A Proposed Fully Homomorphic for Securing Cloud Banking Data at Rest

Zana Thalage Omar\textsuperscript{1,2*}, Fadhil Salman abed\textsuperscript{1,2}

\textsuperscript{1}University of Human Development, College of Science and Technology, Department of Computer Science, Sulaymaniyah, Kurdistan Region of Iraq, Iraq, \textsuperscript{2}University of Sulaimani, College of Science, Computer Department, Sulaymaniyah, Kurdistan Region of Iraq, Iraq

\begin{abstract}
Fully homomorphic encryption (FHE) reaped the importance and amazement of most researchers and followers in data encryption issues, as programs are allowed to perform arithmetic operations on encrypted data without decrypting it and obtain results similar to the effects of arithmetic operations on unencrypted data. The first (FHE) model was introduced by Craig Gentry in 2009, and it was just theoretical research, but later significant progress was made on it, this research offers FHE system based on directly of factoring big prime numbers which consider open problem now. The proposed scheme offers a fully homomorphic system for data encryption and stores it in encrypted form on the cloud based on a new algorithm that has been tried on a local cloud and compared with two previous encryption systems (RSA and Paillier) and shows us that this algorithm reduces the time of encryption and decryption by 5 times compared to other systems.

\textbf{Index Terms:} Cloud Computing Security, Encryption, Decryption, Cloud Storage, Homomorphic Encryption
\end{abstract}

1. INTRODUCTION

Computing technology is seeing significant progress and significant interest, especially when the computation outsourcing has been outsourced to a third party as the cloud is the most frequently used form \cite{1}. That is why many companies no longer trust to store their sensitive data in the cloud, which uses traditional unsecured encryption systems \cite{2}. From this, the need to use homomorphic encryption for banking data is coming, which is a new approach that can help banks to increase data security and management \cite{3}. There are two types of homomorphic cryptosystems: Partially homomorphic systems and fully homomorphic systems \cite{4}. Partially homomorphic schemes support one of the additions or multiplication operations, these systems are divided into two parts according to the process that supports like the RSA, where it only supports the multiplication process and does not support the addition process, for example, if we have two numbers M1, M2 and they are encrypted by the RSA, then its value becomes C1, C2 and on obtaining the product of multiplying the two encrypted values C1 * C2 = C3 and then we decrypt the encrypted output C3, we will get a result similar to M1 * M2 = M3, but if we add the two values C1 + C2 = C4 and when decrypting the result C4 we do not get a result similar to M1 + M2 = M4. On the contrary, when the two values are encrypted using Paillier, we find that only the result of C1 + C2 = C4 and when decrypting the result C4 we do not get a result similar to M1 + M2 = M4. Therefore, we say that the two algorithms (RSA and Paillier) are not a fully homomorphic systems \cite{5}, \cite{6}. The first FHE was given in 2009 by Craig Gentry \cite{7}. Researchers first researched a (FHE) system in the late last century, specifically at the end of the seventies, and soon after, in 1987, RSA was published, the RSA algorithm became a leading approach by many researchers because at that time there was no idea of the public key cipher
that was presented during the RSA scheme for the first time [5]. Because this kind of encryption allows the key to decrypt the encrypted data, and thus one can read and know all the data, and for this reason, if one does not have the secret key, the data become useless. Therefore, a question and an issue were asked: Can mathematical operations apply to encrypted data without decrypting it, and from this, the idea of using fully homomorphic systems (FHE) was raised. After that, several attempts were made to develop these systems, but most of the research did not succeed as they received partially homomorphic schemes such as RSA and Goldwasser-Micali [8]. The algorithm that achieves the addition and multiplication properties can be considered as FHE, as it is regarded as a special algorithm that contains the feature of performing mathematical operations (addition and multiplication) on data without decrypting it and obtaining correct results [9]. FHE is an encryption technology that allows calculations to be performed on encrypted data without decrypting it, and this results in an encrypted result where when this result is decrypted we get a result similar to the result of the calculations on the data without encrypting it [9]. The world of computing is in constant progress, and the main challenge is to create a guarantee and trust among customers when storing their sensitive data on the cloud to ensure and respect their privacy. This is a new approach that cloud providers follow to encrypt users’ data, upload it to the cloud, and perform operations on it without decrypting it to ensure the integrity of customer data [10]. This paper presents a fully homomorphic system (the correct numbers and texts) based on a new algorithm that will be explained later in this paper as this scheme relies on data encryption and operations performed on it without decrypting and reducing computational complications and the time used to encrypt and decrypt data and reduce energy consumption. Most of the previous research in this field deals with data when encrypting after converting it to the binary system and this means more time. As for our current research, data operations are encrypted without the need to convert them to binary representation and this reduces mathematical operations and there is a reduction in the time of encryption and encryption solution, as well as a mathematical model has been suggested that deals with the inverse calculation and the process of raising to the exponential and increases the complexity of attacking the new system.

2. LITERATURE REVIEW

C. Gentry et al. (2012), this paper introduces contrast/orientation techniques to transfer the elements of plain text across these vectors very efficiently so that they are able to perform general calculations in a batch way without the need to decrypt the text and also make some improvements that can accelerate the normal homomorphic, where you can make homogeneous evaluation of arithmetic operations using multi-arithmetic head only [11].

J. Fan et al. (2012), this paper concludes two copies of the redefinition that lead to a quick calculation of homogeneous processes using the parameter transformation trick, as this paper conveys Brakerski’s fully homomorphic scheme based on the learning with errors (LWE) problem to the ring-LWE [12].

Z. Brakerski et al. (2012), this paper introduces squash and bootstrapping techniques to convert a somewhat symmetric encryption scheme into an integrated symmetric encryption scheme [13].

X. Cao et al. (2014), this paper presents a completely symmetric encryption scheme using only a basic unit calculation as it relies on the technique of using multiplication and addition instead of using ideal clamps on a polynomial loop [14].

C. Xiang et al. (2014), this paper presents an entirely symmetric encryption scheme on integers, as it reduces the size of the public key using the square model encryption method in public key elements instead of using a linear model based on a stronger variant of the approximate-GCD problem [15].

M. M. Potey et al. (2016), this paper presents a completely symmetric encryption system where it focuses on storing customer data on the cloud in an encrypted form so that customer data remain safe and when any data modification is made the system loads data on the customer’s device and modifies it and then stores it again on the cloud in encrypted form [16].

K. Gai et al. (2018), this paper proposes a new solution for mixing real numbers on a novel tensor-based FHE solution that uses tensor laws to reduce the risk of unencrypted data storage [17].

S. S. Hamad et al. (2018), these heirs offer a completely symmetric encryption system, as it relies on the principle of encryption a number from the plain text with another number using a secret key without converting to binary format and then comparing the result with a DGHV and SDC system [18].

S. S. Hamad et al. (2018), this paper presents a fully homomorphic encryption system based on Euler’s theory and the time complexity has been calculated and compared with other systems with an encrypt key size up to 512 bits while the size of the key in our proposed scheme reaches more
than 2048 bits and the encrypting process is done through more complex and powerful mathematical equations [19].

V. Kumar et al (2018), this paper presents fully homomorphic encryption system with probabilistic encrypting and relies on Euler's theory. The encrypting process is done through the following mathematical equation \( C = M^{e \cdot \varphi(n) + 1} \mod n \) while in our proposed scheme a more complex and difficult mathematical algorithm is used which helps to stand more against hacker attacks and deter them [20].

R. F. Hassan et al. (2019), this paper proposes a blueprint for building asymmetric cloud-based architecture to save user data in the form of unusual text. This pattern uses the elliptic curve to create the secret key for data encryption. This pattern is a new algorithm that reduces processing time and storage space [21].

### 3. STATEMENT OF THE PROBLEM

Cloud providers provide many services, including applications and storage many companies and users do not trust the providers of these services due to security concerns. Where the user does not upload his personal data to the cloud because the cloud providers are able to read and modify every bit loaded on the cloud and use it for personal purposes, and this thing does not comply with respecting the user's privacy. Furthermore, some cloud providers still use traditional security techniques that are not secure with low-security level to protect user privacy. Some of the cloud providers have started to use high-level technologies to protect the privacy of users and the security of their data, but there remains a problem that the provider of the cloud itself is still able to access user data, and this is not safe for users. This problem can be solved when following FHE systems when storing data on the cloud where these systems can encrypt the data and store it in the cloud in an encrypted form and thus the cloud provider or others cannot see the data and use it, so the privacy of users and the security of their data are protected.

### 4. PROPOSED FHE SYSTEM

The proposed scheme works as follows:

Generating the encryption key and then encrypting the numbers and texts and storing them in encrypted form on the cloud. In our work, we use a local cloud and experiment with the proposed scheme on it. The purpose of this process is to save the data encrypted on the cloud so that no one can view the data and use it for personal purposes. Therefore, when the data owner needs to perform an amendment of the encrypted data on the cloud, an encrypted request is sent to the server, and the server performs mathematical operations on the encrypted data and returns an encrypted result where this encrypted result can only be decrypted through the private encryption key which is with the owner of data only so that he can decrypt the encrypted result and see his data. In this way, we have been able to maintain the privacy and security of the data when stored in the cloud. These procedures go through three stages. Generation the encryption key stage, the encryption stage, and the decryption stage. The model of the proposed scheme is given in Fig. 1, and the flowchart of the proposed scheme is given in Fig. 2. The proposed scheme performs several random examples with multiplication and addition as follows:

**A. Key Generation:**
1. Generate two large Prime number p and q
2. Compute \( n = p \cdot q \)
3. Calculate \( L = ((P^{-1} \mod q) \cdot p) + ((q^{-1} \mod p) \cdot q) \)
4. Select \( r \): Where \( r \) is a big random integer

**B. Messages Encryption**
The conditions:

\( (M_1 \cdot M_2), (M_1 + M_2), (M_1 \cdot M_2) < n \)

Where \( M_1 \) and \( M_2 \) are the Messages.
The schema of message encryption is:
\[ C = L \cdot M^{\mu(p) + 1} \mod n. \]  
\[ \text{(1)} \]
Where \( \mu(p) = (p-1) \), Euler function and \( C \), the ciphers text.

C. Message Decryption
The schema of cipher decryption is:
\[ M = C \mod p \]  
\[ \text{(2)} \]
Where \( M \) is the number or text that will be encrypt and \( C \) is the result of the encrypted number or text we named it cipher text

D. Euler’s Theorem
All of us know that Euler’s Theorem contains two-part they are:
1. \( M^{\mu(p)} \equiv 1 \mod p \), when \( p \) and \( m \) are prime to each other.
2. \( M^{r \cdot \mu(p) + 1} \equiv M \mod n \), when \( r \) is an integer, \( M < n \) and \( n = p \cdot q \) where \( q \) and \( p \) are two primes number.

E. A simple example of how to make an amendment to encrypted data
We have two values \( M_1 = 3 \), \( M_2 = 5 \). We encrypt them through a simple encryption equation that is multiplied by each value, so we get \( C_1 = M_1 \times 2 \) and \( C_2 = M_2 \times 2 \), so \( C_1 = 6, C_2 = 10 \) when we add the two values \( C_1 + C_2 = C_3 \) so \( C_3 = 16 \) We decrypt \( C_3 \) so we get the result \( 16/2 = 8 \) which is the same result when we add \( M_1 + M_2 = M_3 \) where \( 3 + 5 = 8 \) as shown in Fig. 3.

5. THE PROVE OF OUR SCHEMA IS FHE

We choose two numbers \( M_1, M_2 \) and encrypt them to get two encrypted or (ciphers) \( C_1 \) and \( C_2 \), respectively, and then we combine \( C_1 + C_2 \) to get a new ciphered result we name it \( C_3 \) then we decrypt \( C_3 \) and compare the result with \( M_3 \) which is the result of combine \( M_1 + M_2 \) we also multiply \( C_1 \times C_2 \) to get \( C_4 \) and compare it to \( M_4 \) which is the result of \( M_1 \times M_2 \).

A. The Prove of Additive Homomorphic
If the following condition is fulfilled, it becomes clear to us that the proposed scheme additive homomorphic:

\[ M_1 + M_2 = \text{dec} \{ \text{enc}(M_1) + \text{enc}(M_2) \} \]  
\[ \text{(4)} \]
Where \( \text{dec} \) is the decryption function and \( \text{enc} \) is the encryption function

Proof:
\[ C_1 = L \cdot (M_1^{\mu(p) + 1} \mod n). \]
\[ C_2 = L \cdot (M_2^{\mu(p) + 1} \mod n). \]
\[ C_1 + C_2 = L \cdot (M_1^{\mu(p) + 1} \mod n) + L \cdot (M_2^{\mu(p) + 1} \mod n). \]
\[ \text{dec} (C_1 + C_2) = (C_1 + C_2) \mod p \]
\[ = [L \cdot (M_1^{\mu(p) + 1} \mod n) + L \cdot (M_2^{\mu(p) + 1} \mod n)] \mod p \]
\[ = [(L \mod p) + ((M_1^{\mu(p) + 1} \mod n) \mod p) + (L \mod p) + ((M_2^{\mu(p) + 1} \mod n) \mod p)] \mod p] \]
\[ = [(M_1^{\mu(p) + 1} \mod n) \mod p + (M_2^{\mu(p) + 1} \mod p) \mod n] \mod p \]

We know \( M_1^{\mu(p) + 1} \mod p = M_1 \) and \( M_2^{\mu(p) + 1} \mod p = M_2 \) by Euler’s Theorem so
(M_1 \mod n) + (M_2 \mod n)
= (M_1 + M_2) \mod n
Because M_1 + M_2 less than < (n)
= M_1 + M_2
\text{dec} (C_1 + C_2) = M_1 + M_2 so the condition is fulfilled

B. The Prove of Multiplicative Homomorphic

If the following condition is fulfilled, it becomes clear to us that the proposed scheme multiplicative homomorphic:

\[ M_1 \times M_2 = \text{dec} [\text{enc} (M_1) \times \text{enc} (M_2)] \quad (5) \]

Where \text{dec} is the decryption function and \text{enc} is the encryption function

Proof:
\[ C_1' = L \cdot (M_1 \cdot r \cdot \mu (p) + 1) \mod n \]
\[ C_2' = L \cdot (M_2 \cdot r \cdot \mu (p) + 1) \mod n \]
\[ C_1' \cdot C_2' = (L \cdot (M_1 \cdot r \cdot \mu (p) + 1) \mod n) \cdot (L \cdot (M_2 \cdot r \cdot \mu (p) + 1) \mod n) \]
\[ \text{dec} (C_1' \cdot C_2') = \text{dec} (C_1' \cdot C_2') \mod p \]
\[ = (L \cdot (M_1 \cdot r \cdot \mu (p) + 1) \mod n) \cdot (L \cdot (M_2 \cdot r \cdot \mu (p) + 1) \mod n) \]
\[ = (M_1 \cdot r \cdot \mu (p) + 1) \mod n \cdot (M_2 \cdot r \cdot \mu (p) + 1) \mod n \]
We know \( M_1 \cdot r \cdot \mu (p) + 1 \mod n \) and \( M_2 \cdot r \cdot \mu (p) + 1 \mod n \) by Euler’s Theorem so
\[ = (M_1 \mod n) \cdot (M_2 \mod n) \]
\[ = (M_1 \cdot M_2) \mod n \]
Because \( M_1 \cdot M_2 \) less than < (n)
\[ = M_1 \cdot M_2 \]
\[ \text{dec} (C_1' \cdot C_2') = M_1 \cdot M_2 \text{ so the condition is fulfilled} \]

6. REAL EXAMPLE

Let us choose two different number \( M_1 = 10 \), \( M_2 = 40 \), select two big prime numbers \( p=523 \), \( q=617 \), select random number \( r=100 \) and compute \( n \), \( L \) where \( n = p \cdot q \) and \( L = (P^{-1} \mod q) \cdot p + (q^{-1} \mod p) \cdot q \), as in Fig. 1, so \( n = 322691 \) and \( L = 322692 \) now we will compute \( C_1 \), \( C_2 \) as shown in Fig. 4 where

\[ C_1 = L \cdot (M_1 \cdot r \cdot \mu (p) + 1) \mod n \]
\[ C_2 = 322692 \cdot (10^{100}) \mod 322691 \]
\[ C_1 = 84555952836 \]
\[ C_2 = 322692 \cdot (40^{100}) \mod 322691 \]
\[ C_2 = 70220360736 \]

A. Check the Additive Homomorphism

As shown in Fig. 5, Let us define \( C_3 \) is the result of \( C_1 + C_2 \)

\[ C_3 = C_1 + C_2 \]
\[ C_3 = 84555952836 + 70220360736 \]
\[ C_3 = 154776313572 \]
\[ M_3 = C_3 \mod p \]
\[ M_3 = 154776313572 \mod 523 \]
\[ M_3 = 50 \text{ which is the same of } M_1 + M_2 = 10 + 40 = 50 \]

B. Check the Multiplication Homomorphism

As shown in Fig. 6, Let us define \( C_4 \) is the result of \( C_1 \cdot C_2 \)

\[ C_4 = C_1 \cdot C_2 \]
\[ C_4 = 84555952836 \cdot 70220360736 \]
\[ C_4 = 5937549510520122247296 \]
\[ M_4 = C_4 \mod p \]
\[ M_4 = 5937549510520122247296 \mod 523 \]
\[ M_4 = 50 \text{ which is the same of } M_1 \cdot M_2 = 10 \cdot 40 = 400 \]

7. RESULTS

Our proposed method has been applied in Java Language on a laptop that has these characteristics Intel (R) core (TM)
Omar and abed: Cloud Banking Data Security

### Table

<table>
<thead>
<tr>
<th>Select Prime number for P</th>
<th>523</th>
<th>617</th>
<th>713</th>
<th>821</th>
<th>1009</th>
<th>1223</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Prime number for Q</td>
<td>557</td>
<td>773</td>
<td>1061</td>
<td>1361</td>
<td>1663</td>
<td>3131</td>
</tr>
<tr>
<td>Select Prime number for R</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Generate Key</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Dec Number 1</td>
<td>84559652836</td>
<td>84559652836</td>
<td>84559652836</td>
<td>84559652836</td>
<td>84559652836</td>
<td>84559652836</td>
</tr>
<tr>
<td>Dec Number 2</td>
<td>7020306736</td>
<td>7020306736</td>
<td>7020306736</td>
<td>7020306736</td>
<td>7020306736</td>
<td>7020306736</td>
</tr>
<tr>
<td>Dec The Result</td>
<td>154776313572</td>
<td>154776313572</td>
<td>154776313572</td>
<td>154776313572</td>
<td>154776313572</td>
<td>154776313572</td>
</tr>
</tbody>
</table>

### Fig. 5. Verification that the proposed scheme supports additive homomorphic system.

### Fig. 6. A real example of generating an encryption key and encrypting two different numbers.

The language considered at the university is English.

大学考慮の言語は英語

The language considered at the university is English.

大学考慮の言語は英語

The language considered at the university is English.

大学考慮の言語は英語

The language considered at the university is English.

大学考慮の言語は英語

The language considered at the university is English.

大学考慮の言語は英語

The language considered at the university is English.

大学考慮の言語は英語
Omar and abed: Cloud Banking Data Security

0064335766935428531920125614334654205627755229
67733815424718245851181348017265387648390873338
1520305739379477275508814108236135822895062302531
94050622551405063552873019444449238666440140085803
282915319755486979906430558612883401366594381416
546811288365649567304972181175836521739451237520
50707601405826938178893152614067930451476175622
29294044441603942736262064420492291134834700560
07271825256901103199457484857

Fig. 7. Computation encryption time of various schema.

Fig. 8. Computation decryption time of various schema.

TABLE 1: Computation encryption time of various schema

<table>
<thead>
<tr>
<th>Key Size</th>
<th>Proposed method</th>
<th>RSA</th>
<th>Pailler</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 Bit</td>
<td>88 ms</td>
<td>59 ms</td>
<td>103 ms</td>
</tr>
<tr>
<td>128 Bit</td>
<td>139 ms</td>
<td>182 ms</td>
<td>182 ms</td>
</tr>
<tr>
<td>256 Bit</td>
<td>218 ms</td>
<td>727 ms</td>
<td>727 ms</td>
</tr>
<tr>
<td>512 Bit</td>
<td>1141 ms</td>
<td>4212 ms</td>
<td>4212 ms</td>
</tr>
<tr>
<td>1024 Bit</td>
<td>6058 ms</td>
<td>55139 ms</td>
<td>55139 ms</td>
</tr>
<tr>
<td>2048 Bit</td>
<td>65876 ms</td>
<td>263303 ms</td>
<td>263303 ms</td>
</tr>
<tr>
<td>Average</td>
<td>12253 ms</td>
<td>5451 ms</td>
<td>53944 ms</td>
</tr>
</tbody>
</table>

TABLE 2: Computation decryption time of various schema

<table>
<thead>
<tr>
<th>Key Size</th>
<th>Proposed method</th>
<th>RSA</th>
<th>Pailler</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 Bit</td>
<td>31 ms</td>
<td>72 ms</td>
<td>193 ms</td>
</tr>
<tr>
<td>128 Bit</td>
<td>43 ms</td>
<td>116 ms</td>
<td>330 ms</td>
</tr>
<tr>
<td>256 Bit</td>
<td>50 ms</td>
<td>315 ms</td>
<td>1429 ms</td>
</tr>
<tr>
<td>512 Bit</td>
<td>60 ms</td>
<td>1450 ms</td>
<td>11441 ms</td>
</tr>
<tr>
<td>1024 Bit</td>
<td>131 ms</td>
<td>122628 ms</td>
<td>122628 ms</td>
</tr>
<tr>
<td>2048 Bit</td>
<td>260 ms</td>
<td>976289 ms</td>
<td>976289 ms</td>
</tr>
<tr>
<td>Average</td>
<td>95 ms</td>
<td>14898 ms</td>
<td>185385 ms</td>
</tr>
</tbody>
</table>

The Message after encryption (Cipher Text)

L=24236073045746910110572201683394297457755108
7685682557530519112512074767485603470169601180
558389243234424825336810918650824789494269717337
049640187572585863569789414738414794462518054
2517408399006433576693542853192012561434336
54205627755229677338154247182845581881348017260
53876483908783338152305739774727550881410823
6013852289506203253190450622514510635528730194
444492386664401400858032829153319755489679604
3055861288340136659438141654681128836546956730
947281175838652173945123752050707687014058269
3187898315261406793045147617562229249044441603
942377626206442049229113483470056007271825256
265091103199457484858

TABLE 1: Computation encryption time of various schema

<table>
<thead>
<tr>
<th>Key Size</th>
<th>Proposed method</th>
<th>RSA</th>
<th>Pailler</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 Bit</td>
<td>59 ms</td>
<td>103 ms</td>
<td>103 ms</td>
</tr>
<tr>
<td>128 Bit</td>
<td>182 ms</td>
<td>182 ms</td>
<td>182 ms</td>
</tr>
<tr>
<td>256 Bit</td>
<td>727 ms</td>
<td>727 ms</td>
<td>727 ms</td>
</tr>
<tr>
<td>512 Bit</td>
<td>4212 ms</td>
<td>4212 ms</td>
<td>4212 ms</td>
</tr>
<tr>
<td>1024 Bit</td>
<td>55139 ms</td>
<td>55139 ms</td>
<td>55139 ms</td>
</tr>
<tr>
<td>2048 Bit</td>
<td>263303 ms</td>
<td>263303 ms</td>
<td>263303 ms</td>
</tr>
<tr>
<td>Average</td>
<td>53944 ms</td>
<td>53944 ms</td>
<td>53944 ms</td>
</tr>
</tbody>
</table>

TABLE 2: Computation decryption time of various schema

<table>
<thead>
<tr>
<th>Key Size</th>
<th>Proposed method</th>
<th>RSA</th>
<th>Pailler</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 Bit</td>
<td>72 ms</td>
<td>193 ms</td>
<td>193 ms</td>
</tr>
<tr>
<td>128 Bit</td>
<td>116 ms</td>
<td>330 ms</td>
<td>330 ms</td>
</tr>
<tr>
<td>256 Bit</td>
<td>315 ms</td>
<td>1429 ms</td>
<td>1429 ms</td>
</tr>
<tr>
<td>512 Bit</td>
<td>1450 ms</td>
<td>11441 ms</td>
<td>11441 ms</td>
</tr>
<tr>
<td>1024 Bit</td>
<td>122628 ms</td>
<td>122628 ms</td>
<td>122628 ms</td>
</tr>
<tr>
<td>2048 Bit</td>
<td>976289 ms</td>
<td>976289 ms</td>
<td>976289 ms</td>
</tr>
<tr>
<td>Average</td>
<td>185385 ms</td>
<td>185385 ms</td>
<td>185385 ms</td>
</tr>
</tbody>
</table>
Omar and Abed: Cloud Banking Data Security

REFERENCES


8. CONCLUSION

Our scheme relies on FHE on whole numbers, texts, and supports all languages such as English, Arabic, Kurdi, and Chinese and others. Very large prime numbers (up to 617 digits, 2048 bit) represent the strength for the attack of our scheme because the proposed system depends on the problem of factorization to the primary factors, which are considered mathematical problems under discussion at the present time when taking the time. We have come to the conclusion that our scheme is very effective in relation to the time when encrypt and decrypt numbers and texts with other techniques and approaches that are circulated and used at the present time.

REFERENCES


