I. J. Computer Network and Information Security, **2017, 5, 44-51**
Published Online May 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2017.05.06

Homomorphic Cryptosystem

Alisha Rohilla
Department of Computer Science and Engineering & Information Technology, the NorthCap University Gurugram, 122002, India
E-mail: alisha15csp001@ncuindia.edu

Mehak Khurana and Meena Kumari
Department of Computer Science and Engineering & Information Technology, the NorthCap University Gurugram, 122002, India
E-mail: mehakkhurana@ncuindia.edu , drmeenakumari@yahoo.in

***Abstract*—**In 2009 Craig Gentry proved that Fully Homomorphic Encryption can be applied and realized in principle. Homomorphism allowed us to perform arbitrary computations and calculations on encrypted data. With RSA being the first cryptosystem to hold homomorphic properties, there came other additive and multiplicative cryptosystems. However, fully Homomorphic encryption proved to be the ultimate cryptographic solution to ensure security of data on cloud. It enables processing and computing arbitrary functions over the encrypted data thereby reducing the probability of accessing the plain text.

Index Terms—Homomorphism, Additive/Multiplicative Homomorphism, Somewhat Homomorphic encryption, Fully Homomorphic encryption.

I. Introduction

Homomorphic encryption "Fig .1," works on the concept of encrypting cipher text based on specific types of calculations and computations and generates an encrypted output which on decryption gives the result of calculations performed on the plaintext. [5]

Fully homomorphic encryption is a kind of ring homomorphism. Ring Homomorphism preserves the ring structure. We know real numbers are rings. Also the set of all 2×2 matrices is also a ring (under two matrix operations - addition and multiplication). If we define a function, f, between these rings as follows:

$$F(a) = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}$$

Where a is a real number, then F is a homomorphism of rings,

$$F(a+b) = \begin{bmatrix} a+b & 0 \\ 0 & a+b \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix} + \begin{bmatrix} b & 0 \\ 0 & b \end{bmatrix} = F(a) + F(b) \quad (1)$$

The above expression shows that F preserves additive homomorphism.

$$F(a\,.\,b) = \begin{bmatrix} a.b & 0 \\ 0 & a.b \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix} . \begin{bmatrix} b & 0 \\ 0 & b \end{bmatrix} = F(a)\,.\,F(b) \quad (2)$$

The above expression shows that F preserves multiplicative homomorphism.

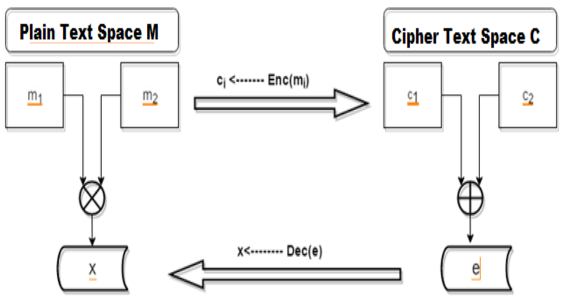

Fig.1. Homomorphic Property

Taking an Example, this explains the additive and multiplicative homomorphism.

Example 1:

Consider a set of natural numbers with addition (+) operation.

$$F(y) = 9y$$

Any function which preserves addition homomorphism should follow property stated in equation (1)

$$F(a+b) = F(a) + F(b)$$

Now using the equation (1), $F(y) = 9y$ can be written as

$$F(a+b) = 9(a+b) = 9a + 9b = F(a) + F(b).$$

Example 2:

Consider a set of natural numbers with multiplication (.) operation.

$$F(z) = 5z$$

Any function which preserves multiplication homomorphism should have the property stated in equation (2)

$$F(a\,.\,b) = F(a)\,.\,F(b)\,\}$$

Now using the equation (2), $F(z) = 5z$ can be written as

$$F(a\,.\,b) = 5(a\,.\,b) = 5a\,.5b = F(a)\,.\,F(b)$$

II. Homomorphic Encryption Techniques

There are many homomorphic encryption techniques which are explained below.

A. Multiplicative Homomorphic Encryption

1) *RSA:* If the RSA public key is modulus N and exponent g, then the encryption of a message M is given by[14][16]

$$E(M) = M^g \bmod N \tag{3}$$

The homomorphic property unpadded RSA holds is:

$$\begin{aligned} E(M_1)\cdot E(M_2) &= M_1^g M_2^g \bmod N \\ &= (M_1M_2)^g \bmod N = E(M_1M_2) \end{aligned} \tag{4}$$

Thus if we consider two plaintext messages M_1 and M_2, multiply them and then encrypt the result using RSA, we get a cipher text.

The multiplicative property says that you can also encrypt each plaintext individually and then multiply the two corresponding cipher texts together and you can obtain exactly the same result.

However, for security reasons RSA has to add padding bits to a plain text message before encrypting it. This padding of the message results is losing the homomorphic property. [1] Also, RSA is only Partially Homomorphic since the additive property does not apply. Thus, it can be said that RSA is not semantically secure. [15]

2) ELGAMAL Cryptosystem: It is defined over acyclic group G, this encryption scheme consists of following three sections, first is encryption, decryption and key generation.

Key Generation

There exists a cyclic group G of order d with generator g

Alice randomly selects x such that $x \in \{1,\ldots..,d-1\}$

$$\text{Calculate } b = g^x \tag{5}$$

Public Key: (G, g, d, b)
Private Key: x

Encryption

With Public key (G, g, d, b), Bob follows the following steps to encrypt plain text.

Chooses a random $n = \{1, \ldots., d-1\}$, calculates

$$c_1 = g^{\,n} \tag{6}$$

Calculates a shared secret

$$s = b^n \tag{7}$$

Converts his secret message M, into $M' \in G$ and calculates

$$c_2 = M'.s \tag{8}$$

Sends the following cipher text to Alice.

$$(c_1, c_2) = (g^n, M'.s) \tag{9}$$

Equation (9) becomes the cipher text to be sent to Alice.

Although if one knows M' he can easily find b^n. Consequently, in order to improve security a new 'n', is generated for every message. Therefore n is also called an ephemeral key.

Decryption

To decrypt cipher text pair (c_1, c_2) obtained in equation (9) with her private key x,

Alice computes the shared secret $t = c_1^x$ and calculates

$$\begin{aligned} M' = c_2.t^{-1} &= (M'.s)(c_1^x)^{-1} = (M'.b^n)(g^{-xn}) \\ &= (M'.g^{xn}.g^{-xn}) = (M') \end{aligned} \tag{10}$$

Homomorphic Property

ELGAMAL encryption scheme is a homomorphic scheme. This can be proved using example (3).

Let us consider example with two encryptions

Example 3:

$$\begin{aligned} (c_{11}, c_{12}) &= (g^{n1}, M_1' b^{n1}), \\ (c_{21}, c_{22}) &= (g^{n2}, M_2' b^{n2}) \end{aligned}$$

Where n_1, n_2 are randomly chosen from $\{1, \ldots\ldots, d-1\}$ and $M_1, M_2 \in G$, one can compute

$$\begin{aligned} (c_{11}, c_{12})(c_{21}, c_{22}) &= (c_{11}c_{12}).(c_{21}c_{22}) \\ &= \left((g^{n1}\,g^{n2}), (M_1'\,b^{n1}\;\; M_2'\,b^{n2})\right) \\ &= \left((g^{n1+n2}), (M_1M_2)b^{n1+n2})\right) \end{aligned} \tag{11}$$

B. XOR Homomorphic Encryption

3) *Goldwasser–Micali Encryption Scheme*: A probabilistic public-key encryption algorithm, the GM Encryption scheme has proven to be secure under standard cryptographic assumptions. [3]

It is the first semantically secure encryption scheme under the assumption that solving the quadratic residues problem is hard [4]. However, in GM encryption scheme cipher texts may be several times larger than the initial plaintext. This is because this scheme encrypts each bit of information and the length (size) of the resultant cipher text is equal to the length of the composite number n used in the scheme. Therefore it is not an efficient cryptosystem.

It consists of following three sections:

Key Generation

Choose two distinct random prime numbers p and q of similar bit-length.
Calculate $N = p.q$
Find a non-residue a such that

$$a_p^{(p-1)/2} = -1 \; mod \; p, \quad a_q^{(q-1)/2} = -1 \; mod \; q \qquad (12)$$

$Public\ key: \ (a, N)$
$Private\ key: \ (p, q)$

Encryption

To encrypt plain text M with public key (a, N) ,
Bob first encodes M as a string of bits $(M_1, M_2, \dots, M_n)$.
For every bit M_i, Bob generates a random value b_i , such that, $gcd(b_i, N) = 1$.
Calculate

$$C_i = b_i^2 . a^{M_i} (mod \; N) \qquad (13)$$

Decryption

Alice receives $(C_1, C_2, \dots, C_n)$ as cipher text from equation (13).
For each i, if C_i is a quadratic residue, $M_i = 0$, else $M_i = 1$,
Therefore message

$$M = (M_1, \dots., M_n) \qquad (14)$$

Goldwasser–Micali Encryption Scheme can be illustrated using example (4).

Example 4:

Key Generation

Let $p = 7, q = 11$
Where $p = q = 3 \; (mod \; 4)$
Thus, $N = pq = 77$
Let $a = 6$, where

$$6^{(7-1)/2} = -1(mod \; 7), 6^{(11-1)/2} = -1(mod \; 11)$$

Public Key: $(6,77)$
Private Key: $(7,11)$

Encryption

To encrypt 3-bit message $m_1 m_2 m_3 = 110$.
Choose $b_1 = 2, \; b_2 = 3, \; b_3 = 5$
Compute

$$c_1 = 22.61 = 24(mod \; 77)$$
$$c_2 = 32.61 = 54(mod \; 77)$$
$$c_3 = 52.60 = 25(mod \; 77)$$

Ciphertext is $(24,54,25)$

Decryption

To decrypt Cipher text (24, 54, 25)
Compute

$$24^{(7-1)/2} = -1(mod \; 7)$$
$$54^{(7-1)/2} = -1(mod \; 7)$$
$$54^{(11-1)/2} = -1(mod \; 11)$$
$$25^{(7-1)/2} = 1(mod \; 7)$$

This shows that 25 is quadratic residue and 24 and 54 are quadratic non-residue and thus the resultant plaintext is 110

Homomorphic Property

If C_1, C_2 are the encryptions of bits m_0, m_1
Then $C_1, C_2 \; (mod \; N)$ will be an encryption of $M_0 \oplus M_1$
Let us consider

$$C_1 = b_1^2 . a^{M_1} (mod \; N)$$
$$C_2 = b_2^2 . a^{M_2} (mod \; N)$$

We have

$$C_1 . C_2 = (b_1^2 . a^{M_1})(b_2^2 . a^{M_2})(mod \; N) = (b_1^2 b_2^2 \, . a^{M_1 + M_2}) mod \; N \qquad (15)$$

Analyzing equation (15),
When $M_0 + M_1$ is either 0 or 1, we have

$$M_0 + M_1 = M_0 \oplus M_1.$$

When $M_0 = M_1 = 1$, $M_0 + M_1 = 2$ and $C_0 C_1 (mod \; N)$ is a quadratic residue and thus it is an encryption of 0. In this case also we have

$$M_0 \oplus M_1 = 1 \oplus 1 = 0$$

C. Additive Homomorphic Encryption

1) *Paillier Encryption Scheme:* Paillier Cryptosystem is a probabilistic asymmetric key encryption scheme which uses different pairs of public and private key to encrypt and decrypt any plaintext. Paillier cryptosystem depends on a random element $\boldsymbol{r}$ for encryption per message bit.

Key Generation

Choose two large prime numbers p and q at random such that

$$gcd(pq\ ,(p-1)(q-1))=1$$

Calculate

$$n=pq$$

Calculate

$$\lambda = lcm\ (p-1,q-1) \tag{16}$$

Select generator g, such that $g \epsilon \mathbb{Z}^{*}_{n^2}$,

$$gcd\ (\ (g^{\lambda}\ mod\ n^2\ -\ 1)/n\ \ ,n)=1$$

Calculate

$$\mu = (L(g^{\lambda}\ mod\ n^2\ ))\ -\ 1\ mod\ n, \tag{17}$$

where $L(u)=(u-1)/n$.

This function is only used on input values u that actually satisfy u = 1 mod n [6].

Public Key: (n,g)
Private Key: (λ, μ)

Encryption

Plaintext, where $m\ \in\ \mathbb{Z}_n$
Select random r where $r \in \mathbb{Z}^{*}_{n}$
Compute cipher text as:

$$c\ =g^m \cdot r^n\ mod\ n^2 \tag{18}$$

Decryption

As implied from equation (18),
Cipher text $c\ \in \mathbb{Z}^{*}_{n^2}$
Compute message:

$$m\ = L(c^{\lambda}\ mod\ n^2)\cdot \mu\ \ mod\ n \tag{19}$$

Paillier Encryption Scheme can be illustrated using the following example (5).

Example 5:

Key Generation

Let

$p\ =\ 49727\ q\ =\ 56737$
$n\ =\ pq\ =\ 2821360799$
$n^2\ =\ 7960076758133918401$
$\lambda\ \ =\ lcm\ (p-1,q-1)\ =\ 1410627168$
Choose a random, $g\ =\ \ 1624691728$
$L\ =\ 2197925655$
$\mu\ =\ 1779031213$

Public Key: $(2821360799, 1624691728)$
Private Key: $(1410627168, 1779031213)$

Encryption

Plaintext, $m\ = 1468689009$
r =1489472025
Cipher text

$$c\ =\ g^m\ \cdot r^n\ mod\ n^2 = (956651757)^{1468689009}\ .(1489472025)^{2821360799}\ mod\ 7960076758133918401 =\ 2015851325878209300$$

Decryption

$$m\ = L\big(c^{\lambda}\ mod\ n^2\big)\cdot \mu\ \ mod\ n = \boldsymbol{L}\ (2015851325878209300^{1410627168}\ mod\ 7960076758133918401).1779031213\ mod\ 2821360799 = 1468689009$$

Homomorphic Property

Paillier Cryptosystem holds the property of additive homomorphism.

The product of two ciphers gives the sum of their corresponding plaintexts on decryption.

$$D\big(E(m1,r1) *\ E(m2,r2)\big)mod\ n^2 =\ m1\ +\ m2\ mod\ n \tag{20}$$

III. FULLY HOMOMORPHIC ENCRYPTION

Let (P,C,K,E,D) be an encryption scheme where [2][11]

P: Plaintext
C: Ciphertext
K: Keyspace
E: Encryption Algorithm
D: Decryption Algorithm.

Assume that the plaintexts form a ring $(M\ ,\otimes_M,\oplus_M)$ and the ciphertexts form a ring $(C,\otimes_C,\oplus_C)$ the encryption algorithm E is a map from the ring M to C, i.e. ,

$$E_k\ :\ M\ \rightarrow\ C,$$

where $k\ \epsilon\ K$ is either a secret key or a public key .
For all x and y in M and k in K, if

$$E_k(x)\ \oplus_C\ E_k(y)\ =\ E_k\ (x\ \oplus_M\ y) \tag{21}$$

$$E_k(x)\ \otimes_C\ E_k(y)\ =\ E_k\ (x\ \otimes_M\ y) \tag{22}$$

then the encryption scheme is fully homomorphic.

A. *Classification of Fully Homomorphic Encryption*

Let us begin with a space $P\ =\ \{0,1\}$, plaintext space, and a family F of functions from tuples of plaintexts to P, expressed as a Boolean circuit on its inputs, referred by C. **[7]**

Input tuple $(m_1, m_2, \ldots, m_n)$ denotes the plain text.

The classification of fully homomorphic encryption is depicted in Fig. 2. Corresponding definitions and explanation about each classification can be found in the following subheadings.

1) ***ℓ** – Evaluation Policy*

Let ℓ be a set of circuits. A ℓ –evaluation policy for ℓ is a tuple of probabilistic polynomial–time algorithms $(KeyGen, Encr, Eval, Decr)$ such that:

The key generation algorithm, $KeyGen(1^{\lambda}, \alpha)$, takes two inputs, security parameter λ and an auxiliary input α, and outputs a key triplet (p, s, e), where p denotes the public encryption key used for encryption, s denotes the secret key used for decryption and e denotes the evaluation key used for evaluation.

The encryption algorithm, $Encr(p,m)$, takes a plaintext m and the public encryption key p and as inputs and outputs a cipher text c.

The evaluation algorithm, $\mathrm{Eval}(\mathrm{e}, \mathrm{C}, \mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{n})$, takes a circuit $\mathrm{C} \in \ell$, the evaluation key e and a tuple of inputs that can be a mix of ciphertexts and previous evaluation results as inputs and generates an evaluation output.

The decryption algorithm, $(\mathrm{s_k}, \mathrm{c})$, accepts the secret key s and either a ciphertext or an evaluation output and produces a plaintext m.

Assuming the following convention,

$\mathcal{X}$ denotes the ciphertext space
$\mathcal{Y}$ denotes the space of evaluation outputs, and
$\mathcal{Z}$ is the union of both X and Y.
$\mathcal{Z}^*$ contains arbitrary length tuples made up of elements in Z.

The key spaces are denoted by $\mathcal{K}_\mathrm{p}, \mathcal{K}_\mathrm{s}$ and $, \mathcal{K}_\mathrm{e}$, respectively for p, s and e.

The public key contains a description of the plaintext and ciphertext spaces.

ℓ is the set of permissible circuits, i.e. all the allowed circuits which the evaluation policy can evaluate.

The domain and range of the algorithms are given by

KeyGen: $\boldsymbol{N} \times \boldsymbol{A} \rightarrow \mathcal{K}_p \times \mathcal{K}_s \times \mathcal{K}_e$
Encr: $\mathcal{K}_\mathrm{p} \times \mathrm{M} \rightarrow \mathcal{X}$
Decr: $\mathcal{K}_\mathrm{s} \times \mathcal{Z} \rightarrow \mathrm{M}$
Eval: $\mathcal{K}_\mathrm{e} \times \ell \times \mathcal{Z}^* \rightarrow \mathcal{Y}$

where $\mathcal{X} \cup \mathcal{Y} = \mathcal{Z}$ and A is an auxiliary space.

Formally,

$$\mathcal{X} = \{\mathrm{c} \,|\Pr[\mathrm{Encr(p,m)} = \mathrm{c}] > 0, m \in P\}, \qquad (23)$$

$\mathcal{X}$ in equation (23) can be considered an image of encryption.

And

$$\mathcal{Y} = \{ \mathcal{Z} \,|\Pr[\mathrm{Eval}(\mathrm{e}, \mathrm{C}, \mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{n}) = \mathcal{Z}] > 0, \; \mathrm{c_i} \in \mathcal{Z}, \text{ and } \mathrm{C} \in \ell \}, \qquad (24)$$

$\mathcal{Y}$ in equation (24) can be considered an image of evaluation.

a) Strict Decryption

Any ℓ–evaluation policy $(KeyGen, Encr, Eval, Decr)$ is said to correctly decrypt if for all $m \in P$,

$$Pr[Decr(s, Encr(p,m)) = m] = 1, \qquad (25)$$

Where s and p are outputs of $KeyGen(1^{\lambda}, \alpha)$.

This means that we must be able to decrypt a cipher text to the correct plaintext, without any error. [8]

b) Strict Evaluation

Any ℓ–evaluation policy $(KeyGen, Encr, Eval, Decr)$ is said to correctly evaluate all circuits in C if for all $c_i \in \boldsymbol{\mathcal{X}}$ where $m_i \leftarrow Decr(s, ci)$, for every $C \in \ell$, and some negligible function ε, it satisfies equation (26).

$$\Pr\big[Decr\big(s, Eval(e, C, c_1, \ldots, c_n)\big) = C(m_1, \ldots, m_n)\big] = 1 - \varepsilon(\lambda) \qquad (26)$$

Where s, p *and* e are outputs of $KeyGen(1^{\lambda}, \alpha)$

This means that decryption of the homomorphic evaluation of an allowed circuit yields the correct result. [7][12,Def 3.3]

Thus, it can be said that a ℓ- evaluation scheme is correct if it has the properties of both correct evaluation and correct decryption.

Consequently the encryption scheme is Somewhat Homomorphic.

2) Somewhat Homomorphic Encryption

Any ℓ–evaluation policy $(KeyGen, Encr, Eval, Decr)$ that holds a correct and valid decryption as well as an evaluation is called Somewhat Homomorphic Encryption Scheme (SHE).

This level of homomorphic encryption doesn't require Compactness, and as a result the size of the cipher text can substantially increase with each homomorphic operation. Also, while making the set of permissible circuits, C, there is no requirement to mention which circuits this must include.

Secret Key Somewhat Homomorphic Encryption

$KeyGen$: From some interval $p \in [2^{\eta-1}, 2^{\eta}]$, choose an odd integer which acts as a secret key for encryption.

$Encr(pk, m)$: In order to encrypt plain text bit, $\in \{0,1\}$:

Choose an integer whose residue $mod\ p$ has the same parity as the plaintext and set the cipher text as this integer.

Namely, set

$$c = pq + 2r + m; \qquad (27)$$

Where q and r are chosen randomly in some other intervals, such that $p/2$ is greater than $2r$ in absolute value.

$Decr\ (p, c)$ **:** Given a cipher text c and the secret key p, output

$$m = (c(mod\ p))(mod\ 2)$$
$$= \big((pq + 2r + m)mod(p)\big)(mod\ 2)$$
$$= 2r + m\ (mod\ 2) = m \qquad (28)$$

Example 6:

Suppose p= 23 ; bit to encrypt $m = 1$,
Then $c = 23.2 + 2.0 + 1 = 47$,
where $q = 2, r = 0$
Now to decrypt it back to m,

$$m = (c(mod\ p))(mod\ 2)$$
$$= (47\ mod\ 23)\ (mod\ 2) = 1\ (mod\ 2) = 1 = m$$

Property of Fully Homomorphic Encryption

Suppose we have two cipher texts,

$$c_1 = pq_1 + 2r_1 + m_1$$

and

$$c_2 = pq_2 + 2r_2 + m_2$$

Then

$$c_1 + c_2 = \big(p(q_1 + q_2) + 2(r_1 + r_2) + (m_1 + m_2)\big) \qquad (29)$$

$$c_1 . c_2 = (p . q_1 . q_2 + 2\, q_1\, r_2 + 2\, q_2\, r_1 + m_1 + q_1\, m_2)p + 2(2\, r_1\, r_2 + r_2\, m_1 + r_1\, m_2) + (m_1\ m_2) \qquad (30)$$

When

$$(r_1 + r_2) < p/2$$
$$2\, r_1\, r_2 + r_2\, m_1 + r_1\, m_2 < p/2$$

Thus we have,

$$(c_1 + c_2(mod\ p))mod\ 2) = m_1 + m_2 \qquad (31)$$

$$(c_1 . c_2(mod\ p))mod\ 2) = m_1 . m_2 \qquad (32)$$

Example 7:

Let

$$p = 23, m_1 = 0 \qquad m_2 = 1,$$
$$q_1 = 1, q_2 = 2, r_1 = 1, r_2 = 2$$

$$c_1 = 23.1 + 2.1 + 0 = 25$$
$$c_2 = 23.2 + 2.2 + 1 = 51$$

Now,

$$(c_1 + c_2(mod\ p))mod\ 2)$$
$$= \big((25 + 51)(mod\ 23)\big)(mod\ 2)$$
$$= 76(mod\ 23)(mod\ 2)$$
$$= 7\ mod\ 2 = 1$$
$$= m_1 + m_2$$

And

$$(c_1 . c_2(mod\ p))mod\ 2) = (25.51)\ (mod\ 23))(mod\ 2)$$
$$= 1275(mod\ 23)(mod$$
$$= 10\ mod\ 2 = 0$$
$$= m_1 . m_2$$

However, it has been seen that while using the fully homomorphic property to evaluate a Boolean function $f(m_1, m_2\ \dots, m_n)$ where $m_i \epsilon \{0, 1\}$, given c_i, the encryption of m_i, for $i = 1, 2, \dots, n$.

As the number of the additions and multiplications in the Boolean function grow so does the size of the noise component r in the resultant cipher text. Consequently the size of the noise component is proportional to the number of operations.

And hence only low-degree Boolean functions (circuits) can be evaluated over encrypted data.

This is the reason this scheme is termed Somewhat Homomorphic.

a) Compactness

A Somewhat Homomorphic Scheme (SHE) is said to be compact if there exists a polynomial q = q(λ) , such that for any key-triplet (s, p, e) generated by $KeyGen(1^{\lambda}, \alpha)$, any circuit $C \in \ell$ and all cipher texts $c_i \in \mathcal{X}$, the size of the output from Eval (e,C, c_1,….., c_n) is at most q(λ) bits long (regardless of the number of inputs or C).

According to Craig Gentry, if in addition the run time of the decryption circuit depends only on λ and not on any of its inputs, the scheme is said to compactly *evaluate* C. (Gentry, 2014)

However it was observed [8] that any ℓ – evaluation policy $(KeyGen, Encr, Eval, Decr)$ *compactly evaluates* all circuits in ℓ if the scheme is compact and correct.

This implies that the cipher text size doesn't grow much during homomorphic operations and the output size depends on the security parameter, λ, only.

b) Circuit Privacy

Anyℓ– evaluation policy $(KeyGen, Encr, Eval, Decr)$ is said to be perfectly/statistically/computationally circuit private if for any key-triple (s, p, e) output by $(1^{\lambda}, \alpha)$, for all circuits C ∈ $\mathcal{C}$ and all $c_i \epsilon \mathcal{X}$, such that $m_i \leftarrow Dec(s, c_i)$, the two distributions on $\mathcal{Z}$

$$D_1 = Eval(e, C, c_1, , , , , , , , , c_n) \qquad (33)$$

And

$$D_2 = Enc(p, C(m_1, , , , , , , m_n) \qquad (34)$$

both taken over the randomness of each algorithm, are perfectly, statistically or computationally indistinguishable, respectively.[8]

3) Levelled Homomorphic Encryption

Aℓ – Evaluation $policy$ $(KeyGen, Encr, Eval, Decr)$ is said to be "levelled homomorphic" if its key generation algorithm, *KeyGen*, accepts an auxiliary input $\alpha = d$ which clearly identifies the maximum depth (size) of circuits that can be evaluated. Also the encryption should be correct, compact and the length of evaluation output should not depend on depth,d of the circuit. ([7], Def. 3.6)

4) Fully Levelled Homomorphic Encryption

A ℓ – Evaluation policy $(KeyGen, Encr, Eval, Decr)$ is said to be "fully levelled homomorphic" if the set ℓ is the set of all binary circuits with depth atmost d.

Apparently, in Somewhat Homomorphic Encryption, the depth of the circuit can vary depending on a parameter. This means that the length of cipher text will increase depending on the depth of the permissible circuits. However, this is not the case with Levelled Homomorphic Encryption in which the length of the cipher text does not depend on the depth d, of the circuit.

5) Fully Homomorphic Encryption

A fully homomorphic encryption scheme is a ℓ - evaluation $(KeyGen, Encr, Eval, Decr)$ that is compact, correct and where ℓ is the set of all circuits. ([7], Def. 3.5)[9].

$Bootstrapability$ is a condition in which the degree of the evaluation polynomial that is to be applied on cipher text exceeds the degree of the decryption polynomial. Once the scheme becomes bootstrappable it can be converted into a fully homomorphic encryption scheme by entering the encryption of the secret key bits inside the public key. [10]. According to Gentry, a somewhat encryption scheme can be converted into fully homomorphic encryption using boot strapping .[12]

Given a homomorphic scheme, we can homomorphically compute any function. Theoretically we can: [13]

- Encrypt the encrypted data with a new key
- Encrypt the old key with the new one
- Evaluate the decryption procedure homomorphically, thereby resulting in a cipher text encrypted with the second key.

Based on Gentry's approach, two different fully homomorphic schemes are known: Gentry's scheme [11] based on ideal lattices and a scheme by van Dijk, Gentry, Halevi and Vaikuntanathan (DGHV) over the integers which appeared at Eurocrypt 2010 [9].

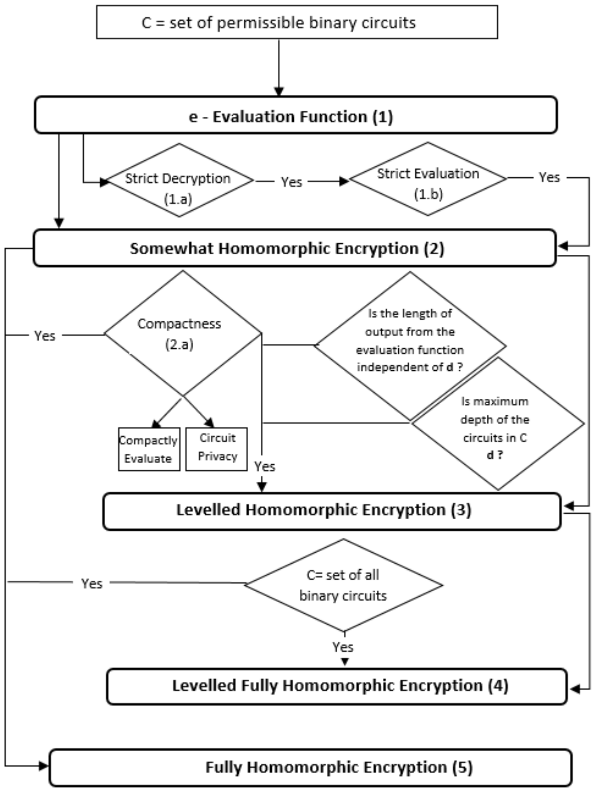

Fig.2. Classification of Fully Homomorphic Encryption

IV. CONCLUSION

This paper provides its readers with the basic idea and mechanism involved in the recently evolved homomorphic and fully homomorphic encryption schemes.

Using homomorphic encryption to secure data prevents plain text from being exposed. Thus, homomorphic encryption has given a new dimension to cloud storage and security. There are various homomorphic cryptosystems available and now there is a need to develop Fully Homomorphic cryptosystems which meet all the criteria of being compact, correct and applicable on all functions/circuits. With the advent of Fully Homomorphic Cryptosystem, the data has become semantically secure.

ACKNOWLEDGEMENT

I am highly indebted to Dr. Meena Kumari for not just providing guidance and supervision but also for providing encouragement and necessary information while carrying out research on this topic. I would like to express my gratitude towards Ms. Mehak Khurana for her kind co-operation and guidance which helped me accomplish this paper. My thanks and appreciations also go to my colleagues in developing the research base and people who have willingly helped me out with their abilities.

REFERENCES

[1] Xun Yi, Russell Paulet , Elisa Bertino, *Homomorphic Encryption and Applications*, Springer 2014

[2] Gentry C., *A Fully Homomorphic Encryption Scheme*, 2009, Chapter 2, Available at http://crypto.stanford.edu/craig

[3] S. Goldwasser, S. Micali, *Probabilistic encryption and how to play mental poker keeping secret all partial information*, in Proceedings of 14th Symposium on Theory of Computing, 1982,pp. 365–377

[4] Kazue Sako, *Goldwasser–Micali Encryption Scheme*, Encyclopaedia of Cryptography and Security, 2011

[5] Iram Ahmad and Archana Khandekar, Homomorphic Encryption Method Applied to Cloud Computing, International Journal of Information & Computation Technology,2014, pp. 1519-1530

[6] Pascal Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*. Advances in Cryptology - EUROCRYPT'99, vol. 1592 of Lecture Notes in Computer Science, pp. 223-238, 1999

[7] Vaikuntanathan, Zvika Brakerski and Vinod, *Efficient Fully Homomorphic Encryption*, IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, IEEE, 2011, pg: 97-106

[8] Frederik Armknecht, Colin Boyd, Christopher Carr, Kristian Gj_steen, Angela Jaschke, Christian A. Reuter, and Martin Strand , *A Guide to Fully Homomorphic Encryption*, 2015.

[9] M. van Dijk, C. Gentry, S. Halevi and V. Vaikuntanathan, *Fully Homomorphic Encryption over the Integers*. In H. Gilbert (Ed.), EUROCRYPT 2010, LNCS, vol. 6110, Springer, 2010, pp. 24–43

[10] Jean-S ́ebastien Coron, Avradip Mandal, David Naccache , and Mehdi Tibouchi; *Fully Homomorphic Encryption over the Integers with Shorter Public Keys*.

[11] Gentry, C. (2009). *Fully Homomorphic Encryption Using Ideal Lattices*. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC'09), pp. 169-178, ACM Press, New York, NY, USA.

[12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. *Fully homomorphic encryption without bootstrapping*. Electronic Colloquium on Computational Complexity (ECCC), 18:111, 2011.

[13] http://blog.quarkslab.com/a-brief-survey-of-fully-homomorphic-encryption-computing-on-encrypted-data.html

[14] R. L. Rivest., A. Shamir, L. M. Adleman " A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, 21(2):120–126, 1978

[15] D. Boneh, "Twenty Years of Attacks on the RSA cryptosystem", Notices of the AMS, 46(2):203–213, 1999.

[16] Mehak Khurana, Meena Kumari, "Security Primitives: Block and Stream Ciphers", International Journal of Innovations & Advancement in Computer Science (IJIACS), ISSN 2347 – 8616, Vol. 4, March 2015.

Authors' Profiles

Alisha Rohilla is a MTech student of Department of Computer Science and Engineering & Information Technology of The NorthCap University, Gurugram, specialised under the huge umbrella of Cyber Security. She completed her BTech in Computer Science from Institute of Technology and Management, Gurgaon in 2012 after which she worked with TATA Consultancy Services for 2.5 years as a System Engineer. Her area of interest are cryptography, cyber security, digital forensics, and identity and access management.

Mehak Khurana is currently working as assistant professor in The NorthCap University in CSE & IT and has around 6 years of experience. She completed her M.Tech from USIT, GGSIPU in 2011 and B.Tech from GTBIT, GGSIPU in 2009. Her key areas of interest are Cryptography, Information Security and Cyber Security. She is lifetime member of Cryptology Research Society of India (CRSI).

Meena Kumari, has worked as a professor, Dept of CSE&IT at The NorthCap University. She has also worked as Scientist 'G' at DRDO (Defence Research & Development Organization) and has 37 years of research experience in cryptology.

How to cite this paper: Alisha Rohilla, Mehak Khurana, Meena Kumari,"Homomorphic Cryptosystem", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.5, pp. 44-51, 2017.DOI: 10.5815/ijcnis.2017.05.06