International Journal of Computer Applications (0975 - 8887)
Volume 160 - No.6, February 2017

Homomorphic Encryption Techniques for securing Data in Cloud Computing: A Survey

V. Biksham
Associative Professor
Dept. of CSE
CMR Engineering College, Hyderabad-India

D. Vasumathi, PhD
Professor
Dept. of CSE
JNTU University, Hyderabad-India

ABSTRACT

Cloud computing is an arising computational model, where data and its several services associated with its scalable data centers in the cloud and can be obtained from the Internet. Computing gives an added amount of risk as vital services which are usually deployed to any third party, which creates the difficulty to enable data security, privacy factor, confidentiality, integrity, and authentication. Most of the users prefer to store their data inside the cloud in an encrypted/unoriginal format to decrease the security concerns. However, to perform any operation on data at server, cloud needs to first decrypt the data. This operation might cause the challenging issues like - confidentiality along with privacy of confidential data, stored inside the cloud.
Here, This paper presents state of the art in this Homomorphic Encryption (HE) domain, and solve the problems of confidentiality and privacy of stored data in a cloud. HE is a kind of encryption mechanism that give ability to users for computations to be prosecuted on cipher text itself, thus producing an unoriginal/encrypted result when decrypted it shows similarity on the result of operations prosecuted on the plain text. Homomorphic Encryption is generally of two types i.e. Partial Homomorphic encryption (PHE) and Fully Homomorphic Encryption (FHE). FHE considered to be as more secure and efficient in the form of third party computations, since it gains the advantage of both properties - Additive as well as multiplicative homomorphism. Based on the research done in past years, identification of the problem in the existing system is also presented in this paper and have given our future research directions.

Keywords

Cloud Environment, Data security, Homomorphic encryption, Privacy.

1. INTRODUCTION

Cloud computing is having the benefits of centralized large computational power, space and efficiency, so that the customers/clients can outsource their complex problem to the cloud for computation purpose. Also, it suffers from the new security challenges like customer's data privacy, confidentiality and checkability. Cloud is having the great potential of robust computational power to the aggregated management of the elastic resources. To protect the client's personal data from unauthorized use, customers need to encrypt their data prior to outsourcing, but further performing the computations on this encrypted data makes a very hard problem for cloud server. There exist extremely powerful intention to provide the security at several levels eg: network, host, application and data etc.
Homomorphic Encryption technique enables to perform the computational functionalities on unoriginal data itself, without the knowledge private key, the individual customer is the only possessor of private secret key. Whenever result is decrypted for any computational function, it is the similar as if we had prosecuted the calculation procedure on the unanalyzed and unstructured data. In 2010, Gentry has proposed Fully Homomorphic Encryption (FHE) mechanism [13] allows the user to perform additive and multiplicative homomorphic operations on encrypted data. It was an evolutionary breakthrough in the field of cryptography. Still, it is not much practical in real time scenarios. Only either multiplicative or additive homomorphic computation is allowed in a partially homomorphic encryption mechanism. An algorithm is considered as fully homomorphic encryption, if both additive and multiplicative homomorphism properties are satisfied simultaneously. "Fully Homomorphic Encryption" is used to store data in "DynamoDB" of "Amazon Web Service(AWS)" public cloud [1]. "Amazon DynamoDB" collection is having a fully distributed recovery, NoSQL database service, entirely proposed by Amazon.com is as part of the "Amazon Web Services(AWS)" collection. "DynamoDB" produces a similar kind of data model and extracts its name from Dynamo but has a different underlying implementation. Users operational procedures are carried on unoriginal/encrypted data itself inside public cloud. When results requirement is there, they can be simply downloaded on the client machine. In this particular scenario, users confidential data will never be stored in form of plain text on a public cloud.

1.1 Motivation to the problem

In today's scenario, computing addresses an extra amount of risk as essential services be normally deploy to a third party, which makes it challenging to maintain the security outlines like - data security, privacy, confidentiality, integrity, authentication etc. Most of the users favor to store their data inside cloud environment in an unoriginal form to decrease the security concerns. However, to execute any operation on the data, which is residing at server, cloud needs to first decrypt the data. This operation might create the confidentiality and privacy issues of data stored in the cloud.
Homomorphic Encryption(HE) is a kind of encryption mechanism that give ability to users for computations to be prosecuted on ci-

pher text itself, thus producing an unoriginal/encrypted result when decrypted it shows similarity on the result of operations prosecuted on the plain text. [18] Homomorphic Encryption solves the problems of confidentiality and privacy of the stored data inside the cloud.

1.2 Organization of the paper

In the remaining of this paper - section 2 presents some preliminaries that are used in this paper. Section 3 presents the related work, done in this area. Section 4, points out some security issues in cloud computing. In particular section 5, some cryptographic techniques in cloud computing environment are mentioned. Section 6 presents the overview of homomorphic encryption in cloud computing medium. In section 7, future research directions are given. Finally conclusions are presented in the section 8.

2. PRELIMINARIES

Some basic preliminaries required in this paper are described as below:-
Cryptography($crypto = secret + graphy = writing$) is a process of storing and transmitting data in an unoriginal form, so that only authorized person(for whom it was meant), can access and process it. Cipher is a mechanism of writing secret messages, where original/plaintext is transformed into an unoriginal/cipher text. Converting the original text into unoriginal text is reffered as encryption and mechanism of decryption is vice versa.

2.1 Symmetric key cryptography

It is also known as private key cryptography, is kind of encryption mechanism in which sender as well as receiver, both entities use to share the similar key. These are realized as either Block ciphers phenomenon or as Stream ciphers phenomenon. Block cipher takes entire blocks of plain text as the form of input, whereas individual characters are taken as input, in case of stream ciphers. Data Encryption Standard(DES) as well as Advanced Encryption Standard(AES), both are the well known block cipher design paradigms that have been selected as cryptographic benchmarks.
Private key cryptosystems utilizes the one same key for both mechanisms - encryption as well as decryption of the information, though a particular message or collection of several messages may posses different keys. A notable limitation of symmetric ciphers is that, the key management process becomes intensely necessary to use them securely. Key management consists of creation, distribution and refreshing of the secret keys, involved in the communication.

2.2 Asymmetric key cryptography

It is also called as Public key cryptography. In this mechanism, sender encrypts the message using public key of receiver and further receiver decrypts the information utilizing his own private secret key. Public key cryptography technique can be utilized for implementing the various Digital signature methods. RSA mechanism and DSA are two most prominent digital sign. mechanisms. Public-key methods are mostly based on the very large computational complexity of *"hard problems"*. Eg. the hardness of well known RSA algorithm is based on hard problem of integer factorization, while hardness of DiffieHellman and DSA algorithms is based on discrete logarithm problem. Recent times, elliptic curve cryptography(ECC) has been evolved, in which security is totally based on the number theoretic computationally hard problems involving elliptic curves. Paillier cryptosystem, invented by Pascal Paillier in 1999, is also an example of probabilistic behavioural asymmetric mechanism for public key cryptography.

2.3 Ring Homomorphism

Let, P and Q are rings.
A function $f : P \to Q$ will be satisfying ring homomorphism, if $\forall h_1, h_2 \in P$.
—$f(h_1 + h_2) = f(h_1) + f(h_2)$
—$f(h_1 * h_2) = f(h_1) * f(h_2)$
—$f(1_P) = 1_Q$

3. RELATED WORK

Here, in this section, we will review some of existing methods which have been proposed in past years. Maha Tebba et al. [2] inspected the core application scenarios of different Homomorphic Encryption cryptosystems eg: (RSA, Paillier, El Gamal, Gentry etc.) on a Cloud Computing environment. Further, comparision is being performed based on main four specialities - "Homomorphic Encryption type", "Privacy of data", "Security applied to" and "the keys used". Reem Alattas et al. [3] introduced the application of Algebraic Homomorphic Encryption mechanism, based on Fermat's Little Theorem on cloud computing for better security. To fix the challenging problem of data privacy along with confidentiality in the cloud, Fully Homomorphic Encryption(FHE) mechanism is an explication, where the encrypted information can be handled, and it returns the results in encrypted manner. In spite of, fully homomorphic encryption mechanism runs in comparatively slower mode hence, the faster fully homomorphic encryption mechanisms are very much needed. Gentry's proposed encryption mechanism is fully homomorphic but having impediment of slower performance. Lot of various mechanisms have been suggested in recent years to remarkably speed up the performance achievement of fully homomorphic encryption schemes. Parallel processing of given information is one compelling way of executing this. Parallel processing for Gentrys encryption was presented by Ryan Hayward et al. [4] in their paper and also tested in a private cloud computing domain. Frederik A. et al. [5] presented the simplified and structured wide definitions in the homomorphic encryption discipline, and interrogated whether presently existing applications need homomorphic encryption thought as a explanable solution to their problems, both in theoritical along with practical approaches. In 1978, Rivest et al. [6] presented the first homomorphism technique. Rivest et al. [7] presented the RSA, which gives a multiplicative homomorphism. Yao [8] proposed partial homomorphic encryption scheme. [9] [10] [11] [12] presented the work done in past years in homomorphic encryption mechanism. Craig Gentry [13] has introduced fully homomorphic encryption in his thesis. It was a significant breakthrough in cryptographic mechanisms. Homomorphic encryption on smaller size cipher text is proposed by [14]. Van Dijk et al. [15] proposed implementations for arithmetic operations over integers. [16] suggest a faster improvement to Gentry's model. Ramaiah et al. [17] has proposed an Efficient public key homomorphic encryption over integer plaintexts.

4. SECURITY ISSUES IN CLOUD COMPUTING

We have pointed out some security challenges in cloud computing, which are as below:-

—When customers are outsourcing/transfering their private data to any third party, then there is much responsibility of both security

and compliance. Therefore, it is necessary that customers should fully faith in their cloud service provider.

—Cloud computing consists of several technologies eg. databases, network structure, operating systems, virtualization scenario, resources and processes scheduling, transaction management, load balancing factor, memory management etc. So, due to use of these wide variety of technologies, a small security weakness in any one of these technologies may knock down the complete system.

5. CRYPTOGRAPHIC TECHNIQUES IN CLOUD COMPUTING

Some of the significant and mostly used public key cryptosystems are presented as below:-

5.1 RSA Cryptosystem[Multiplicative Homomorphic Encryption]

RSA cryptosystem satisfies multiplicative homomorphic property or in other words, RSA cryptosystem is an example of partial homomorphic encryption mechanism.
Suppose, CT_1 and CT_2 are two ciphertexts. msg_1 and msg_2 are the plain texts.
$CT_1 = msg_1^e \bmod n$
$CT_2 = msg_2^e \bmod n$
where, e: is public key exponent; $n = p.q$: is product of two large prime numbers p and q.
$CT_1 \,.\, CT_2 = msg_1^e \,.\, msg_2^e \bmod n$
So, multiplicative homomorphic property is: $(msg_1.msg_2)^e \bmod n$.
So, if the encryption of a message msg is given by - $E(msg) = msg^e \bmod n$
Homomorphic property is then -
$E(msg_1) \,.\, E(msg_2) = msg_1^e \,.\, msg_2^e \bmod n = (msg_1.msg_2)^e \bmod n = E(msg_1.msg_2)$

5.2 Paillier Cryptosystem[Additive Homomorphic Encryption]

In Paillier Cryptosystem, encryption function is additively homomorphic -
Given, $E(m_1)$ and $E(m_2)$,
where, m_1 and m_2 are plain texts. The computation of cipher text (encryption method) in Paillier Cryptosystem is as -
$c = g^m \,.\, r^n \bmod n^2$
we can not get $E(m_1.m_2)$. We can only get $E(m_1 + m_2)$

6. HOMOMORPHIC ENCRYPTION IN CLOUD COMPUTING

The general mechanism of Homomorphic Encryption in the cloud computing is presented as figure below:-

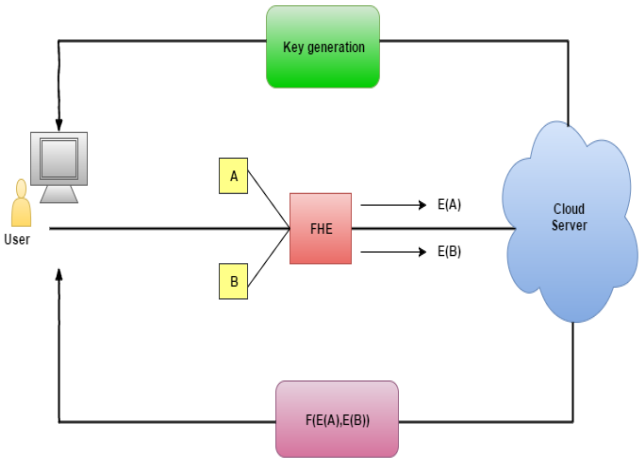

Fig.1

Homomorphic Encryption mechanism is an special form of encryption that allows the various computations to be performed on non-plain text itself, thus producing the result in an encrypted form which, when decrypted, matches the result of operations performed on the plain text. There are numerous partially homomorphic cryptosystems as well as fully homomorphic cryptosystems. Fully homomorphic encryption(FHE) is considered to be more secure than partially homomorphic encryption.

6.1 Partially Homomorphic Encryption

A cryptosystem is thought as partially homomorphic, if it manifests either additive or multiplicative homomorphism property, but not both. Some examples are: RSA(based on multiplicative homomorphism), Paillier(based on additive homomorphism), ElGamal(based on multiplicative homomorphism).

6.2 Fully Homomorphic Encryption

A cryptosystem is thought as fully homomorphic, if it manifests both additive and multiplicative homomorphism property. The first(and currently only) before mentioned system is a lattice-based cryptosystem which was in 2010, proposed and developed by Craig Gentry.[13] FHE is considered as far more powerful and a great way to secure the outsourced data in an efficient manner.
Gentry's proposed scheme has three significant components:-

(1) A somewhat homomorphic encryption scheme(SWHES)
(2) A bootstrappable encryption scheme(BES)
(3) A combination of above two components

This scheme was possesing the capability of performing the homomorphic computaion on the polynomials with low degree. The research community is trying to figure out possible scenarios on malleability property for homomorphic encryption mechanisms.
Consider, the fully homomorphic scheme, where -
the encryptions on the plain text M_1 and M_2 can be $Encr(M_1)$ and $Encr(M_2)$. Now, since FHE acheives both additive and multiplicative properties, so both $Encr(M_1+M_2)$ and $Encr(M_1*M_2)$ can be computed in a secure and efficient manner.
Various researchers are also trying to find out the practical scenarios and to finding ways for tackle big issues of fully homomorphic encryption, also to think about - what tools we have with

us and what amount of computations can be perfomed with existing power. Though the evolutionary step of Gentry's proposal left whole research community with the huge number of significantly open questions and practical issues.

7. RESEARCH DIRECTIONS

The problem identification and further future research directions are presented as follows:-

7.1 Problem Identification

Today, data privacy and security becomes an essential part of various cloud based applications, multiparty computation scenarios etc. Homomorphic Encryption is a recently evolved technique, which solves the problems of confidentiality and privacy of the stored data. Still, there exist many complications to practically apply these Homomorphic encryption mechanisms. The core problems, which we have identified in the present existing system are as below -

—For some cryptographic algorithms, after applying the encryption algorithm on plain text data, the size of cipher text is more as compare to original plain text. The reason may be due to some padding procedure. So, to perform computations on this encrypted data, will take more computational time.

—Cipher text may comprise some noise elements in it that becomes relatively massive with the subsequent homomorphic multiplication computations, and only those cipher texts, whose noise estimation remains within a certain threshold value, can be decrypted accurately.

7.2 Future Research Directions

It has been always the dream of researchers to go in cloud environment. Security and privacy of data, specially in clouds, has become most essential part for various applications in present scenario.
Our future research directions will be as follows:-

—The development of the fully homomorphic encryption by Gentry in 2010, is a revolutionary advancement in cryptography, greatly broadening the scope of the computations which can be correlate to process the encrypted data homomorphically. With fully homomorphic encryption the cloud entity can operate the computational procedures on behalf of the user and return only the encrypted result. However, it is not practical from a performance point of view.
Our further research directions are to implement the fully homomorphic encryption scheme and compare it with different existing cryptographic algorithms, which are following partial homomorphic properties in their operations.

—Another limitation is - FHE does not support for multiple users. The practical applications which involve the running of enormously large and complex algorithmic computations homomorphically, fully homomorphic encryption (FHE), have a massive computational overhead, which makes the intermediate complex functional computations impractical. We will see the possible ways to solve this problem.

8. CONCLUSION

In any outsourced computing or third party computations, there exist a powerful thrust to provide security at infrastructure - network level, host level, application level and data. Homomorphic Encryption technique enables computing on encrypted data itself inside the cloud. It means one can perform the operations of this data without converting into the plain text. In 2010, Gentry proposed the Fully Homomorphic Encryption mechanism, which was a great breakthrough in cryptography. This paper presents an overview of Homomorphic Encryption and state of the art in this area. The problem identification in this domain is also discussed in this paper and have given our future research directions.

9. REFERENCES

[1] Manish M Poteya, Dr C A Dhoteb, Mr Deepak H Sharmac, "Homomorphic Encryption for Security of Cloud Data, Computing and Virtualization, 7th International Conference on Communication, Procedia Computer Science volume 79 pages 175 181 (2016).

[2] Maha TEBAA, Said EL HAJII, "Secure Cloud Computing through Homomorphic Encryption", International Journal of Advancements in Computing Technology(IJACT), Volume-5, Number-16, December 2013.

[3] Reem Alattas, Khaled Elleithy, "Cloud Computing Algebra Homomorphic Encryption Scheme Based on Fermat's Little Theorem", The American Society of Engineering Education, ASEE 2013, Northfield, VT, USA, 09 December 2016.

[4] Ryan Hayward, Chia-Chu Chiangb, "Parallelizing fully homomorphic encryption for a cloud environment", Journal of Applied Research and Technology 13 (2015), 245-252.

[5] Frederik Armknecht et. al., "A Guide to Fully Homomorphic Encryption", iacr, 2015.

[6] Rivest, Ronald L., Len Adleman, Michael L. Dertouzos, "On data banks and privacy homomorphisms.", Foundations of secure computation 4.11 (1978): 169-180.

[7] Rivest, Ronald L., Adi Shamir and Len Adleman, "A method for obtaining digital signatures and public-key cryptosystems.", Communications of the ACM 21.2 (1978): 120- 126.

[8] A. C. Yao, "Protocols for secure computations" (extended abstract). In 23rd Annual Symposium on Foundations of Computer Science (FOCS '82), pages 160-164. IEEE, 1982.

[9] Goldwasser, Shafi and Silvio Micali, "Probabilistic encryption", Journal of computer and system sciences 28.2 (1984): 270-299.

[10] ElGamal, Taher, "A public key cryptosystem and a signature scheme based on discrete logarithms", Advances in cryptology. Springer Berlin Heidelberg, 1985.

[11] Paillier, Pascal, "Public-key cryptosystems based on composite degree residuosity classes", Advances Heidelberg, 1999, in cryptology-EUROCRYPT99.

[12] Fontaine, Caroline, Fabien Galand, "A survey of homomorphic encryption for nonspecialists", EURASIP Journal on Information Security (2007).

[13] Craig G., "Fully homomorphic encryption using ideal lattices", STOC. Vol. 9. 2009.

[14] Smart, Nigel P., Frederik Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes", Public Key Cryptography-PKC, Springer Berlin Heidelberg, 2010, P-(420-443).

[15] Van Dijk, Marten, "Fully homomorphic encryption over the integers", Advances in cryptology-EUROCRYPT 2010. Springer Berlin Heidelberg, 2010. 24-43.

[16] Stehle, Damien, Ron Steinfeld, "Faster fully homomorphic encryption", Advances in Cryptology-ASIACRYPT 2010. Springer Berlin Heidelberg, 2010. 377-394.

[17] Ramaiah, Y. Govinda and G. Vijaya Kumari, "Efficient public key homomorphic encryption over integer plaintexts", Information Security and Intelligence Control (ISIC), 2012 International Conference on. IEEE, 2012.

[18] Shafi Goldwasser, Yael Kalai, "Introduction to Homomorphic Encryption", (6.889), New Developments in Cryptography, MIT CSAIL: February 1, 2011.