Scientific Journal of Impact Factor (SJIF): 3.134

ISSN (Online): 2348-4470
ISSN (Print) : 2348-6406

International Journal of Advance Engineering and Research Development

Volume 2, Issue 2, February -2015

A SURVEY ON HOMOMORPHIC ENCRYPTION TECHNIQUES IN CLOUD COMPUTING

Jigar M. Shah[1], Asst. Prof. Hemangi Kothadiya[2]

[1]*Student Computer Science & Engineering Department, Parul Institute of Engineering and Technology, jigarshah1302@gmail.com*

[2]*Information Technology Department, Parul Institute of Engineering and Technology, hemangi1501@gmail.com*

Abstract- *People can work on applications or programs, which would be use by them on their computers, Instead cloud computing allows people to do same thing without downloading on their system. Security is a major concern in cloud as other users can access the data stored by one user. To overcome the security issue the cloud encryption is a method whereby data converted using algorithms and then stored in cloud. Homomorphic encryption is a process in which the user first encrypts the data and then stores the encrypted data in cloud, so the cloud provider is unable to recognize the data..In this paper, different Homomorphic Encryption techniques have been surveyed and discussed.*

Keywords: *Cloud Computing, Security, Homomorphic Encryption.*

I. INTRODUCTION

In the simplest terms, Cloud Computing means storing and accessing data and programs over the Internet instead of your computer's hard drive. NIST define specific definition for cloud computing here [5].

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provision and release with minimal management effort or service provider interaction[5].

A. Characteristics of Cloud Computing [10]:

- **Elasticity:** it is one of the most essential characteristics of our vision of Cloud. It defines the ability of a given infrastructure to dynamically adapt to a scale.
- **Ability to adapt:** Cloud must provide a set of automatization allowing it self-management. Its administration should require a minimum human intervention.
- **Quality of Service:** is another key aspect of Cloud, using metrics such as time response, the number of operations in a second; the service provides guarantees to its users. It no longer belongs to the user having to decide what resources to deploy but rather to define terminals that the service should meet. Cloud adapts to ensure its terminals.
- **High Availability:** playing on replicated data in different data canters, the Cloud must provide reliable, not sensitive to the failure of an instance or a data centres .
- **Cost reduction:** Pay Per Use, means that the use only pays for the service based on its utilization.
- **Ecological approach:** the allocation of resources to the strict necessity to reduce the energy consumption of IT parks. Beyond the economic aspect, these reductions allow the ecological energy reduction footprint of the company.

B. Delivery Model [8]:

Cloud Software as a Service (SaaS) [8]: Offers users an easier way to access many of their standard business applications and services such as email and word processing packages etc, by allowing users to access these programs through the internet, there is no need to install and run the special software on your computer if you use the SaaS.
Examples of SaaS are Google's Gmail.

Cloud Platform as a Service (PaaS) [8]: Is a set of cloud-delivered services that provide an environment for application development, deployment, management and integration in the cloud.
Examples of PaaS are Google's App Engine and Microsoft's Azure platform.

Cloud Infrastructure as a Service (IaaS) [8]: Is known for providing computational and storage infrastructure in a centralized, location-transparent service. The infrastructure that is provided by the CSP includes storage, servers, bandwidth and network equipment, which includes software that monitors the use of the infrastructure and allows the user to only pay for what they use.
Examples of IaaS are Amazon's Elastic Compute Cloud (EC2).

C. Deployment Model [10]

Public Cloud [10]: As the name suggests, it is to share with the "public" (user at large) an infrastructure that belongs to a cloud provider, which leases its services to companies on demand. Its main role is to host applications, web in general, only accessible via the Internet, so it is an optimal pooling of resources based on the creation of a multitude of execution environment on a same platform.

Private Cloud [10]: It is to transform the internal infrastructure of a computer system through virtualization technologies, providing services and resources to clients on demand. These services are hosted by the client company or by the cloud provider (with a VPN connection).

Hybrid Cloud [10]: It is to coexist and communicate a private cloud and public cloud. The infrastructure consists of two or more clouds (private, Community or Public). The hybrid is often used to encompass the peak time charges as with the public, remaining linked to private Cloud. Both infrastructure therefore communicate and form a hybrid cloud, then it is a way to combine the benefits of both platforms.

Community Cloud [10]: The infrastructure is shared by several organizations that have common interests (e.g. security requirements, compliance considerations...). As the Private Cloud, it can be managed by the organizations themselves or by a third party.

II. OVERVIEW OF HOMOMORPHIC ENCRYPTION

Homomorphic encryption allows us to perform operations on encrypted data without knowing the private key and without decrypting that data. When we decrypt the result of any operation, it is the same as if we carried out the calculation on the plain data [11].

In homomorphic encryption scheme first, user encrypts data and then store it in the cloud. Weather cloud providers don't know about which data are actually stored in the cloud by end user. If user wants to add data in cloud, for that used additive and multiplicative properties of homomorphic encryption.

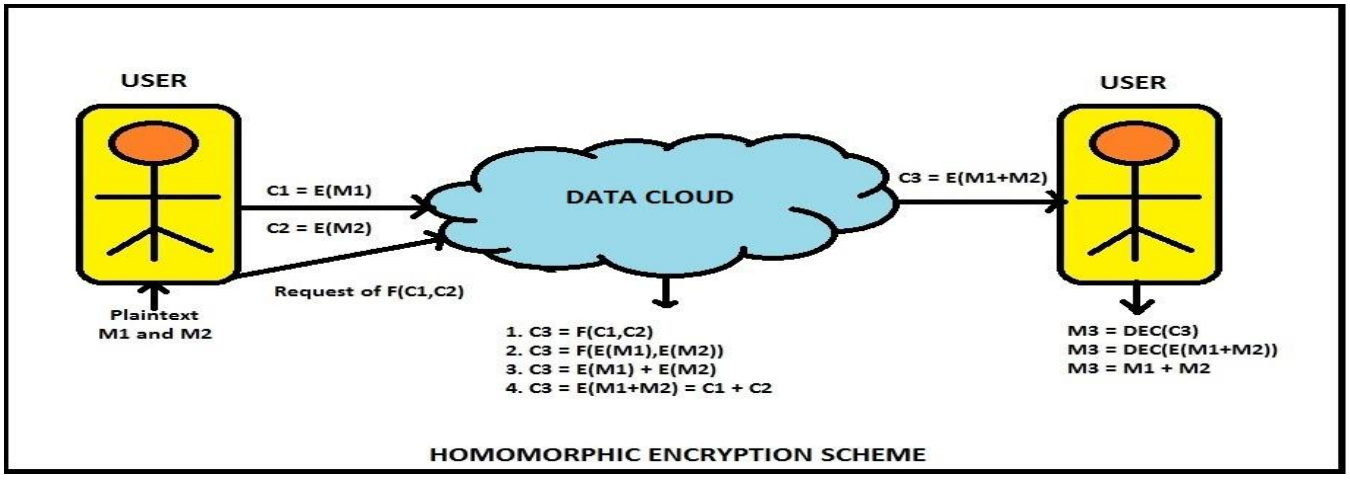

Figure 1. Homomorphic Encryption Scheme

Here we give basic notions about homomorphic encryption and schemes presented [11].

- A Homomorphic encryption is additive if:

Enc(x + y) = Enc(x) (H+) Enc(y)

- A Homomorphic encryption is multiplicative if:

  Enc(x * y) = Enc(x) (H*) Enc(y)

In this paper signs used (H+) as homomorphic addition and (H*) as homomorphic multiplication.

Procedure follows For Homomorphic Encryption algorithm [11]

1. Key generation is the algorithm that outputs public key Pk and secret key Sk.
2. Encrypt function is the algorithm that takes public key Pk and a message m and outputs ciphertext c.
3. Decrypt function is the algorithm that takes ciphertext c and a secret key Sk and outputs a message m.
4. Evaluate function is an algorithm that takes evaluation key, circuit, a set of ciphertexts c1,…..,cn and outputs a ciphertext ce. Circuit represents a certain function realized with logical gates.

Types of Homomorphic Encryption in Cloud Computing :

There are mainly two types of Homomorphic Encryption Techniques:

2.1 Partially Homomorphic Encryption Technique:

In partially homomorphic encryption scheme only supports one type of operation, either addition or multiplication. Some example of partially homomorphic encryption is electronic voting, financial transaction, and medical record. In partially homomorphic encryption scheme mainly 3 sub system are there:

Additive homomorphic encryption system: In Additive homomorphic encryption system, only one type of addition operation is allowed. Example of this system is paillier encryption scheme.

Multiplicative homomorphic encryption system: In Multiplicative homomorphic encryption system, only one type of multiplication operation is allowed. Example of this system is RSA encryption scheme and El-Gamal Encryption scheme.

Additive and multiplicative homomorphic encryption system: In Additive and multiplicative homomorphic encryption system, in that arbitrary time many operations of one type and limited (one) number of operation of other type.
Example of this system is BGN encryption scheme.

2.2 Fully Homomorphic Encryption Technique:

In fully homomorphic encryption scheme, any number of addition and multiplication operations can be perform on encrypted data without decrypting the data.
Some examples of fully homomorphic encryption scheme is Gentry's fully homomorphic encryption scheme, DGHV homomorphic encryption scheme and BGV homomorphic encryption scheme.

- **Comparison Between Existing Homomomorphic Encryption Method :** Comparison between partially homomorphic encryption scheme and fully homomorphic encryption scheme is mention below.

Sr No.	Functionality	Partially HE	Fully HE
1	Operation Support on Data	Only one type of operation (* or +)	Both type of operation (* and +)
2	Ciphertext Size	Small	Large
3	Versatility	Low	High
4	Speed	Fast	Slow
5	Security	Low	High

6	Current application	Available	Not Available
7	Noise	No noise is added in Ciphertext	Noise is added in ciphertext linearly or exponentially

Table 1.Comparision Between PHE and FHE

III. EXISTING HOMOMOMORPHIC ENCRYPTION SCHEMES

In this section, some existing Homomorphic Encryption Schemes have explained briefly from papers reviewed. This all homomorphic encryption Schemes are for different intentions, each algorithm is founded in references provided.

3.1 Paillier Encryption Scheme [3]:

This technique used in public key cryptography is the use of asymmetric key algorithms, where the key used to encrypt a message is not the same as the key used to decrypt it. Each user has a pair of cryptographic keys - a public key and a private key. The private key is kept secret, while the public key may be widely distributed.
This scheme is additive homomorphic encryption scheme, which can perform multiple times of addition operation.
Paillier's main application is Electronic Voting where each vote is encrypted, only the sum is decrypted.

- Description of paillier scheme is given below:

The following notations are used frequently in Paillier Cryptosystem explanation:

Z_n - Set of integers n

Z^*_n - Set of integers coprime to n - this set consists of $\Phi(n)$ number of integers

$Z^*_{n^2}$ - Set of integers coprime to n^2 - this set consists of $n\Phi(n)$ number of integers

Key Generation: KeyGen (p,q)

1. Choose two large prime numbers *p* and *q* randomly and independently of each other such that, gcd (pq , (p-1)(q-1))=1 This property is assured if both primes are of equivalent length.
2. Compute RSA modulus n = pq and Carmichael's function λ = lcm(p-1, q-1) it can be computed using
$$\lambda = \frac{(p-1)(q-1)}{\gcd(p-1)(q-1)}.$$
3. Randomly select generator g where g € $Z^*_{n^2}$ where λ
$$\gcd\left(\frac{g^{\lambda} \bmod n^2-1}{n}, n\right) = 1$$
4. Calculate the following modular multiplicative inverse
$\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n$

Where the function L is defined as $L(u)= \frac{u-1}{n}$.

This multiplicative inverse exists if and only if valid generator was selected in previous step.

- The public encryption key is (n, g).
- The private decryption key is (λ, μ).

Encryption: Enc (m, p_k)

1. Let m be a message to be encrypted where m € Z_n
2. Select random r where r € Z^*_n
3. Compute ciphertext as c = $g^m . r^n \bmod n^2$

Decryption: Dec(c, s_k)

1. Ciphertext c € $Z^*_{n^2}$
2. Compute message: m = L ($c^{\lambda} \ mod \ n^2$). μ mod n

Evaluate function:
This function is nothing but Additive Homomorphic operation performs on encrypted data.
1. Homomorphic operation performs using ciphertext c.
2. Suppose that cloud have two ciphertext c1,c2
3. c1 = $g^{m1}.r_1{}^n \ mod \ n^2$, c2 = $g^{m2}.r_2{}^n mod \ n^2$
4. c3 = c1 (H+) c2 = $g^{m1+m2}.(r_1 r_2)^n \ mod \ n^2$

It simply evaluate that paillier scheme proved the property of additive homomorphic encryption summation of two ciphertext will decrypt to the sum of their corresponding plaintext.

3.2 RSA Encryption Scheme [1] :

This scheme is multiplicative homomorphic encryption scheme, which can perform multiple times of multiplication operation.

Key Generation: KeyGen (p,q)
1. Generate two large prime numbers *p* and *q* randomly and independently of each other. Size of prime numbers is approximately equal such that its bit length is equal.
2. Compute n = pq and phi φ = (p-1)(q-1).
3. Chose an integer e, 1 < e < φ, such that gcd(e, φ) = 1.
4. Compute the secret exponent d, 1 < d < φ, such that ed = 1 .mod φ(n).
5. The public key: p_k = (e,n)
6. The secret key: s_k = d

Encryption: Enc (m, p_k)

1. Let m be a message to be encrypted where m € Z_n
2. Compute ciphertext as c = $m^e \ mod \ n$

Decryption: Dec(c, s_k)

1. Ciphertext c € Z_{n^2}
2. Compute message: m = $c^d \ mod \ n$

Evaluate function:
This function is nothing but Multiplicative Homomorphic operation performs on encrypted data.
1. Homomorphic operation perform using ciphertext c.
2. Suppose that cloud have two ciphertext c1,c2
3. $c_1 = m_1{}^e \ mod \ n$ and $c_2 = m_2{}^e \ mod \ n$
4. $c_3 = c_1$ (H*) $c_2 = (m_1.m_2)^e \ mod \ n$

It simply evaluates that RSA scheme proved the property of multiplicative homomorphic encryption. Multiplication of two ciphertext will decrypt to the multiply of their corresponding plaintext.

3.3 El Gamal Encryption Scheme [2] :

This scheme is multiplicative homomorphic encryption scheme, which can perform multiple times of multiplication operation.

Key Generation: KeyGen (1^K):

1. Generates an efficient description of a multiplicative cyclic group G of order q with generator g. See below for a

discussion on the required properties of this group.

2. Chooses a random x from {0,…..,q-1}.
3. Computes h = g^x.
4. Publishes h, along with the description of (G,q,g,h) as her Public key. And retains x as its Private key which must be kept secret.

Encryption: Enc (m, p_k):

1. Let m be a message to be encrypted where m € Z_n and public key = (G,q,g,h).
2. Compute ciphertext as c1 = $(g^r, m.h^r)$

Decryption: Dec (c, s_k):

1. Ciphertext c1 € Z_{n^2}
2. Compute message m.

Evaluate function:

This function is nothing but Multiplicative Homomorphic operation performs on encrypted data.

1. Homomorphic operation perform using ciphertext c
2. Suppose that cloud have two ciphertext c1,c2
3. c_1= $(g^{r1}, m1.h^{r1})$ and c_2 = $(g^{r2}, m2 * h^{r2})$
4. c_3=c_1 (H*) c_2= $(g^{r1+r2}, (m1 * m2).h^{r1+r2})$ = E (m1 * m2).

It simply evaluates that El-Gamal scheme proved the property of multiplicative homomorphic encryption. Multiplication of two ciphertext will decrypt to the multiply of their corresponding plaintext.

3.4. Boneh-Goh-Nissim Encryption Scheme [4] :

This scheme is additive and multiplicative homomorphic encryption scheme. This algorithm is closer to fully homomorphic encryption scheme. BGN schemes supports of an unlimited number of additions but at most one multiplication.

Key Generation: KeyGen (r)

1. Generates r∈ z^+
2. Compute λ(r) to obtain tuple (q1, q2, G, G1, e)
3. Calculate : n = p * q
4. Choose : g, u ∈ Random
5. Compute h = u^{q2}
6. Public key : Pk = (n, G, G1, e, g, h) Sk = q1.

Encryption: Enc (m, p_k)

1. Let m be a message to be encrypted where m ∈ Z_n.
2. Chose random number r.
3. Compute ciphertext as c =$(g^m.h^r)$.

Decryption: Dec (c, s_k)

1. Input ciphertext c and secret key s_k .

Compute message m = dilog (c^{q1}). $mod\ g^{q1}$

3.5 Gentry's Fully Homomorphic Encryption Scheme (Using Ideal Lattices) [11]

This scheme is in terms of rings and ideals. In this scheme, fix Ring R, that is set appropriately with respect to the security

parameter λ, and two relatively prime ideals I and J. I.e. I + J = R. We fix a basis B_I of I, and an algorithm IGen (R,B_I) that outputs public and secret bases B_J^{pk} and B_J^{sk} of J.

Key Generation: KeyGen(R, B_I):

1. Input for KeyGen is Ring R and basis B_I
2. Compute the basis : B_J^{pk} , B_J^{sk} = IGen (R, B_I)
3. Compute the public key and private key pk = (R, B_I, B_J^{pk},samp) , sk= B_J^{sk}

Encryption: **Enc(m, pk):**

1. Let m be a message to be encrypted where m is message and pk is public key
2. Compute Encrypt (pk , m) : c' = samp (m, R, B_I, B_J^{pk})
3. Compute ciphertext as c = c' mod B_J^{pk}

Decryption: Dec(c, sk):

1. Decryption of ciphertext c with input of secret key sk:
2. Compute message m = Decrypt(sk,c) : m = [c mod B_J^{sk}] mod B_I

Evaluate function:

This function is nothing but some Homomorphic operation performs on encrypted data.

1. Homomorphic operation perform using public key pk , circuit C and ciphertext c
2. Evaluate (pk,C,c) : takes input the public key pk ,circuit C
3. Add(pk,c1,c2) → output : c1 (H+) c2 mod B_J^{pk}
4. Mult(pk,c1,c2) → output : c1 (H*) c2 mod B_J^{pk}

3.6 DGHV Encryption Scheme [6][9] :

This scheme is fully homomorphic encryption scheme. So this encryption scheme performs both operation addition as well as multiplication.

Key Generation: **KeyGen:**

1. The key is an odd integer, chosen from some interval p ∈ (2^{n-1}, 2^n). Here p is secret key.

Encryption: Enc (m, p):

1. m is the bit plaintext is 0 or 1
2. Generate q is a large random number and r is small random number .
3. Calculate ciphertext, c = pq + 2r + m.

Decryption: Dec (p, c):

1. Decryption of ciphertext c with input of secret key p.
2. Compute message m = (c mod p) mod 2. It is easy to see that 2r is smaller than p in absolute value.

Evaluate function:

This function is nothing but some Homomorphic operation performs on encrypted data.

1. Suppose that we have two ciphers c1 and c2 .
2. c1 = (q1*p) + (2*r1) + m1 and c2 = (q2*p) + (2*r2) + m2
3. c3 = c1 (H+) c2 = (q1 + q2) * p + (2*(r1+r2)) + (m1+m2)
4. c3 = c1 (H*) c2 = q12*p + 2*(2*r1*r2 + r1*m2 + r2*m1) + m1+m2

From above equations, we see that we can obtain the encryption of m1+m2 by computing c1 (H+) c2 .similarly encryption of m1.m2 can be obtain by computing c1 (H*) c2.

3.7 Brakrski-Vaikunthan Scheme [7] :

BV scheme is the simplest and more powerful fully homomorphic encryption scheme. The whole construction is based on LWE problems

The LWE assumption states that if s € Z_q^n is an n dimensional (secret) vector.

In RLWE problem for security parameter λ, let $f(x) = x^d + 1$ .where d = d(λ) is a power of 2. Ring R = Z[x]/ f(x) and R_q = R/q R. where q be an integer.

Key Generation: KeyGen (l^λ):

1. Select sample $S_k = s \leftarrow R_q$ randomly
2. Polynomial, small coefficients ∈ error distribution.

Encryption: Enc (S_k , m ∈ R_q):

1. To encrypt a bit m choose a random a ∈ Z_q^n , a "noise" e ∈ Z_q
2. Compute: c = (a, a_s = 2e + m)

Decryption: Dec (c = (a, b), S_k):

1. Select sample b ← R_q randomly
2. Compute m' = b*a_s ∈ R_q
3. Compute m = m' mod 2

The key distribution of this scheme is the technique of managing noise so that it increases linearly with the multiplicative level instead of exponentially.

IV. LIMITATION OF EXISTING HOMOMOMORPHIC ENCRYPTION SCHEMES

In Homomorphic Encryption Scheme, Some Limitation are there:

1. Homomorphic Encryption not provides verifiable computing.
2. Performance is often a disadvantage of Homomorphic Encryption scheme.

Limitation of some existing schemes or methods of homomorphic encryption is given below:

1. Paillier Scheme: Negative value can't be decrypted in this scheme. Other disadvantage of the scheme has high computational complexity.
2. RSA Scheme: Limitation of RSA, It's mainly depending on difficulty of factoring large prime number. Other main issue of RSA is not semantically secure.
3. El-Gamal Encryption scheme: In this ciphertext is twice as compare to the plaintext
4. BGN scheme: It is limited in the size of message space due to the need to compute discrete logarithms during decryption of ciphertext.
5. Gentry's FHE with Ideal lattice: noise is increasing rapidly if addition operation done using this formula, noise is increasing linearly otherwise exponentially.
6. DGHV scheme: In this scheme, new ciphertext with noise roughly twice larger than in the original ciphertext.
7. Brakerski-Vaikuntanathan scheme: In this scheme noise is increasing linearly.

V. CONCLUSION

In cloud computing Encryption offers better performance, ease of development, but the security of data is challenge for the cloud provider. Therefore, the concept of homomorphic Encryption Scheme is useful for security of data while data are stored or rest in data cloud. This paper presents a survey of all important homomorphic encryption techniques and highlights scenarios where homomorphic encryption could be an appropriate solution for cloud computing security.

VI. REFERENCES

[1] R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM 21, Feb 2012, 120–126.
[2] T. El Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in Proceedings of CRYPTO 84 on Advances in Cryptology. New York, NY, USA: Springer-Verlag New York, Inc., 1985, pp. 10–18.
[3] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes", in Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques, ser. EUROCRYPT' 99. Berlin, Heidelberg: Springer-Verlag, 1999, pp. 223–238.
[4] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in Proceedings of the Second

International Conference on Theory of Cryptography, ser. TCC'05. Berlin, Heidelberg: Springer- Verlag, 2005, pp. 325–341.

[5] Michael Glas and Paul Andres"Achieving the Cloud Computing Vision", An Oracle White Paper in Enterprise Architecture-October 2010

[6] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in EUROCRYPT, 2010, pp. 24–43.

[7] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) lwe," in FOCS, 2011, pp. 97–106.

[8] Sean Carlin, Kevin Curran, "Cloud Computing Technologies", International Journal of Cloud Computing and Services Science (IJ-CLOSER) Vol.1, No.2, June 2012, pp. 59~65.

[9] Jian Li, Danjie Song, Sicong Chen, Xiaofeng Lu,"A simple Fully homomorphic encryption Scheme available in cloud computing", Proceedings of IEEE CCIS -2012.

[10] Maha TEBAA and Said EL HAJII, "Secure cloud computing through homomorphic encryption", International Journal of Advancements in Computing Technology-December 2013.

[11] Darko Hrestack and Stjepan Picek, "Homomorphic Encryption in the Cloud", Institute of Electrical and Electronics Engineers (IEEE)- MIPRO -May 2014.

