International Journal of Engineering & Technology, 7 (4.17) (2018) 71-74

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

A Survey on Homomorphic Encryption in Cloud Security

Kavitha C.R.[1]*, Bharati Harsoor[2]

[1] *Research Scholar, Dept. of Information Science & Engineering, PDACE, Kalburgi, India*
[2] *Professor, Dept. of Information Science & Engineering, PDACE, Kalburgi, India*
**Corresponding Author E-mail: kavicr@rediffmail.com*

Abstract

An outsourcing of data is increasing the data storage in Cloud. These raise numerous new challenges of privacy concern for persons and business. Sending data in the encrypted form to the cloud is a common approach to handle the privacy concern. Homomorphic Encryption technique is used to carry out significant computations on the data in the cloud. Random computations over ciphertext are allowed in Fully Homomorphic Encryption. Many solutions using fully homomorphic encryption have been proposed and also many researchers have tried to improve, proving efficiency is very hard. In this paper, Delegated Parallel Homomorphic Encryption is proposed. Also, an analysis has been made to exhibit various applications in the real world. The system must work efficiently without compromising the required cloud security services.

Keywords: *Delegated Parallel Homomorphic Encryption; Fully Homomorphic Encryption; MapReduce; privacy; security;*

1. Introduction

Resource sharing is an essential aspect of efficient and secure sharing of data with the other authorized users in cloud computing. The cloud computing platforms and storage development supports users to outsource storage and computations over the data. The worth of private data in business and privacy concerns are always a challenge to the acceptance of cloud services by business as well as clients. A brilliant way to address the privacy concerns is to use data stored as encrypted, and perform computations on encrypted data in the cloud. One such method is Fully Homomorphic Encryption that permits random computations on encrypted data [1] [4]. The basic encryption scheme to accomplish security is the Caesar cipher which is not very secure. Modular exponentiations used in the public key encryption schemes are more secure and the drawback is modular exponentiation is a complex operation. A simple and good scheme if initiated from the scratch is as follows:

Key-Gen: Select an odd integer as a Key in some interval p ε $[2^{n-1}, 2^n]$ with n being no. of bits.

Encrypt (m, p): Used to encrypt an input bit m ε {1, 0}, the ciphertext and plaintext has same parity, ciphertext c = (m+ p*y + 2r) mod p, where 2r < p/2 is an absolute value and y, r are random integers in some intervals.

Decrypt (p, c): Used to decrypt as *c mod p mod 2* [5].

2. Related Work

Preserving privacy on the cloud data consists of the dimensions availability, integrity and confidentiality [2]. Direct consequence on privacy of users happens if the Data confidentiality or integrity is breached [6]. Availability of information ensures that authoritative users are eligible to access the required information. A fundamental need is to have the privacy protection of an individual in big data storage system. This requirement is satisfied using some existing mechanisms. The approaches for privacy protection of the user on cloud stored data are as follows [3]:

- Attribute based encryption (ABE) is one where attribute is an Identity of a user in the Access control.
- Homomorphic encryption is used in ABE or IBE schemes to update cipher text of the receiver [7] [8].
- Storage path encryption is used to secure big data storage on clouds.
- Hybrid clouds are a mix of both private cloud and public cloud services [20].

3. Motivation

The major deficiencies of several modern techniques have given us the necessary challenges to carry out the current investigation on the Cloud Security.

- An appropriate cryptography technique is suggested to attain safe data storage and transaction in the cloud computing.
- The data privacy problem faced in the third party auditing cannot be steered clear of completely with the encryption technique, though it may be just converted into the composite key management domain.[26]
- The data safety guaranteed by means of the Privacy-Preserving Public Auditing is not effective.
- It may not be easily possible for any user to access the entire data of significance from the cloud data centre. Because several cloud service providers accumulate the needed data. Therefore, a gloom of doubt pervades the users, when the data is accessed through the medium of the cloud service providers.

4. Preliminaries

a. Homomorphic Public Key Encryption consists of Key-Generate, Encrypt, Decrypt and Evaluate algorithms. The

Evaluate algorithm reads input as a circuit C, a tuple of ciphertexts, public key p_k, and results in another ciphertext c [9].

b. Fully Homomorphic Encryption FHE has Key-Generate, Encrypt, Decrypt and Evaluate applied on a class C1 of Boolean circuits, if it works correctly for every circuit C ε C1. Several efficient parts of a fully homomorphic encryption has been implemented. The system produced is "somewhat" homomorphic that can perform only few calculations, but the speed is good enough to use in real software. Additions and a few multiplications are applied on a segment of ciphertext sent to the system which is sufficient for most of the services [1] [10]. Certain statistical functions and analysis like logistical regression could be done to predict the probability of a person getting heart attack. The complexity of the computations used for prediction can be increased practically in Fully Homomorphic encryption [10]. Group Homomorphic Encryption (GHE) is the public- key encryption [12] which performs operation on ciphertext [11] [15]. The extension of GHE is FHE scheme [13]. Somewhat Homomorphic Encryption (SHE) performs operations on the ciphertext with particular group of functions. Arbitrary number of a type of operations is supported in this scheme.

c. Parallel Homomorphic Encryption (PHE) computes using an evaluation algorithm over encrypted data to execute efficiently in parallel [14]. In PHE scheme, a client outsources using cluster of w machines for the computation of a function f on input x which is private [16]. The client encrypts the private input x, then sends function f and the ciphertext to the controller. Controller creates n jobs which are distributed to the workers using the ciphertext. Then the jobs are executed by the workers in parallel. Once the computation is completed, the client gets ciphertext through which client recovers f(x) by decrypting the ciphertext. The variant of PHE is that even function f is hidden which is referred as delegated PHE [14]. A token T is created by running the token generation algorithm which reads f as input and information about f is not revealed. This token is used to return encryption of f(x) by executing the evaluation algorithm [14] [22].

d. MapReduce-Parallel Homomorphic Encryption MapReduce has Parse, Map, Partition, Reduce and Merge algorithms. The Parse algorithm reads n-bit input x and gives label/value pairs $\{(l_i, v_i)\}_i$ set. Map algorithm takes input pair (l_i, v_i) and outputs intermediate pairs $\{(\lambda_j , \gamma_j)\}_j$ set, where λ_j is from some label space ʌ.The input pairs and intermediate pairs need not belong to the same space. Partition algorithm reads intermediate pairs set as input and results in the partition sets $\{P_l\}_l$. The Reduce algorithm reads label λ, a partition P_l as input and produces string say z. Merge algorithm reads a set of strings $\{z_r\}_r$ to merge and outputs y. The crucial parts are Map and Reduce algorithm which need to be stateless to work properly [23].

e. Paillier Encryption- In this method, additive homomorphism is the major focus, without revealing the plaintexts it computes a new ciphertext of plaintext (p1 + p2) depending on two ciphertexts of plaintext p1 and p2 [17]. Specifically, for the given [p1] and [p2], compute [p1+p2] = [p1]. [p2] where [p] represents Paillier ciphertext of a given plaintext p.

f. Boneh-Goh-Nissim (BGN) Encryption- This is a somewhat homomorphic encryption. By using homomorphic property it computes a random number of additions and a multipli cation on enciphered data [18]. Specifically, for the given $// m1//$ and $// m2//$, compute $// m1 + m2// = // m1// \Theta // m2//$

- Given $// m1//$ and $// m2//$, computes $// m1 \times m2// = // m1// \otimes // m2//$ where $// m//$ denotes the BGN ciphertext for a plaintext m, Θ and ⊗ represents the corresponding operations in the domain of the ciphertext. Paillier and BGN are used in evaluating applications with different privacy desires leading to secure association rule mining.

5. Challenges for Encrypted Programs Execution

The general purpose computer is configured to perform required operations in solving a problem and commonly the program runs on unencrypted data. The sequences of operations in the form of instructions are stored in the memory. Many new challenges are put forth by the algorithms on encrypted data. Important challenges are:

- Handling of arithmetic operations: The all-purpose computers handle usual computations such as additions, multiplications, comparisons, equality, bitwise operations using ALU. Encrypted programs insist the need for complex operations in FHE operations. Efficient tackling is needed in the code translation from an unencrypted to encrypted program [12].
- The conditional operations are encrypted, therefore branching and termination are done depending on encrypted conditions [12] [21].
- Another major challenge is handling recursion with encrypted operations. The existing stacks provide normal operations on unencrypted form [12].

6. Basic Operations To Handle Encrypted Variants

An algorithm consists of specific mathematical steps and logical statements. Many variables and instructions are to be executed. The very first step is the translation of any algorithm with the variables in encrypted domain [13]. The input integer variables will be translated to their corresponding type encrypted variables. Sequence of instructions defined on variables is either function to be executed or a single operation. Identification of the different class of plaintext operators to handle those operations on ciphertext is done first and then respective encrypted counterparts are designed. Many standard programming languages classify the operations or manipulations into different types:

- Arithmetic operators include additions, subtractions, multiplications and divisions of two operands.
- Relational operators check the equality of values of two operands. Also decision on the lesser or greater relations are made between two operands.
- Assignment operators assign values from right to left operands.
- Logical operators and Bitwise operands include OR, AND and NOT operations on the operands [20].

7. Proposed Work

The sequential homomorphic encryption involves private outsourced computation. Due to the increase of computations done on massive datasets in the cloud, clusters of machines are required for the computation [14]. The following are proposed to address this:

- PHE and delegated PHE considers MapReduce parallel computation model in MapReduce parallel PHE schemes. This lead to the practical significance of MapReduce model in the cloud-based clusters [19].
- Randomized Reductions (RR) are done with a few variables on univariate and multivariate polynomials. The reduction in univariate polynomials is theoretically secure where as the reduction in multivariate polynomials are proved to be secured based on multiple dimensions [9] [14] [24].
- A broad transformation to a MR-parallel HE scheme from RR is done so that given any public-key HE scheme evaluates the reductions' recovery algorithm. The MR-Parallel scheme is said to be C-homomorphic for any function within a class C if RR works [25].

- A delegated MR-Parallel HE on any function f is computed by running evaluation of a single polynomial of input values [16].
- MR-parallel HE is used to evaluate a variety of database queries on encrypted datasets including simple set membership testing and complex keyword search.

7.1. Methodology/Modules

Delegated Map Reduce Parallel Homomorphic Encryption scheme consists of:
Step 1: Generate algorithm takes input security parameter s and gives a key K.
Step 2: Encrypt algorithm takes key K, a message x that belongs to some message space X as input and results in ciphertext c.
Step 3: Token algorithm takes input key K, a function f and generates a token T.
Step 4: Parse algorithm takes input token T, ciphertext c and gives input pairs (l_i,v_i) sequence.
Step 5: Map algorithm takes input pair (l,v) and gives intermediate pairs (λ,γ) sequence.
Step 6: Partition algorithm takes input pair (λ, γ) and produces a value h that belongs to some space H.
Step 7: Reduce algorithm reads label λ, partition P of intermediate values and gives an output pair (λ, z).
Step 8: Merge algorithm reads a set of output pairs and results in a ciphertext c^1.
Step 9: Decrypt algorithm reads key K, ciphertext c^1 as input which outputs y. Correctness is proved by y = f(x) [19].

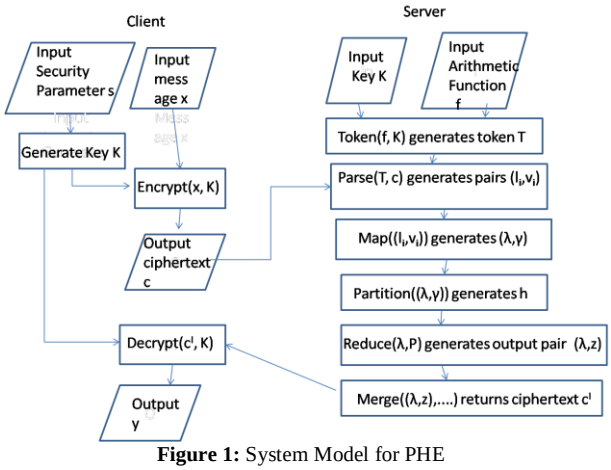

Figure 1: System Model for PHE

The system model of PHE is as shown in Figure 1. The client encrypts the message using key K and outputs ciphertext c. Ciphertext c is sent to the server for applying PHE scheme. Server generates token using an arbitrary function f and key K. Then parse, Map, Reduce, Merge and sends the ciphertext c^1 to the client. Client decrypts using K and outputs y. This is same as f(x).

8.Applications

a. Simple database queries: The simplest database query uses homomorphic encryption for set membership. The query set(x, S) gives an n-bit string such that the i^{th} bit indicates whether $x_i \varepsilon S$ and S is subset of F.
b. Keyword search: This is complex query used to search keyword in the dataset. Let $x = (w_1, v_1), \ldots(w_n,v_n)$ be a dataset where w_i and v_i are in F. The query KS(x, w) produces output sequence $(z_1, .. z_n)$ such that $z_i = v_i$ if $w_i = w$ and such that $z_i = 0$ if $w_i \neq w$.
c. Medical Applications: Patient Controlled Encryption is a private medical records cloud storage system in which healthcare providers encrypts all patients' medical record data before uploading into the cloud storage system. The access and sharing of the record is controlled by the patient using secret keys shared with specific providers. This system provides only cloud search service without allowing any computation on it [1]. On behalf of the patient, the capability to perform operations on the ciphertext is added in the FHE implementation [19].
d. Financial Applications: This is an application in the financial companies, in which computation is on both private data and private function [1]. To make investment decisions, data about the companies, inventory, their stock price and performance is relevant [1]. Streamed data helps in decision making for trading purposes. Computation of functions is on private data. The predictive models for performance of stock price and the financial analysis are needed. These models are maintained private to preserve their advantages and investment. Few functions can be computed privately with FHE. The customer uploads function in the encrypted form to the cloud [1] and the streaming data is encrypted with the public key of the customer which is then uploaded to the cloud. Encrypted description of the program is applied by cloud to evaluate the private function on the received encrypted inputs. After the complete processing, cloud sends output to the customer in the encrypted form [1] [19].
e. Advertising and Pricing: Consider an advertiser company who uses contextual information to target advertising for the customers [1]. Mobile devices are commonly used by the customer to upload related information like location, time of the day, and information through email or browsing activity about the customer including Suppose the information is uploaded constantly from video devices either from a video stream through a camera or pictures of objects like faces or brands which are identified automatically [1]. Information is uploaded to cloud server so that it is made accessible to the company. Some computation using functions of the contextual data are done by the company and targeted advertisement is determined to send to the consumer's mobile device. The problem is the privacy attack which provides detailed information to the server regarding the advertising company or consumer. In the encrypted version, the encrypted contextual data are sent to the server. The encrypted advertisements are uploaded by advertiser to the server; the server then performs computation on a function of the encrypted inputs which identifies the encrypted advertisement to send to the consumer. This function may be either private or not. All contextual data and ads are encrypted with the consumer's public key. Then the cloud operates on this data, and the consumer can decrypt the received ad. As long as the secure FHE encryption is employed it is assured that the advertisers and cloud don't learn about the consumer's data [1].

9.Conclusion

We have described various homomorphic encryption schemes that use simple integer arithmetic operations. The Proposed system model of Delegated Parallel Homomorphic Encryption can be used in various applications. The primary open challenge is to improve the scheme efficiency to the best possible extent. Therefore optimization is very much essential to reduce the time required and the CPU utilization must be increased as the resources are shared using the cloud environment.

10. Future Scope

Optimization of communication with the cloud: FHE is the solution to help the problem of large sized ciphertext. In FHE scheme, a client uploads encrypted data to cloud; the cloud operates and

sends encrypted output to the client [1]. Each ciphertext is large size as the functions are having large number of multiplications. AES-256 is used for all encryptions performed by client [1]. AES-256 decryption is done on FHE-encrypted entries. The client uploads the FHE-encryption of its AES-256 secret key K as FHE(K) to Cloud. Then, Client sends $AES_K(m)$ to Cloud for each segment of content m encrypted with secret key K. Cloud computes and returns FHE-encryptions of input functions. On the Cloud side, first the cloud computes the FHE-encryption of received ciphertext say $FHE(AES_K(m))$ using some function f and forwards it to the client [1].

Acknowledgements

This is a survey work related to my research domain. I would like to thank all the people who have been an inspiration to initiate this work.

References

[1] Kristin Lauter, Michael Naehrig, and Vinod Vaikuntanathan, Can Homomorphic Encryption be Practical?, https://eprint.iacr.org/2011/405.pdf.
[2] Siddhi Khamitkar, A survey on Fully Homomorphic Encryption**,** IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 6, Ver. III (Nov – Dec. 2015), PP 10-14.
[3] Rajnish Choubey, Functional Encryption Algorithm for Communication in Cloud Computing based onAttribute Based Encryption (ABE) International Journal of Innovations in Engineering and Technology (IJIET), Volume 4 Issue 4 December 2014, ISSN: 2319 – 1058, pp 299-303.
[4] Arjunsingh Yadav, Priya Tonde, Priyanka Yewale, Smita Chavan, Proposition for Complete Homomorphic Encryption using Grids in Cloud Computing, IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015, ISSN 2348 – 7968, pp 313-316.
[5] Marten van Dijk, Craig Gentry, Shai Halevi and Vinod Vaikuntanathan, Fully Homomorphic Encryption over the Integers, https://eprint.iacr.org/2009/616.pdf.
[6] Priyank Jain, Manasi Gyanchandani and Nilay Khare, Big data privacy: a technological perspective and review, Journal of Big data, 2016.
[7] Maha TEBAA, Said EL HAJII, Secure Cloud Computing through Homomorphic Encryption, International Journal of Advancements in Computing Technology (IJACT) Volume5, Number16, December 2013, pp 29-38.
[8] Mohd Rahul, Hesham A. Alhumyani, Mohd Muntjir, Minakshi Kamboj, An Improved Homomorphic Encryption for Secure Cloud Data Storage, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 8, No. 12, 2017,pp 441-446.
[9] Mitchell Harper, Fully Homomorphic Encryption, https://sites.math.washington.edu/~morrow/336_14/papers/mitchell.pdf.
[10] Tom Simonite, A Cloud that Can't Leak, https://www.technologyreview.com/s/424942/a-cloud-that-cant-leak, August 2011.
[11] C. Gentry, S. Halevi, and V. Vaikuntanathan. A simple bgn-type cryptosystem from lwe. In Advances in Cryptology - EUROCRYPT '10, pages 506{522. Springer, 2010.
[12] Chatterjee, Ayantika, and Indranil SenGupta. "Translating Algorithms to handle Fully Homomorphic Encrypted Data on the Cloud", IEEE Transactions on Cloud Computing, 2015.
[13] Chatterjee and Sengupta: Translating Algorithms to Handle Fully Homomorphic Encrypted Data on the Cloud, IEEE Transactions on Cloud Computing, Vol. 6, No. 1, 2018.
[14] Seny Kamara, Mariana Raykova, Parallel Homomorphic Encryption, https://eprint.iacr.org/2011/596.pdf.
[15] Michael Clear and Ciar´an McGoldrick, Attribute-Based Group Homomorphic Encryption and Additively Homomorphic IBE, https://eprint.iacr.org/2017/752.pdf.
[16] Seny Kamara, Parallel Homomorphic Encryption https://www.microsoft.com/enus/research/wpcontent/uploads/2016/02/phe.pdf.
[17] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes." Springer, 1999, pp. 223–238.
[18] Craig Gentry Shai Halevi Vinod Vaikuntanathan, A Simple BGN-type Cryptosystem from LWE, March 30, 2010, https://eprint.iacr.org/2010/182.pdf.
[19] www.research.microsoft.com.
[20] www.journalofbigdata.springeropen.com
[21] https://iacr.org/archive/tcc2007/43920574/43920574.pdf
[22] Advances in Cryptology -- CRYPTO 2014: 34th Annual Cryptology, https://books.google.co.in/books?isbn=3662443813
[23] https://www.tutorialspoint.com › Map Reduce › MapReduce – Partitioner
[24] https://www.microsoft.com/en-us/research/.../parallel-homomorphic-encryption/
[25] www.academia.edu/2751344/Parallel_Homomorphic_Encryption
[26] Sheik Al Farhan, Kavitha C. R., End-to-End Encryption Scheme for IoT Devices Using Two Cryptographic Symmetric Keys, IJCTA (International Journal of Control Theory and Applications), Volume 9, Issue 20, September 2016 Pages: 43-49.