A Secured Homomorphic Encryption Technique in Cloud Computing

Nitin Kamble
INURTURE, Bangalore

Pragati Hiwarkar, PhD
INURTURE, Bangalore

Monali Bachhav
INURTURE, Bangalore

ABSTRACT
Cloud computing security challenges to many researchers. Priority was to focus on security which is the biggest concern of organizations moving to the cloud. Cloud computing help in costs optimization, easy maintenance and re-provisioning of resources, and so the increased profits. The adoption of Cloud Computing applies only if the security is ensured. How to guaranty a better data security and also how can we keep the client private information confidential? There are two major queries that present a challenge to Cloud computing providers.

When the data transferred to the Cloud we use standard encryption methods to secure the operations and the storage of the data. For processing data located on a remote server, the Cloud providers need to access the raw data. In this paper we are proposing a method to execute operations on encrypted data without decrypting them. It will yield the same results after calculations as if we have worked directly on the raw data.

Keywords
Cloud computing, homomorphic encryption, Ciphertext, Data security

1. INTRODUCTION
The rapid abundance of Smartphone technology in urban communities has enabled mobile users to utilize context aware services on their devices. Service providers take benefit of this vibrant and ever-growing technology landscape by proposing innovative context-dependent services for mobile subscribers. Location based Services (LBS), for example, are used by millions of mobile subscribers every day to obtain location-specific information [1].

Cloud computing as a concept is the result of the natural evolution of our everyday approach to using Technology delivered via the Internet. Cloud computing came into the forefront as a result of advances in virtualization (e.g. VMware), distributed computing with server clusters (e.g. Google) and increase in the availability of broadband Internet access. Business leaders illustrate cloud computing simply as the delivery of applications or IT services, which are offered by an intermediary over the Internet (Microsoft, IBM). The recent global economic recession served as a booster for interest in cloud computing technologies as organizations sought for ways to reduce their IT finances, while in harmony with performance and profits. The cloud computing buzz began in 2006 with the Launch of Amazon EC2, gaining footing in 2007.

The National Institute of Standards and Technology defines cloud computing as follows: “Cloud computing is a model for facilitating convenient, on demand network access to a common group of configurable computing resources (e.g., networks, servers, storage, applications, and services) can be swiftly provisioned and released with minimal management effort or supplier relations.”

Cloud computing is currently characterized by having an on demand access to elastic resources via a lease model. According to reports of CSA (Cloud Security Alliance), the 13 security domains [2] and the 7 top threats [3] on cloud computing were defined as follows;

1) Security domains in cloud computing:
- Cloud Computing Architectural Framework
- Governance and Enterprise Risk Management
- Legal and Electronic Discovery
- Compliance and Audit
- Information Lifecycle Management
- Portability and Interoperability
- Traditional Security, Business Continuity, and Disaster Recovery
- Data Center Operations
- Incident Response, Notification, and Remediation
- Application Security
- Encryption and Key Management
- Identity and Access Management
- Virtualization

2) Top threats in cloud computing security:
- Abuse and Nefarious Use of Cloud Computing
- Insecure Application Programming Interfaces
- Malicious Insiders
- Shared Technology Vulnerabilities
- Data Loss or Leakage
- Account, Service & Traffic Hijacking
- Unknown Risk Profile

"Homomorphic" is an adjective which describes a property of an encryption method. That property, in simple terms, is the capability to perform computations on the cipher text without decrypting it first. The basic concept was to encrypt the data before sending to the Cloud provider. But, this one will have to decrypt each time he has to work on. The client will need to provide the private key to the server to decrypt the data before execute the calculations requisite, which might concern the confidentiality of data stored in the Cloud. The Homomorphic Encryption method is able to perform operations of encrypted data without decrypting them [4].

Homomorphic encryption is the conversion of data into ciphertext that can be analyzed and worked with as if it were still in its unique state. It allows complex mathematical operations to be performed on encrypted data without compromising the encryption. In arithmetic, homomorphic depicts the conversion of one data set into another while preserving relationships. The idiom is resultant of the Greek words "same structure." Since the information in a homomorphic encryption scheme retains the similar structure,

identical mathematical operations whether they are performed on encrypted or decrypted data will yield equivalent results.

Homomorphic encryption is likely to play vital role in cloud computing, allowing companies to store encrypted data in a public cloud and take advantage of the cloud provider's analytic services.

Here is a straightforward example of how a homomorphic encryption scheme might work in cloud computing:

- Business XYZ has a *very important data set* (VIDS) that consists of records 5 and 10. To encrypt the data, company XYZ multiplies each element by 2, creating new members as 10 and 20.
- Company XYZ sends the encrypted VIDS set to the cloud for secure storeroom. A few months later, the administration contacts company XYZ and requests the sum of VIDS elements.
- Company XYZ is very full of activity, so it asks the cloud provider to perform the action. The cloud provider, who only has admittance to the encrypted data, get the sum of 10 + 20 and returns the answer 30.
- Company XYZ decrypts the cloud provider's reply and provides the administration with the decrypted answer, 15.

The spotlight is on the application of Homomorphic Encryption Technique on the Cloud Computing security, particularly to perform the calculations of confidential data encrypted without decrypting them.

All secure communication is limited to those that have that secret key. Of course, here we are simplifying the situation and assume that key and methods of encryption are appropriately chosen. If some third party, Eve, intercepts encrypted message it should be practically infeasible for her to decrypt it. In order to be able to encrypt and afterwards decrypt the data, Alice and Bob must use the same key. If we add an additional condition that the key needs to be changed from time to time, question becomes, how to securely communicate (in order to exchange the key) before they can securely communicate to exchange the actual data. Almost 40 year ago, public key cryptography arose to solve that difficulty. In public key cryptography security depends on hard mathematical problems and each party needs a public key for encryption and a private key for decryption [5].

Additionally, the communicating parties also want for instance to be sure that the messages cannot be changed while transmitted. For each of those constraints, appropriate solution is devised and implemented in practice. However, only a couple of months after the publication of the RSA algorithm paper, Rivest asked the question whether it is possible to work with encrypted data, with no need for decrypting it first. That question started the search for homomorphic encryption systems. It turned out that this question would pose unsolvable problem in following decades. If we have some information and we send it to the server for some additional processing, we may not want that the server can read our information. In the modern society, the need for privacy is present as never before and homomorphic encryption can help us in preserving it.

2. LITERATURE SURVEY

Cloud computing builds on established trends for driving the cost out of the delivery of services while increasing the speed and quickness with which services are deployed. In cloud computing, there are many vendors who offer different cloud services with different pricing models like Amazon, Google and Microsoft. Every vendor has their own policies and agreement. It provides the facility to store their data on the cloud. Then it is the liability of the cloud service provider to prevent the data from an illicit access. To secure the data various types of security mechanism are used.

2.1 Cloud deployment models

In cloud computing, there are different models used for cloud storage that allow users to maintain control over their data. These models are: private, public or hybrid. A private cloud means using own resources and own data centers making organization in control. The resources used in private cloud are not shared with other customers. The cloud infrastructure is maneuvered exclusively for an organization. It may be administered by the organization or a third party. Resources are dedicated only to the customer. While in public cloud, organization will be relieved from the management but the organization have less control.

2.2 Cloud deployment models

2.2.1 Software as a service (SaaS)

This features a complete application offered as a service on demand.

2.2.2 Platform as a service (PaaS)

The goal of this layer is to enable the developers to build their own applications.

2.2.3 Infrastructure as a service (IaaS)

Infrastructure as a service conveys fundamental storeroom and compute capabilities as standardized services over the network.

2.3 Security issues in Cloud Computing

The protection goals form the basis for the security requirements that must be fulfilled by IT systems in general and cloud computing systems in particular. These goals are usually fixed for a specific scenario in the framework of the requirement definition and are part of the non-functional requirements to be met by the cloud service provider as well as by the cloud service itself. As per NIST's characterization, information security is the practice of preserving integrity, confidentiality and availability of data from malicious access, system failure and etc.

2.3.1 Integrity

It means the information provided is authentic, complete and trustworthy. The data over the cloud shall not be changed or altered by any unauthorized user or by any malicious activities.

2.3.2 Confidentiality

Confidentiality means information is accessed only by an authorized person or shared among authorized groups. An authentication method includes credential verification that can be applied to protect data against malicious attack.

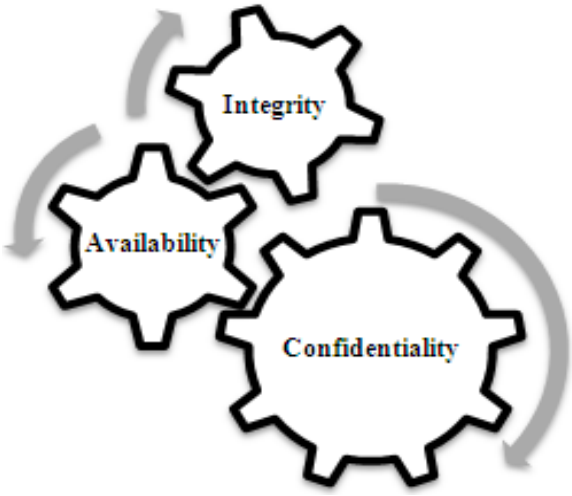

Fig. 1: Pillars of Cloud Computing Security

2.3.3 Availability

It refers to the availability of the requested data resource. Data should be available under authorized operation together with read, write and etc.

There are two different types of homomorphic encryption systems, fully homomorphic encryption systems and partially homomorphic encryption systems. Fully homomorphic encryption systems are defined over a ring and therefore support both operations, addition and multiplication. Partially homomorphic encryption systems are defined over a group and therefore support just a single operation on the encrypted data. In last thirty or so years there have been several encryption systems that belong to the partially homomorphic encryption systems set. Those systems can be segregated on a base of the operation they support on additive homomorphic encryption systems and multiplicative homomorphic encryption systems. There are also systems that allow arbitrary many operations of one type and only a limited number (usually just one) of operations of other type.

In 1978, Rivest gave Rivest algorithm, which is good example of multiplicative homomorphic cryptosystem. Due to the homomorphic encryption multiplicativity, product of the ciphers equals cipher of the product. It can be observed that RSA as a homomorphic encryption system has some security weaknesses. For example, if we assume that two messages m_1 and m_2 correspond to ciphers C_1 and C_2, and if the client sends that pair (C_1 , C_2) to the server, the server will perform multiplication of C_1 and C_2 and send it encrypted to the customer. If the foe captures ciphers C_1 and C_2 which are encrypted with the same private key, he will be able to decrypt all messages exchanged between the server and the customer. Due to homomorphic encryption multiplicativity, where product of the ciphers equals cipher of the product [6].

In 1978, Rivest, Adleman, and Dertouzos provided a novel way of ensuring the privacy of data which must be operated on. They are of intrinsically inadequate applicability, since evaluations may not in common be contained in the set of operations to be used. Additionally, it stays to be seen whether it is possible to have a privacy homomorphism with a large set of operations which is highly secure. The results presented give a basis for some optimism about finding useful privacy homomorphisms; the examples are suggestive if not very practical. This approach does not have enough utility to make it worthwhile in practice [7].

In 1984, Goldwasser and Micali gave first probabilistic Public key encryption scheme which is provably secure under standard cryptographic postulations. Nonetheless, it is not proficient cryptosystem, as ciphertexts may be quite a few hundred times larger than the original plaintext. To verify the security properties of the cryptosystem, Goldwasser and Micali anticipated the extensively used definition of semantic security [8].

In 1985, T ElGamal proposed a digital signature scheme which is based on the difficulty of computing discrete logarithms. The ElGamal signature algorithm described in this article is rarely used in practice. A variant developed at NSA and acknowledged as the Digital Signature Algorithm is a lot further widely used. There are several other deviations. The ElGamal signature scheme not to be confused with ElGamal encryption which was also invented by Taher ElGamal. The ElGamal signature method permits a third-party to confirm the authenticity of a message sent over an insecure channel [9].

In 1999, the Paillier cryptosystem, invented by Pascal Paillier, is a probabilistic asymmetric algorithm for public key cryptography. A few of well acknowledged examples of additive homomorphic encryption systems include Paillier system. This cryptosystem is based on the decisional composite residuosity assumption (DCRA). DCRA assumption can be stated as the following: given a composite n ($n = p \times q$ for primes p and q) and an integer z, it is tough to make a choice whether z is n residue modulo n^2 or not (i.e. it is hard to find out whether there exists y such that $z = y^n mod n^2$. Paillier cryptosystem computes encryption of $m_1 + m_2$ with public key and encryptions of m_1 and m_2. It's most important application is electronic voting where each vote is encrypted but only the sum is decrypted [10].

In 2001, Damgard and Jurik, proposed a generalization of Paillier's probabilistic public key technique, wherein the expansion factor is reduced and which allows adjusting the block length of the scheme even after the public key has been set, with no loss of the homomorphic property. It showed that the generalization is as secure as Paillier's original system and propose several ways to optimize implementations of both the generalized and the original scheme [11].

Each of the examples listed above allows homomorphic computation of only one operation either addition or multiplication) on plaintexts. A step closer to the full homomorphic encryption systems represents the systems that allow arbitrary many operations of one type and limited number of operations of other type. In 2005, Boneh-Goh-Nissim gave a cryptosystem which supports evaluation of an unlimited number of additions but at most one multiplication. This encryption scheme is based on bilinear pairings on elliptic curves. Scheme is limited in the size of message space due to the need to compute discrete logarithms during decryption [12].

The first construction of fully homomorphic scheme is described by Craig Gentry in 2009. This work is important not only because it represents the first fully homomorphic encryption technique, but as well as it served as a prototype for later fully homomorphic encryption schemes [13].

Gentry's scheme consists of more than a few steps. In first step it builds a somewhat homomorphic encryption scheme (SWHE), which only supports a limited number of operations. In SWHE system there are usual Encrypt and Decrypt functions, but also an Evaluate function. Evaluate function

carry outs an operation on ciphertext where it is realized as a combinatorial circuit. However, the problem with homomorphic encryption is that ciphertext contains random noise where addition roughly doubles the noise and multiplication squares the noise. This noise turn out to be superior with succeeding homomorphic operations and only ciphertexts whose noise size remains below a certain threshold can be decrypted correctly. The origin of noise is in the probabilistic encryption process where we add some noise during the encryption process, and eliminate that noise during the decryption method. Since this constraint, homomorphic encryption cannot have arbitrary number of operations and is therefore called somewhat homomorphic encryption.

To overcome this problem, Gentry devised bootstrapping method. In bootstrapping, Evaluate function is used to run Decrypt function. Since Evaluate works with encrypted data, the result is not the plaintext but a new encryption with condensed noise. This revitalized ciphertext can be used in subsequent homomorphic operations. By repeated refreshing of the ciphertexts, the number of homomorphic processes turn out to be unconstrained, and that results in a fully homomorphic encryption scheme. In order for system to work as described Decrypt function have not to go beyond the noise threshold. If this would not be the case, we would not be capable to apply bootstrapping to the SWHE scheme. Indeed, in Gentry's original FHE scheme this clause was not fulfilled. For that reason, he added a squashing procedure to the Decrypt function. Here, squashing means transforming SWHE scheme into one with same homomorphic capacity but with simpler Decrypt function [14]. The cost of the squashing method is that the key grows larger and more complicated.

3. PROPOSED SOLUTION

In proposed solution, many improvements have been made, introducing new variants, improving efficiency, and providing new features. It describes a different framework where the ciphertext noise grows only linearly instead of exponentially with the multiplicative level, as a result, bootstrapping technique is no longer needed to find a scheme supporting the homomorphic evaluation of any given polynomial size circuit.

The basic notions about homomorphic encryption and schemes presented here as;

Additive Homomorphic encryption is:

$$Enc(x + y) = Enc(x) + Enc(y).$$

Multiplicative Homomorphic encryption is:

$$Enc(x \cdot y) = Enc(x) \cdot Enc(y).$$

In DGHV scheme a ciphertext has the form of

$$C = q \cdot p + 2 \cdot r + m\,,$$

wherever p is the secret-key, m is plaintext (m is 0 or 1), q is a large random number and r is a small random number.

To decrypt, one calculates

$$m \;=\; (C \bmod p) \bmod 2$$

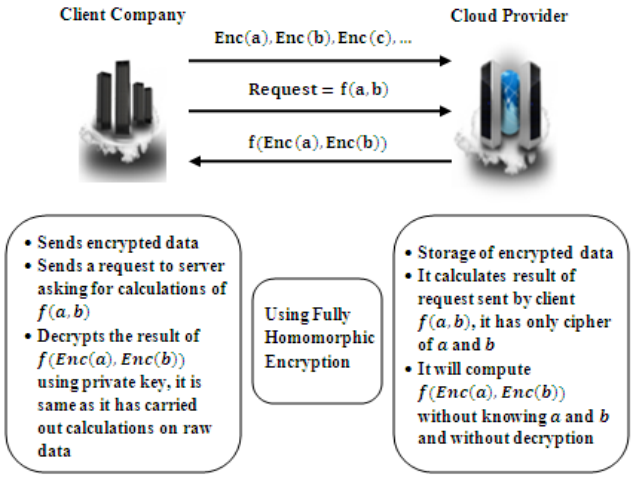

Fig. 2: Homomorphic Encryption in Cloud Computing

It is uncomplicated to observe that decryption works, providing the noise $2 * r$ is smaller than p.

Suppose that we have two ciphers C_1 and C_2 such as subsequent equations hold:

$$C_1 \;=\; q_1 \cdot p \;+\; 2 \cdot r_1 \;+\; m_1,$$

$$C_2 \;=\; q_2 \cdot p \;+\; 2 \cdot r_2 \;+\; m_2,$$

$$C_1 + C_2 = (q_1 + q_2) \cdot p \;+ 2\,.\,(r_1 \;+\; r_2) + m_1 + m_2,$$

$$C_1 \cdot C_2 \;=\; q_{12} \cdot p + \; 2 \cdot (2 \cdot r_1 \cdot r_2 \;+\; r_1 \cdot m_2 \;+\; r_2 \cdot m_1) + m_1 + m_2.$$

From these equations, we can obtain the encryption of $m1 + m2$ $(mod = m1 \oplus m2)$ by computing $C_1 + C_2$. Similarly, encryption of $m_1 \cdot m_2$ can be obtained by computing $C_1 \cdot C_2$. However one gets a new ciphertext with noise roughly twice larger than in the original ciphertexts C_1 and C_2. While the noise ought to stay below p, the number of permitted multiplications on ciphertexts is therefore limited. This is an example of somewhat homomorphic encryption scheme.

The technique known as "Fully Homomorphic Encryption without Bootstrapping" in which the approach is based on Learning with Errors (LWE) and Ring Learning with Errors (RLWE) problems.

In RLWE problem for security parameter λ, let $f(x) = x^d + 1$ where $d = d(\lambda)$ is a power of 2. Furthermore, let $q = q(\lambda) \geq 2$ be an integer, $R = Z[x] / f(x)$ and $R_q = R / qR$. Let $\xi = \xi(\lambda)$ be a distribution over R. The RLWE problem is to distinguish two distributions, in the first distribution, one samples (a_i, b_i) uniformly from R_{q^2}. In the second distribution, one first selects $s <- R_q$ uniformly and then samples (a_i, b_i) $2\,R_q^2$ by sampling $a_i <- R_q q$ uniformly, $e_i <- \xi$ and setting $b_i = a_i\, s + e_i$. The RLWE assumption is that the RLWE problem is infeasible.

In this scheme we are able to obtain a direct construction of a bootstrappable encryption scheme without having to "squash the decryption circuit", and therefore no necessity to apply a bootstrapping technique to achieve fully homomorphic encryption. Somewhat homomorphic encryption which was constructed to work over a ring R, where R can be described as $R = Z$ for integers, or $R = Z[x] / (x^d + 1)$ for polynomial ring with d being a power of 2. For integer q, R_q is used to

indicate R / qR. The error distribution parameter ξ is introduced to work over the ring R, where it is set to be as small as possible.

The major role of this system is the technique of managing noise so that it increases linearly with the multiplicative level instead of exponentially which was the case in previous FHE schemes.

4. CONCLUSION AND FUTURE SCOPE

As a result of the characteristics of the homomorphic encryption, carry out ciphertext operations and recovery on the cloud directly guarantees the security of cloud computing data and avoids the problem of the efficiency of the traditional encrypted data. Therefore, a practical fully homomorphic encryption scheme derived from Gentry cryptosystem, by means of merely basic modular arithmetic is proposed to ensure the privacy-preserving in cloud storage, which excellently realizes the need of ciphertext retrieval and other processing in untrusted servers. Homomorphic encryption systems evolved significantly in the last couple of years since the design of first fully homomorphic encryption system. However, all those systems are still to impractical for real-world applications. Because of that, it makes it interesting problem not only from educational viewpoint, but also from the industrial one. This will lead to the design of more efficient homomorphic encryption systems in the following years. Naturally, since there are applications where somewhat homomorphic encryption systems would be powerful enough, it seems we are closer to that goal than it may look like.

5. REFERENCES

[1] National Institute of Standards and Technology - Computer Security Division http://csrc.nist.gov/groups/ SNS/cloud-computing/

[2] Cloud Security Alliance, Security Guidance for Critical Areas of Focus in Cloud Computing V2.1 http://www.cloudsecurityalliance.org/csaguide.pdf.

[3] Cloud Security Alliance, Top Threats to Cloud Computing V1.0 http://www.cloudsecurityalliance.org/ topthreats/csathreats.v1.0.pdf.

[4] Maha TEBAA, Said EL HAJJI, Abdellatif EL GHAZI, "Homomorphic Encryption Applied to the Cloud Computing Security," Proceedings of the World Congress on Engineering 2012 London, U.K., Vol. 1, 2012.

[5] Darko Hrestak and Stjepan Picek, "Homomorphic Encryption in the Cloud," MIPRO 2014, Opatija, Croatia, pp. 1400-1404, May 2014.

[6] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, pp. 120–126, 1978.

[7] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," Foundations of Secure Computation, Academia Press, pp. 169–179, 1978.

[8] S. Goldwasser and S. Micali, "Probabilistic encryption," J. Comput. Syst. Sci., vol. 28, no. 2, pp. 270–299, 1984.

[9] 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., pp. 10–18, 1985.

[10] 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.

[11] I. Damgard and M. Jurik, "A generalisation, a simplification and some applications of paillier´s probabilistic public-key system," in In proceedings of PKC '01, LNCS series. Springer-Verlag, pp. 119–136, 2001.

[12] 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, pp. 325–341, 2005.

[13] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford, CA, USA, 2009.

[14] V. Vaikuntanathan, "Computing blindfolded: New developments in fully homomorphic encryption," in FOCS, pp. 5–16, 2011.