International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
NATIONAL CONFERENCE on Developments, Advances & Trends in Engineering Sciences
(NCDATES- 09th & 10th January 2015)

RESEARCH ARTICLE OPEN ACCESS

Homomorphic encryption applied on Cloud

[1] Mr.V.Biksham, [2]Dr.D.Vasumathi,
Research Scholar, Dept. of CSE JNTU College of Engg & Tech, Hyderabad-501401
Professor, Department of CSE, CMR Engineering College, JNTUniversity,Hyderabad-85

Abstract:
Cloud computing is something that businesses have long been talking about, and in the recent past, many organization have adopted cloud technologies. A highly virtualized data centre environment, with high degrees of automation and the ability to seamlessly move between on and off premise infrastructure is the optimal architecture for the future. In addition data privacy in cloud is become more challenged and its also an issue to many researches in recent development. When data is send to the cloud we used a standard encryption method to secure the computations and storage of the data. Due to perform computations on the encrypted data by the cloud service provider, need to access the plaintext or raw data. In this paper we proposes a method called homomorphic encryption to run operations in cloud encrypted data without decrypting them which will provide same results after computations as if we have worked directly on the plain text.
Keywords: virtualization, homomorphic, encryption, automation

I. Introduction

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrust remote services with a user's data, software and computation. Cloud computing consists off hardware and software resources made available on the Internet typically provide access to advanced software applications and high –end networks of server computers

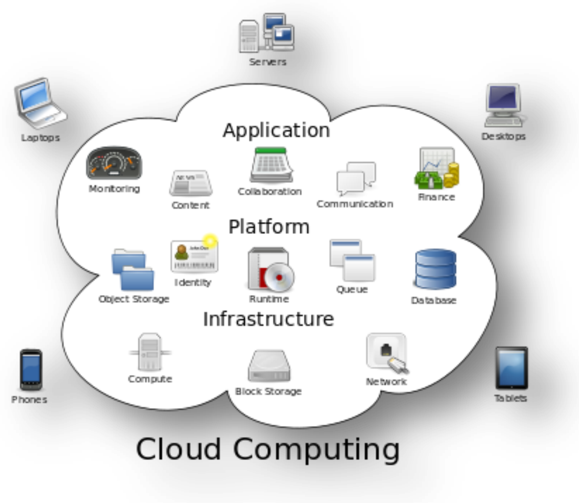

Fig: 1 simple cloud computing

The goal of cloud computing is to apply traditional supercomputing, or high-performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second, in consumer-oriented applications such as financial portfolios, to deliver personalized information, to provide data storage or to power large, immersive computer games.

The cloud computing uses networks of large groups of servers typically running low-cost consumer PC technology with specialized connections to spread data-processing chores across them. This shared IT infrastructure contains large pools of systems that are linked together. Often, virtualization techniques are used to maximize the power of cloud computing.

The salient characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below:

On-demand self-service: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.

Broad network access: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).

Resource pooling: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.

Rapid elasticity: Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

Measured service: Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be managed, controlled, and reported providing transparency for both the provider and consumer of the utilized service.

II. Virtualization:

In cloud computing, virtualization means to create a virtual version of a device or resource, such as a server, storage, device, network or even an operating system where the framework divides the resource into one or more execution environments. Even something as simple as partitioning a hard drive is considered virtualization because you take one drive and partition it to create two separate hard drives. Devices, applications and human users are able to interact with the virtual resource as if it were a real single logical resource.

Software Defined Data Centre (SDDC), Hybrid Cloud and End-User Computing vision becomes a reality in India; Estimated to help Indian businesses avoid spending $5.5 billion by 2020.VMware, Inc., the global leader in virtualization and cloud infrastructure, showcased the future of IT at its annual vForum conference in Mumbai. Charting a path from today till the year 2020, the enterprise in India will leverage IT outcomes to drive business benefits and increase efficiency, control and agility as a transition to the mobile cloud era.“The old model of IT (Information Technology) is not good enough in the software defined era, where traditional businesses are being challenged and a new approach to align IT with the business is needed to compete. Asia Pacific and Japan, VMware. “We are on the cusp of the next big technology era in Asia where IT will adapt to the speed of business and drive business outcomes with a software defined enterprise and many Indian organizations are already on this journey”.

According to an IDC report commissioned by VMware titled 'Empowering Organizations in a Software Defined World,' an estimated US$4 billion is expected to be avoided between 2014 and 2020 with a software defined approach to managing IT in India as shown in fig(2). For the Asia Pacific and Japan region, an estimated US$92.4 billion is expected to be avoided. At the forum event, VMware also showcased new products and solutions designed to drive business benefits in the software defined era for Indian businesses. VMware's SDDC strategy is the future of the data centre. VMware’s SDDC is the ideal architecture in which all resources are fully virtualized and made available via private, public, and hybrid cloud, and they all deliver the peak of flex ibility, efficiency, agility, control and choice for IT and the business for the organization. VMware EVO: RAIL jumpstarts a customer's journey to the Software-Defined Data Centre.

The appliance removes the burden of integration, optimization, configuration, testing, installation and on-going management, upgrading and patching from IT teams. VMware NSX is the net work virtualization and security platform for the software defined data centre.NSX brings virtualization to enterprises' existing network and transforms network operations and economics. VMware's Virtual SAN automatically and dynamically matches requirements with underlying storage resources. With Virtual SAN, many manual storage tasks are automated and in the process, delivering a more efficient and cost-effective operational model that is highly effective in the long run.

2.1 Open Cloud Technology.

VMware's Integrated Open Stack solution enables IT organizations to quickly and cost-effectively deliver developer-friendly Open Stack APIs and tools on top of their existing VMware Infrastructure. They can provide API-driven infrastructure for internal developers, and repatriate workloads from unmanageable and insecure public clouds. IT can manage and troubleshoot an Open Stack cloud with the same familiar VMware tools they already use every day, providing significant operational cost savings and faster time-to value.

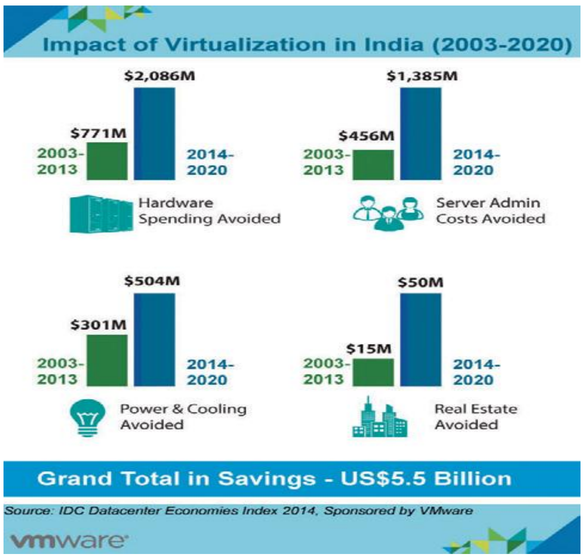

Fig 2: impact of virtualization in India 2003-2020

III. Homomorphic Encryption:

Homomorphic Encryption systems are used to perform operations on encrypted data without knowing the private key (without decryption), the client is the only holder of the secret key. When we decrypt the result of any operation, it is the same as if we had carried out the calculation on the raw data. 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: +, ×, ⊕ and without using the private key. Among the Homomorphic encryption we distinguish, according to the operations that allows to assess on raw data, the additive Homomorphic encryption (only additions of the raw data) is the Pailler [2] and Goldwasser-Micalli [3] cryptosystems, and the multiplicative Homomorphic encryption (only products on raw data) is the RSA [4] and El Gamal [5] cryptosystems.

Homomorphic encryption: Encryption with the property such that performing a function on two values separately and then encrypting the result yields the same final value as first encrypting two values separately and then applying the function to the results.

Importance of homomorphism: Homomorphism decouples the ability to perform computations from the necessity to view the data as cleartext. This allows owners of sensitive data to manipulate encrypted secret data while it resides in an insecure location or to outsource computations on secret data to an untrusted third party.

3.1 History of the Homomorphic Encryption

In 1978 Ronald Rivest, Leonard Adleman and Michael Dertouzos suggested for the first time the concept of Homomorphic encryption [8]. Since then, little progress has been made for 30 years. The encryption system of Shafi Goldwasser and Silvio Micali was 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 was also proposed a provable security encryption system that was also an additive Homomorphic encryption. Few years later, in 2005, Dan Boneh, Eu-Jin Goh and Kobi Nissim [9] invented a system of provable security encryption, with which we can perform an unlimited number of additions but only one multiplication. B. Additive Homomorphic Encryption A Homomorphic encryption is additive, if:

3.2Partial Homomorphic Encryption:

Different cryptosystems support varying levels of homomorphism.

Partially homomorphic: Homomorphic over a limited set of functions.

Cryptosystem	Homomorphic Operations
RSA	Multiplication mod n
Elgamal	Multiplication, Exponentiation*
Paillier	Addition, Subtraction, Multiplication*
Goldwasser-Micali	XOR
Benaloh	Addition, Subtraction
Naccache-Stern	Addition, Subtraction, Multiplication*

* By a constant only

Table1: List of Homomorpic cryptosystems.

Somewhat homomorphic: Allow functions to be performed only a limited number of times. This limitation stems from an error term which increases with each operation. Once the error exceeds a certain tolerance, the result becomes irreparably garbled and can no longer be decrypted. For example, Boneh-Goh-Nissim supports unlimited additions, but only a single multiplication.

.

3.3Fully Homomorphic Encryption:

First fully homomorphic cryptosystem: Developed by Craig Gentry at Stanford in 2009. Revised by van Dijk, Gentry, Halevi, and Vaikuntanathan.

Homomorphism over any function: Enabled by making the system homomorphic over the logical NAND function. Any logical circuit can be constructed using only NAND functions.

Mathematical hardness: Original cryptosystem was based on ideal lattices. Later published a revised scheme, called DGHV, which retained the original homomorphic properties and techniques, but relied on the simpler foundation of modular arithmetic and the approximate GCD problem.

Partially homomorphic algorithm: The key is a positive, odd integer *p*. Given suitably chosen random , the encryption of a bit *m* {0,1} is given by . The sum 2r +m is called noise term. The plaintext is the parity of the noise term. As long as , the noise term, and thus the plaintext, is recoverable by computing the residue class of c modulo p in .

Bootstrapping: The process of converting a somewhat homomorphic cryptosystem into a fully homomorphic cryptosystem. A somewhat homomorphic cryptosystem is bootstrappable if it can evaluate its own decryption circuit with a sufficiently small error constant.

IV. Implementation:

This enables the implementation of a Recrypt function which homomorphically decrypts the message and re-encrypts, reducing the error constant.

Functions: KeyGen, Encrypt, Evaluate, Decrypt

RSA Algorithm (Cryptosystem)

Key Generation: key Gen(p,q)
Input: p,q €P
Compute: n=p.q Choose e such that Ø(n)=(p-1)(q-1) Gcd(e, Ø(n))=1 Determine d such that e.d mod Ø(n))=1
Output: (pk,sk) Public key=(e,n) Secret key=(d,n)

Encryption: Enc(m,pk)
Input: m€Z_n
Compute: C=m^e mod n
Output: C €Z_n

Decryption:Dec(C,sk)
Input: e €Z_n
Compute: m=C^d mod n
Output: m €Z_n

Fig3: RSA algorithm

Suppose we have two ciphers C1 et C2 such that

$C1=m1^e$ mod n

$C2=m2^e$ mod n

$C1.C2=m1^e\ m2^e$ mod n = $(m1m2)^e$ mod n

RSA cryptosystem realize the properties of the multiplicative Homomorphic encryption, but it still has a lake of security, because if we assume that two ciphers C2,C2 corresponding respectively to the messages m1,m2, so:

$C1=m1^e$ mod n

$C2=m2^e$ mod n

The client sends the pair (C1,C2) to the cloud server, the server will perform the calculations requested by the client and send the encrypted result (C1XC2) to the client.

If the attacker intercepts two ciphers C1 et C2, which are encrypted with the same private key, he/she will be able to decrypt all messages exchanged between the server and the client. Because the Homomorphic encryption is multiplicative, i.e. the product of the ciphers equals the cipher of the product.

Example: the application of rsA multiplicative Homomorphic encryption on two messages m1 and m2 Let, for p=3,q=11,e=7and d=3 with block size =1Two messages, m1 and m2 and their ciphers C1 and c2 respectively, obtained using the RSA encryption.

M1=458742→C1=000400050008000700040002

M2=345635→C2=000300040005000600030005

We convert the ciphers into binary system

The Block of C1 in binary system	The Blocks of C2 in binary system
00 04⇒00 0100	00 03⇒00 0011
00 05⇒00 0101	00 04⇒00 0100
00 08⇒00 1000	00 05⇒00 0101
00 07⇒00 0111	00 06⇒ 00 0110
00 04⇒00 0100	00 03⇒00 0011
00 02⇒00 0010	00 05⇒00 0101

The binary multiplication of the ciphers block by block is as follows

00 0101 x00 0011=00 1100	00 12
00 0101x00 0100=00 10100	00 20
00 1000x00 0101=00 101000	00 40
00 0111x00 0110=00 101010	00 42
00 0100x00 0011=00 1100	00 12
00 0010x00 0011=00 1010	00 10

If we decrypt the cipher C1 X C2 with the private key, we get, C1C2=00 01 00 02 00 02 00 00 04 00 00 04 00 02 00 01 00 02 00 01 00

So:M1M2=12 2 0 4 0 4 2 1 2 1 0

This is exactly the same raw message obtained by multiplying m1 X m2

M1=458742

M2=345635

M1,M2=122040421210(we are multiplying m1X m2 block by block)

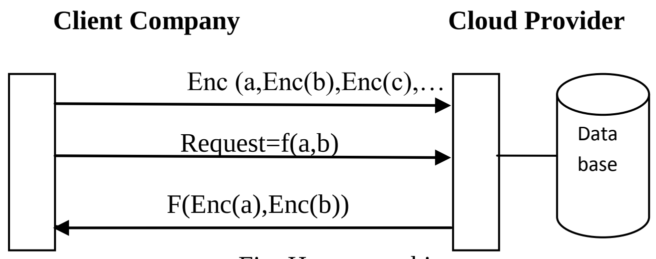

Fig: Homomorphic

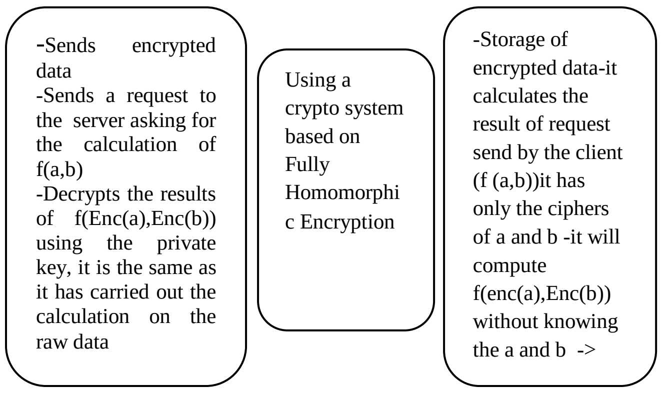
-Sends encrypted data
-Sends a request to the server asking for the calculation of f(a,b)
-Decrypts the results of f(Enc(a),Enc(b)) using the private key, it is the same as it has carried out the calculation on the raw data
Using a crypto system based on Fully Homomorphic Encryption
-Storage of encrypted data-it calculates the result of request send by the client (f (a,b))it has only the ciphers of a and b -it will compute f(enc(a),Enc(b)) without knowing the a and b ->

Fig4. Encryption applied to the Cloud Computing.

For all types of calculation on the data stored in the cloud, we must opt for the fully Homomorphic encryption which is able to execute all types of operations on encrypted data without decryption. The application of fully Homomorphic encryption is an important stone in Cloud Computing security, more generally, we cloud outsource the calculations on confidential data to the Cloud server, keeping the secret key that can decrypt the result of calculation.

V. Applications

Cloud computation: Homomorphism enables querying, retrieving, and operating on encrypted data in the cloud. Electronic voting: Statistics can be computed by a third party while maintaining the privacy of individual votes. Data mining: Companies can contract a third party to perform large data mining computations on secret data by sending the data homomorphically encrypted. Financial transactions: Online financial companies can compute credit scores and other financial data homomorphically, without ever seeing the users finances in cleartext. Electronic cash: Uses blind signatures, where transactions can be authenticated homomorphically. Medical records: Homomorphism would allow medical records to be processed by a third party, without breaking patient confidentiality.

VI. Conclusion:

Speed and security: Current fully homomorphic cryptosystems are prohibitively slow and insecure for many promising applications. Current applications: Some applications with limited requirements have been able to successfully employ stronger, partially or somewhat homomorphic cryptosystems. Future of homomorphic cryptography: The great number and diversity of potential applications for homomorphic encryption should incentivize researchers to make progress toward a strong, fast, fully homomorphic cryptosystem. An ideal homomorphic cryptosystem: To achieve the full potential as a ubiquitous encryption method, an ideal homomorphic cryptosystem would possess speed and security comparable to current cryptosystems, while allowing full homomorphism over all functions.

References

[1] Vic (J.R.) Winkler, "Securing the Cloud, Cloud Computer Security, Techniques and Tactics", Elsevier, 2011.
[2] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In 18th Annual Eurocrypt Conference(EUROCRYPT'99), Prague, Czech Republic , volume 1592, 1999
[3] Julien Bringe and al. An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication, Springer-Verlag, 2007.
[4] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public key cryptosystems. Communications of the ACM, 21(2) :120-126, 1978. Computer Science, pages 223-238. Springer, 1999.
[5] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 469-472, 1985.
[6] Craig Gentry, A Fully Homomorphic Encryption Scheme, 2009.
[7] WiebBosma, John Cannon, and Catherine Playoust. The Magma algebra system I: The user language. J. Symbolic Comput., 24(3-4):235-265, 1997. Computational algebra and number theory, London, 1993.
[8] Ronald L. Rivest, Leonard Adleman, and Michael L. Dertouzos. On Data Banks and Privacy Homomorphisms, chapter On Data Banks and Privacy Homomorphisms, pages 169-180. Academic Press, 1978.
[9] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Theory of Cryptography Conference, TCC'2005, volume 3378 of Lecture Notes in Computer Science, pages 325-341. Springer, 2005.
[10] Proceedings of the World Congress on Engineering 2012 Vol I WCE 2012, July 4 - 6, 2012, London, U.K.