Big Data Security using Homomorphic Encryption

[1]R.Kanagavalli, [2]Bhagyashri R Hanji, [3]Sushmitha S
[1,2,3]Department of CSE Global Academy of Technology
kanagavalli@gat.ac.in bhagyashri@gat.ac.in sushmitha@gat.ac.in

***Abstract*— :** Due to the swift growth of communication technology, the data size and the rate at which data being generated has considerably increased known as big data era. One of the vital challenges that are faced by the application environments is getting big data secured. Homomorphic Encryption (HE) schemes, a special direction of cryptography can be adopted to address big data security issues. In this paper, we have presented big data security model along with HE schemes.
Keywords— Big Data , Cloud Data Storage, Data Security, Data Encryption, Homomorphic Encryption.

I. INTRODUCTION

The term Big data refer to the datasets that grow so large that they become complex to work with using on-hand database management tools. These datasets need massive softwares that can run on thousands of servers in parallel for work. Big data comes with big values. According to Apache Hadoop in 2010, big data can be defined as datasets whose data values cannot be processed i.e. captured, managed by general computers within an acceptable scope [1]. Big data change the way that the data is managed and used. Big data analytics provide new ways for businesses and government to analyze unstructured data. There are some specific characteristics [2] of big data that make the technology so popular which are as described below.

1. Volume – Big data refers to an enormous amount of data that are generated by machines, networks, social media sites and many more. Processing this high volume of data is a challenging task.
2. Variety- In big data systems, the data originates from various types of miscellaneous sources like websites, mobile phones, sensor devices and many more. From these sources, the data can be of both structured, unstructured and also semi structured. Analyzing these variable kinds of data is not an easy task.
3. Velocity- Velocity means the rate at which data is being generated. In big data domain, many applications generate data in a streaming way which is massive and continuous. Analyzing such real time data is a challenging task.
4. Veracity- Since data is produced from many miscellaneous sources, there may be abnormalities, biasing and noise. They are termed as "Dirty Data". Big data applications require cleaning up of dirty data before processing.
5. Volatility- As the data generated is continuous in nature, there may be some point after which the data becomes futile and hence it should be truncated before further processing.
6. Validity – Validity of Big data refers to the data correctness and accuracy of the same. Big data analytical applications should concentrate on validity of data at all time.

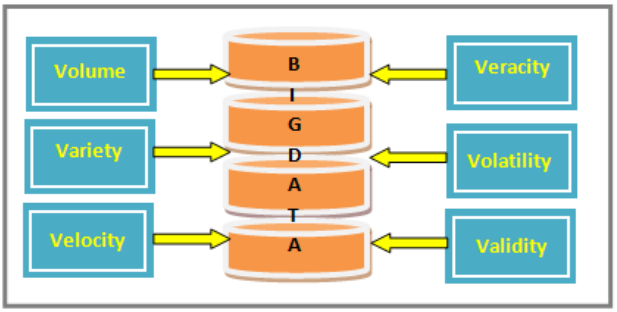

Figure 1. Characteristicsof Big Data

Traditionally the size of data was very small and the cryptographic algorithms were designed to suit the data with a smaller key size [3]. The cryptographic systems depend on keys. The key is exchanged between the parties involved in the communication. In case of distributed environment scenario, the key is shared between the user and the service provider who has exclusive rights to the data. Sometimes, the content that has been encrypted needs to be shared with a third party who may be susceptible. In this scenario, a special type of encryption scheme which allows any third party to work on encrypted data without advance decryption is necessary. In today's technological advancement applications, traditional data encryption makes it impractical to leverage data's value to third party without decryption [4]. This concern makes homomorphic encryption a better choice for big data applications in cloud scenario.
The rest of the paper is organized such as a brief introduction to homomorphic encryption and its types in section II followed by the proposed method along with the experimental results in section III.

II. HOMOMORPHIC ENCRYPTION

According to Greek terminology, the term homo was used to mean the "same" and morphe was used to indicate "form" or "shape" [5]. In modern times, the term homomorphism was coined and used in different areas. In abstract algebra, the term

"Homomorphism" is defined as a plot i.e. an operation or a function preserving all the algebraic structures involving the inputs from the set of domain and outputs an element in the range of an algebraic set [6]. Webster's dictionary defines homomorphism as "a mapping of a mathematical set into or onto another set or itself in such a way that the result obtained by applying operations to the elements of the first set is mapped onto the results obtained by applying those corresponding operations to their respective images in the second set" [7]. In Information security, homomorphic encryption is a cryptographic method used for data encryption. It is an encryption scheme which allows a third party to perform certain computable functions on the encrypted data while preserving the features of the function and format of the encrypted data in order to match the results of operations that are performed on the plain texts after being decrypted [8].
Homomorphic encryption is a computational encryption mechanism applied to data which is already encrypted rather than on original data and the result is taken as the computation is done on plain text. The symbolic notation of a homomorphic scheme is given by Eqn. (1) and can be pictorially represented as shown in figure 2.

$$H = \{Key\ Generation, Encryption, Decryption, Evaluation\} \quad (1)$$

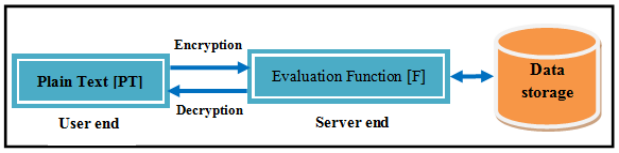

Figure 2. Homomorphic Encryption Scheme

From the figure it is clear that the HE scheme can be characterized by four functions which are described as follows.
Key Generation: In terms of cryptography, the process of generating secret keys is known as key generation phase. HE schemes can be both symmetric and asymmetric. In symmetric HE schemes a secret key is generated which can be used for both encryption and decryption. In asymmetric HE schemes; a secret and public key pairs are generated. The main concept in HE schemes is the usage of keys is not restricted only to a key or keypairs. Here various keys can be used.
Encryption: In cryptography dictionary, encryption is defined as the process where plaintext is converted into cipher text. The conversion takes place in such a way that without the knowledge of the key used for conversion, no authorized access can be made to data. The main goal is to ensure the confidentiality of data that is stored and transmitted via public channels. Encryption is very important process as the process guarantees Authentication, Integrity, Confidentiality and Non-repudiation, the QoS parameters of any security mechanism. Although encryption does not avoid intrusion but makes sure that no prospective interceptor can access logical content.
Decryption: Decryption is the inverse process of encryption, i.e., the cipher text is converted back to plain text. For this process, the encoded data acts as input and converted readable data acts as the output..
Evaluation: This is the procedure that makes HE schemes different from other cryptographic schemes. It takes cipher texts as inputs and produces a cipher text as output corresponding to functioned plain texts. This operation performs the function f () over the cipher texts without the knowledge of corresponding plaintexts i.e

C1, C2= Eval (c1, c2) where c1=Enc (M1) and c2=Enc (M2)

Homomorphic cryptosystems are classified based on the operations performed as

i) Partially Homomorphic Encryption (PHE): They are the cryptosystems in which one operation is allowed to be done on encrypted data. The chosen operation can be either addition or multiplication. Both the operations are not permitted. The permitted operation can be performed unlimited number of times.

ii) SomeWhat Homomorphic Encryption (SWHE): This system allows limited number of either addition or multiplication operation to be performed on encrypted data.

iii) Fully Homomorphic Encryption (FHE): This homomorphic encryption system allows both addition and multiplication operations on encrypted data without any constraints.

The time lines and the description of the existing HE schemes are summarized in Figure 3, Table 1, Table 2 and Table 3.

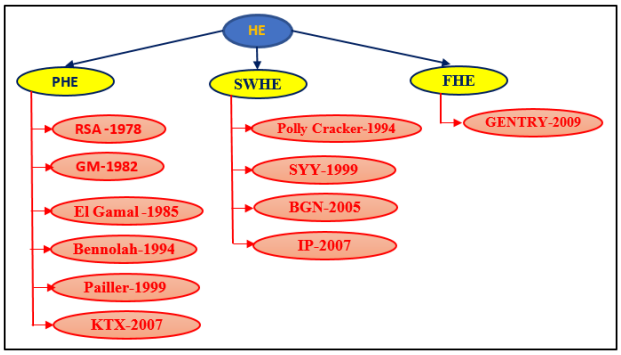

Figure 3. Timeline of HE schemes

TABLE I. PROPERTIES OF PHE SCHEMES

Name of the Scheme	Year	Homomorphic Operation	Description
RSA [9]	1978	Multiplication	The hardness property of finding the factor of two large prime number is used for producing the secret.
GM [10]	1982	Addition	Makes use of hardness property of solving quadratic residuosity problem.
ElGamal [11]	1985	Multiplication	The hardness of solving discrete logarithms is applied for finding the factors.
Benolah [12]	1994	Addition	The problem of finding higher order residuosity is applied.
Paillier [13]	1999	Addition	Method is based on composite residuosity problem.
KTX [14]	2007	Addition	The hardness of solving lattices is used.

The above Table I describes about the existing PHE schemes. From the table it can be noted that the existing PHE schemes works basically on one of the arithmetic operations and are all computationally polynomial hard problems.

TABLE II. PROPERTIES OF SWHE SCHEMES

Name of the Scheme	Year	Homomorphic Operation	Description
Polly Cracker[15,16,17]	1994	Multiplication and Addition	Size of the cipher text grows exponentially and the multiplication is expensive.
SYY [18]	1999	Many AND & one OR/NOT operations.	Method is based over a semi-group. The operations are carried out over different sets and evaluated. There is constant multiplication along with each OR/NOT and hence cipher text grows

			exponentially.
BGN [19]	2005	Arbitrary additions and one multiplication	2-DNF5 formulas on cipher text have been evaluated. Size of cipher text remains constant.
IP [20]	2007	Arbitrary additions and two multiplications	Set of branching programs which are directed acyclic graphs. Size of the cipher text does not depend on the size of the evaluation function.

The above Table II describes about the existing SWHE schemes. From the table it can be noted that the existing SWHE schemes although overcomes the computation hardness, has the disadvantage of growth of cipher text which is exponential.

Table III describes about the existing FHE schemes. From the table it can be noted that the existing FHE schemes are based on the theory of Gentry scheme. They follow the principle of evaluation of the data in depth wise thereby reducing the growth of noise.

TABLE III. PROPERTIES OF FHE SCHEMES

Name of the Scheme	Year	Homomorphic Operation	Description
Ideal Lattice based FHE scheme[21]	2009	Unlimited Homomorphic Operations	Kind of symmetric HE. The method is based on ideals, rings and lattices. Squashing and bootstrapping procedures are used. Decryption complexity is reduced.
FHE Scheme over Integers [22]	2010	Unlimited Homomorphic Operations	Kind of asymmetric HE. The scheme is over integers and based on hardness of Approximate Greatest Common Divisor problem.
FHE with Learning with Error [23]	2011	Unlimited Homomorphic Operations	Method is based on hardness of polynomial time quantum algorithm. Uses RLWE on ideal lattices.
NTRU-Like FHE [24,25]	2012	Unlimited Homomorphic Operations	Method is based on NTRU Encrypt with the difference that noise sampled is changed from a deterministic set to distribution.

III. PROPOSED MODEL

Encrypting data in the traditional way is not trustworthy since it does not reach at leverage data's value. But FHE schemes allow reaching at the leverage data's value. They help to get the same result of applying operation on encrypted data as done on plain data. Although it helps, FHE schemes in cloud computing i.e., big data environment with distributed servers are slow for a practical system. According to the assumption of homomorphic security model proposed by Wang et al. [26], the services like storage, processing and computing will be the responsibility of cloud service provider. With this as a background, we propose our

model with the entities along with their functionalities as shown in Figure 4.

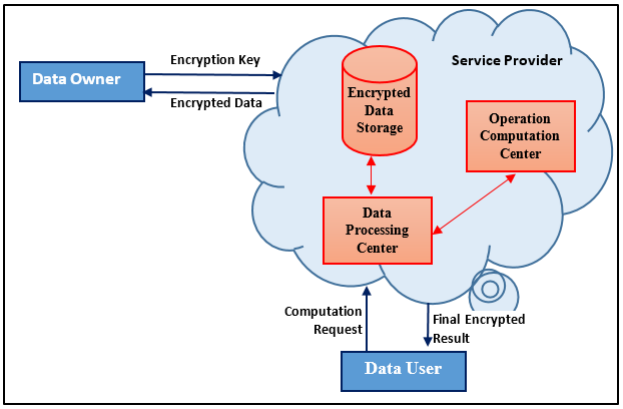

Figure 4. Big Data Processing model with Homomorphic Encryption

Data Owner: Data Owner is the possessor of the raw data. Usually, the enterprise or organizations who generate data takes up the role of data owner. The data owner receives the public key from the service provider and encrypts the data. Then the encrypted data is sent to the cloud database. Takes care of the encryption functionality in a homomorphicsecurity system.
Encrypted Data Storage: This entity is the cloud data base. Responsible for the storage of encrypted data.
Data Processing Center: This entity is responsible for the key generation. It generates the key pair public and secret keys and shares the public key with the data owner. The entity also analyses the users' request and instructs the operation computation center about which data requires certain computation.
Operation Computation Center: This entity is responsible for evaluation functionality. It performs the operations requested by the user. The entity gets cipher texts and functions for operation from the processing center and returns the final encrypted result.
Data User: This entity does the decryption functionality. Data user has the purpose to perform operations on the data. It requests for the operation and receives the final result from the processing center. It decrypts the encrypted result and uses the same.
The proposed method makes use of key switching matrices for encryption. Let 'F' be the data file .The file F is represented as a matrix of 'n' vectors of equal size. Each of these vectors contains 'm' blocks of data. Let 'D' be the key switching matrix. Let 'E' be the encoded file matrix. The steps of encryption are as follows

1. Let key be the key received and let f be the function that chooses random key and f is defined as

$$f:\{0,1\}^{*}\ X\ q\ \rightarrow GF(2^{r}) \qquad (2)$$

The function R to obtain the permutation of the random key vector is defined as

$$R_{key}(\cdot):\{0,1\}^{log_{2}^{n}}\ X\ k \rightarrow \{0,1\}^{log_{2}^{n^2}} \qquad (3)$$

2. The key switching matrix D can be written using n X (n+k) vandermande matrix. The matrix can be written as

$$\begin{pmatrix} & & & & & & & \\ 1 & 1 & 1 & \dots & \mathbf{1} & \mathbf{1} & \dots & \mathbf{1} \\ \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_p & \alpha_{p+1} & \cdots & \alpha_q \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{p-1} & \alpha_2^{p-1} & \alpha_3^{p-1} & \cdots & \alpha_p^{p-1} & \alpha_{p+1}^{p-1} & \cdots & \alpha_q^{p-1} \end{pmatrix} \qquad (4)$$

After a series of elementary row transformations, D can be written as

$$D = \begin{pmatrix} 1 & 0 & \cdots & 0 & p_{11} & p_{12} & \cdots & p_{1k} \\ 0 & 1 & \cdots & 0 & p_{21} & p_{22} & \cdots & p_{2k} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & p_{n1} & p_{n2} & p_{n3} & p_{nk} \end{pmatrix} \qquad (5)$$

Since D is derived from vandermande matrix, it has the property that any 'm' out of the (m+k) columns forms an invertible matrix.
3. The encoded file matrix E can be obtained multiplying (dot product) F and D.

$$E = F \cdot D \qquad (6)$$

$$= \left(E^{(1)}, E^{(2)}, \cdots, E^{(m)}\right) \qquad (7)$$

The method is tested for files with sizes varying from 3MB to 1031 MB and the size of the cipher text, time taken for encryption and decryption are noted and tabulated in Table 4 and the analysis graph is plotted as shown in Figure 5.

TABLE 1. TIME TAKEN FOR ENCRYPTION, DECRYPTION AND SIZE OF CIPHER TEXT

Size of PT(MB)	Size of CT(MB)	Encryption time (seconds)	Decryption time (seconds)
3	3	4.13	2.4
9	9.5	12.39	8.4
16	16.6	22.03	24.7
32	32	44.05	48.25
128	128	176.21	176
256	256	352.43	387.32
512	513	704.85	900.7
1031	1034	1419.34	1743.76

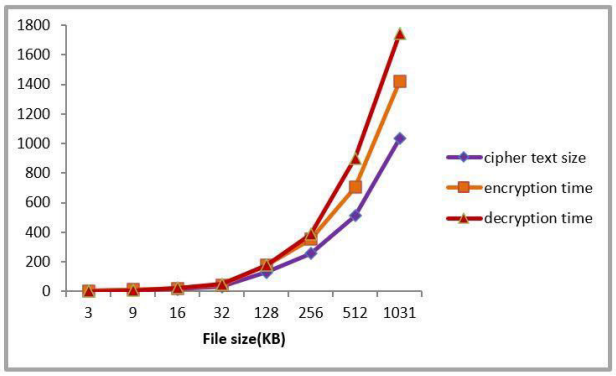

Figure 5. Size of Cipher Text, Encryption Time and Decryption Time

The proposed method is also tested for different computations and the response time taken and the data verification time for files ranging from 16 MB to 1031MB by varying the block size for computation as 10 bytes, 50 bytes and 100 bytes are noted as shown in Table 5 and Table 6 respectively. The response time taken is as shown in Figure 6 and the data verification time is as shown in Figure 7.

TABLE V. TIME TAKEN FOR COMPUTATION RESPONSE

File Size (MB)	Response Time (Seconds)		
	10 Bytes	50 Bytes	100 Bytes
16	11.2	9.2	4.67
32	13.33	11.33	9.03
128	15.49	13.49	12.44
256	18.24	16.2	14.09
512	20.46	18.46	16.2
1031	22.57	21.3	19.4

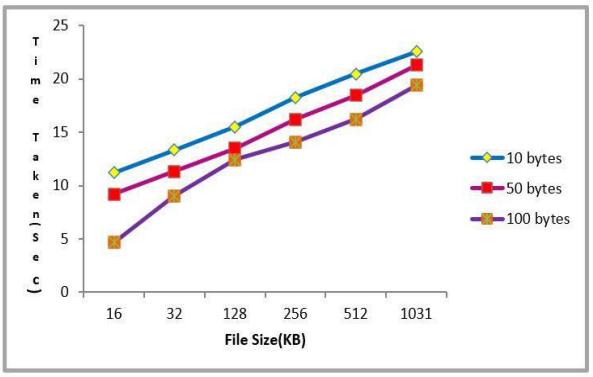

Figure 6. Computation Response Time

TABLE VI. TIME TAKEN FOR DATA VERIFICATION

File Size (MB)	Data Verification Time (Seconds)		
	10 Bytes	**50 Bytes**	**100 Bytes**
16	25.4	23.4	20.4
32	28.1	27.3	25.4
128	33	31.24	29.13
256	41.42	38.14	36.3
512	46.22	44.22	42.21
1031	50.35	49.35	47.12

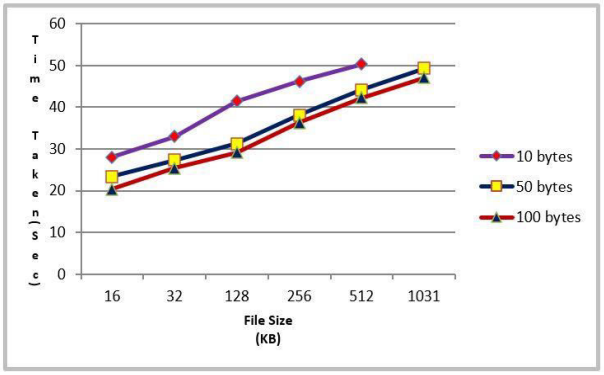

Figure 7. Data Verification Time

The response time denotes the time taken by the server to acknowledge and process the computation request and the data verification time denotes the time taken for downloading, decryption and verification. The response time and the verification time are considerably reduced when the block size is increased for the same file. This shows that the block level operations can enhance the security of data when Homomorphic operations are applied.

IV. CONCLUSION

In this paper, we have presented an explanatory understanding of big data and its characteristics. The various types of HE schemes along with their descriptions are also presented. A big data security model based on HE schemes is presented in this work. The main aim of the work is securing the data via encryption thereby achieving confidentiality. It can be noted that using the proposed method any computation on data can be performed without decryption and hence integrity of data is maintained in contrast to the existing traditional works. The work is still in its infancy stage and in future we would like to test our method for other types of data files

REFERENCES

[1] Chen, M., Mao, S., & Liu, Y. (2014). "Big data: A survey". Mobile Networks and Applications, 19(2), 171-209

[2] Mohammed, A. F., Humbe, V. T., &Chowhan, S. S. (2016, February). A review of big data environment and its related technologies. In Information Communication and Embedded Systems (ICICES), 2016 International Conference on (pp. 1-5). IEEE.

[3] Moura, J., &Serrão, C. (2016). "Security and privacy issues of big data". arXiv preprint arXiv:1601.06206.

[4] M. Beunardeau, A. Connolly, R. Geraud, and D. Naccache, "Fully homomorphic encryption: Computations with a blindfold," IEEE Security Privacy, vol. 14, no. 1, pp. 63–67, Jan 2016.

[5] Henry George Liddell and Robert Scott. 1896. "An intermediate Greek-English lexicon": founded upon the seventh edition of Liddell and Scott's Greek-English lexicon. Harper & Brothers.

[6] Malik, D. S., Mordeson, J. N., & Sen, M. K. (2007). MTH 581-582 "Introduction to Abstract Algebra".

[7] The Miriam-Webster's Dictionary website. [Online]. Available: http://www.merriam-webster.com

[8] Yi, X., Paulet, R., &Bertino, E. (2014). "Homomorphic encryption and applications" (Vol. 3). Cham: Springer.

[9] Rivest, R. L., Shamir, A., &Adleman, L. (1978). "A method for obtaining digital signatures and public-key cryptosystems". Communications of the ACM, 21(2), 120-126

[10] Goldwasser, S., &Micali, S. (1982, May). "Probabilistic encryption & how to play mental poker keeping secret all" partial information. In Proceedings of the fourteenth annual ACM symposium on Theory of computing (pp. 365-377). ACM

[11] ElGamal, T. (1985). "A public key cryptosystem and a signature scheme based on discrete logarithms". IEEE transactions on information theory, 31(4), 469-472.

[12] Benaloh, J. (1994, May). Dense probabilistic encryption. In Proceedings of the workshop on selected areas of cryptography(pp. 120-128).

[13] Paillier, P. (1999, May). "Public-key cryptosystems based on composite degree residuosity classes". In International Conference on the Theory and Applications of Cryptographic Techniques (pp. 223-238). Springer, Berlin, Heidelberg

[14] Kawachi, A., Tanaka, K., &Xagawa, K. (2007, April). "Multi-bit cryptosystems based on lattice problems". In International Workshop on Public Key Cryptography (pp. 315-329). Springer, Berlin, Heidelberg

[15] Fellows, M., &Koblitz, N. (1994). Combinatorial cryptosystems galore!. Contemporary Mathematics, 168, 51-51.

[16] Le, V. L. (2003). "Polly two-a public key cryptosystem based on Polly cracker" (Doctoral dissertation, Ruhr University Bochum, Germany).

[17] Albrecht, M. R., Farshim, P., Faugere, J. C., & Perret, L. (2011, December). "Polly cracker, revisited". In International Conference on the Theory and Application of Cryptology and Information Security (pp. 179-196). Springer, Berlin, Heidelberg

[18] Sander, T., Young, A., & Yung, M. (1999). Non-interactive cryptocomputing for NC/SUP 1. In Foundations of Computer Science, 1999. 40th Annual Symposium on (pp. 554-566). IEEE.

[19] Boneh, D., Goh, E. J., & Nissim, K. (2005, February). Evaluating 2-DNF formulas on ciphertexts. In Theory of Cryptography Conference (pp. 325-341). Springer, Berlin, Heidelberg

[20] Melchor, C. A., Gaborit, P., &Herranz, J. (2010, August). Additively homomorphic encryption with d-operand multiplications. In Annual Cryptology Conference (pp. 138-154). Springer, Berlin, Heidelberg.

[21] Gentry, C. (2009). A fully homomorphic encryption scheme. Stanford University

[22] Van Dijk, M., Gentry, C., Halevi, S., &Vaikuntanathan, V. (2010, May). Fully homomorphic encryption over the integers. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 24-43). Springer, Berlin, Heidelberg.

[23] Brakerski, Z., &Vaikuntanathan, V. (2011, August). Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Annual cryptology conference (pp. 505-524). Springer, Berlin, Heidelberg.

[24] Hoffstein, J., Pipher, J., & Silverman, J. H. (1998, June). NTRU: A ring-based public key cryptosystem. In International Algorithmic Number Theory Symposium (pp. 267-288). Springer, Berlin, Heidelberg

[25] López-Alt, A., Tromer, E., &Vaikuntanathan, V. (2012, May). On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Proceedings of the forty-fourth annual ACM symposium on Theory of computing (pp. 1219-1234). ACM

[26] Wang, D., Guo, B., Shen, Y., Cheng, S. J., & Lin, Y. H. (2017, March). A faster fully homomorphic encryption scheme in big data. In Big Data Analysis (ICBDA), 2017 IEEE 2nd International Conference on (pp. 345-349). IEEE