A Modified Feistel Cipher Involving a Key as a Multiplicant on Both the Sides of the Plaintext Matrix and Supplemented with Mixing, Permutation, and Modular Arithmetic Addition

1V.U.K. Sastry, 2K. Anup Kumar

1Director School of Computer Science and Informatics, Dean(R & D), Dean (Admin), Department of Computer Science and Engineering, Sreenidhi Institute of Science and Technology, Ghatkesar, Hyderabad, 501301, Andhra Pradesh, India.
2Associate Professor, Department of Computer Science and Engineering, Sreenidhi Institute of Science and Technology, Ghatkesar, Hyderabad, 501301, Andhra Pradesh, India.
E-mail: 1vuksastry@rediffmail.com and 2k_anupkumar@yahoo.com

Abstract: In this investigation, we have generalized the classical Feistel cipher by representing the plaintext in the form of a matrix instead of a binary string used in the case of classical Feistel cipher. In this, the plaintext matrix is divided into two matrices and one of these two is multiplied with the key matrix on both the sides. In the iteration process, involved in this cipher, we have included a pair of functions namely Mix ( ) and Permute ( ), and also utilized modular arithmetic addition. All these features are expected to strengthen the cipher. The avalanche effect and the cryptanalysis clearly show that the cipher is a potential one.

Key words: Encryption, Decryption, Key matrix, Mix, Permute and Modular Arithmetic Addition.

1. Introduction

In a recent investigation [1], we have developed a block cipher by generalizing the classical Feistel cipher wherein we have considered a plaintext which can be represented in the form of a pair of matrices instead of a pair of binary strings that was used in the case of classical Feistel cipher. In this development, we have used a key on both the sides of a portion of the plaintext matrix, and made use of iteration. In the iteration process we have included three additional features, namely, mixing, permutation and XOR operation. In this, we have seen that the strength of the cipher enhances quite significantly as all the three features, involved in the iteration process thoroughly modify the plaintext before it becomes the cipher text. The avalanche effect and the cryptanalysis discussed in this analysis effectively indicate that the cipher is a strong one.

In the present paper, our objective is to modify the aforementioned block cipher by including modular arithmetic addition instead of the XOR operation, used in the previous analysis.

The basic equations governing the encryption and the decryption of this cipher are given by

\[ P_i = Q_{i-1}, \]
\[ Q_i = (P_{i-1} + (KQ_{i-1}K)) \mod N \]

\[ Q_{i-1} = P_i, \]
\[ P_{i-1} = (Q_i - (KP_iK)) \mod N \]

where, \( P_i \) and \( Q_i \) are the plaintext matrices at the \( i \)th stage of the iteration, \( K \) the key matrix and \( N \) is a positive integer chosen appropriately. Here \( n \) denotes the number of iterations.

In the development of this cipher, we have utilized the functions Mix ( ) and Permute ( ), utilized in the earlier analysis [1]. The present analysis can be considered as a development over the block cipher developed by Sastry et.al [2] in which we have included the additional features namely Mix ( ) and Permute ( ).

In what follows, we present the plan of the paper. In section 2, we discuss the development of the cipher and present the flowcharts and the algorithms describing the cipher. Section 3 is devoted to an illustration of the cipher, and in this we have determined the avalanche effect. We have examined the cryptanalysis in section 4. Finally in
section 5, we have given the details of the computations and arrived at the conclusions.

2. Development of the cipher

Consider a plaintext $P$ consisting of $2m^2$ characters. On employing the EBCDIC code, the plaintext can be written in the form of a pair of square matrices $P_0$ and $Q_0$, wherein, each one is of size $m$. Let us consider a key matrix $K$, whose size is $mxm$.

In the matrices $P_i$ and $Q_i$, all the numbers are lying in the interval $[0\text{-}255]$. Here $F = K Q_{i-1} K$. In this analysis, we have taken $n=16$.

In this cipher, the encryption and the decryption are governed by the relations (1.1) and (1.2) respectively. In what follows we present the flow charts and the algorithms describing the encryption and the decryption processes.

In the flowchart, for the sake of elegance, we have written the functions Mix () and Permute (), arising in encryption, as $M ()$ and $\Phi ()$ respectively. The reverse processes represented by IMix () and IPermute (), arising in decryption, are denoted by IM () and I$\Phi ()$.

Fig 1. The process of Encryption

Fig 2. The process of Decryption
The processes of encryption and decryption depicted in the flow charts are described by the algorithms given below.

**Algorithm for Encryption**

1. Read P, K, n and N.
2. \( P_0 = \) Left half of P.
   \( Q_0 = \) Right half of P.
3. for \( i = 1 \) to \( n \) begin
   \( P_i = Q_{i-1} \)
   \( F = ( K Q_{i-1} K ) \mod N \)
   \( Q_i = ( P_{i-1} + F ) \mod N \)
   \( P_i = M ( P_i ) \)
   \( Q_i = \Phi ( Q_i ) \)
end
4. \( C = P_n Q_n \) /* || represents concatenation */
5. Write(C)

**Algorithm for Decryption**

1. Read C, K, n and N.
2. \( P_n = \) Left half of C
   \( Q_n = \) Right half of C
3. for \( i = n \) to \( 1 \) begin
   \( P_i = I\Phi ( P_i ) \)
   \( P_i = IM ( P_i ) \)
   \( Q_i = I\Phi ( Q_i ) \)
   \( Q_i = IM ( Q_i ) \)
   \( P_{i-1} = ( Q_i - F ) \mod N \)
end
4. \( P = P_0 || Q_0 \) /* || represents concatenation */
5. Write (P)

Let us now see how the functions (1) Mix ( ) and (2) Permute ( ) can be developed.

As we proceed with the iteration process, we come across a pair of square matrices of size \( m \) at each stage of the iteration process. Let us suppose that a square matrix of size \( m \) denoted by \( T \) can be written in the form

\[
T = [ T_{ij} ], \quad i = 1 \text{ to } m, \quad j = 1 \text{ to } m.
\]

On representing each element of the matrix in its binary form we get a matrix of size \( m \times 8m \). Considering 8 binary bits, at each instance, in a column wise manner and finding its decimal equivalent, we can represent the \( m \times 8m \) matrix as a square matrix of size \( m \), by arranging the decimal numbers in a row wise manner. This has led to a through mixing of the binary bits related to the key and the plaintext. For a detailed discussion of this mixing process we may refer to [1].

Now let us consider the development of the function Permute ( ).
Let \( K \) be a square matrix of size \( m \).
We arrange the \( m^2 \) decimal numbers of \( K \) in ascending order, and find the position of each number by looking at the elements of the matrix in a row wise manner.
This representation can be written in the form of a table given by.

<table>
<thead>
<tr>
<th>L_1</th>
<th>L_2</th>
<th>L_3</th>
<th>. . .</th>
<th>. . .</th>
<th>m^2</th>
</tr>
</thead>
</table>

Table 1. Correspondence between the elements of the matrix and their positions.
Here, \( L_1 \), \( L_2 \) ...... \( L_{m^2} \) are the numbers denoting the positions of the elements in \( K \).

Consider the plaintext, represented in the form of a square matrix \( T \) of size \( m \). On converting each element into its binary form, \( T \) can be written as a matrix of size \( m \times 8m \). In this matrix, we go on interchanging the \( i \)th column with \( L_i \)th column (without any unwanted interchanges) till we exhaust all the columns. Then, we convert the resulting plaintext matrix into its decimal form. Thus we get back an \( m \times m \) matrix. This is the process of permutation. For a clear idea concerned to this permutation process, we may refer to [1].

3. Illustration of Cipher

Consider the plaintext given below.
Dear friend, do not have any worry. I agree, we are thrown into side track. People call us as terrorists. Are we the terrorists? or are they the terrorists? Their fathers and forefathers earned millions and millions of rupees, and they became highly affluent and influential. They joined politics. They transformed the whole country with their outlook and to their advantage at all instances. We are no way inferior to them in ethical values. (3.1)
Let us focus our attention on the first 128 characters of the plaintext. This is given by Dear friend, do not have any worry. I agree, we are thrown into the side track. People call us as terrorists. Are we the terrorists? (3.2)

On applying the EBCIDIC code we get,

\[
P_0 = \begin{bmatrix}
68 & 101 & 97 & 114 & 32 & 102 & 114 & 105 & 101 & 110 & 100 & 44 & 32 & 100 & 111 & 32 \\
\end{bmatrix}
\]

\[Q_0 = \begin{bmatrix}
101 & 110 & 100 & 44 & 32 & 100 & 111 & 32 \\
32 & 97 & 110 & 121 & 32 & 119 & 111 & 114 \\
114 & 101 & 101 & 44 & 32 & 119 & 101 & 32 \\
119 & 110 & 32 & 105 & 110 & 116 & 111 & 32 \\
99 & 107 & 46 & 32 & 80 & 101 & 111 & 112 \\
117 & 115 & 32 & 97 & 115 & 32 & 116 & 101 \\
46 & 32 & 65 & 114 & 101 & 32 & 119 & 101 \\
114 & 111 & 114 & 105 & 115 & 116 & 115 & 63
\end{bmatrix}
\]

Let us take the Key in the form

\[
K = \begin{bmatrix}
122 & 13 & 150 & 27 & 53 & 200 & 15 & 86 \\
33 & 2 & 176 & 111 & 55 & 170 & 110 & 115 \\
67 & 38 & 17 & 22 & 83 & 37 & 151 & 60 \\
11 & 77 & 90 & 120 & 116 & 153 & 173 & 154 \\
131 & 66 & 88 & 51 & 75 & 04 & 69 & 186 \\
130 & 182 & 156 & 199 & 162 & 178 & 133 & 140 \\
184 & 176 & 139 & 182 & 166 & 170 & 215 & 240 \\
165 & 189 & 137 & 170 & 182 & 138 & 111 & 217
\end{bmatrix}
\]
On using (3.4) - (3.6) and the functions Mix( ) and Permute( ), we adopt the encryption algorithm, mentioned in section 2, and obtain the cipher text given by

\[
C = \begin{bmatrix}
146 & 222 & 155 & 198 & 118 & 240 & 179 & 12 & 79 & 15 & 108 & 194 & 89 & 28 & 146 & 197 \\
246 & 118 & 206 & 121 & 35 & 235 & 250 & 182 & 111 & 165 & 98 & 210 & 55 & 34 & 172 & 216 \\
128 & 179 & 24 & 218 & 73 & 122 & 144 & 52 & 200 & 12 & 59 & 199 & 114 & 97 & 194 & 192 \\
121 & 250 & 228 & 159 & 229 & 139 & 126 & 115 & 52 & 200 & 58 & 43 & 236 & 16 & 147 & 88 \\
\end{bmatrix} \tag{3.7}
\]

On applying the decryption algorithm with the inputs (3.6), (3.7) and the functions IMix( ) and IPermute( ), we get back the original plaintext given by (3.3).

Let us now examine the avalanche effect which indicates the strength of the cipher in a qualitative manner. To carry out this one, we change the first character of the plaintext (3.2) from D to E. As the EBCIDIC codes of these characters are 68 and 69, we have a change of one bit in the plaintext. On using the modified plaintext and the key (3.6), we apply the encryption algorithm, given in section 2, and obtain the new cipher text \(C\) in the form

\[
C = \begin{bmatrix}
73 & 223 & 177 & 125 & 217 & 66 & 231 & 69 & 212 & 228 & 114 & 158 & 192 & 155 & 95 & 219 \\
163 & 236 & 107 & 44 & 238 & 137 & 38 & 133 & 187 & 218 & 88 & 128 & 123 & 166 & 217 & 146 \\
196 & 189 & 183 & 143 & 57 & 229 & 243 & 114 & 103 & 82 & 190 & 153 & 3 & 83 & 24 & 199 \\
29 & 190 & 100 & 44 & 57 & 41 & 43 & 69 & 185 & 109 & 189 & 49 & 164 & 240 & 246 & 122 \\
220 & 157 & 188 & 7 & 106 & 88 & 201 & 219 & 71 & 157 & 182 & 54 & 189 & 252 & 223 & 0 \\
67 & 249 & 202 & 140 & 172 & 153 & 144 & 26 & 141 & 203 & 45 & 91 & 155 & 195 & 189 & 185 \\
228 & 143 & 166 & 179 & 114 & 43 & 99 & 127 & 55 & 38 & 89 & 140 & 109 & 95 & 74 & 197 
\end{bmatrix} \tag{3.8}
\]

On comparing (3.7) and (3.8) in their binary form, we find that they differ by 507 bits (out of 1024 bits). This shows that, the cipher is expected to be a strong one.
Let us now consider a one bit change in the key \(K\) given by (3.6). To this end, we change the 5th row 4th column element 222 to 223. This leads to a change of one binary bit the key. Now on using the original plaintext (3.3) and the modified key, as we have obtained here, we apply the encryption algorithm and obtain the cipher text \(C\) in the form

\[
C = \begin{bmatrix}
206 & 121 & 35 & 235 & 227 & 114 & 179 & 125 & 46 & 114 & 223 & 253 & 37 & 138 & 179 & 76 \\
131 & 183 & 190 & 120 & 128 & 179 & 17 & 60 & 61 & 164 & 115 & 209 & 220 & 152 & 112 & 233 \\
158 & 96 & 61 & 221 & 59 & 236 & 200 & 115 & 150 & 239 & 114 & 80 & 118 & 247 & 201 & 16 \\
22 & 96 & 0 & 185 & 183 & 47 & 203 & 75 & 24 & 118 & 210 & 182 & 3 & 86 & 250 & 155 \\
\end{bmatrix}
\]

(3.9)

After converting (3.7) and (3.9) into their binary form and comparing them, we notice that they differ by 510 bits (out of 1024 bits). This result also firmly indicates that the cipher is a potential one.

### 4. Cryptanalysis

The different types of Cryptanalytic attacks that are seen in the literature of cryptography [3] are

1. Cipher text only (Brute Force) attack.
2. Known Plaintext attack.
3. Chosen Plaintext attack.
4. Chosen Cipher text attack.

In all these cryptanalytic attacks, the cipher text and the encryption algorithm are available to the attacker. Generally an algorithm is designed to withstand the first two attacks [3].

Let us now focus our attention on the brute force attack. In this cipher, as the key \(K\) is consisting of 512 bits, the size of the key space is

\[
2^{512} = (2^{10})^{51.2} = (10)^{51.2}.
\]

Let us suppose that, the time required for the computation of the algorithm with one value of the key is \(10^7\) seconds. Then the total time required for the execution with all possible keys in the key space is

\[
\frac{(10)^{51.6} \times 10^7}{365 \times 24 \times 60 \times 60} \approx (10)^{138.6} \times 3.12 \times 10^{15} = 3.12 \times (10)^{138.6} \text{ years}
\]

As this number is very large, it is not at all possible to break the cipher with this attack.

Now let us examine the known plaintext attack. In order to carry out this one, we know any as many pairs of plaintext and cipher text as we require for this purpose. In this cipher, as we have an iteration process which includes multiplication with the key on both the sides of a portion of the plaintext and the functions mix (\(\) ) and permute (\(\) ) for thoroughly shuffling the results, at each stage of the iteration process. It is totally impossible to determine the key or a function of the key which is required for breaking the cipher. Hence, this attack is an utter failure.

In the cases 3 and 4, that is, in the chosen plaintext attack and in the chosen cipher text attack, the process of attack is expected to be very cumbersome, as the plaintext in the cipher is under going several operations. Hence we say that we do not find any scope for these attacks.

From the above discussion, we conclude that the cipher is a strong one from the cryptanalytic point of view.

### 5. Computations and Conclusions

In this paper, we have investigated a modified Feistel cipher which includes iteration and modular
arithmetic addition. In order to achieve through confusion and diffusion, this cipher involves a pair of functions, namely, mix ( ) and permute ( ) for creating diffusion and confusion. As these two characteristic features have mixed and scattered the binary bits of the plaintext and the key, at various stages of the iteration, the cipher is found to be a strong one.

The programs required for encryption and decryption are written in C Language.

In order to carry out the encryption of the entire plaintext, we divide the plaintext (3.1) into 4 blocks, wherein each one of the first three blocks contain 128 characters. As the last block is containing only 100 characters, we have included 28 blanks to make it a complete block of 128 characters. On adopting the encryption algorithm, with the necessary inputs, we have obtained the corresponding cipher text given below.

As the avalanche effect supports the cipher, and as the cryptanalysis clearly indicates that this cipher is a strong one, we conclude that this cipher is quite comparable with any other cipher in cryptography. It is interesting to note that this cipher can be made use of for the security of any amount of information.
5. References

[1] V. U. K. Sastry, K. Anup Kumar, “A Modified Feistel Cipher involving a key as a multiplicant on both the sides of the plaintext matrix and supplemented with Mixing, Permutation and XOR Operations” (Sent for Publication)


Authors profile:

**Dr. V. U. K. Sastry** is presently working as Professor in the Dept. of Computer Science and Engineering (CSE), Director (SCSI), Dean (R & D), SreeNidhi Institute of Science and Technology (SNIST), Hyderabad, India. He was Formerly Professor in IIT, Kharagpur, India and Worked in IIT, Kharagpur during 1963 – 1998. He guided 12 PhDs, and published more than 40 research papers in various international journals. His research interests are Network Security & Cryptography, Image Processing, Data Mining and Genetic Algorithms.

**Mr. K. Anup Kumar** is presently working as an Associate Professor in the Department of Computer Science and Engineering, SNIST, Hyderabad India. He obtained his B.Tech (CSE) degree from JNTU Hyderabad, and done his M.Tech (CSE) from Osmania university, Hyderabad. He is now pursuing his PhD from JNTU, Hyderabad, India, under the supervision of Dr. V.U.K. Sastry in the area of Information Security and Cryptography. He has 10 years of teaching experience and his interest in research area includes, Cryptography, Steganography and Parallel Processing Systems.