OB-MECC: An Efficient Confidentiality and Security Enhancement for Cloud Storage System

P. NagaRaju[1,*] and 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*
E-mail: nagaraj528@gmail.com
[*]*Corresponding Author*

Received 18 August 2020; Accepted 22 November 2020;
Publication 06 February 2021

Abstract

Cloud computing (CC) is one amongst the developing technologies, which gets more attention from academia as well as industries. It offers diverse benefits like sharing computing resources, service flexibility, reducing costs, etc. The Cloud Services Provider (CSP) is accountable for the data that are delivered to the cloud. The threat of seeing the stored data and using sensitive raw data by strangers is the main barrier in the utilization of cloud services. So, Data Security (DS) along with privacy is the chief issue, which is an obstacle while adopting the CC. Countless techniques are existent for ensuring data confidentiality, but they do not completely give protection to the data. To trounce these drawbacks, this paper introduces the Obfuscation (OB) based Modified Elliptical Curve Cryptography (MECC) algorithm for protecting data as of malicious attacks, which is termed as OB-MECC. Primarily, the proposed method obfuscates the data before they are uploaded to the cloud. For the OB of the data, the proposed work employs methods like substitution cipher (SC), position update, Ceaser cipher, binary conversion, 8-bit binary conversion, decimal(), two complex(), and ASCII(). Then, encryption of the obfuscated data is done with the utilization of the MECC algorithm. After encryption, the data on the cloud is retrieved. The retrieved data is then decrypted by reversing the OB and encryption process to get the actual data. The outcomes corroborate that the confidentiality and security level are maximum for the proposed OB-MECC when contrasted to the existing approaches.

Keywords: Data confidentiality, data Security, obfuscation (OB), data encryption, data decryption, elliptical curve cryptography (ECC), modified ECC (MECC).

1 Introduction

CC is a process of computing, and it delivers services (software or hardware) over a network. The CC's computing infrastructure is a very powerful one encompassing a pool of chiliads of computers and servers [1] and has '3' basic service models: (i) Software as a Services (SaaS), (ii) Platforms as a Services (PaaS) and iii) Infrastructures as a Services (IaaS) [2–6]. The cloud offers data centers to move their data worldwide, eradicates the responsibility of local nodes for maintaining their data, and also supports customizable resources on the web. CSP utilizes software to maintain computing resources along with data automatically [7].

Due to the CC's extensive application, more and more sensitive information together with private data is being stored on the cloud by users [8]. The cloud system will be susceptible to disparate attacks and doubted by the users unless a robust security scheme is implemented [9]. Therefore, DS and privacy are the main concern, particularly for business-level data. DS mainly includes data confidentiality, availability, together with integrity [10]. Numerous CSP is available who can provide security for the users' data. But these CSP develop a tendency to misuse or tamper the sensitive data devoid of the prior knowledge about the users while providing security for data. Thus, the users are forced into a place where they should conceal the originality of their data before storing it into cloud storage [11].

Data encryption together with access control is the '2' most utilized traditional methods for DS. To accomplish all the aforesaid requirements, novel methods ought to be developed. Encrypting data ahead of outsourcing it to CC will be the most excellent approach to handle this issue [12]. CSP is simply accountable for maintaining, monitoring, along with controlling the data once it is outsourced to the cloud. Cryptography, OB, and also the steganography technique can be employed to ensure the confidentiality of data. Cryptography is an effective tool in protecting the data by hiding it from illegal access when the data is in the cloud server [13]. AES, DES, RSA, and Homomorphic, etc are some examples of encryption algorithms. These algorithms perform encryption along with decryption. Encryption converts the data to a scrambled format; in reverse, Decryption converts it back to human-comprehendible form. In Symmetric algorithms, one key is generated for encryption in addition to decryption; while in Asymmetric algorithms, two separate keys are employed for encryption and decryption [14].

Relying only on the standard cryptographic process will not be satisfactory since there is a chance of being hacked. And it will be worse if the data (sensitive ones) are stored in remote places. There are abundant of Confidentiality Techniques (CT) present out there, which protect data when confidentiality is broken in CS, but in turn, it causes loss of data. OB techniques come to the rescue against illegal access while amassing data (sensitive) in unknown places [15]. It is a procedure implemented to information to make it hard to undo without having the knowledge about the algorithm that was used. Data OB is utilized for security, which makes it tough to restructure the plaintext. This technique has currently become popular for data storage security in cloud storage [16].

The remaining sections of the proposed work are Section 2 reviews 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] recommended a privacy manager (PM) for CC that controlled policy-centric OB and de-obfuscation (D-OB) of personal, confidential, or sensitive data. CC users might lessen the risk of their private data being misused or stolen, and also assist the CC providers conforming to privacy law. Diverse feasible architectures were delineated for such privacy management, an algebraic delineation of OB features were rendered by the PM, and also elucidated how policies were defined for controlling such OB. Moreover, the performance along with scalability of this approach was charged and mechanisms for ameliorating usability were concerned. Multiple instances of how to utilize the PM were given, encompassing the protection of confidential data (share portfolios) and private metadata (online photos).

Nabeil et al. [18] propounded an encryption approach for trouncing the working burden on the encrypted data. An attribute-centric offline/online searchable encryption framework was introduced with the succeeding contributions: Initially, trapdoor and encryption algorithms were separated into 2 phases. Secondly, the attribute control policy together with message encryption was done offline. Thirdly, the propounded structure was secured against both plaintext and keyword attacks. At last, the applicability of the propounded scheme was expounded in a cloud-centric smart grid.

Neela and Kavitha [19] examined a framework that followed the decentralized structure that does not rely on any 3rd-party system. In this framework, the security was ameliorated by utilizing an algorithm termed cyclic shift transposition and data confidentiality was ensured. Secondly, an approach for data forwarding, that is, quick responses (QRs) code was introduced. For effectual authentication, it provided QR code only to authorized users only. Thirdly, data integrity was checked by utilizing the hash-centric timestamp. This approach was a decentralized structure such that any 3rd party could not access this system. The recommended system encrypted the previously encoded message and determined the hash value for that encoded message in order that the attacker could not alter the file content. The recommended system does not pay attention on certain attacks like malicious insiders.

Arul Oli and Arockiam [20] recommended an OB approach for encrypting the required data type on the cloud by providing the utmost protection as of unknown hackers. Whilst encrypting and uploading data to public cloud storage, this approach produced a higher security level and has consumed minimal time for OB and D-OB processes on contrasting to the prevailing approaches. Data confidentiality was as well facilitated. The outcomes corroborated that the time consumed for OB was lower and the confidentiality percentage was higher with prevailing approaches.

Zegzhda et al. [21] examined the issue of feasible attacks on the data confidentiality in CC systems that come as of the CSP. A secured CC architecture grounded on Intel SGX (Software Guard Extensions) which encompassed "2" parts: protected (enclave) and unprotected was introduced. On startup, the unprotected part generated an enclave in the memory (protected), and transferred data along with code as of unprotected memory to the enclave and initialized it by executing integrity checks. Merely protected operations could see the enclave data. Other access (even the OS) was prohibited, on account of data along with code encryption. Enclaves indicate the regions in the memory filled with encrypted code together with data. Enclaves were then directly decrypted in the CPU when the entire checks were accomplished. This was entirely shielded against the information leakage in an open form to I/O devices or RAM. This approach and even its implementation for facilitating the confidentiality of users' data were proffered.

Sajay et al. [22] proffered a hybridized algorithm for ameliorating the security of cloud data utilizing encryption algorithms. The encryption algorithms were utilized for securing or storing copious information. This study integrated the blowfish and homographic encryption for ameliorating cloud security. The homographic encryption's performance was flexible by design and it securely executed the computations. Blowfish algorithm was utilized for building privacy along with security issues. It created the keys for security, and the symmetric key block was employed for decryption together with encryption. The homographic one proffered a dimension for cloud storage and also proffered data confidentiality as no stage information was disclosed in plain text. It was inferred that if the existing security issues were tackled, then in the future, it will serve as the solution for cloud storage in small or large firms.

Prabu Kanna and Vasudevan [23] put forward a privacy preservation strategy by employing a fully homomorphic (FH)–elliptic curve cryptography (ECC) algorithm. The user (data owner) encrypted the actual data by transmuting it to the cipher utilizing ECC and employed the FH functions on those encrypted data prior to storing it in the CC environs. When the user sent the data request, the CSP validated the access policy of the intended user for facilitating the restricted data access. After validating that access policy, those encrypted data were delivered to the user, and from that, the ciphertext (CT) was extracted. Then, the above-said FH functions and ECC decryption were employed for generating the actual text. Grounded on the analyses, the suggested approach was assessed with the aid of disparate performance metrics namely encryption time (ET), execution time, and decryption time (DT).

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 Proposed Methodology

This work proposes the Obfuscation (OB) centric encryption algorithm for data confidentiality. Primarily, obfuscation and encryption of the original data

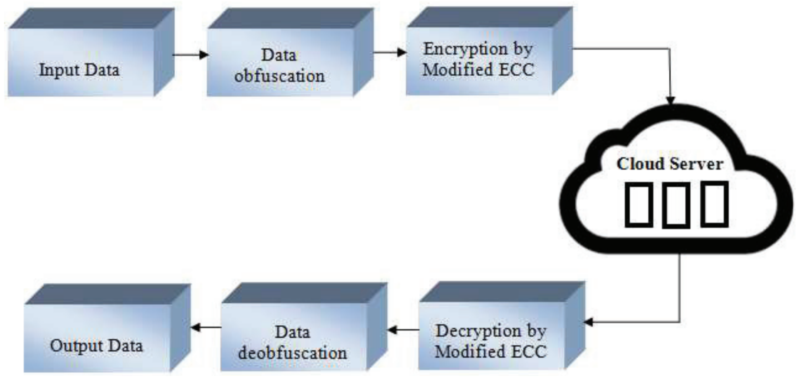


Figure 1 Proposed OB-MECC architecture.

are done prior to uploading them into the cloud utilizing the modified ECC (MECC) algorithm. This combination is termed as the OB-MECC encryption algorithm. The data OB approach utilizes disparate mathematical techniques or certain critical programming logic for concealing the original/actual data. The proposed work utilizes methods like SC, Ceaser cipher, binary conversion, position update, 8-bit binary conversion, decimal(), two complex(), and ascii() for obfuscating the actual data. Afterward, the obfuscated data is encrypted with the aid of the Modified ECC algorithm for attaining a higher level of data security. ECC algorithm is a sort of mechanism that is employed during the implementation of public-key cryptography (PKC). This technique is grounded on a curve with specific base points, which utilizes a prime number function. In this proposed system, three sorts of keys like a private key, public key, along with secret key (public key + point on curve + private key) are generated. During encryption, the secret key "Sc_K" was added with encryption formula and then the encrypted data gets uploaded into the cloud. To access this encrypted data, it should be decrypted. For data decryption, the MECC algorithm is reversed, i.e., Sc_K is subtracted as of the decryption formula. Afterward, the data is de-obfuscated by reversing the methods used in data OB. Finally, the actual data is attained by utilizing the D-OB technique. Figure 1 delineates this proposed work in detail

3.1 Data Obfuscation

The user's data is obfuscated and encrypted before uploading them into the cloud. OB is done in the user's machine connected to the cloud. The data confidentiality is ensured in the cloud by the technique termed data OB. The proposed OB technique encompasses several steps for obfuscating the original information. The steps used for obfuscating the data are

(i) Original input data is converted into SC text
(ii) SC text is then converted to caesar CT
(iii) The position of caesar cipher is marked using SC table
(iv) The position values are converted into binary numbers
(v) The binary numbers are converted into 8-bit binary numbers
(vi) 8-bit binary numbers' 2's complements are calculated
(vii) Result of 2's complement is converted into decimal numbers
(viii) The attained decimal numbers are finally converted to ASCII codes

The steps for the proposed data OB technique with an example are briefly explained below.

3.1.1 Substitution Cipher

It is an encryption approach in cryptography through which the units of plain text are substituted with the CT. Here, the "units" indicate single (common), pairs and triplets of letters, and their combinations. The receiver endeavors to decipher the provided text by inversely executing the substitution. In the proposed work, a simple SC is utilized. The substitution of every letter is performed separately. To demonstrate it, write the alphabets in a certain order to signify the substitution and this is termed as a substitution alphabet. Shift or reverse or scramble the cipher alphabet in a complex manner and this process is termed as a deranged or mixed alphabet. The proposed method employs a CT on the actual message grounded on Table 1. The 26 alphabets (A–Z) are reversed (Z–A) to attain an SC.

For example, if the data has a message, "CRYPTOGRAPHY", the SC of this message is written as "XIBKGLTIZKSB" grounded on the substitution parameters (table 1). For the letter C, the substitution parameter is X, and for R, it is I. Likewise, the entire letters in "CRYPTOGRAPHY" are substituted as,

$$\begin{tikzcd}
C \arrow[d] & R \arrow[d] & Y \arrow[d] & P \arrow[d] & T \arrow[d] & O \arrow[d] & G \arrow[d] & R \arrow[d] & A \arrow[d] & P \arrow[d] & H \arrow[d] & Y \arrow[d] \\
X & I & B & K & G & L & T & I & Z & K & S & B
\end{tikzcd}$$

Table 1 SC 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

3.1.2 Caeser Cipher

It takes the result of SC as input. It is one of the eminent simple cipher and is a sort of SC that shifts every letter in plaintext to a specified number of positions down the alphabet. For instance, by shifting 1 place, A would be replaced by B, B by C, and so on. The proposed OB approach utilizes this Caesar cipher as a step for encryption. For passing the encrypted message as of one to another person, parties should know the 'key' for a cipher, thereby the sender could encrypt it and the intended receiver decrypts it. This key indicates the number of characters for shifting the cipher alphabet. Here, the outcome of the previous step "XIBKGLTIZKSB" is then encrypted with key=5 (5 shifts). After shifting 5 positions, the caesar cipher attains the result "SDWFBGODUFNW". Every letter in the given text is shifted five times to attain a caesar cipher, which is grounded on table 1. For the substitution parameter "X", this caesar cipher shifts '5' parameters as of X, that is, W, V, U, T, and S. Here, S is the 5th position as of X and hence, the attained letter for X is "S". Likewise, shift all letters in the provided data. The "XIBKGLTIZKSB" forms the caesar cipher of,

$$\begin{tikzcd}
X \arrow[d, "+5"'] & I \arrow[d, "+5"'] & B \arrow[d, "+5"'] & K \arrow[d, "+5"'] & G \arrow[d, "+5"'] & L \arrow[d, "+5"'] & T \arrow[d, "+5"'] & I \arrow[d, "+5"'] & Z \arrow[d, "+5"'] & K \arrow[d, "+5"'] & S \arrow[d, "+5"'] & B \arrow[d, "+5"'] \\
S & D & W & F & B & G & O & D & U & F & N & W
\end{tikzcd}$$

3.1.3 Position Marking

Here, the caesar cipher's position is marked with the utilization of the SC table. For the former step outcome, i.e., SDWFBGODUFNW, the position is written as,

$$\begin{tikzcd}
S \arrow[d] & D \arrow[d] & W \arrow[d] & F \arrow[d] & B \arrow[d] & G \arrow[d] & O \arrow[d] & D \arrow[d] & U \arrow[d] & F \arrow[d] & N \arrow[d] & W \arrow[d] \\
8 & 23 & 4 & 21 & 25 & 20 & 12 & 23 & 6 & 21 & 13 & 4
\end{tikzcd}$$

The letter W, S, and D are placed in the 4th, 8th, and 23rd positions in the substitution table and likewise, the remaining letters' place in the data is written.

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

3.1.4 Binary Conversion

Here, the decimal numbers attained from the previous step are converted to their corresponding binary numbers. There are 10 possible values (0, 1, 2,. . . , 9) for every place-value in the decimal numeral system. Contrarily, the possible values of binary number system are 0 or 1for every place-value. The steps to convert decimal into binary is as follows:

1. Divide the given number with 2
2. Attain an integer quotient for every subsequent iteration
3. Note the remainder for the binary digit
4. Repeatedly execute the steps till the quotient is equivalent to 0

The outcome of the previous step is 8 23 4 21 25 20 12 23 6 21 13 4. Table 2 performs the binary conversion for 21.

Hence, the binary value for 19 will be 10011. And in this way, the other decimal numbers are transmuted to their corresponding binary values. The final outcome of this step is written as

$$\begin{tikzcd}
8 \arrow[d] & 23 \arrow[d] & 4 \arrow[d] & 21 \arrow[d] & 25 \arrow[d] & 20 \arrow[d] & 12 \arrow[d] & 23 \arrow[d] & 6 \arrow[d] & 21 \arrow[d] & 13 \arrow[d] & 4 \arrow[d] \\
1000 & 10111 & 100 & 10101 & 11001 & 10100 & 1100 & 10111 & 110 & 10101 & 1101 & 100
\end{tikzcd}$$

3.1.5 8-bit binary conversion

The attained binary numbers are now converted to an 8bit binary number by placing 0's in their left positions. For instance, "10011" has only 5bits, to convert it into 8bit, additional 3 bits are needed. For this, place three 0's in their Least Significant Bit (LSB), that is, on the left side and hence, 10011 becomes 00010011. In this way, the entire binary numbers from step 3 are converted to 8-bit as,

Binary Number		**8-bit Binary Number**
1000	→	00001000
10111	→	00010111
100	→	00000100
10101	→	00010101
11001	→	00011001
10100	→	00010100
1100	→	00001100
10111	→	00010111
110	→	00000110
10101	→	00010101
1101	→	00001101
100	→	00000100

3.1.6 2's Complement

Here, the detailed description is provided for the determination of 2's complement of an 8 bit binary number. 2's complement, an instance of the radix complement, implies a mathematic operation done mainly on the binary numbers. It is utilized by computers as an approach of signed number representations and is generally a fixed point binary value. The Binary number could be complemented in two ways like,1's complement along with 2's complement. In 1's complement, simply invert a given binary number, for example, the 1's complement of 010010 is 101101. In 2's complement first invert a provided binary number and subsequently add 1 to its LSB, for instance, for 10010, its 2's complement is 01110 $(01101 + 1 = 01110)$.

3.1.7 Binary to Decimal Conversion

Decimal numbers are converted to Binary (base-2) with the use of weighted columns to find the order of the digits. The digits' order is identified for determining a number's final value. Conversion of binary to decimal (base-10) numbers is an imperative concept to comprehend as the binary number system is the basic one for all digital systems. In the Base-of-10 or decimal numbering system, all digits in the number take one of 10 possible values, termed digits, as of 0 to 9, example- 21410 (Two Hundred and Fourteen).

Here, the outcome of every 2’s complement is converted to a decimal and is signified as,

$$\begin{array}{lcl}
11110111 & \longrightarrow & 247 \\
11101001 & \longrightarrow & 233 \\
11111100 & \longrightarrow & 252 \\
11101011 & \longrightarrow & 235 \\
11100111 & \longrightarrow & 231 \\
11101100 & \longrightarrow & 236 \\
11110100 & \longrightarrow & 244 \\
11101001 & \longrightarrow & 233 \\
11111010 & \longrightarrow & 250 \\
11101011 & \longrightarrow & 235 \\
11110011 & \longrightarrow & 243 \\
11111100 & \longrightarrow & 252
\end{array}$$

3.1.8 Decimal to ASCII Conversion

The attained decimal numbers from the above section are converted to ASCII codes. ASCII implies the “American Standard Code for Information Interchange” and is utilized as a standard for character encoding in electronic communication. ASCII codes signify text in telecommunication gadgets, computers, etc. Although the latest character-encoding frameworks support several additional characters, they are centered on ASCII. ASCII is grounded on the English alphabets. It could encode 128 characters to 7bit integers wherein 95 of these encoded characters are highly printable. The characters embrace the digits “0 to 9”, punctuation symbols, lowercase letters “a to z”, together with uppercase letters “A to Z”. Also, the actual ASCII specification encompassed “33” non-printing control codes that are originated by Teletype machines. Most of them are obsolete, even though some are now in use, namely the tab codes, line feed together with a carriage return. The former step has the subsequent ASCII codes.

$$\begin{array}{cccccccccccc}
247 & 233 & 252 & 235 & 231 & 236 & 244 & 233 & 250 & 235 & 243 & 252 \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
\approx & \Theta & ^{n} & \delta & \tau & \infty & \lceil & \Theta & \cdot & \delta & \leq & ^{n}
\end{array}$$

After performing the ASCII conversion, the input data or text "cryptography" is obfuscated as "$\approx$ Θn$\delta\tau\infty \int \Theta\delta\cdot$ $\leq$n" utilizing the proposed OB approach. The obfuscated message is now encrypted utilizing the MECC algorithm for elevating the confidentiality and security of the data.

3.2 Modified ECC Algorithm

ECC is adopted while implementing PKC. This algorithm is grounded on a curve with certain base points, which utilizes a prime number function. This function is concerned as a maximum limit. The ECC has the mathematical denotation of,

$$g^2 = x^3 + ax + b \tag{1}$$

Where, a and b signifies the integers.

During the cryptographic process, the strength of the encryption approach is purely contingent on the mechanism that is deployed for key generation. In this proposed work, 3 keys have to be generated. The public key "$\mathrm{Pb_K}$" is generated from the server for encrypting the message, the private key "$\mathrm{Pr_K}$" is created on the server-side for decrypting the considered message and the $\mathrm{Sc_K}$ is created as of the $\mathrm{Pb_K}$, $\mathrm{Pr_K}$, and point on curve " $\mathrm{H_i}$".

Key generation is a significant part where both $\mathrm{Pb_K}$ and $\mathrm{Pr_K}$ are generated. With the receiver's $\mathrm{Pb_K}$, the sender could encrypt the data, and utilizing the $\mathrm{Pr_K}$, the receiver could decrypt the data.

Now, a number "d" is chosen in the gamut of 'n'.

For generating the $\mathrm{Pb_K}$, consider the subsequent equation,

$$Pb_K = Pr_K * H_i \tag{2}$$

With the subsequent equation, generate the $\mathrm{Sc_K}$.

$$Sc_K = Pb_k * Pr_K * H_i \tag{3}$$

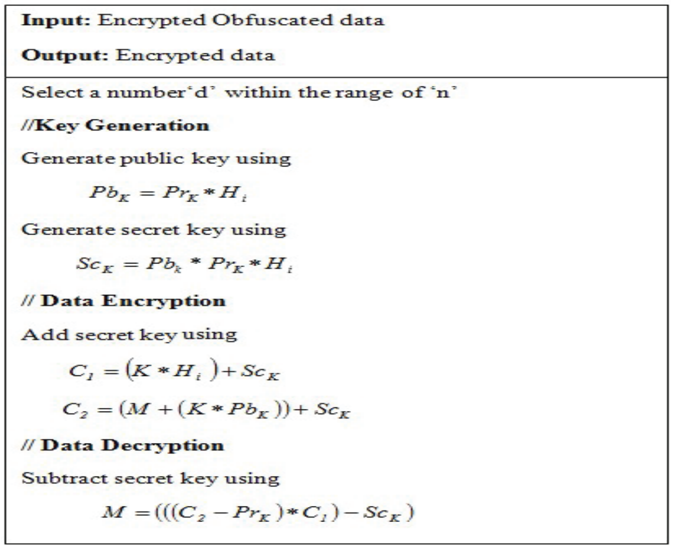
Subsequent to key generation, the information gets encrypted in the form of two CTs that are signified mathematically as,

$$\mathrm{C_1 = (K * H_i) + Sc_K} \tag{4}$$

$$\mathrm{C_2 = (M + (K * Pb_K)) + Sc_K} \tag{5}$$

Here, the $\mathrm{Sc_K}$ is added with "2" CTs. During decryption, this $\mathrm{Sc_K}$ is subtracted as of the decryption equation (as in Equation (6)) to attain an actual message, and this is named as MECC.

$$M = (((C_2 - \underset{K}{\mathrm{Pr}}) * C_1) - Sc_K) \tag{6}$$



```
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
        $C_1 = (K * H_i) + Sc_K$
        $C_2 = (M + (K * Pb_K)) + Sc_K$
// Data Decryption
Subtract secret key using
        $M = (((C_2 - Pr_K) * C_1) - Sc_K)$
```

Figure 2 Pseudocode for Modified ECC algorithm.

Where
M – Original message,
K – Random number which is generated in the gamut of 1 to $n - 1$.

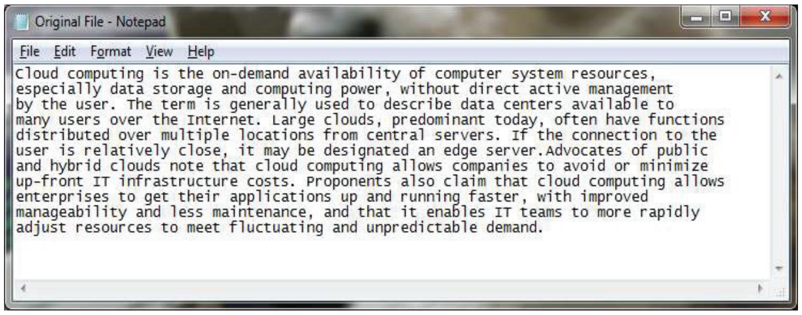
This encrypted information with the changed source IP address is delivered to the user. The proposed MECC algorithm is elucidated with Pseudocode which is evinced using Figure 2.

3.3 Data Decryption

A reverse of encryption process is regarded as decryption. After OB and encryption, the data is retrieved as of the cloud and it is decrypted utilizing the MECC algorithm. Subsequently, the data is de-obfuscated by reversely performing the OB methodologies. That means it executes ascii(), decimal(), two complex(), binary conversion, SC, position marking, and caesar cipher. After performing D-OB, the actual data is retrieved.

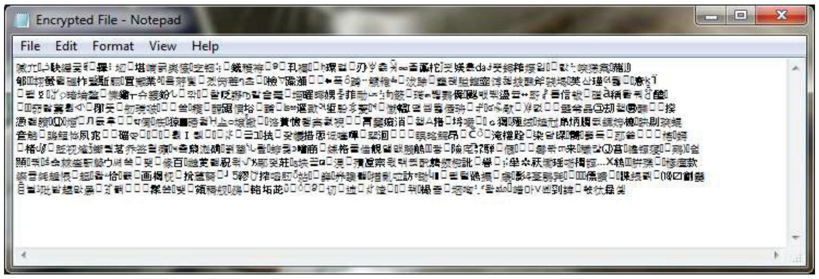
4 Result and Discussion

Here, the proposed methods that are utilized in the data OB and encryption are analyzed.



```
Cloud computing is the on-demand availability of computer system resources,
especially data storage and computing power, without direct active management
by the user. The term is generally used to describe data centers available to
many users over the Internet. Large clouds, predominant today, often have functions
distributed over multiple locations from central servers. If the connection to the
user is relatively close, it may be designated an edge server.Advocates of public
and hybrid clouds note that cloud computing allows companies to avoid or minimize
up-front IT infrastructure costs. Proponents also claim that cloud computing allows
enterprises to get their applications up and running faster, with improved
manageability and less maintenance, and that it enables IT teams to more rapidly
adjust resources to meet fluctuating and unpredictable demand.
```

Figure 3 Input text file.



[illegible]

Figure 4 Encrypted text file.

4.1 Encryption and Decryption

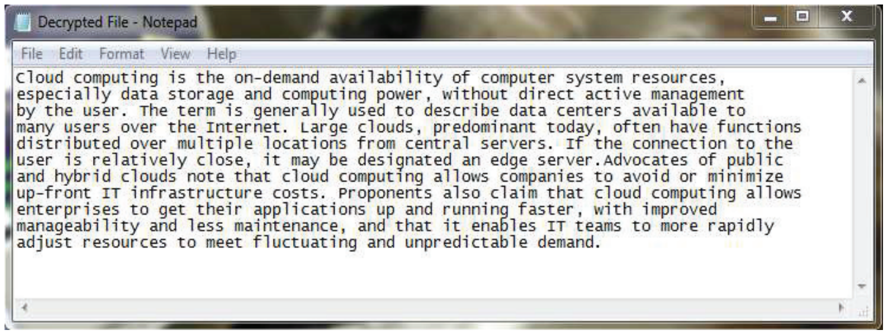
The data submitted to the cloud is obfuscated and encrypted utilizing the OB-MECC algorithm. The plaintext with the file name “Original File.txt” is sent as an input, which is evinced using Figure 3.

Figure 3 evinces the plaintext file for input, which is obfuscated and encrypted utilizing OB-MECC. The encrypted text is stored in another file termed Encrypted File.Txt. The output attained from the proposed encryption is displayed using Figure 4.

The above-encrypted file is now uploaded to the cloud. To access the above-obfuscated data, it should be de-obfuscated. The de-obfuscated data saved in a file is named as Decrypted File.txt. The output acquired from the proposed decryption method is displayed using Figure 5.

4.2 Performance Analysis

Here, the performance rendered by the proposed data OB and proposed data encryption techniques is individually analyzed by contrasting it with the



```
Cloud computing is the on-demand availability of computer system resources,
especially data storage and computing power, without direct active management
by the user. The term is generally used to describe data centers available to
many users over the Internet. Large clouds, predominant today, often have functions
distributed over multiple locations from central servers. If the connection to the
user is relatively close, it may be designated an edge server.Advocates of public
and hybrid clouds note that cloud computing allows companies to avoid or minimize
up-front IT infrastructure costs. Proponents also claim that cloud computing allows
enterprises to get their applications up and running faster, with improved
manageability and less maintenance, and that it enables IT teams to more rapidly
adjust resources to meet fluctuating and unpredictable demand.
```

Figure 5 Decrypted text file.

existing approaches. However, this sort of comparison is done concerning the security level.

4.2.1 Performance Comparison of Data Obfuscation

The proposed data OB approach is contrasted to the existing hexadecimal encoding, Huffman encoding (HE), and Base 64 in respect of OB time and de-OB time, which is evinced in Table 3. The time consumed by the existing and proposed techniques is evaluated for disparately sized plaintexts. Contrasted to the existing OB techniques, the proposed data OB has taken a lower time for obfuscating different size of plaintext.

In Table 3(a), for 1000 kb file, the existing Base 64 takes more time (2045 ms) for data OB, whereas hexadecimal encoding and HE takes 1658 ms and 1504 ms for the same process. But the proposed OB takes 1251 ms for data OB. It is comparatively lower on considering others. The OB time elevates gradually when the file size increases, but the proposed one takes the least time for all (1000 to 5000) file sizes contrasted to others. And, the time taken for D-OB is elucidated using Table 3(b). Here, the proposed D-OB takes a lower time when contrasted to others. The Base 64 method takes maximal time for every file size, and the proposed one takes extremely less time contrasted to others. Table 3 could be graphically elucidated using Figure 6 along with Figure 7.

4.2.2 Performance Comparison of Encryption and Decryption

Here, the proposed MECC algorithm is contrasted to the existing Rivest-Shamir-Adleman (RSA) and ECC approaches in respect of ET and DT. Figure 8 proffers the comparative examination of existing and proposed

Table 3 Performance Comparison of proposed OB with existing techniques in terms of (a) obfuscation time and (b) deobfuscation 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 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

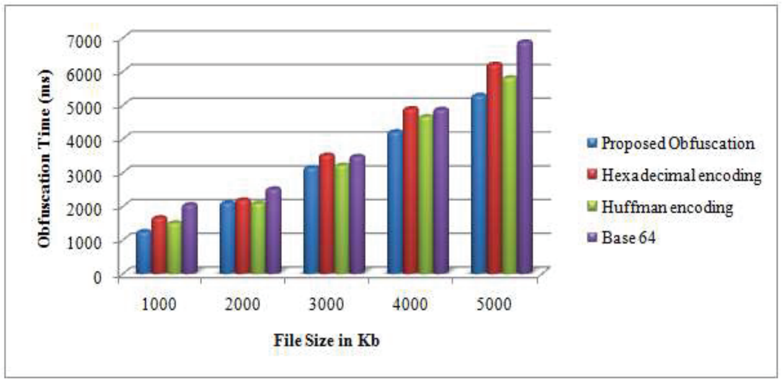


Figure 6 Performance comparison of OB techniques by time.

approaches. Contrasted to the existing encryption algorithms, the MECC approach has taken a lower time for encrypting as well as decrypting the obfuscated data.

Figure 8 contrasted the proposed and existing approaches in respect of ET and DT. In Figure 8(a), the proposed MECC is weighted against the existing RSA and ECC in respect of ET for disparate file sizes. For 1000 kb file, the proposed one takes 6 seconds for data encryption, but the existing ECC and

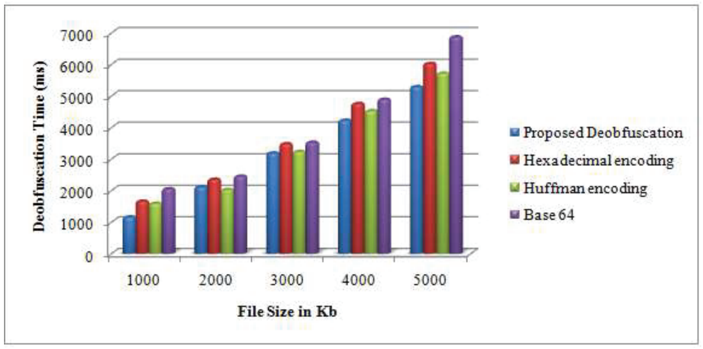
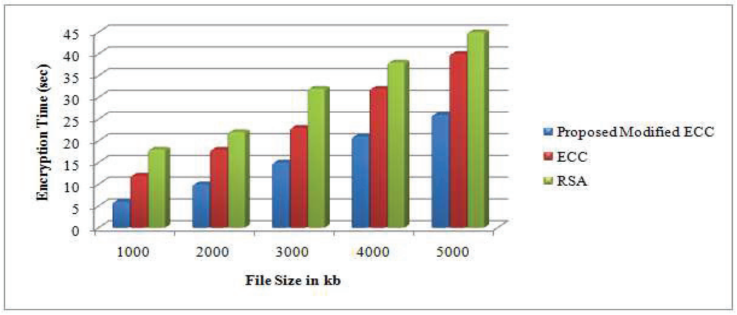
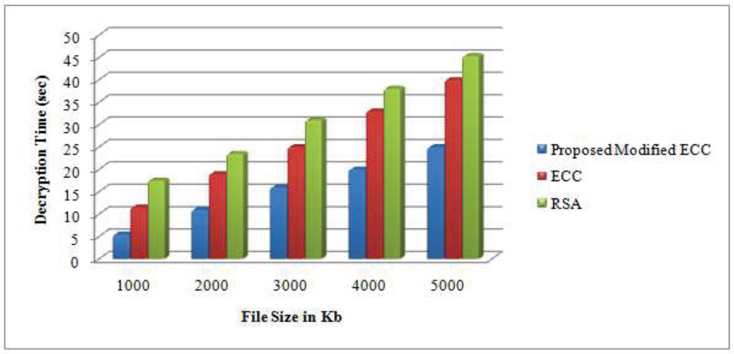


Figure 7 Performance comparison of deobfuscation techniques by time.



(a)



(b)

Figure 8 Comparative analysis of proposed MECC with existing techniques in terms of (a) encryption time, (b) decryption time.

Table 4 Security level of proposed OB-MECC and existing techniques

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

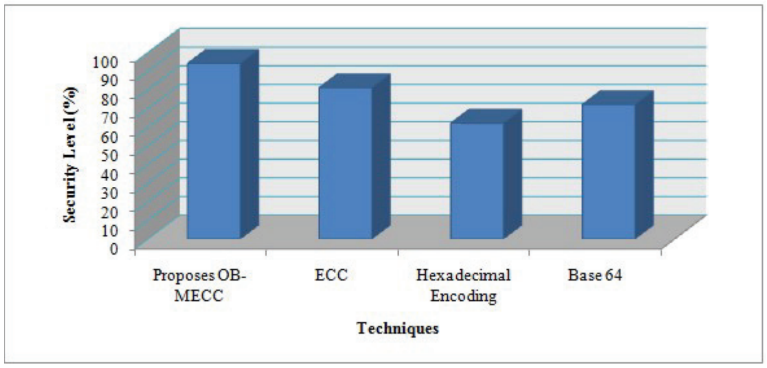


Figure 9 Security Level of proposed and existing methods.

RSA methods take 12 and 18 seconds. The ET elevates gradually when the file size increases, but the proposed one consumes lesser ET contrasted to other existing approaches. Also, the DT is plotted for the different file sizes in Figure 8(b). From Figure 8(b), the proposed one is found to take less DT for every file size when contrasted to the RSA and ECC approaches.

4.2.3 Performance Comparison of Security Level

Table 4 proffers the level of security showed by the proposed and existing approaches. The proposed OB-MECC and the existing ECC, Hexadecimal Encoding, and Base 64 approaches are compared in respect of their security levels. From the attained result, the proposed OB-MECC is found to have maximum security contrasted to the existing approaches.

The graphical illustration of Table 4 is shown in below Figure 9.

Figure 9 signifies that the performance comparison with respect of security level. The comparison made between the proposed OB-MECC with existing techniques such as ECC, Hexadecimal Encoding, and Base 64. The ECC, hexadecimal encoding, and Base 64 attained 81%, 62%, and 72% – security level, respectively, whereas the proposed OB-MECC shows 94% –

security level. Hence, the proposed OB-MECC has the topmost security level when contrasted to the other three approaches.

5 Conclusion

This work proposes to facilitate data confidentiality using a new OB-MECC technique. Before uploading the user's data to the cloud, they are first obfuscated and encrypted. The uploaded data is retrieved as of the cloud and they are decrypted by utilizing modified ECC and D-OB. This method rendered a high-security level (94%) when contrasted to the existing ECC, Base 64, and hexadecimal encoding approaches. While encrypting and uploading data into the public cloud storage, the proposed MECC takes a minimum ET and DT for different file sizes when contrasted to the existing approaches. In this manner, data confidentiality is facilitated in cloud storage and the security is enhanced, through this proposed OB-MECC approach. In the future, the diverse encryption algorithms would be implemented to ameliorate the algorithms' performance and enable CSP to render a secure environment of data storage 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 storaWge”, 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, and 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, and 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.

Biographies

P. Nagaraju received his Bachelor of Technology from JNT University in 2007 and Master of Technology from Andhra University in year 2009. He is currently pursuing Ph.D. and currently working as Assistant Professor in Department of Computational Science, GMR Institute of Technology, Rajam since 2012. He has published more than 5 research papers in reputed international journals. His main research work focuses on Cryptography Algorithms, Network Security, Cloud Security and Privacy, Big Data Analytics, IoT and Computational Intelligence based education. He has 11 years of teaching experience and 5 years of Research Experience.

N. Nagamalleswara Rao received his B.Tech degree in Computer Science and Engineering in 1991 and M.E. degree in CSE in 1993. He is in the teaching field since January 1993. He is currently working as Professor in Department of Information Technology, RVR&JC College of engineering, Guntur since 2012. He completed his Doctorate degree in 2000. His area of specialization is Computer Algorithms, Compilers, Image Processing. He has more than 30 International publications in Journals and fifteen papers in International Conferences to his credit. He is a Senior Member of IEEE. He has 25 years of teaching experience and 20 years of Research Experience.