Detailed Study of Homomorphic Cryptosystem: A Review

Samridhi Singh [1], Hemani[2]

[1,2]*PG Student, Department of Information Technology,*
G.B.P.U.A&T, Pantnagar, Uttrakhand, India

***Abstract*-** Homomorphic encryption is encryption with the property of performing a function on two values separately and then encrypting the result yields the same final value as first and encrypting two values separately and then applying the function to the result. In this paper, there is a description of the various homomorphic encryption scheme and properties also with the levels of homomorphic encryption and application where it can be used to secure the data or text.

***Keywords*-** Homomorphic encryption, Pallier algorithm, RSA algorithm, Homomorphism, Bootstrapping.

I. INTRODUCTION

Security is important to transmit data through an unsecured channel and for storage. Steganography and cryptography are methods to secure the transmission of an image through an unsecured channel. Where cryptography is a method of transmitting and storing data in a form that unauthorized person cannot read the data and Steganography takes a step farther than cryptography by hiding an encrypted message so that no other can suspect its exist. A homomorphic cryptosystem is an ability to perform addition and multiplication operation on encrypted data without revealing any information about original data.

II. BASIC OF HOMOMORPHIC ENCRYPTION

Homomorphic encryption is a conversion of data in cipher text by performing computation on it and generating an encrypted result and when it is decrypted matches the result of an operation performed on original text.

In mathematics, homomorphic describes the transformation of one data set into another while preserving the relationship between elements in both sets. The term is derived from the Greek word for “same structure”, because the data in homomorphic encryption scheme retain same structure, whether they are performed on encrypted or encrypted data.

Homomorphic encryption plays an important role in cloud computing, allows a company to store encrypted data in a public cloud.

An example of Homomorphic encryption might work on cloud computing.

- Company ABC has an important data set that consists of the number 10 and 5.To encrypt the data set, Company ABC multiplies each element in the set by 2, creating a new set whose members are 20 and 10.
- Company ABC sends the encrypted data set to the cloud for safe storage. A few months later, the government contacts Company ABC and requests the sum of data set elements
- Company ABC is very busy, so it asks a cloud provider to perform the operation. The cloud provider, who only has the access to the encrypted data set, finds the sum of 20 +10 and return answer 30.
- Company ABC decrypts the cloud provider reply and provides the decrypted answer 15 to the government.

A. Function of Homomorphic encryption:

Homomorphic Encryption H is a set of four functions-

H= {Key Generation, Encryption, Decryption, Evaluation}

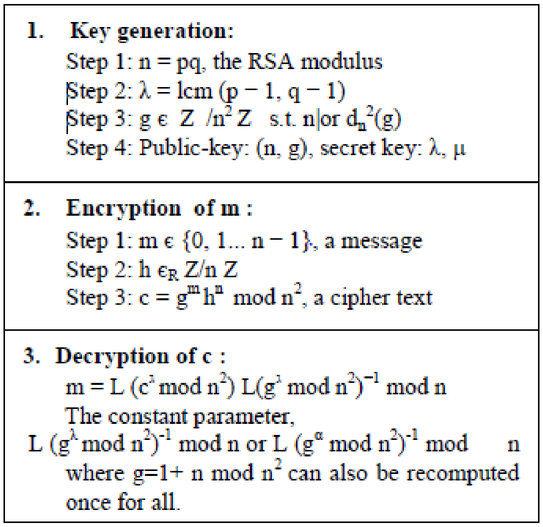
1. Key generation: The client will generate the pair of keys, public key and secret key for encryption of plaintext.
2. Encryption: Using secret key client encrypt the plain text and generate encrypted plain text and along with public key this cipher text will be sent to the server.
3. Evaluation: Server has a function for doing an evaluation of cipher text and performs this as per the required function using the public key.
4. Decryption: Generated evaluated plain text will be decrypted by a client using its secret key and it gets the original result.

B. Properties of Homomorphic encryption

Homomorphic Encryption has two properties:

1. Additive homomorphic encryption: Encryption function described by-

$$Ek\ (PT \oplus PT2) = Ek\ (PT1) \oplus Ek\ (PT2)$$



1. Key generation:

Step 1: $n = pq$, the RSA modulus

Step 2: $\lambda = \text{lcm}\ (p-1, q-1)$

Step 3: $g \in Z\ /n^2 Z$ s.t. $n |$ or $d_n{}^2(g)$

Step 4: Public-key: (n, g), secret key: λ, μ

2. Encryption of m :

Step 1: $m \in \{0, 1\ldots n-1\}$, a message

Step 2: $h \in_R Z/n\ Z$

Step 3: $c = g^m h^n \bmod n^2$, a cipher text

3. Decryption of c :

$m = L\ (c^{\lambda} \bmod n^2)\ L(g^{\lambda} \bmod n^2)^{-1} \bmod n$

The constant parameter,

$L\ (g^{\lambda} \bmod n^2)^{-1} \bmod n$ or $L\ (g^{\alpha} \bmod n^2)^{-1} \bmod n$

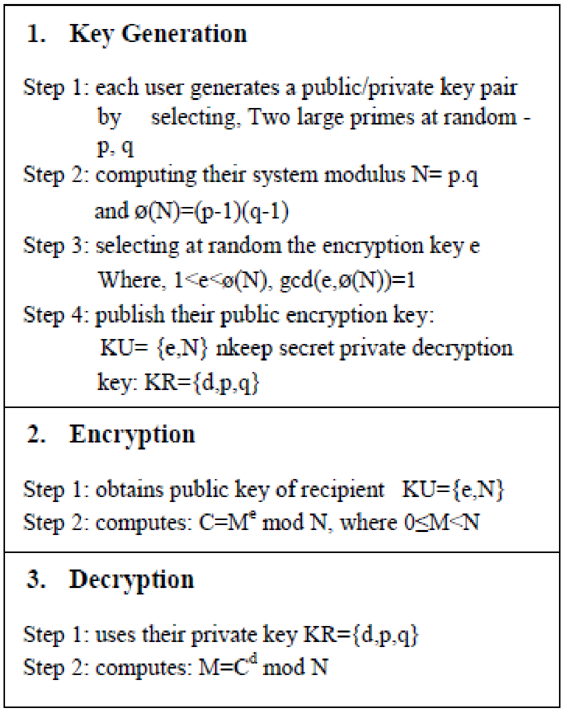
where $g=1+ n \bmod n^2$ can also be recomputed once for all.

Fig. 1: Paillier Algorithm

Suppose there are two ciphers CT1 and CT2 such that

CT1 = gm1x1n mod n2
CT2 = gm2x2n mod n2
CT1·CT2 = gm1x1n·gm2x2n mod n
Additive Property is: gm1+m2(x1x2)n mod n2

2. Multiplicative Homomorphic Encryption:
Multiplicative encryption function described by-
Ek (PT1⊗PT2) = Ek (PT1) ⊗ Ek (PT2)



1. Key Generation

Step 1: each user generates a public/private key pair by selecting, Two large primes at random - p, q
Step 2: computing their system modulus N= p.q and ø(N)=(p-1)(q-1)
Step 3: selecting at random the encryption key e Where, 1<e<ø(N), gcd(e,ø(N))=1
Step 4: publish their public encryption key: KU= {e,N} nkeep secret private decryption key: KR={d,p,q}

2. Encryption

Step 1: obtains public key of recipient KU={e,N}
Step 2: computes: $C=M^e$ mod N, where 0≤M<N

3. Decryption

Step 1: uses their private key KR={d,p,q}
Step 2: computes: $M=C^d$ mod N

Fig. 2: RSA Algorithm

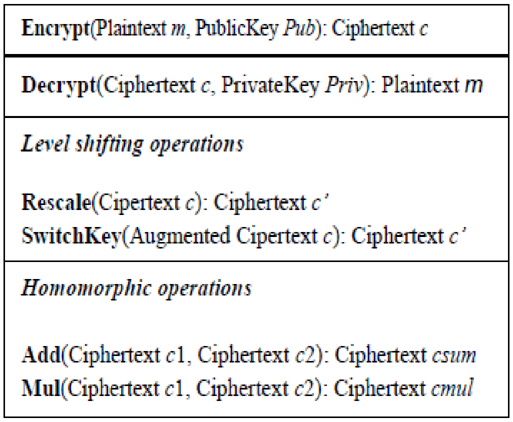
Suppose there are two cipher texts, CT1 and CT2.
CT1 = m1e mod n
CT2 = m2e mod n
CT1 · CT2 = m1e · m2e mod n
So, multiplicative property: (m1 · m2) e mod n

III. HOMOMORPHIC ENCRYPTION SCHEME

There are various Encryption Schemes such as BGV (Brakerski-Gentry-Vaikuntanathan), EHC (Enhanced Homomorphic Cryptosystem), NEHE (Non-iterative Exponential Homomorphic Encryption Algorithm), and AHEE (Algebra Homomorphic Scheme based on updated Elgamal).

A. BGV Encryption Scheme:

BGV is asymmetric encryption scheme which can be used for encryption of the bits.



Encrypt(Plaintext *m*, PublicKey *Pub*): Ciphertext *c*

Decrypt(Ciphertext *c*, PrivateKey *Priv*): Plaintext *m*

Level shifting operations

Rescale(Cipertext *c*): Ciphertext *c'*
SwitchKey(Augmented Cipertext *c*): Ciphertext *c'*

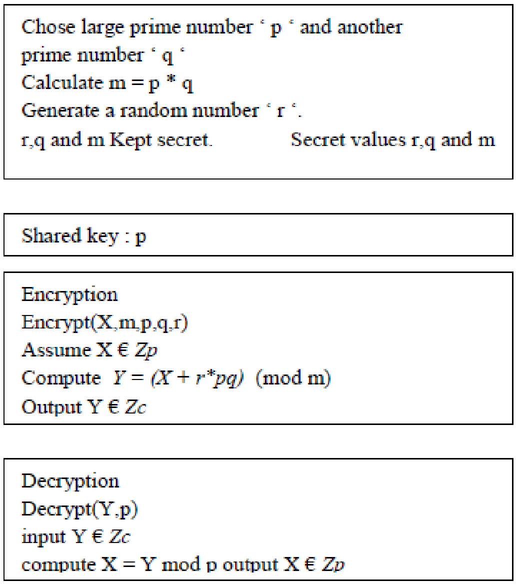
Homomorphic operations

Add(Ciphertext *c*1, Ciphertext *c*2): Ciphertext *csum*
Mul(Ciphertext *c*1, Ciphertext *c*2): Ciphertext *cmul*

Fig. 3: Encryption scheme.

B. Gotri's Enhanced Homomorphic Cryptosystem (EHC):

Homomorphic encryption has the concept of performing computation operation on the already encrypted data without having any information of real value and then encrypted data is will be sent back as a result and decrypted. This decrypted result will be equal to the computed data when performed on real data. For this Encryption Scheme:



Chose large prime number ' p ' and another prime number ' q '
Calculate m = p * q
Generate a random number ' r '.
r,q and m Kept secret.

Secret values r,q and m

Shared key : p

Encryption
Encrypt(X,m,p,q,r)
Assume X € *Zp*
Compute $Y = (X + r*pq)$ (mod m)
Output Y € *Zc*

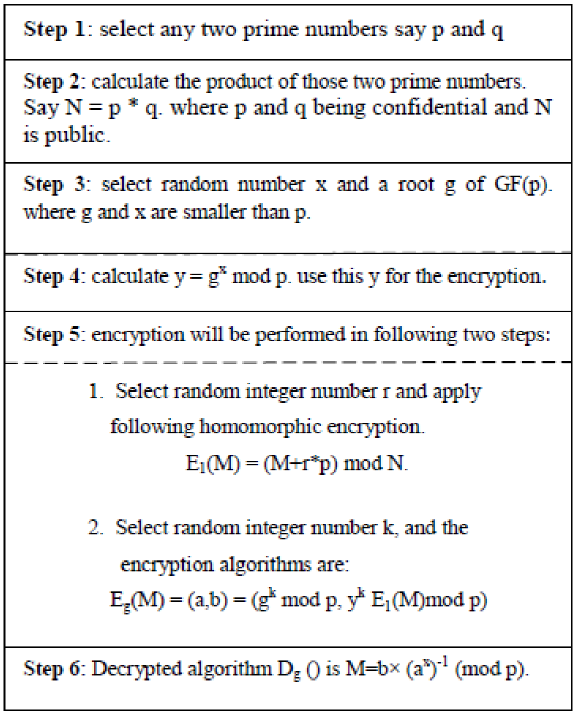
Decryption
Decrypt(Y,p)
input Y € *Zc*
compute X = Y mod p output X € *Zp*

Fig. 4: encryption/decryption of EHC scheme.

C. Algebra Homomorphic Encryption Scheme Based On Updated ELGamal (AHEE):

AHEE is modified form of Digital signature standard presented by NIST in America. It has been proved to be secured.

Additive and Multiplicative operation on integer ring:



Step 1: select any two prime numbers say p and q

Step 2: calculate the product of those two prime numbers. Say N = p * q. where p and q being confidential and N is public.

Step 3: select random number x and a root g of GF(p). where g and x are smaller than p.

Step 4: calculate $y = g^x \bmod p$. use this y for the encryption.

Step 5: encryption will be performed in following two steps:

1. Select random integer number r and apply following homomorphic encryption.
   $E_1(M) = (M+r*p) \bmod N.$
2. Select random integer number k, and the encryption algorithms are:
   $E_g(M) = (a,b) = (g^k \bmod p, y^k E_1(M) \bmod p)$

Step 6: Decrypted algorithm D_g () is $M = b\times (a^x)^{-1}$ (mod p).

Fig. 5: AHEE homomorphic encryption scheme.

Multiplicative: E (M1M2) = E(M1)·E (M2), or M1.M2=D (E (M1)·E(M2)) Additive: E (M1+M2) = E (M1) ⊕E (M2), or M1+M2=D (E (M1) ⊕E (M2)).

IV. LEVELS OF HOMOMORPHISM

There are two levels of Homomorphic Encryption:

1. Somewhat homomorphic encryption (SWHE):-

SWHE allows very few operations to be performed on the encrypted data without using any secret key. It can perform addition and multiplication up to a certain level only where noise falls so decryption is not possible due to increased noise levels with a message in cipher text.

2. Fully homomorphic encryption (FHE):-

FHE allows any kind of operation to be performed on the encrypted data; hence fully homomorphic encryption can perform addition and multiplication operation up to any level.

A. Basis of Fully Homomorphic Encryption:

1. On lattices and Bases: A lattice is a set of a vector that is a linear combination of the basis vector with integer coefficients. Thus it is a subset of the vector space and with the operation, on the point it is called a lattice. To solve the Closest Vector Problem (CVP) find closest lattice point and this point defines the closest lattice vector to the vector defined by the point in the vector space.
2. On learning of Error Problem:
   Distinguishing the problem with respect to the resulting linear combination with an error from a completely random vector is called Learning with error problem.

B. Bootstrapping:

The process of converting the somewhat homomorphic cryptosystem into a fully homomorphic cryptosystem. If somewhat homomorphic cryptosystem can evaluate its own decryption circuit with a small error constant than it boots trappable and this enables the implementation of a decrypt function which homomorphically decrypts the message and re-encrypts reducing the error constant.

Functions: KeyGen, Encrypt, Evaluate, Decrypt.

V. APPLICATION OF HOMOMORPHIC ENCRYPTION:

1. Cloud computation: Homomorphism allows retrieving and operating the operation on the encrypted data in the cloud.
2. Medical records: Homomorphism allows the third party to process the medical record without breaking the patient confidential record.
3. Electronic voting: Maintaining the privacy of individual votes statistics can be computed by the third party.
4. Financial transaction: Online Financial companies without ever seeing the user finance in clear text company can compute credit score and other financial data homomorphically.
5. Electronic cash: Transaction can be authenticated homomorphically by using blind signature.

VI. CONCLUSION

This paper presents the basic concept of homomorphic encryption with various encryption scheme and algorithm. This survey is helpful to know the detailed concept of homomorphic encryption for privacy preservation. In future, the applications should incentivize researchers to make progress toward a strong, fast fully homomorphic cryptosystem.

VII. REFERENCES

[1]. Craig Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009. crypto.stanford.edu/Craig.

[2]. Maha TEBAA, SaId EL HAJJI, Abdellatif EGHAZI , Homomorphic Encryption Applied to the Cloud Computing Security

[3]. Zvika Brakerski, Vinod Vaikuntanathan, "Efficient Fully Homomorphic Encryption from (Standard) L WE", FOCS, 2011

[4]. Oded Regev. The learning with errors problem (invited survey). In IEEE Conference on Computational Complexity, pages 191-204. IEEE Computer Society, 2010.

[5]. Ronald L. Rivest, Leonard Adleman, and Michael L. Dertouzos. On Data Banks and Privacy Homomorphisms, chapter On Data Banks and Privacy Homomorphisms, pages 169-180. Academic Press, 1978.

[6]. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS 2011. IEEE Computer Society (2011)

[7]. Samridhi singh, H.L.Mandoria, " A Review on Image Encryption Technique and to Extract Feature from Image" International Journal of Computer Applications (0975 – 8887) Volume 163 – No 1, April 2017

[8]. Hemani, H.L Mandoria, " Digital Watermarking Approaches and its Applications : A Review" International Journal of control theory and applications"(0974-5572) Volume 10-No 18, 2017

[9]. Hemani, Samriddhi Singh, " A Survey of Digital Watermarking Techniques and Performance Evaluation Metrics" International Journal of Engineering Trends and Technology"(2231-5381) Volume 46- No 2- April 2017