International Journal of Research Studies in Computer Science and Engineering (IJRSCSE)
Volume 2, Issue 2, February 2015, PP 18-27
ISSN 2349-4840 (Print) & ISSN 2349-4859 (Online)
www.arcjournals.org

A New Digital Encryption Scheme: Binary Matrix Rotations Encryption Algorithm

Kommerla Siva Kumar

Assistant Professor
Department of Computer Science& Engineering
RVR&JC CE, Chowdavarm, Guntur, AP, India
sivakumar_kvr@yahoo.com

Meda Srikanth

Associate Professor, Department of CSE
RVR&JC CE, Chowdavarm, Guntur, AP, India
medasrikanth@gmail.com

Tenali Anusha

Assistant Professor, Department of CSE
P.N.C &Vijay Institute of Engg &Tech
Guntur, AP, India.

Abstract: *In today's world most of the communication is done using electronic media. Data security plays an important role in such communication. So, there's a requirement for a stronger encoding that is extremely exhausting to crack. Completely different encrypted algorithms are planned thus far to come up with encrypted information of original information. In this we have proposed a new replacement algorithmic rule for Digital encoding called as "Binary Matrix Rotations Technique" (BMR) which reduces size of the data as well as form of the data. The experimental results show that the new theme has very fast encoding and safer.*

Keywords: *Information security, Matrix Rotations, Bytes, Encryption, Decryption, Plain Text, Cipher text*

1. INTRODUCTION

In today's world most of the communication is done using electronic media.Information security could be a difficult issue in today's technological world. Hence, there is a need to protect data from malicious attacks. This can be achieved by Cryptography. Security of the information over the insecure mode of communication, Internet, has been an area of research for several years. There are several techniques developed for encryption/decryption of the information over theyears [10][11].

In this paper we discusses a new technique for encrypting data which enables good diffusion and is having a unique technique of decrypting it back to the plaintext and is easy to implement using matrix rotations technique. The encryption algorithm of magic cube projected by Yongwei et al, which is implemented in three dimensional space [1][2][3]. The algorithm is complicated and very difficult to understand. F.Y.LI Min. Proposed queue transformation based digital image encryption algorithm [4], which works efficiently with low time complexity compared to Yongwei et al. M. Kiran Kumar, S.Mukthyar Azam proposed efficient digital encryption algorithm based on matrix scrambling technique [5], which works efficiently with low time complexity but drawback of this algorithm is don't change the form and size of the data. To overcome this drawback we proposed a "Binary Matrix Rotations Encryption Algorithm" (BMR) Technique which is based on random functions, shifting and reverse techniques of circular queue, with efficient time complexity.

In section II, Encryption Process is explained. In section III, Decryption Process is explained. In section IV, results are explained and finally Conclusion is provided in section V.

2. ENCRYPTION PROCESS

Encryption is the process of encoding messages or information in such a way that only authorized parties can read it. In an encryption scheme, the message or information, referred to as 'plaintext', is encrypted using an encryption algorithm, generating 'cipher text' that can only be read if decrypted.

BMR techniqueis especially concerned in 2 levels. In the 1st level we perform "8-7 Bytes" Technique. In 8-7 Bytes Technique, each 8 bytes of Text becomes 7 bytes of text. So, the advantage of this technique is it reduces the size of data. In the 2nd level we perform "Matrix Rotations Technique". In "Matrix Rotations Technique", we have to perform shifts and reverse operations on given data.In BMR encryption process first off all, plain text is chooses which is from input text file and given binary equivalent of plain text as input to the "8-7 Bytes technique", each 8 bytes of Text becomes 7 bytes of text. So, the advantage of this technique is it reduces the size of data. The output of the "8-7 Bytes technique" is given as input to the "Matrix rotations technique". In this technique, we would like to perform two operations like Row Transformation and Column Transformation. In Row Transformation we are able to perform three operations like row left, row right and row reverse operations and conjointly we are able to perform three operations in Column Transformation like column up, column down and column reverse operations. Finally cipher text is obtained from encryption algorithm which is in Binary form so we can convert into ASCII equivalent of cipher text. The Fig1 shows total encryption process of BMR technique.

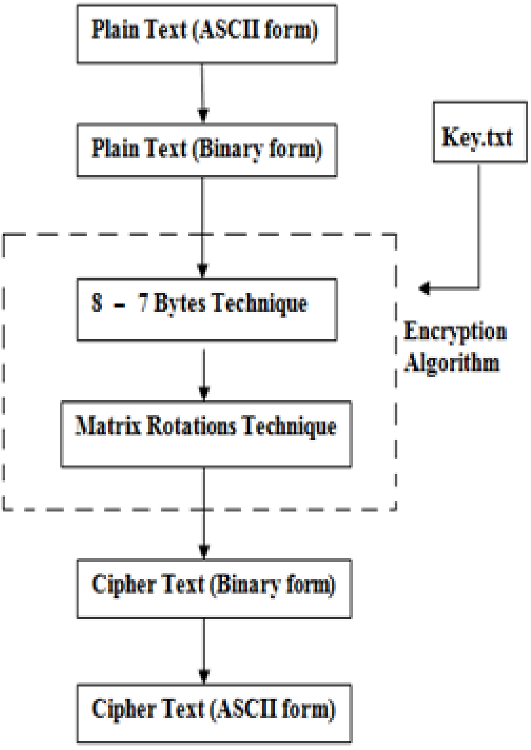


Fig1. *Block Diagram for Encryption Process*

2.1. Algorithm for "8-7 Bytes Technique"

In "8-7 Bytes technique", each 8 bytes of text becomes 7 bytes of text. So, the advantage of this technique is it reduces the size of data and change the form of data also.

The following are the steps for doing "8-7 Bytes technique".

1. Read data into the matrix, 'A'.
2. Read every 8 letters as one group and set count value with number of 8 letters groups.
3. Read binary equivalent 8 letters of the group into matrix B[ ][ ].
4. Do, B[0][0] = B[7][7],B[1][0] = B[7][6],B[2][0] = B[7][5], B[3][0] = B[7][4],B[4][0] = B[7][3],B[5][0] = B[7][2], B[6][0] = B[7][1].
5. Write ASCII equivalent values of Binary values on output file.
6. Check whether Count is > 0 or not

i. If count > 0 then move to ‘step 3’.

ii. Otherwise move to end.

2.2. Algorithm for Matrix Rotations(MR) Technique

In this technique, we would like to perform two operations like Row Transformation and Column Transformation. In Row Transformation we are able to perform three operations like row left, row right and row reverse operations and conjointly we are able to perform three operations in Column Transformation like column up, column down and column reverse operations [7]. The Fig 2 shows total process of Matrix Rotations Technique.

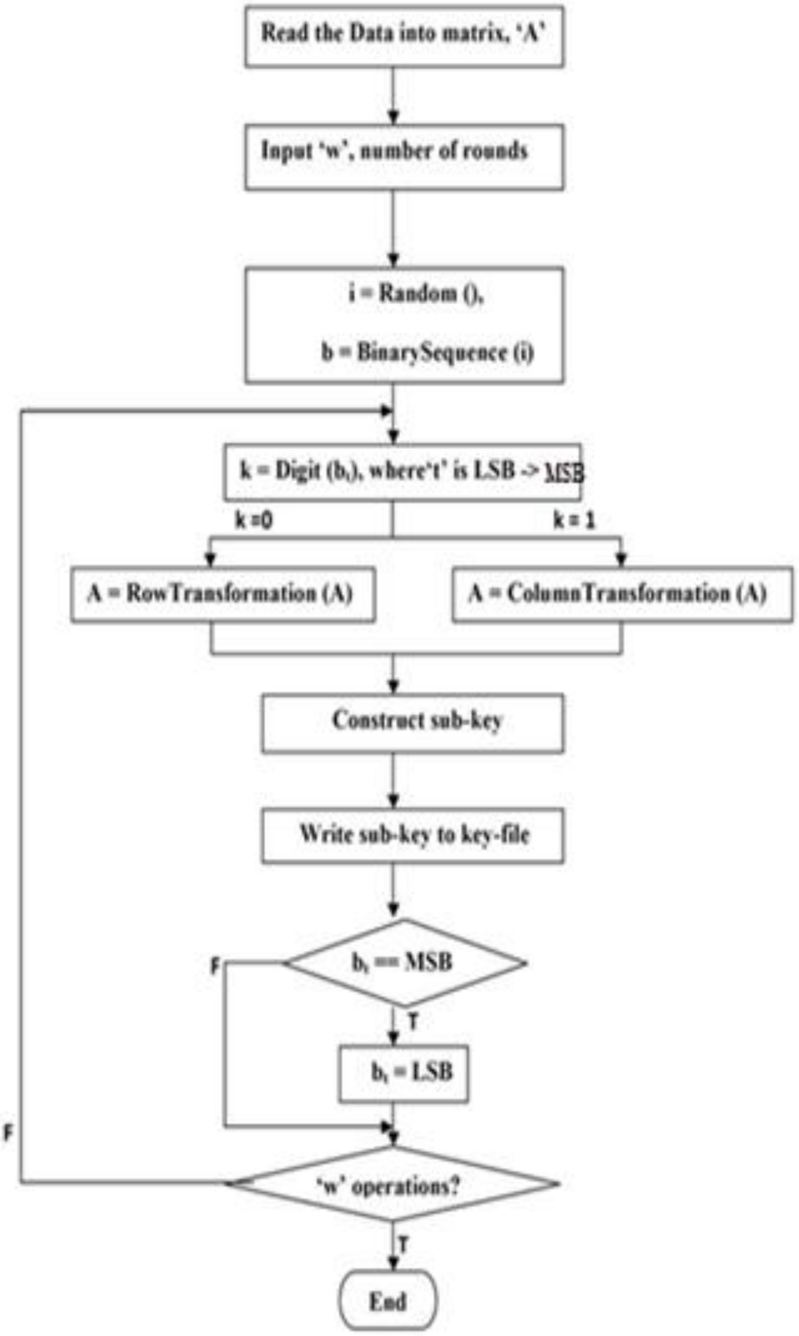

Read the Data into matrix, ‘A’
Input ‘w’, number of rounds
i = Random (),
b = BinarySequence (i)
k = Digit (b_t), where‘t’ is LSB -> MSB
k =0
k = 1
A = RowTransformation (A)
A = ColumnTransformation (A)
Construct sub-key
Write sub-key to key-file
F
b_t == MSB
T
b_t = LSB
F
‘w’ operations?
T
End

Fig2. *Block Diagram for Matrix Rotations Technique*

The following are the steps for doing Matrix Rotation Encryption technique.

a. Read data into the matrix, ‘A’.

b. Set number of rounds as ‘w’.

c. Generate random numberas ‘i’ and takes binary sequence of that random number (b = Binarysequence (i)).

d. Let k= Digit(b_t), where‘t’ is bit position, k value is either 0 or 1, which deciding either to perform Row transformation or Column Transformation.

  i. If k = 0, then Row transformation is performed.

  ii. If k = 1, then Column transformation is performed.

e. Check whether binary sequence in vector b is completed. If it is completed, again start from first digit in binary sequence of b.(LSB-->MSB).

f. Repeat steps’d’ and ‘e’ for ‘w’ number of times.

g. Record all sub keys sequentially in a key file.

Row Transformation: The Row transformation algorithm is show in Fig 3, two rows are selected randomly, r1 = Random(m),r2 = Random(m) similarly two random values of columns c1,c2 are selected i.e. c1=Random(n), c2=Random(n), to determine the range of rows on which transformation has to be performed. The constraint here is r1 ≠ r2 and c1≠ c2. Let x1 = min(c1,c2), x2 = max(c1,c2), Here x1 and x2 becomes lowest index and highest index of the sub array selected in the rows r1 and r2 . Let op = Random() mod 3, hence op takes three possible values 0 , 1 and 2. Thus three row operations are performed on the matrix. If op = 0, then circular left shift operation is performed on rows r1 and r2 on the sub portion of range x1 to x2. If op = 1, then circular right shift operation is performed on rows r1 and r2 on the sub portion of range x1 to x2. If op = 2, then perform reverse operation on the sub array of r1 and r2 in the range from x1 to x2. For each operation a sub-key is constructed and recorded in a key file.

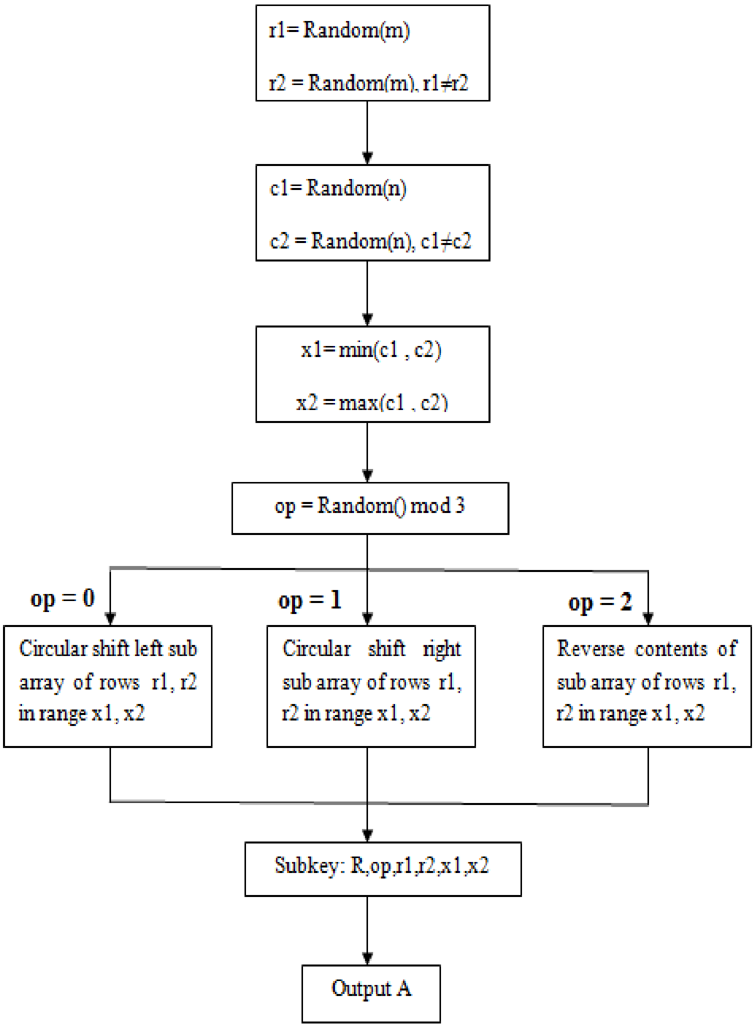

r1= Random(m)
r2 = Random(m), r1≠r2
c1= Random(n)
c2 = Random(n), c1≠c2
x1= min(c1 , c2)
x2 = max(c1 , c2)
op = Random() mod 3
op = 0
op = 1
op = 2
Circular shift left sub array of rows r1, r2 in range x1, x2
Circular shift right sub array of rows r1, r2 in range x1, x2
Reverse contents of sub array of rows r1, r2 in range x1, x2
Subkey: R,op,r1,r2,x1,x2
Output A

Fig3. *Row Transformation Process*

Column Transformation: Column Transformation algorithm is shown in Fig 4, Two Columns are selected randomly, c1 =Random(n), c2 = Random(n) similarly two random values of rows r1,r2 are selected i.e. r1=Random(m), r2=Random(m), to determine the range of columns on which transformation has to be performed. The constraint here is c1 ≠ c2 and r1 ≠ r2. Let x1 = min (r1, r2), x2 = max (r1, r2), Here x1 and x2 becomes lowest index and highest index of the sub array selected in the columns c1 and c2. Let op = Random() mod 3, hence op takes three possible values 0 , 1 and 2. Thus three column operations are performed on the matrix. If op = 0, then circular upward shift operation is performed on columns c1 and c2 in the sub portion of range x1 to x2. If op = 1, then circular downward shift operation is performed on columns c1 and c2 in the sub portion of range x1 to x2. If op = 2, then perform reverse operation on the sub array of c1 and c2 in the range from x1 to x2. For each operation a sub-key is constructed and recorded in a key file.

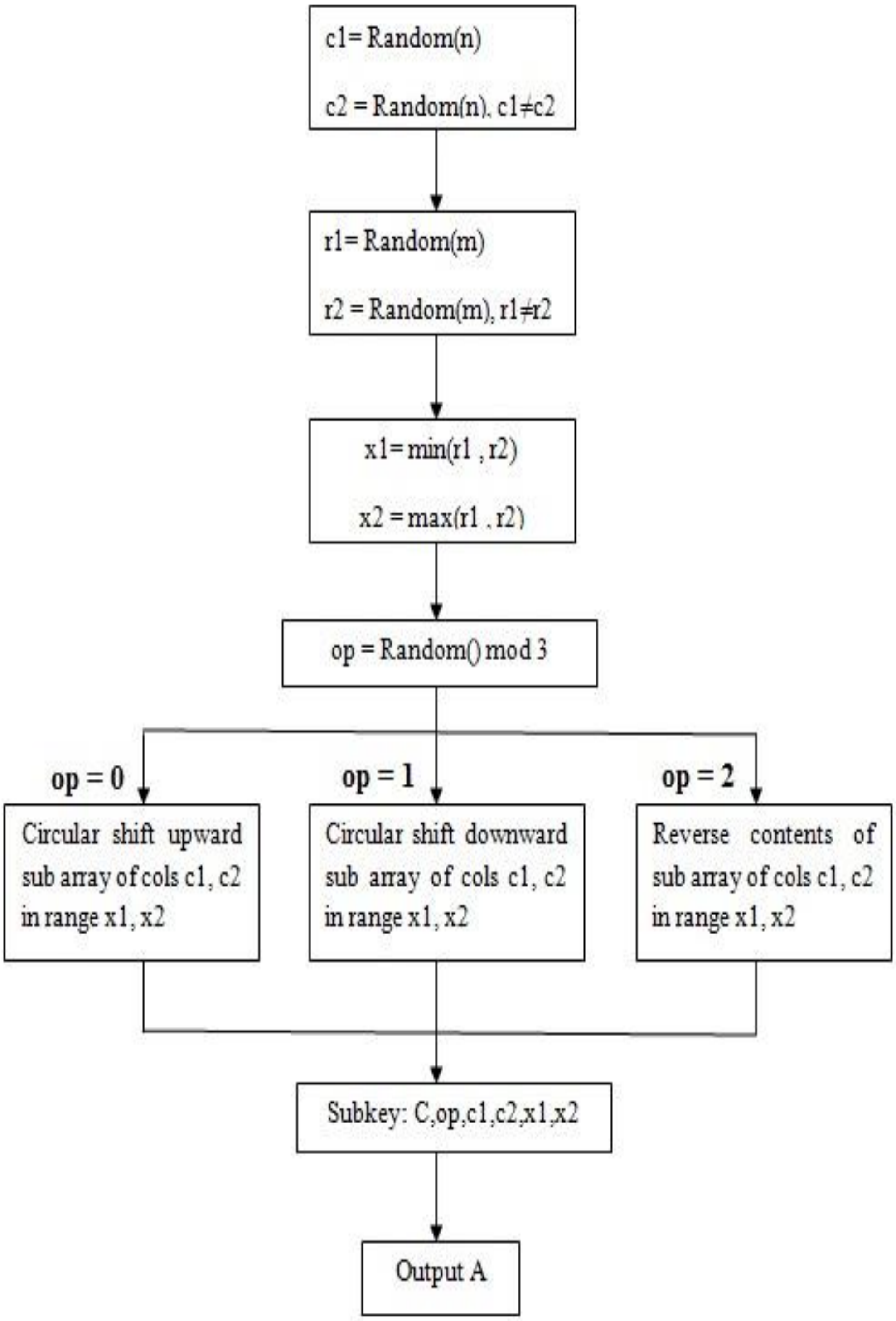

c1= Random(n)
c2 = Random(n), c1≠c2
r1= Random(m)
r2 = Random(m), r1≠r2
x1= min(r1 , r2)
x2 = max(r1 , r2)
op = Random() mod 3
op = 0
op = 1
op = 2
Circular shift upward sub array of cols c1, c2 in range x1, x2
Circular shift downward sub array of cols c1, c2 in range x1, x2
Reverse contents of sub array of cols c1, c2 in range x1, x2
Subkey: C,op,c1,c2,x1,x2
Output A

Fig4. *Column Transformation Process*

3. Decryption Process

Decryption is the reverse process of Encryption.Decryption process is done by reading the operations in the key file in reverse order and applying necessary operations on the matrix, which coverts cipher text to get plain text. The cipher text is converts into its equivalent binary form and then it's arranged into a matrix of same order in encryption as m and n. The Fig 5 and Fig 6 show how the decryption process is done.

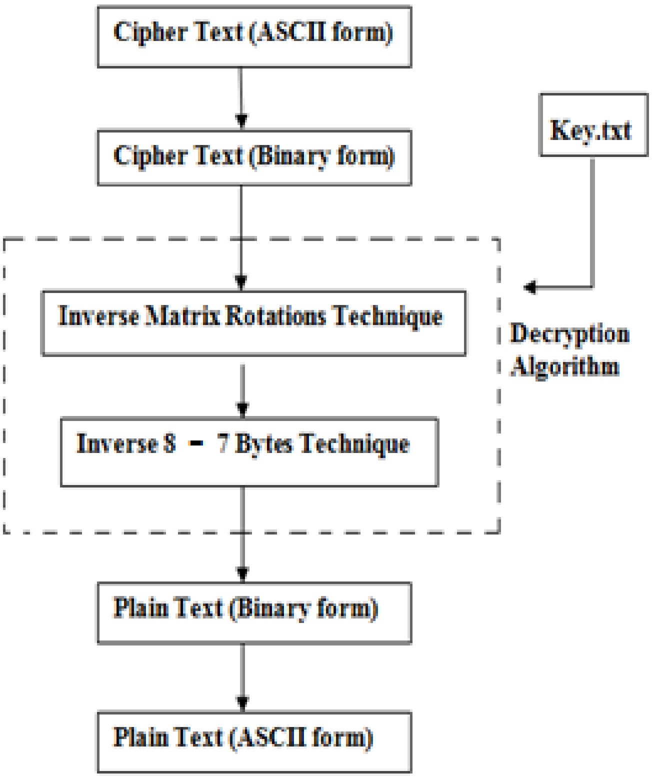


Fig5. *Block Diagram for Decryption Process*

Algorithm for Decryption process:

Steps for Decryption Process explained in brief as follows.

1. The key file is partitioned by the R (row) operations and C (column) operations. Each line in key file can be considered as one sub key.
2. The sub keys are decrypted one by one from last sub key to first sub key in key file.
3. For each sub key T, op, α1, α2, β1, β2 values are obtained, where T represents either R or C operation, 'op' represents either 0 or 1 or 2 (if T is R then 'op' is shiftleft (0) or shiftright (1) or shiftrevesre (2), if T is C then 'op' is shiftupward (0) or shiftdownward (1) or shiftrevesre (2)).Based on 'T' value either R or C operation is decoded which are given as A = InverseRowTrans(A) and A = InverseColTrans(A).
4. In InverseRowTrans(A) depending on the value of 'op' rotations are performed. For '0' RightShift, for '1' LeftShift and for '2' Reverse operations are performed on columns.
5. In InverseColTrans(A) depending on the value of 'op' rotations are performed. For '0' DownwardShift, for '1' UpwardShift and for '2' Reverse operations are performed on rows.
6. The steps from 2 to 5 are done until the key file is completed and end of process matrix A contains the required message in binary form.
7. The output binary message is given as input to "Inverse8-7 Bytes Technique" and finally gets required message is in binary form then converts into its ASCII equivalent form which is called as required original message (i.e. Plain Text).

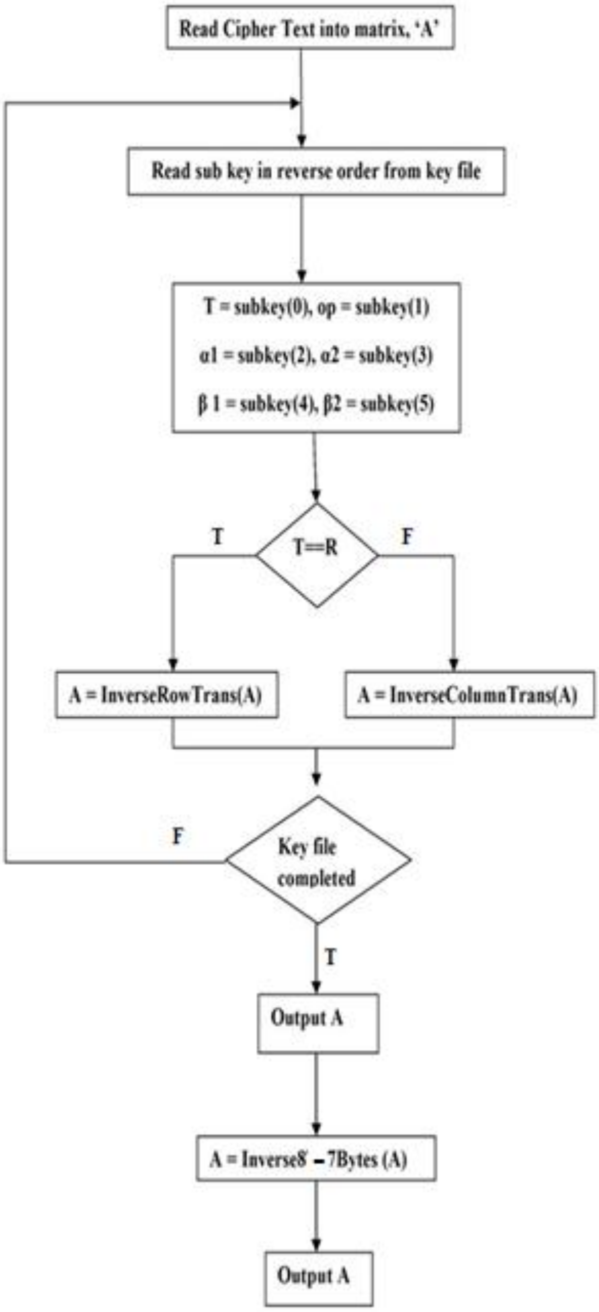

Read Cipher Text into matrix, 'A'
Read sub key in reverse order from key file
T = subkey(0), op = subkey(1)
α1 = subkey(2), α2 = subkey(3)
β 1 = subkey(4), β2 = subkey(5)
T
T==R
F
A = InverseRowTrans(A)
A = InverseColumnTrans(A)
F
Key file completed
T
Output A
A = Inverse8 – 7Bytes (A)
Output A

Fig6. *Block Diagram for BMR Decryption Technique*

4. Experimental Results

We have applied this proposed system on different sizes of data to encryption and decryption. Lets us consider a sample data and apply BMR technique as show in Fig 7 and Fig 8.

4.1. Encryption Process Example

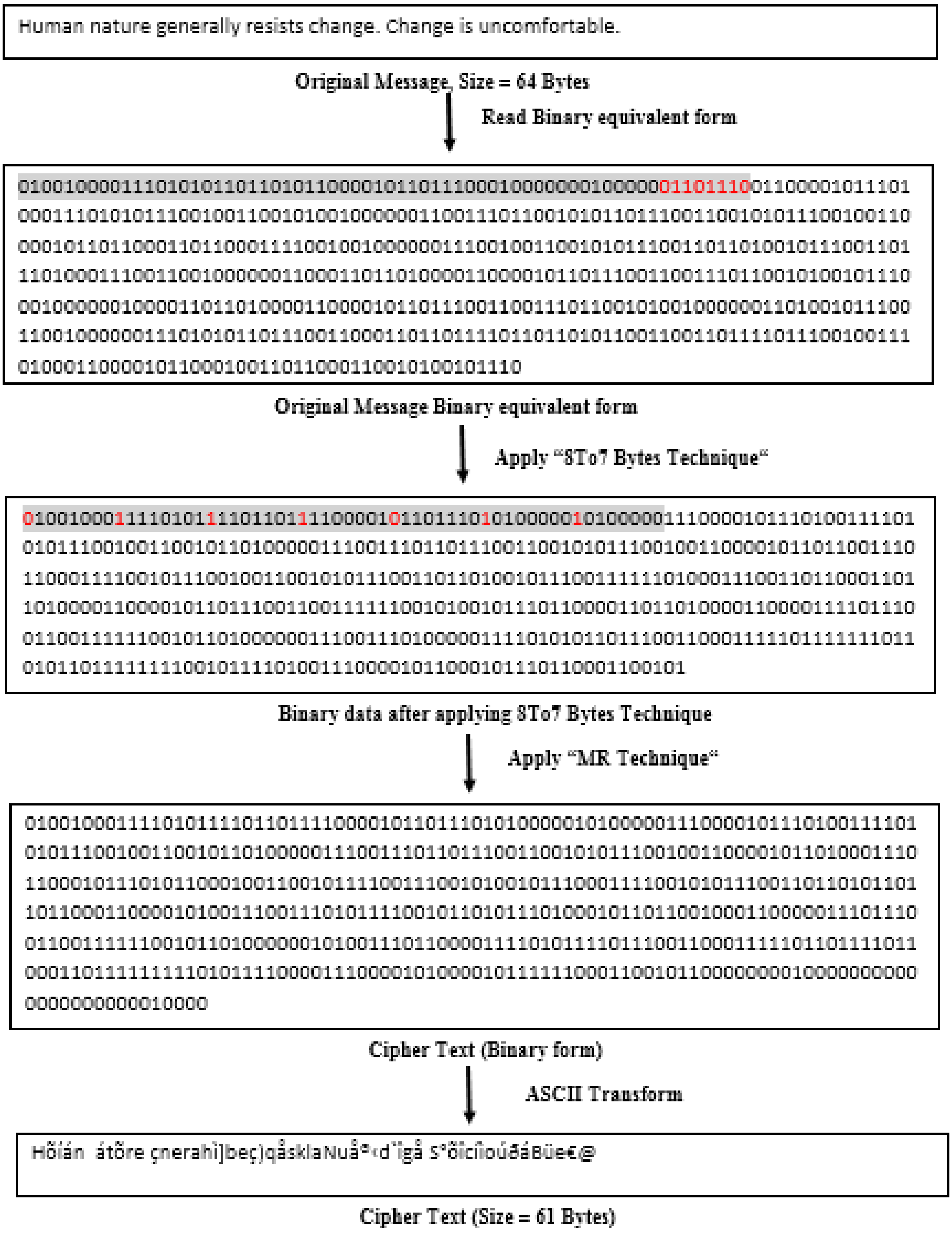
Let's us consider a sample data and apply BMR Encryption Technique on sample data which is show in Fig 7.

In this example, the Original message (size 64 Bytes) is converts into its equivalent binary form and then given as input to BMR technique. In BMR technique, the binary equivalent of Original message given as input to "8-7 Bytes" Technique, which converts each 8 bytes of text to 7 bytes of text. So, the advantage of this technique is it reduces the size of data. The output of "8-7 Bytes" technique is given as input to "Matrix Rotations (MR)" technique which changes the form of the data. Finally, the output of MR technique (in Binary form) is converts into its ASCII equivalent and called as Cipher Text (size 61 bytes).

4.2. Decryption Process Example

The Cipher text which is get from BMR encryption technique as output and given as input to BMR Decryption Technique. The total decryption process was show in Fig 8.

The Cipher text (size 61 Bytes) is converts into its equivalent binary form and then given as input to BMR Decryption algorithm. In BMR decryption technique, binary equivalent of cipher text is given as input to "Inverse Matrix Rotations (MR) technique". In "Inverse MR Technique", depending on sub key value of 'T' we can perform either "Inverse row transformation" or "Inverse Column transformation". Output of "Inverse MR Technique" is given as input to "Inverse 8-7 technique". In Inverse 8-7 technique, each 7 bytes of text becomes 8 bytes of text. Finally, we obtained Plain text in binary form as output from "Inverse MR Technique" and then convert it into its ASCII equivalent form and called as Plain Text or Original message (size 64 Bytes).



Human nature generally resists change. Change is uncomfortable.

Original Message, Size = 64 Bytes
Read Binary equivalent form

```
0100100001110101011011010110000101101110001000000010000001101110011000010111101
0001110101011100100110010100100000011001110110010101101110011001010111100100110
0001011011000110110001111001001000000111001001100101011100110110100101111001101
1101000111001100100000011000110110100001100001011011100110011101100101001011110
0010000000100001101101000011000010110111001100111011001010010000001101001011100
1100100000011101010110111001100011011011110110110101100110011011110111001001111
0100011000010110001001101100011001010010111 0
```

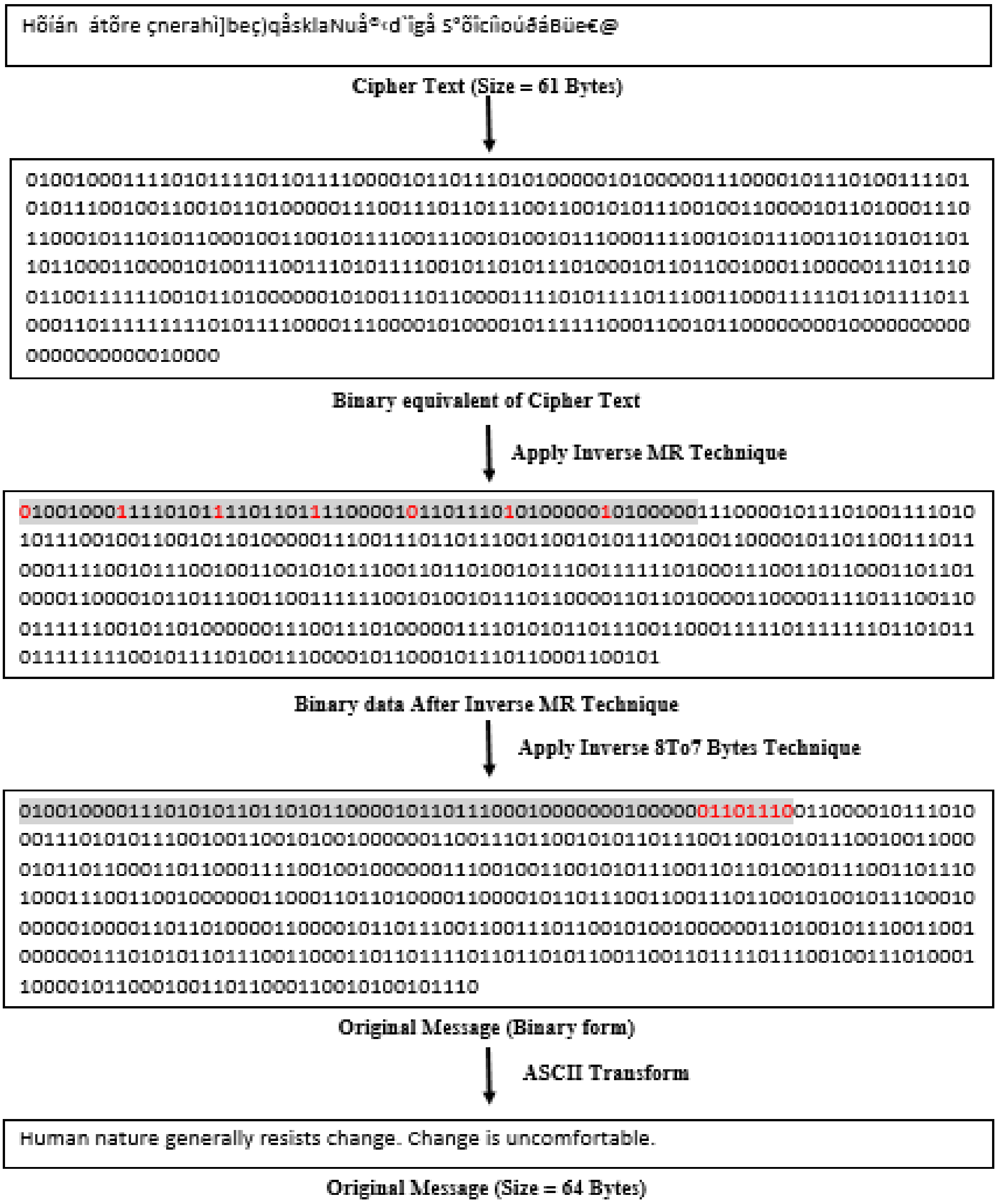
Original Message Binary equivalent form
Apply "8To7 Bytes Technique"

```
0100100011110101111011011110000101101110101000001010000011100001011101001111101
0101110010011001011010000011100111011011100110010101110010011000010110110011 10
1100011110010111001001100101011100110110100101110011111101000111001101100011 01
1010000110000101101110011001111110010100101110110000110110100001100001111101110
0110011111100101101000000111001110100000111101010110111001100011111011111111011
01011011111111100101111010011100001011000101110110001100101
```

```
0100100011110101111011011110000101101110101000001010000011100001011101001111101
0101110010011001011010000011100111011011100110010101110010011000010110100011 10
1100010111010110001001100101111001110010100101110001111001010111001101101011 01
1011000110000101001110011101011110010110101110100010110110010001100000111011 10
0110011111100101101000000101001110110000111101011110111001100011111011011110 11
0001101111111110101111000011100001010000101111110001100101100000000100000000 00
000000000010000
```

Hõïán átõre çnerahì]beç)qåsklaNuå*‹d¨igå s°õïcîioúðáBüe€@

Fig7. *Encryption process with example*



```
Hõíán  átõre çnerahì]beç)qåsklaNuå°‹d¨igå 5°õîcîioúđáBüe€@
```

```
01001000111101011110110111100001011011101010000010100000111000010111010011110 1
010111001001100101101000001110011101101110011001010111001001100001011010001110
110001011101011000100110010111100111001010010111000111100101011100110110101101
101100011000010100111001110101111001011010111010001011011001000110000011101110
011001111110010110100000001010011101100001111010111101110011000111110110111101 1
000110111111111010111100001110000101000010111111000110010110000000010000000000
000000000000010000
```

```
010010001111010111101101111100001011011101010000010100000111000010111010011110 10
101110010011001011010000011100111011011100110010101110010011000010110110011101 1
000111100101110010011001010111001101101001011100111111010001110011011000110110 1
000011000010110111001100111111001010010111011000011011010000110000111101110011 0
011111100101101000000111001110100000111101010110111001100011111011111111011010 11
0111111110010111101001110000101100010111011000110010 1
```

Binary data After Inverse MR Technique
Apply Inverse 8To7 Bytes Technique

```
010010000111010101101101011000010110111000100000001000000110111001100001011101 0
001110101011100100110010100100000011001110110010101101110011001010111001001100 0
010110110001101100011110010010000001110010011001010111001101101001011100110111 0
100011100110010000000110001101101000011000010110111001100111011001010010111000 10
000001000011011010000110000101101110011001110110010100100000011010010111001100 1
000000111010101101110011000110110111101101101011001100110111101110010011101000 1
1000010110001001101100011001010010111 0
```

```
Human nature generally resists change. Change is uncomfortable.
```

Fig8. *Decryption process with example*

4.3. Analysis

After encryption process the size of the data is reduced and after decryption process the size of the data is increases. The following Table 1 represented the various sizes of different data sets after encryption and decryption process.

Table1. *Represented the Various Sizes of Different Data Sets after Encryption and Decryption Process*

S.No	Data Size (in Bytes)	After Encryption (Cipher Text in Bytes)	After Decryption (Plain Text in Bytes)
1.	64	61	64
2.	128	113	128
3.	512	450	512
4.	1024	904	1024
5.	2048	1800	2048
6.	5120	4513	5120
7.	10240	8978	10240
8.	20480	17956	20480

5. Conclusion

In this paper we presented an implementation of BMR encryption algorithm. The main advantage of this algorithm was reduces size of data as well as change the form of data. The results show that the new theme has very fast encoding and safer with reduces the size of data. In our future work with this methodology was, BMR technique is applicable for 0-127 ASCII values only. So, we can extend our methodology to remaining ASCII values also.

References

[1] S.-J.-b. BAO Guan-jun and JI SHI-ming, (2002) "Magic cube transformation and its application in digital image encryption", Computer Applications, 11:22–25.

[2] X.Y.CHEN Tao, (2005) "Design and implementation of encryption algorithm based on n-dimension magic cube", Journal of Information, 2:13-14.

[3] C.C.LUO Pu, (2006) "An image encryption algorithm based on chaotic sequence", Journal of Information, 12.

[4] F.Y.LI Min, (2005) "A New class of digital image scrambling algorithm based on the method of queue transformation", Computer Engineering, 01(31):148-149.

[5] M. Kiran Kumar, S. Mukthyar Azam, "Efficient Digital Encryption Algorithm based on Matrix Scrambling Technique", IJNSA, Vol2, No.4, Oct-2010.

[6] Z.L.Z.X. feng and FAN Jiu-lun, (2007) "A digital image encryption algorithm based on chaotic sequences", Microelectronics & computer, 2.

[7] William Stallings "Cryptography and Network security", 3rd edition, Prentice-Hall Inc., 2005.

[8] William Stallings, "Cryptography and Network Security: Principles &Practices", second edition.