Journal of Theoretical and Applied Information Technology
30th June 2016. Vol.88. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MULTI-PARTY PROTOCOL WITH ACCESS CONTROL ON SYMMETRIC FULLY HOMOMORPHIC ENCRYPTION SCHEME

[1]WAMDA NAGMELDIN, [2] SITI MARIYAM SHAMSUDDIN

[1, 2] Department of Computer Science, Faculty of Computing, Universiti Teknologi Malaysia

81310 UTM Skudai, Johor Bahru, Malaysia.

E-mail: [1] wamda_82@hotmail.com , [2] mariyam@utm.my

ABSTRACT

Homomorphic encryption is a particular class of encryption presented by Rivest, Adleman et al. in 1978 that permits mathematical operations on the encrypted data without decrypting it [1], in fact, without knowing the decryption key. In the last few years, homomorphic encryption techniques have been studied deeply since they have become more and more vital and important in several different cryptographic applications such as lottery protocols, voting protocols, anonymity, privacy, and electronic auctions.This paper introduces the symmetric fully homomorphic scheme (Sym-FHE) and multi-party protocol with access control to allow many users access and manipulate the data in the cloud without violating the confidentiality of sensitive data.

Keywords: *Homomorphic Encryption, Cloud Computing, Multi-party*.

1. INTRODUCTION

Cloud computing is a particular kind of computing that depends on sharing computing resources instead of having local servers or individual devices to manipulate applications. In cloud computing, clients use software and hardware resources to send data through the internet to access more sophisticated applications. The recipient has to be trustworthy because he deals with personal data of other users.

The name came from the symbol that looks like a cloud which represents complicated system format. The idea is to use this technology which has long been utilized by the armed forces to send extensive data and perform a huge number of operations within a second, to make such service available for financial institutions, to save personal data or to power computer games.

According to the National Institute of Standards and Terminology NIST [2]: the following are the most important features of cloud computing:

- On-demand self-service: consumers can use time and storage capabilities without interference from the service provider.
- Broad access: available for mobile and laptop users.
- Resource polling: resources are added to each other to provide the service for users with various resources and multiple ever-changing tasks.
- Rapid elasticity: (very flexible) in a way that responds to the needs of the users in a customized and timely manner.
- Measured service: all service types (storage, user accounts and bandwidth) are measured by the metering feature so that the consumer and the service provider can control and monitor the use of data.

Cloud computing enables exchanging of services and concentrates on increasing the effectiveness of the shared resources. In cloud computing, users save their data in the cloud, and any operation on saved data will be executed on the cloud instead of the user device. Therefore, the cloud server providers must supply guarantees on the protection of privacy and private data saved on their data centers shared between numerous of users. The data must be encrypted on cloud and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

allow to manipulate with it using the concept of homomorphic encryption.

2. HOMOMORPHIC ENCRYPTION

Homomorphic encryption systems used to save the sensitive data in the cloud. HE systems deal with data without breaking the privacy of the user. These systems do not decrypt the data of the client because they do not use his secret key. The encryption is called homomorphic when users can compute functions (addition, multiplication, etc.) of encrypted data without using the secret key.

Definition: An encryption scheme is called homomorphic if: from $\mathrm{Enc}(x)$ and $\mathrm{Enc}(y)$ it is possible to calculate $\mathrm{Enc}(f(x,y))$, where f can be: $+, \times, \oplus$ and without using the secret key. [3]

The idea of homomorphic encryption (privacy) introduced by Rivest in [1]. The HE scheme is a cryptosystem that allows computation operations to be performed on encrypted data without decrypting it. Plenty of encryption systems allow partial homomorphic encryption (support only addition or multiplication operation on encrypted data). The issue of fully homomorphic encryption (FHE) supporting both addition and multiplication with an unlimited number of operations had fundamentally remained open until IBM researcher Craig Gentry came up with the first fully homomorphic encryption scheme in 2009. Gentry in [4] showed the first fully homomorphic encryption system based on ideal lattices. He developed a two steps approach. In the first step, Gentry invented a Somewhat Homomorphic encryption scheme SHE, which an encryption scheme that satisfies the homomorphism requirement. However, arithmetical manipulation on encrypted data produces false results; thus decryption of the encrypted data remains valid only if the number of algebraic operations that performed on the encrypted data are limited. In the second step, Gentry developed a bootstrapping technique that cleans the accumulated errors. A combination of these two steps enables a construction of Fully Homomorphic Encryption scheme (FHE).

Many other Homomorphic Encryption Schemes suggested after Gentry's success Brakin2011 published an FHEsystemconstructed on the Learning with Errors (LWE) problem [5]. The computational complexity of the mentioned schemes motivated research aimed at improving their efficiency [6]. An approach to prove efficient homomorphic encryption with practical application to real-world problems described in [7]. The method based on somewhat homomorphic encryption supporting a limited number of homomorphic operations. The practical results in [8] based on the use of the somewhat homomorphic scheme proposed in [9]. This method first set out to satisfy the requirement for efficiency (practicality), and then found the solution in the SHE schemes.

This paper presents symmetric fully homomorphic encryption (sym-FHE) scheme which is practical albeit mainly applicable to cryptographic and security algorithms. The scheme based on matrix operations. It has a small key size wherefore it is suitable for several data-centric applications. The system based on [10] and [11] it extracts its security from the hardness of factorizing a large integer, which is the basis of several public key cryptosystems.

2.1 Types of Homomorphic Encryption

Homomorphic encryption scheme has the property such that executing a function on two values individually and then encrypting the result produces the same final value as first encoding two values separately and then performing the function to the final results. According to the type of operation that supported by homomorphic encryption scheme, there are two types of homomorphic encryption.

2.1.1 Partially homomorphic cryptosystems

Partially homomorphic scheme allows homomorphic computation of only one operation (either addition or multiplication) on ciphertexts. There are several efficient partially homomorphic cryptosystems. RSA[1]and ElGamal [12] schemes are multiplicatively homomorphic cryptosystems. while Goldwasser-Micali [13] in 1984, Paillier [14]in 1999, Benaloh[15], Naccache-Stern[16], Okamoto-Uchiyama[17], Damgard-Jurik[18], and SCHMIDT-SAMOA-TAKAGI[19]'s schemes are additively homomorphic cryptosystems.

2.1.2 Fully homomorphic encryption scheme

Fully homomorphic encryption scheme can solve an essential open problem in cryptography. This issue is related to dealing with the encrypted data without any idea about the decryption key. In the traditional encryption schemes, without the decryption key, the ciphertext is completely ineffective and useless. This issue has raised many

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

questions and was a source of controversy for a long time. In 1978 Rivest, Adleman and Dertouzos were the pioneers who posed the question: Can we do arbitrary computational operations on data while it remains encrypted, without ever decrypting it? [1]

In 2009, Gentry proposed a significant breakthrough which showed the first construction of a fully homomorphic encryption scheme [4] that permits computation of arbitrary functions on encrypted data and produces a short ciphertext.

A fully homomorphic encryption scheme is a quadruplet of algorithms involves (KeyGen, Enc, Dec, Eval) such that:

Key$G(\lambda)$: Is a key generation algorithm, which takes a security parameter λ as an input and produces a public and secret keys (pk,sk)as an outputs.
$E(m,pk)$: Is an encryption algorithm, takes a plaintext m and a public key pk as input and produces a ciphertext c as an outputs.
$D(c,sk)$: Is a decryption algorithm, takes a ciphertext c and a secret key sk as an input and produces a plaintext m as outputs.
$Eval(C, c_1, \dots, c_n)$: Is an evaluation algorithm, takes a circuit C and ciphertexts $c_1, \dots, c_n$ as inputs and verify $D(Eval(C, c_1, \dots, c_n), sk)=C(m_1, \dots, m_n)$. Anyone can perform evaluation algorithm, since it does not require the secret key sk.

There are two categories of fully homomorphic encryption cryptosystems symmetric and asymmetric.

- **Symmetric fully homomorphic encryption**

In this type, key Generation algorithm produces one key which used in both encryption and decryption processes.

- **Asymmetric fully homomorphic encryption**

This type works with two keys, public and private key. The first fully homomorphic scheme constructed by Gentry [4] was asymmetric. Many of schemes followed Gentry's way to design asymmetric cryptosystems. Now, the number of existing asymmetric fully homomorphic cryptosystems exceeds it's of symmetric fully homomorphic cryptosystems.

2.2 Gentry's Fully Homomorphic Scheme

Gentry constructed a fully homomorphic encryption scheme using ideal lattices. He showed that a fully homomorphic encryption scheme can be constructed in three phases:

- Somewhat homomorphic encryption scheme (SWHE) which supports a limited number of multiplications operations on the ciphertext.

- Squashing technique of the decryption circuit "the encrypter starts decryption". This step cost Gentry a security assumption is the assumption of the robustness of the Sparse Subset Sum Problem (SSSP).

- Bootstrapping technique to obtain a scheme called 'leveled' FHE for evaluating any circuit with a depth defines the circuit at the start. The scheme becomes bootstrappable, and therefore it can be converted simply into a fully homomorphic cryptosystem.

2.3 Related Works

There are many schemes followed Gentry's work to design an efficient fully homomorphic cryptosystem. Some of these works described as the following subsections.

2.3.1 The Smart and Vercauteren scheme

The first attempt to implement the original scheme was made in 2010 by Smart and Vercauteren [20]. They specialized Gentry's scheme a particular way and produced a scheme which is more efficient and conceptually easier to understand by removing all the need to understand lattice. They defined a decryption technique where the secret key is symbolized using a single integer instead of the lattice.

They were able to implement the somewhat homomorphic scheme. Nevertheless, their scheme has a limitation to support large enough arguments to obtain bootstrapping scheme using Gentry's squashing approach. The advantage of this cryptosystem is its small key size and ciphertexts sizes compared to Gentry's cryptosystem.

2.3.2 Chunsheng's scheme:

More practical fully homomorphic encryption presented by Chunsheng in [21] using the same Smart and Vercautern's algorithms and constructed a new refreshment algorithm following Gentry's

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

framework nonetheless without using the SSSP assumption.

2.3.3 Faster fully homomorphic encryption

In 2010 Stehle and Steinfeld presented two enhancements of Gentry's scheme: first they posed other conditions when selecting SSSP settings to enhance the semantic security of the algorithm. Secondly, they introduced a probabilistic decryption algorithm which can be applied with a small multiplicative degree. Combined the two enhancement together, produce a faster fully homomorphic scheme [22].

2.3.4 SIMD Gentry, 2011

The Gentry's scheme performs operations on a 1-bit length. For that reason, it is intuitive to think that some operations may be executed on several bits in parallel to decrease the runtime. In [23], Smart and Vercauteren show how to select parameters for the implementation of SIMD operations (Single Instruction, Multiple Data) for the Gentry and Halevi's scheme [24].

2.3.5 Gentry-Halevi's scheme

Gentry-Halevi used a hybrid approach to constructing FHE scheme [24]. This method based on the utilization of SWHE algorithm and another algorithm (multiplicatively homomorphic Encryption) as ElGamal to evaluate the multiplication.

2.3.6 The DGHV's scheme 2010

Van Dijk, Gentry, Halevi and Vaikuntanathan constructed a fully homomorphic encryption over the integers [25]. They used a new approach to design FHE. The significant contribution of this scheme is its simplicity. All operations are done on integers rather than lattices.

2.3.7 The BV's scheme:

An efficient fully homomorphic encryption using learning with error problem LWE presented in [26]. Brakerski and Vaikantanathan introduced a new approach to design FHE schemes; it's the re-linearization technique. This method permits making fully homomorphic SWHE scheme and using modulus switching as a noise management technique, so no need to use the Gentry's bootstrapping technique. This scheme decreases the size of ciphertexts and the complexity of the decryption algorithm.

2.3.8 The BGV's scheme:

Following the same concept of [26], Brakerski, Gentry and Vaikantanathan [27] construct the BGV scheme. The main contribution of this scheme is the enhancement of the technique of modulus reduction to better manage noise in ciphertexts.

2.3.9 Chen-Wang-Zhang-Song's scheme :

This scheme [28] based on a new different of LWE problem named binary-LWE (BLWE). To decrease the size of the noise, the authors select the secret key from $\{0,1\}^n$rather than using the binary decomposition of this key as in the Brakerski's scheme [29].

2.3.10 The Xiao-Bastani-Yen's scheme

All of mentioned above FHE schemes are noise-based constructions. To decrypt correctly, should remove this noise from the ciphertext. To scale down the noise on the ciphertexts, the noise management techniques are used. These refreshment techniques impact negatively the performances of FHE schemes after implementation. The researchers work to enhance the performance of the FHE and designed a free-noise FHE schemes. The first noise-free fully homomorphic encryption scheme introduced by Xiao et al. [10] in 2012. It is a symmetric encryption scheme. The security of the scheme established on the factoring problem. Homomorphic properties are obtained via matrices operations.

2.3.11 Li-Wang's scheme

Li and Wang [30] proposed a free-noise symmetric FHE scheme using the non-commutative rings. The scheme is very similar to the Xiao-Bastani-Yen's [10]. The purpose of using non-commutative rings was to overcome the attack suffered by its similar previous cryptosystem. Recently in [31], Gjosteen and Strand introduced a new attack on Li-Wang cryptosystem. This means that Li-Wang scheme is not secure today.

2.4 Applications of Homomorphic Encryption

Homomorphism allows querying, retrieving and working on encrypted data in the cloud and used in electronic voting where the privacy is maintained while data is computed by a third party. HE also employed in data mining where a third party can perform data mining on encrypted data. In financial transactions can calculate credit score without seeing user finances in clear text and electronic cash transactions can be authenticated. Medical records can be sent and processed by a

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

third party while saving users secrecy using homomorphic encryption.

3. SYMMETRIC FULLY HOMOMORPHIC ENCRYPTION SCHEME

As mentioned in section 2, there are many noise-based fully homomorphic schemes and a few number of the schemes. We design a fully homomorphic scheme using a symmetric key which is a free-noise scheme. The proposed system is an efficient and practically feasible and then we present a protocol for its use in multi-party cloud or data centric applications. The essential concept is based on Xiao-Bastani-Yen's scheme to convert integer operations in a ring Z_N to operations in a ring $M_4(Z_N)$. Therefore, all operations are on square matrices of size 4, which are adequately small to be used practically. In the context of building a homomorphic scheme to be valuable enough, the following functions are defined:

- **Cryptographic operations:** Functions to generate the symmetric key, encrypting and decrypting operations.
- **Evaluation operations:** Functions to perform any arbitrary operation on encrypted data homomorphically, the scheme needs to translate into these basic evaluation operations and then evaluate it.
- **Application specific functions:** functions offer services and facilities, like functions of key translate and re-encryption, for the scheme's adaptation to an application setting.

The main idea is to establish a matrix with an eigenvalue equal to the plaintext a. This can be very simply achieved with matrices of size 2, in ring $M_2(Z_N)$, where $N = p\,q$, where p and q being two large mutually prime numbers and random value r, as the following form:

$$E(a) = \begin{pmatrix} a & 0 \\ 0 & r \end{pmatrix} \bmod N.$$

3.1 Key Generation Process

In the original key generation algorithm in [10], choose $2m$ numbers p_i and q_i, $1 \le i \le m$, which are prime and of sizeλ /2bits. We modify the key generation algorithm by removing the constraint of p and q to be prime and increase the number of probability by the p and q are mutually prime not prime numbers. Figure 1 describes the key generation process.

Keygen(m, λ)

1. Select 2m numbers $p_i$and q_i, $1 \le i \le m$, which are mutually prime and of size $\lambda/2$ bits.
2. Let $f_i = p_i q_i$and$N = \prod_{i=1}^{m} f_i$
3. Select an invertible matrix k of size 4, $k \in M_4(Z_N)$
4. Compute its inverse as k^{-1} modulo (Z_N).
5. Output $\langle f_i, N, K, k^{-1} \rangle$ as K-λtuple.

Figure 1: Key Generation Algorithm

3.2 Encryption and Decryption Processes:

Encryption and decryption operations are same as original scheme[10]. To encrypt any message a the following steps can used.

Encrypt (a, Key)

1. Select a random value $r \in Z_N$, $r \neq a$
2. Create a matrix A $(m \times 3)$ such that each row has only one element equal to a, and other two equal to r.
3. Apply Chinese Remainder Theorem set $1 \le j \le 3$ to be an answer to the simultaneous congruences $a_j = A_{ij} mod f_i$, $1 \le j \le m$.
4. $C = k^{-1} * diag(a, a_1, a_2, a_3) * k.$

To decrypt a ciphertext C a single step method used which involves applying an inverse transformation of the ciphertext matrix and extracting the original plaintext as the first element of the diagonal matrix obtained.

Decrypt(C, Key)
Output the original plaintext as $a = (kCk^{-1})$

3.3 Evaluation Process

To do any evaluation function on ciphertexts C_1 C_2, ... C_n using this Symmetric FHE scheme, there is only one general evaluation function defined for computation$f$. It is supposed that f be converted to the basic operations on matrices of integers. Namely, to perform arithmetic

Journal of Theoretical and Applied Information Technology
30th June 2016. Vol.88. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

operations on two numbers homomorphically, perform these operations of their ciphertexts simply as two matrices.

$$Y \leftarrow Eval\ (f, C_1, C_2, \dots C_n)$$

This equation performs computation function f on operands $C_1, C_2, \dots C_n$. The evaluation function doesn't require any evaluation key and all operations on matrices performed within the ring$M_4(Z_N)$.

3.4 Example

Let $m = 2$, let $p = \{2,9\}$ and $q = \{3,7\}$. This gives $f_1 = 6, f_2 = 63$ and $N = 378$. For these parameters, suppose the Key generation algorithm generates the key:

$$k = \begin{bmatrix} 339 & 104 & 81 & 33 \\ 14 & 297 & 328 & 175 \\ 164 & 266 & 33 & 62 \\ 148 & 281 & 322 & 238 \end{bmatrix}$$

The inverse of the key is

$$k^{-1} = \begin{bmatrix} 235 & 361 & 279 & 209 \\ 332 & 108 & 246 & 9 \\ 142 & 204 & 115 & 118 \\ 372 & 101 & 200 & 181 \end{bmatrix}$$

Select random number r= 184, Consider addition and multiplication of two integers 10 and 6.First, encrypt 10 and 6 using k

$$Enc(10, k) = C_1 = \begin{bmatrix} 34 & 324 & 30 & 300 \\ 18 & 190 & 54 & 288 \\ 222 & 60 & 262 & 276 \\ 240 & 264 & 312 & 28 \end{bmatrix}$$

$$Enc(6, k) = C_2 = \begin{bmatrix} 356 & 350 & 63 & 105 \\ 84 & 153 & 0 & 336 \\ 294 & 56 & 125 & 42 \\ 252 & 42 & 126 & 251 \end{bmatrix}$$

Next, adding the ciphertexts as an alternative of the original plaintext as:

$$C_1 + C_2 = \begin{bmatrix} 12 & 296 & 93 & 27 \\ 102 & 343 & 54 & 246 \\ 138 & 116 & 9 & 318 \\ 114 & 306 & 60 & 279 \end{bmatrix}$$

When we decrypt the result we have the same result of addition on the plaintexts:

$$Dec(C_1 + C_2) = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 57 & 0 & 0 \\ 0 & 0 & 57 & 0 \\ 0 & 0 & 0 & 135 \end{bmatrix}$$

The result is 16 = 10+6.
To multiply the ciphertexts we get

$$C_1 * C_2 = \begin{bmatrix} 134 & 152 & 222 & 372 \\ 66 & 216 & 324 & 48 \\ 72 & 122 & 242 & 144 \\ 12 & 156 & 192 & 224 \end{bmatrix}$$

When we decrypt the result we have the same result of multiplication on the plaintexts:

$$Dec(C_1 \times C_2) = \begin{bmatrix} 60 & 0 & 0 & 0 \\ 0 & 320 & 0 & 0 \\ 0 & 0 & 320 & 0 \\ 0 & 0 & 0 & 116 \end{bmatrix}$$

The result is $60 = 10 \times 6$

This implementation showed in figure 1 and proved that scheme is fully homomorphic.

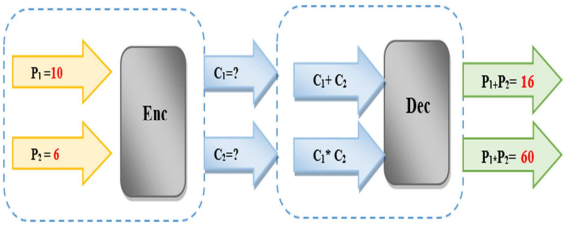

Figure 1: Symmetric FHE operations

3.4 Selecting N

Suppose a security parameter λ in situation of making the Sym-FHE scheme indistinguishable under chosen-plaintext attack (IND-CPA) secure, that is in order to holdη number of plaintext attacks we select m and λ such that$\eta = m\ ln\ poly(\lambda)$, where $poly(\lambda)$represents a fixed polynomial in λ . N is calculated as product of $2m$ numbers. Xiao et al in [10] propose all these $2m$ numbers to be unique and prime which is modified in this scheme in order to relax this requirement, $2m$ considered mutually prime numbers not prime numbers.

Adopting Sym-FHE schemes for cloud applications that consist of single user and the server are explicit the data can be encrypted using the symmetric key and outsourced to the server, the server can then execute operations on ciphertexts

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and return the encrypted results. The user then decrypts the results using the same symmetric Key. However, the majority of the cloud applications consist of multiple users mutually utilizing a service. Encrypting data using their symmetric keys would not lead to ciphertexts that can be manipulated upon by them collectively.

4. MULTIPARTY SYSTEM

The Sym-FHE scheme cannot be securely used in practical systems. To permit computation operations on encrypted data, the data in the cloud should be encrypted using the symmetric key. The symmetric key then has to be shared by multiple cloud users who need to manipulate this data in the cloud. A user may need to encrypt her sensitive data and send them with a request to the cloud service provider.Also, the cloud service provider may execute operations using Evaluate method without any key and send a response, which holds some encrypted data, to a user and the user then decrypts the ciphertexts using the same symmetric Key to obtain the result. Having all users holding the same symmetric key can affect the security of the system, particularly if the cloud users are from various and different domains.

The proposed solution is divided the symmetric key into master key k and exclusive user keysk_i. Each user owns an exclusive key and uses it to encrypt her data. Conversion function can be used to convert the ciphertext encrypted by the user key k_i to the ciphertext encrypted by the master key k. In this Sym-FHE scheme, data are encrypted using matrix representation and a similarity conversion function can be used to achieve the goal. The equivalent communication protocol can be improved for sending secret data between the cloud user and the cloud service provider using individual user keys and then use similarity conversion function to transform the exclusive user keys to the master encryption key.

4.1 System Model

The simplified system model considered where the cloud involved a single cloud server provider and a set of users accessing the data in the cloud. For security guarantee, a key distributor is added in between the cloud server provider and the users.Let CSP indicates the cloud server provider; KD means the key distributor, and set of users $U = \{U_i \mid i \geq 1\}$. User U_i holds an exclusive key k_i , KD holds the first corresponding key of k_i denoted as k_i'. CSP holds the second corresponding key of k_i , denoted ask_i'', where the master key $k = k_i \cdot k_i' \cdot k_i''$. The keys are produced and distributed by a trusted party TP at the system initialization period.

Each entity CSP, KD, or a user may be malicious or may be exposed by an external attacker and will perform as a passive attacker to reveal some sensitive information. One or more users may conspire with CSP or KD to obtain sensitive information. When the CSP and KD protected better, the possibility that both of them are malicious is slight. The adversary construction can specified as in [10] and suppose that an adversary can access all the information its role allows.

$AC = \{\{CSP\}, \{KD\}, U_A, \{CSP, KD\}, \{CSP\} \cup U_A, \{KD\} \cup U_A | U_A \in U\}$

The data hosted by the CSP may have various criticality grades and may be protected in different manners.

4.2The Multi-party Protocol (MP-Protocol)

First, the similarity conversion function which plays the pivotal role in the construction of the MPA protocol introduced. Let φbe the similarity conversion function

$$\varphi(U, k') = k'^{-1} \cdot U \cdot k'$$

Where $U = E(x, k)$is the ciphertext, and$k, k'$are encryption keys. Then φ can convert the encryptionkey from k to $k \cdot k'$based on the following lemma.

Lemma:

If$C = E(x, k)$, then$\varphi(U, k') = E(x \cdot k \cdot k')$

The procedure of the system involves two stages: the initialization stage and the request-response stage. At the system initialization period, a trusted party TP produces and distributes the keys to the key distributor, cloud server provider, and users, then TP exits the system and removes all the key related information. In the request-response stage, the user U_i sends a request to CSP, and CSP processes it and sends the result toU_i.

4.2.1 The initialization stage

In the initialization stage trusted party TP produces the master key $k$using KeyGen algorithm of Sym-FHE schemediscussed in Section 2.1.After that TP produces various key triplesk_i, k'_i, k''_i. The key k_i and the first matching $k_i'$are generated randomly as in keyGen. Then, TP calculates the

Journal of Theoretical and Applied Information Technology
30th June 2016. Vol.88. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

second matching key$k_i^{''} = k_i^{'-1} \cdot k_i^{-1} \cdot k$.TP sends$k_i$to user$U_i$, $k^{'}{}_i$to KD, and $k^{''}{}_i$to CSP as shown in figure 1. In a static system, TP terminates the systemafter key generation and distribution process. In a dynamic system where new users may join the systemdynamically, a key manager KM is also introduced tomanage user keys. TP sends the list of unemployed keys to KM to be distributed to new users later. TP terminates after key generation and distribution.

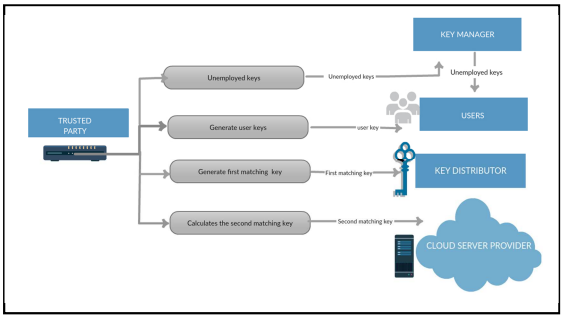

Figure 2: The Initialization Stage Of Multiparty Access Protocol

4.2.2 Request-response stage

The essential issue in the request-response phase is how to encrypt the data to be sent with the request and how to decrypt it in the response. The step of request-response protocols given in figure 2.

First step User $U_i$makes a request q with a critical data which was encrypted using user's keyk_i, resulting $Enc(data, k_i)$moreover, passes q with $Enc(data, k_i)$Key distributor KD. Then KD implements the conversion function $\varphi(Enc(data, k_i), k^{'}{}_i) = Enc(data, k_i \cdot k^{'}{}_i)$ and sends the updated request q' to cloud server provider CSP. The CSP implements the conversion function$\varphi(Enc(data, k_i \cdot k^{'}{}_i), k^{''}{}_i) = Enc(data, k_i \cdot k^{'}{}_i \cdot k^{''}{}_i) = Enc(data, k)$. After that CSP manipulates qwith$Enc(data, k)$and produces a response with an encrypted data $Enc(data^{'}, k)$. Then CSP calculates$\varphi\left(Enc(data^{'}, k), k_i^{''-1}\right) = Enc(data^{'}, k_i \cdot k^{'}{}_i)$and sends$Enc(data^{'}, k_i \cdot k^{'}{}_i)$ with response r to KD. The KD further calculates $\varphi((data^{'}, k_i \cdot k^{'}{}_i), k_i^{'-1}) = Enc(data^{'}, k_i)$and sends $Enc(data^{'}, k_i)$with $r$to the user U_i. Finally the user receives the response and decrypts $Enc(data^{'}, k_i)$ with $k_i$and gets result.

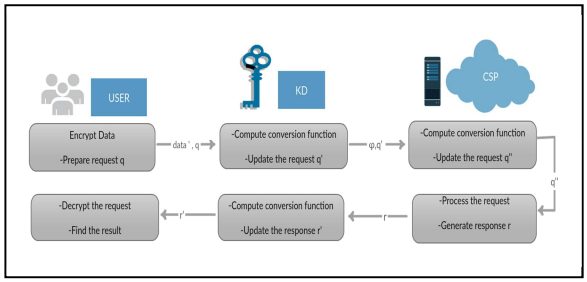

Figure 3: The request-response protocol

4.3 The Multi-party Protocol with Access Control (MPAC-Protocol)

A much richer context is where user authorizes his partner with different access control permissions. In several of real world scenarios, a user would allow partners to get results of only certain operations. Also, a user would like to have control of insertion, deletion, grant or revoke privileges for the partner users. Informally, Access control is authorizing subjects such as users, applications or processes with particular levels of access to different resources such as files or devices and enforcing such authorization when the resources are located. A general model of all the traditional ones is Attribute-Based Access Control (ABAC)[32]. ABAC is an excellent option for cloud computing [33].

We first give an introduction to Attribute Based Access Control model and explain how access control can be obtained on encrypted data using Sym-FHE schemes and then use those methods in the Multi-party protocol.

4.3.1 Attribute-based access Control (ABAC)

In ABAC model, privileges are defined based any characteristics known as attributes. The attributes for subjects and resources can be identifiers, names, title, etc. ABAC also has a particular type of environmental attributes such as current date and security levels. A policy rule defined as a Boolean function of attributes to make a decision if a subject S has permission to access a resource R in an environment E as below.

CanAccess= *f*(ATTR(S),ATTR(R),ATTR(E))

An Administrator identifies such attributes and determines the policy rules. The policy rules are kept in a Policy Rule Base (PRB). The responsibility for getting requests for authorization

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

decisions and implementing them rests with a Policy Enforcement Point PEP. Policy Decision Point (PDP)added and used to examine the validity of authorization request by handling Policy Rules [34].

In used cloud setup, the subjects are users. Encrypted data are resources. The data owner takes the role of an administrator, defines the attributes for partner users and resources. The user also sets the policy rules he wants to grant as permissions for his partners. Data owner outsources such PRB to the server. The server acts as both Policy Enforcement and Decision Point. Also, insert, delete and revoke of partner users can be obtained using the PRB. For example, the absence of privileges for a particular user in PRB can be considered as removed or revoked user.

Server Oblivious Attribute Based Access Control (SOABAC) can be achieved using FHE schemes composed of the symmetric Key method introduced in [34]. Attributes can be defined for users identified by their respective Keys. The resource identifiers can be encrypted using server Key of Sym-FHE scheme.

The ABAC protocol is basically an extension of previous MP-protocol. *The Server* knows that *the user* $U_1$is also the administrator and has the authorizations to control his PRB. Initial data preparation would be same as previous protocol. In addition, *user* U_1states PRB. *The server* in turn grants access to users as per encrypted PRB. U_1 as data owner Administrator configures PRB and subsequently U_2as partner user requests operations. When an user requests certain operations, he/she needs to send certain attributes. The attributes of a user U_2used for identification isk_2, there can be other attributes too. The attributes for resource and environment depends on the operation being requested lets assume they are *R,E*. All the attributes and PRB are encrypted using *Master* key k.

To use ABAC, all Users encrypt their Policy Rule Base using symmetric key k_i as Enc(PRB)= PRB', Users send PRB' to the server. When any user partner want to access any resources in the cloud they must send a function, encrypted data and some attribute then the server verify the access of the user using encrypted PRS after decrypted it homomorphicly and return the response as the previous MP-protocol

$ATTR'\,(a_1 \ldots a_n) \;=\; Enc_k(a_1 \ldots . a_n)\; a_i \in S, R, E$

U_i	$:=$	$\mathrm{Enc}(\,\mathrm{PRB}) = \mathrm{PRB}'$
$U_i \rightarrow CSP$	$:=$	$(f, \mathrm{data}', \mathrm{ATTR}'(a_i \ldots a_i)$
CSP	$:=$	$verifyAccess\ (f, \mathrm{data}', \mathrm{ATTR}'(a_i \ldots a_i)$
CSP	$:=$	$result' = eval(f, \mathrm{data}', k)$
$CSP \rightarrow U_i$	$:=$	$result'$
U_i	$:=$	$result = dec(result')$

Figure 4: Attribute-based access Control (ABAC)

4.4 Security of ABAC protocol

If the third party is an unauthorized partner but attempts to access resources in the cloud, such requests are first rejected while verifying access by CSP, assumed that CSP knows all authorized users through an initial configuration stage. If the third party is an eligible partner but attempts to access forbidden resources, such requests are thwarted while verifying the access control privileges. If the third party tries to overwrite the PRB itself, such applications are prevented too since such administrative operations are allowed by CSP only from the users.A risk exists though, If the third party discloses the platform of the cloud itself and alters the policy roles in PRB or avoid the access check, this can lead to a complete denial of service [34].

5. PERFORMANCE

To evaluate the performance of the proposed request and response protocols for multi-party systems on Sym-FHE scheme, Java programming language is used to implement and evaluate the execution time. The computations were performed on a 2.40 GHz Intel Core(TM) i3-3110 processor. Table 1 displays the execution time for a key generation, encryption and decryption algorithms for different lengths of N.

Table 2 shows the implementation of the algorithms and evaluate its execution time for sending and receiving a request with protocol and without using protocol. The large integer multiplication and addition were implemented using the GNU Multiple Precision (GMP) Arithmetic Library [35].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Execution Time of Key Generation, Encryption and Decryption Algorithms on Sym-FHE

Length of N	Key Generation	Encryption	Decryption
10	29 ms	25 ms	12 ms
12	37 ms	35 ms	17 ms
18	375 ms	213 ms	134 ms

Table 2: The Performance of the Multiparty Protocol

Operation	Performance
Encryption	198 ms
Decryption	32 ms
Transform	198 ms
Sending request (without protocol)	91 ms
Sending encrypted request (protocol)	936 ms
Receiving respond	87 ms
Receiving respond (protocol)	922 ms

As can be seen, using the communication protocol for sending a request and receiving a response with symmetric fully homomorphic encryption cryptosystem degraded the performance by approximately ten times from the case where the data in the request is not encrypted and is sent from user to the cloud server provider directly.

Sym-FHE is an efficient and practically feasible fully homomorphic scheme. Table 3 shows the comparison between DGHV [25] and Sym-FHE and evaluate the key, plaintext and ciphertext size using the complexity of an algorithm. Also evaluate the computation overhead. Symmetric FHE schemes are more efficient than asymmetric while asymmetric schemes are more secure.

Table 3: Comparison between DGHV and Sym-FHE

	DGHV FHE Scheme	Sym-FHE Scheme
Based on	Integer	Matrix
Key type	Asymmetric	Symmetric
Noise	Noise-based	Free-noise
Key size	(λ^{10})	$O(m\lambda)$
Plaintext size	$O(\log \lambda)$	$O(1)$, Exactly $4 \times 4 = 16$
Ciphertext size	$O(\lambda^{5})$	$O(m\lambda)$
Computation Overhead	$O(\lambda^{3.5})$	$O(m\lambda)$

6. CONCLUSIONS

Scope and promises of homomorphic cryptography in cloud computing environments cannot be ignored. Researchers around the world are taking great interesting in recent years to construct fully homomorphic cryptosystems that can be implemented practically. Much of the focus is on imparting skills to homomorphic public-key cryptosystems, while some applications can also be managed with symmetric key cryptosystem. Therefore, our efforts have been to propose ideas on how symmetric keys and simple matrices based operations could also lead to simple, feasible and efficient fully homomorphic scheme for computing on the cloud, specifically for the delegation of computing and data processing private data in clouds.

This paper introduced fully homomorphic encryption scheme with a Symmetric key technique based on [10] with modification the constraint of prime numbers. The security of the proposed Sym-FHE scheme is equivalent to the well-known large integer factorization problem LIFP (which is also the basic security problem for RSA). The proposed fully homomorphic encryption algorithm is symmetric-key based while most of the existing algorithms are asymmetric key based. The only advantage of the public key homomorphic encryption schemes is the
the possibility of encrypting data without needing to know the private key.

Also, this paper presented protocols for multi-party scenarios. Showed that Sym-FHE schemes for the single user could be scaled to a multi-user scenario with access control. Communication costs implicated in cloud computing are regularly large, to make up for this we emphasize on having low time complexity for encryption and decryption primitives. The multiparty protocol with Sym-FHE scheme degraded the performance by approximately ten times, but it is important to protect the sensitive data in the cloud and sent a secure request.

REFERENCES:

[1] 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.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[2]P. Mell, T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, U. S. Department of Commerce, September 2011.

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

[5] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "Fully homomorphic encryption without bootstrapping," Cryptology ePrint Archive, Report 2011/277, 2011, http: //eprint.iacr.org/2011/277.pdf.

[6] Z. Brakerski and V. Vaikuntanathan, " Fully homomorphic encryption from ring-LWE and security for key-dependent messages," in CRYPTO, vol. 6841, 2011, p. 501

[7] V. Vaikuntanathan, "Computing blindfolded: New developments in fully homomorphic encryption," in *Foundations of Computer Science(FOCS), 2011 IEEE 52nd Annual Symposium on*. IEEE, 2011, pp.5–16.

[8]C. Gentry and S. Halevi, "Fully homomorphic encryption without squashing using depth-3 arithmetic circuits," Cryptology ePrint Archive, Report 2011/279, 2011, to appear in this Proceedings.

[9] C. Gentry, S. Halevi, and N. Smart, "Implementing gentry's fully-homomorphic encryption scheme," in EUROCRYPT, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed., vol. 6632. Springer, 2011, pp. 129–148.

[10] L. Xiao, O. Bastani, and I.-L. Yen, "An efficient homomorphic encryption protocol for multi-user systems." Cryptology ePrint Archive, Report 2012/193

[11] Kipnis and E Hibshoosh. "Efficient Methods for Practical Fully-HomomorphicSymmetric key Encryption, Randomization, and Verification." Available athttp://eprint.iacr.org/2012/637.

[12] T.ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, 1985, pp. 469 – 472.

[13] S. Goldwasser, S. Micali, "Probabilistic encryption and how to play mental poker keeping secret all partial information," in Proceedings of 14th Symposium on Theory of Computing, 1982, pp. 365 – 377

[14]P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in EUROCRYPT, 1999, pp. 223–238.

[15]Benaloh, J. Dense probabilistic encryption. In In Proceedings of the Workshop on Selected Areas of Cryptography (1994), pp. 120–128.

[16] Naccache, D., and Stern, J. A new public-key cryptosystem. In EUROCRYPT (1997), pp. 27–36.

[17] Okamoto, T., and Uchiyama, S. A new public-key cryptosystem as secure as factoring. In Eurocrypt '98, LNCS 1403 (1998), Springer-Verlag, pp. 308–318.
Volume 1592 of Lecture Notes in Computer Science., Springer (1999) 223-238.

[18] Damgård, I., and Jurik, M. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography : Public Key Cryptography (London, UK, UK, 2001), PKC '01, Springer-Verlag, pp. 119–136.

[19] Schmidt-Samoa, K., and Takagi, T. Paillier's cryptosystem modulo p2q and its applications to trapdoor commitment schemes. In Proceedings of the 1st International Conference on Progress in Cryptology in Malaysia (Berlin, Heidelberg, 2005), Mycrypt'05, Springer-Verlag, pp. 296–313.

[20] Smart, N.P., Vercauteren, F. Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. Cryptology ePrint Archive, Report 2009/571, 2009. http://eprint.iacr.org/.

[21] G. Chunsheng. " More practical Fully Homomorphic Encryption" Available at http://eprint.iacr.org/2011/121.

[22]D. Stehle and R. Steinfeld. "Faster fully homomorphic encryption." Cryptology ePrint Archive Report 2010/299.

[23] N. P. Smart and F. Vercauteren, "Fully homomorphic SIMD operations", IACR Cryptology ePrint Archive, Report 2011/133.

[24] C. Gentry and S. Halevi, "Fully homomorphic encryption without squashing using depth-3 arithmetic circuits", Cryptology ePrint Archive, Report 2011/279.

[25] van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V. Fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2009/616, 2009. http://eprint.iacr.org/.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[26]Z. Brakerski, V. Vaikantanathan. " Efficient Fully Homomorphic Encryption from (Standard) LWE". Available at http://eprint.iacr.org/2011/344.
[27] Z. Brakerski, C. Gentry, V. Vaikantanathan. "Fully Homomorphic Encryption without Bootstrapping." Available at http://eprint.iacr.org/2011/277.
[28] Z. Chen, J. Wang, ZN. Zhang, X. Song, "A Fully Homomorphic Encryption Scheme with Better Key Size". Available at ttps://eprint.iacr.org/2014/697.pdf
[29] Z. Brakerski, "Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP". Available at http://eprint.iacr.org/2012/78.
[30] J. Li and L. Wang, "Noise-free Symmetric Fully Homomorphic Encryption based on noncommutative rings", Cryptology ePrint Archive, Report 2015/641.
[31] K. Gjosteen, M. Strand. Fully homomorphic encryption must be fat or ugly?. Cryptology ePrint Archive, Report 2016/105, 2016. http://eprint.iacr.org/.
[32] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering AC, mac, and rbac," in Data and ApplicationsSecurity and Privacy XXVI. springer, 2012, pp. 41–55.
[33] E. Yuan and J. Tong, "Attributed based access control (abac) for web services," in Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEEInternational Conference on. IEEE.
[34] S. Dara:Multi-user protocols with access control for computational privacy in public clouds. CoRR abs/1406.1823 (2014)
[35] GMU MP library, http://gmplib.org/.