A Modified Feistel Cipher Involving Substitution, shifting of rows, mixing of columns, XOR operation with a Key and Shuffling

V.U.K Sastry
Dean R&D, Department of Computer Science and Engineering, Sreenidhi Institute of Science & Tech.
Hyderabad, India.

K. Anup Kumar
Associate Professor, Department of Computer Science and Engineering, SNIST,
Hyderabad, India

***Abstract*— In this paper, we have developed a modification to the Feistel cipher by taking the plaintext in the form of a pair of matrices and introducing a set of functions namely, substitute, shifting of rows, mixing of columns and XOR operation with a key. Further we have supplemented this process by using another function called shuffling at the end of each round of the iteration process. In this analysis, the cryptanalysis clearly indicates that the strength of the cipher is quite significant and this is achieved by the introduction of the aforementioned functions.**

Keywords- encryption; decryption; cryptanalysis; avalanche effect; XOR operation.

I. Introduction

The study of the Feistel cipher has been a fascinating fundamental area in the development of block ciphers in cryptography. In the recent years, we have offered several modifications [1-4] to the classical Feistel cipher by taking the plaintext in the form of a pair of matrices. In all these investigations, we have made use of the multiplication with a single key matrix or the multiplication with a pair of key matrices as a fundamental tool in the development of the cipher. This is associated with the mod operation. Further, we have introduced some operations such as mixing, permutation, blending or shuffling in order to achieve confusion and diffusion, so that, the strength of the cipher becomes significant.

In the present investigation, our objective is to study a modification of the Feistel cipher, wherein we use the fundamental operations such as substitution, shifting of rows, mixing of columns, XOR operation and Shuffling. It may be noted here that the operations, substitution, shifting of rows and mixing of columns are very well utilized in Advanced Encryption Standard (AES) [5]. Our interest here is to develop a strong block cipher which exceeds, in strength, almost all the other ciphers available in the literature.

In what follows we present the plan of the paper. In section 2, we deal with the development of the cipher and present the flowcharts and algorithms required in this analysis. In section 3, we mention an illustration of the cipher and describe the avalanche effect. We study the cryptanalysis in section 4. Finally, in section 5, we discuss the computations and draw conclusions.

II. Development Of The Cipher

Consider a plaintext P containing $2m^2$ characters. On using the EBCIDIC code, the characters occurring in the plaintext can be represented in terms of decimal numbers wherein each number lies in [0 - 255]. Then, these numbers 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, where K is a square matrix whose size is m.

The flowcharts depicting the encryption and the decryption are given below.

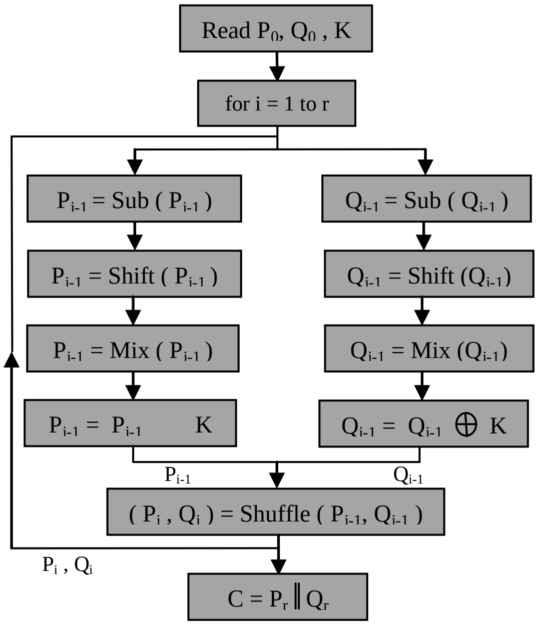
Read P_0, Q_0 , K
for i = 1 to r
P_{i-1} = Sub (P_{i-1})
Q_{i-1} = Sub (Q_{i-1})
P_{i-1} = Shift (P_{i-1})
Q_{i-1} = Shift (Q_{i-1})
P_{i-1} = Mix (P_{i-1})
Q_{i-1} = Mix (Q_{i-1})
P_{i-1} = P_{i-1} K
Q_{i-1} = Q_{i-1} $\oplus$ K
P_{i-1}
Q_{i-1}
(P_i , Q_i) = Shuffle (P_{i-1}, Q_{i-1})
P_i , Q_i
C = $P_r \| Q_r$

Fig 1. The process of Encryption

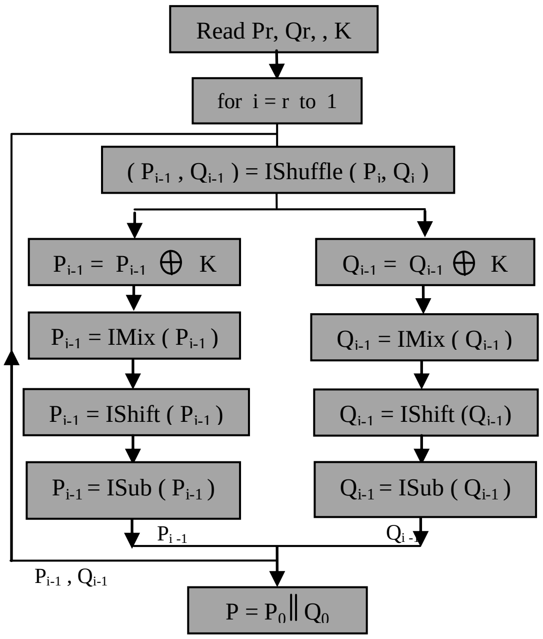
Read Pr, Qr, , K
for i = r to 1
(P_{i-1} , Q_{i-1}) = IShuffle (P_i, Q_i)
P_{i-1} = P_{i-1} ⊕ K
Q_{i-1} = Q_{i-1} ⊕ K
P_{i-1} = IMix (P_{i-1})
Q_{i-1} = IMix (Q_{i-1})
P_{i-1} = IShift (P_{i-1})
Q_{i-1} = IShift (Q_{i-1})
P_{i-1} = ISub (P_{i-1})
Q_{i-1} = ISub (Q_{i-1})
P_{i-1}
Q_{i-1}
P_{i-1} , Q_{i-1}
P = P_0 ‖ Q_0

Fig 2. The process of Decryption

Now we write the algorithms for the processes encryption and decryption as given below.

A. *Algorithm for Encryption*

```
1. Read P, K
2. $P_0$ = Left half of P.
3. $Q_0$ = Right half of P.
4. for i = 1 to r
     begin
       $P_{i-1}$ = Sub ($P_{i-1}$)
       $P_{i-1}$ = Shift ($P_{i-1}$)
       $P_{i-1}$ = Mix ($P_{i-1}$)
       $P_{i-1} = P_{i-1} \bigcirc K$
       $Q_{i-1}$ = Sub ($Q_{i-1}$)
       $Q_{i-1}$ = Shift ($Q_{i-1}$)
       $Q_{i-1}$ = Mix ($Q_{i-1}$)
       $Q_{i-1} = Q_{i-1} \bigcirc K$
       ($P_i$ , $Q_i$) = Shuffle ( $P_{i-1}$, $Q_{i-1}$ )
     end
5. C = $P_r \| Q_r$ /* $\|$ represents concatenation */
6. Write(C)
```

B. *Algorithm for Decryption*

```
1. Read C, K
2. $P_r$ = Left half of C.
3. $Q_r$ = Right half of C.
4. for i = r to 1
     begin
       ($P_{i-1}$ , $Q_{i-1}$) = IShuffle ( $P_i$, $Q_i$ )
       $P_{i-1} = P_{i-1} \oplus K$
       $P_{i-1}$ = IMix ($P_{i-1}$)
       $P_{i-1}$ = IShift ($P_{i-1}$)
       $P_{i-1}$ = ISub ($P_{i-1}$)
       $Q_{i-1} = Q_{i-1} \oplus K$
       $Q_{i-1}$ = IMix ($Q_{i-1}$)
       $Q_{i-1}$ = IShift ($Q_{i-1}$)
       $Q_{i-1}$ = ISub ($Q_{i-1}$)
     end
5. P = $P_0 \| Q_0$ /* $\|$ represents concatenation */
6. Write (P)
```

Let us now explain the basic ideas underlying in the functions Sub ( ), Shift ( ), Mix ( ), used for substitution, shifting of rows, mixing of columns respectively.

Firstly, Let us focus our attention on the substitution process involved in the function Sub ( ).

Consider the EBCIDIC code which can be written in the form a matrix given by

E (i, j) = 16*(i-1) + (j–1), i = 1 to 16 and j = 1 to 16 (2.1)

All these numbers can be placed in the form of a table.

Let us arrange these numbers, which are lying in the interval [0-255], in a random manner.

We represent these numbers in the hexadecimal notation. All these numbers can be written in the form of a table given below (table 2).

In the encryption process, when we come across a number lying in [0-255], we will replace it by the corresponding number in the substitution table. For example, if we come across the number 70, in the process of encryption, this will be converted into hexadecimal number as 46. Then, 70 will be replaced by the number which is occurring in the 4th row, 6th column of the substitution table, i.e by 5A (= 90 in decimal notation). This is the process of substitution. Keeping the EBCIDIC code matrix and the substitution table in view, we form the inverse substitution table which is given in Table 2.

The inverse substitution table will be utilized while carrying out the decryption process and it is denoted by function ISub ( ).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	AF	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	6A	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Table 1. Substitution Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	F9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	1A	71	1D	29	E5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	F7	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Table 2. Inverse Substitution Table

Now let us see the process of shifting involved in the function Shift ( ), during the encryption process we come across plaintext P_{i-1} and Q_{i-1} in the process of iteration. As P_{i-1} is a square matrix of size m, it can be written in the form

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots\cdots\cdots & p_{1m} \\ p_{21} & p_{22} & p_{23} & \cdots\cdots\cdots & p_{2m} \\ p_{31} & p_{32} & p_{33} & \cdots\cdots\cdots & p_{3m} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ p_{m1} & p_{m2} & p_{m3} & \cdots\cdots\cdots & p_{mm} \end{bmatrix} \tag{2.4}$$

On converting each decimal number in (2.4) into its binary form, we get

$$\begin{bmatrix} p_{111}p_{112}\ldots\ldots p_{118} & p_{121}p_{122}\ldots\ldots p_{128} & \cdots\cdots & p_{1m1}p_{1m2}\ldots\ldots p_{1m8} \\ p_{111}p_{112}\ldots\ldots p_{118} & p_{121}p_{122}\ldots\ldots p_{128} & \cdots\cdots & p_{1m1}p_{1m2}\ldots\ldots p_{1m8} \\ p_{111}p_{112}\ldots\ldots p_{118} & p_{121}p_{122}\ldots\ldots p_{128} & \cdots\cdots & p_{1m1}p_{1m2}\ldots\ldots p_{1m8} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ p_{111}p_{112}\ldots\ldots p_{118} & p_{121}p_{122}\ldots\ldots p_{128} & \cdots\cdots & p_{1m1}p_{1m2}\ldots\ldots p_{1m8} \end{bmatrix} \tag{2.5}$$

Here each row contains 8m binary bits. In the process of shifting, we offer a right shift of 4 bits in the first row, 12 bits in the second row, 20 bits in the third row and in general,

4 + 8 * (i-1) bits right shift in the i^{th} row.

This process is carried out till we exhaust all the rows. It may be noted here that IShift ( ) denotes the reverse process of Shift ( ).

In this, the binary bits are obviously given a left shift in an appropriate manner.

To have a clear insight into the mixing process denoted by the function Mix ( ), let us consider again the matrix P_{i-1}, which is represented in the form (2.5).

Let us restrict our attention only to a plaintext matrix, wherein, m=4. This can be written in the form given below

$$\begin{bmatrix} p_{111}\ p_{112}\ldots\ldots p_{118} & p_{121}\ p_{122}\ldots\ldots p_{128} & \cdots\cdots & p_{141}\ p_{142}\ldots p_{148} \\ p_{211}\ p_{212}\ldots\ldots p_{218} & p_{221}\ p_{222}\ldots\ldots p_{228} & \cdots\cdots & p_{241}\ p_{242}\ldots p_{248} \\ p_{311}\ p_{312}\ldots\ldots p_{318} & p_{321}\ p_{322}\ldots\ldots p_{328} & \cdots\cdots & p_{341}\ p_{342}\ldots p_{348} \\ p_{411}\ p_{412}\ldots\ldots p_{418} & p_{421}\ p_{422}\ldots\ldots p_{428} & \cdots\cdots & p_{441}\ p_{442}\ldots p_{448} \end{bmatrix} \tag{2.6}$$

This has 4 rows and 32 columns. On concatenating the binary bits of the 1^{st} column and the 17^{th} column we get a string of binary bits, which can be converted into a decimal number. This can be considered as new p_{11}.

On considering the binary bits of the 2^{nd} column and the 18^{th} column and concatenating them, we get another decimal number which will be called as p_{12}.

On adopting the same process till we exhaust all the columns taken in pairs, we get the decimal numbers which correspond to the other elements of the matrix written in the row wise order. Thus we have, the new plaintext matrix, which is obtained after the completion of mixing. Imix ( ) is the reverse process of Mix ( ).

For a detailed discussion of the function shuffle ( ), wherein we are shuffling the columns of two matrices, we refer to [4].

III. Illustration Of The Cipher

Consider the plaintext given below

My dear young lady! We both are well qualified. You have done your B.Tech and I have completed my M.S, where is the problem! We can fly anywhere. Why your father and mother are not accepting our marriage. We both belong to the same cast, we both are farmers. What is the objection of your father and your mother, are they having any thinking regarding my financial status? We are having as much landed property as your father is having. My father and your father both are well trained seasonal politicians. I wonder why your father is not accepting and why your mother is not accepting. Our marriage must happen soon. Yours loving Mr.X (3.1)

Let us focus our attention on the first 32 characters of the plaintext. This is given by

My dear young lady! We both are

On using EBCIDIC code, we get the plaintext matrix P in the form

$$P = \begin{bmatrix} 77 & 121 & 32 & 100 & 101 & 97 & 114 & 32 \\ 121 & 111 & 117 & 110 & 103 & 32 & 108 & 97 \\ 100 & 121 & 33 & 32 & 87 & 101 & 32 & 98 \\ 111 & 116 & 104 & 32 & 97 & 114 & 101 & 32 \end{bmatrix} \tag{3.2}$$

This can be written in the form of a pair of matrices given by

$$P_0 = \begin{bmatrix} 77 & 121 & 32 & 100 \\ 121 & 111 & 117 & 110 \\ 100 & 121 & 33 & 32 \\ 111 & 116 & 104 & 32 \end{bmatrix} \tag{3.3}$$

and

$$Q_0 = \begin{bmatrix} 101 & 97 & 114 & 32 \\ 103 & 32 & 108 & 97 \\ 87 & 101 & 32 & 98 \\ 97 & 114 & 101 & 32 \end{bmatrix} \quad (3.4)$$

Let us take the key matrix K in the form

$$K = \begin{bmatrix} 45 & 128 & 192 & 53 \\ 133 & 200 & 150 & 16 \\ 100 & 150 & 33 & 120 \\ 13 & 189 & 164 & 55 \end{bmatrix} \quad (3.5)$$

On applying the encryption algorithm, given in section 2, we get the ciphertext C in the form

$$C = \begin{bmatrix} 51 & 145 & 164 & 146 & 108 & 237 & 147 & 173 \\ 155 & 18 & 82 & 72 & 85 & 155 & 19 & 71 \\ 182 & 102 & 90 & 237 & 150 & 142 & 218 & 60 \\ 11 & 150 & 219 & 226 & 237 & 177 & 36 & 100 \end{bmatrix} \quad (3.6)$$

On using the decryption algorithm on (3.6), we get back the original plaintext P given by (3.2).

Let us now study the avalanche effect which throws some light on the strength of the cipher.

On changing the first row, first column element of P_0, from 77 to 76, we get a 1 bit change in the plaintext. On applying the encryption algorithm on the modified plaintext, keeping up the key as it is, we get the ciphertext C in the form

$$C = \begin{bmatrix} 218 & 88 & 129 & 219 & 201 & 58 & 54 & 101 \\ 157 & 209 & 7 & 186 & 109 & 153 & 44 & 75 \\ 219 & 120 & 243 & 158 & 95 & 55 & 38 & 117 \\ 43 & 233 & 147 & 229 & 81 & 38 & 133 & 187 \end{bmatrix} \quad .(3.7)$$

On comparing (3.6) and (3.7), after converting them into their binary form, we notice that they differ by 128 bits out of 256 bits. This indicates that the cipher is quite good from the view point of its strength.

Let us now consider a one bit change in the key. This is achieved by changing first row, first column element of the key K, given by (3.5), from 45 to 44.

Now on using the modified key and applying the encryption algorithm, keeping the plaintext as it is, we get the cipher text C in the form

$$C = \begin{bmatrix} 79 & 149 & 68 & 154 & 22 & 239 & 105 & 98 \\ 232 & 131 & 221 & 63 & 57 & 229 & 243 & 114 \\ 103 & 82 & 190 & 152 & 14 & 222 & 73 & 209 \\ 179 & 44 & 237 & 153 & 44 & 75 & 219 & 120 \end{bmatrix} \quad (3.8)$$

Now on comparing (3.6) and (3.8), after converting both into their binary form, we find that these two ciphertexts differ by 134 bits out of 256 bits.

This also shows that, the strength of the cipher is expected to be significant.

IV. Cryptanalysis

In cryptography, determination of the strength of the cipher is a very important aspect. In the literature of cryptography, it is well known that the cryptanalysis can be carried out by the following approaches.

1. Ciphertext only attack (Brute force attack)
2. Known plaintext attack
3. Chosen plaintext attack
4. Chosen ciphertext attack

As William Stallings [6] has pointed out that every cipher must be designed so that it withstands the first two attacks at least.

Let us now consider the brute force attack.

Here the key is containing m^2 decimal numbers. Thus the size of the key space

$$= 2^{8m^2} .$$

Let us suppose that, the time required for the computation of the cipher with one value of the key is 10^{-7} seconds. Then the time required for processing the cipher with all the possible values of the key in the key space is

$$\frac{(2)^{8m^2} \times 10^{-7}}{365x24x60x60} = \frac{10^{(2.4)\,m^2-7}}{365x24x60x60} = 3.12 \times 10^{(2.4)m^2-15} \text{ years}$$

This time is very large when m is greater than or equal to 3.

In our example as we have taken m=4, the attack on this cipher, by the brute force approach, is totally ruled out.

Let us now investigate the known plaintext attack. In this case, we know as many plaintext and ciphertext pairs as we require, making an attempt for breaking the cipher. In the light of the above information, we have as many pairs of P and C as we require.

If we take r=1, that is, if we confine our attention to a single round of the iteration process, then we have the relations connecting C and P as follows:

$$P_0 = \text{Sub}\ (P_0) \tag{4.1}$$
$$P_0 = \text{Shift}\ (P_0) \tag{4.2}$$
$$P_0 = \text{Mix}\ (P_0) \tag{4.3}$$
$$P_0 = P_0 \oplus K \tag{4.4}$$

$$Q_0 = \text{Sub}\ (Q_0) \tag{4.5}$$
$$Q_0 = \text{Shift}\ (Q_0) \tag{4.6}$$
$$Q_0 = \text{Mix}\ (Q_0) \tag{4.7}$$
$$Q_0 = Q_0 \quad \oplus \quad K \tag{4.8}$$

$$(P_1\ , Q_1) = \text{Shuffle}\ (\ P_0, Q_0\ ) \tag{4.9}$$

$$C = P_1\ \|\ Q_1 \tag{4.10}$$

In the known plaintext attack, we know P_0 and Q_0 corresponding to the initial stage. We also know the C obtained at the end.

As C is known to us, we can determine P_1 and Q_1 from (4.10)

On using the IShuffle ( ), on (4.9), we get the current P_0 and Q_0 which are occurring on the left hand side of (4.4) and (4.8). On using initial the P_0 and the Sub ( ), we get P_0 on the left hand side of (4.1). After that, on using shift ( ) on the available P_0, we get P_0 occurring on the Left hand side of (4.2). Then on using the function Mix( ) on the current P_0, we have the P_0 occurring on the left side of (4.3). Thus, we can readily determine the key K from (4.4). Hence this cipher can be broken by the known plaintext attack if we confine only to one step in the iteration process.

Let us now study the cipher when r = 2. Then the equations governing the cipher are (4.1) to (4.10) and the following

$$P_1 = \text{Sub}\ (P_1) \tag{4.11}$$
$$P_1 = \text{Shift}\ (P_1) \tag{4.12}$$
$$P_1 = \text{Mix}\ (P_1) \tag{4.13}$$
$$P_1 = P_1 \oplus K \tag{4.14}$$

$$Q_1 = \text{Sub}\ (Q_1) \tag{4.15}$$
$$Q_1 = \text{Shift}\ (Q_1) \tag{4.16}$$
$$Q_1 = \text{Mix}\ (Q_1) \tag{4.17}$$
$$Q_1 = Q_1 \quad \oplus \quad K \tag{4.18}$$

$$(P_2\ , Q_2) = \text{Shuffle}\ (\ P_1, Q_1\ ) \tag{4.19}$$

$$C = P_2\ \|\ Q_2 \tag{4.20}$$

In the known plaintext attack, we know C, obtained at the end of the iteration process, and the corresponding P_0 and Q_0, which are available at the very beginning of the iteration process.

As we know C, we can determine P_2 and Q_2 from (4.20). On using IShuffle on (4.19), we get P_1 and Q_1 which are occurring on the left side of (4.14) and (4.18). We cannot determine K as we do not know the P_1 and Q_1 occurring in the right hand side of (4.14) and (4.18). Here, we notice that, though P_0 and Q_0 are known to us, we cannot determine the P_1 and Q_1 which are occurring on the right hand side of (4.14) and (4.18), by starting at the beginning as the key K is occurring in (4.4) and (4.8). In the light of these facts, this cipher cannot be broken by the known plaintext attack, when we have confined to r=2. This shows that it is impossible to break the cipher by the known plaintext attack when we carry out all the sixteen rounds in the iteration.

Intuitively choosing a plaintext or ciphertext and determining the key or a function of the key is a formidable task in the case of this cipher.

From the above discussion we conclude that this cipher is not breakable by all the possible attacks that are available in cryptography.

V. COMPUTATIONS AND CONCLUSIONS

In this investigation, we have offered a through modification in the Feistel cipher by taking the plaintext in the form of a pair of matrices, and by applying several procedures, namely, substitution, shifting, mixing, XORing with the key and shuffle operation. Each one of these procedures modifies the plaintext in a through manner and creates confusion and diffusion in the development of the cipher. The iteration process, which is the basic one in this cipher, supports all the above procedures in a strong way.

Here it may be noted that the substitution table generated in a random manner by using the numbers [0-255] is to be sent to the receiver by the sender.

The programs for encryption and decryption are written in C language.

The plaintext given in (3.1) is divided into 20 blocks, wherein each block is containing 32 characters. We have appended in the last block by adding 13 blank characters, so that it becomes a complete block. On applying the encryption algorithm given in section 2 we get the cipher text corresponding to the entire plaintext (excluding the first block for which the cipher text is already given in (3.6)), in the form

```
212  111  166  213  179  183  219  102  51   84   223  38   165  45   198  253
244  153  37   69   150  119  82   206  223  122  100  147  82   145  190  142
122  45   157  190  115  140  161  154  229  63   77   179  44   237  243  158
140  154  148  153  53   41   110  76   146  115  202  111  223  77   50   100
147  158  94   147  126  250  105  153  103  121  34   63   71   62   155  102
51   93   211  211  35   125  54   173  157  186  100  149  22   94   115  140
161  154  229  63   77   179  44   237  243  158  140  154  148  153  53   41
110  55   38   73   81   237  201  146  84   89   103  121  34   63   71   62
155  102  51   93   211  211  14   113  148  51   92   228  201  42   61   185
79   211  108  203  59   124  231  142  242  68   126  142  140  154  148  153
53   41   110  76   146  115  218  100  201  39   60   189  38   253  244  211
50   206  242  68   126  142  125  54   204  102  187  167  166  70   250  109
91   59   124  231  25   67   53   202  126  155  102  89   219  231  61   25
53   41   14   113  148  51   92   228  201  42   61   185  79   211  108  203
59   124  231  142  242  68   126  142  140  154  148  153  53   41   110  76
146  115  218  100  201  39   60   189  38   253  244  211  50   206  242  68
126  142  125  54   204  102  187  167  166  91   81   190  155  86   206  223
217  140  219  103  172  102  143  209  207  108  198  109  70   250  109  91
59   125  182  99   53   77   242  106  82   220  111  223  73   146  84   89
103  117  44   237  247  166  73   53   41   27   232  231  162  217  219  231
56   202  25   174  83   244  219  50   206  223  57   232  201  169  73   147
82   150  228  201  39   60   166  253  244  211  38   73   57   229  233  55
239  166  153  150  119  146  35   244  115  233  182  99   53   221  61   50
55   211  106  217  219  166  73   81   101  231  56   202  25   174  83   244
219  50   206  223  57   232  201  169  73   147  82   150  227  114  100  149
30   220  153  37   69   150  119  146  37   221  61   48   231  25   67   53
206  76   146  163  219  148  253  54   204  179  183  206  120  239  36   71
232  232  201  169  73   147  82   150  228  201  39   61   166  76   146  115
203  210  111  223  77   51   44   239  36   71   232  231  211  108  198  107
186  122  100  111  166  213  179  183  206  113  148  51   92   167  233  182
```

101	157	190	115	209	147	82	144	231	25	67	53	206	76	146	163
219	148	253	54	204	179	183	206	120	239	36	71	232	232	201	169
73	147	82	150	228	201	39	61	166	76	146	115	203	210	111	223
77	51	44	239	36	86	233	233	150	212	111	166	213	179	183	206
203	206	113	148	51	92	167	233	182	101	157	190	115	209	147	82
147	38	165	45	198	228	201	42	61	185	50	74	139	44	239	36
218	182	118	250	198	104	253	26	93	211	211	14	113	148	51	92
228	201	42	61	185	79	211	108	203	59	124	231	142	242	68	126

The cryptanalysis, carried out in this investigation, clearly shows that this cipher is a strong one. This has become a very good cipher as we have taken the length of the plaintext as large as possible (2048 bits), and supported the encryption process with a good number of functions so that the plaintext undergoes a through transformation (in each round of the iteration process) before it becomes the ciphertext. In this analysis, the substitution table generated in the random manner plays a very important role.

REFERENCES

[1] V.U.K Sastry and 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 Operation", International Journal of Computer Technology and Applications ISSN: 2229-6093. Vol. 3, No.1, pp. 23-31, 2012.

[2] V.U.K Sastry and 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 Modular Arithmetic Addition", International Journal of Computer Technology and Applications ISSN: 2229-6093. Vol. 3, No.1, pp. 32-39, 2012.

[3] V.U.K Sastry and K. Anup Kumar, "A Modified Feistel Cipher Involving a Pair of Key Matrices, Supplemented with XOR Operation, and Blending of the Plaintext in each Round of the Iteration Process", International Journal of Computer Science and Information Technologies ISSN: 0975-9646. Vol. 3, No.1, pp. 3133-3141, 2012.

[4] V.U.K Sastry and K. Anup Kumar, "A Modified Feistel Cipher involving a pair of key matrices, Supplemented with Modular Arithmetic Addition and Shuffling of the plaintext in each round of the iteration process", International Journal of Computer Science and Information Technologies ISSN: 0975-9646. Vol. 3, No.1, pp. 3119-3128, 2012.

[5] Daemen J, and Rijmen V, "Rijndael, the Advanced Encryption Standard (AES)", Dr. Dobbs Journal, Vol. 26(3), pp. 137 -139, Mar 2001.

[6] William Stallings, Cryptography and Network Security, Principles and Practice, Third Edition, Pearson, 2003.

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 963 – 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 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.