IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-4, April 2019 E-ISSN: 2347-2693

An Expensive Study of Homomorphic Encryption to Secure Cloud Data

P. Venkateswarlu[1*], B. Manasa[2], K. Srikanth[3]

[1,2,3]Dept. of IT, University College of engineering, Vizianagaram, Andhra Pradesh, India

Corresponding Author:venkat.pynam@gmail.com

DOI: https://doi.org/10.26438/ijcse/v7i4.765770 | Available online at: www.ijcseonline.org

Accepted: 18/Apr/2019, Published: 30/Apr/2019

Abstract: The data present in cloud should be provided security because the data is stored on distributed servers or systems. The security issues concerning data are loss of authentication and privacy. In cloud platform security is provided through encryption techniques only during transmission of data. To process the data present on a remote server, the cloud providers need to access the raw data to allow us to perform operations on the data. It can't provide confidentiality to the data as it is exposed in the form of plain text in operational stage. The disadvantages are the data can't remain confidential and invisible to cloud service provider and the data can be reused by the cloud service provider. Encryption solves major privacy issues but performing computations, one needs to perform the decryption first. The data privacy issue can be resolved if user is able to carry out computations on encrypted data. Homomorphic Encryption technique enables computing with encrypted data. That means one can perform the operations on this data without converting into the plain text. Data is not in its original form in its most of the operational stages on the cloud. It enables computations on encrypted data. Our idea is to encrypt data before sending it to the cloud, but to execute the calculations; the data should be decrypted every time to work on it. Decryption can be performed directly without encryption where the client is the only holder of the secret key. When the result of any operation is decrypted, it is the same as carrying out the calculations on the raw data.

Keywords: cloud computing, encryption technique, plain text, decryption and Homomorphic encryption.

I. INTRODUCTION

Cloud computing has gained a lot of importance as it incorporates many needs of today's technology. It is defined as set of resources or services offered via the internet to users on their own demand by cloud service providers. As every organization is moving its data to the cloud for means it uses the storage service provided by the cloud provider. So it is necessary to protect that data against modification, unauthorized access or denial of services. Cloud Computing can be made more secure using cryptographic algorithms. Cryptography can be defined as the art or science of keeping data secure by converting it into non readable forms. Cloud computing is consignment of computing services over the Internet. It has been credited with high priority in technological market trends due to cost reduction, greater flexibility and optimal resource utilization. Cloud services allow individuals and business organizations to use the software and hardware services that are managed by the third parties at remote locations. Cloud services' examples include social networking sites, online file storage, webmail, and online business applications. The cloud computing architecture allows access to computer resources and information from anywhere that a network connection is available. The main Features of Cloud computing are Self-service provisioning, Pay-per-use, Maintenance of the servers, Location-free, Backup, Cloud-based workload and file sharing streamline [1].

Cryptography implement for secure communication in the presence of malicious third-parties known as attacker. Encryption uses an algorithm and a key to transform an input into an encrypted output. The Algorithm always converts the same plaintext into the same cipher text if the same key is used. Cryptography associates creating written or generated code that allows information to be kept secret. It follower data into a format that is unreadable for an unapproved user grant it to be disseminate without unauthorized entities decoding it back into a readable format thus negotiate the data.

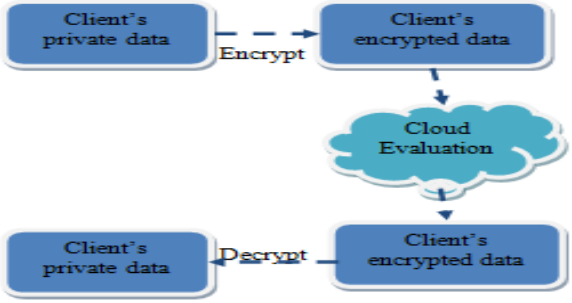

Figure 1: A general framework for data protection over cloud

Cryptography in current days is considered sequence of three types of algorithms. They are Symmetric-key algorithms, Asymmetric-key algorithms and Hashing. Integrity of data is ensured by hashing algorithms. Data cryptography mainly is the clamber of the content of the data, such as text, image, audio, video and so forth to make the data unreadable, invisible or meaningless during communication or storage is call Encryption. The main aim of cryptography is to take care of data secure from attacker. The reverse development of getting back the original data from encrypted data is decryption, which restores the original data. To encrypt data at cloud storage the couple of symmetric-key and asymmetric-key algorithms can be used. Cryptography is the discipline of securing data; cryptanalysis is the discipline of analyzing and breaking secure communication. Classical cryptanalysis affects an attractive combination of analytical reasoning, application of mathematical tools, pattern finding, patience, determination. Cryptanalysts are also called attackers.

II. HOMOMORPHISM

Homomorphism is a map which preserves structure between groups like algebraic structures. Homomorphism is a structure perpetuate map between two algebraic structures, such as groups. The word homomorphism aroused from words: homo meaning "same" and morphism meaning "form" or "shape" from the ancient Greek language. Homomorphism maps two groups which respects group structure. Defining in a formal way let G, H be two groups, and f maps G to H for every g ∈ G, f (g) ∈ H. Then f derives homomorphism if for all g1, g2 ∈ G, f(g1 g2) = f(g1)f(g2) [2].

2.1 Homomorphic encryption

Homomorphic encryption is a type of encryption that allows specific types of calculation that can be execute on cipher text and access result in encrypted form which when decrypted equals results of operation executed on plain text". For example, if a person can add two encrypted numbers and then the second person can decrypt the result, without being able to find the value of the individual numbers. When the data is conveyed to the cloud we use standard encryption methods to secure this data, but when we want to do the calculations on data located on a remote server, it is necessary that the cloud provider has connection to the raw data, and then it will decrypt them. As we all know that, the demand for privacy of data and algorithms to handle the information of activity has increased extremely over the last decades. To accomplish this type of technology such as data encryption methods with the use of tamper resistant hardware is used. Demanding situations arise when data secrecy issues are concerned, however data privacy is established when Homomorphic Encryption comes into picture. It allows us to perform operations on data in encrypted form without decryption.

III. SYSTEM ANALYSIS

In cloud storage security is provided through encryption techniques only during data storage and transmission of data. To process the data present on a remote server, the cloud providers need to access the raw data to allow us to perform operations on the data. It can't provide confidentiality to the data as it is exposed as plain text in operational stage. But the existing cryptographic systems use single level encryption algorithms. There are chances that Cyber criminals can crack single level encryption algorithms. Hence we propose a system which uses multilevel encryption and multilevel decryption to provide more security for Cloud data Storage.

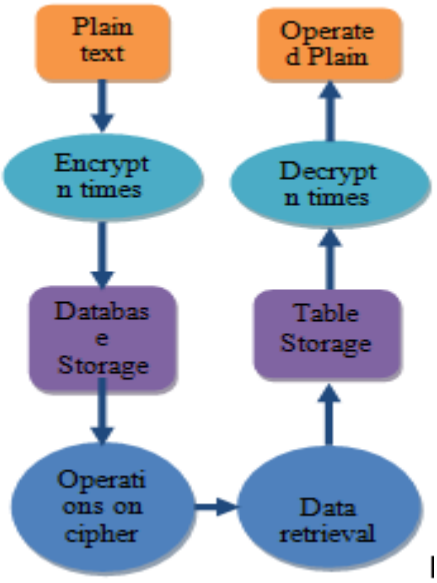

Figure 2: System Architecture for Homomorphic encryption to secure cloud data

Generally, when data is encrypted it is not easily understood by unauthorized people and to get plain text back decryption is used. For any kind of computation, one needs to perform the decryption first. Encryption solves major issues. But the capability of cloud can be abused if user is able to carry out computation on encrypted data. Homomorphic Encryption technique enables computing with encrypted data. It means, one can perform the operations on this data without converting into the plaintext. The data is in encrypted state in its better of the stages on the cloud. This proposal is to encrypt data before sending it to the cloud, but to execute the calculations; the data should be decrypted every time we need to work on it. Until now it was hopeless to encrypt data and to trust a third party to keep them safe and able to perform inaccessible calculations on them. Homomorphic encryption enables operations being performed on encrypted data without knowing private key i.e., without decryption where the client is the only holder of the secret key [3]. The Advantages are Plain text is not exposed at any stage,

Encrypted query processing, and also some Limitations are there one is A bit slower because operations are performed on encrypted data, Homomorphic encryption is computationally intensive.

IV. ALOGORITHM USED

RSA ALOGORITHM:

RSA is an algorithm used in modern computers to encrypt messages and decrypt them. It is an asymmetric key cryptographic algorithm. The word Asymmetric defines that there are two different keys. This is also called public key cryptography, because one of the key can be given to anyone. The other key rather than public key must be kept private. Security of RSA is based on the fact that finding the factors of a large integer into primes is hard (the factoring problem). RSA is abbreviated as Ron Rivest, Adi Shamir and Leonard Adleman, which is described firstly in public described in 1978. A user of RSA creates the product of the two large prime numbers and then publishes, combined with an auxiliary value, as the public key. The prime factors must be kept private. The public key which can be used either for encryption or decryption is made of the modulus and the public (or encryption) exponent. The private key which can be used either for encryption or decryption is made of the modulus and the private (or decryption) exponent which must be kept secret [4].

Encryption and Decryption

Once the pair of keys has been generated, performing encryption and decryption are computationally easy and relatively straight forward. RSA doesn't operate directly on strings of bits as in the case of symmetric key cryptography. It operates on number modulo n. Hence, it is necessary to represent plaintext as a series of numbers which are less than n.

RSA Encryption

- Suppose the sender wish to send some text message to someone he should know receiver's public key (n, e).
- The sender represents the plaintext as series of numbers that are less than n.
- To encrypt first plaintext P, which is number modulo n? The encryption process is a simple mathematical step as

$$C=P^e \bmod n$$

- In other sense, cipher text C is equal to plaintext P multiplied by itself some e times and then reduced modulo by n. This means that C is also a number which is less than n.

RSA Decryption

The decryption procedure for RSA is also very straight forward. Suppose the receiver of public-key pair (n,e) has received ciphertext C, receiver raises the C to the power of private key d. The result of modulo n will be plaintext P.

$$\text{Plaintext} = C^d \bmod n$$

RSA Analysis

The security of RSA depends on the strength of two separate functions. The RSA cryptosystem is the most popular public-key cryptosystem, strength of which is based on practical difficulty of factoring the primes from very large numbers.

- **Encryption Function** − It is considered as a function that works in one-way of converting plaintext into cipher text, it can be reversed only with knowledge of private key, d.
- **Key Generation** − The difficulty of computing private key from RSA public key is equivalent to the factorization of the modulus n. Intruders thus cannot determine the private key just with the knowledge of an RSA public key, unless he can factor n to primes.

Generation of RSA Key Pair

Each party or a person, who would like to participate in communication using encryption, needs to generate a pair of keys, namely public key and private key. The process of generation of keys is described as below

- Generate the RSA modulus (n)
  1. Select two large prime numbers, p and q.
  2. Calculate n=p*q. Let n be a large number, for strong unbreakable encryption.
- Find Derived Number (e)
  1. The number e must be less than $(p-1)(q-1)$ and greater than 1.
  2. No common factor should exist for $(p-1)(q-1)$ and e except for 1. In other sense two numbers e, $(p-1)(q-1)$ are co-primes.
- Form the public key
  1. The pair of numbers (e, n) forms the RSA public key and is published to everyone.
  2. Difficulty in factorizing large number into prime factors ensures that the attacker can't find the two primes (p & q) which are used to obtain n, though n is part of the public key. This is strength of RSA.
- Generate the private key
  1. From p, q, and e private key d is calculated. For given numbers n and e, there is a unique number d.
  2. Inverse of e modulo $(p-1)(q-1)$ is d. This means that d is number less than (p-1) (q - 1) such that when multiplied by e, it is equal to 1 modulo (p - 1)(q - 1).
  3. This relationship can be written mathematically as $e*d = I \bmod (p-1)(q-1)$

An example of developing RSA Key pair is given below [5]. (For the comfort of understanding, the primes p & q taken here are small values. Practically, these values very high).

- Let the two primes be p = 7 and q = 13. Thus, modulus n = p*q = 7 * 13 = 91.
- Calculate $(p - 1)(q - 1) = 6 \times 12 = 72$.Choose e = 5, which is a valid choice since no greatest common divisor exists for 72 except for 1.
- Pair of numbers (e, n) = (5, 91) forms the public key and can be published and made available to everyone who we wish to send receiver, encrypted messages.
- Input e = 5, p = 7, q = 13 and to Extended Euclidean Algorithm. The output will be d=29.
- Check that the d calculated is correct by computing de = $29 \times 5 = 145 = 1 \bmod 72$
- Hence, private key is (91, 29) public key is (91, 5).

V. METHODOLOGY

Multiplicative Homomorphic encryption

A Homomorphic encryption is multiplicative if

$$\text{Enc}(x.y) = \text{Enc}(x) \,.\, \text{Enc}(y) \qquad 1.3$$

$$\text{Enc}\left(\pi^{1}_{i=1} m_i\right) = \pi^{1}_{i=1} \text{Enc}(m_i) \qquad 1.4$$

Homomorphic encryption uses RSA algoritham which has multiplicative homomorphic property of RSA, In RSA scheme, assume that the public key= (n,e) the plain text from a group(P,.) and the cipher text from a group (C,.), where is the modular multiplication. For any two plaintext m1, m2 in P, it holds that

$$E(m_1, pk).E(m_2, pk) = m_1^e.m_2^e \pmod n$$
$$= (m_1, m_2)^e \pmod n$$
$$= E(m_1.m_2.pk)$$

Let p=3, q=5, e=9 and d=1 with block size=1

Two messages m1 and m2 and their cipher C1 and C2 respectively, obtained using the RSA encryption.

m1=589625 and

C1=00 05 00 08 00 09 00 06 00 02 00 05

m2=236491 and

C2=00 02 00 03 00 06 00 04 00 09 00 01

The block of C1 in binary system is as described in table 1.

Table 1: Block of C1 in binary system

00 05 => 00 0101
00 05 => 00 0101
00 08 => 00 1000
00 09 => 00 1001
00 06 => 00 0110
00 02 => 00 0010
00 05 => 00 0101

The block of C2 in binary system is as described in table 2.

Table 2: Block of C2 in binary system

00 02 => 00 0010
00 02 => 00 0010
00 03 => 00 0011
00 06 => 00 0110
00 04 => 00 0100
00 09 => 00 1001
00 01=> 00 0001

The binary multiplication of the cipher block by block is as shown in table 3.

Table 3: binary multiplicative of the cipher block by block

00 0101×00 0010 = 00 1010	00 10
00 1000×00 0011 = 00 11000	00 24
00 1001×00 0110 = 00 110110	00 54
00 0110×00 0100 = 00 11000	00 24
00 0010×00 1001 = 00 10010	00 18
00 0101×00 0001 = 00 0101	00 05

If we decrypt the cipher C1 X C2 with the primary key we get

C1 C2= 00 10 00 02 00 04 00 05 00 04 00 02 00 04 00 01 00 08 00 05

So m1 m2=10 2 4 5 4 2 4 1 8 5

This is exactly the same raw message obtained by multiplying m1x m2

m1= 5 8 9 6 2 5

m2= 2 3 6 4 9 1

m1 m2 = 10 24 54 24 15 5(we are multiplying m1x m2 block by block)

RSA procedure

Step 1: Choose 2 prime numbers p=3238150651, q=3763501697

Step 2: Get product of those numbers

n=p*q=12186785470180154747;

Find the totient of product of primes

Totient = (p-1)*(q-1)

= 3238150650*3763501696

= 12186785463178502400;

Step 3: Find integers that result in 1 mod Totient

12186785463178502401, 24373570926357004801, 365603563895365507201.

Step 4: Choose a 1 mod Totient value with exactly 2 prime factors

12186785463178502401=56401*7474645299333946801

Step 5: Choose key for encryption e, key for decryption d Let e=56401,

d=7474645299333946801

Step 6: Encrypt with C= (p pow e) mod n formula, Decrypt

with P=(C pow d) mod n

Let p1= investment made in a company=25000$

Let p2 = rate of growth of investment = 2

Step 7: Compute

C1 = (25000 pow 56401) mod 12186785470180154747

= 5449514454734497269

C2 = (2 pow 56401) mod 12186785470180154747

= 7653678737364155024

```
C:\Users\Sahithi\Desktop>javac RSAHomomorphic.java

C:\Users\Sahithi\Desktop>java RSAHomomorphic
enter two numbers
25000
2
x1 = 25000
x2 = 2
E ( x1 ) = 5449514454734497269
E ( x2 ) = 7653678737364155024
E ( x1 ) * E ( x2 ) = 41708832911160038795182057019620629456
D ( E ( x1 ) * E ( x2 ) ) = 50000

C:\Users\Sahithi\Desktop>
```

Figure 3: Homomorphic property of RSA

Now store the encrypted data in cloud platform. Whenever required, operations are performed on the data. The result retrieved and decrypted are same as performing operations on plain text.

Multi Level Security

Multi layer security can be provided to the data by encrypting it n times as shown in figure below [6]. Then the data can be stored in cloud platform and can be operated with data should be decrypted n times to get original plain text as shown in figure 4. The procedure is as follows:

- Encrypt the data using RSA algorithm
- Given as input to the RSA algorithm, the obtained cipher text.
- In a similar way encrypt the data n times.
- Now the cipher text obtained can be stored on cloud platform.
- Operations can be performed on this encrypted data.
- Retrieve the data from cloud storage.
- Decrypt the data n times using RSA algorithm to obtain original plain text as shown in figure 5

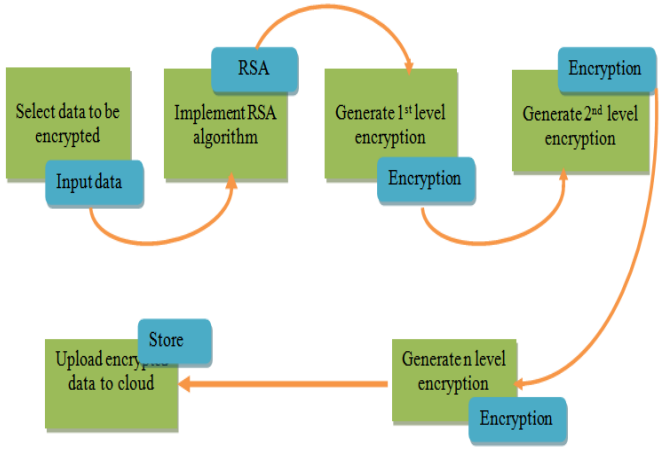

Figure 4: Multilevel Encryption of plain text using RSA algorithm

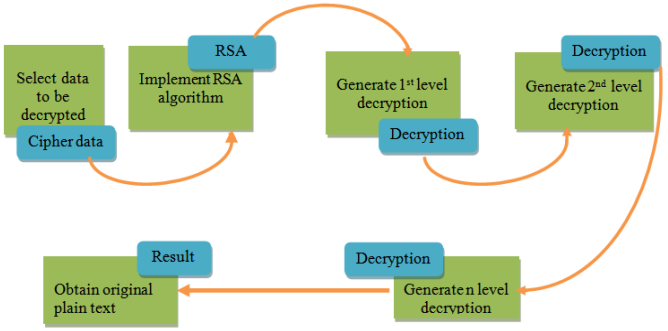

Figure 5: Multilevel decryption of cipher text using RSA algorithm

VI. CONCLUSION

The Security of Cloud Computing is based on Homomorphic Encryption is a new approach of security which is implementing to provide the results of computations on encrypted data without knowing the raw entries on which the calculation was carried out concerning the confidentiality of data. In the proposed system an application of a method to execute operations on encrypted data without decrypting them is worked upon, which will provide the same results after calculations as if we have worked precisely on the raw data. In this system n-layer security is provided to the cloud data using RSA algorithm i.e., the data is encrypted n times before being placed in cloud platform and when retrieved, decrypted n times to get original plain text. Homomorphic encryption has several benefits including homomorphic encryption solves the confidentiality problems when data is shared by different users and performs different operations on it, provides privacy by having ability to directly operate on encrypted data. However, it's computational and storage overhead has restricted its use. The future enhancement for this is efforts can be made to build up a Multicloud architecture as an efficient scheme that can provide security using Homomorphic schemes.

REFERENCES

[1]. Sean Marston and al. "Cloud computing the business perspective", Volume 51, Issue 1, Pages 176–189, http://www.sciencedirect.com, April 2011.

[2]. Vic (J.R.) Winkler, "Securing the Cloud, Cloud Computer Security, Techniques and Tactics", Elsevier, 2011.

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

[4]. Pascal Paillier, "Public-key cryptosystems based on composite degree residuosity classes", In 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic, volume 1592, 1999.

[5]. https://www.centos.org/docs/5/html/5.1/Deployment.../s3-openssh-rsa-keys-v2.html

[6]. https://www.ijecs.in/index.php/ijecs/article/download/1527/1410/

