Volume 3, Issue 4, April 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Data Storage Security Using Partially Homomorphic Encryption in a Cloud

Sunanda Ravindran *
Student, MCA III-II
Department of Computer Applications
Sreenidhi Institute of Science and Technology, Hyderabad,India

Parsi Kalpana
Assistant Prof.
Department of Computer Applications
Sreenidhi Institute of Science and Technology, Hyderabad,India

Abstract— Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility over a network. Traditionally organizations stored and maintained data in their own data centres, over which they had complete control. With the emergence of cloud computing organizations can store data in the cloud provider's data centre, but the security of the data in the cloud is a major concern. Data can be stored in the cloud in an encrypted format, but the problem is that while data can be sent to and from a cloud provider's data centre in encrypted form, the servers that power a cloud can't do any work on it that way. With homomorphic encryption, a company could encrypt its entire database and upload it to a cloud and it is possible to analyse data without decrypting it. Homomorphic encryption is a form of encryption which allows specific types of computations to be carried out on ciphertext and obtain an encrypted result which is the ciphertext of the result of operations performed on the plaintext. In this paper we are describing two multiplicative homomorphic cryptosystems, how they differ and how we can use them to secure our data and perform operations on the data with the help of ciphertexts.

Keywords— Cloud computing, Homomorphic encryption, RSA, ElGamal, Cloud security

I. INTRODUCTION

In order to implement new ideas we need business applications. These applications are costly, in order to keep these applications running we need data centres which occupy a lot of office space, we also have to maintain these servers, in terms of hardware and software. The software has to be upgraded from time to time in order to improve the functionality of the application. Cloud computing is a better way to run our applications; the applications are hosted on the cloud. When we use any application on the cloud, we just have to login, customize and use it. We can also develop new applications and make it available to the public. The applications can be up and running in a few days, also the cost of development and maintenance of the application is less. Cloud computing uses the concept of multi-tenancy where there is one application and many users can customize and use this application along with the custom build applications. Upgrades and maintenance is taken care by the cloud service provider. While yesterday's business applications required thousands, if not millions, of dollars and sometimes years of professional services help to set up and customize, the technologies offered by the Internet today make it much easier to create, configure, and use business applications of all kinds. Indeed, the power of the Internet has given us the ability to solve new kinds of business problems that, because of complexity or cost, had previously remained out of reach. Cloud Computing is an Internet-based technology. The increasing network bandwidth and reliable yet flexible network connections in the recent past makes it possible for users to use the power of cloud computing in order to subscribe to services and to utilize the cloud environment to store data.

A. Cloud computing service models

In cloud computing, everything is delivered as a Service (XaaS). There are three service models:

1) Software as a service (SaaS): Cloud-based applications—or software as a service (SaaS)—run on distant computers "in the cloud" that are owned and operated by others and that connect to users' computers via the Internet and, usually, a web browser.

Some of the Cloud service providers for SaaS: SalesForce CRM, Citrix's Goto Meeting, Google Docs, Cisco's WebEx.

2) Platform as a service (PaaS): Platform as a service provides a cloud-based environment with everything required to support the complete lifecycle of building and delivering web-based (Cloud) applications—without the cost and complexity of buying and managing the underlying hardware, software, provisioning and hosting.

Some of the Cloud service providers for PaaS: Google App Engine, Force.com, MS Azure, Engine Yard.

3) Infrastructure as a service (IaaS): Infrastructure as a service provides companies with computing resources including servers, networking, storage, and data centre space on a pay-per-use basis. Some of the Cloud service providers

for IaaS: Amazon EC2, SQL Azure, VMWare, Amazon EC2: Elastic Cloud Computing, Amazon S3: Simple Storage Service.

Moving data into the cloud offers great convenience. While these internet-based online services do provide huge amounts of storage space and customizable computing resources, the user loses control of the data and he/she is at the mercy of the cloud provider. In order to secure our data we can encrypt our data. 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. In this paper we focus on two main multiplicative homomorphic cryptosystems namely, RSA [1] and ElGamal [2].

II. MULTIPLICATIVE HOMOMORPHIC CRYPTOSYSTEM

The idea of performing simple computations on encrypted messages was first introduced by Rivest, Adleman, and Dertouzous [3]. The original motivation for these homomorphisms was to allow for an encrypted database to be stored by a third party and to allow the owners and other authorized people to perform calculations with the data without decrypting it.

A multiplicative homomorphic cryptosystem has an encryption function E that satisfies the following property:

$$E(M1) * E(M2) = E(M1*M2)$$

where M1 and M2 are plain text messages

Some of the applications of homomorphic encryption are: Cloud computation, Electronic voting, Data mining, Financial transactions, Electronic cash, Medical records.

III. RSA CRYPTOSYSTEM

The RSA cryptosystem is the most widely used public-key cryptosystem. It was developed in the year 1978 by Rivest, Shamir, and Adleman. It is one of the first homomorphic encryption schemes.

RSA Cryptosystem:

Key Generation: KeyGen(p, q)
Input: Two large primes – p, q

Compute $n = p \cdot q$
$\varphi(n) = (p - 1)(q - 1)$
Choose e such that $\gcd(e, \varphi(n)) = 1$
Determine d such that $e \cdot d \equiv 1 \bmod \varphi(n)$

Key:
public key = (e, n)
secret key= (d, n)

Encryption:
$c = m^e \bmod n$
where c is the cipher text and m is the plain text

RSA has a multiplicative homomorphic property i.e., it is possible to find the product of the plain text by multiplying the cipher texts. The result of the operation will be the cipher text of the product.

Given $c_i = E(m_i) = m_i^e \bmod n$, then

$$\mathbf{(c1 \cdot c2) \bmod n = (m_1 \cdot m_2)^e \bmod n}$$

Example:

- Choose p = 3 and q = 11
- Compute n = p * q = 3 * 11 = 33
- Compute φ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
- Choose e such that 1 < e < φ(n) and e and n are co-prime. Let e = 7
- Compute a value for d such that $e \cdot d \equiv 1 \bmod \varphi(n)$. One solution is d = 3
- Public key is (e, n) => (7, 33)

- Private key is (d, n) => (3, 33)

Example of homomorphic property:

Now, let m1=2 and m2=3
$c1 = m_1^e \bmod n = 2^7 \bmod 33 = 29$
$c2 = m_2^e \bmod n = 3^7 \bmod 33 = 9$
c1.c2 = 29 * 9 = 261

By decrypting (c1.c2) we get:
$261^3 \bmod 33 = 6 = 2 * 3$

There are a number of attacks against plain RSA as described below:

- When encrypting with low encryption exponents (e.g., $e = 3$) and small values of the m, (i.e., $m < n^{1/e}$) the result of m^e is less than the modulus n. In this case, ciphertexts can be easily decrypted by taking the eth root of the ciphertext over the integers.
- If the same clear text message is sent to e or more recipients in an encrypted way and the receivers share the same exponent e, but different p, q, and therefore n, then it is easy to decrypt the original clear text message via the Chinese remainder theorem [4].
- An attacker can encrypt likely plaintexts and compare them with the ciphertexts.

The length of the plaintext that can be encrypted using RSA is limited to the size of n. RSA Laboratories currently recommends key sizes of 1024 bits for corporate use and 2048 bits for extremely valuable keys like the root key pair used by a certifying authority [5].

IV. ELGAMAL CRYPTOSYSTEM

The ElGamal encryption system is an asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie–Hellman key exchange. It was developed by Taher Elgamal in 1984.

ElGamal is randomized, the trivial attack does not work, which is one of its advantages over RSA. Also, ElGamal uses a smaller key length when compared to RSA.

.

ElGamal Cryptosystem:

Generate a key pair:

- Select a large prime p and the generator g of a multiplicative group Zp^* of the integers modulo p.
- Select an integer X_A from the group Z by random and with the constraint $1 \leq X_A \leq p-2$.
- Private key: X_A
- Public key: $Y_A = g^{X_A} \bmod p$

Encryption procedure:

- Select a random exponent X_B
- Compute $c1 = g^{X_B} \bmod p$ and combine it with the ciphertext that shall be sent to the

ElGamal also has a multiplicative homomorphic property.

Given ciphertexts (c1, c2) and (d1, d2) that are encryptions of m1 and m2, using random values X_{B1} and X_{B2}, respectively, then

$$\mathbf{(c1d1, c2d2) = (g^{X_{B1}} g^{X_{B2}} , (m1 \cdot S_1)(m2 \cdot S_2)) = (g^{X_{B1}+X_{B2}} , m1m2 \cdot S_1+S_2)}$$

is a valid encryption of m1m2.

Example:

p=139, g=3
$X_A = 12 \quad Y_A = g^{X_A} \bmod p = 3^{12} \bmod 139 = 44$

$X_{B1} = 52$

Encryption:

- Compute $c1 = g^{X_{B1}} \bmod p = 3^{52} \bmod 139 = 38$
- Compute $S = (Y_A)^{X_{B1}} \bmod p = 44^{52} \bmod 139 = 112$
- Cipher text = (m1 * S) mod p
  Assume m1=2
  c2 = (2 * 112) mod 139 = 85

Decryption:

- Compute $R = (c1)^{p-1-X_A} \bmod p = 36$
- Decryption
  m = (R * c2) mod p = (36 * 85)mod 139 =2

Similarly, if we take X_{B2} as 62 and m2 as 7, we will get

$d1 = g^{X_{B2}} \bmod p = 3^{62} \bmod 139 = 124$

$S = (Y_A)^{X_{B2}} \bmod p = 44^{62} \bmod 139 = 64$

d2 = (7 * 64)mod 139 = 31

Homomorphic multiplication

(c1d1,c2d2) = (4712,2635)

Decryption: $((c1d1)^{p-1-X_A} \bmod p)*(c2d2) \bmod p = 44*2635 = 14 = (m1 * m2)$

The ElGamal cryptosystem provides the homomorphic operation of multiplication of two encrypted messages, as well as multiplication by a known constant and exponentiation by a known constant. But, in most applications homomorphic operation of addition will be beneficial; hence we can modify ElGamal cryptosystem to provide homomorphic addition.

$$\mathbf{(c1,c2) = (g^{X_B} \bmod p, g^m h^{X_B} \bmod p)}$$

where $h \equiv g^{X_B} \bmod p$ and m is the plaintext

The Homomorphism: Let m1 and m2 be plaintexts. Then,

$$E(m1, X_{B1})E(m2, X_{B2}) = (g^{X_{B1}} \bmod p, g^{m1}h^{X_{B1}} \bmod p)\,(g^{X_{B2}} \bmod p, g^{m2}h^{X_{B2}} \bmod p)$$
$$= (g^{X_{B1}+X_{B2}} \bmod p, g^{m1+m2}\, h^{X_{B1}+X_{B2}} \bmod p)$$
$$= E(m1+m2, X_{B1}+ X_{B2})$$

V. CONCLUSION

In this paper we have described what cloud computing is and how we can benefit from it. As every technology has a flaw so does cloud computing. Security is a major concern in cloud computing as our data is stored in a cloud and it becomes very difficult to perform operations on the encrypted data, hence we can use homomorphic encryption to secure our data and also perform operations on it. We have described the RSA cryptosystem and the ElGamal cryptosystem and also described how they can be used to perform calculations. Currently the homomorphic cryptosystems can be used to perform only certain operations like addition, subtraction, multiplication, XOR and exponentiation. We can enhance these algorithms to perform various other operations. With homomorphic encryption, a company could encrypt its entire database of e-mails and upload it to a cloud. Then it could use the cloud-stored data as desired—for example, to search the database to understand how its workers collaborate. The results would be downloaded and decrypted without ever exposing the details of a single e-mail.

REFERENCES

[1] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. *A method for obtaining digital signatures and public-key cryptosystems*. Commun. ACM, 21(2):120–126, 1978.

[2] Taher El Gamal. *A public key cryptosystem and a signature scheme based on discrete logarithms*. In G. R. Blakley and David Chaum, editors, CRYPTO 1984, volume 196 of Lecture Notes in Computer Science, pages 10–18. Springer, 1984.

[3] 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.

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7

[5] RSA Laboratories. [Online]. Available: .http://www.rsa.com/rsalabs/node.asp?id=2218

[6] John W. Rittinghouse, James F. Ransome, *Cloud Computing Implementation, Management, Security*, CRC Press 2009 by Taylor and Francis Group, LLC.

[7] http://en.wikipedia.org/wiki/Homomorphic_encryption

[8] Brian Hayes (2012) A new form of encryption allows you to compute with data you cannot read. American Scientist archive, September – October 2012, Volume 100, Number 5 [Online]. Available: http://www.americanscientist.org/issues/pub/2012/5/alice-and-bob-in-cipherspace/3.

