamal algorithm is a public key encryption algorithm which can be well applied to encryption and digital signature. The security of the algorithm mainly depends on the difficulty of computing discrete logarithms in finite fields. Improving the security of the algorithm requires an increase in the choice of finite fields. Its ciphertext depends not only on the plaintext, but also on the choice of random numbers. Random encryption increases the security of the algorithm. Because of the high efficiency of building public key cryptosystems on elliptic curves, more digital signature schemes based on elliptic curves have been proposed.

Paillier's cryptosystem is a public key cryptosystem with semantic security, which has been widely applied in triangulation, dot product protocol, and secret comparison. It is very difficult to calculate and judge the n redundancy times on $\mathbf{Z}_{n^2}^*$, so Paillier's cryptosystem has higher security. Moreover, in the encryption process, due to the randomness of r, even if the same plaintext is encrypted each time, different ciphertexts are generated. Therefore, it is difficult to attack with plaintexts, and the security of the algorithm is correspondingly improved.

Table. 1 Mathematical problems on which single homomorphic encryption algorithms are based

Algorithms	Hill	RSA	ElGamal	Paillier
Mathematical problems	Linear transformation matrix	Integer factorization	Discrete logarithms in finite fields	Determining data redundancy on N-order

Table 1 shows the mathematical problem on which each algorithm is based. Through the above analysis, it can be seen that the security of the Hill encryption algorithm is general. The security of RSA and ElGamal is relatively good, which can meet the requirements of general encryption, and the security of the Paillier encryption algorithm is the highest.

3.3 Efficiency comparison of single homomorphic encryption algorithms

The hardware and software of this experiment are as the following: CPU is Intel Xeon E3-1225 v3, memory is 16 GB (2*8 GB) 1333 MHz Dual Ranked RDIM, the hard disk is 1 TB 3.5-inch 7200 RPM SATA II, and the operating system is Linux CentOS. In this paper, 50MB files were encrypted and decrypted by Hill algorithm, RSA algorithm, ElGamal algorithm and Paillier algorithm, respectively. The results are shown in Table 2.

Table 2 Time of Encryption and Decryption

Algorithm / Time	Hill	RSA（512）	ElGamal (256)	Paillier (256)
Encryption（s）	26	201	271	299
Decryption（s）	25	384	161	350

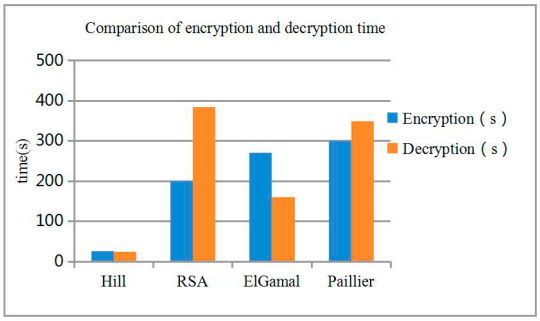

Figure. 1 Comparison of encryption and decryption time

As can be seen from Table 2 and Figure 1, Hill algorithm required the shortest encryption time because Hill algorithm is symmetric encryption. The decryption time of RSA algorithm is 1.8 times the encryption time because the length of the secret key for decryption is twice the length of the public key for encryption. ElGamal's encryption time is 1.7 times as long as decryption time. Because encryption generates ciphertext consisting of two parts, it needs two operations, and decryption has only one operation, so decryption speed is faster. The Paillier algorithm is the slowest because it is modular to the plaintext.

Through the above comparative analysis, we can see that we Hill encryption can be selected for occasions with low- security requirements, and RSA, ElGamal or Paillier can be selected for those with high-security requirements. RSA can be selected if encryption speed is required and ElGamal can be selected if decryption speed is required. Paillier is an additive homomorphism, which is more suitable for the parallel algorithm in the cloud environment.

Although single homomorphic encryption algorithms only have the characteristics of addition or multiplication, they have been applied to some extent in the cloud environment due to a homomorphism.

With a large number of electronic health files stored in the cloud, a data storage scheme by reducing image pixels and based on Paillier encryption was proposed in 2013 for the processing of a large number of image data in the files [12]. Doctors can access the electronic health records in the cloud through the secret key provided by hospitals. In the same year, Meng et al. designed a data retention check scheme based on Paillier and ECC (Elliptic Curve Cryptography) cryptosystem [13] to ensure the integrity and availability of cloud data. In order to protect the privacy of cloud data owners and users, an encryption scheme based on ElGamal and Paillier was proposed in 2013 [14], which proposed a new update and query algorithm for NoSQL (not only structured query language).

4. Fully Homomorphic Encryption Algorithms

Fully homomorphic encryption algorithm refers to an encryption algorithm which has the characteristics of additive homomorphism and multiplicative homomorphism and can perform any multiple addition and multiplication operations. In 2009, Gentry of IBM proposed the first fully homomorphic encryption scheme based on ideal lattices, which can recover re-encrypted data. Since then, more fully homomorphic encryption schemes have been proposed.

4.1 Overview of five fully homomorphic encryption algorithms

4.1.1 Gentry's homomorphic encryption scheme based on ideal lattice

Gentry's first fully homomorphic encryption scheme was based on the ideal lattice. Gentry's scheme mainly includes three steps: the first step is to construct a SWHE, which can handle low-order polynomials and maintain homomorphism; the second step is to squeeze the decryption circuit of the SWHE and reduce the computational complexity of the decryption algorithm in the scheme by " compressing" the decryption circuit, i.e. reducing the number of decryption polynomials. The last step is bootstrapping transformation, using "ciphertext refresh" to process decryption circuits and their extended decryption circuits, thus obtaining a fully homomorphic encryption scheme. The security of the scheme is based on the computational difficulty of bounded distance coding on ideal lattices and the computational difficulty of sparse subsets.

4.1.2 DGHV -fully homomorphic encryption scheme based on Integer

Dijk, Gentry, and Halevi, et al. proposed a fully homomorphic encryption scheme based on modular operation over integers in 2010 [10]. The scheme is still based on Gentry, that is, constructing a symmetric HE based on an integer, then converting symmetric HE into asymmetric SWHE by introducing approximate greatest common divisor (AGCD), and finally reducing decryption times by using compression techniques, the SWHE is transformed into FHE (Fully Homomorphic Encryption). The advantage of this scheme is that no operation based on ideal lattices of a polynomial ring is needed, and all operations are based on an integer. The disadvantage is that the length of the public key is so long that it cannot be implemented on any systems.

$KeyGen(1^{\lambda})$: $x_i \leftarrow D_{\gamma,\rho}(\rho)$ i=0, 1, L, τ, select the maximum number as x_0, and make sure x_0 is odd and x_0 modulo p is even, then $pk^* = (x_0, x_1, x_2, \cdots x_\tau)$, $sk^* = p$. Let $x_p = \lfloor 2^k / p \rfloor$, randomly select the vector of 0,1, only θ is 1, $h = [h_1, h_2, \cdots h_\theta]$, $H = \{i : h_i = 1\}$. Select the random integer $u_i = \mathbf{Z} \cap [0, 2^{k+1}]$, $i = 1, 2, \cdots, \Theta$, satisfy $\sum_{i \in \varepsilon} u_i = x_p \bmod 2^{k+1}$, let $y = \{y_1, y_2, \cdots, y_\theta\}$, $y = \{y_1, y_2, \cdots, y_\theta\}$, $\sum_{i \in \varepsilon} y_i \bmod 2 = (1/p) - \Delta_p$, where $|\Delta_p| < 2^{-k}$, so $pk = (pk^*, y)$, $sk = h$.

$Encrypt(pk, m)$:

$$c^* = (m + 2r + 2\sum_{i \in s} x_i) \bmod x_0 \qquad (11)$$

where $s \subseteq \{1, 2, \ldots\ldots \tau\}$, $r \in (-2^{\rho'}, 2^{\rho'})$, let $z_i = c * y_i \bmod 2$, only the decimal point of z_i is reserved after the $\lceil \log\theta \rceil + 3$ bit, output c^* and z_i.

$Decrypt(sk_i, c^*, z_i)$:

4.1.3 BGV Scheme Based on RLWE

In 2011 Brakerski and Vaikuntanathan designed a fully homomorphic encryption scheme based on LWE problem for the first time [15]. Then Brakerski proposed a fully homomorphic encryption technique, BGV [11], which does not require Bootstrapping, using key exchange and modulo reduction techniques. The basic idea is to use the key exchange technology to convert the ciphertext product into a new one with the same dimension as the original ciphertext after each ciphertext calculation, and then and then enter the next layer of circuit calculation. The noise of ciphertext is reduced by means of modular switching technology.

$KeyGen()$: For j from L to 0, generate $s_j \in R_q^2$ and $A_j \in R_q^{N\times 2}$. $\tau(s'_{j+1} \to s_j) \leftarrow switchKeyGen(s'_{j+1}, s_j)$, where $s_j' = s_j \otimes s_j$.

So $sk = (s_0, s_1, \cdots, s_l)$, $pk = (A_0, A_1, \cdots, A_l, \tau(s_1^* \to s_0), \cdots, \tau(s_L^* \to s_{L-1}))$ .....(13)

$Encrypt(params, pk, m)$:

$$c = m + A_L^T r \qquad (14)$$

where $m \in \mathbf{R}_2, r \leftarrow \mathbf{R}_2^N$ 。

$Decrypt(params, sk, c)$:

$$m = ((\langle c, s_j \rangle \bmod q) \bmod 2) \qquad (15)$$

PK exchange: $c_1 \leftarrow switchKey(\tau(s'_i \to s_{i-1}), c, q_i)$, c_1 is the ciphertext encrypted with pk s_{i-1}, and q_i is the corresponding module. Module exchange: $c_2 \leftarrow scale(c_1, q_i, q_{i-1}, 2)$, c_2 is ciphertext encrypted with pk s_i', and q_{i-1} is the corresponding module.

4.1.4 GSW13 Scheme based on approximate eigenvectors

In 2013, Gentry improved the BGV scheme and proposed a fully homomorphic encryption algorithm based on approximate eigenvectors [16]. The scheme no longer uses mode switching and key exchange technology and solves the expansion of ciphertext dimension caused by ciphertext product, which reduces the length of the public key and improves the space efficiency, but the computing efficiency is not as good as other schemes based on RLWE.

$KeyGen(params)$: select $t \leftarrow \mathbf{Z}_q^n$, output $sk = s \leftarrow (1, -t_1, \cdots -t_n) \in \mathbf{Z}_q^{n+1}$, v is the power of 2. Choose a matrix $B \leftarrow \mathbf{Z}_q^{m\times n}$, $e \leftarrow \chi^m$, let $b = B \times t + e$, define $A = [b \,||\, B]$, that is, A is composed of column vectors b and n matrix B. $pk = A$, where $A \times s = e$.

Plaintext is $m \in Z_q$, selecting uniform matrix $\boldsymbol{R} \in \{1,0\}^{N\times m}$ $Encrypt(params, pk, m)$:

$$C = Flatten(\mu \times I_N + BitDecomp(\boldsymbol{R} \times \boldsymbol{A})) \in \mathbf{Z}_q^{N\times N} \qquad (16)$$

where $Flatten(a) = bitDeomp(bitDecomp^{-1}(a))$

$Decrypt(params, sk, c)$:

$$v_i = 2^i \in (q/4, q/2) \qquad (17)$$

where $v_i \in (1, 2, \cdots, 2^{l-1})$, C_i is line i of C, compute $x_i \leftarrow \langle C_i, V \rangle$, output $\mu' = \lfloor x_i / v_i \rceil$.

4.1.5 Multi-key fully homomorphic encryption scheme based on NTRU

All the fully homomorphic encryption schemes mentioned above use the same public key to encrypt data and perform homomorphic calculations. However, in many cases, when performing homomorphic operations on the ciphertext of multiple users, the keys of different users are different. So how to construct homomorphic encryption with multiple public keys. In 2012, Adriana Lopez-Alt et al. proposed a multi-key fully homomorphic encryption scheme based on NTRU [26]. The scheme is based on NTRU cryptosystem. It mainly constructs a multi-key homomorphic encryption scheme by using the techniques of re-linearization and modulo reduction. It can homomorphically operate polynomial ciphertext under up to N public keys. And the decryption of these ciphertexts requires all the keys used.

$KeyGen(1^{\kappa})$: randomly select the restricted polynomial $u^{(i)}, g^{(i)} \leftarrow \chi(i = 1,2,\cdots,d)$,let $f^{(i)} = 2u^{(i)} + 1$ (where $f^{(i)} \equiv 1 \bmod 2$, and $f^{(i)}$ must be reversible on R), $h^{(i)} = 2g^{(i)}(f^{(i)})^{-1} \in R_{ai-1}$, so $pk = h_0 \in R_{a_\infty}$, $sk = f^{(d)} \in Rq_d$.
Randomly select the restricted polynomial s , $e \leftarrow \chi$, $Encrypt(pk, m)$:

$$c = hs + 2e + m \in R_{q_0} \tag{18}$$

$$Decrypt(sk_i, c):$$

$$m = \mu \bmod 2 \tag{19}$$

where $\mu = f_1 \ldots f_N \times c \in R_{qd}$, $sk_i = f^{(i)}, i \in [N]$.

4.2 Optimization Measures of fully homomorphic encryption algorithm

On PKC in 2010, Smart and Vercauteren proposed a fully homomorphic encryption scheme with a relatively small key and ciphertext size based on the ideal lattices [18] and tried to implement the fully homomorphic encryption scheme for the first time, providing a new idea for later researchers to implement the fully homomorphic encryption schemes. Gentry and Halevi made many optimizations to Smart and Vercauteren's scheme in 2011 [19]. They replaced the "Three-for-two" technology with "carry addition", making the decryption less complex and truly realizing fully homomorphic encryption for the first time. In 2012 Gentry et al. proposed a module reduction method [20] for the continuous module reduction process in bootstrap, which can be combined with SIMD technology to make the speed faster and reduce the size of the public key. In 2015, Gu Chunsheng proposed a fully homomorphic encryption scheme on an approximate ideal lattice [21]. This scheme extends the approximate maximum common divisor to the approximate ideal lattices and uses Gentry's guiding technique to realize the fully homomorphic encryption scheme based on the approximate ideal lattice.

In order to solve the problem that the key length of the integer-based fully homomorphic encryption scheme is too long, Coron et al. proposed an optimization scheme to reduce the key length at Crypto in 2011 [22]. In order to improve the efficiency of plaintext processing, Cheon et al. improved the DGHV scheme in 2013 by proposing an optimization scheme [23] for batch processing of multiple plaintext bits. In 2015, Kim and Lee and others improved the DGHV scheme again [24], which supports SIMD-type operations and has a larger plaintext size, reducing computational overhead.

Lauter et al. tested the homomorphic encryption scheme based on LWE in 2011 [25]. The two messages with different coding forms were converted into one ciphertext, and many optimization measures were proposed for LWE scheme. The practical homomorphic encryption open source software library HELib launched by IBM in 2013 is the BGV scheme based on RLWE, which contains optimization measures for BGV.

In 2014, a non-bootstrapping hierarchical homomorphic encryption scheme based on RLWE was proposed [26]. This scheme introduces the key exchange in the re-linearization, which reduces the complexity of decryption algorithm and improves the homomorphic operation capability. In 2015, a fully homomorphic encryption scheme based on RLWE batch processing was proposed [27], which allows multiple plaintexts to be compressed into a single ciphertext to support SIMD operations, thus reducing the expansion rate of ciphertext.

For the homomorphism calculations in GSW scheme are only matrix-based addition and multiplication, Hiromasa et al. proposed a method of compressing ciphertext and optimizing bootstrap in 2015 [28]. This scheme is the first fully homomorphic encryption scheme that can encrypt the matrix and perform matrix homomorphic operations.

Gaithuru proposed an encryption scheme based on NTRU in 2014 [29]. This scheme gives specific constraints on parameter selection, and by comparing it with RSA and ElGamal, both of which belong to public key encryption, its security is proved.

4.3 Performance comparison of fully homomorphic encryption algorithms

Early fully homomorphic encryption algorithms based on ideal lattice and integer cannot be implemented by programming because the size of the key is too large and the ciphertext expansion ratio after encryption is too large. Many researchers have improved various types of encryption schemes. Each encryption scheme has different improvement methods, and its encryption efficiency is also different. Therefore, this paper compares them from the aspects of mathematical problems, construction methods, circuit calculation complexity, safety indexes. The results are shown in Table 3.

Table 3. Performance Comparison of Homomorphic Encryption Algorithms

Algorithm	DGHV	BGV	GSW13	NTRU
Mathematical Problem	AGCD problem and sparse subset sum problem	RLWE problem	LWE problem	RLWE problem
Construction Method	Bootstrapping	Key exchange, module conversion	Approximate eigenvector	Re-linearization, module conversion, Bootstrapping
Circuit Calculation Complexity	$O(\lambda^{16})$	$O(\lambda \times L^3)$	$\tilde{O}(n(nL)^{\omega})$, $\omega<2.372$	O logN($log\ log\ q+log\ n$))
Safety IndeX	$O(\lambda^{10})$	$(poly(\lambda))^L$	$(poly(n))^L$	$\sqrt{q} \times ploy(n)$

4.4 Application of Fully homomorphic encryption algorithms in cloud environment

Cloud storage security is one of the important security issues in the cloud environment. In order to protect data privacy, the common method is for users to encrypt data and store the encrypted information in the server. Homomorphic encryption technology can process ciphertext data under privacy protection and can directly search, calculate and count ciphertext in the cloud. The application of homomorphic encryption technology in cloud computing mainly has four aspects:

(1) Retrieving encrypted data in cloud computing. With the wide application of cloud computing technology, the encrypted data stored in the cloud will increase rapidly. How to retrieve encrypted data has become an urgent problem. Data retrieval based on homomorphic encryption technology can not only retrieve encrypted data directly but also ensure that the retrieved data are not counted and analyzed. In 2011, Thuraisingham et al. used Hive and Hadoop technology to achieve data access and retrieval on cloud platform [30]. At present, some companies have provided encrypted retrieval services。

(2) Encrypted data processing in cloud computing. Encrypted data processing in cloud computing mainly includes data retrieval, calculation, statistics, analysis. The cloud server directly operates on the encrypted data to fulfill the users' requirements and return the processed data to the users. Users can decrypt the ciphertext after receiving it, which reduces the consumption of data communication.

When applied to medical treatment, a cloud storage system for medical records has been proposed. In 2009, Ristenpart et al. proposed the application of homomorphic encryption in agricultural cloud computing service platform and compared the efficiency of different encryption algorithms [31]. In 2013, new progress was made in the application of fully homomorphic encryption in biometric identification. Hamming distance is often used as an indicator to compare two biological feature vectors. Yasuda et al. proposed a fully homomorphic encryption algorithm based on ideal lattices to calculate hamming distance[32].

(3) Bank of private data. At the emergence of the concept of fully homomorphic encryption, it was predicted that bank of private data could be established. Through cloud computing, user data can be placed in the cloud, and fully homomorphic encryption technology can ensure that the needed information is not counted and analyzed. In 2015, Teaa proposed using hybrid homomorphic encryption to protect data bank privacy in cloud environment[33].

(4) Secure multi-party computation. Bendlin et al. discussed secure multi-party computation and its relationship with homomorphic encryption. Document [34] illustrates how to use homomorphic encryption to solve the general secure multi-party computation problem. In 2012, a cloud data sharing scheme based on the Paillier algorithm was proposed [54], and proxy re-encryption technology was also applied in cloud computing security.

In addition to the above applications, homomorphic encryption technology also has related research and applications in other aspects. Abidin et al. proposed two biological privacy protection schemes in 2014 [35], based on ideal lattice and RLWE respectively. The two schemes have the same distributed architecture. Elbasheer et al. realized digital certificate issuance and signature based on NTRU public key encryption algorithm through JAVA language [36]. Compared with the RSA algorithm, this algorithm has higher efficiency. In 2016, Xiang Shijun and others proposed a reversible image watermarking algorithm in the

homomorphic encrypted domain based on Haar-DWT, which has low data expansion and can completely recover the original carrier and corresponding watermark information [37].

5. Conclusions

With the wide application of cloud computing, the security of cloud platform has become one of the core issues of the cloud computing, which restricts the development of cloud computing. Homomorphic encryption can directly process ciphertext data, effectively ensuring the security of cloud user data. However, there are still some problems to be solved in homomorphic encryption, which can be summarized as follows:

(1) For single homomorphic encryption algorithms: Hill encryption algorithm with symmetric encryption is not safe, so most commonly used encryption algorithms are public key encryption algorithms. However, the current public key encryption efficiency is not high, and the encryption and decryption speed needs to be improved.

(2) For fully homomorphic encryption algorithms : the current circuit homomorphic encryption algorithms should be replaced by the algebraic homomorphic encryption algorithms; the ciphertext expansion rate and computational complexity should be reduced and the computational efficiency should be improved; more research and improvement should be needed in the NTRU based fully homomorphic encryption and vector-based fully homomorphic encryption.

References

[1] Rivest R L, Adleman L, Dertouzos M L. On data banks and privacy homomorphisms[C] Foundations of Secure Computation. New York: Academic Press, 1978: 169-179.
[2] Hill L S. Cryptography in an algebraic alphabet [J]. The American Mathematical Monthly, 1929, 36(6): 306-312.
[3] Rivest R, Shamir A, Adleman L. A method for obtaining digital signatures and public key cryptosystems [J]. Communications of ACM, l978, 21(6): 120-126.
[4] Goldwasser S, Micali S. Probabilistic encryption [J]. Journal of Computer and System Sciences, 1984, 28(2): 270-299.
[5] Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms [J]. IEEE Transactions of Information Theory, 1985, 31(4): 469-472.
[6] Benaloh J, Tuinstra D. Receipt-free secret- ballot elections [C] Proc. of the 26th Annual ACM Symposium on the Theory of Computing, New York, ACM, 1994: 544-553.
[7] Paillier P. Public-key cryptosystems based on composite degree residuosity classes [J]. Proc Eurocrypt, 1999, 547(1) : 223-238.
[8] Boneh D, Goh E J, Kobbi N. Evaluating 2-DNF formulas on ciphertexts [C] Theory of Cryptography - Second Theory of Cryptography Conference, LNCS 3778. Berlin: Springer, 2005: 325-341.
[9] Gentry C. Fully homomorphic encryption using ideal lattices [J]. Proc. of the Annual ACM Symposium on Theory of Computing. 2009, 19(4) :169-178.
[10] Dijk M, Gentry C, Halevi S, et al. Full homomorphic encryption over the integers [C] Proc. of EUROCRYPT'2010, LNCS 6110, Berlin: Springer, 2010: 24-43.
[11] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled) Full homomorphic encryption without bootstrapping [C] Proc. of the 3rd Innovations in Theoretical Computer Science Conference. New York: ACM, 2012 : 309-325.
[12] Aiswarya R, Divya R, Sangeetha D, et al. Harnessing healthcare data security in cloud [C] Recent Trends in Information Technology (ICRTIT), 2013 International Conference on, IEEE, 2013: 482-488.
[13] Meng X L, Ai M S, Yu P P. An improved data possession checking scheme in cloud computing [J]. Advanced Materials Research, 2013, 760-762,1733-1737.
[14] Guo Y B, Zhang L K, Lin F R, et al. A solution for privacy-preserving data manipulation and query on NoSQL database [J]. Journal of Computers, 2013,8(6): 1427-1432
[15] Brakerski Z, Vaikuntanathan V. Efficient full homomorphic encryption from (standard) LWE [J]. Foundations of Computer Science Annual Symposium on, 2011(2): 97-106.
[16] Gentry C, Sahai A, Waters B. Homomorphic encryption from learning with errors: Conceptually- simpler, asymptotically-faster, attribute-based [C] Proc. of the 33rd Annual International Cryptology Conference. Berlin: Springer, 2013: 75-92.
[17] López-Alt A, Tromer E, Vaikuntanathan V. On-the-fly multiparty computation on the cloud via multikey full homomorphic encryption [C] Proc. of the Annual ACM Symposium on Theory of Computing. New York: ACM, 2012: 1219-1234.
[18] Smart N P, Vercauteren F. Full homomorphic encryption with relatively small key and ciphertext sizes [C] 13th International Conference on Practice and Theory in Public Key Cryptography, PKC 2010, LNCS 6056, Berlin: Springer, 2010: 420-443.
[19] Gentry C, Halevi S. Implementing Gentry's fully homomorphic encryption scheme [C] Proc. of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology. Springer Verlag, 2011:129--148.
[20] Gentry C, Halevi S, Smart N P. Better bootstrapping in fully homomorphic encryption [C] 15th International Conference on Practice and Theory in Public Key Cryptography, Springer Berlin Heidelberg, 2012: 1-16.
[21] Gu C S. Fully homomorphic encryption from approximate ideal lattices [J]. Journal of Software, 2015,26(10):2696-2719.
[22] Coron J S, Mandal A, Naccache D, et al. Full Homomorphic encryption over the integer with shorter public keys [C] 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2012: 446-464.

[23] Cheon J H, Coron J S, Kim J, et al. Batch full homomorphic encryption over the integer [C] 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Berlin: Springer, 2013: 315-335.
[24] Cheon J H, Kim J, Lee M S, et al. CRT-based fully homomorphic encryption over the integers [J]. Information Sciences, 2015, 310 (10): 149-162.
[25] Lauter K, Naehrig M, Vaikuntanathan V. Can homomorphic encryption be practical? [C] Proc. of the ACM conference on computer and Communications Security, 2011: 113-124.
[26] Tang D H, Zhu S X, Wang L. Full homomorphic encryption scheme from RLWE [J]. Journal of Communications, 2014,35(1):173-182.
[27] Chen H, Hu Y P, Lian Z Z. Double batch for RLWE-based leveled fully homomorphic encryption [J]. Chinese Journal of Electronics, 2015, 24(3):661-666.
[28] Hiromasa R, Abe M, Okamoto T. Packing messages and optimizing bootstrapping in GSW- FHE [C] 18th IACR International Conference on Practice and Theory of Public-Key Cryptography, PKC 2015, 2015, 699-715.
[29] Gaithuru J N, Bakhtiari M. Insight into the operation of NTRU and a comparative study of NTRU, RSA and ECC public key cryptosystems [C] 2014 8th Malaysian Software Engineering Conference, MySEC 2014: 273-278.
[30] Thuraisingham B, Khadilkar V, Gupta A, et al. Secure data storage and retrieval in the cloud [C] Proc. of the 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom, 2010: 1-8.
[31] Ristenpart T, Tromer E, Shacham H, et al. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds [C] Proc. of the 16th ACM Conference on Computer and communications security, 2009: 199-212.
[32] Yasuda M, Shimoyama T, Kogure J, et al. Packed homomorphic encryption based on ideal lattices and its application to biometrics [C]// Security Engineering and Intelligence Informatics- CD-ARES 2013 Workshops, Berlin: Springer, 2013: 55-74.
[33] Mani M. Enabling secure query processing in the cloud using fully homomorphic encryption [C] Proc. of the 2nd Workshop on Data Analytics in the Cloud, ACM, 2013:36- 40.
[34] Teaa M, Zkik K, Hajji S E. Hybrid homomorphic encryption method for protecting the privacy of banking data in the cloud [J]. International Journal of Security and Its Applications, 2015, 9(6): 61-70.
[35] Abidin A, Mitrokotsa A. Security aspects of privacy-preserving biometric authentication based on ideal lattices and ring-LWE [C] 2014 IEEE International Workshop on Information Forensics and Security, 2014: 60 – 65.
[36] Elbasheer M O， Mohammed T. Signing and verifying certificates by NTRU and RSA algorithms [C] 2015 International Conference on Cloud Computing, 2015: 7149655.
[37] Xiang S J, Luo X R, Shi S X. A Novel Reversible Image Watermarking Algorithm in Homomorphic Encrypted Domain [J]. Chinese Journal of Computers, 2016, 39(3): 571-581.