Multi-layer encryption algorithm for data integrity in cloud computing

Arwa Zabian¹,*, Shakir Mrayyen², Abram magdy Jonan³, Tareq Al-shaikh⁴, Mohammed tthazi Al-khaiyat⁵

*Faculty of Science and Information Technology, Jadara University
¹ rw@jadara.edu.jo*, ² smrayyen@jadara.edu.jo, ³ abramjonan@gmail.com, ⁴ tareqalshaikh996@gmail.com, ⁵kh.hmada@yahoo.com
* corresponding author

1. Introduction

The introduction of cloud computing in all life aspects has changed the way of handling every think in our life. Today, with cloud computing is decreased the possibility of losing or damaging our important data like names, photos, important documents, which added many benefits on data handling like flexibility, availability, capacity. Data is becoming available any time, any where. The greater the facilities, less our data privacy. For that maintaining data confidentiality and integrity in the Internet and cloud world is a primary requirement to continue working and using these facilities. In this paper, we propose a security mechanism that makes accessing the data from unauthorized persons more complex. Our mechanism easy to implement and understand but is complex to decrypt because it works in three levels in each level is used a new encryption key generated randomly and in each level is used a different encryption algorithm. Our results show that the encryption/decryption process time is increased linearly with the text size for the authorized persons and is increased exponentially for unauthorized persons.

This is an open access article under the CC-BY-SA license.
of information, to benefit from these facilities information security is an essential requirement. As more people work on information security, as more people tries to access the data in illegal manner and tries in different manners to breach the systems security. Encryption is an easy way that help in protecting our data to be accessed only by authorized persons. Cloud computing is composed of hardware, software, storage, network, interface and services that provides the user facilities, applications, services on demand which are independent of user location.

Cloud computing reduced the hardware cost, increased the storage space and give maximum capability with minimum resources that means providing services to the user with best use of resources. To allow the user to utilize all these benefits of cloud computing, his data must be preserved from change, modification, intercept and any other operation that can be breach the data security.

When we speak about cloud we speak about three tier architecture in which the client part is the first tier, the server part is the second tier and the database part is the third tier, data security is important in each part, figure 1. Data privacy is an aspect of information technology that deals with the ability to determine who can read, access our data while it is in transmission over the network or stored on the cloud server.

Encryption is one of the methods used for data protection, encryption means converting the message called plain text to incomprehensive text using some of encryption algorithms, the goal of encryption algorithms is to make the plain text unclear for unauthorized persons, the authorized persons can use a decryption algorithm to decipher the text and read it. The decryption process is the reverse process of encryption. There are two types of encryption are symmetric encryption in which the same key is used for encryption and decryption, and asymmetric encryption in which two different keys are used, one is used for encryption and the other is used for decryption. In this paper, we will propose a multilevel symmetric encryption/decryption algorithm for maintaining data integrity in cloud computing. As mentioned previously, in the cloud data is subject to many threats while transmitting over the network or when stored on the server. Data integrity means data can be accessed and modified only by the authorized persons. In this paper we are interested in presenting a three tier encryption/decryption algorithm that allow the user data to be transmitted encrypted and stored encrypted on the cloud server. In our proposed system, data is transferred and stored and retrieved encrypted only the authorized persons that will read and modify the data have the decryption keys. Encrypting data more than one time with different method and different keys will increase the number of attempts needed to crack the key and will make the access to the plain text more difficult.

Problem formulation There are many encryption algorithms proposed to maintain data integrity and privacy but all these algorithms work separately and concern on working between sender and receiver, where data is encrypted in one part and decrypted in the other part. In our work, we will use three of these algorithms consecutively to make accessing the data more difficult and complicated. Penetrating the data in our system require knowing many information, it requires cracking three keys and knowing how three encryption algorithms will work.
This paper is organized as follows: Section 2 will present our proposed algorithm. Section 3 will present the algorithm analysis. In Section 4, contains simulation results. Section 5, will present some of the related works in the field of multilevel encryption/decryption algorithms and finally conclusion and future works.

2. Algorithm Description

Our algorithm work in phases, in the first and the second phase is used the substitution method to encrypt the data with two different keys and two different methods. In the third phase the resultant text will be converted to binary and XORed with a new key and then stored in binary or hexadecimal number.

2.1 Encryption algorithm

1) Assumptions

M will refer to the message that is plain text
E will refer to encrypted text and
K will refer to the key.

2) First phase

- Insert the plain text M
- Generate a random key k1 with length of 10-30 character, the key can be composed on small, capital letters and the numbers from 0-9
- Prepare a matrix 8 × 8 in which is inserted in the first location (1 × 1) @ character and in the last location (8 × 8) the # character. Then, is inserted the key characters one by one by without repetition, where in each cell is inserted only one character. Then complete the matrix with the remaining characters and numbers that are not presented in the key. To generate the first encryption text E1 follow the following rules:

3) Encryption rules

Divide the plain text M into different parts each one is composed on only two characters. Then, each two characters are encrypted together following the following rules:

- If the two characters are located on the same row, the encryption of each character is that the character that follow it on the same row from left to right
- If the two characters are on the same column, the encryption of each one is that the character that follow it on the same column from top to down.
- If the two characters are located in two different columns, two different rows the encryption is the cross.
- To encrypt the last character in the row, move in counter clockwise
- To encrypt the last character in the column, move in circular manner from bottom to up.
- The consecutive repeated character are separated by the @ character, and encrypted following the rules from 1-5.
- If the size of the text is odd, add at the end the @ character and encrypt using the rules from 1-5.

4) Second phase

Generate k2 that is a random number between 2-63, and calculate the encryption text using the following equation:

\[ E_2 = (E_1 + k_2) \mod 63 \]  \hspace{1cm} (1)

5) Third phase
• Convert E2 to hexadecimal then binary and store it into E3
• Generate a new hexadecimal key k3, convert to binary, then apply the following equation for encryption:

\[ E = E_3 \oplus k_3 \] (2)

Example:

In this example we will describe in simple mode how the algorithm will work taking a fixed key value, but in the implemented system the three keys are selected randomly. That means encrypting the file different times will be encrypted by different keys, but in each one time the key is stored to be used in the decryption phase

M = extend, k= Sandra09

division phase : extend

matrix : 

\[ \begin{array}{cccccccc}
@ & S & a & n & d & r & 0 & 9 \\
A & B & C & D & E & F & G & H \\
I & J & K & L & M & N & O & P \\
Q & R & T & U & V & W & X & Y \\
Z & b & c & e & f & g & h & i \\
j & k & l & m & o & p & q & s \\
t & u & v & w & x & y & z & 1 \\
2 & 3 & 4 & 5 & 6 & 7 & 8 & # \\
\end{array} \]

encryption : 

\[ \begin{array}{cccc}
& ex & te & nd \\
w & Z & w & dr \\
\end{array} \]

So, the encrypted text in the first phase E1 = "wf Zwdr".

In the second encryption phase consider the k2 = 26. So, E2 = UE0UCQ

In the third encryption phase E2 is converted to hexadecimal E3 = (554530554351) then binary (010101010100110100000110101000011010100001101010000110101000001) and a random hexadecimal key is generated k3 = (123bf9cab75) then is XORed (exclusive OR) with the binary of E3 the final encrypted text in binary will be :

E = (101000110011011000111110110100001101000001100100). In simulation results section 4.1 is presented a complete scenario of encryption and decryption.

![Encryption algorithm description](image_url)
2.2 Decryption algorithm

As presented in the encryption algorithm the final encrypted text is in binary or in hexadecimal, that is considered the input to the decryption algorithm. Considering that we use symmetric encryption, so the same key is used for encryption and decryption in each phase. The final encrypted text is decrypted three time in reverse manner to arrive to the plain text.

- First decryption phase consist on using exclusive or between the key and the final encrypted text E following the following rules, from equation 2 \( E = E_3 \oplus k_3 \) making exclusive or for the key \( k_3 \) to the two part of equation 2, will produce the following: \( k_3 \oplus E = E_2 \oplus k_3 \oplus k_3 = E_2 \) that means knowing E and \( k_3 \) will turn to \( E_2 \)

- Second decryption phase \( E_1 = E_2 - k_2 \mod 63 \)

- Third decryption phase: to obtain the plain text the decryption is the reverse process follow the same rules applied on the same matrix used in section encryption 2.1 but in the reverse order.

3. Algorithm Analysis

The main idea behind our work were to maintain the data confidentiality and integrity, that mean data can be accessed or changed only by the authorized persons , for that we apply the encryption in levels to ensure that no one can read the plain text without having many information that cannot be obtained in easy manner, in the first level encryption data is encrypted and sent to the cloud server. That means it travel in the network encrypted if some intruder tries to access the data he must know the key and the encryption method, our key is with length 10-30 character for that the number of attempts to crack the key must be in the degree of \( 10^{64-30} \) considering capital and small letters and the numbers from 0-9. In addition our matrix is extended to 64 cells. For that accessing the data cannot be easy for any one. The second level encryption is performed on the server, the number of attempts to crack the second key is on the range to 63 but still the original data hidden. The last encryption step is performed before store the data in the cloud database and the number of attempt to crack the key is 1616 where the key is with length 16 character in hexadecimal system. While we speak about cloud we cannot speak about absolute security, the possibility of attacks all the time exists, in our work we have made the possibility to read the original text more complex. Our algorithm is easy to implement and does not requires complex computation , it is easy to encrypt any text but it is not easy to decrypt it without knowing the needed information.

4. Simulation and performance analysis

Our system is implemented using visual C#, in which is implemented our algorithm in its three phases. In the first phase is implemented the function that represents the encryption rules in the first phase 2.1 on the matrix of size 8*8. In the second phase, is implemented the substitution method that uses a key from 2-63 and is used a random function (next in class random in .net) that generates the key randomly. In the third phase, is used the function “texttobinary” to convert the resultant encrypted text to binary and then apply to it the exclusive or operator with the key that is generated randomly using the function “atmkeygenerator”. Figures 3, 4,5 show an example on the algorithm implementation.

4.1 Performance analysis

Cloud security is an important challenge that poses many problem on the efficiency of cloud services, for that, researchers pay attention on working in this field. In the next section 5, we will presents some of the works published that are concentrated on cloud security, our paper is the first one that is implemented and concatenate three symmetric methods to obtain best results, most of the works are not implemented but only proposed, and some other works consist on simple comparison between algorithms like DES, AES, blowfish. To measure the performance of our work we have use the meantime parameter . Meantime means the difference between starting and ending time of encryption taken for a particular algorithm. Our results show if the text size is increased the encryption time will increase and the decryption time is increased and that is compatible with the results presented in [16] as mentioned in table 1. Best encryption algorithm that easy to encrypt difficult to decrypt.
## Table 1. Encryption-Decryption time

<table>
<thead>
<tr>
<th>Text length (character)</th>
<th>$k_1$ length (character)</th>
<th>$k_2$ length (int)</th>
<th>$k_3$ length (bit)</th>
<th>Encryption time (ms)</th>
<th>Decryption time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>30</td>
<td>10</td>
<td>64</td>
<td>13</td>
<td>22</td>
</tr>
<tr>
<td>100</td>
<td>27</td>
<td>31</td>
<td>64</td>
<td>26</td>
<td>39</td>
</tr>
<tr>
<td>150</td>
<td>23</td>
<td>50</td>
<td>64</td>
<td>38</td>
<td>53</td>
</tr>
<tr>
<td>200</td>
<td>14</td>
<td>40</td>
<td>64</td>
<td>47</td>
<td>72</td>
</tr>
<tr>
<td>250</td>
<td>28</td>
<td>13</td>
<td>64</td>
<td>60</td>
<td>92</td>
</tr>
</tbody>
</table>

### 5. Related Works

Cloud computing is composed on three layers are software as services (Saas), platform as services (Paas), and infrastructure as services (Iaas). The mobile applications is increased day after day. And the use of cloud application is becoming necessary today. So, the user today are involved in all cloud computing layers where is stored his sensitive data like picture, personal information, banking information for that data security is becoming a necessity and a key requirement for all of us today. In [8] is proposed a cipher cloud algorithm that allow the user to upload his file to cipher cloud to
encrypt it and then transfer via encrypted HTTPS connection to the server. Then, it is decrypted on
the server and stored in its original form. The file is uploaded encrypted but retrieved in its original
form, in addition the user can choose the encryption algorithm that can be one of the following: RSA,
DES, AES, Blowfish [10], once the encryption algorithm is chosen, it cannot be changed but the user
can view file, modify, delete, edit it when is uploaded before encrypting it. The paper does not present
any simulation results or any analysis of the system performance. Considering the open challenges in
cryptographic operations in the cloud, some basic measures must be taken in consideration to protect
data against cyber attacks like: preventing direct access to data, maintain data encrypted, there are
some softwares that prevent attacks called Wannakiwi software [4] is based on wana encrypt method
that relies on scanning the address space of the process that generates private user key, and can recover
the key and the files in the case of attacks. It is not a perfect software but is a best solution in the case
of victims without backup data. In [14], is proposed a public key algorithm ECC (Elliptic Curves
Cryptography) that is an open key encryption algorithm that has many advantage over RSA.

Fig. 5. Decryption process last phase

Homomorphic encryption can be proposed as a solution to maintain data security and confiden-
tiality, where a mathematical operations are performed on encrypted data without using original data.
Data are encrypted and decrypted in client side, the service provider work on cipher text only. In [6],
is shown that homomorphic encryption can provide the same level of security of any encryption
system and can maintain a complete privacy between client and cloud server without decreasing
system functionality. In [3], is proposed two times encryption method that is based on HMAC (Hash
Message Authentication Code) scheme to encrypt data two times one time when the file is
uploaded and the other when the file is distributed. Using two times encryption the computation
complexity time of the encryption method is increased. In [9], is proposed ASIF algorithm that is a
hybrid encryption model in which data is compressed then encrypted when the user want to upload
his data to the server, in this manner the model will save space on the server and reduce link and
channel congestion. In this algorithm, is generated two keys (k1, reverse key) then is calculated
random number that represents the position of character in k1, this random number is added to k1
and is applied exclusive or (XOR) on it and the reverse key, then is converted each resultant value to
its correspondent character in ASCII code that will generate k2, then the plain text is XORed with k1
or k2. The length of k1 is equal to 72 bits. The decryption algorithm performs the reverse order of
operation of the encryption algorithm. The results presented in the paper conclusion indicates that the
algorithm performs better security. In [13], is proposed a multilevel encryption algorithm that uses
in the first level AES algorithm and in the second level rounded shifted algorithm that is modified
Caesar algorithm. When the user upload file it is first encrypted using AES, then the resultant cipher
text will be encrypted using Caesar algorithm and stored encrypted on the cloud database. For
retrieve data, it is decrypted on the server in the reverse order of encryption and send the original data
to the server. In [2], is proposed a multilevel encryption technique that use AES algorithm for the text encryption and Elliptic Curve Cryptography to encrypt the key during transmission over the network. In [5], is proposed a multilevel encryption /decryption algorithm that use DES algorithm as first level encryption when the user upload the file to the server, then it is used RSA algorithm as second encryption algorithm, then data is stored on the database encrypted figure.6. For data retrieve data is decrypted in the reverse manner as encryption on the server and sent in plain text to the user figure.7. In this scenario data is travelled in plain text send-receive in the network.

In [11], is proposed an algorithm that uses first a transposition cipher for the plain text, then Caesar cipher is applied to obtain the final encrypted text. The transposition will move down the letter in the odd location, then added at the end of the text.

example: Summer will be divided as follow:

higher level: S m e
low level: u m r. final text be encrypted s m e u m r.

Then the final text will be encrypted using Caesar algorithm. In [1] is proposed the implementation of three level encryption algorithm that uses RSA, random number generator and 3DES to ensure data security. The algorithm is implemented in C# using SQL Azure service database in which the user can choose the order of the algorithm they want. In [15], the author implement the work presented in [5] to be applied on DBMS, the system levels, first data is encrypted using DES then is encrypted using RSA algorithm. The results show that if some intruder access the data and intercept the data, and decrypt it, accessing the original data will be difficult.

![Fig. 6. encryption process](image)

![Fig. 7. decryption process](image)
In [12] is proposed data and key rotation algorithm that works to maintain the integrity of data on the cloud server, in a manner that data is encrypted on the server side and decrypted on the client side. The main basic idea is to use for any character a key different to encrypt and decrypt. The proposed algorithm divide the text file to blocks, the binary equivalent of blocks is stored in an array then this data is circularly rotated and encryption in any rotation is divided the data by 2 to ensure the privacy of data. That means if the file is composed on n blocks each block is with n character, the number of operations performed is n*n and the complexity of the algorithm is O(n*n).

The performance analysis presented in the paper indicates that the proposed encryption / decryption algorithm maintain the security of data without increasing storage space and overhead computation. In[7], to maintain the data security in cloud computing is proposed the homomorphic encryption system that makes an operation on data without decryption. The idea is to store data encrypted and to retrieve it encrypted, in this manner the holder of the private key is only the sender and only he can read or work on plain text. The cloud server make operations, storage only on encrypted data.

6. Conclusion

In this paper we have proposed a new encryption method that apply encryption in three levels to ensure data confidentiality and integrity. Our goal in this work were to distract the attention of the attackers from our data while is in transmission over the internet or on the server. We have made difficult the comprehension of the data even if some one has intercept it successfully, because knowing the explicit text needs to know many information like 3 keys generated randomly and three encryption methods . Our algorithm were easy to implement, it does not need complex computation for encryption or decryption, the time needed to encrypt the data is small while decrypting the data require more time as was presented in our results 4.1. Our future works is to work more on the binary file to make more difficult to return from the encrypted text to its original binary form.

References


