ISSN:0975-9646

Secure Computation in Cloud Environment

Fazail Amin

USICT, GGSIPU

Dwarka, New Delhi, India

Abstract— **This paper presents an overview of security issues in cloud environment and use of fully homomorphic encryption scheme to provide blindfold computation of data. Though there are various homomorphic encryption schemes available but most of them are designed to work in ideal conditions only and not all schemes are fully homomorphic. In this paper we present a secure banking model for outsourcing banking services into cloud, which will enable the banks to trust the services of cloud without the need to worry about the security of customer data.**

Keywords— **Homomorphic encryption, adaptive cipher text attack, PIR (personal information retrieval system) .**

I. Introduction

With recent advances in cloud computing environment it has now become very efficient and affordable to use it. Now it is used by almost all organizations because it relieves them form the issues related software and hardware, they just use what they want and pay for it accordingly. To make cloud environment acceptable to the customers it must meet the requirements of security and confidentiality of the user data in the cloud and over the communication network as well.

Currently the privacy of data can be ensured wih the existing techniques like DES,AES, IDEA or any other secure encryption scheme during the data acquisition and storage. To exploit full advantage of cloud we need the ability to do the computations on the cloud. This poses a high risk to the security of data because to do the computations on the data it must be extracted form storage and the decrypted as well to perform the computations. This decryption requires the key to put through the network which is considered to be very insecure and makes data vulnerable to active as well as passive attacks.

II. Cloud computing security issues

1) *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 [3]. The cloud service provider is required to provide sufficient security policy that guarantees the use of efficient activities are being deployed to mitigate the risk to the data when a user outsources the data to the cloud. This poses another risk as the security is here based on trusting the processes and the computing base implemented by the cloud owner.
2) *Confidentiality*: Data confidentiality in the cloud means isolating the data of individual users from one another. Data confidentiality can be breached, due to data persistence. Data remanence is the residual data that has been partially erased or removed. Due to virtual separation of logical drives and lack actual physical hardware separation between multiple users on a single infrastructure, data remanence may lead to the undesirable leakage of private data. Also a malicious user, may claim a large amount of disk space and then scavenge for sensitive data, which can lead to unprecedented loss of data
   Software confidentiality is also important aspect to meet overall system security. It refers to trusting that specific applications or processes will maintain and handle the users data in a secure manner[3].
3) *Privacy:* privacy refers to the willingness of a user to control the disclosure of private information. The cloud presents a number of legal challenges towards the privacy breach due to fuzzy perimeter of the cloud environment and laws controlling the privacy in various countries differ.
4) *Integrity:* Integrity is associated with data, software and hardware and it maintains 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.

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.

III. Homomorphic encryption

A. *Overview of homomorphic encryption*

The basic idea of blindfold computation was originated when computation on encrypted data was first proposed by Rivest, Shamir, Adleman and Dertouzos in 1978 in[4]. They proposed exponentiation function and the RSA function as additive and multiplicative homomorphics, respectively. However these were not able to provide good security. There are a number of cryptographic systems which are either multiplicative or additively homomorphic but not both. The first semantically secure homomorphic encryption scheme was developed by Goldwasser and Micali [5].

Craig gentry was the first person to come up with a plausible construction of a fully homomorphic encryption scheme in 2009 [6]. It enabled the computation of any arbitrary function on encrypted data and produced compact cipher texts. Gentry's encryption was based on ideal lattices.

Homomorphism is a property by which a problem in one algebraic system can be transformed to a problem in another algebraic system, and after solving it, the solution can also be transformed back effectively. Hence, cryptographic schemes with homomorphic properties would suffice the need of security as well as preserve the system usability in clouds[2].

In a layman's language homomorphic encryption scheme refers to the function which performs binary operation on encrypted data without decrypting it. That is a function that enables us to compute the binary operation on the plaintexts by only manipulating the ciphertexts without any knowledge of the encryption key : $E(x_1) * E(x_2) = E(x1 \otimes x2)$[1].

Definition of HE : A homomorphic public key encryption scheme HE is a probabilistic polynomial time algorithm as shown below[9]:

$HE=(H_{keygen} , H_{enc} ,H_{dec} , H_{eval})$

- key generation: H_{keygen} is algorithm that generates pubik key, evaluation key and secret decryption key.
- Encryption: H_{enc} is algorithm takes public key and and a single bit message $\mu \in \{0,1\}$ and outputs a ciphertext c.
- Decryption: H_{dec} is algorithm that takes the secret key and encrypted data c and outputs a message $\mu^* \in \{0,1\}$.
- Homomorphic Evaluation:
  $C_f \leftarrow H_{eval}(f,c_1,.......c_l)$

  H_{eval} is algorithm that takes the evaluation key and and a function $f : \{0,1\}^l$ and a set of l ciphertexts $c_1,......c_l$ and outputs a ciphertextf c_f.

Compactness: A homomorphic scheme HE is compact if there exists a polynomial $s = s(\kappa)$ such that the output length of $H_{Eval}(\ )$ is at most s bits long (regardless of f or the number of inputs). In other words, compactness requires that the size of the ciphertext after homomorphic evaluation does not depend on either the number of inputs or the complexity of the function *f*, but only on the size of the output of *f* [9].

Here we note that homomorphic encryptions schemes do not make any difference between public and private keys. That is both the private and the pulic key can be used for the either purpose.

B. *Gentry's FHE scheme*

Gentry's construction has three components: a "somewhat homomorphic" encryption scheme that can evaluate a limited class of functions, a method of "bootstrapping" a sufficiently powerful homomorphic encryption scheme – called a "bootstrappable" encryption scheme into a fully homomorphic encryption scheme and finally, a specialized method of turning the somewhat homomorphic scheme into a bootstrappable scheme[9].

Step 1: The first step is to construct a somewhat homomorphic encryption (SWHE) scheme, namely an encryption scheme capable of evaluating "low-degree" polynomials homomorphically.

Step2: Bootstrapping
The somewhat homomorphic encryption (SWHE) scheme is only able to evaluate "low degree" polynomials, falling well short of the eventual goal of fully homomorphic encryption (FHE). To obtain FHE, Gentry provided a bootstrapping theorem which states that given an SWHE scheme that can evaluate its own decryption function, one can transform it into a "leveled" FHE scheme, in a completely generic way. Such an SWHE scheme is called a bootstrappable encryption scheme. If we assume that it is safe to encrypt the leveled FHE secret key under its own public key, a requirement that is referred to as "circular security" then the transformation gives us a "pure FHE scheme. Bootstrapping "refreshes" a ciphertext by running the decryption function on it homomorphically, using an encrypted secret key (given in the evaluation key), resulting in a reduced noise.

Step3 : Squashing
Squashing the decryption circuit : the final step is to squash the decryption circuit of the SWHE scheme, namely transform the scheme into one with the same homomorphic capacity but a decryption circuit that is simple enough to allow bootstrapping. This is done by adding a "hint" about the secret key to the evaluation key. The hint is a large set of elements that has a secret sparse subset that sums to the original secret key. In order to ensure that the hint does not reveal damaging information about the secret key, the security of this transformation relies on a new "sparse subset sum" assumption. The sparsity pushes the decryption complexity at the cost of the additional assumption.

C. *Applications of FHE*

There are number of applications where FHE can be implemented, some applications are explained below:

- Fully homomorphic encryption can provide secure search engine queries, where the search provider does not know what is being searched for.
- Use of FHE in spam filtering, where the spam filtering agent would not need access to the plain text email at all.
- Another use for fully homomorphic encryption is that of secure voting, in which each voter is guaranteed to only vote once, and the ballots can be tallied without any individual ballot being decrypted, ensuring the privacy of voting. Cohen and Fischer proposed an additively homomorphic encryption scheme based on higher

order residuosity, and showed how to use it to perform secure electronic voting. This proposal and its descendants have made its way to modern day web-based voting systems such as Helios.

- The trapdoor functions(lossy) developed by Peikert and Waters is constructed from additively homomorphic encryption schemes (with some extra properties), which they in turn used to construct chosen ciphertext secure (CCA-secure) public-key encryption schemes.
- Private Information Retrieval (PIR) protocols. An important application of homomorphic encryption comes from the work of Kushilevitz and Ostrovsky who showed how to construct (single-server) Private Information Retrieval (PIR) protocols with sub-linear communication, from any additively homomorphic encryption scheme. Private Information Retrieval, defined by Chor, Kushilevitz, Goldreich and Sudan, is the problem wherein a user attempts to retrieve the i th item in a database of size N, revealing no information about the index i to the database owner .

IV. PROPOSED WORK

A. *Introduction:*

Outsourcing the banking operations to cloud platform

Most important aspect of banking is privacy because banks have confidential information about their customers which are very valuable to banks and other competitors. Insecurity about information leakage is one thing and securely processing the data (that is banking transactions) is another important aspect which the cloud service provider must provide without failure. Banking operations are data intensive they access and process huge volumes of data on regular basis. So what we need is a secure environment which can provide platform for secure computation remotely without leaking any information about the data being processed. Also the access to the database must also not leak any information about the data being accessed.

Keeping in view the above mentioned requirements a fully homomorphic encryption scheme can achieve the goals if it can be implemented efficiently. Currently many FHE schemes are available but the efficiency remains the major issue.

B. *Overview of the banking model:*

- User authentication system
- Data access (controlling the access to data)
- Banking operations (this includes the various transactions)

1) User authentication system:

A secure user authentication system is needed to provide authorized access to data. For this purpose a public key system can be used to set up secure communication between user and the cloud server.

A security scheme as proposed by Zissis and Lekkas in [3] can be used. This approach makes use of a combination of public key cryptosystems, single sign-on feature of Shibbeloth and LDAP (lightweight directory access protocol). A TPP (trusted third party) can be relied upon for:

- Low and high level confidentiality
- Server and client authentication
- Generating security domains
- Cryptographic separation of data
- Certificate based authorization

2) Data access:

This is very important aspect of banking service, the cloud service provider must not be able to get any information about which data from database is being accessed, that is provider shoud get information about the index being searched for in the database. This can be achieved by using PIR (personal information retrieval) protocol which makes use of FHE to search for the index.

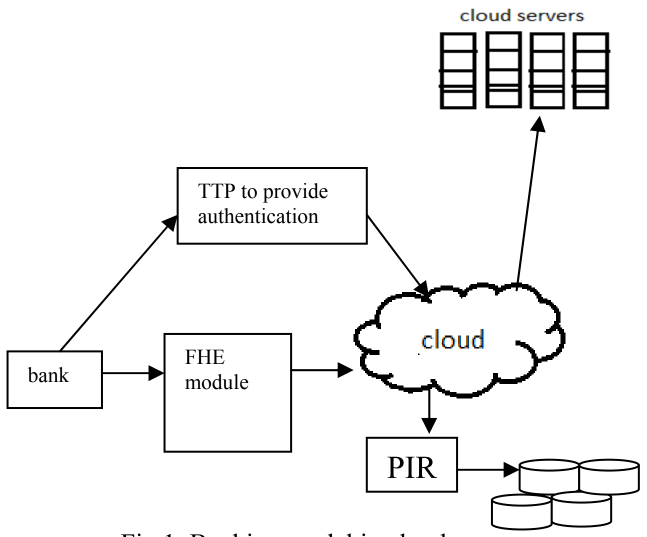

Fig.1. Banking model in cloud

3) Banking operations:

The data should not be in plaintext format at any intermediate level and the user should be able to process the data without decrypting it. User encrypts the data before sending it to the cloud and if any operation needs to be done on the data it should be done without decryption. This will make sure that all the transactions of the bank run without leaking any information about the data being processed. This can be achieved efficient Fully Homomorphic Encryption scheme.

V. FUTURE WORK AND CONCLUSION

FHE has a huge scope for providing security to distributed systems like cloud environment. In current scenario the main issue with the implementation of FHE for providing secure computation is only bounded by its efficient implementation beyond idle cases. Currently we have devised and tested individual algorithms, in future effort will be made to integrate and implement the proposed banking model.

REFERENCES

[1] Shubha Bharil, T. Hamaspriya, Praveen Lalwani. "*a secure key for cloud using threshold cryptography in Kerberos*". International journal of computer applications, Volume 79-no7 oct 2013

[2] Yugi Suga, Jinbocha Mitsui, " *A fast(2,2^m) threshold secret sharing scheme using m linearly Independent binary vectors".* 16th International conference of network based information systems 2013 IEEE.

[3] Dimitrios Zissis, Dimitrios Lekkas, *" Addressing cloud computing security issues",* Future generation computer systems 28 (2012). Pages 583-592.

[4] R.Rivesst, L. Adleman, M. Dertouros,"*on data naks and privacy homomorhism*",Int. foundations of secure computation,Academic press,1978, pg 169-177.

[5] S. Goldwasser and S. Micali, "*Probabilistic encryption,*"Journal of Computer and System Sciences, vol. 28, no. 2,pp. 270–299, 1984.

[6] C. Gentry, "*A fully homomorphic encryption scheme*," Ph.D. dissertation, Stanford University, 2009.

[7] Zhu Ping, Guang Xiang, " *the protection methods for mobile code based homomorphic encryption and data confusion*", 2011 international conference on management of e-commerse and e-government 2011

[8] Chiang Chia-Chu, Hayward Ryan*," An architechture for parallelizing fully homomorphic cryptography on cloud* ", 2013 seventh international conference on complex, intelligent and software instensive systems 2013

[9] Vaikunthnathan V. , "*computing blindfolded: new developments in fully homomorphic encryption*", 2011 IEEE 52nd Annual symposium on foundations of computer science 2011.

[10] Alejandro Llamas and Ra´ul Ernesto Gonz´alez, "*A Cryptographic Scheme for Secure Cloud Computing*". 2013 10th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE) Mexico City, Mexico. 2013

[11] http://docs.cloudstack.apache.org/en/latest/ "introduction to open cloud stack

[12] https://gmplib.org/manual/Introduction-to-GMP.html#Introduction-to-GMP