Volume 4, Issue 7, July 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Survey of Fully Homomorphic Encryption and Its Potential to Cloud Computing Security

Sweta Agrawal*, Aakanksha Choubey

Department of Computer Science & Technology, CSVTU,

Bhilai (C.G), India

Abstract— Cloud computing is considered as one of the most promising innovation in the computing world over a decade. It is an emerging computing technology where applications and all the services are provided through Internet. Cloud Computing is definitely money saver for enterprises, however its usage is still hindered by security concerns.

The paper presents an overview of security issues in cloud computing and use of fully homomorphic encryption scheme to provide solution for this dilemma. The cloud computing security based on homomorphic encryption, is a new concept of security which enables providing results of calculations on encrypted data without knowing the raw data on which the calculation was carried out, with respect of the data confidentiality.

Keywords— Cloud Computing, Cloud Security, Cryptography, Data Security, Homomorphic Encryption, RSA

I. INTRODUCTION

Cloud computing, the use of a large number of connected computers, provides benefits in terms of increased storage, compute-capacity, flexibility and cost reduction. A cloud service has three different characteristic that differentiate it from traditional host, first one is it is sold on-demand, second is its elasticity property that enables user to use its service as per their requirement and third is it is fully controlled by the provider. The security risks are involved with moving sensitive data, such as medical or financial records, into the cloud which prevent a faster adoption of cloud services. Strong encryption algorithms to protect data do exist, however they prevent the cloud from performing meaningful computational tasks on this data, and hence greatly reduce its power.

Homomorphic encryption schemes do, in theory, provide a solution for this dilemma: Such schemes allow for functions to evaluate encrypted data, the result of which is still encrypted data, but can be decrypted back into the result of the (logically) same function applied to the plain data.

Section-2 describes definition, services, models and security issues of cloud computing. Section-3 describes fully homomorphic encryption and its potential to cloud computing. Section-4 describes fully homomorphic encryption and its literature survey. Section-5 describes implementation methodology. Section-6 concludes the paper.

II. CLOUD COMPUTING

Cloud Computing defines a new class of network based computing that takes place over Internet. Cloud computing is the delivery of computing services over the Internet. Cloud services allow individuals and businesses to use software and hardware that are managed by third parties at remote locations. Examples of cloud services include online file storage, social networking sites, webmail, and online business applications.

A. Definition

The following definition of cloud computing has been developed by the U.S. National Institute of Standard And Technology[1]:

"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 provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models".

B. Cloud Computing Services

In cloud computing, everything is delivered as a service from testing and security to collaboration and meta modeling [2]. Today there are three main service models, which are agreed and defined by NIST document [3]:

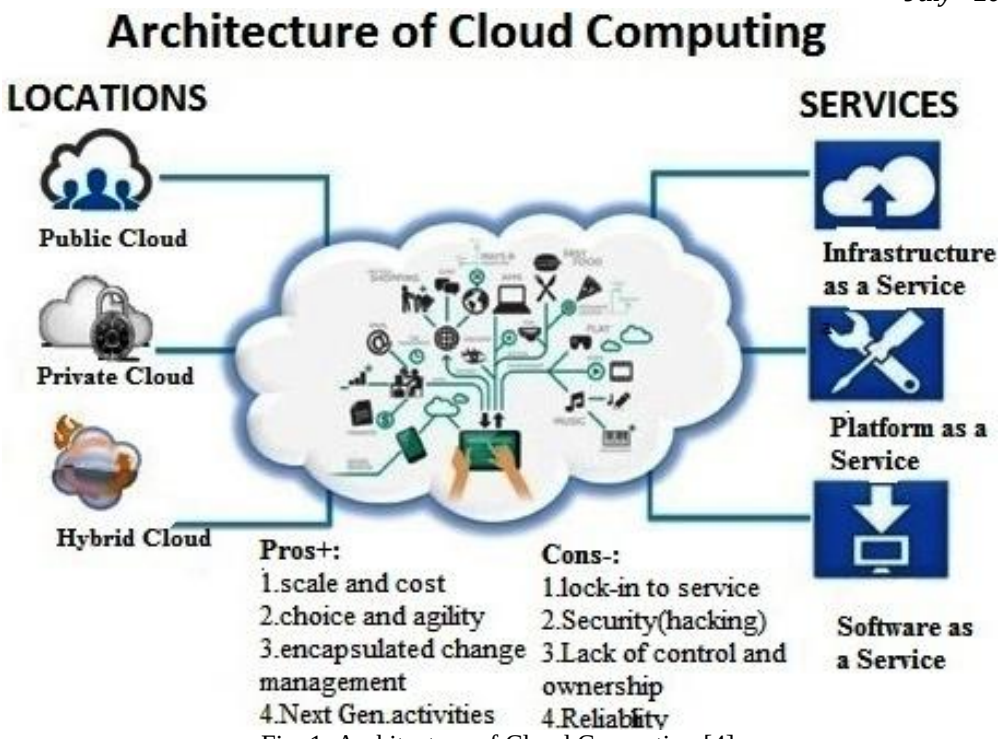

Fig. 1 Architecture of Cloud Computing [4]

1) *SaaS (end users):* Acronym for software as a service is also known as On-demand Service. It is an application that can be accessed from anywhere, provided there is internet connectivity. We can access this cloud hosted application without any additional hardware or software, thus can save cost and space. They can provide Security features such as SSL encryption, a Cryptographic protocol.
Example: G-mail, Yahoo-mail, Hotmail etc.

2) *PaaS (Application Developer):* It is Acronym for platform as a service. It provides an application programming interface (API) and takes care of implementation. In PaaS model, cloud providers deliver a computing platform and solution stack that includes OS, Programming language execution environment, database and web server. It is a platform for developers to write and create their own application that allow rapid development at low cost.
Example: Salesforce.com, Window Azure etc.

3) *IaaS (Network Architect):* Acronym for infrastructure as a service is also known as hardware as a service. It provides computing power that can be rented for a limited period of time. It allows existing applications to be run on a cloud supplier's hardware. Host provides server, client and network. Cloud providers offer computers as physical or more often as Virtual Machine raw storage, firewalls load balances and networks.
Example: Amazon EC2, SQL Azure, VMware etc.

C. Models of Cloud Computing

Each company chooses a deployment model for a cloud computing solution based on their specific business, operations and technical requirements. There are four types of model as described:

1) *Public Cloud:* Computing Infrastructure here is hosted by cloud Vendor at Vendor premises and can be shared by various organizations.

   Example: Amazon, Google, Microsoft, Salesforce etc.
2) *Private Cloud:* Computing Infrastructure here is dedicated to a private organization and is not shared with other organization. It is more expensive and more secure when compare to public cloud.
Example: HP data center, IBM, SUN, ORACLE, 3-tera etc.

3) *Hybrid Cloud:* Organization may host only critical applications on private cloud since there is relatively less security concern on public cloud. Hence hybrid cloud involves the usage of both public and private cloud together.
Example: Amazon, Google, Microsoft, Salesforce etc.

4) *Community Cloud*: The Cloud infrastructure is shared by several organizations and supports a specific community that has communal concerns. It may be managed by the organizations or a third party, and may exist on premise or off premise.

   Example: Emission, security requirements, policy, and compliance considerations.

D. Security Issue in Cloud Computing:

An organization uses the cloud in a variety of different service models and deployment models. Moving your infrastructure to the cloud comes with many benefits such as elasticity, scalability and reduced cost of cloud. But it also come with the challenges especially data security and regulatory compliance. There are a number of security issues/concerns associated with cloud computing, these issues fall into two broad categories: security issues faced by cloud providers and security issues faced by their customers.

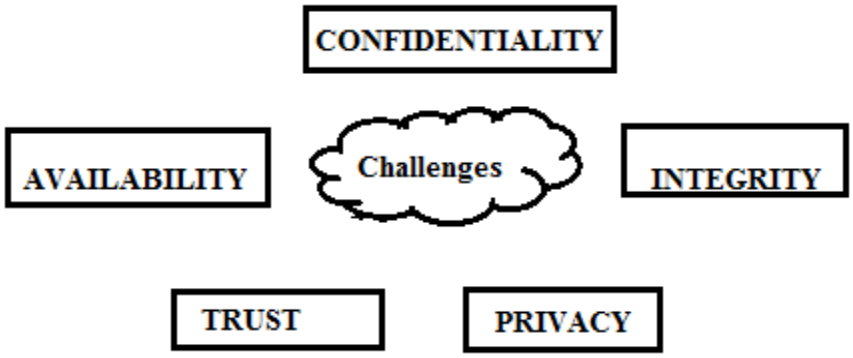

Fig. 2 Challenges of Cloud Computing

The common security issues of cloud computing can be divided into five main categories:

1) *Availability:* This border on data being available whenever it is required. This is one of the prime concerns of mission and safety critical organizations.

2) *Trust:* When two parties are involved in a transaction then the trust can be described as follows: An entity A is said to trust another entity B when entity A believes that the entity B will behave exactly as expected and required. This comprises cloud service provider to provide sufficient security policy that guarantees the efficient use of activities.

3) *Confidentiality:* Data confidentiality in the cloud means isolating the data of individual users from one another. It refers to trusting that specific applications or processes will maintain and handle the user's data in a secure manner.

4) *Privacy:* Privacy refers to the willingness of a user to control the disclosure of private information, (authentication, authorization and access control *i.e.* AAA) encrypted data communication, and user identity management.

5) *Integrity:* Integrity is associated with data, software and hardware and it monitor that these can only be manipulated by authorised persons and by authorised processes. Cloud service provider should also provide security against insider attacks on data.

III. HOMOMORPHIC ENCRYPTION AND ITS POTENTIAL TO CLOUD COMPUTING

Cloud computing security issues can be resolved to a great extent if we can make sure that the data once outsourced to the cloud is never interpreted by any other user as well as the cloud service provider itself at any stage even during processing on the data in the cloud. To achieve this we require encryption schemes which can provide secure encryption of data along with the ability to compute the data without decrypting it at any stage. For computing blindfolded fully homomorphic encryption schemes are needed which are practical to implement with satisfactory efficiency.

To achieve secure data transaction in cloud, suitable cryptography method is used. The data owner must encrypt the file and then store the file to the cloud. If a third person downloads the file, he/she may view the record if he/she had the key which is used to decrypt the encrypted file. Sometimes this may be failure due to the technology development and the hackers.

The problem that arises now is that while data can be sent to and from a cloud provider's data centre in an encrypted form; we can't do any work on it without decrypting it. homomorphic encryption is a form of encryption which allows specific types of computations to be carried out on cipher text and obtain an encrypted result which when decrypted matches the result of operations performed on the plaintext.

The aim of homomorphic cryptography is to ensure privacy of data in communication and storage processes, such as the ability to delegate computations to untrusted parties. Fully Homomorphic Encryption combines security with usability. It can help preserve customer privacy while outsourcing various kinds of computation to the cloud, besides storage. There are two aspects of the computation considered: the data itself (confidentiality) and the function to be computed on this data (circuit privacy).

IV. HISTORY OF THE HOMOMORPHIC ENCRYPTION

A. Homomorphic Encryption

In the beginning, there was symmetric encryption, in which sender and receiver used same key to encrypt and decrypt the message. Then there began the development of popular asymmetric encryption to achieve secure cryptosystem. Their idea was to enable secure message exchange between the parties without ever having to meet in reality to agree on a common secret. However, till now only encryption was enabled through given algorithm. Requirements were aroused to build homomorphism so as to allow owners and other authorized people to perform calculation with the data without decrypting it. Homomorphic Encryption systems were developed then, which were used to perform operations on encrypted data without knowing the private key (without decryption), that is the client is the only holder of the secret key. The idea of performing simple computation on encrypted messages was first proposed by Rivest, Adleman and Dertozous in 1978.

Definition: An encryption is homomorphic, if from Enc(a) and Enc(b) it is possible to compute Enc(f (a, b)), where f can be: +, ×, $\oplus$ operation, without using the private key.

A homomorphic encryption scheme is composed of the following four algorithms:

a) Key generation: This involves producing public key pk and secret key sk for security parameters.
b) Encryption: In this, we input plain text m, where $m \epsilon\{0,1\}^*$, we encrypt the plaintext using public key pk to get cipher text c.
c) Decryption: Applying decryption on private key sk and cipher text c, we get original plain text m.
d) Evaluate: Inputting public key pk, t input circuits and C a set of cipher text $c = (c_1, c_{2\ldots}c_t)$,we can get the output result c^* =Evaluate (pk,C,c^-)

This should meet given conditions:

$$dec(sk, c^*) = C(m_1, m_2 \ldots m_t)$$

There are two types of Homomorphic Encryption: Somewhat Homomorphic Encryption (SHE) and Fully Homomorphic Encryption (FHE).

Each type differs in the number of operations that can be performed on encrypted data. FHE allows for an unlimited, arbitrary number of computations (both addition and multiplication) to be performed on encrypted data. SHE cryptosystems support a limited number of operations (i.e., any amount of addition, but only one multiplication) and are faster and more compact then FHE cryptosystem.

B. Somewhat Homomorphic Encryption (SHE)

In 1978 Ronald Rivest, Leonard Adleman and Michael Dertouzos suggested for the first time the concept of homomorphic encryption[5]. Since then, little progress has been made for 30 years. The encryption system of Shafi Goldwasser and Silvio Micali proposed in 1982, was a provable security encryption scheme which reached a remarkable level of safety, it was an additive homomorphic encryption, but it can encrypt only a single bit. In the same concept in 1999 Pascal Paillier also proposed a provable security encryption system, which was also an additive homomorphic encryption. Few years later, in 2005, Dan Boneh, Eu-Jin Goh and Kobi Nissim[6] invented a system of provable security encryption, with which we can perform an unlimited number of additions but only one multiplication.

According to the operations that allow assessing on raw data, homomorphic encryption can be differentiated into:

a) Additive Homomorphic Encryption
b) Multiplicative Homomorphic Encryption

1) Additive Homomorphic Encryption:

An additively homomorphic scheme is one with a ciphertext operation that results in the sum of the plaintexts. That is,

Encrypt(m1) +Encrypt(m2) = Encrypt(m1 + m2)

where the decryption of both sides yields the sum of the plaintexts. There are many known additive schemes including El-Gamal encryption, the Goldwasser-Micali, Benaloh and Paillier schemes, which compute addition modulo sum of a number. The Boneh-Goh-Nissim scheme also allows for unlimited additions and a single multiplication.

Pailler cryptosystem realizes the property of additive homomorphic encryption. An application of an additive homomorphic encryption is electronic voting: Each vote is encrypted but only the "sum" is decrypted
In an additive homomorphic cryptosystem given only the public-key and the plain text m1 and m2, one can compute the encryption of m1+ m2 [7].

Algorithm
The Pailler scheme works as follows:

Key generation

- 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, i.e., p,q ϵ 1||{0,1}$^{s-1}$for security parameter
- Compute n=pq and λ=lcm(p-1,q-1)
- Select random integer g where gεZ_{n2}^{*}
- Ensure n divides the order of g by checking the existence of the following modular multiplicative inverse,

  μ=(L(g $^{\lambda}$ mod n^2))$^{-1}$ mod n

where function L is defined as L(u) = u-1/n.
If using p, q of equivalent length, a simpler variant of the above key generation steps would be to set g=n+1, λ =φ(n) and μ= φ(n)$^{-1}$, where φ(n)=(p-1)(q-1).

Encryption
1. Let m be a message to be encrypted where m ϵZ_n
2. Select random integer ϒ where ϒ ϵZ_n^{*}
3. Compute cipher text as: c=$g^m . r^n \ mod\ n^2$

Decryption
1. Cipher text cεZ_{n2}^{*}
2. Compute message: m=L($c^{\lambda} mod\ n^2$). μ mod n

2) Multiplicative Homomorphic Encryption:
A multiplicatively homomorphic scheme is one that has an operation on two cipher texts that results in the product of the plaintexts. That is,

Encrypt(m1) * Encrypt(m2) = Encrypt(m1 * m2)

where the decryption of both sides yields the product of the plaintexts.
The most famous multiplicatively homomorphic scheme is RSA encryption.The original El-Gamal scheme is also multiplicatively homomorphic.

Algorithm
RSA involves a public key and a private key**.** The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

Key generation
1. Choose two distinct prime numbers p and q.
   For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length.
2. Compute n = pq.
   n is used as the modulus for both the public and private keys.
3. Choose an integer e such that 1 < e < φ(n) and gcd(e, φ(n)) = 1; i.e., e and φ(n) are co-prime.
4. Determine d as d ≡ e^{-1} (mod φ(n)), i.e., d is the multiplicative inverse of e (modulo φ(n))

Encryption
Cipher text is computed by using sender public key(n,e):
C=m^e(mod n)

Decryption
Plain text m can be recovered from c using private key d by computing as
M=c^d (mod n)

C. Fully Homomorphic Encryption

A cryptosystem which supports both addition and multiplication is known as fully homomorphic encryption (FHE) and is far more powerful. Using such a scheme, any circuit can be evaluated homomorphically, and effectively allowing the construction of programs which may be run on encryptions of their inputs to produce an encryption of their output. Since such a program never decrypts its input, it can be run by an untrusted party without revealing its inputs and internal state. The existence of an efficient and fully homomorphic cryptosystem will have great practical implications in the outsourcing of private computations, especially, in the context of cloud computing.

1) Gentry's Scheme:

In 2009, Gentry came with practically first fully homomorphic encryption scheme [8]. This was a landmark event in cryptographic research that eventually had huge implication in privacy and security. His proposed scheme consists of several steps:

a) It constructs a somewhat homomorphic scheme that supports evaluating low-degree polynomials on the encrypted data.
b) It squashes the decryption procedure so that it can be expressed as a low-degree polynomial which is supported by the scheme i.e. Modulus reduction procedure.
c) It applies a bootstrapping transformation to obtain a fully homomorphic scheme [9].

2) Smart and Vercauteren Scheme:

At 2010 Smart and Vercauteren made the first attempt to implement Gentry's scheme using a variant based on principal ideal lattices and requiring that the determinant of the lattice be a prime number [10].However the authors could not obtain a bootstrappable scheme because that would have required a lattice dimension of at least n = 227, whereas due to the prime determinant requirement they could not generate keys for dimensions n > 2048, which is essential for security purposes. This implied that Gentry's blueprint was not yet practical.

3) Gentry and Halevi Scheme:

In 2010, Gentry and Halevi presented a novel implementation approach for the variant of Smart and Vercauteren proposition which had a greatly improved key generation phase [11]. In particular, key generation (for cyclometric fields) is essentially an application of a Discrete Fourier Transform (DFT), followed by a small quantum of computation, and then application of the inverse transform. The key generation method of Gentry and Halevi is fast.

4) Stehle and Steinfield Scheme:

In 2010, Stehle and Steinfield improved Gentry's fully homomorphic scheme and obtained a faster fully homomorphic scheme with O(n3.5) bits complexity per elementary binary addition or multiplication gate [12].However, the hardness assumption of the security of the scheme is stronger than that of Gentry's scheme. The finer analysis of Stehle and Steinfield for sparse subset sum problem (SSSP) takes into account the complexity of approximate SVP.

5) Brakerski and Vaikuntanathan Scheme:

In 2011, Brakereski and Vaikuntanahan have constructed a fully homomorphic encryption scheme based on ring learning with error [13-15]. This scheme inherits the simplicity and efficiency of Gentry's scheme, as well as the worst case relation to ideal lattices. Moreover, the scheme have key dependent message security, hence it can securely encrypt polynomial functions of its own secret key. However this becomes a major drawback with regard to the usability and the efficiency of the schemes. The size of the public key can be made independent of the circuit depth if the somewhat homomorphic scheme can securely encrypt its own secret key. These scheme transform the proposed scheme into a fully homomorphic encryption scheme following Gentry's blueprint of squashing and bootstrapping.

6) Boneh & Freeman Scheme:

In 2011, Boneh and Freeman propose a linearly homomorphic signature scheme that authenticates vector subspaces of a given ambient space [16]. First, the scheme is the first of its kind that enables authentication of vectors over binary fields; previous schemes could not authenticate vectors with large or growing coefficients. Second, the scheme that is based on the problem of finding short vectors in integer lattices, and therefore, it enjoys the worst-case security guarantee that is common to lattice-based cryptosystems. The scheme can be used to authenticate linear transformations of signed data, such as those arising when computing mean and Fourier transform or in networks that use network coding.

7) Chunsheng Scheme:

In 2012, Chunsheng proposed a modification of the fully homomorphic encryption scheme of Smart and Vercauteren [17].These scheme applied a self-loop bootstrappable technique so that the security of the modified scheme only depends on the hardness of the polynomial cosset problem and does not require any assumption of the sparse subset problem. the security of the improved fully homomorphic encryption scheme in this work is based on use of three mathematical approaches: (i)

hardness of factoring integer problem, (ii) solving Diophantine equation problem, and (iii) finding approximate greatest common divisor problem.

V. HOMOMORPHIC SCHEME AND ITS IMPLEMENTATION

In contrast to conventional methodology, homomorphic encryption not only allows storage privacy but also allow computation on encrypted data, which would benefit various researches in medical, marketing and other application.
Homomorphic encryption is expected to play an important part in cloud computing, allowing companies to store encrypted data in a public cloud and take advantage of the cloud provider's analytic services. Suppose we want to do computation on$\{m_1, m_2, .. m_n\}$ but want to utilize cloud server for the computation. Since we do not want to give cloud server access to data itself, homomorphic encryption method proves to be best option.

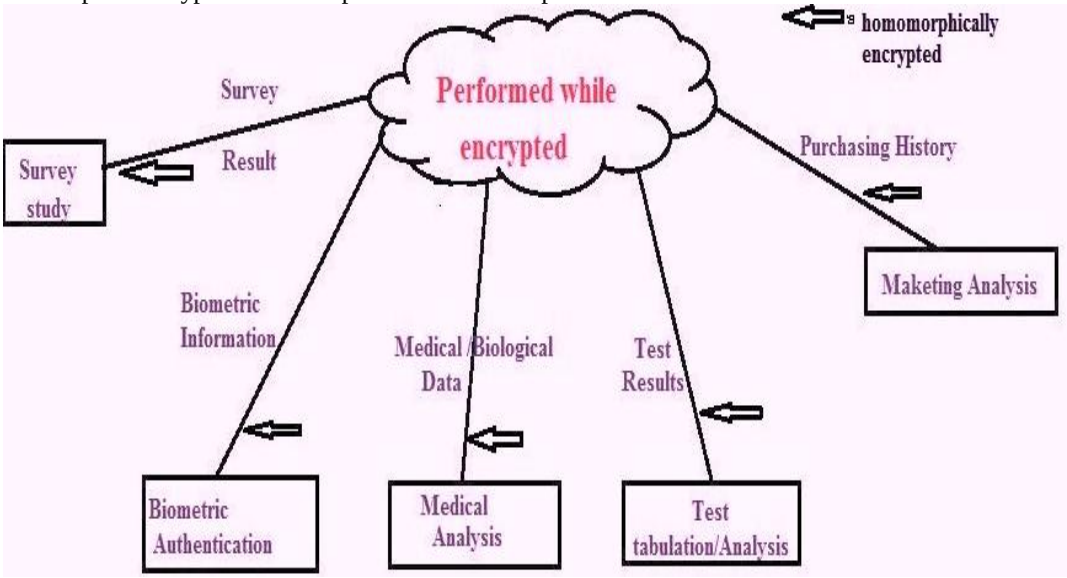

Fig. 3 Implementation of homomorphic encryption in different Application

The methodology adapted by these areas is discussed below:

Mechanism:

1. Client sends an encrypted data to cloud
   - Eg: let two encrypted number be a and b
2. Client sends request to Cloud for calculating function
   - i.e. f(a,b)
3. Client and Cloud communicate through a cryptosystem based on fully homomorphic encryption.
4. Cloud stores encrypted data
5. Cloud calculates the result of request sent by the client without knowing actual number
   - i.e. f(a,b) is calculated
6. Cloud then compute f(Enc(a),Enc(b)) without knowing a and b
7. Client decrypts f(Enc(a),Enc(b)) using its private key.

Here is a very simple example of how a homomorphic encryption scheme might work in cloud computing [18]:
Example: Suppose Business has a *Very Important Data Set* (VIDS) that consists of two numbers 5 and 10. To encrypt the data set, Business multiplies each element in the set by 2, creating a new set whose members are 10 and 20.

- Business XYZ sends the encrypted VIDS set to the cloud for safe storage. A few months later, the government contacts Business XYZ and requests the sum of VIDS elements.
- Business XYZ is very busy, so it asks the cloud provider to perform the operation. The cloud provider, who only has access to the encrypted data set, finds the sum of 10 + 20 and returns the answer 30.
- Business XYZ decrypts the cloud provider's reply and provides the government with the decrypted answer, 15.

Thus, fully homomorphic encryption allow outsourcing calculation on our confidential data. However here are some practical limitations of using FHE. First One is performance, which can be billions of times slower than the "somewhat" or "partial" approach. Another limitation to FHE is its computational complexity.
There are only few specific application which can make use of homomorphic encryption in very practical. The world's first commercial implementation is homomorphic key management, a "Partially Homomorphic" approach. Homomorphic Key

Management is focused on the specific problem of encrypting the encryption keys themselves and keeping them safe and secret in the cloud. Typically voting system such as, Helios voting is used. However these work through partially homomorphic encryption.

VI. CONCLUSIONS

With the more and more interest of the IT community in the Cloud, new security problems are found. The protection of the data stored in the Cloud face new vulnerabilities. Best solutions for remote data protection remain cryptography. This article presents the new techniques that provide security to the private data, and also provide mechanisms for searching or processing encrypted data. Researches are being done on the given scheme however, to ensure its practicality we must focus more on the minimization of its computational complexity.

REFERENCES

[1] Grance, P. Mell and T. ""The NIST Definition of Cloud Computing,"." National Institute of Standards and Technology, U. S. Department of Commerce, 2011.

[2] Correia, F. Rocha and M. "Lucy in the Sky without Diamonds: Stealing Confidential Data in the Cloud." IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops, 2011: 129-134.

[3] Y. Shen, W. Cui, Q. Li and Y. Shi. "Hybrid Fragmentation to Preserve Data Privacy for SaaS." EighWeb Information Systems and Applications Conference (WISA), 2011: 3 - 6.

[4] HCL TECHNOLOGIES: http://www.hcltech.com/blogs/transformation-through-technology/rise-cloud.

[5] R. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public key cryptosystems." Communications of the ACM, 1999.

[6] Boneh, D. & Freeman,D .M . "Linearly Homomorphic Signatures over Binary Fields and New Tools for Lattice-Based Signatures." Public key Cryptography, 2011.

[7] Suresh, K.Revana. "Ensuring Data Security Using Homomorphic Encryption In Cloud Computing." 2014.

[8] S. Sobitha Ahila, Dr.K.L.Shunmuganathan. "State Of Art in Homomorphic Encryption Schemes." Int. Journal of Engineering Research and Applications, 2014.

[9] Lattices, Fully Homomorphic Encryption Using Ideal. "Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC'09)." ACM Press, New York, NY, USA., 2009.

[10] Smart, N. & Vercauteren. "Fully Homomorphic SIMD Operations." Design Codes and Cryptography, Springer, USA, 2012.

[11] Gentry, C. & Halevi, S. "Implementing Gentry's Fully-Homomorphic Encryption Scheme. In: Advances in Cryptology." Proceedings of EUROCRYPT'11, Lecture Note in Computer Science, 2011.

[12] Stehle, D. & Steinfeld, R. "Faster Fully Homomorphic Encryption." In: Advances in Cryptology – Proceedings of ASIACRYPT'10,, 2010.

[13] Brakerski, Z. & Vaikuntanathan, V. "Efficient Fully Homomorphic Encryption from (Standard) LWE." Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS'11). ACM Press, New York, NY, USA.

[14] Brakerski, Z. & Vaikuntanathan, V. "Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages." In Advances in Cryptology- Proceedings of CRYPTO'11, by Springer-Verlag, 505-524. 2011.

[15] Brakerski, Z., Gentry, C., & Vaikuntanathan. "Fully Homomorphic Encryption without Bootstrapping." Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS'12), 2011: 309-325.

[16] Vaikuntanathan, V. "Computing Blindfolded: New Developments in Fully Homomorphic Encryption." Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS'11),, 2011.

[17] Chun-sheng, Gu. "Attack on Fully Homomorphic Encryption over the Integers." International Journal of Information & Network Security (IJINS), 2012.

[18] Rouse, M. (2011, August). Search Security. Retrieved from ttp://searchsecurity.techtarget.com/definition/homomorphic-encryption

