Survey on Homomorphic Encryption

[1]Y Mohamed Sirajudeen, [2]R Anitha
[1, 2]Department of Computer Science and Engineering,
Sri Venkateswara College of Engineering, Tamilnadu, India.
ducksirajsmilz@gmail.com, ranitha@svce.ac.in

***Abstract*–Over the last decade, Cloud computing is the most effective technology in the field of computing. The vast development in cloud has several reasons like, on-demand service, elasticity, flexibility of workspace, etc. It also reduces the capital investment for small scale companies by allowing the users to work with cloud server instead of their own. Cloud storage is a services to users to store, manage and maintain their data in the cloud environment.Cloud storage servers are geographically located in a distributed manner, which increase the vulnerability of the user data. Despite the technical and business improvement, the security remains as a threat to the users. In some scenarios, even the CSP's may not be trustworthy. While decrypting the data in cloud for processing, CSP's has a possibility to look into the user's data. To avoid such problems, research came with an idea of Homomorphic Encryption (HE) technique, in which it allows to perform operations like addition and multiplication on the encrypted data. With this effective encryption scheme, the cost for decrypting the data on every operation you perform and cost of re-encrypting can be eliminated. This paper intends to presenta historical perspective and overview of homomorphicencryption algorithms which is been developed by the researchers in the recent years.**

***Keywords* – Homomorphic Encryption, Data Encryption, Homomorphism.**

I. INTRODUCTION

Homomorphic is a word derived from ancient Greek terminology. *Homos,* is a Grecian words which means "same" and *Morphe,* means "shape", put together it is coined as homomorphism and used in modern algebra to represent a map between two algebraic structure (groups) thatpreserves the operations of the structures (i.e.) a function $f : G \rightarrow H$ between two groups is homomorphic when, $fun(xy) = fun(x)\,.\,fun(y)\ \forall\, x, y\, \in G$. Here, fun is a function, which takes the input from a group and performs operation (addition and multiplication) to map with the other set.

The word "Homomorphic encryption" became popular in the field of cryptography as the third party data storage become popular. Rivest was the first researcher to give an idea to implement homomorphism in encryption. In 1978, after a little while developing a successful RSA security algorithm,Rivest introduced the idea of incorporating the homomorphic property in the encryption technique on his research paper, "On Data Banks and Privacy Homomorphism", which focused on a sensitive bank-loan customer data [1]. Later in the same year Rivest et al, proposed few more homomorphic encryption algorithms, which included the "Partially Homomorphic RSA algorithm". But the practical implementationswere not done as it was vulnerable to various attacks.

Due to the lack of practicality and hardware issues, the research on homomorphic encryption was little lagging in 1980's and 1990's. Later in 2000s, as cloud computing started emerging and drastic development in using the third party storage service madethe researcher to focus back on Homomorphic Encryption with higher attention[7,8]. The researcher named Craig Gentry from Stanford University developed the first practical "Fully Homomorphic Encryption" in 2009. But originally it was "Somewhat Homomorphic Encryption", which is limited to perform some operation on the cipher text. The homomorphic operation on Gentry's work was limited because the noise is attached with cipher text and increases in each operation. After Gentry's research, there were several concrete developments. This paper presents a detailed review on the improvements of Homomorphic Encryption (HE) both in the theory and implementation.

Figure.1. represent the simple Homomorphic Encryption on cloud environment. Initially, the user encrypts the data using a secure encryption technique which supports homomorphic operations on cipher text *viz.,* the user encrypts the data with RSA algorithm, which supports multiplicative homomorphic operation. Later the encrypted data is send to Cloud Service Provider (CSP) to store in the Cloud Storage Server [12, 13].

In step 4, the Cloud user requests the CSP to perform a particular operation ($func()$) on top of the cipher text. In step 5, the CSP perform the user requested function on the cipher text without disturbing the original content. Later the encrypted results were stored back in the cloud itself. While the cloud user decrypts the encrypted data with the private key, the modified result will be displayed[2]. By this advanced encryption technique, it is possible to perform operation on the server side without needing the user's private key.

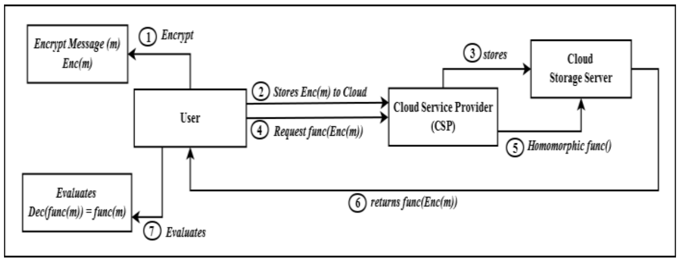

Fig.1. Simple workflow of Homomorphic Encryption in Cloud Environment

Figure In general the Homomorphic encryption can be classified in to three categories[13]. (1) Partial Homomorphic Encryption (PHE), which allows one type of operation without any limitation (i.e.) unlimited number of time a single function can be performed. (2) Somewhat Homomorphic Encryption (SWHE), which allows to perform different operation with a limited number of times. (3) Fully Homomorphic Encryption (FHE) allows the CSP to perform unlimited number operation like addition, multiplication, sortingfor unlimited number of times[14]. The rest of this paper describes all the three categories of the Homomorphic Encryption in detail.

II. HOMOMORPHIC ENCRYPTION TECHNIQUES

A Homomorphic encryption(HE) is a technique, which allows the users to carry out the computational operations on top of the cipher text without modifying the data format or compromising the securityof the cloud system[1, 11]. This homomorphic encryption has two major operations or functionalities. (1) Multiplicative and (2) Additive Homomorphic Encryption.

Definition: 1: Multiplicative Homomorphic Operations:
An encryption scheme is called as Multiplicative Homomorphic Encryption, if,

$$E(a * b) = E(a) * E(b) \quad \forall\, a, b \in n$$

Where 'E' represents encryption and 'n' representsa set of all possible messages.

Definition: 2: Additive Homomorphic Operations:

An encryption scheme is called as Additive Homomorphic Encryption, if,

$$E(a + b) = E(a) + E(b) \quad \forall\, a, b \in n$$

Where 'E' is encryption and 'n' is set of all possible messages.

To developa practical Partial Homomorphic Encryption (PHE), additive or multiplicative[14,15] functions are the only option to perform homomorphic operation on top of the encrypted data because any Boolean circuit can be designed only through XOR and NAND gate, where XOR performs the addition and NAND performs the multiplication.As far as the PHE is concerned, the various encryption scheme developed so far are based on additive or multiplicative operation only.

III. PARTIAL HOMOMORPHIC ENCRYPTION (PHE)

PHE was the first homomorphic encryption which was introduce by Rivest in 1976. But it was not named as Partial Homomorphic Encryption rather it was called a Privacy Homomorphism [3]. PHE allows a single operation to perform unlimited times on the encrypted data *viz.;*itallows performing addition or multiplication operation 'n' number of times.

There are several popular PHE algorithms. This section will describe about those encryption technique and its advantages.

A. RSA Algorithm (1976)

RSA algorithm was developed in 1976 by Rivest, Shamir and Adleman.It is one of the first globally acceptable Asymmetric key encryption.RSA algorithm [2] was an idea attributed from Diffie-Hellman encryption scheme which was published on the same year.The homomorphic property of the RSA was later published in 1978. It follows three general steps as, (i) Key generation(ii) Encryption (iii) Decryption. RSA follows a multiplicative homomorphic operation on the encrypted data.

RSA Key Generation:

Chose a large possible prime number, say $p$and q. Here, $n = p * q$; $\Phi = (\mathrm{p} - 1)$ and $(\mathrm{q} - 1)$. Then find, $\lambda(n) = LCM\left((p-1)(q-1)\right)$.Chose any co-prime '$p$' number between 1 and $\lambda(n)$ (i.e.)$1 < e < \lambda(n)$chose'd' by modular inverse multiplicative $d = e \ mod \ \lambda(n)$.

RSA Encryption:

The public keys are 'n' and 'e'. Encryption, $c(m) =$ m $mod\ (n)$, *'m' is message* $m < n$.

RSA Decryption:

By using private key 'd', decryption can be performed by, $m =$ c^d $mod\ (n)$; 'c' is cipher text.

Homomorphic Property:

Consider two messages $m_1$and m_2,m^e $mod\ (n) =$ $\mathrm{m_1}$^e $mod\ (n) *$ $\mathrm{m_2}$^e $mod\ (n)$;

m^e $mod\ (n) =$ $\mathrm{m_1}$^e $*$ $\mathrm{m_2}$^e $(mod\ n)$
$= (m_1 * m_2)$^e $(mod\ n\ )$

From this, RSA supports the multiplicative homomorphic encryption technique. This scheme doesn't support additive homomorphic encryption.

Attacks on RSA Algorithm:

RSA algorithm has number of security vulnerability in the plain text[9]. While choosing a lower prime value for 'e' it is possible to identify the decryption key. And also RSA algorithm is vulnerable against chosen cipher text attack (CCA) and adaptive chosen cipher text attack (CCA2). So performing homomorphic on RSA algorithm is not secure. In 1978, Rivest et al. accepted the vulnerability and the lack of security in using homomorphism in RSA algorithm.

B. Goldwasser - Micali Cryptosystem (1982)

Goldwasser - Micali cryptosystem was developed in 1982, which was the first probabilistic based PKE system. It is semantically secure under the assumptions of standard cryptographic. But the GM encryption technique is not efficient as the cipher texts are very largecompare to the plain text. GM supports the additive homomorphic encryption but it doesn't support multiplicative operations.

GM Key Generation*:*

Chose a large possible prime value, say p and q. $n = p * q$; where p, q should be random and independent. Find 'a' where,$a_p^{(p-1)/2} = -1(mod\ p)$; $a_q^{(q-1)/2} = -1(mod\ q)$a ;Where '$a$' and '$n$' are public keys and '$p$' and '$q$' are private keys.

GM Encryption*:*

Encode the message as string of bits$(m_1, m_2 \dots m_n)$. For every bit m_i senders creates a random bit b_i where, $\gcd(b_i, N) = 1$; $c_i = b_i^2 . a^{m_i} (Mod\ N)$; Where $c_i$is the cipher text of messagem_i

GM Decryption:

Cipher Text $(c_1, c_2 \dots c_n)$ can be decrypted by the private keys factorization of 'p' and 'q'. Receiver determines whether the value c_i is quadratic residue. If it is quadratic residue, $m_i = 0\ or\ m_i = 1$

Homomorphic Property:

The GM encryption scheme has additive homomorphic property (i.e.) it uses XOR operation on the encrypted data.Consider the message bits m_1 and $m_2$and their cipher bits are$c_1$andc_2,

$c_1 . c_2 (Mod\ N) = Dec(\ m_1 \text{XOR} m_2)$where$Dec$ is Decryption.

C.Elgamal Algorithm (1985)

*Elg*amal Encryption method was developed by Taher Elgamal in 1985. The basic ideology of Elgamal algorithm is DH Key exchange algorithm. It supports multiplicative homomorphic operation. The below section describes the working model of Elgamal Encryption technique.

Elgamal Key Generation

Sender creates an efficient cyclic group $'G'$ of order $'n'$ and chooses a random value $x \in \{1,2 \dots q-1\}$. Sender computes the value of $y = g^x$; Where $y,\ q,\ g$ as public key and $'x'$ as public key.

Elgamal Encryption

Sender computes $c_1 = g^r$and computes $s = y^r$ where r is random number which belongs to $\{1,2 \dots q-1\}$ and converts the message m into $m' \in G$ and computes$c_2 = m'.s$; where s is the shared secret key.

Elgamal Decryption

To decrypt $c_1$and$c_2$with the private keys, find the shared secret key $s = c_1^y$; Where $'y'$ is the secret key. Then decrypt by,$c_2.s^{-1} = mg^{sx} . g^{-xy} = m$where$'m'$ is message.

Homomorphic Property

Elgamal supports a multiplicative homomorphic encryption. Consider two encryptions such as,(c_{11}, c_{12}) and (c_{21}, c_{22}) where$(c_{11}, c_{12}) = g^{r1}, m_1 . y^{r1}$ and$(c_{21}, c_{22}) = g^{r2}, m_2 . y^{r2}$, here $r_1$and r_2 are randomly chosen.$(C_{11}, C_{12})(C_{21}, C_{22}) = (C_{11}.C_{21}\ , C_{12}.C_{22})$is equal to $(g^{r1+r2}, (m_1.m_2)y_1^{r1+r2})$. Elgamal doesn't support additive homomorphism.

D. Pallier Cryptosystem (1999)

Pallier Cryptosystem [5] was developed by Pascal Pallier in the year 1999. It also supportsadditive homomorphic operations.

Pallier Key Generation

Chose two numbers, say p, q which should be independent to one another.Gcd$\big(pq, (p-1)(q-1)\big) = 1. n = p.q$; $\lambda = lcm(p-1, q-1)$. Choose an integer g by checkinggcd$(n, L\big(g^\lambda\ mod\ n^2\big)^{-1} = 1$;in which $L(x) = \frac{x-2}{n}$; Here, public keys and private keys are(n, g) and (p, q) respectively.

Pallier Encryption

A number r is chosen randomly. Encrypt by, $c = g^m . r^n\ mod\ n^2$ where $0 < r < n$and $0 < m < n$

Pallier Decryption

To compute the plain text from cipher text $c, m = L\big(C^\lambda\ mod\ n^2\big)\mu\ mod\ n$Decryption is exponential modulo of n^2.

Homomorphic Property

Pallier Cryptosystem supports additive homomorphic property and also it was first proposed to implemente-voting system. But Pallier Cryptosystem does not support multiplicative homomorphism.

TABLE I
VARIOUS PH ENCRYPTIONS AND ITS OPERATIONS

Algorithm	**Homomorphic**	
	Additive	**Multiplicative**
Diffie - Hellman Key Exchange	×	×
GM Algorithm	✓	×
RSA Algorithm	×	✓
Elgamal Algorithm	✓	✓
Benoloh Algorithm	✓	×
Okamoto and Uchiyama Algorithm	✓	×
Elliptic curve Cryptography	✓	×
Pallier	✓	×

These are all the various Partial Homomorphic Encryption (PHE) techniques. Table. I. represents the homomorphic property of PHE algorithms. The next section will describe about the important Somewhat Homomorphic encryptions.

IV. SOMEWHAT HOMOMORPHIC ENCRYPTION

There are several Somewhat Homomorphic Encryption (SWHE) are globally accepted. One of the popular methods is BGN encryption scheme. This scheme was developed by Beneh-Goh-Nissim at Stanford University, which is also the first practical Somewhat Homomorphic Encryption.

Before this encryption scheme, all the operations are limited to addition or multiplication only. This scheme gave a major advancement towards the practical Fully Homomorphic Encryption.

A. BGN Algorithm

BGN algorithm computes a 2-DNF value over the encrypted cipher text and so it supports arbitrary number of addition and multiplication without changing the cipher text size constant. The hardness of the algorithm depends on the subgroup decision problem.

BGN Key Generation

The public key are (n, G, G_1, e, g, h), Such that, $e: G * G \rightarrow G_1$ where G_1andG_2 are groups of order n. Here, $n = q_1.q_2$ and $h = u^{q_2}$. h is kept secret.

BGN Encryption

A random number r is created, where $0 < r < n - 1$ and encryption can be done by the following function,$c = g^m.h^r \; mod \; n$.

BGN Decryption

To decrypt, find the c' and g', where $c' = (g^m h^r)^{q_1}$ and $g' = g^{q_1}$ and decryption can be done by, $m = \log_{g'} c'$Where m is the plain text.

Homomorphic Property

BGN follows both the additive and multiplicative property of the homomorphism.$c_1.c_2 = (g^{m_1}.h^{r_1} \; mod \; n).(g^{m_2}.h^{r_2} \; mod \; n) \; and \; c_1.c_2 = g^{m_1+m_2}.h^r$This makes the BGN encryption scheme as additive homomorphic encryption.

The homomorphic multiplication of $M_1$and m_2 using the Cipher text $c_1$and c_2.

$$e(c_1.c_2)h_1^r = e(g^{m_1}.h^{r_1}, g^{m_2}.h^{r_2} \; mod \; n)h_1^r$$

$$e(c_1.c_2)h_1^r = g_1^{m_1 m_2} h_1^{m_1 r_2 + r_2 m_1 + \alpha q_2 r_1 r_2 + r} = g_1^{m_1 m_2} h_1^{r'}$$

This makes the BGN encryption scheme as multiplicative homomorphic encryption.The major disadvantage of BGN scheme is the size of the ciphertext increase as operation increases, though the level of noise in the cipher is also getting increase in every iteration. So, it was very difficult to implement BGN algorithm in real time. To overcome this issue, Fully Homomorphic Encryption is proposed. The upcoming section will describe the Gentry FHE Scheme in detail.

V. FULLY HOMOMORPHIC ENCRYPTION

Fully Homomorphic Encryption (FHE) allows the CSP to perform unlimited number operation like addition, multiplication, sorting for unlimited number of times [11, 13]. The first practical FHE was proposed by Craig Gentry in his Ph.D. thesis in the year 2009. Gentry's proposed scheme has designed a general framework for FHE scheme. The algorithm proposed by Gentry is based on the ideal lattice property. A lattice L is a linear combination of independent vector. Gentry algorithm has three major steps (1) Creating a Somewhat Homomorphic Encryption using ideal lattice (2) Create the squashing and Decryption circuit (3) Bootstrapping

Gentry's Key Generation:

For the given ring R and the basis B_i of ideal I, $IdealGen(R; BI)$ algorithm generates the pair of B_j^{sk}, B_j^{pk}. A $Samp()$ algorithm is also used in key generation to sample from the given coset of the ideal, where coset is obtained by shifting an ideal by a certain amount. Public keys are, $R, B_i, B_j^{pk}, Samp$ and the secret key is B_j^{sk}.

Gentry's Encryption:

Two random vectors are created, r and g,$c = m + r.B_I + g.B_j^{pk}$where B_i is basis of the ideal lattice L. Here, $m + r.B_i$ is called "noise" parameter.

Gentry's Decryption:

By using the secret key $B_j^{sk}$the decryption is done,$m = c - B_j^{sk}.\left[\left(B_j^{sk}\right)^{-1}.c\right] \; mod \; B_i$. It returns the nearest integer of the coefficient vector.

VI. CONCLUSION

In order to increase the level of the security in the cloud services, researchers are developing several Public Key Encryptions (PKE). In addition to that, the homomorphic encryption scheme and its important has opened the eye for performing computation on top of the cipher text in the cloud environment. In this regard this paper detailed about implementing the various homomorphic encryption in cloud computing in a theoretical perspective. In future, an effective homomorphic encryption scheme will be developed with reduced overhead.

ACKNOWLEDGEMENT

This initial survey was taken in account to create an efficient homomorphic Cramer Shoup Cryptosystem. The work of this paper is financially supported by the Science and Engineering Research Board (SERB), Government of India. The Grant Number is ECR/2016/000546.

REFERENCES

[1] Ronald L. Rivest, Leonard, Adleman, Micheal," On Data Banks and Privacy Homomorphism", Academic Press, pp. 169-1, 1978.

[2] R. L Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, Vol. 21, No. 2, pp. 120-126, 1978.

[3] Taher Elgamal, "A public key Cryptosystem and a signature scheme based on discrete logarithms", Advances in Cryptology-CRYPTO '84, pp. 10-18, 1985.

[4] Victor S. Miller, "Use of Elliptic Curves in Cryptography", Advances in Cryptology-CRYPTO '85, pp. 418 – 426, 1986.

[5] Pascal Paillier, "Public Key Cryptosystems Based on Composite Degree Residuosity Classes", EUROCRYPT,Vol. 1592, Springer Verilag pp. 223–238, 1999.

[6] Julien Bringe, Herve Chabanne, Malika Izabachene" An application of the Goldwasser Micali Cryptosystem to Biometric Authentication", Information Security and Privacy, Australian Conference of Information Security and Privacy, pp. 19-106, 2007

[7] Zhang Shaomin, Li Xiaoqiang, Wang Baoyi ,"Study on the Protection Method of Data Privacy Based on Cloud Storage", International Journal of Information and Computer Science , Vol.1 , No. 2, pp. 46-51,2012.

[8] Sharma, Divya, Preeti Vaidya, and Oves Khan. "Survey on Security Issues in Cloud Computing", International Journal of Innovative Technology and Exploring Engineering, Vol. 3, No.1, pp.83-87, 2013.

[9] R Anitha, Saswati Mukherjee, “Metadata driven Efficient CRE based Cipher Key Generation and Distribution in Cloud Security”, International Journal of Security and Its Applications, Vol. 8,No. 3, pp. 377-392, 2014.

[10] R Anitha, P. Pratheepan, P. Yogesh, Saswati Mukherjee, “Data Storage Security in Cloud using Meta data”, International Conference on Machine Learning and Computer Science, pp.,26-30, 2014

[11] R. Kanagavalli, Vagdevi S, “ A Survey of Homomorphic Encryption Schemes in Cloud Data Storage”, International Journal of Recent Development in Engineering and Technology, Vol. 3, Issue 1, pp. 71-75, 2014.

[12] Malina, Lukas, and Jan Hajny. "Efficient Security Solution for Privacy - Preserving Cloud Services“, Journal of Applied Research and Technology, Vol. 13, pp.20-31, 2015.

[13] Kamal Benzekki, Abdeslam El Fergougui, Abdelbaki, “ A secure Cloud Computing Architecture Using Homomorphic Encryption”, International Journal of Advanced Science and Applications, Vol. 7, No. 2, pp. 293-298, 2016

[14] Abbas Acar, Hidayet Aksu, Selcuk Uluagac, Mauro Conti, “A survey on Homomorphic Encryption Schemes : Theory and Implementation”, ACM Computing Survey, 2017.

[15] Daniel Okunbor, Chekad Sarami, “Homomorphic Encryption: A Survey”, Review of Business and Technology Research, Vol. 14, No. 1,pp. 64 - 69, 2017.