EFFICIENT DIGITAL ENCRYPTION ALGORITHM BASED ON MATRIX SCRAMBLING TECHNIQUE

{M. Kiran Kumar, S. Mukthyar Azam}[1], Shaik Rasool[2]

[1]Dept. of Computer Science & Engg, R.V.R & J.C College of Engg, Guntur, India
kirankumar.rvr@gmail.com

[1]Dept. of Research & Development, Synfosys, Hyderabad, India
mukthyar@live.in

[2]Dept. of Computer Science & Engg, S.C.E.T., Hyderabad, India
shaikrasool@live.in

ABSTRACT

This paper puts forward a safe mechanism of data transmission to tackle the security problem of information which is transmitted in Internet. We propose a new technique on matrix scrambling which is based on random function, shifting and reversing techniques of circular queue. We give statistical analysis, sequence random analysis, and sensitivity analysis to plaintext and key on the proposed scheme. The experimental results show that the new scheme has a very fast encryption speed and the key space is expanded and it can resist all kinds of cryptanalytic, statistical attacks, and especially, our new method can be also used to solve the problem that is easily exposed to chosen plaintext attack. We give our detailed report to this algorithm, and reveal the characteristic of this algorithm by utilizing an example.

KEYWORDS

Cipher text, Encryption, Decryption, Random Number Generator, Circular queue.

1. INTRODUCTION

With the rapid growth of Internet, global information tide expends the application of information network technology. It also brings about great economic and social benefit along with the extensive use of this technology. However, because Internet is an open system which faces to public, it must confront many safe problems. The problems include network attack, hacker intruding, interception and tampering of network information which lead huge threat to Internet [3]. Information security becomes a hot problem which is concerned by our society. Security of the information over the insecure mode of communication, Internet, has been an area of research for years. There have been several techniques developed for encryption/decryption of the information over the years.

This paper discusses a new technique of 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 scrambling technique. The encryption algorithm of magic cube projected by Yongwei et al. which is implemented in three-dimensional space [5][11][4]. The algorithm is complicated and difficult to understand. F. Y. LI Min. proposed queue transformation based digital image encryption algorithm [6], which works efficiently with low time complexity compared to Yongwei et al.. However, it still has some shortcomings. Firstly, the algorithm signifies some certain regularity. According to F. Y. LI Min, algorithm, assume we randomly select one element, (i, j), from the matrix, rows (columns) with row (column) ID smaller than i(j) cyclically shifts leftward (upward) with increasing steps, and rows (columns) with row (column) ID greater than i(j) cyclically shifts rightward (downward) with proceeding steps. This regularity makes the encryption output represent some certain periodicity with the increasing of steps, which make the algorithm vulnerable to attacks with ease of decryption. Secondly, one operation to a $n\times n$ matrix will cause $n(n-2i-1)/2$ cyclic shifts. At least n operations is needed for scrambling, which makes the time complexity of this algorithm be $O(n^3)$.

By enhancing this algorithm, in this paper, we propose an Efficient Digital Encryption Algorithm Based on Matrix Scrambling Technique which is based on random function, shifting and reversing techniques of circular queue, with efficient time complexity.

2. ENCRYPTION ALGORITHM

The plaintext chosen is arranged into a Bi-directional circular Queue data structure [10],In the matrix A of order $m \times n$. Input an integer parameter, w, as the count of operations, say, the time of transformation we made to matrix. To some certain degree, this parameter represents the intensity of the encryption. However, this does not necessarily mean that the greater w is, the more intensive the encryption is. Random() function is used to generate random positive integer, this integer value has to be converted into a binary equivalent [9]. Depending on the binary bit from least significant bit to most significant bit, the choice to select rows or columns is made. In case of row, two rows r1 and r2 are selected randomly from the matrix similarly two random values of columns c1, c2 are chosen to determine the range of rows on which transformation has to be performed. To perform transformation operations, a value is calculated from Random( ) mod3, circular left shift, circular right shift and reverse operations on the selected rows is performed depending on the value which may be 0,1 or 2 In case of column, two columns c1 and c2 are selected randomly from the matrix, two random values of rows r1,r2 are chosen to determine the range of columns on which transformation has to be performed.. Then to perform transformation operations similar to rows, a value is calculated from Random( ) mod 3, based on it circular upward shift, circular downward shift and reverse operations are performed similar as rows.

The entire process is repeated w number of times; if $w \leq$ length (binary sequence) then the process is performed only w number of times from LSB to MSB else the binary sequence is repeated from LSB to MSB until the w operations. In this manner, the entire matrix elements are transposed. For each operation performed, the operation should be recorded as a sub~key in a file, which becomes the key file. The key file should be maintained secret. The decryption process is done by reading the operations in the key file in reverse order and applying necessary operations on the matrix, which contains cipher text as to get plain text.

Here we make introduction to our encryption algorithm, which is given in figure 1. The function Random(n) generates random positive integers less than or equal to n and Random(m) generates random positive integers less than or equal to m, where m, n N and Random( ) be the random function which generates random positive integers. These random numbers could be implemented by technology of chaotic sequence, wavelet transformation, elliptic curve and so on [8][1][7][2].The entire process of encryption is summarized below:

1. The original plain text is stored in an $m \times n$ matrix A, where m N, n N.
2. Input an integer parameter w, which represents no. of rounds of operations to perform. Depending on w, w transformations are applied on the matrix. To some extent, this parameter represents the intensity of encryption. However, this does not necessarily mean, the greater w is, the more intensive encryption is.
3. Let i = Random(), hence 'i' is a random positive number which decides which transformation is applied either row transformation or column transformation. The value of i provides some sort of strength to the encryption. After choosing i , binary value of i is calculated and placed in a vector b, b = Binary Sequence(i). The good choice of i gives better binary sequence to perform transformations on the matrix.
4. Let k = Digit(b_t), where 't' is bit position, k value is either 0 or 1, which provides a way of deciding either to perform row or column transformation. If k = 0, then A = RowTrans(A) is performed i.e. row transformation operations are applied on the matrix as given below

(i). If k = 1, then A = ColumnTrans(A) i.e. column transformation operations are applied on the matrix as given below (ii).

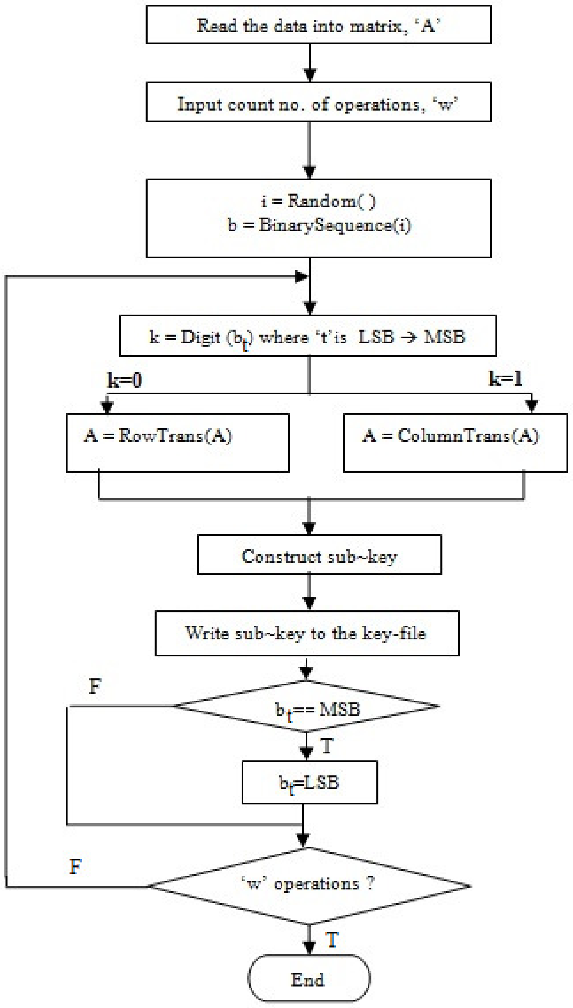

Read the data into matrix, 'A'
Input count no. of operations, 'w'
i = Random( )
b = BinarySequence(i)
k = Digit (b_t) where 't' is LSB → MSB
k=0
k=1
A = RowTrans(A)
A = ColumnTrans(A)
Construct sub~key
Write sub~key to the key-file
F
b_t== MSB
T
b_t=LSB
F
'w' operations ?
T
End

Figure 1: Encryption Algorithm

(i) We illustrate the process of row transformation operations i.e. A = RowTrans(A) in this section, the algorithm is shown in figure 2,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. The sub~key is 6 tuple and is given as follows, sub~key = (T, op, α1, α2, β1, β2) where

T = Transformation applied either row or column and takes values 'R' or 'C'

Op = 0 or 1 or 2 (Meaning of these values changes for row and column operations)

α1, α2 = Two random rows or columns selected depending on the transformation

β1, β2 = min and max values of range for two selected α1, α2

In case of row transformation sub~key is recorded as, R op/r1/r2/x1/x2.

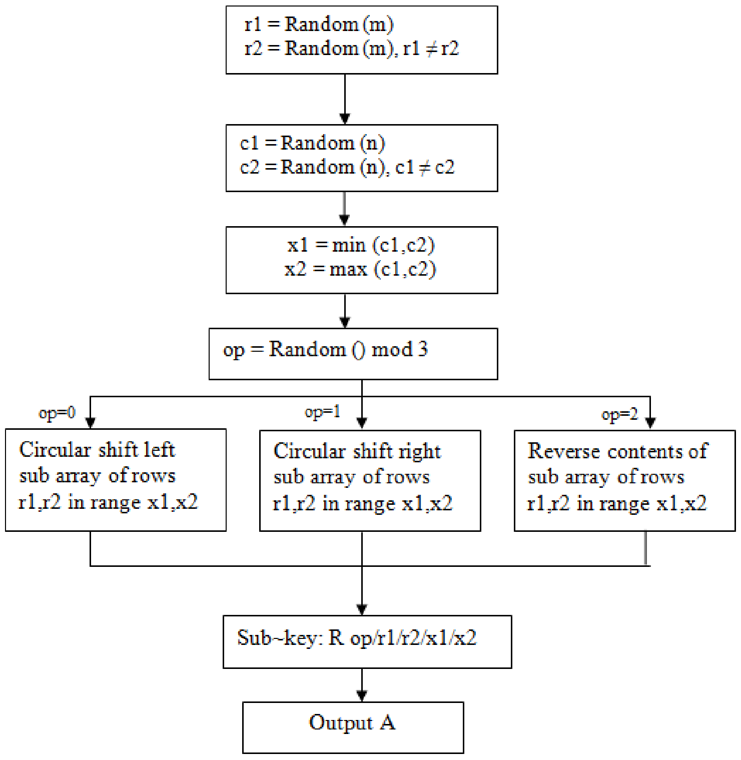

r1 = Random (m)
r2 = Random (m), r1 ≠ r2
c1 = Random (n)
c2 = Random (n), c1 ≠ c2
x1 = min (c1,c2)
x2 = max (c1,c2)
op = Random () mod 3
op=0
op=1
op=2
Circular shift left sub array of rows r1,r2 in range x1,x2
Circular shift right sub array of rows r1,r2 in range x1,x2
Reverse contents of sub array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

Figure 2: Process of Row Transformation

(ii) We illustrate the process of column transformation operations i.e. A = ColumnTrans(A) in this section, the algorithm is shown in figure 3, 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.

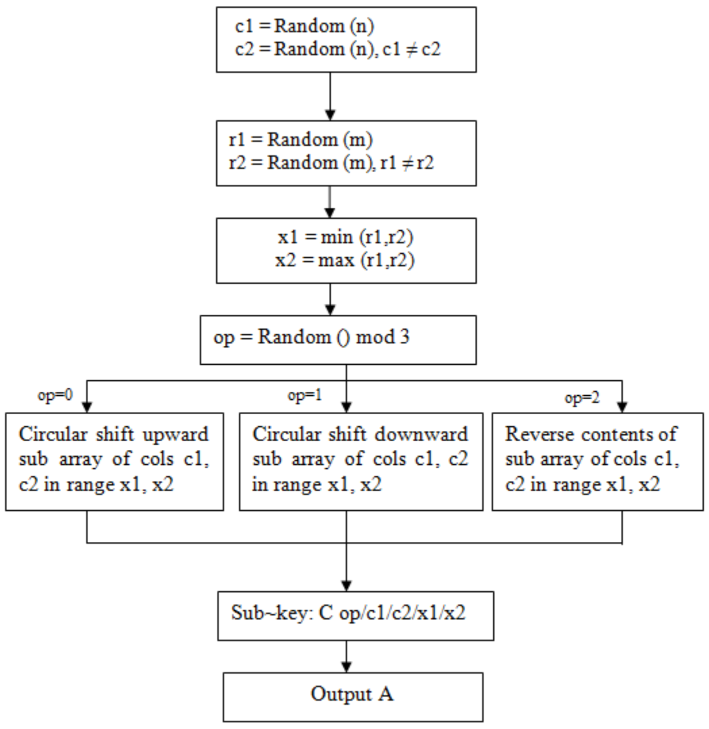

c1 = Random (n)
c2 = Random (n), c1 ≠ c2
r1 = Random (m)
r2 = Random (m), r1 ≠ r2
x1 = min (r1,r2)
x2 = max (r1,r2)
op = Random () mod 3
op=0
op=1
op=2
Circular shift upward sub array of cols c1, c2 in range x1, x2
Circular shift downward sub array of cols c1, c2 in range x1, x2
Reverse contents of sub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

Figure 3: Process of column transformation

5. Check whether binary sequence in vector b is completed. If it is completed, again start from first digit in the binary sequence of b (LSB→ MSB).
6. Repeat steps 4 and 5 for w no of times.
7. Record all the sub keys sequentially in a key~file.

2.1. Case Study of the Encryption Process

In this section, we show the detailed process of our encryption algorithm by an example. Let the data to be encrypted is taken as follows

1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20.

Let us set m = 4, n = 5, w = 7, i = 14, b = 01110. In vector b only 5 digits of the binary sequence is considered in the example. It depends on the user to consider how many bits to use without changing the actual bits of the value z. This provides irregularity and is used to increase the intensity of encryption. After the plain text is set into A, the layout of the matrix is shown in figure 4. After w operations and based on binary values in b, sub~keys recorded in the key~file is given in figure 5.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Figure 4: Layout of the plain text

R 1/2/1/2/4, C 0/4/1/0/3

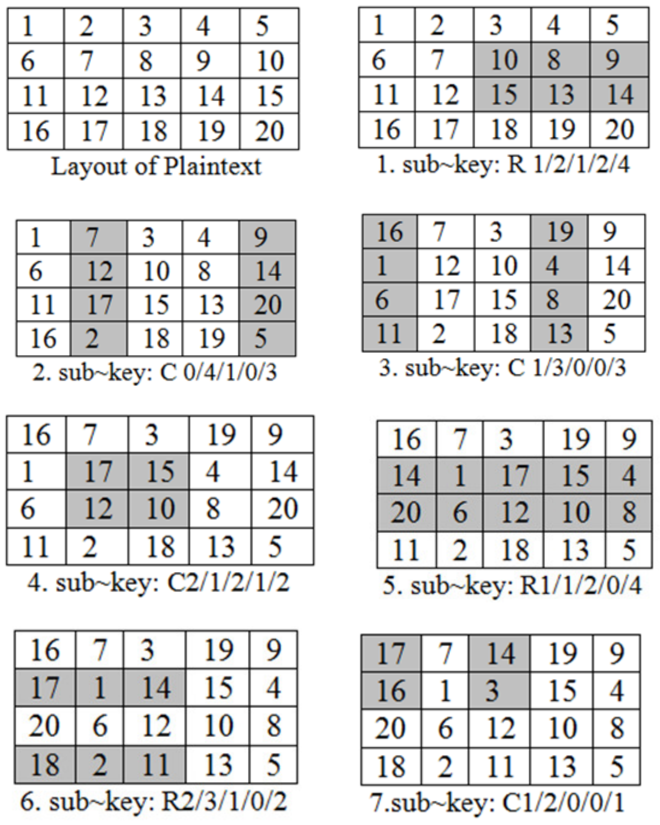
C 1/3/0/0/3, C 2/1/2/1/2

R 1/1/2/0/4, R 2/3/1/0/2

C 1/2/0/0/1

Figure 5: Content of the key~file

The process of disordering or scrambling the matrix elements based on sub~keys is explained in figure 6.



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Layout of Plaintext

1	2	3	4	5
6	7	10	8	9
11	12	15	13	14
16	17	18	19	20

1. sub~key: R 1/2/1/2/4

1	7	3	4	9
6	12	10	8	14
11	17	15	13	20
16	2	18	19	5

2. sub~key: C 0/4/1/0/3

16	7	3	19	9
1	12	10	4	14
6	17	15	8	20
11	2	18	13	5

3. sub~key: C 1/3/0/0/3

16	7	3	19	9
1	17	15	4	14
6	12	10	8	20
11	2	18	13	5

4. sub~key: C2/1/2/1/2

16	7	3	19	9
14	1	17	15	4
20	6	12	10	8
11	2	18	13	5

5. sub~key: R1/1/2/0/4

16	7	3	19	9
17	1	14	15	4
20	6	12	10	8
18	2	11	13	5

6. sub~key: R2/3/1/0/2

17	7	14	19	9
16	1	3	15	4
20	6	12	10	8
18	2	11	13	5

7.sub~key: C1/2/0/0/1

Figure 6: Process of obtaining cipher text using sub~keys in the key~file

The cipher text obtained after the entire process is given as

17,7,14,19,9,16,1,3,15,4,20,6,12,10,8,18,2,11,13,5

3. DECRYPTION ALGORITHM

The proposed algorithm is a kind of symmetric encryption algorithm, with decryption process is done by reversing the operations done in the encryption process. The cipher text is arranged into a matrix of the same order in the encryption as m and n. The algorithm for decryption is given in figure 7.

The steps of the decryption process explained in brief as follows

1. The key file is partitioned by the R (row) operations & C (col) operations. A sub key starts from R or C operation, ends at the start of another R or C operation or the end of the key file.
2. The sub keys are decrypted one by one from the last sub key to the first sub key.
3. For each sub key T, op, α1, α2, β1, β2 values are obtained whose terms are already explained in the encryption algorithm. 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' transpositions are performed. For 0 right shifts, for 1 left shift and for 2 reverse operation is performed on the columns.
5. In InverseColTrans(A) depending on the value of 'op' transpositions are performed. For 0 downward shift, for 1 upward shift and for 2 reverse operation is performed on the rows.
6. The process is done until the key file is completed and at the end of process matrix A contains the required original message i.e. plain text.

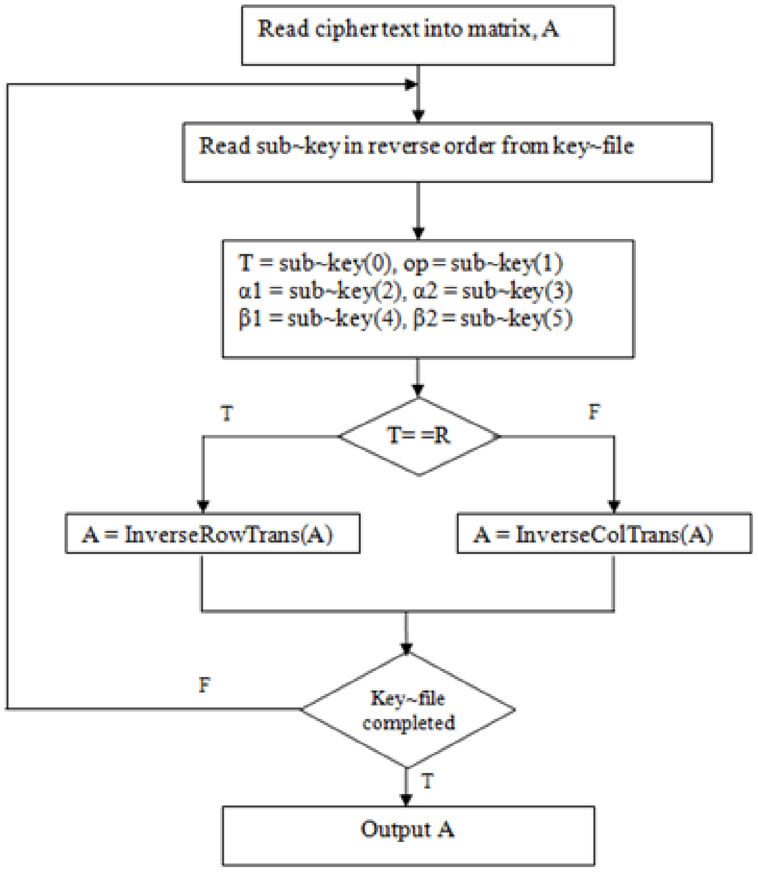

Read cipher text into matrix, A
Read sub~key in reverse order from key~file
T = sub~key(0), op = sub~key(1)
α1 = sub~key(2), α2 = sub~key(3)
β1 = sub~key(4), β2 = sub~key(5)
T
T= =R
F
A = InverseRowTrans(A)
A = InverseColTrans(A)
F
Key~file completed
T
Output A

Figure 7: Decryption algorithm

3.1. Case Study of the Decryption Process

In this section, we show the detailed process of our decryption algorithm with an example. Consider the cipher text obtained after encryption as 17, 7, 14, 19, 9, 16, 1, 3, 15, 4, 20, 6, 12, 10, 8, 18, 2, 11, 13, 5. The layout of the cipher text after placing in matrix A of order m and n is given in figure 10. The decryption is done by reading the sub keys in key file in reverse order and sub keys in reverse order is given in figure 9.

The steps of the decryption are given as follows in figure 10.

17	7	14	19	9
16	1	3	15	4
20	6	12	10	8
18	2	11	13	5

Figure 8: Layout of the cipher text

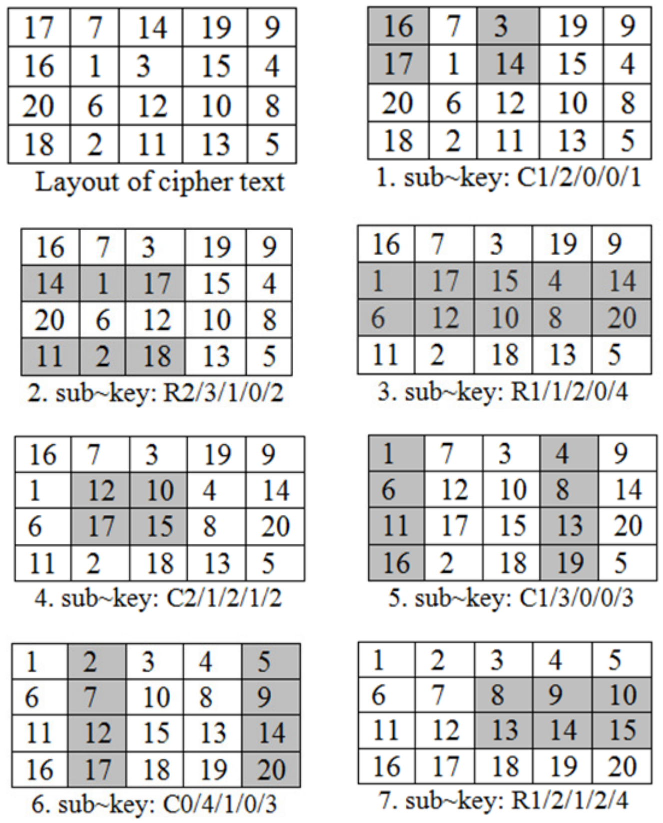
C 1/2/0/0/1, R 2/3/1/0/2

R 1/1/2/0/4, C 2/1/2/1/2

C 1/3/0/0/3, C 0/4/1/0/3

R 1/2/1/2/4

Figure 9: sub keys in reverse order



17	7	14	19	9
16	1	3	15	4
20	6	12	10	8
18	2	11	13	5

Layout of cipher text

16	7	3	19	9
17	1	14	15	4
20	6	12	10	8
18	2	11	13	5

1. sub~key: C1/2/0/0/1

16	7	3	19	9
14	1	17	15	4
20	6	12	10	8
11	2	18	13	5

2. sub~key: R2/3/1/0/2

16	7	3	19	9
1	17	15	4	14
6	12	10	8	20
11	2	18	13	5

3. sub~key: R1/1/2/0/4

16	7	3	19	9
1	12	10	4	14
6	17	15	8	20
11	2	18	13	5

4. sub~key: C2/1/2/1/2

1	7	3	4	9
6	12	10	8	14
11	17	15	13	20
16	2	18	19	5

5. sub~key: C1/3/0/0/3

1	2	3	4	5
6	7	10	8	9
11	12	15	13	14
16	17	18	19	20

6. sub~key: C0/4/1/0/3

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

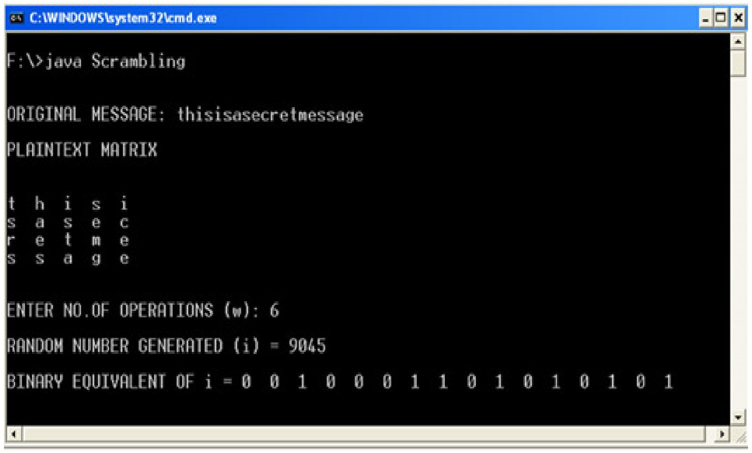
7. sub~key: R1/2/1/2/4

Figure 10: Process of Decryption

4. SIMULATION AND EXPERIMENTAL RESULTS

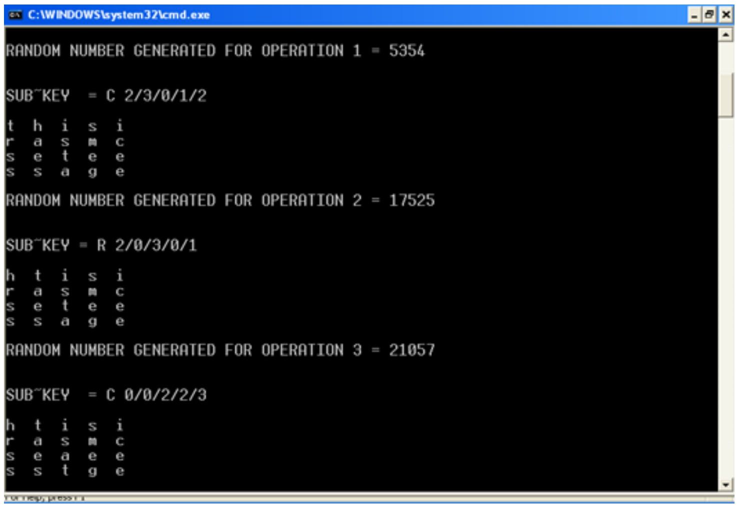
We present here results generated from practical implementation of our algorithm. A plain text of 20 characters is taken as the input to the algorithm. The various steps involved in the encryption algorithm are carried out. The Random number generators, Binary sequence generators are used in to full fill the need for generating random number and binary sequence in the algorithm. The input plaintext and the Random number and Binary sequence generated is shown in the figure 11.

As per the algorithm, binary sequence is processed and the row or column is selected. The operation to be performed is done based on the random number generated by the random number generator. The first three rounds of operations are shown in the figure 12 and the next three rounds in the figure 13. Each round operation's output is also shown in the figures. For every round operation a sub~key is generated and after every round of the operation the sub~key is stored in a separate text file which is kept secret.



```
C:\WINDOWS\system32\cmd.exe

F:\>java Scrambling

ORIGINAL MESSAGE: thisisasecretmessage

PLAINTEXT MATRIX

t  h  i  s  i
s  a  s  e  c
r  e  t  m  e
s  s  a  g  e

ENTER NO.OF OPERATIONS (w): 6

RANDOM NUMBER GENERATED (i) = 9045

BINARY EQUIVALENT OF i = 0  0  1  0  0  0  1  1  0  1  0  1  0  1  0  1
```

Figure 11: Plaintext and Generated Binary Sequence

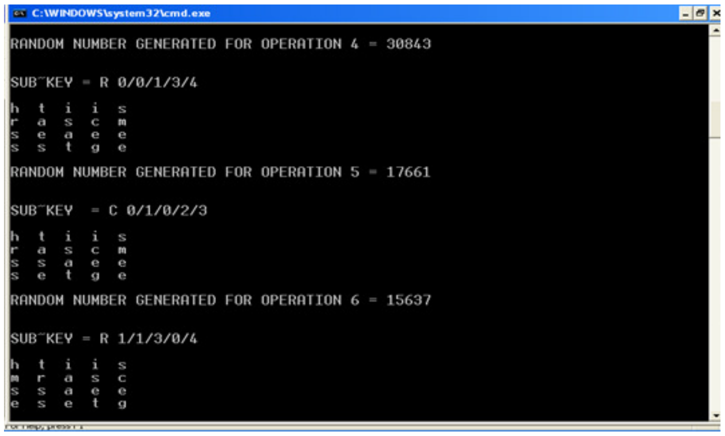


```
C:\WINDOWS\system32\cmd.exe

RANDOM NUMBER GENERATED FOR OPERATION 1 = 5354

SUB~KEY  = C 2/3/0/1/2

t  h  i  s  i
r  a  s  m  c
s  e  t  e  e
s  s  a  g  e

RANDOM NUMBER GENERATED FOR OPERATION 2 = 17525

SUB~KEY = R 2/0/3/0/1

h  t  i  s  i
r  a  s  m  c
s  e  t  e  e
s  s  a  g  e

RANDOM NUMBER GENERATED FOR OPERATION 3 = 21057

SUB~KEY  = C 0/0/2/2/3

h  t  i  s  i
r  a  s  m  c
s  e  a  e  e
s  s  t  g  e
```

Figure 12: Resultant matrix for the operations of 'w' (w=1, 2, 3)

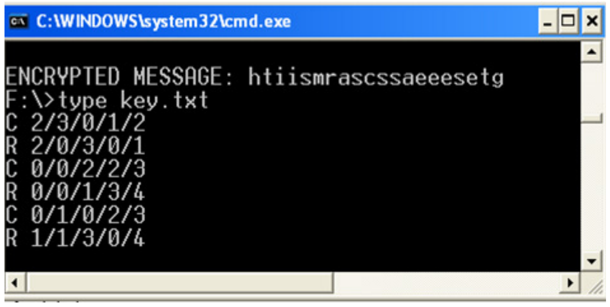
The contents of the key file are shown in figure 14. It can be seen that it contains the entire sub~key's generated after the completion of the encryption process. The number of sub~key's equal to the 'w' no of operations. This key file is sent along with the cipher text to the receiver. Depending on the key file each sub~key is used in the reverse order for the decryption process to get back the plaintext.

Now we present the graphs that represent the amount of text scrambled in the algorithm based on the experimental results. The total count of operations is 6 and for every round the number of transpositions performed can be seen in the figure 15. It can be observed form the figure that as with the every round of operation is complete the amount of transpositions goes on increasing resulting in the very good scrambled matrix. When the same plaintext is taken and when the encryption is performed again then the transpositions will be different and resultant scrambled matrix is also different. This can be observed in the figure 16.



```
RANDOM NUMBER GENERATED FOR OPERATION 4 = 30843

SUB~KEY = R 0/0/1/3/4

h   t   i   i   s
r   a   s   c   m
s   e   a   e   e
s   s   t   g   e
RANDOM NUMBER GENERATED FOR OPERATION 5 = 17661

SUB~KEY  = C 0/1/0/2/3

h   t   i   i   s
r   a   s   c   m
s   s   a   e   e
s   e   t   g   e
RANDOM NUMBER GENERATED FOR OPERATION 6 = 15637

SUB~KEY = R 1/1/3/0/4

h   t   i   i   s
m   r   a   s   c
s   s   a   e   e
e   s   e   t   g
```

Figure 13: Resultant matrix for the operations of ‘w’ (w=4, 5, 6)



```
ENCRYPTED MESSAGE: htiismrascssaeeesetg
F:\>type key.txt
C 2/3/0/1/2
R 2/0/3/0/1
C 0/0/2/2/3
R 0/0/1/3/4
C 0/1/0/2/3
R 1/1/3/0/4
```

Figure 14: Resultant Cipher text and Key~file

Thus it is hard that we will be getting same result when we process the same text again and again i.e. the scrambling is not same. It shows the efficiency of the algorithm and strong encryption that is not vulnerable to cryptanalysis.

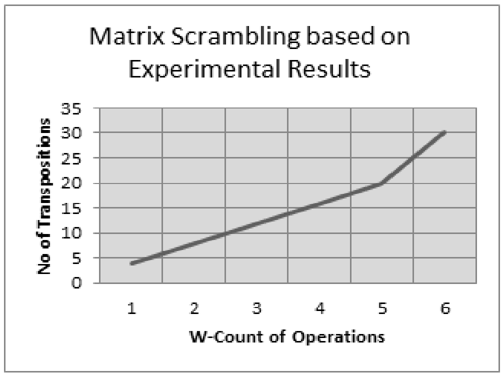


Figure 15: Analysis of Plaintext Scrambled for w=6

The value of parameter ‘w’ plays an important role in control the intensity of the encryption. ‘w’ should not be too small or too large. For a matrix of order m and n, e ≤ w ≤ 2e is good choice to scramble the matrix, where e = max(m, n). Under this condition, time complexity of our algorithm is O(e).

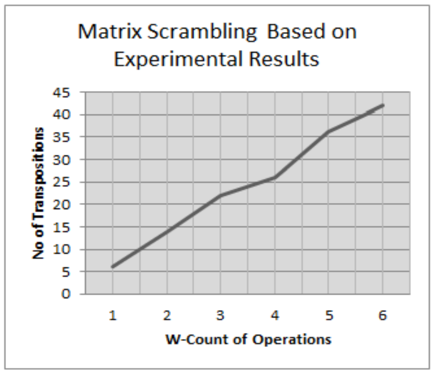


Figure 16: Analysis of Plaintext Scrambled for w=6

The second parameter 'i' also plays an important role in strengthening the intensity of encryption along with the parameter 'w'. The binary digit sequence of 'i' chosen also avoids regularity in the resultant cipher text. The good choice of 'i' gives better binary sequence, which is used to scramble the matrix in both directions row wise and column wise efficiently.

Our algorithm is based on magic rectangle, and hence it is easy to understand and easy to implement the proposed algorithm, which also has the following characteristics.

1. Running Speed: The algorithm belongs to classis shifting encryption algorithm and it uses only 3 kinds of linear array operations, circular horizontal shifting(left, right) , circular vertical shifting(up, down) and reverse operations. So it works efficiently with little system resources.
2. Simple Algorithm: Depending on shifting operations and reverse operation of the bi-directional circular queue, the algorithm could be implemented by software as well as hardware, which relax the limitation that most encryption algorithm needs to be implemented by hardware.
3. Strong Encryption Intensity: For an n × n matrix, the encryption intensity of the algorithm is n^w

5. CONCLUSION AND FUTURE WORK

In this approach the usage of random number selection firstly for generating binary sequence and, secondly for row (column) and column (row) selections, thirdly for selecting the operations for scrambling, avoids the regularity in the resultant cipher text which is transformed from plain text matrix; and hence improves the difficulty for decrypting.

The algorithm can be applied to text encryption, image encryption, and multimedia encryption and so on. In future work we will focus also on UMTS and GSM wireless transmission systems. Consequently, we can expect rapid growth in this area. In particular, with respect to real time applications, it will be interesting to see if the entertainment and telecommunication industries will make extensive use of the new standards MPEG IPMP and JPSEC. From the more research oriented perspective, the integration and interoperability of different multimedia security techniques poses a huge amount of open questions. Therefore, more research can be done in this field.

REFERENCES

[1] W. Y. YE Yongwei and YANG Qinghua, (2003) "Magic cube encryption for digital image using chaotic sequence", Journal of Zhejiang University of Technology, 31(2):173–176.

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

[3] Kui-He Yang and Shi-Jin Niu, "Data Safe Transmission Mechanism Based on Integrated Encryption Algorithm", Staffordshire University.

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

[5] 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.

[6] 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.

[7] Z. X. L. Z. G. Chuan, (2006) "A digital image encryption algorithm based on chaotic sequences", Computer Engineering and Applications, 19.

[8] G. J. HUANG Xiaosheng, (2007) "Image encryption algorithm based on compound chaotic sequence and wavelet transform", Computer Engineering, 14.

[9] M. Morris Mano, (2001) "Digital Design", Third Edition, Prentice Hall, pages (5-30).

[10] W. W. Yan Weimin, (1992) "Data Structure", Tsinghua University Press, Beijing.

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

Authors

Macharla Kiran Kumar[1] received the Bachelor of Technology in Computer Science & Engineering from Acharya Nagarjuna University, Guntur, India in 2004, and Master of Technology in Computer Science & Engineering from Acharya Nagarjuna University, Guntur, India in 2009. He is working as a Lecturer at the Department of Computer Science in R.V.R & J.C College of Engg, Guntur, India. His main research interests are Data mining, Information Security, Biometrics, web Information Retrieval, Image Processing, Discrete Structures, and Network Security.

S. Mukthyar Azam[1] received the Bachelor of Technology in Information Technology from Jawaharlal Nehru Technological University, Hyderabad, India in 2006, and Master of Technology in Computer Science & Engineering from Jawaharlal Nehru Technological University, Kakinada, India in 2009.He did Diploma in Embedded Systems Design from Centre for Development of Advanced Computing (Scientific Society of Department of IT, Ministry of Communications and IT, Govt. of India), Noida, India. He is working as a Software Engineer at Research & Development Lab in Synfosys, Hyderabad, India. His research interests are information security, Network access control, Intrusion detection and prevention systems, vulnerable access Management, programming methodologies.

Shaik Rasool[2] received the Bachelor of Technology in Computer Science & Engineering from Jawaharlal Nehru Technological University, Hyderabad, India in 2008. He is currently pursuing Master of Technology in Computer Science & Engineering from S.C.E.T. and also working as Assistant Professor at the Department of Computer Science & Engineering in S.C.E.T., Hyderabad, India. His main research interest includes Data mining, Network Security, Biometrics, Information Security, Intrusion Detection, Programming Language and security and Artificial Intelligence.