ISSN 2347 - 3983
Volume 8. No. 10, October 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter1338102020.pdf
https://doi.org/10.30534/ijeter/2020/1338102020

Data Confidentiality and Security Enhancement for Cloud Storage utilizing OB-MECC Encryption

P Nagaraju[1], Dr N Nagamalleswara Rao[2]
[1]Research Scholar, ANU College of Engineering and Technology, Gunture, A.P., India and Assistant Professor, Department of CSE, GMRIT, Rajam, Srikakulam, AP, India
[2]Professor, Dept. of IT, RVR & JC College of Engineering, Guntur, AP, India

ABSTRACT

Cloud computing is a developing technology which gets more attention from both the industries and academia. Once the data is sent to the cloud, the cloud service provider (CSP) alone is responsible for the data. The fear of seeing the stored and used sensitive raw data is the major barrier in the adoption of cloud services. So data security and privacy are the major issues which act as an obstacle in the adoption of cloud computing. More techniques are available to ensure the protection of data confidentiality. But they do not completely serve the purpose of protecting data. To overcome these drawbacks, this paper introduces the Obfuscation (OB) based Modified Elliptical Curve Cryptography (MECC) algorithm for protecting data from malicious attacks, which is termed as OB-MECC. First, the proposed method obfuscates the data before they are uploaded into the cloud. The proposed technique uses different methods like substitution cipher, Ceaser cipher, position update, binary conversion, 8-bit binary conversion, two complex() decimal(), and ASCII() to obfuscate the original data. After that, the obfuscated data is encrypted using the MECC algorithm. After encryption, the data is retrieved from the cloud and decrypted by reversing the encryption and obfuscation process to get the original data.

Key words: Data confidentiality, Security, Obfuscation (OB), Encryption, Decryption, Elliptical Curve Cryptography (ECC), Modified ECC (MECC).

1. INTRODUCTION

Cloud computing is a process of computing that delivers services over a network, and the services may be hardware or software. It has a more powerful computing infrastructure with a pool of thousands of computers and servers [1]. In general, cloud computing is composed of three basic service models: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [2, 3, 4, 5, and 6]. For the organization, the cloud offers data centers to move their data globally. It eliminates the responsibility of local nodes for maintaining their data and also cloud supports customizable resources on the web. Cloud Service Providers maintains computing resources and data automatically via software [7].

With the widespread application of cloud computing, more and more sensitive information and private data are stored in the cloud by users [8]. Unless robust security scheme is implemented, the cloud system will be vulnerable to various attacks and susceptible by the users [9]. Thus, data security and privacy have become the chief concern, especially for business-level data. Data security mainly includes data confidentiality, availability, and integrity [10]. There are many CSPs who provide the security for data of the users. In the process of providing security for data, the CSPs develop a tendency to tamper or misuse the sensitive data without the prior knowledge of the users. So, the users are forced to hide the originality of their data before storing into cloud storage [11].

Traditional methods for data security usually rely on data encryption and access control. Novel methods should be developed and applied to fulfill all the mentioned requirements. The best approach is to encrypt data before outsourcing them to cloud computing [12]. Once the data is outsourced to the cloud, CSP is only responsible for maintaining, monitoring, and controlling the data. Confidentiality of data is to be ensured by cryptography, obfuscation, and steganography technique. Cryptography is an effective tool that helps to protect the data from unauthorized access while the data is at rest in cloud server [13] Cryptographic algorithms hide the data from unauthorized users. Examples of encryption algorithms are AES, DES, RSA, and Homomorphic, etc. Two operations performed by these algorithms are encryption and decryption. Symmetric algorithms use one key for encryption and decryption, while Asymmetric algorithms use two keys for encryption and decryption [14].

Relying on the standard cryptographic process is not satisfactory since stored data will get hacked. They are still worse when sensitive data are stored in remote places. Lot

more Confidentiality Techniques (CTs) are in existence to protect data when confidentiality is broken in CS, which in turn causes loss of data. Obfuscation techniques come to the rescue against unauthorized access while storing sensitive data in unknown places [15]. Obfuscation is a process applied to information to make it difficult to reverse without knowing the algorithm that was applied. Data obfuscation is used for security, which makes it hard to reconstruct the plaintext. This technique has recently become popular for data storage security in cloud storage [16].

The remaining sections of the proposed method is given below. Section 2, Surveys the associated work. Section 3, signifies a brief discussion about the proposed methodology. Section 4 analyses the experimental outputs. At last, section 5 deduces the paper.

2. RELATED WORK

Miranda Mowbray et al [17] suggested a privacy manager for cloud computing that controlled policy-based obfuscation and de-obfuscation of personal, sensitive, or confidential data within cloud service provision. By these means, cloud computing users might reduce the risk of their private data being stolen or misused, and also assistance might be given to cloud computing providers in helping them conform to privacy law. An algebraic description of obfuscation features were provided by the privacy manager, and also described how policies might be defined to control such obfuscation. Furthermore, the performance and scalability of this approach were charged and mechanisms to enhance usability were considered.

Nabeil et al [18] recommended an encryption method to facilitate the burden of working on encrypted data. An attribute-based online/offline searchable encryption scheme was introduced.

Neela and Kavitha[19] investigated a model which followed the decentralized architecture that does not depend on any third-party system. In this model, the data security could be enhanced by using a cyclic shift transposition algorithm. It ensured data confidentiality. Secondly, a quick response code, a technique for data forwarding was introduced. Authorized users only could get this QR code, and so it provided an effective authentication mechanism. Third, data integrity could be checked by using the hash-based timestamp. The suggested system encrypted over the encoded message and found the hash value for the encoded message so that the attacker cannot alter the content of the file.

Arul Oli and Arockiam[20] suggested an obfuscation technique to encrypt the desired data type on the cloud providing more protection from unknown hackers. This method produced high percentage in security level and has taken minimum time for obfuscation and deobfuscation process when compared with the existing techniques while encrypting and uploading data into public cloud storage. In this way, data confidentiality was protected in cloud storage

Zegzhda et al[21] investigated the problem of possible attacks on the confidentiality of user data in cloud systems that come from the cloud provider's side. A secure cloud computing system architecture based on Intel Software Guard Extensions technology has been introduced. Intel SGX technology consists of two parts: unprotected and protected (enclave). On startup, the unprotected part created an enclave in protected memory, transfers code, and data from unprotected memory to the enclave and initialized it by performing integrity checks. Only protected functions could see the data of the enclave. Any other external access (including the operating system) was prohibited, due to total data and code encryption. Enclaves, areas in the memory filled with encrypted pages of code and data. Enclaves were decrypted directly on the central processor when all checks were completed. This completely protected against the threat of information leakage in the open form to RAM or other I/O devices. A method for implementing this approach into existing cloud systems was presented.

K. R. Sajay et al [22] presented a hybrid algorithm to enhance the security of cloud data using encryption algorithm. The main purpose of using encryption algorithms was to secure or store huge amount of information in the cloud. This study combined the homographic encryption and blowfish encryption to enhance cloud security. The homographic encryption method performance was flexible by design and this performed the computations securely. Blowfish algorithm was used for developing the security and privacy issues in the cloud. It generated the key for security, and the symmetric key block was used for both decryption and encryption. The homographic encryption offered a dimension to storage in the cloud and also offered data confidentiality as in no stage information was exposed in plain text. It can be concluded that if the security issues are resolved, then in the future, it will be the solution for cloud storage in small as well as large firms.

Prabu Kanna and Vasudevan [23] developed a privacy preservation mechanism by implementing fully homomorphic–elliptic curve cryptography (FH-ECC) algorithm. The data owner encrypted the original data by converting it into the cipher format with the use of the ECC algorithm and applied the FH operations on the encrypted data before storing it in the cloud. When the user gave the data request to the cloud, the Cloud Service Provider verified the access control policy of the user for enabling the restricted access on the data. If the access policy was verified, the encrypted data was provided to the user, from that the ciphertext was extracted. Then, the ECC decryption and FH operations were applied to generate the original text. Based on the several analysis, the research work was evaluated with the help of different performance measures such as execution time, encryption time, and decryption time.

Nagaraju and Nagamalleswara rao[24] investigated problem of understand to enhance the security and privacy of cloud computing with a detailed learning of the obfuscation techniques which rescues the data from malicious attacks in an uncontrolled environment.

3. METHODOLOGY OF PROPOSED WORK

Obfuscation (OB) based encryption algorithm is proposed for data confidentiality. The proposed technique obfuscates and encrypts the data before they are uploaded into the cloud. Initially, the original data is obfuscated and then encrypted using the modified ECC (MECC) algorithm. So this combination is named as OB-MECC encryption algorithm. The data obfuscation is a technique which uses different mathematical methods or some critical programming logics to hide original data. The proposed technique uses different methods like substitution cipher, Ceaser cipher, position update, binary conversion, 8-bit binary conversion, two complex() decimal(), and ascii() to obfuscate the original data. After that, the obfuscated data is encrypted using the Modified ECC algorithm to obtain a higher level of data security. ECC algorithm is a type of mechanism that is adopted in the implementation of public-key cryptography. This technique is based on a curve with specific base points, which uses a prime number function. In the proposed system, three types of keys are generated. The first step is to generate a public key, private key, and secret key (public key + private key + point on curve). During encryption time, the secret key was added with encryption formula. After performing encryption, the encrypted data is uploaded to the cloud. To access the encrypted data, it should be decrypted. For data decryption, the modified ECC algorithm is reversed, i.e., the secret key is subtracted from the decryption formula. Afterward, the data is de-obfuscated by reversing the methods used in data obfuscation. Finally, the original data is obtained by using the de-obfuscation technique. The proposed method used for data confidentiality is shown in figure 1.

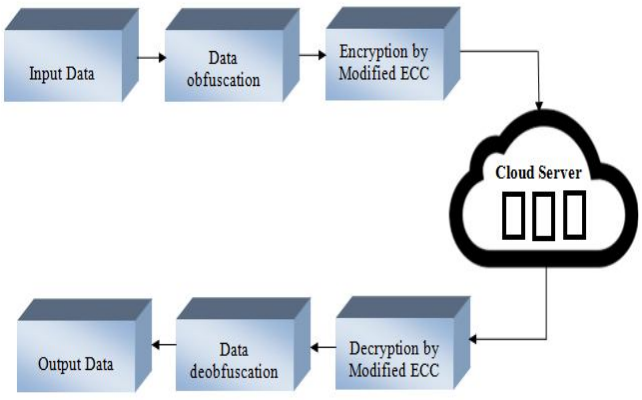


Figure 1: Proposed OB-MECC Architecture

3.1 Data obfuscation

The users' data are obfuscated and encrypted before it is uploaded to the cloud. The obfuscation is done in the user's machine connected to the cloud. The confidentiality of data stored in the cloud is ensured by the data obfuscation technique. The proposed obfuscation technique consists of several steps to obfuscate the original information. The steps used for obfuscating the data are as follows

i) Original input data is converted into substitution cipher text.
ii) Substitution cipher text is converted into caeser cipher text.
iii) The position of caeser cipher is marked using substitution cipher table.
iv) The position values are converted into binary numbers.
v) The binary numbers are converted into 8-bit binary numbers
vi) 2's complement is calculated for 8-bit binary numbers.
vii) Result of 2's complement is converted into decimal numbers.
viii) The decimal numbers are converted into ASCII codes.

The steps for proposed data obfuscation technique with an example are explained briefly as follows

3.1.1 Substitution Cipher

In cryptography, a substitution cipher is a method of encryption by which the units of plaintext are replaced with the cipher text. Here, "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing the inverse substitution.

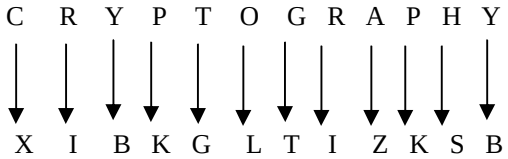
In the proposed method, a simple substitution cipher is used. Substitution of single letters is performed separately. Simple substitution can be demonstrated by writing out the alphabet in some order to represent the substitution. The cipher alphabet may be shifted or reversed or scrambled in a more complex fashion. The proposed method applies a cipher text on the original message based on the below table 1. The 26 alphabets (A-Z) are reversed (Z-A) to get a substation cipher.

Table 1: Substitution Cipher Table

Position	Alphabets	Substitution ciphers
1	A	Z
2	B	Y
3	C	X
4	D	W
5	E	V

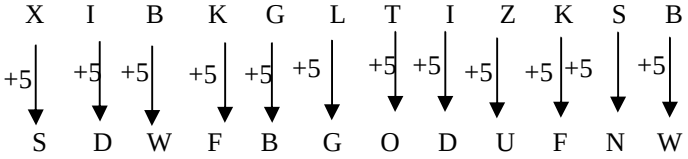
6	F	U
7	G	T
8	H	S
9	I	R
10	J	Q
11	K	P
12	L	O
13	M	N
14	N	M
15	O	L
16	P	K
17	Q	J
18	R	I
19	S	H
20	T	G
21	U	F
22	V	E
23	W	D
24	X	C
25	Y	B
26	z	A

For example, the data contained a message, i.e., CRYPTOGRAPHY. The substitution cipher of this data can be written as "XIBKGLTIZKSB" based on the substitution parameters used in table 1. For the letter C, the substitution parameter is X, and for letter R, the substitution parameter is I. Like this way, all the letters in the word "CRYPTOGRAPHY" are substituted. The substitution can be performed as follows,



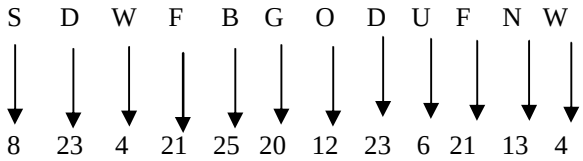
3.1.2 Caeser Cipher

In the second step, the result of substitution cipher is given to caesar cipher. The proposed obfuscation method uses the Caesar cipher as one of the steps for the encryption process. To pass an encrypted message from one person to another, first, it is necessary that both parties have the 'key' for the cipher, so that the sender may encrypt it and the receiver may decrypt it. For the caesar cipher, the key is the number of characters to shift the cipher alphabet. Here, the text to encrypt is the result of the previous step, i.e., XIBKGLTIZKSB with a shift (key) of 5. With the shift of 5 positions, the caesar cipher obtains the result as SDWFBGODUFNW. Each letter in the text is shifted five times to get a caeser cipher, which is based on table 1. For the substitution parameter X, the caeser cipher shifts '5' parameters from X, i.e., (X W, V, U, T, and S). S is the 5th position from X. So the obtained letter for X is S. Like this way, all letters in the data are shifted. The caeser cipher for XIBKGLTIZKSB is performed as follows,



3.1.3 Position Marking

In this step, the position of caesar cipher is marked using the substitution cipher table. For the previous step result, i.e., SDWFBGODUFNW, the position can be written as follows,



The letter S, D, and W are placed in 8th, 23rd, and 4th position in the substitution table. Similarly, the position of the remaining letters in the data can be written.

3.1.4 Binary Conversion

Here, the decimal number obtained from the previous step is converted into a binary number. The decimal (base ten) numeral system has ten possible values (0, 1, 2,3,4,5,6,7,8, or 9) for each place-value. In contrast, the binary (base two) numeral system has two possible values represented as 0 or 1 for each place-value. The steps to convert decimal number to binary number is as follows,

1. Divide the number by 2
2. Get the integer quotient for the next iteration
3. Get the remainder for the binary digit
4. Repeat the steps until the quotient is equal to 0

The result of previous step is 8 23 4 21 25 20 12 23 6 21 13 4. The binary conversion for 21 is performed using below table 2

Table 2: Decimal to Binary Conversion

Division by 2	Quotient	Remainder
21/2	10	1
10/2	5	0
5/2	2	1
2/2	1	0

So for 19, the binary number is 10011. Like this way, all decimal numbers are converted into a binary number.

3.1.5 8-bit binary conversion

The binary numbers from step 3 are now converted into an 8-bit binary number. The binary numbers are in a different format. They are written as an 8-bit binary number to place 0's in their left positions. For example, the binary number 10011 has only 5 bits, but for 8-bit binary number 3 bits are required. To obtain this, three 0's are placed in their Least Significant Bit (LSB). So it will be written as 000100011. Like this way, all binary numbers are converted into an 8-bit binary number. For example, from the above step binary number 10011 can be written as 00010011.

3.1.6 2's Complement

Here, 2's complement of 8-bit binary number is carried out. It is used in computing as a method of signed number representation. There is a simple algorithm to convert a binary number into 2's complement. To get 2's complement of a binary number, simply invert the given number and add 1 to the least significant bit (LSB) of given result. To get 1's complement of a binary number, simply invert the given number. For example, 1's complement of binary number 110010 is 001101. To get 2's complement of binary number is 1's complement of given number plus 1 to the least significant bit (LSB). For example 2's complement of binary number 10010 is (01101) + 1 = 01110.

3.1.7 Binary to Decimal Conversion

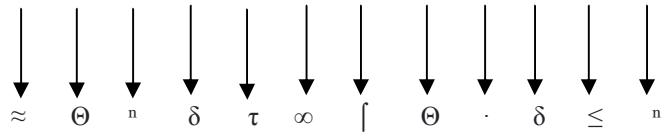
Binary to Decimal Conversion of numbers uses weighted columns to identify the order of the digits to determine the final value of the number. Conversion of binary to decimal (base-2 to base-10) numbers is an important concept to understand as the binary numbering system forms the basis for all computer and digital systems. The decimal or denary counting system uses the Base-of-10 numbering system where each digit in a number takes on one of ten possible values, called digits, from 0 to 9, e.g., 21310 (Two Hundred and Thirteen). The result of 2's complement is converted into a decimal number, which is represented as follows,

11110111 → 247
11101001 → 233
11111100 → 252
11101011 → 235
11100111 → 231
11101100 → 236
11110100 → 244
11101001 → 233
11111010 → 250
11101011 → 235
11110011 → 243
11111100 → 252

3.1.8 Decimal to ASCII Conversion

The obtained decimal numbers from 3.1.5 are converted into ASCII codes. ASCII codes represent text in computers, telecommunications equipment, and other devices. Most modern character-encoding schemes are based on ASCII, although they support many additional characters. It is based on the English alphabet, ASCII encodes 128 specified characters into seven-bit integers, in which Ninety-five of the encoded characters are printable: these include the digits 0 to 9, lowercase letters a to z, uppercase letters A to Z, and punctuation symbols. Also, the original ASCII specification included 33 non-printing control codes which originated with Teletype machines; most of these are now obsolete, although a few are still commonly used, such as the carriage return, line feed, and tab codes. The ASCII codes for the previous step is attained as follows,

247 233 252 235 231 236 244 233 250 235 243 252



After the ASCII conversion, the input text or data "cryptography" is obfuscated as "≈Θⁿδτ∞∫Θ·δ≤ⁿ" using the proposed obfuscation method. In this way, the input is obfuscated, and this obfuscated message is encrypted using the Modified ECC algorithm for enhancing the confidentiality and security of the data.

3.2 MODIFIED ECC ALGORITHM

ECC algorithm is a type of mechanism that is adopted in the implementation of public key cryptography. This technique is based on a curve with specific base points, which uses a prime number function. This function is used as a maximum limit. The mathematical representation of the ECC is shown in equation (1).

$$g^2 = x^3 + ax + b \quad (1)$$

Where a and b are the integers.

In a cryptographic process, the strength of the encryption technique depends purely on the mechanism that is employed for the generation of the key. In the proposed system, there are three types of keys that have to be generated. The first step is to generate the public key from the server and to encrypt the message. The second step is to generate a private key on the server-side and to decrypt the message. The third step is to generate the secret key from the public key, private key, and point on the curve.

Key generation is an important part where both public key and private key are generated. The sender encrypts the data with the receiver's public key, and the receiver decrypts the data using the private key.

Now, select a number 'd' within the range of 'n'.

Consider the following equation that can generate the public key.

$$Pb_K = Pr_K * H_i \quad (2)$$

Where, H_i is the point on the curve, Pb_K is the public key, and Pr_K is the private key. Consider the following equation that can generate the secret key.

$$Sc_K = Pb_k * Pr_K * H_i \quad (3)$$

After the key generation, the information's are encrypted. The encrypted information contains two ciphertexts that are mathematically represented as follows.

$$C_1 = \left(K * H_i\right) + Sc_K \quad (4)$$

$$C_2 = \left(M + \left(K * Pb_K\right)\right) + Sc_K \quad (5)$$

Here, the secret key Sc_K is added with two cipher texts. And in decryption time, this secret key is subtracted from the decryption equation to get an original message. This process is formulated by the following equation (6)

$$M = \left(\left(\left(C_2 - Pr_K\right) * C_1\right) - Sc_K\right) \quad (6)$$

Where M denotes the original message, and K is the random number generated in the range of 1 and $n-1$. In the decryption process, the secret key is subtracted from the normal decryption equation which is called as modified ECC. This encrypted information with the change source IP address is sent to the destination user. The psedocode for the proposed modified ECC algorithm is illustrated in below figure 2

Input: Encrypted Obfuscated data

Output: Encrypted data

Select a number'd' within the range of 'n'

//Key Generation

Generate public key using

$$Pb_K = Pr_K * H_i$$

Generate secret key using

$$Sc_K = Pb_k * Pr_K * H_i$$

// Data Encryption

Add secret key using

$$C_1 = \left(K * H_i\right) + Sc_K$$

$$C_2 = \left(M + \left(K * Pb_K\right)\right) + Sc_K$$

// Data Decryption

Subtract secret key using

$$M = \left(\left(\left(C_2 - Pr_K\right) * C_1\right) - Sc_K\right)$$

Figure 2: Pseudocode for Modified ECC algorithm

3.3 DATA DECRYPTION

The decryption process is a reverse process of encryption. After obfuscation and encryption, the data is retrieved from the cloud and the data is decrypted using the modified ECC algorithm. And then the data is de-obfuscated by performing the obfuscation methodologies in reverse. That means it performs ascii(), decimal(), two complex(), binary conversion, position marking, caeser cipher, substitution cipher. After performing de-obfuscation, the original data is retrieved.

4. RESULT AND DISCUSSION

The proposed methods used in data obfuscation and encryption are analyzed.

4.1 Encryption and Decryption

The data submitted to the cloud is obfuscated and encrypted using OB-MECC algorithm. The plaintext is given as an input. The name of the file is "Original File.txt", which is shown in below figure 3.

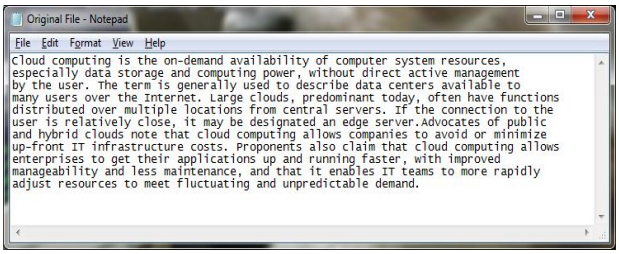


Figure 3: Input Text File

Figure 3 shows the plaintext file for input. This file is obfuscated and encrypted using OB-MECC and the encrypted text is stored in another file called Encrypted File.Txt. The output of the proposed encryption method is shown in figure 4.

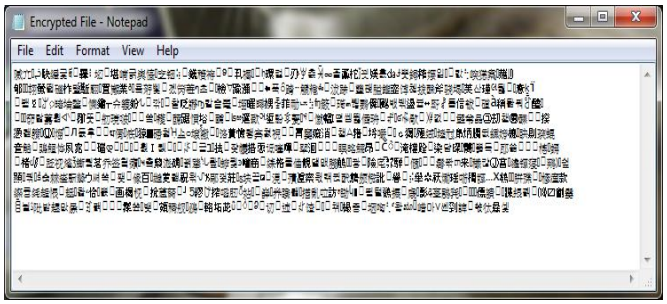


Figure 4: Encrypted Text File

The encrypted file is uploaded to the cloud. To access the obfuscated data, it should be de-obfuscated. The de-obfuscated data stored in a file is called Decrypted File.txt. The output of the proposed decryption method is shown in figure 5.

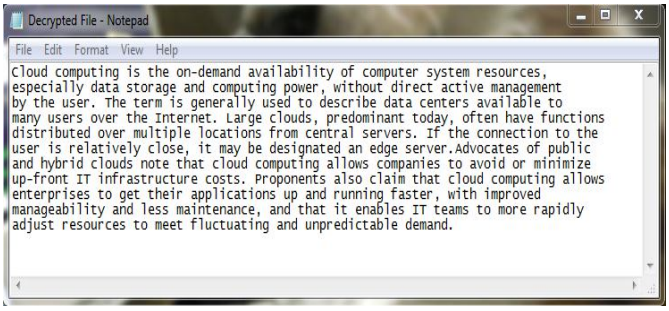


Figure 5: Decrypted Text File

4.2 Performance Analysis

Here, the performance analysis of proposed data obfuscation and proposed data encryption techniques are compared with the existing techniques. And the security level of proposed and existing methods is compared.

4.2.1 Performance Comparison of Data Obfuscation

The proposed data Obfuscation technique is compared with the existing techniques such as hexadecimal encoding, Huffman encoding, and Base 64 in terms of obfuscation time and de-obfuscation time, which is shown in below table 3. The time taken by the existing and proposed obfuscation techniques is calculated for different sized plaintexts. The result shows that compared to the existing obfuscation techniques, the proposed technique has taken low time for obfuscating different size of plaintext.

Table 3: Performance Comparison of proposed Obfuscation with existing techniques in terms of (a) obfuscation time and (b) de-obfuscation technique

(a)

Size (Kb)	Proposed Obfuscation	Hexadecimal encoding	Huffman encoding	Base 64
1000	1251	1658	1504	2045
2000	2102	2189	2085	2514
3000	3148	3518	3218	3478
4000	4211	4897	4658	4875
5000	5291	6214	5814	6874

(b)

Size (Kb)	Proposed de-obfuscation	Hexadecimal encoding	Huffman encoding	Base 64
1000	1156	1658	1587	2048
2000	2113	2354	2025	2448
3000	3186	3478	3228	3525
4000	4222	4754	4521	4889
5000	5287	6021	5714	6868

In table 3 (a), for 1000 kb file, the Base 64 method takes more time (2045 ms) for data obfuscation, whereas hexadecimal encoding and Huffman encoding takes 1658 ms and 1504 ms for the same process. But the proposed obfuscation takes 1251 ms for data obfuscation. It is comparatively low when compared with others. The obfuscation time increases gradually when the file size increases, but the proposed one takes the lowest time for all size of files contrasted to others. And also the time taken for de-obfuscation is illustrated in table 3 (b). Here, the time taken for de-obfuscation is also low for the proposed method contrasted to others. The Base 64 method takes more time for every file types, and the proposed one takes very less time compared to others. The graphs for table 3 is shown in figure 6 and 7.

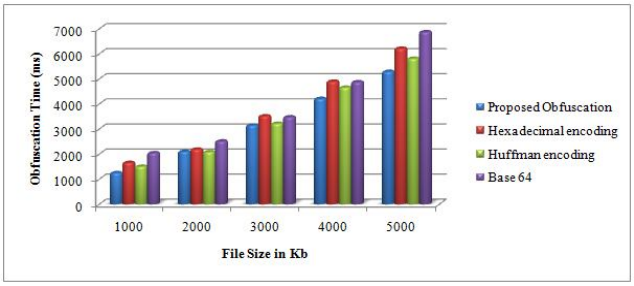


Figure 6: Performance comparison of obfuscation techniques by time

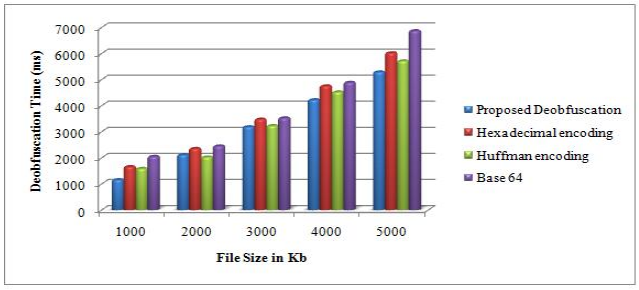
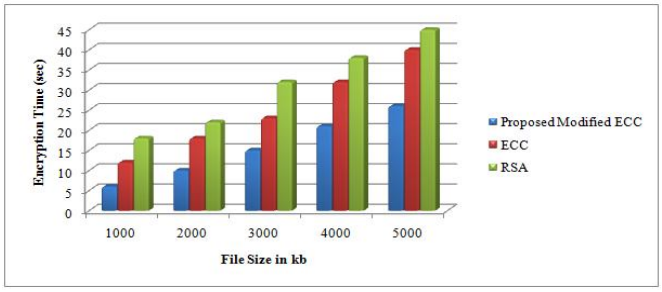


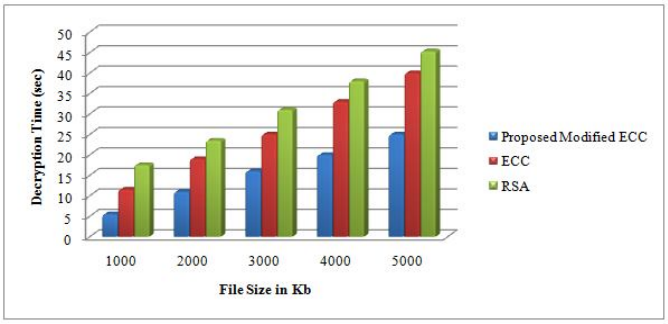
Figure 7: Performance comparison of de-obfuscation techniques by time

4.2.2 Performance Comparison of Encryption and Decryption

The proposed modified ECC algorithm is compared with the existing techniques such as ECC and Rivest-Shamir-Adleman (RSA) in terms of encryption time and decryption time. The comparative examination of proposed and existing methods is exhibited in figure 8. The result shows that compared to the existing encryption algorithms, the modified ECC technique has taken low time for encrypting and decrypting the obfuscated data.



(a)



(b)

Figure 8: Comparative analysis of proposed Modified ECC with existing techniques in terms of (a) encryption time, (b) decryption time

The above figure 8 delineates the performance analysis of proposed and existing techniques. In figure (a), the encryption execution time of the proposed one is weighted against existing ECC and RSA for different file sizes. For 1000 kb file, the proposed one takes 6 seconds for encrypting the data, but the existing methods such as ECC and RSA takes 12 and 18 seconds for their execution. The encryption time increases gradually when the file size increases, but the proposed technique consumes lesser time for every execution contrasted with other existing methods. And also in figure 8 (b), the decryption time is plotted for the different file sizes. The figure (b), clearly displays that the proposed one gives less decryption time for every file size when compared with the existing one.

4.2.2 Performance Comparison of Security Level

The security level of proposed and existing techniques is shown in below table 4. The security level for proposed OB-MECC with existing techniques such as Elliptical Curve Cryptography (ECC), Hexadecimal Encoding, and Base 64 are compared. The result shows that compared to the existing techniques, the proposed technique has the maximum percentage of security.

Table 4: security level of proposed OB-MECC and existing techniques

S.No	Techniques	Security Level
1	Proposes OB-MECC	94
2	ECC	81
3	Hexadecimal Encoding	62
4	Base 64	72

The graphical illustration of table 3 is shown in below figure 6

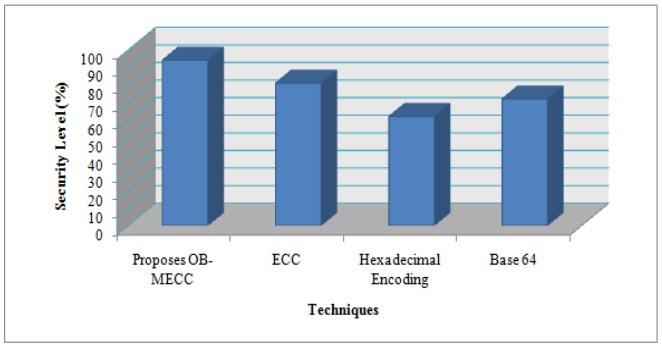


Figure 9: Security Level of proposed and existing methods

The above figure represents the graphical performance comparison of security level for proposed OB-MECC with existing techniques such as ECC, Hexadecimal Encoding, and Base 64. The security level achieved for ECC, hexadecimal encoding, and Base 64 is 81%, 62%, and 72% whereas the proposed OB-MECC technique achieves 94% of security level. So the security level of the proposed technique is higher compared to the other three techniques.

5. CONCLUSION

In this paper, a new OB-MECC technique is proposed for data confidentiality. The data is first obfuscated and then encrypted before it is uploaded into the cloud. After the data is retrieved from the cloud, it is decrypted by using modified ECC and de-obfuscation. This method produced a high-security level (94%) when compared to the existing say ECC, hexadecimal encoding, and Base 64. And, the proposed modified ECC algorithm takes a minimum time of encryption and decryption for different file sizes when compared with the existing techniques while encrypting and uploading data into public cloud storage. In this way, data confidentiality is provided in cloud storage. Thus, security is enhanced through this proposed OB-MECC method. In the future, the different encryption algorithms will be implemented to improve the performance of these algorithms and to enable CSP to provide a secure data storage environment for their clients.

REFERENCES

1. George Amalarathinam DI, "**Security Enhancement for Public Cloud Storage**", International Journal of Pure and Applied Mathematics, vol. 118, no. 6, pp. 1-9, 2018.
2. Khalid EL Makkaoui, Abdellah Ezzati, Abderrahim Beni-Hssane, Cina Motamed, "**Data confidentiality in the world of cloud**", Journal of Theoretical and Applied Information Technology, vol. 84, no. 3, 2016.
3. Vithya Vijayalakshmi A, Veeraragavan N and Arockiam L, "**A unified model for cloud data confidentiality**", Asian Journal of Science and Applied Technology, vol. 7, no. 1, pp.23-27, 2018.

4. Kelsey Rauber, "**Cloud cryptography**", International Journal of Pure and Applied Mathematics, Vol. 85, pp. 1-11, 2013
5. Silki Jain, Abhilasha Vyas, "**An Improved Security Framework for Cloud Environment using ECC algorithm**", International Journal for Research in Applied Science & Engineering Technology (IJRASET), vol. 6, 2018.
6. Sulochana M and Ojaswani Dubey, "**Preserving data confidentiality using multi-cloud architecture**", Procedia Computer Science, vol. 50, pp. 357-362, 2015.
7. Manpreet Kaur and Rajbir Singh, "**Implementing encryption algorithms to enhance data security of cloud in cloud computing**", International Journal of Computer Applications, vol. 70, no. 18, 2013.
8. Yang Geng, "**Homomorphic encryption technology for cloud computing**", Procedia Computer Science, vol. 154, pp. 73-83, 2019.
9. Arockiam D L and Monikandan S, "**A security service algorithm to ensure the confidentiality of data in cloud storage**", Int J Eng Res Technol (IJERT), vol. 3, no. 12, pp. 1053-1058, 2014.
10. Yue Shi, "**Data security and privacy protection in public cloud**", In IEEE International Conference on Big Data (Big Data), pp. 4812-4819, IEEE, 2018.
11. Arul Oli S, Arockiam L, "**Confidentiality technique for enhancing data security using encryption and obfuscation in public cloud storage**", International Journal of Advanced Research in Computer and Communication Engineering, vol. 5, no. 2, 2016.
12. Ghassan Sabeeh Mahmood, Dong Jun Huang, and Baidaa Abdulrahman Jaleel, "**Achieving an effective, confidentiality and integrity of data in cloud computing**", IJ Network Security, vol. 21, no. 2, pp. 326-332, 2019.
13. Arockiam L, Monikandan S, Sheba K Malarchelvi P D, "**Obfuscrypt: A novel confidentiality technique for cloud storage**", International Journal of Computer Applications, vol. 88, pp. 17-21, 2014.
14. Nasarul Islam K V, Mohamed Riyas K V, "**Analysis of various encryption algorithms in cloud computing**", International Journal of Computer Science and Mobile Computing, IJCSMC, vol. 6, no. 7, pp. 90 – 97, 2017.
15. Arul Oli S, and Arockiam L, "**Confidentiality technique to obfuscate the numerical data to enhance security in public cloud storage**", In International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET), pp. 1-6, IEEE, 2017.
16. Fathima Mary B, and DI George Amalarethinam, "**Data security enhancement in public cloud storage using data obfuscation and steganography**", In World Congress on Computing and Communication Technologies (WCCCT), pp. 181-184, IEEE, 2017.
17. Miranda Mowbray, Siani Pearson, and Yun Shen, "**Enhancing privacy in cloud computing via policy-based obfuscation**", The Journal of Supercomputing, vol. 61, no. 2, pp: 267-291, 2012.
18. Nabeil Eltayieb, Rashad Elhabob, Alzubair Hassan, and Fagen Li, "**An efficient attribute-based online/offline searchable encryption and its application in cloud-based reliable smart grid**", Journal of Systems Architecture, vol. 98, pp. 165-172, 2019.
19. Neela K L, and Kavitha V, "**Enhancement of data confidentiality and secure data transaction in cloud storage environment**", Cluster Computing, vol. 21, no. 1, pp. 115-124, 2018.
20. Arul Oli S, Arockiam L, "**Enhanced obfuscation technique for data confidentiality in public cloud storage**", In MATEC Web of Conferences, vol. 40, 2016.
21. Dmitry Zegzhda P, Usov E S, Nikol'skii A V, and Yu Pavlenko E, "**Use of intel SGX to ensure the confidentiality of data of cloud users**", Automatic Control and Computer Sciences, vol. 51, no. 8, pp. 848-854, 2017.
22. Sajay K R, Suvanam Sasidhar Babu, and Yellepeddi Vijayalakshmi, "**Enhancing the security of cloud data using hybrid encryption algorithm**", Journal of Ambient Intelligence and Humanized Computing, pp. 1-10, 2019.
23. Prabu Kanna G, Vasudevan V, "**A fully homomorphic–elliptic curve cryptography based encryption algorithm for ensuring the privacy preservation of the cloud data**", Cluster Computing, pp. 1-9, 2018.
24. Nagaraju P, Naga Malleswara Rao N, "**Obfuscation Techniques in Cloud Computing: A Systematic Survey**", International Journal of Scientific & Technology Research, pp. 1097-1102, 2019.