Journal of Theoretical and Applied Information Technology
15th April 2018. Vol.96. No 7

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

PUBLIC KEY FULLY HOMOMORPHIC ENCRYPTION

[1]SARAH SHIHAB HAMAD, [2]ALI MAKKI SAGHEER
[1, 2] Department of Computer Science, College of Computer Science and Information Technology, University of Anbar, Anbar, Iraq
E-mail: [1]sarah.sh1985@gmail.com, [2]ali_makki@uoanbar.edu.iq

ABSTRACT

The meaning of cloud computing is the Information Technology (IT) model for computing, which is consist of all the IT components (software, hardware, services and, networking) that are needed to enable delivery and development of cloud services through the private network or the internet. In the cloud computing the client put his data on the cloud, and any computations on his stored data will be implemented in the cloud. Intentionally the cloud service provider can access, change or even delete the stored data of the client. To produce the effective services, some of the cloud service providers share the information with third parties. The third party can access to the client private data and modifies on this data to make it beneficial to him. Moreover, providers, untrusted servers, and cloud operators can keep identifying elements of users long after users end the relationship with the services. For these reasons, the security has become the most important thing in cloud computing. The mechanism that protects the privacy of sensitive information is called the encryption. To protect the data that it's stored on the cloud we must use an encryption system that can perform computations on the encrypted data. The scheme that allows to executing several computations on the encrypted message without decrypting this message is called homomorphic encryption. This paper provides encryption scheme (PKFHE) that can applied it on a cloud computing .The proposed scheme is based on Euler's theorem, which proved both addition and multiplication operations at the same time on ciphertext without decryption. we compute the time complexity of the encryption and decryption function for the Elgamal, RSA cryptosystems and our proposed PKFHE scheme which, the order of encryption function of all the schemes is O $((\log (n))^3)$ and the order of decryption function of Elgamal, RSA cryptosystems is O $(\log (n)^3)$, while the order of decryption function of PKFHE scheme is O $(\log (n)^2)$ that mean the proposed PKFHE scheme is faster in the time complexity. Also compared the execution time among the three schemes with five sizes of the messages and used five lengths of a secret key, where we concluded that the execution time of the proposed scheme was slower than the execution time of Elgamal, RSA cryptosystems except in the case of the key 128-bit which, the execution time of RSA was slower than the two others. Also we noticed that the length of the private key was effect on the execution time of the algorithms in which the execution time increased when the length of the private key was increase and finally the security of the schemes is analyzed. The proposed scheme showed a good security for the stored data on the cloud.

Keywords: *Fully Homomorphic Encryption; Public Key Cryptosystem; Homomorphism; Cloud Computing Security.*

1. INTRODUCTION

Cloud computing is a technology that uses central remote servers and the internet to share the applications and maintain the data. It allows client to use an application without installation and he can access to their personal files from any computer just verify internet connection [1]. To protect the data that it's stored in the cloud we must use an encryption system that can perform computations on the encrypted data. The technique that allows to computation on a ciphertext without previous decryption is called homomorphic encryption HE [2]. HE scheme can be either, symmetric or asymmetric. An encryption scheme contains three algorithms: KeyGen, Encrypt, and Decrypt. In a symmetric (or secret key) encryption scheme, KeyGen utilize λ (is a security parameter that determines the bit length of the keys) to produce a one key that is used in each of encryption and decryption, first transform a message to a ciphertext, and later transform the ciphertext back to the message. In an asymmetric (or public key) encryption scheme, KeyGen utilizes λ to generate two keys: a public encryption key pk, which may be made available to everyone, and a secret

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

decryption key sk. As a physical analogy for an asymmetric encryption scheme, one can think of Alice public key as a lock, which builds and distributes, which can be locked without a key. Anyone can place a message inside a box that secured by Alice's lock (encrypt) and sending it across the general channel to Alice, but Alice just has the key to open it (decrypt) [3]. An encryption technique is called Fully Homomorphic (FHE) if it performs both addition and multiplication at the same time, and can compute any Operation [4]. The organization of a remainder of the paper: section 2, provides Related Works about HE, Section 3, describes categories of Homomorphic Encryption, Section 4, display Fermat and Euler Theorems. Section 5, offer the proposed scheme, also we illustrate an example about it. Section 6, describes the results and discussion. Finally, our conclusions are mentioned in Section 7.

2. RELATED WORK

The first homomorphism suggested by **Rivest, Adleman, and Dertouzos** in [5]. The multiplicative homomorphism is given by **RSA** [6]. A partial homomorphic encryption scheme is suggested by **Yao** [7], **Goldwasser and Micali** [8], **ElGamal** [9] and **Paillier** [10]. **Fontaine & Galand** in [11] has presented a survey of homomorphic encryption schemes .**Gentry** has proposed fully homomorphic encryption in his thesis and paper [4].Many researchers proposed the variants of Gentry's model with some improvement. Homomorphic encryption on smaller size cipher text is proposed by **Smart and Vercauteren** [12]. The arithmetic operations over integers are proposed by **Dijk, Gentry, Halevi, and Vaikuntanathan** (DGHV scheme) [13]. Faster improvement to Gentry's model is proposed by **Stehle and Steinfield** [14]. **Y Govinda Ramaiah** has proposed "Efficient Public Key Homomorphic Encryption over Integer Plaintexts". [15] **Coron et al** [16] was extended the work of DGHV [13] over the integers for the purpose of batching FHE to a scheme supporting encryption and homomorphic processing of a vector of plaintext bits as a single ciphertext. **Emura et al** [17] aimed at controlling that can perform the homomorphic operations, when and where. Because of the malleability property of homomorphic encryption that anyone can perform the operations"freely", it becomes difficult to achieve adaptive chosen ciphertext (CCA) and homomorphic. **Kim et al** [18] presented a fully homomorphic scheme that generalizes the DGHV [13] scheme and modifies the third proposal of RSA [19] on the Chinese Remainder theorem and ring homomorphism.

3. HOMOMORPHIC ENCRYPTION CATEGORIES

There are three main categories of homomorphic encryption schemes: Partially Homomorphic Encryption PHE, Somewhat Homomorphic Encryption SWHE, and Fully Homomorphic Encryption FHE schemes. PHE schemes, such as RSA [6], ElGamal [9], Paillier [10], Etc., allow to applying either addition or multiplication on encrypted data. Boneh et al. [20], allowing unlimited additions and a single multiplication. Construction of scheme supporting both operations addition and multiplication simultaneously is possible in 2009 by Gentry [4] by using fully homomorphic encryption.

3.1 Properties of Homomorphic Encryption

3.1.1 Additive Homomorphic Encryption:

A homomorphic encryption is additive if:
$Enc(P_1 \oplus P_2) = Enc(P_1) \oplus Enc(P_2)$.

3.1.2 Multiplicative Homomorphic Encryption:

A homomorphic encryption is multiplicative, if:
$Enc(P_1 \otimes P_2) = Enc(P_1) \otimes Enc(P_2)$ [21].

3.2 Partially Homomorphic Encryption (PHE)

An encryption technique is called a Partially Homomorphic Encryption (PHE) if it applying only one operation on encrypted data, i.e., either addition or multiplication but not both [22]. Any encryption scheme which supports either addition or multiplication, but necessarily not both, is termed to be a partial homomorphic scheme. Mathematically, an encryption scheme supporting group operations on the ciphertexts is a partially homomorphic encryption system.

3.2.1 Additive Homomorphic Schemes

A Homomorphic Encryption is additive if there is an algorithm that can calculate Enc(x + y) from Enc(x) and Enc(y) without knowing x and y [23].Such as Goldwasser-Micali algorithms [8] and Paillier [10].

3.2.2 Multiplicative Homomorphic Schemes

A Homomorphic Encryption is called multiplicative if there is an algorithm that can calculate Enc(x × y) from Enc (x) and Enc (y)

Journal of Theoretical and Applied Information Technology
15th April 2018. Vol.96. No 7

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

without knowing x and y [23]. Such as RSA [6] and ElGamal [9] Algorithms.

3.3 Somewhat Homomorphic Encryption (SWHE)

The scheme that supports a limited number of homomorphic operations known as somewhat homomorphic encryption [24]. An encryption technique is called Somewhat Homomorphic encryption (SWHE) if it performs a limited number of addition and multiplication operations on encrypted data [4].

To understand SWHE scheme's work, first thing that we have to know: all scheme's ciphertexts have a noise inside of them and this noise unfortunately become larger if we have performed a lot of homomorphic operations, there is so much noise (at some point) which the encryptions become useless (i.e. they have incorrect decrypt). This is the basic limitation of SWHE schemes and this is the reason that they can only perform a specific set of computations. [25]

3.4 Fully Homomorphic Encryption (FHE)

An encryption technique is called Fully Homomorphic (FHE) if it performs both addition and multiplication at the same time, and can compute any operation as in Figure 1 [4].

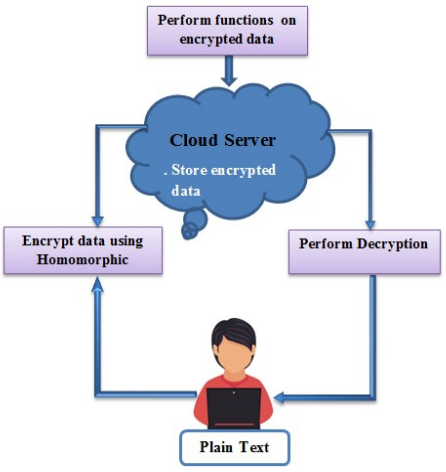

Figure 1: Fully Homomorphic Encryption (FHE) [26].

Craig Gentry in 2009, presented the first construction of a fully homomorphic scheme [4]. Since then researchers have proposed variants and improvements to Gentry's model.

4. FERMAT AND EULER THEOREMS

Two important theorems presented the first by Pierre de Fermat and the second by Leonhard Euler. Both theorems related to powers in modular arithmetic.

4.1 Fermat's Little Theorem

Suppose that p is prime and gcd $(a, p) = 1$ (or a and p are relative prime or p does not divide a).

Then:

$$a^{p-1} \equiv 1 \pmod{p} \quad (1)$$

4.2 Euler's Theorem

Euler's Theorem is a generalize of Fermat's Little Theorem. Suppose n be an arbitrary positive integer, $ø(n)$ denote the number of integers $1 =< a <= n$ such that if $gcd(a, n) = 1$,

then:

$$a^{ø(n)} \equiv 1 \pmod{n} \quad (2)$$

Note: When $(n = p)$ is prime, then $ø\ (p) = p\text{-}1$, we get Fermat's Little Theorem.

The following facts are true about the Euler Ø-function.

- $ø(p) = p-1$ if p is prime.
- $ø\ (p.q) = (p-1)(q - 1)$ if p and q are prime.
- The general formula of $ø\ (n)$ is:

$$Ø\ (n) = n \prod_{p/n} \left(1 - \frac{1}{p}\right) \quad (3)$$

Where, the product is over distinct primes dividing n [27].

5. THE PROPOSED SCHEME:PUBLIC KEY FULLY HOMOMORPHIC ENCRYPTION

Our proposed scheme (PKFHE) is essentially based on Euler's theorem that we explained it in Section three, which it's an asymmetric (or public key) encryption scheme and supports both additive and multiplicative homomorphism property, that means its Fully Homomorphic Encryption scheme. The description of PKFHE scheme as follows:

Key Gen: select two prime numbers p and q

Calculate: $n = p.\ q$

$m = p + q$

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

Then choose another prime number u such that:

$\gcd(n, u) = 1$,

$S = n * u$

And Select random big integer t.

$e = t\,(n - m + 1)$

Where, e is big integer.

Put (e, S) as a public key and keep (n) as a secret key

Encryption: the message M will always be less than $< n$, select random big integer r.

$c = M^{r * e + 1} \bmod S$

Where, c is a ciphertext.

Evaluate: apply addition and multiplication on ciphertexts ci, and then decrypt the result of ci, we get integer number which is the same as the integer number that it result of applying addition and multiplication on input Mi.

Decryption:

$M = c \bmod n$

To proof correctness of the scheme:

$c = M^{r * e + 1} \bmod S$

$Dec = c \bmod n$

$= M^{r * e + 1} \bmod S \bmod n$

$= M^{r * e + 1} \bmod n \bmod S$

We know that:

$a^{r * e + 1} \equiv a \pmod n$, (Euler's theorem)

$= M \bmod S = M,\ M < S$ *(proved).*

Additive Homomorphism:

The two messages M_1 and M_2 will always be less than $< n$, also addition $(M_1 + M_2)$ should be less than n,

$M_1 + M_2 = Dec\,[Enc\,(M_1) + Enc\,(M_2)]$,

(***Enc*** is Encryption function and ***Dec*** is Decryption function).

Proof:

Select two random big integer r_1 *and* r_2

$c_1 = (M_1^{r1 * e + 1} \bmod x)$,

$c_2 = (M_2^{r2 * e + 1} \bmod x)$

$c_1 + c_2 = M_1^{r1 * e + 1} \bmod S + (M_2^{r2\,(n - m + 1) + 1} \bmod S)$

$Dec\,(c_1 + c_2) = (c_1 + c_2) \bmod n$

$= [(M_1^{r1 * e + 1} \bmod S) + (M_2^{r2 * e + 1} \bmod S)] \bmod n$

$= (M_1^{r1 * e + 1} \bmod S \bmod n) + (M_2^{r2 * e + 1} \bmod S \bmod n)$

$= (M_1^{r1 * e + 1} \bmod n \bmod S) + (M_2^{r2 * e + 1} \bmod n \bmod S)$

Now we know that $a^{r * e + 1} \equiv a \pmod n$ so

$= M_1 \bmod S + M_2 \bmod S = M_1 + M_2$.

Multiplicative homomorphism

The two messages M_1 and M_2 will always be less than $< n$, also multiplication $(M_1 \times M_2)$ should be less than n,

$M_1 \times M_2 = Dec\,[Enc\,(M_1) \times Enc\,(M_2)]$.

Proof:

$c_1 = M_1^{r1 * e + 1} \bmod S$,

$c_2 = M_2^{r2 * e + 1} \bmod S$

$c_1 \times c_2 = (M_1^{r1 * e + 1} \bmod S) \times (M_2^{r2 * e + 1} \bmod S)$

$Dec\,(c_1 \times c_2) = (c_1 \times c_2) \bmod n$

$= [(M_1^{r1 * e + 1} \bmod S) \times (M_2^{r2 * e + 1} \bmod S)] \bmod n$

$= (M_1^{r1 * e + 1} \bmod S \bmod n) \times (M_2^{r2 * e + 1} \bmod S \bmod n)$

$= (M_1^{r1 * e + 1} \bmod n \bmod S) \times (M_2^{r2 * e + 1} \bmod n \bmod S)$

Now we know that $a^{r * e + 1} \equiv a \pmod n$

$= [(M_1 \bmod S) \times (M_2 \bmod S)] = M_1 \times M_2$.

Algorithm (PKFHE)

1- Key Generation

1-1 generate prime big integers p and q

1-2 n = p. q

1-3 m = p +q

1-4 choose another prime no. u , gcd(u, n) = 1

1-5 S = n * u

1-6 select random big integer t

1-7 e = t (n – m +1)

- Public encryption key = (e, S)
- Private decryption key = (n)

2- Encryption

2-1 generate random big integer r

2-2 $C = M^{r * e + 1} \bmod S$

3- Decryption

3-1 M = C mod n

Journal of Theoretical and Applied Information Technology
15th April 2018. Vol.96. No 7

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

4- Evaluation

4-1 Suppose there are two ciphertexts:

$C_1 = M_1^{r1 * e + 1} \bmod S$

$C_2 = M_2^{r2 * e + 1} \bmod S$

4-2 $C_3 = C_1 + C_2 \Rightarrow M_3 = \text{Dec}(C_3)$ (or $M_3 = M_1 + M_2$)

4-3 $C_4 = C_1 . C_2 \Rightarrow M_4 = \text{Dec}(C_4)$ (or $M_4 = M_1 . M_2$)

Let $C = f(c_1 ... c_i)$, such as

$C = [(C_1 . C_3) + C_2] . C_4$

Let $M = f(M_1 ... M_i)$, such as

$M = [(M_1 . M_3) + M_2] . M_4$,

Where f is any function (Add or Mult) applied on the ciphertexts or messages.

4-4 C mod n ≡ M, where M <n, otherwise that we must take (M mod n)

Example (PKFHE):-

1- Key Generation

Choose two prime numbers $p = 11$ and $q = 7$, then

$n = p * q \rightarrow n = 11 * 7 \rightarrow n = 77$

$m = p + q \rightarrow m = 18$

Select another prime number which, gcd $(n, u) = 1$

Let $u = 23$, and $gcd(77, 23) = 1$,

Now calculate $S = n * u \rightarrow S = 77 * 23 \rightarrow S = 1771$

And Select random big integer $t = 9$.

$e = t(n - m + 1)$

$e = 9(77 - 18 + 1)$

$e = 540$

Public key $(e, S) = (540, 1771)$

Secret key $(n) = (77)$

2- Encryption

Now choose two random integer $r_1 = 2$ and $r_2 = 3$, and two messages $m1 = 4$ and

$m2 = 5$,

We must ensure that $(m1 + m2)$ & $(m1 * m2) < n$

Now calculate c_1:

$c_1 = m_1^{r1 * e + 1} \bmod S$,

$= 4^{2 * 540 + 1} \bmod 1771 \rightarrow c_1 = 4^{1081} \bmod 1771$

$= 1467$

Then calculate c_2:

$c_2 = m_2^{r2 * e + 1} \bmod S$

$= 5^{3 * 540 + 1} \bmod 1771 \rightarrow c_2 = 5^{1621} \bmod 1771$

$= 1468$

3- Decryption

$M_1 = C_1 \bmod n$

$= 1467 \bmod 11$

$= 4$

$M_2 = C_2 \bmod n$

$= 1468 \bmod 11$

$= 5$

4- Evaluation

$c_3 = c_1 + c_2 \rightarrow c_3 = 1467 + 1468$

$c_3 = 2935$

Now decrypt of c_3:

$m_3 = c_3 \bmod n \rightarrow m3 = 2935 \bmod 77$

$m_3 = 9$, this is equal to $(m_1 + m_2)$ (i.e. $4+5=9$)

$c_4 = c_1 * c_2 \rightarrow c_4 = 1467 * 1468$

$c_4 = 2153556$

Now decrypt of c_4:

$m_4 = c_4 \bmod n \rightarrow m_4 = 2153556 \bmod 77$

$m_4 = 20$, this is equal to $(m_1 * m_2)$ (i.e. $4 * 5 = 20$).

Let $C = f(c_1 ... c_i)$, such as:

$C = [((C_1 . C_3) + C_2) . C_4] \bmod n$

$= [((1467 * 2935) + 1468) . 2153556] \bmod 77$

$= 9275609043828 \bmod 77$

$= 50$

Let $m = f(m_1 ... m_i)$, such as:

$m = [((m_1 . m_3) + m_2) . m_4] \bmod p$

$= [4 . 9 + 5] . 20 \bmod 77$

$= 820 \bmod 77$

$= 50 \equiv C$ *(proved).*

Journal of Theoretical and Applied Information Technology
15th April 2018. Vol.96. No 7

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

6. RESULTS AND DISCUSSION

In this section we'll calculate the bit complexity of the encryption and decryption function for the Elgamal, RSA Cryptosystems and our proposed PKFHE scheme and also we'll compute the execution time of them.

6.1 Big O Notation (Time Complexity)

The *O*-notation is very useful in guiding the designers of the algorithms in search for the "best" algorithms for important problem.

The goal of the study of the time complexity of an algorithm is to indicate that its running time is $O(f(N))$ for some function f[27]. Before performing calculations of the time complexity, we must first analyze the input numbers of the encryption and decryption algorithm, which are either binary integers or decimal digits, where:

The time complexity of a binary integers is $O(n)$, while the time complexity of a decimal digits is O(log(n)),except the constant number, which the time complexity to its $O(1)$, where n is the size of input numbers.

The complexity of the basic arithmetic operations in Z_n is shown in Table 1:

Table 1: Time complexity of the basic arithmetic operations.

Operation	Time complexity of binary integers of size n	Time complexity of decimal digits of size n
Addition x + y	O (n)	O (log (n))
Subtraction x – y	O (n)	O (log (n))
Multiplication x × y	O (n^2)	O($(\log (n))^2$)
Division &Modular	O (n^2)	O($(\log (n))^2$)
Inverse x^{-1}	O ($n^2 \log (n)$)	O($(\log (n))^3$)
Modular exponentiation x^n	O ($n^2 \log (n)$)	O($(\log (n))^3$)

6.1.1 Time Complexity of Elgamal Cryptosystem

Let n is the size of input message that in the type of decimal digits.

Encryption function:

$K = Y^r \pmod p$

$C_1 = g^r \pmod p$

$C_2 = M.\ K \pmod p$

Then

$T(K) = O((\log (n))^3)$, (Modular exponentiation)

$T(C_1) = O((\log (n))^3)$

$T(C_2) = O(2(\log (n))^2) = O((\log (n))^2)$

Where, ($\log_2 n$) is the number of bits of n.

T (Encryption) $= O((\log (n))^3)$ *bit operation*, according the most costly operation.

Also,

T (Encryption) $= O(\log (n))$ *arithmetic operation.*

Decryption function:

$K = C_1^x \pmod p$

$M = K^{-1}.\ C_2 \pmod p$

Then:

$T(K) = O((\log (n))^3)$

$T(M) = O(2(\log (n))^2) + T(K^{-1})$

$T(K^{-1}) = O((\log (n))^3)$, (by extend Euclid's method)

$T(M) = O(2(\log (n))^2) + O((\log (n))^3)$

$T(M) = O((\log (n))^3)$ *bit operation*

T (Decryption) $= O((\log (n))^3)$ *bit operation,* according the most costly operation.

Also,

T (Decryption) $= O(\log (n))$ *arithmetic operation.*

6.1.2 Time Complexity of RSA Cryptosystem

Let n is the size of input message that in the type of decimal digits.

Encryption function:

$C = M^e \bmod (n)$

Then:

$T(C) = O((\log (n))^3)$ *bit operation.*

T (Encryption) $= O((\log (n))^3)$ bit operation, according the most costly operation.

Also,

T (Encryption) $= O(\log (n))$ arithmetic operation.

Decryption function:

$M = C^d \bmod (n)$

Then:

$T(M) = O((\log (n))^3)$ *bit operation.*

T (Decryption) $= O((\log (n))^3)$ *bit operation*, according the most costly operation.

Also,

T (Decryption) $= O(\log (n))$ *arithmetic operation.*

Journal of Theoretical and Applied Information Technology
15th April 2018. Vol.96. No 7

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

6.1.3 Time Complexity of PKFHE scheme

Let n is the size of input message that in the type of decimal digits.

Encryption function:

$C = M^{r*e+1} \bmod S$

Then:

$T(C) = O((\log(n))^3)$.

Where, $(\log_2 n)$ is the number of bits of n.

$T(Encryption) = O((\log(n))^3)$ *bit operation*, according the most costly operation.

Also,

$T(Encryption) = O(\log(n))$ *arithmetic operation.*

Decryption function:

$M = C \bmod n$

Then:

$T(M) = O((\log(n))^2)$

$T(Decryption) = O((\log(n))^2)$ *bit operation*, according the most costly operation.

Also,

$T(Decryption) = O(1)$ *arithmetic operation.*

6.2 ExecutionTime

We compute the execution time of the Elgamal, RSA Cryptosystems and our proposed PKFHE scheme with five sizes of the message and five lengths of private key as following:

In Table 2: notes that we take five size of the messages begin whith (12 byte) and end with (2.5 K Byte) and we use a secret key with length of 32-bit that equivalent (10 digit) to calculate of the execution time of Elgamal, RSA Cryptosystems and our proposed PKFHE scheme as shown in Figure 2.

Table 2:Execution time with Size of Secret key 32- bit

Size of the message	ElGamal (m.s)	RSA (m.s)	PKFHE (m.s)
12 Byte	15	16	18
1 K Byte	9491	11377	11372
1.5 K Byte	21497	25149	26461
2 K Byte	37816	45866	46286
2.5 K Byte	58699	72900	75600

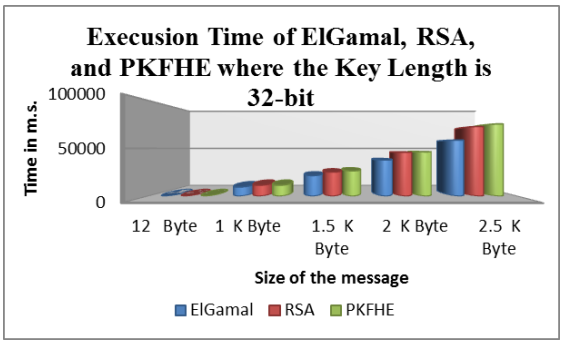

Figure 2: Execution time with length of Secret key 32-bit.

In Table 3: notes that we take also five size of the messages begin whith (12 byte) and end with (2.5 K Byte) and we use a secret key with length of 64-bit that equivalent (20 digit) to calculate of the execution time of Elgamal, RSA Cryptosystems and our proposed PKFHE scheme as shown in Figure 3.

Table 3:Execution time with length of Secret key 64- bit

Size of the message	ElGamal (m.s)	RSA (m.s)	PKFHE (m.s)
12 Byte	15	17	19
1 K Byte	15137	15625	19750
1.5 K Byte	33456	34589	40543
2 K Byte	58738	61477	78730
2.5 K Byte	100048	105859	126001

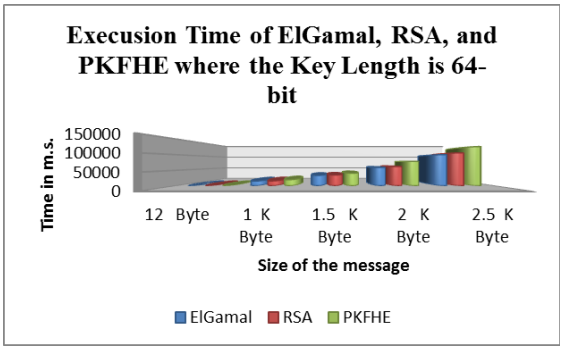

Figure 3: Execution time with length of Secret key 64- bit

In Table 4: notes that we take also five size of the messages begin whith (12 byte) and end with (2.5 K Byte) and we use a secret key with length of 128-bit that equivalent (39 digit) to calculate of the execution time of Elgamal, RSA Cryptosystems and our proposed PKFHE scheme as shown in Figure 4.

Journal of Theoretical and Applied Information Technology
15th April 2018. Vol.96. No 7

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

Table 4: Execution time with length of Secret key 128- bit

Size of the message	ElGamal (m.s)	RSA (m.s)	PKFHE (m.s)
12 Byte	21	22	27
1 K Byte	30751	36431	34783
1.5 K Byte	72440	90424	77504
2 K Byte	140054	163767	151828
2.5 K Byte	217615	263558	238704

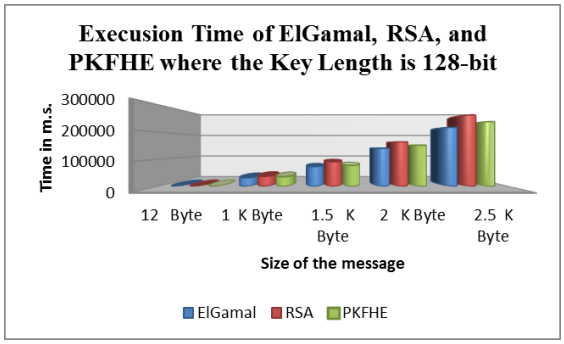

Figure 4:Execution time with length of Secret key 128-bit

In Table 5: notes that we take also five size of the messages begin whith (12 byte) and end with (2.5 K Byte) and we use a secret key with length of 256-bit that equivalent (78 digit) to calculate of the execution time of Elgamal, RSA Cryptosystems and our proposed PKFHE scheme as shown in Figure 5.

Table 5:Execution time with length of Secret key 256- bit

Size of the message	ElGamal (m.s)	RSA (m.s)	PKFHE (m.s)
12 Byte	22	24	46
1 K Byte	34541	39965	75258
1.5 K Byte	82953	96749	165610
2 K Byte	155187	172906	308932
2.5 K Byte	241098	276159	487468

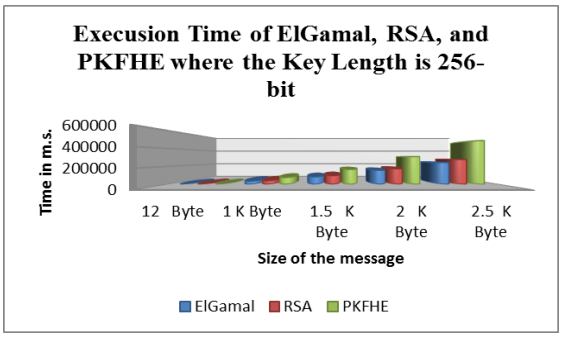

Figure 5:Execution time with length of Secret key 256-bit

In Table 6: notes that we take also five size of the messages begin whith (12 byte) and end with (2.5 K Byte) and we use a secret key with length of 512-bit that equivalent (155 digit) to calculate of the execution time of Elgamal, RSA Cryptosystems and our proposed PKFHE scheme as shown in Figure 6.

Table 6:Execution time with length of Secret key 512- bit

Size of the message	ElGamal (m.s)	RSA (m.s)	PKFHE (m.s)
12 Byte	35	23	58
1 K Byte	93711	37787	128837
1.5 K Byte	212789	93554	298497
2 K Byte	394844	178951	540880
2.5 K Byte	658823	287563	853183

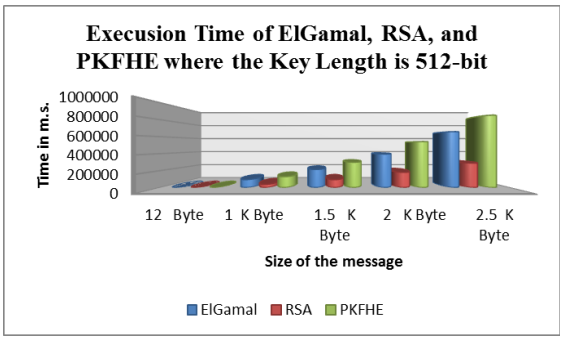

Figure 6:Execution time with length of Secret key 512-bit

As a result, noticed that the length of the key was effect on the execution time of the algorithms in which the execution time increased when the length of the key was increase.

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

6.3 Security

The RSA security depends on truth that it is easy to multiply two large primes to construct a modulus, the inverse operation of factoring the modulus into its prime factors can be very difficult, the difficult until solve the integer factorization problem for the sizes of the numbers involved. Attempt to break RSA by developing an integer factorization solution for the moduli involved is known as a mathematical attack. That is, a mathematical attack on RSA consists of discover the prime factors p and q of the modulus n. Clearly, that knowing p and q, the attacker will be able to discover the private exponent d for decryption. Another way of stating the same as above would be that the attacker would try to discover the totient $\varphi(n)$ of the modulus n. But as stated earlier, knowing $\varphi(n)$ is equivalent to knowing the factors p and q. If an attacker can somehow figure out $\varphi(n)$, the attacker will be able to set up the equation $(p-1)(q-1) = \varphi(n)$, that, along with the equation $p \times q = n$, will allow the attacker to determine the values for p and q. Over the years, various mathematical techniques have been developed for solving the integer factorization problem involving large numbers such as Trial Division, Fermat's Factorization Method, Sieve Based Methods, and Pollard-ρ Method etc.

The security of ElGamal is based on the discrete logarithm problem. To encrypt and decrypt a message, a discrete power is executed. This operation is efficient to compute. An attacker that wants to decrypt an intercepted message may try to recover the private key. To this end a logarithm needs to be computed. No actual method exists for this, given certain requirements on the initial group are met. Under these circumstances, the encryption is secure. ElGamal has the disadvantage that the ciphertext is twice as long as the plaintext. It has the advantage the same plaintext gives a different ciphertext each time it is encrypted. Today the ElGamal algorithm is used in many cryptographic products.

In brief, the security of the Elgamal and RSA depend on:

A. The security of RSA depends on the ***difficulty of factoring large integers***.

B. The security of ElGamal algorithm depends on the ***difficulty of computing discrete logs in a large prime modulus***.

The proposed PKFHE, since it is a public key FHE and based on Euler's theorem, we treat the security of it as the security of RSA cryptosystem, which depends on the difficulty of factoring large integers as shown in above.

7. CONCLUTION

In this paper, we have a Public Key Fully Homomorphic Encryption scheme (PKFHE) essentially based on Euler's theorem and it proved his correctness of supporting additive and multiplicative homomorphism and we offered an example explains the steps of how it works. As a result, we compute the time complexity of the encryption and decryption function for the Elgamal, RSA cryptosystems and our proposed PKFHE scheme which, the order of encryption function of all the schemes is O $((\log (n))^3)$ and the order of decryption function of Elgamal, RSA cryptosystems is O $(\log (n)^3)$, while the order of decryption function of PKFHE scheme is O $(\log (n)^2)$ that mean the proposed PKFHE scheme is faster in the time complexity. Also compared the execution time among the three schemes with five sizes of the messages and used five lengths of a secret key, Where we concluded that the execution time of the proposed scheme was slower than the execution time of Elgamal, RSA cryptosystems except in the case of the key 128-bit which, the execution time of RSA was slower than the two others. Also we noticed that the length of the private key was effect on the execution time of the algorithms in which the execution time increased when the length of the private key was increase. Finally the security of the schemes is analyzed. We applied the proposed scheme PKFHE on a cloud computing and it performed both addition and multiplication operations at the same time on ciphertext without decryption.

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

REFERENCES

[1] Srinivasan, S., and R. Bala Krishnan. "Data property analyzer for information storage in cloud." *Pattern Recognition, Informatics and Mobile Engineering (PRIME), 2013 International Conference on*. IEEE, 2013.

[2] Yang, Jing, et al. "Simulation Study Based on Somewhat Homomorphic Encryption." *Journal of Computer and Communications* 2.02 (2014): 109.

[3] Gentry, Craig. "Computing arbitrary functions of encrypted data." *Communications of the ACM* 53.3 (2010): 97-105.

[4] Gentry, Craig. "Fully homomorphic encryption using ideal lattices." *STOC*. Vol. 9. No. 2009. 2009.

[5] Rivest, Ronald L., Len Adleman, and Michael L. Dertouzos."On data banks and privacy homomorphisms." *Foundations of secure computation* 4.11 (1978): 169-180.

[6] Rivest, Ronald L., Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems." *Communications of the ACM* 21.2 (1978): 120-126.

[7] Yao, Andrew C. "Protocols for secure computations." *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*. IEEE, 1982.

[8] Goldwasser, Shafi, and Silvio Micali. "Probabilistic encryption." *Journal of computer and system sciences* 28.2 (1984): 270-299.

[9] ElGamal, Taher. "A public key cryptosystem and a signature scheme based on discrete logarithms." *IEEE transactions on information theory* 31.4 (1985): 469-472.

[10] Paillier, Pascal. "Public-key cryptosystems based on composite degree residuosity classes." *Eurocrypt*. Vol. 99. 1999.

[11] Fontaine, Caroline, and Fabien Galand. "A survey of homomorphic encryption for nonspecialists." *EURASIP Journal on Information Security* 1 (2009): 41-50.

[12] Smart, Nigel P., and Frederik Vercauteren. "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes." *Public Key Cryptography*. Vol. 6056. 2010.

[13] Van Dijk, Marten, et al. "Fully homomorphic encryption over the integers." *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer Berlin Heidelberg, 2010.

[14] Stehlé, Damien, and Ron Steinfeld. "Faster fully homomorphic encryption." *Advances in Cryptology-ASIACRYPT 2010* (2010): 377-394.

[15] Ramaiah, Y. Govinda, and G. Vijaya Kumari. "Efficient public key homomorphic encryption over integer plaintexts." *Information Security and Intelligence Control (ISIC), 2012 International Conference on*. IEEE, 2012.

[16] Cheon, Jung Hee, et al. "Batch fully homomorphic encryption over the integers." *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Berlin, Heidelberg, 2013.

[17] Emura, Keita, et al. "Chosen Ciphertext Secure Keyed-Homomorphic Public-Key Encryption." *Public Key Cryptography*. 2013.

[18] Cheon, Jung Hee, et al. "CRT-based fully homomorphic encryption over the integers." *Information Sciences* 310 (2015): 149-162.

[19] Diffie, Whitfield, and Martin Hellman. "New directions in cryptography." *IEEE transactions on Information Theory* 22.6 (1976): 644-654.

[20] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. "Evaluating 2-DNF formulas on ciphertexts." In Theory of Cryptography Conference, TCC'2005, volume 3378 of Lecture Notes in Computer Science, pages 325-341. Springer, 2005.

[21] Tebaa, Maha, Saïd El Hajji, and Abdellatif El Ghazi. "Homomorphic encryption applied to the cloud computing security." *Proceedings of the World Congress on Engineering*. Vol. 1. 2012.

[22] Ogburn, Monique, Claude Turner, and Pushkar Dahal. "Homomorphic encryption." *Procedia Computer Science* 20 (2013): 502-509.

[23] Guang-Li, Xiang, et al. "A method of homomorphic encryption." *Wuhan University Journal of Natural Sciences* 11.1 (2006): 181-184.

[24] Pallavi. Homomorphic encryption schemes: steps to improve the proficiency, International Journal of Science Technology and Management, Vol. No.5, Issue No. 02, 2016.

[25] Benzekki, Kamal, A. E. Fergougui, and A. E. B. E. Alaoui. "A secure cloud computing architecture using homomorphic encryption." *International Journal of*

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

Advanced Computer Science and Applications (IJACSA) 7.2 (2016): 293-298.

[26] Potey, Manish M., C. A. Dhote, and Deepak H. Sharma. "Homomorphic Encryption for Security of Cloud Data." *Procedia Computer Science* 79 (2016): 175-181.

[27] Ali M. Sagheer, "Enhancement of Elliptic Curves Cryptography Methods", *MSc Thesis, Computer Science Department, University of Technology,Iraq,2004.*