CHAPTER – V

MATRIX BASE ELLIPTIC CURVE BASED CRYPTO
TECHNIQUE FOR IMAGE BLOCK

5.1 Introduction

Cryptography is commonly employed security is addressed by choosing a security protocol. The security protocols realize the objectives using appropriate cryptographic algorithms. In recent years, significant development in multimedia technologies in the transmission of multimedia data such as audio, video and images over the internet is now very common. Internet is a very insecure channel and these possess a number of security issues and to achieve confidentiality and security of multimedia data.

The processing and transmission of multimedia data over insecure networks, possesses several security problems. The multimedia data security has become a serious and major issue in telemedicine, military, E-Commerce, financial transaction and mobile phone applications. To provide security attributes to multimedia data needs to protect communicated information (plaintext) from unauthorized users. Multimedia data are needs to be secured from different type of attacks. Cryptography enables to transmit data across insecure networks it cannot be read by anyone except the authorized recipient.
5.2 Methodology

5.2.1 Advanced Encryption Standard (AES)

The National Institute of Standards and Technology (NIST) adopted the Rijndael Algorithm as the Advanced Encryption Standard in 2001. This algorithm was invented by two Belgian scientists, Vincent Rijmen and Jon Daemen. The first round of the selection process was focused on the three main criteria that were evaluated to select a winner of the AES process were security, costs and its implementation characteristics (should be easily understood and implemented). The Advanced Encryption Standard has no weakness in its security. Its cost with regards to intellectual patent rights is free and implementation on hardware and software is cheapest among all the finalists. AES is versatile in that it can be implemented on both memory-bound hardware like 8-bit microcontrollers as well as dedicated hardware to provide real-time encryption of streaming data at processing rates reaching gigabits per second.

5.2.2 AES Algorithm

| Step 1: Substitute Byte transformation AES contains 128 bit data block, which means each of the data blocks has 16 bytes. |
| Step 2: In sub-byte transformation, each byte (8-bit) of a data block is transformed into another block using an 8-bit substitution box which is known as Rijndael Sbox. |
| Step 3: It is a simple byte transposition, the bytes in last three rows of the state, depending upon the row location, are cyclically shifted. |
| Step 4: For 2nd row, 1 byte circular left shift is performed. For the 3rd and 4th row 2-byte and 3-byte left circular left shifts are |
performed respectively.

Step 5: This round is equivalent to a matrix multiplication of each Column of the states.

Step 6: A fix matrix is multiplied to each column vector. In this operation the bytes are taken as polynomials rather than numbers.

Step 7: It is a bitwise XOR between the 128 bits of present state and 128 bits of the round key. This transformation its own inverse.

5.2.3 Data Encryption Standard

Data encryption standard is the most widely used method of data encryption using a secret key. For each given message, the key is chosen at random from among this enormous number of keys. Like other private key cryptographic methods, both the sender and the receiver must know and use the same private key. It was developed in the 1970s by the National Bureau of Standards with the help of the National Security Agency. Its purpose is to provide a standard method for sensitive commercial and unclassified data. IBM created the first draft of the algorithm, calling it LUCIFER. DES officially became a standard in November of 1976. Data encryption algorithm has a 64-bit block size and uses a 56bit key during execution (8 parity bits are stripped of from the full 64-bit key). The DEA can also be used for single user encryption, such as to store files in a hard in encrypted form. NIST re-certifies DES every 5 years. DES has been in worldwide use for over 20 years, and due to the fact that it is a defined standard that any system implementing DES can communicate with any other system using is it.
5.3 Proposed MBECC Methodology

Information security is one of the most important issues now-a-days where information is sent from one to another place with rapid rates. Multitude use of digital data in the applications of medical, defense, military, banking and other multimedia channel leads the concept of authentication of digital data. So the best way of transferring huge amount of digital data is in form of image. Due to inherent property of image, such as huge information capacity and high correlation among pixels, it is selected for the encryption algorithms. There are many image encryption algorithms which utilized chaotic map, logistic map, advance encryption standard, Arnold map, affine transformation, Fourier transform and fractional Fourier transform. Researcher finished the purpose of encryption by scrambling the image pixels only, some have changed the spatial domain of image to frequency domain by using Fourier transform. The extension of Fourier transform is fractional Fourier transform which is also applied in large extent in the field of image encryption. These techniques do not fulfill the requirements of the authenticity of the image against malicious users. Recently, Linear Canonical transform is applied multitude in the field of double image encryption process due to its inherent property.

The primary goal is to provide security of images which is travelling over internet. Moreover, an image-based data requires more effort during encryption and decryption. In this work, enhanced
technique has been developed for mapping the image using Matrix Base Elliptic Curve Cryptography (MBECC) analysis the entropy and correlation between pixels value of various image encryption algorithm. The need to develop new encryption schemes comes from the fact that traditional encryption schemes for textual data are not suitable for multimedia data stream. This paper presents a framework to evaluate image encryption schemes proposed. MBECC considered the input plain image into ASCII values. In this algorithm the input image is first converted into its ASCII values. It performs a string manipulation algorithm which will change the relative position of atomic data values by reversing them. Here, divide the string into square matrices of maximum possible order and then add magic square matrix of same size is considered. The base conversion is performed on the basis of key which is calculated by the size of square matrix generated. The base conversion is also performed on the remaining elements which could not be containing in the square matrices. The experimental result shows that MBECC provides better for encrypting and decrypting for digital images when compared with the existing methods AES and DES.
**Fig.5.1 Process Flow of MBECC**

1. **Image Database**
2. Shuffle Pixels of Image in Selected Block
3. Binary Code Conversion
4. Addition/Subtraction Function Using Base Key
5. Cipher Image

**Encryption Procedure**

**Decryption Procedure**

1. Addition/Subtraction Function Using Base Key
2. Reverse Binary Function
3. Binary Code Conversion
4. Match
5. Original image
Image Encryption

Image encryption is the process of encoding image in such a way that eavesdroppers or hackers cannot read it, but that authorized parties can. In an encryption scheme, the image is encrypted using an encryption algorithm, turning it into an unreadable image. This is usually done with the use of encryption keys, which specifies how the image is to be encoded. Any adversary that can see the encrypted image should not be able to determine anything about the original image. An authorized party, however, is able to decode the encrypted image using a decryption algorithm. That usually requires a secret decryption key, so adversaries do not have access.

The technique of converting a given number from one number system to another by means of simple calculations is known as Matrix Base Conversion. A square matrix in which the sum of all elements in each column and in each row is same is called Magic Square matrix. To calculate the sum use the formula \( r^2 \frac{(r^2+1)}{2} \), where \( r \) is the size of square matrix. A square matrix order refers to a matrix with equal number of rows and columns.

This new proposed block encryption algorithm is block cipher. It divides data into blocks of pixels of equal length. Some blocks of pixels are selected and the only selected blocks of pixels are encrypted using a special mathematical set of functions known as key. Symmetric key technique is used in this algorithm for both encoding and decoding i.e. same key is used at both ends. Some additional tasks are performed
to provide strong security to this algorithm like shuffling. There is another plus point of this algorithm is that it protects the cipher image from unauthorized access such as Brute-force as selection process is applied and the key is changed many times in the encryption process. It will be very hard to attain original image from cipher image.

**Elliptic Curve Cryptography**

The basic idea of Elliptic Curve Cryptography (ECC) and its implementation through co-ordinate geometry for data encryption. The implementation of ECC on two finite fields, prime field and binary field. An overview of ECC implementation on two dimensional representations of plaintext coordinate systems and data encryption technique. Much attention given to the mathematics of elliptic curves starting with their derivations and the proof of how points upon them form an additive abelian group for cryptographic purposes, specifically results for the group formed by an elliptic curve over a finite field can form public key cryptographic systems for encryption and key exchange. The algorithm is mainly based on scan patterns depicted. The steps formed the new image encryption. The 128-bit key is gradually explained and formed during each step implemented by gray scale images.

To use ECC all parties must agree on all the elements defining the elliptic curve, that is, the domain parameters of the scheme. The field is defined by p in the prime case and the pair of m and f in the
binary case. The elliptic curve is defined by the constants a and b used in its defining equation. Finally, the cyclic subgroup is defined by its generator point G. For cryptographic application the order of G, that is the smallest value of n such that nG = 0 is a prime number.

Elliptic Curve Cryptography, operations are performed on the coordinate points of an elliptic curve. The main operations are key agreement for encryption and decryption, signature generation for signing and verification involve point addition, point multiplication techniques. To perform addition of two distinct points coordinate the following calculation is used.

//Rules for Point Addition and Subtraction//

Step 1: Consider two distinct points P1 and p2 such that

\[ P1 = (x_1, y_1) \text{ and } P2 = (x_2, y_2) \]

Let \( p3 = p1 + p2 \) where \( p3 = (x_3, y_3) \), then

Step 2: If \( x_1 \neq x_2 \)

\[ m = (y_2 - y_1) / (x_2 - x_1) \mod p, \]

//m is the slope of the line through P1 and P2//

Otherwise, \( x_1 == x_2 \), \( m=(3x_1^2 + b) / (2y_1) \mod p \)

\[ X3 = m^2 - x_1 - x_2 \mod p, \]

\[ Y3 = m (x_1 - x_3)-y_1 \mod \]

To perform point subtraction, get a mirror coordinate of the subtracted point along x-axis and perform point addition on the resulting coordinate and the other coordinate. Point doubling is perform to add up two points which are same i.e. they have same
coordinate value. Multiplication is repeated addition of the base coordinate point. Many algorithms have been developing to perform point multiplication swiftly.

//Rules for Point Doubling//

Step 1: Consider a point P such that

\[ P = (x_1, y_1) \]

Let \( p_3 = 2p_1 \) where \( p_3 = (x_3, y_3) \) then

Step 2: Let \( M = (3x_1^2 + b) / (2y_1) \mod p \)

\[ X_3 = m^2 - x_1 - x_2 \mod p, \]

\[ Y_3 = m(x_1 - x_3) - y_1 \mod p \]

Pixel grouping into a single integer

Images are made up of pixels. If cryptographic operation is performed on every single pixel it will take more time as the number of pixels present is very large. So, it will be a good option to group the pixels together. The number of pixels to be group depends on the Elliptic Curve parameters used. The larger the parameter of the elliptic curve, the more pixels can be grouped. To convert the group of pixels into a big single integer we have used a function of Mathematica called From Digits [list of pixels, b] which take a list of pixels and convert it to base b. Then add random 1 or 2 to each pixel to avoid error caused while using From Digits function of mathematica the first pixel value of the group is 0 and also to provide low correlated pixel value for the cipher image generated with same pixel value plain image. Pixel value of image in byte form will range from 0 to 255.
**Group of pixels from the big integer**

After the ECC operation the coordinate value will all be in the range of the bit size chosen for the ECC operation. To generate the cipher image from these coordinates we need to bring it down to 0 to 255 ranges. To perform using the Integer Digits [big integer value, 256] function in Mathematica. It takes as input the big integer values in the range of the size chosen for ECC operation and with base 256, the output will be a list of values ranging from 0 to 255. Mathematical operation on an image is done on the pixels value of the image. So first the pixels value of the image. The Elliptic curve parameters \{a, b, G, p\} are agreed between the sender and the receiver. The sender uses the public key ‘Pb’ of the receiver to generate the cipher image from the pixels of the plain image. The receiver use the private key which was used to generate the public key, to decrypt the cipher image back to the plain image.

**Rearrangement of Blocks using Scan Patterns**

In first step, the original image divided into 8x8 blocks (64 blocks in each column and row). Blocks are repositioned by the needed number of predefined scanning algorithm to shuffle the image structure. Obviously, exploiting desired number of different 8 scanning algorithms make the result image being more confusion.

**Rearrangement of Pixels each Block using Scan Patterns**

By repositioning the blocks of image in previous step, tangible amount of correlation between different regions of original image has removed but pixels of each block are correlated yet. It is obvious that
more shuffling cause more independency of pixels and make the concept of image impossible.

In second step, for more shuffling the transformed image obtaining in previous step, the rearrangement is run on the each block of image individually. To reaching that aim the scanning algorithms are exploited again.

Each block is treated as an image so that it can be scanned for 0 to 7 times by different 8 scan patterns. So all pixels of blocks can rearrange several times, so a new order of scanning algorithm can be generated again, and shuffle the arrangement of pixels within each block more than before.

**Fig 5.2 Process Flow of ECC**

In this algorithm can be implemented using mat lab interface provides executor time, processing time, blocking queue and environment variables which provide capability to parallelize a serial program and show less execution time. Several methods have been
used to encrypt and decrypt using elliptic curves. The common one is to simulate the ElGamal cryptosystem using an elliptic curve over GF (P) or GF (2n). Operations such as addition and multiplication are over an elliptic curve group.

![Fig.5.3 ElGamal Cryptosystem Using the Elliptic Curve](image)

**Key generation**

First, select the point value E(a,b) with an elliptic curve over GF(p) or GF(2n). Then choose the point on the alphabetic table corresponding to the letter as plain text and select the private key value as d. To calculate the point as e2=(x2, y2) using the formula d*e1. Finally, announce e1, e2 as public key and keep “d” as a private key.

**Encryption**

User A select the point value p as plaintext and private key r for sender. Then calculate a pair of points on the text as cipher text. The cipher texts are as c1 and c2. Hence, c1=r * e1 and c2=p + r * e2.
**Decryption**

User B after receiving $c_1$ and $c_2$, calculates $p$, the plaintext using the formula as $p = c_2 - (d \cdot c_1)$. Here minus sign means adding the inverse. To prove the point $p$ calculated by receiver is the same as that by sender.

//Rules for Key Generation //

Select $E_{11}(1,1)$, $e_1 = (x_1, y_1) = (1, 3)$, $d=2$

Calculate $e_2 = d \cdot e_1$

$= 2 \cdot (1, 3)$

$= (1, 3) + (1, 3)$

$M = (3x_1 + a) / 2y_1 \mod p$

$= (3(12) + 1) / 2(3) \mod 11 = 8$

$x_3 = (m_2 - x_1 - x_2) \mod p$

$= (82 - 1 - 1) \mod 11 = 7$

$y_3 = m(x_1 - x_3) - y_1$

$= 8(1 - 7) - 3 \mod 11 = 4$

$(x_3, y_3) = e_2 = (7, 4)$

//Rules for Encryption //

Select $r = 1$, Plaintext $P = (2, 3)$

Calculate $c_1 = r \cdot e_1$

$= 1 \cdot (1, 3)$

Cipher text $c_1 = (1, 3)$

Calculate $c_2 = p + r \cdot e_2$
First, \( r \cdot e_2 = 1 \cdot (7, 4) = (7, 4) \)

Second, \( c_2 = (2, 3) + (7, 4) \)

\( M = \frac{y_2 - y_1}{x_2 - x_1} \mod p \)

\[ = \frac{4 - 3}{7 - 2} \mod 11 = 9 \]

\( X_3 = (m^2 - x_1 - x_2) \mod p \)

\[ = (9^2 - 2 - 7) \mod 11 = 6 \]

\( Y_3 = m(x_1 - x_3) - y_1 \mod p \)

\[ = 9(2 - 6) - 3 \mod 11 = 5 \]

Cipher text \( C_2 = (6, 5) \)

//Rules for Decryption //

\( P = c_2 - (d \cdot c_1) \)

First calculate \( d \cdot c_1 = 2 \cdot (1, 3) = (7, 4) \)

\( P = (6, 5) + (7, 4) \)

\( M = \frac{y_2 - y_1}{x_2 - x_1} \mod p \)

\[ = \frac{-4 - 5}{7 - 6} \mod 11 = 2 \]

\( X_3 = (m^2 - x_1 - x_2) \mod p \)

\[ = (2^2 - 6 - 7) \mod 11 = 2 \]

\( Y_3 = m(x_1 - x_3) - y_1 \mod p \)

\[ = 2(6 - 2) - 5 \mod 11 = 3 \]

Plaintext \( P = (2, 3) \)
5.4 MBECC Algorithm

Key Generation Algorithm

```plaintext
//** Algorithm for Key Generation **//
Step 1: Initialize the key
Step 2: Select E (a,b) with an elliptic curve over GF(p) or GF(2n).
Step 3: Select a point on the curve e1= (x1, y1).
Step 4: Select d
Step 5: Calculate e2= (x2, y2) = d * e1
Step 6: Announce e1, e2 as public key and keep “d” as key.
Step 7: End
```

Encrypt Image Algorithm

```plaintext
//** Algorithm for Encrypting Image **/
Input: Plain image
Output: Cipher image

Step 1: Load an input image.
Step 2: Procedure random key ()
 {
   Convert into ASCII value
   then
   Convert binary values into random values
   Store as base key
 }
Step 3: A new key is generated every time on cipher image.
Step 4: Initialize I=0 in encrypted process
Step 5: Procedure image blocks ()
 {
   Step 1 : input M, N   //size of input image
   Step 2 : input p,q   //p,q← horizontal and vertical blocks
   Step 3 : assign q←0, p←0
   Step 4 : for i = 1 to M
            for j=1 to N
               {
               HNB = int (Image width/10)
               VNB = int (Image Height/10)
               }
   Step 5: Establish the horizontal and vertical blocks of the
           input image
   Step 6: Return
 }
Step 6: Extract images with the same size as the original image
Step 7: Construction key-image and encrypted image.
Step 8: Save as the key-image and encrypted images.
Step 9: Reverse the decrypted process
Step 10: Construct and display the decrypted image.
```
Decrypt Image Algorithm

//** Algorithm for Decrypting Image **/

Input: Cipher image
Output: Plain image

**Step 1:** Load an input image.

**Step 2:** Extract images with the same size as the original image.

**Step 3:** Construction key-image and decrypted image.

**Step 4:** Initialize I=0 in decrypted process.

**Step 5:** Procedure image blocks ()

{

**Step 1:** input M, N //size of input image

**Step 2:** input p, q //p, q ← horizontal and vertical blocks

**Step 3:** assign q ← 0, p ← 0

**Step 4:** for i = 1 to M

  for j = 1 to N

  {
    HNB = int (Image width/10)
    VNB = int (Image Height/10)
  }

**Step 5:** Establish the horizontal and vertical blocks
          of the input image

**Step 6:** Return

}

**Step 6:** Procedure random key ()

{

  Convert into ASCII value
  then
  Convert binary values into random values
  Store as base key

}

**Step 7:** A new key is generated every time on plain image.

**Step 8:** Save as the key-image and decrypted images.

**Step 9:** Reverse the decrypted process.

**Step 10:** Construct and display the decrypted image.
5.5 Experiments and Results

The proposed method is experimented with different types of images are processed using MATLAB. The table 5.1 shows the comparison values of Entropy values. The algorithm was applied on a bit mapped (bmp) image that size. To evaluate the impact of the insertion process on the encrypted images, three different cases were tested. The horizontal number of blocks and the vertical number of blocks is 1024 blocks. Therefore, the number of bits that need to be sent within the encrypted image will be 20 bits 10 bits for horizontal number of blocks and 10 bits for vertical number of blocks. These 20 bits will be inserted in the image data randomly based on the secret key by using the LSB insertion. Hence the encrypted images with and without insertion position of data. The experimentation is carried out by MATLAB. It stands for MATrix LABoratory. MATLAB® is a high-performance language for technical computing. It integrates computation, visualization and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include Math and computation Algorithm development Data acquisition Modeling, simulation and prototyping Data analysis, exploration and visualization Scientific and engineering graphics Application development, including graphical user interface building.

In order to evaluate the performance of the proposed system with the existing system the Entropy Values, Mean Square Error and
Block Size are computed with the selective image sets. The obtained results of the image with the proposed model is tabulated in the following Table 5.1 and the performance are evaluated with the existing the AES and CFES techniques.

<table>
<thead>
<tr>
<th>Plain-image</th>
<th>Encrypted Image</th>
<th>Decrypted Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td><img src="image1" alt="Lena Encrypted" /></td>
<td><img src="image2" alt="Lena Decrypted" /></td>
</tr>
<tr>
<td>Pepper</td>
<td><img src="image3" alt="Pepper Encrypted" /></td>
<td><img src="image4" alt="Pepper Decrypted" /></td>
</tr>
<tr>
<td>Barbara</td>
<td><img src="image5" alt="Barbara Encrypted" /></td>
<td><img src="image6" alt="Barbara Decrypted" /></td>
</tr>
</tbody>
</table>

*Fig.5.4 Image Results Obtained for MBECC*
From the below Table 5.1 images has been considered for the experimentation and the performance of the proposed approach compared to the approaches such as AES and CFES is shown in Table 5.1. It is observed from the table that the proposed MBECC has better in entropy values for all the standard images taken into consideration. The entropy value 7.03 is obtained for the proposed MBECC respectively which is high when compared to the approaches such as AES and CFES with entropy value of 7.17 is obtained and AES provides the entropy value of 7.99 respectively. The pictorial representation of the experimentation and its performance evaluation are presented in the Fig.5.5 is presented below.

### Table 5.1 Performance Analysis of MBECC with Entropy Values

<table>
<thead>
<tr>
<th>Images</th>
<th>AES</th>
<th>CFES</th>
<th>Proposed MBECC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cameraman</td>
<td>7.99</td>
<td>7.14</td>
<td>7.03</td>
</tr>
<tr>
<td>Baboon</td>
<td>7.99</td>
<td>7.14</td>
<td>6.83</td>
</tr>
<tr>
<td>Lena</td>
<td>6.54</td>
<td>7.03</td>
<td>6.29</td>
</tr>
<tr>
<td>Pepper</td>
<td>6.52</td>
<td>6.89</td>
<td>5.58</td>
</tr>
</tbody>
</table>

The pictorial representation of the experimentation and its performance evaluation are presented in the Fig.5.6 is presented below.

**Fig.5.5 Performance Analysis of MBECC with Entropy Values**
From the below Table 5.2 shows the experimented values obtained from different methods. The performance was evaluated using the Mean Square Error (MSE). In order to evaluate the performance of the proposed MBECC, the results obtained with the existing techniques such as AES and CFES are compared with the proposed MBECC and are shown in Table 5.2.

**Table 5.2 Performance Analysis of MBECC for MSE Values**

<table>
<thead>
<tr>
<th>Image</th>
<th>AES</th>
<th>CFES</th>
<th>Proposed MBECC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cameraman</td>
<td>40.39</td>
<td>33.86</td>
<td>31.16</td>
</tr>
<tr>
<td>Baboon</td>
<td>40.34</td>
<td>33.31</td>
<td>32.75</td>
</tr>
<tr>
<td>Lena</td>
<td>39.78</td>
<td>33.10</td>
<td>32.43</td>
</tr>
<tr>
<td>Pepper</td>
<td>39.75</td>
<td>33.06</td>
<td>30.17</td>
</tr>
</tbody>
</table>

The pictorial representation of the experimentation and its performance evaluation are presented in the Fig.5.6 is presented below.

*Fig. 5.6 Performance Analysis of MBECC for MSE Values*
From the above Table 5.3 image blocks has been considered for the experimentation with different sizes. The performance is analyzed with encryption and decryption time is shown in Table 5.3. The speed of encoding and decoding of data is the core advantage of any cryptographic algorithm. The block encryption algorithm is specially designed to reduce the cost and execution time of the process. The following table shows the performance analysis of block encryption algorithm against the some well known algorithms. The following table 5.3 shows the memory required by the image block encryption and decryption. The memory requirement of image block encryption is half as compare to other techniques.

<table>
<thead>
<tr>
<th>Image Block Size</th>
<th>Encryption Time (ms)</th>
<th>Decryption Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8X8</td>
<td>20.17</td>
<td>22.86</td>
</tr>
<tr>
<td>16X16</td>
<td>40.38</td>
<td>47.33</td>
</tr>
<tr>
<td>32X32</td>
<td>81.57</td>
<td>98.16</td>
</tr>
<tr>
<td>64X64</td>
<td>165.14</td>
<td>203.74</td>
</tr>
<tr>
<td>128X128</td>
<td>322.36</td>
<td>409.72</td>
</tr>
<tr>
<td>196X196</td>
<td>648.49</td>
<td>963.63</td>
</tr>
<tr>
<td>256X256</td>
<td>611.81</td>
<td>922.24</td>
</tr>
</tbody>
</table>

The pictorial representation of the experimentation and its performance evaluation are presented in the Fig.5.7 is presented below.
5.6 Summary

In this chapter, a matrix base elliptic curve cased crypto technique has been proposed with image blocks for image encryption-decryption has been proposed which utilizes matrix multiplication and inverse matrices. The proposed technique is experimented and compared with existing AES, DES and CFES techniques. Compare with those techniques the proposed model provides better results in image block based encrypt and decrypt efficiently. The performance is evaluated by estimating the MSE and entropy values and compared with methods the proposed MBECC technique provides better encrypting and decryption for images.