International Journal of Latest Trends in Engineering and Technology
Special Issue April-2018, pp. 071-074
e-ISSN:2278-621X

FINANCIAL DATA STORAGE USING BGV ALGORITHM

Ms.Indra Priyadharshini[1], Jampala Chandana[2], R. Padmarekha[3], P.Jayasri[4]

Abstract- Cloud computing finds immense applications in all fields. With the massive shift of data storage from offline to Internet, the cost-effectiveness and flexibility required has been achieved. However, the key issues pertaining to data security remain one of the hindrances to data storage of critical applications such as defense, finance etc., Lot of research focus is currently directed towards securing the data stored in Cloud so that such distributed storage and retrieval could be a panacea for some mission critical applications. In this work, we propose to provide homomorphic encryption where encryption of data occurs in the client side and performing analysis in cloud. At the client side, the data can be decrypted at a later stage and put into intended usage
Keyword – Homomorphic Encryption

1. INTRODUCTION

Cloud computing is the space for computation of data in the near future. The existing infrastructure is moving to a state where the number of benefits like flexibility, scalability, efficiency, cost reduction is on constant rise. One major setback in the field of cloud computing is the security. Clients find this as a threat to data breach and want their data to be protected from service providers. Computation has to be performed on the encrypted data such that service providers cannot get information about our data. After computation, the encrypted data is sent to the client for decryption. With this process, we can ensure that our data is not manipulated or hacked.

2. CLOUD COMPUTING AND ITS SECURITY FEATURES

The default behavior of cloud storage for computation is that the cloud user encrypts the data and sends to the cloud. The cloud service provider decrypts the data, performs computation and encrypts the computed data. Further the encrypted data is sent to the client, where decryption happens with the user's secret key. We can easily consider the factor of data breaching in the cloud service side.

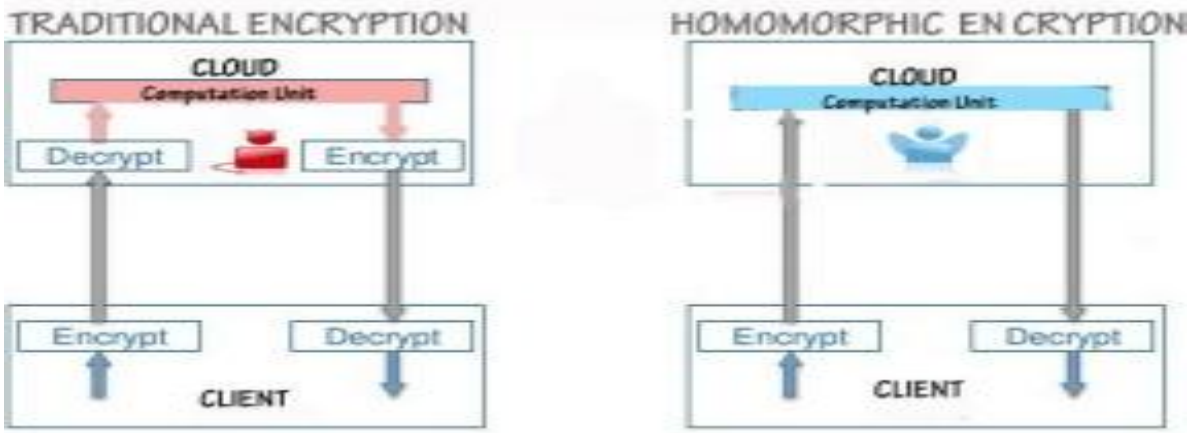

Fig.1: Traditional encryption versus homomorphic encryption

For this problem, clients find it difficult to store data on the cloud and leading to the emergence of homomorphic encryption.

[1] Assistant Professor, Department of Computer Science and Engineering, RMK College of Engineering and Technology, Chennai, Tamil Nadu, India
[2] UG Student, Department of Computer Science and Engineering, RMK College of Engineering and Technology, Chennai, Tamil Nadu, India
[3] UG Student, Department of Computer Science and Engineering, RMK College of Engineering and Technology, Chennai, Tamil Nadu, India
[4] UG Student, Department of Computer Science and Engineering, RMK College of Engineering and Technology, Chennai, Tamil Nadu, India

3. HOMOMORPHIC ENCRYPTION:

The key to solve the security issue of data storage in cloud is the emergence of homomorphic encryption. Assume that M be the data and A be the security parameter. All the algorithms operate on the basic three functionalities:

1. Key generation: Based on the security parameter, we generate the secret key pair sk=(ek,dk) where ek denotes the encryption key and dk denotes the decryption key.
2. Encryption : the algorithm inputs the M data and a encryption key, ek. The output of this process is the E(M,ek)=C where C denotes the ciphertext
3. Decryption: The algorithm inputs the ciphertext C as input and decryption key dk, outputs the data, D(C,dk)=M.

3.1 Properties of Homomorphic encryption:

Mathematically, Homomorphism denotes that transformation of one data set into another preserving the relationship between elements in the data set. Any operation like addition or multiplication performed n the data set tends to apply to the ciphertext generated. If we consider two data sets, m1 and m2, then the corresponding ciphertexts generated are,c1 and c2.The computation inputs ciphertexts c1, c2 and outputs c3=computation(c1,c2) such that m3=D(c3,dk) where m3=m1*m2, the required sensitive data according to our operation.

1. Malleable: Homomorphic encryption is capable of transforming a particular ciphertext to another ciphertext and decrypting to obtain our plaintext.
2. Re-encryption: for a given public key pk, an encryption E is performed such that E(M,r) where M is the message and r is a random value turning into another encryption E(M,r') which cannot be differentiated from the first encryption. Homomorphic encryption, undoubtedly , possess this property.
3. Random self-reducibility: The algorithm breaking a non-trivial fraction of ciphertexts is also capable of breaking a random instance with significant probability.
4. Verifiable encryption/ fair encryption: The encryption provides a mechanism for checking the correctness of the encrypted data without compromising on the secrecy of the data.

3.2 Types of homomorphic encryption:

There exists two types of homomorphic encryption : fully homomorphic encryption and partially homomorphic encryption.
Partially homomorphic encryption accepts less degree polynomial on the encrypted data. The different algorithms under partially homomorphic encryption method are:

1. Unpadded RSA
2. Elgamal
3. Paillier
4. Benaloh
5. Goldwasser-micali.

Fully homomorphic encryption is derived from the partially homomorphic encryption. This encryption applies the decryption process for it to be expressed as a low-degree polynomial that is supported by the partially homomorphic and finally applying a bootstrapping information to obtain fully homomorphic encryption. This encryption is capable of evaluating polynomials of high degree. The different algorithms formed under fully homomorphic encryption are:

1. Gentry's cryptosystem
2. DGHV homomorphic scheme
3. BGV homomorphic scheme
4. FV homomorphic scheme
5. YASHE homomorphic scheme

This paper deals with BGV homomorphic encryption scheme and discusses on the basic notions of the algorithm, its versions and limitations.

4. BGV ALGORITHM

The performance outcome of Gentry's bootstrapping was not satisfactory and all the other algorithms produced exponential noise growth. Brakerski, Gentry and Vaikuntanathan together developed levelled fully homomorphic encryption scheme.BGV is the first algorithm to be implemented for real time applications. It also provides several improvements to Gentry's cryptosystem by reducing the noise growth in multiplication and growth in ciphertext sizes.
The scheme provides a much better approach for lattice based ciphertext using fully homomorphic encryption schemes based on learning with errors(LWE) or ring based learning with errors (RLWE).

Table 1: Versions of BGV showing reducing noise growth

Fhe before bgv	Bootstrapping	exponential

Levelled bgv 1	Without bootstrapping	quasi-linear
Levelled bgv 2	With bootstrapping as optimization	quasi-polynomial
Levelled bgv 3	Batched bootstrapping	linear

Algorithm:

1. Setting up the parameters: input the security parameter, A, total number of levels,L, bit b containing {0,1} for choosing between LWE or RLWE.
2. Generate secret key: After inputting the parameters, we have to generate the secret key for decryption, dk.
3. Generate public key: Input the parameters and the secret key. Generate the matrix and vector. Set the bit to the security parameter along with addition of few key parameters and assign to the public key,ek.
4. Encryption: input the parameters,ek and M,data set. Form the ciphertext as C=E(M,ek).
5. Decryption: input the cipher text and the decryption key and perform decryption to obtain our sensitive data.

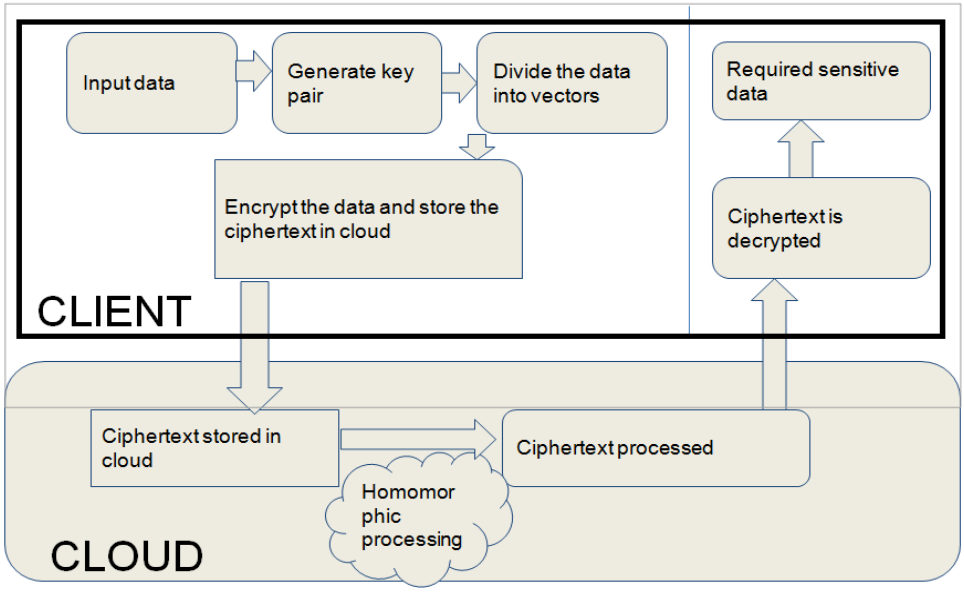

Fig.2: Block diagram showing homomorphic encryption

Module description:

Client module:

1. In this process, the financial data is obtained.
2. Generation of key pair takes place.
3. Using above parameters, we get the vector.
4. Data is encrypted using the public key and sent to the cloud for storage.
5. Data is decrypted to get our sensitive information.

Cloud module:

1. With the encrypted data, we perform the homomorphic encryption processing or operations required.
2. The resultant data is stored in cloud and sent to the client.

5. CONCLUSION

BGV is a leveled fully homomorphic encryption. The BGV algorithm can be used for the security of integer polynomials. It is beneficial to keep all data encrypted and to perform computations on encrypted data. Fully homomorphic encryption can be useful in solving a number of practical problems in cryptography. Two such examples are verifiable outsourcing computation and constructing short non-interactive zero knowledge proofs.

Security of cloud computing on the basis of homomorphic encryption has received a lot of attention in recent years. It provides a platform for performing large scale computations and data storage, statistical analysis, query processing on encrypted form of private data, thus providing extreme confidentiality to data.

6. APPENDIX

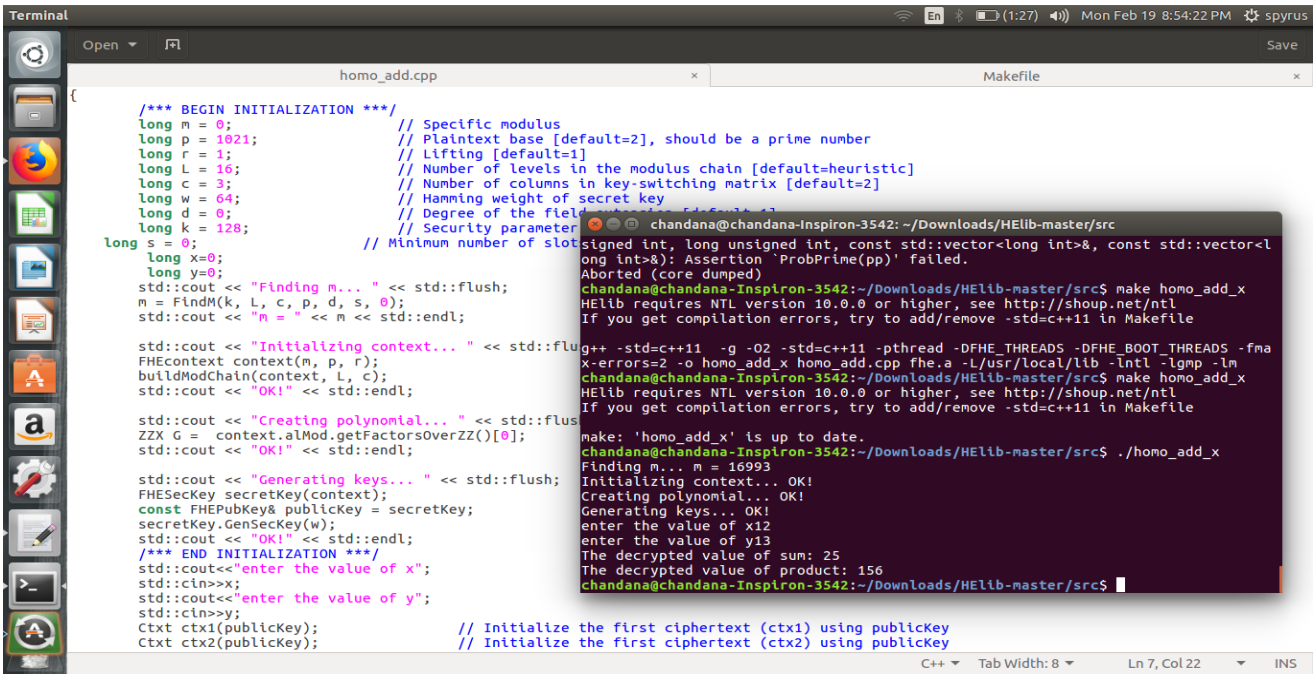

Fig.3: Homomorphic encryption execution

This figure shows us the code along with the execution of random integers using HElib software in our local machine.

7. REFERENCES

[1] Jaydip Sen," Theory and Practice of Cryptography and Network Security Protocols and Technologies",July 7,2013.

[2] Gaurav Somani, Sourabh Garg, "Homomorphic Encryption Algorithms for Securing data against Untrusted Cloud ", July 2016.

[3] Grant Frame,"HEIDE: An IDE for the Homomorphic Encryption Library HElib", June 2015.

[4] Thomas M. DuBuisson, "Secure Computation with HElib" , May 13,2013.

[5] Hubert Hesse, Christoph Matthies, Robert Lehman, "Introduction to Homomorphic Encryption", 2013.

[6] Wenjie Lu, " Ring- Learning With Errors & HElib", 2014.

[7] Shai Halevi, Victour Shoup, " Algorithms in HElib".

[8] Craig Gentry, Shai Halevi, Chris Peikert, " Field Switching in BGV- Style Homomorphic Encryption", " Sum of Encrypted Vectors", "2 + 3using HElib" September 2013.

[9] Mateus S.H. Cruz, " I/O of keys and Ciphertexts using HElib".

[10] Kevin Lin, " How to install GMP library on Ubuntu", January 13, 2015.

[11] PwnHomeResearch, " Guide to HElib", May 3,2013.

[12] Clara Moore, Maire O'Neil, Elizabeth O'Sullivan , " Practical Homomorphic encryption- A Survey", July 28, 2014.

[13] Feng Zhao, Chao Li, Chun Feng Liu, "A cloud computing security solution based on fully homomorphic encryption", March 27,2014.

[14] Deepti Mittalm Damandeep Kaur, Ashish Aggarwal, "Secure Data Mining in Cloud using Homomorphic Encryption", January 22, 2015.

[15] V.Biksham, D.Vasumathi, " Homomorphic Encryption Techniques for securing data in Cloud Computing : A Survey", February 6, 2017.