Journal of Computer Science 4 (2): 117-124, 2008
ISSN 1549-3636

A Block Cipher using Feistal's Approach Involving Permutation and Mixing of the Plaintext and the Additive Inverse of Key Matrix

[1]Aruna Varanasi and [2]S. Udaya Kumar
[1]Department of CSE, SNIST, Hyderabad, India
[2]Aurora's Engineering College Bhongir, A.P., India

Abstract: In this research, we have developed a block cipher for a block of size 112 bits by using an iterative method involving a permutation of the plaintext and the subkeys generated in each iteration. Here we have represented the plaintext as a matrix of size 14×8, comprising binary bits. In the process of encryption, we have used a key matrix (K), which also consists of binary bits and generated subkeys from K for each iteration. For decryption, we have used the Additive inverse $(K'_i)^{-1}$ of the subkeys. In this, we have discussed the cryptanalysis and have shown that the cipher cannot be broken by any cryptanalytic attack.

Key words: Block cipher, subkeys, key matrix, additive inverse

INTRODUCTION

A number of block ciphers[1-6] have been developed in the recent past, which can be found in the literature. Feistal[7,8] has used the concepts of Hill cipher[9] and developed the Feistal cipher. However, subsequently he found that his approach is vulnerable for cryptanalytic attacks. Recently Udaya *et al*.[10-16] developed a few block ciphers using Feistal's approach and have shown that the ciphers developed by them are cryptographically stronger.

In the present research, we proposed a block cipher with a different approach based on the Feistal structure. Here, we use the concept of permutation and diffusion, in the development of cipher. In this, we have shown that a thorough mixing of the elements of the subkeys generated for each iteration and the plaintext permuted in each iteration will lead to a cipher, which cannot be broken by any cryptanalytic attack.

DEVELOPMENT OF CIPHER

In this study before we discuss the development of cipher, we first discuss the generation of key matrix from the given key, denoted by K_0.
Let

$$K_0 = \text{abcdefghijklmnop} \quad (1)$$

Now we convert each element in the key, K_0 to its corresponding 7 bit ASCII code. Since the key, K_0 contains 16 elements, the corresponding ASCII code bits of key, denoted as K_{0A} will comprise 112 bits. Thus

$$K_{0A} = \begin{Bmatrix} 1&1&0&0&0&0&1&1&1&0 \\ 0&0&1&0&1&1&0&0&0&1 \\ 1&1&1&0&0&1&0&0&1&1 \\ 0&0&1&0&1&1&1&0&0&1 \\ 1&0&1&1&0&0&1&1&1&1 \\ 1&0&1&0&0&0&1&1&0&1 \\ 0&0&1&1&1&0&1&0&1&0 \\ 1&1&0&1&0&1&1&1&1&0 \\ 1&1&0&0&1&1&0&1&1&0 \\ 1&1&1&0&1&1&1&0&1&1&0 \\ 1&1&1&1&1&1&1&0&0&0&0 \end{Bmatrix} \quad (2)$$

Let us now generate a matrix and denote it as K'_0 of size 28×4. The process of generation of K'_0 is as follows:

The first element of the 112 bits of K_{0A} is placed in the 1st row of 1st column of the matrix K'_0. The second element in K_{0A} is placed in the 1st row 2nd column of K'_0. Then the third and the fourth elements of K_{0A} are placed in the 1st row, 3rd and 4th columns of K'_0 respectively. Now the fifth element of K_{0A} is placed in the 2nd row 1st column, sixth element is placed in 2nd row 2nd column, the seventh and eighth elements are placed as 2nd row 3rd and 4th columns of K'_0 respectively. This process continues until all the elements of K_{0A} are exhausted. Thus, K'_0 is

Corresponding Author: Aruna Varanasi, Department of CSE, SNIST, Hyderabad, India

$$K'_0 = \begin{bmatrix} 1&1&0&0\\ 0&0&1&1\\ 1&0&0&0\\ 1&0&1&1\\ 0&0&0&1\\ 1&1&1&0\\ 0&1&0&0\\ 1&1&0&0\\ 1&0&1&1\\ 1&0&0&1\\ 1&0&1&1\\ 0&0&1&1\\ 1&1&1&0\\ 1&0&0&0\\ 1&1&0&1\\ 0&0&1&1\\ 1&0&1&0\\ 1&0&1&1\\ 0&1&0&1\\ 1&1&1&0\\ 1&1&0&0\\ 1&1&0&1\\ 1&0&1&1\\ 1&0&1&1\\ 1&0&1&1\\ 0&1&1&1\\ 1&1&1&1\\ 0&0&0&0 \end{bmatrix} \quad (3)$$

From K'_0 let us now generate a 28×8 matrix, wherein the elements of K'_0 are repeated and permuted to give rise to a matrix of size 28×8, denoted as K′. The procedure for generating K′ is detailed as given under:

The 28×4 matrix of K'_0 are placed as it is. Then, the first element under column 1 in 28th row of K'_0 is placed in the first row fifth column and the first element in the 27th row is placed in the second row fifth column and so on. Then we get the first column elements of K'_0 will appear in the descending order in the fifth column of K′ similarly we place the elements in other columns 2, 3, 4 of K'_0 in the sixth, seventh and eighth columns. Thus K′ is

$$K' = \begin{bmatrix} 1&1&0&0&0&0&0&0\\ 0&0&1&1&1&1&1&1\\ 1&0&0&0&0&1&1&1\\ 1&0&1&1&1&0&1&1\\ 0&0&0&1&1&0&1&1\\ 1&1&1&0&1&0&1&1\\ 0&1&0&0&1&1&0&1\\ 1&1&0&0&1&1&0&0\\ 1&0&1&1&1&1&1&0\\ 1&0&0&1&0&1&0&1\\ 1&0&1&1&1&0&1&1\\ 0&0&1&1&1&0&1&0\\ 1&1&1&0&0&0&1&1\\ 1&0&0&0&1&1&0&1\\ 1&1&0&1&1&0&0&0\\ 0&0&1&1&1&1&1&0\\ 1&0&1&0&0&0&1&1\\ 1&0&1&1&1&0&1&1\\ 0&1&0&1&1&0&0&1\\ 1&1&1&0&1&0&1&1\\ 1&1&0&0&1&1&0&0\\ 1&1&0&1&0&1&0&0\\ 1&0&1&1&1&1&1&0\\ 1&0&1&1&0&0&0&1\\ 1&0&1&1&1&0&1&1\\ 0&1&1&1&1&0&0&0\\ 1&1&1&1&0&0&1&1\\ 0&0&0&0&1&1&0&0 \end{bmatrix} \quad (4)$$

We now once again permute K′ to obtain the key matrix K of size 28×8 the procedure for generating K from K′ is described as follows:

The element in the 15th row of the 1st column of the matrix K′ is placed as element in the 1st row 1st column; the element in the 15th row 2nd column of the matrix K′ is placed as the element in the 2nd row fist column and so on. This way the eight elements belonging to the 15th row have now appeared as the first eight elements in the first column of the matrix K. We continue in this fashion till we exhaust the first 4 elements of 18th row of K′ this gives the first column of 28 elements in K. We now use the same procedure and obtain the 2nd, 3rd and 4th column of K from the remaining elements of 18th row (4 bits) till the final element of 28th row. We now come back to row 1 of K′ and place these eight elements as the first eight elements of the fifth column of K. Then we take the eight elements of the second row and third row of K′ respectively and place them as the next sixteen elements of 5th columns of K. We now take the first four elements of the fourth row of K′ and place them as the remaining four elements of 5th column of K. We continue this process till all the remaining elements of the fourth row to end of 14th row. Thus we have generated key matrix, K is given by,

$$K = \begin{bmatrix} 1&1&1&1&1&1&1&1\\ 1&0&1&0&1&0&1&0\\ 0&1&0&1&0&1&0&1\\ 1&1&1&1&0&1&0&1\\ 1&0&0&0&0&0&1&0\\ 0&1&1&1&0&0&1&0\\ 0&0&0&1&0&0&0&1\\ 0&1&0&1&0&1&0&1\\ 0&1&1&1&0&1&1&1\\ 0&0&0&0&0&0&0&0\\ 1&0&1&0&1&1&1&1\\ 1&1&1&0&1&1&1&0\\ 1&1&1&1&1&1&1&1\\ 1&1&1&1&1&1&1&1\\ 1&1&1&1&1&1&1&1\\ 0&0&0&1&1&0&0&0\\ 1&1&1&0&1&1&1&0\\ 0&0&0&0&0&0&0&0\\ 1&1&1&1&0&1&0&1\\ 0&1&1&1&0&1&1&1\\ 0&1&0&0&0&0&0&1\\ 0&1&0&0&1&1&1&0\\ 1&0&0&0&1&0&0&0\\ 1&0&1&0&1&0&1&0\\ 1&1&1&1&1&1&1&1\\ 0&1&0&1&0&1&0&1\\ 1&0&1&0&1&0&1&0\\ 1&0&1&0&1&1&1&1 \end{bmatrix} \quad (5)$$

Let us now generate the subkeys twenty (20) in number to be used in the twenty (20) iterations with one subkey in each iteration. The procedure involved in the generation of subkeys is discussed below.

The first subkey K_1 is generated from K as follows:

The element in the 8th column of the first row of the matrix K is placed as the element in the first row, first column of the matrix K_1. Then the element in the 7th column first row of the matrix K is placed as the element in the 2nd row of the first column of the matrix K_1. This process is continued for the remaining elements of the first row of the matrix K. Thus, we have the elements of the first row of the matrix K have appeared in the descending order in the first column as the first eight elements of the matrix K_1. In a similar manner, this process is continued for the 2nd and 3rd rows of K. At this stage, we have twenty four (24) elements in the first column of the matrix K_1. Now, we take the last four (4) elements of the 4th row of K and we place them as the remaining elements in the first column of K_1. Thus we have 28 elements in the first column of K_1. Now the first four (4) elements of the 4th row are placed as the first four elements in the second column of K_1 and the elements of 5th, 6th, 7th rows of K appear in descending order in the second column of matrix K_1. We continue this process of placing the remaining elements of the matrix K as the remaining elements of the matrix K_1. Thus we have a 28×8 Matrix, K_1 given by

$$K_1 = \begin{bmatrix} 1&1&1&0&1&0&0&1\\ 1&1&0&1&1&0&1&1\\ 1&1&1&0&1&0&1&1\\ 1&1&0&1&1&0&1&1\\ 1&0&1&0&1&1&0&1\\ 1&1&0&1&1&0&0&0\\ 1&0&1&1&1&1&1&1\\ 1&0&0&1&1&0&0&0\\ 0&0&1&0&0&1&0&1\\ 1&0&1&1&0&1&0&0\\ 0&0&1&1&0&1&0&1\\ 1&1&0&1&1&1&1&0\\ 0&0&1&1&1&1&0&0\\ 1&1&1&1&0&1&0&1\\ 0&0&1&1&0&1&0&0\\ 1&0&0&1&0&0&1&1\\ 1&1&0&1&0&1&0&0\\ 0&1&0&1&1&1&1&1\\ 1&1&0&1&1&1&0&0\\ 0&0&0&1&1&0&1&1\\ 1&1&0&1&0&1&0&1\\ 0&0&0&1&1&0&1&1\\ 1&0&0&1&1&0&0&1\\ 0&0&0&1&1&0&1&1\\ 1&1&1&1&0&0&1&0\\ 0&0&1&1&0&0&1&1\\ 1&0&1&1&0&1&1&0\\ 0&0&1&1&0&0&1&1 \end{bmatrix} \quad (6)$$

Now the subkey K_2 is generated from K_1 as follows:

Here, we place the first element of 28th row of K_1 as the element in the first row, first column of the matrix K_2. We then place the 2nd element of the 28th row of K_1 as the 2nd element of the first column of K_2. Similarly we place all the remaining six (6) elements of the 28th row of matrix K as the remaining elements of first column of K_2 Proceeding in a similar, we place the elements belonging to 27th and 26th rows of K_1 as the next sixteen (16) elements in the first column of K_2. We now take the elements of the 25th row belonging to columns 1, 2, 3 and 4 and place them as the remaining 4 elements of the first column of K_2. At this stage, we have 28 elements in the first column of K_2. Similarly the remaining elements of 25th row of the columns 5, 6, 7 and 8 of K_1 are placed as the first four elements of the second column of K_2. Proceeding in this way we place all the remaining elements of K_1 and obtain K_2.

$$K_2 = \begin{bmatrix} 0&0&1&1&1&0&1&1\\ 0&0&1&1&1&1&0&0\\ 1&1&0&1&1&0&1&1\\ 1&0&1&1&1&1&1&1\\ 0&0&0&1&0&1&1&1\\ 0&0&1&1&1&0&1&1\\ 1&0&0&0&0&1&1&1\\ 1&1&1&1&1&1&1&0\\ 1&1&0&0&0&0&1&1\\ 0&0&0&1&0&1&1&0\\ 1&1&0&0&1&0&0&1\\ 1&1&1&0&1&0&1&1\\ 0&1&1&1&1&0&1&1\\ 1&0&0&0&1&0&0&1\\ 1&0&1&0&0&1&0&0\\ 0&1&1&1&0&0&0&1\\ 0&1&1&0&1&0&1&1\\ 0&0&1&0&1&1&0&0\\ 1&0&0&1&0&0&1&1\\ 1&1&1&1&1&1&0&1\\ 0&0&1&0&1&1&1&1\\ 0&0&1&0&1&0&1&1\\ 1&0&0&1&1&0&0&1\\ 1&1&0&1&0&1&1&0\\ 1&1&0&0&0&1&1&1\\ 1&0&1&1&0&0&1&0\\ 1&1&0&0&1&0&0&0\\ 1&1&1&0&1&0&1&1 \end{bmatrix} \quad (7)$$

K_3 is generated from K_2 like the way K_1 was generated from K. In a similar manner K_5, K_7, K_9…..K_{19} are generated from K_4, K_6, K_8…K_{18} respectively.

K_4 is generated from K_3 using the same procedure adopted in the generation of K_2 from K_1. In the same manner K_6, K_8, K_{10}…K_{20} are generated from K_5, K_7, K_9…K_{19} respectively.

We now use the subkey of each iteration (K_1, $K_2 \ldots K_{20}$) which are of size 28×8 matrix and convert each of them in to 14×8 matrix (K'_1, $K'_2 \ldots K'_{20}$) by using addition modulo 2^8. This was done as follows.

First two rows of 28×8 matrix (K_1) are taken and addition modulo 2^8 is performed on these two rows to give rise to a single row of 8 bits, this is the first row of a new matrix, K'_1 of size 14×8, then the next two rows i.e. 3rd and 4th rows of K_1 are taken and addition modulo 2^8 is performed on these two rows, which gives a row of 8 bits which is the 2nd row of K′. In a similar manner the 5th and 6th, 7th and 8th…up to 27th and 28th rows are taken and addition modulo 2^8 is performed on each pair of rows there by generating the complete matrix K'_1, of size 14×8.

$$K'_1 = \begin{bmatrix} 1&1&0&0&0&1&0&0\\ 1&1&0&0&0&1&1&0\\ 1&0&0&0&0&1&0&1\\ 0&1&0&1&0&1&1&1\\ 1&1&0&1&1&0&0&1\\ 0&0&0&1&0&0&1&1\\ 0&0&1&1&0&0&0&1\\ 1&1&0&0&0&1&1&1\\ 0&0&1&1&0&0&1&1\\ 1&1&1&1&0&1&1&1\\ 1&1&1&1&0&0&0&0\\ 1&0&1&1&0&1&0&0\\ 0&0&1&0&0&1&0&1\\ 1&1&1&0&1&0&0&1 \end{bmatrix} \quad (8)$$

Similarly the other matrices K'_2, K'_3,...,K'_{20}, are generated from K_2, K_3,...,K_{20} respectively.

Let us now consider the plaintext P_0 comprising 16 characters, given by

$$P_0 = \text{network security} \quad (9)$$

By taking the 7 bit ASCII code of each character, we have 112 bits of plaintext given by.

$$P_{0A} = \begin{bmatrix} 1101110110\ 0101111010\ 011\\ 1011111011\ 1111100101\ 101\\ 0110100000\ 11100111\ 10010\\ 1110001111\ 1010111100\ 101\\ 1010011110\ 1001111001 \end{bmatrix} \quad (10)$$

Let us take the first 8 bits of P_{0A} given by equation (10) and place them as the first row of a matrix P of size 14×8, the next 8 bits of P_{0A} are taken and placed as the 2nd row of same matrix. Similarly, we continue this process and generate the matrix P until we exhaust all the 112 bits of the plaintext, P_{0A}

$$P = \begin{bmatrix} 1&1&0&1&1&1&0&1\\ 1&0&0&1&0&1&1&1\\ 1&0&1&0&0&1&1&1\\ 0&1&1&1&1&1&0&1\\ 1&1&1&1&1&1&0&0\\ 1&0&1&1&0&1&0&1\\ 1&0&1&0&0&0&0&0\\ 1&1&1&0&0&1&1&1\\ 1&0&0&1&0&1&1&1\\ 0&0&0&1&1&1&1&1\\ 0&1&0&1&1&1&1&0\\ 0&1&0&1&1&0&1&0\\ 0&1&1&1&1&0&1&0\\ 0&1&1&1&1&0&0&1 \end{bmatrix} \quad (11)$$

The ciphertext, C corresponding to the plaintext P_0 is generated using the following procedure:

$$C_1 = (K'_1 + P) \bmod 2^8$$
$$C_2 = (K'_2 + C'_1) \bmod 2^8$$
$$C_3 = (K'_3 + C'_2) \bmod 2^8$$
$$\vdots$$
$$\text{Hence } C_{20} = (K'_{20} + C'_{19}) \bmod 2^8$$

It may be noted here that each row of the matrix C_i is obtained by performing the modulo 2^8 addition on each row of the matrix K'_I and the corresponding row of the matrix C'_{i-l}, i.e.,

$$C_i = (K'_i + C'_{i-l}) \bmod 2^8 \quad (12)$$

where $C'_0 = P$ and C'_{i-l} is a permutation of C_{i-l}. Here i takes the values 1-20. Further, C'_{20} is a permutation of C_{20}. The procedure used for permutation is the same as we have obtained the matrix K_1 from K. Thus $C = C'_{20}$, which is given by

$$C = \begin{bmatrix} 1&0&1&0&0&1&0&0\\ 0&1&1&0&1&1&0&0\\ 1&0&1&0&0&1&0&0\\ 1&0&1&1&0&0&0&0\\ 0&1&1&1&1&0&1&1\\ 1&0&1&0&1&0&0&1\\ 1&0&1&1&1&0&1&1\\ 1&0&1&0&0&0&1&1\\ 0&0&1&1&1&0&0&0\\ 0&1&1&0&1&1&1&1\\ 0&0&1&0&0&0&1&1\\ 1&1&0&0&0&1&0&0\\ 0&1&1&0&1&0&0&0\\ 1&0&0&1&0&1&0&0 \end{bmatrix} \quad (13)$$

where C is the ciphertext, which is a 14×8 matrix for the plaintext P_0 given by Eq. 9.

We concatenate each row of C and get 112 ciphertext bits corresponding to 112 plaintext bits.

$$C=\begin{bmatrix}1010010001101100\\1010010010110000\\01111011101010011\\01110111010001100\\11100001101111001\\00011110001000110\\100010010100\end{bmatrix} \quad (14)$$

Decryption of the cipher is done using the same algorithm as encryption with the input being the ciphertext. The decryption subkeys used are the additive modular arithmetic inverses of the encryption subkeys with the key roles being reversed from that of the encryption.

ENCRYPTION AND DECRYPTION ALGORITHMS

In what follows, we briefly present the algorithms for subkey generation, encryption, decryption and additive inverse of the subkeys respectively.

Algorithm for key generation:

Step 1: Initialize key, K_0 by reading 16 characters

Step 2: Generate K_{0A}, ASCII code of each character from K_0

Step 3: Find K'_0 from K_{0A}

Step 4: Find K' from K_0'

Step 5: Find K from K'

Step 6: Find K_i from K_{i-1} for i = 1-20 where $K_0 = K$

Step 7: Find K_i' from K_i for i = 1-20

Algorithm for encryption:

Step 1: Read P, K'_I for i = 1-20

Step 2: for i = 1-20 do

$C_i = (K'_i + C_{i-1}') \bmod 2^8$, where $C_0' = P$

Step 3: $\mathbf{C} = C_{20}'$

Algorithm for decryption;

Step 1: Read C, $(K'_i)^{-1}$ for i = 1-20

Step 2: Find C_{20} from C_{20}'

Step 2: For i = 20-1 do

$C_{i-1}' = ((K_i')^{-1} + C_i) \bmod 2^8$

Find C_{i-1} from C_{i-1}'

Step 3: $P = C_0'$

Additive inverse of the subkeys:

Step 1: Read K'_i for i = 1-20

Step 2: For i = 1-20 do

Find $(K'_i)^{-1}$ such that $((K'_i)+(K'_i)^{-1}) \bmod 2^8 = 0$

ILLUSTRATION OF THE CIPHER

Let us consider the plaintext, P_0 given by

P_0 = network security (15)

This consists of 16 characters including one blank space. By using the ASCII code of each character, we represent the plaintext, P_0 in terms of seven (7) binary bits. Let us now place the seven (7) binary bits of each character in a row and obtain a matrix, P_{0A} given by

$$P_{0A}=\begin{bmatrix}1101110110010\\1111010011101\\1111011111110\\0101101011010\\0000111001111\\0010111000111\\1101011110010\\1101001111010\\01111001\end{bmatrix} \quad (16)$$

Let us now convert the matrix, P_{0A} into a 14×8 matrix, P given by

$$P = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \quad (17)$$

Let us now consider key, K_0 comprising 16 characters, which is given by

$$K_0 = \text{abcdefghijklmnop} \quad (18)$$

Each character can be represented by the corresponding 7 bit ASCII numbers. Hence we get 112 bits (16×7 = 112) and place them in the form of a 28×4 matrix. As discussed in section 2, we generate 20 subkeys K_1, K_2...K_{20}. From these, we get K'_i for i = 1 to 20 by adopting the modulo 2^8 addition.

We obtain the corresponding modulo arithmetic inverse $(K'_i)^{-1}$ for each K'_i, wherein i takes values 1-20 satisfying the relation

$$(K'_i + (K'_i)^{-1}) \bmod 2^8 = 0 \quad (19)$$

Now on using the algorithm for encryption discussed in section 3.2, we obtain the ciphertext, C corresponding to the plaintext P_0 given by Eq. 9. Thus

$$C = \begin{bmatrix} 1\,0\,1\,0\,0\,1\,0\,0\,0\,1\,1 \\ 0\,1\,1\,0\,0\,1\,0\,1\,0\,0\,1 \\ 0\,0\,1\,0\,1\,1\,0\,0\,0\,0\,0 \\ 1\,1\,1\,1\,0\,1\,1\,1\,0\,1\,0\,1 \\ 0\,0\,1\,1\,0\,1\,1\,1\,0\,1\,1\,1 \\ 0\,1\,0\,0\,0\,1\,1\,0\,0\,1\,1 \\ 1\,0\,0\,0\,0\,1\,1\,0\,1\,1\,1 \\ 1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,0\,0\,0\,1\,1\,0\,1 \\ 0\,0\,0\,1\,0\,0\,1\,0\,1\,0\,0 \end{bmatrix} \quad (20)$$

The receiver who has previously obtained the key from the sender uses the decryption algorithm discussed in section 3.3 by using additive modulo inverse 2^8 as shown in Eq. 19, obtains $(K'_i)^{-1}$ for i = 1-20 and retrieves the original plaintext.

As the process of encryption involving the iterative scheme contains equations which mix the plaintext and the key very thoroughly, it can therefore, be anticipated that the cipher cannot be broken by any cryptanalytic attack. Now we discuss briefly the cryptanalysis.

CRYPTANALYSIS

After the first iteration, the ciphertext can be written as:

$$C_1 = (K'_1 + P) \bmod 2^8 \quad (21)$$

At the end of 2nd iteration the cipher can be written as:

$$C_2 = (K'_2 + C'_1) \bmod 2^8 \quad (22)$$

It is to be noted here that C'_1 is the permutation of C_1 while K'_2 is the permutation of the subkey, K_2. Similarly as the number of iterations takes the value 1-20 we have:

$$C_{20} = (K'_{20} + C'_{19}) \bmod 2^8 \quad (23)$$

and

$$C = C_{20}' \quad (24)$$

Knowing the final value of C for a given P (known plaintext attack), neither K_{20}' nor C_{19}' can be obtained. Hence, the cipher can never be broken by the known paintext attack (or) for that matter by any other crypt analytic attack.

AVALANCHE EFFECT

The strength of any cryptographic algorithm is often evaluated by testing the algorithm against the avalanche effect. Here, we test our algorithm by considering the avalanche effect. Consider the plaintext given by

$$P = \text{management study} \quad (25)$$

The above plaintext is of 16 characters, which corresponds to 112 ASCII bits.

Taking the key

$$K_0 = \text{abcdefghijklmnop} \quad (26)$$

The corresponding 112 ASCII bits of K is given by:

$$K = \begin{bmatrix} 1100001110001 \\ 0110001111001 \\ 0011001011100 \\ 11011001111101 \\ 0001101001110 \\ 10101101011110 \\ 11001101101110 \\ 111011011111110 \\ 000 \end{bmatrix} \quad (27)$$

On using the key generation algorithm discussed in section 3.1 and the encryption algorithm discussed in section 3.2, we obtain the ciphertext C corresponding to the plaintext P given by

$$C = \begin{bmatrix} 1100001101111 \\ 1001111000001 \\ 10100111111000 \\ 0001001010100 \\ 1000000110110 \\ 1101010010001 \\ 01010010111101 \\ 111111111010011 \\ 0011 \end{bmatrix} \quad (28)$$

Now changing the sixth character in the plaintext from e to d, which is equivalent to changing the plaintext in a single bit position, i.e., managdment study and using the same key, K_0 mentioned in 26 and applying the encryption algorithm discussed in section 3.2, we obtain the ciphertext given by

$$C_{new} = \begin{bmatrix} 001011111000 \\ 0111110101010 \\ 111100000000 \\ 0111101011101 \\ 0001100111101 \\ 1011000101010 \\ 0001010110111 \\ 0000110011110 \\ 0100001100 \end{bmatrix} \quad (29)$$

On comparing the Eq. 28 and 29, it is readily observed that the two ciphertexts C and C_{new} differ in 65 bits out of 112 bits. This indicates that the algorithm exhibits a strong avalanche effect.

Now changing the key given by 26 in a single bit position i.e. abcdefghijkllnop (m is replaced by l), which results in the new key given by

$$K_{new} = \begin{bmatrix} 1100001110 \\ 0010110001 \\ 1110010011 \\ 0010111001 \\ 10110011111 \\ 0100011010 \\ 01110101011 \\ 01011110110 \\ 01101100111 \\ 110110111111 \\ 10000 \end{bmatrix} \quad (30)$$

On using the key generation algorithm discussed in 3.1 and the encryption algorithm discussed in section 3.2, we obtain the ciphertext $C_{k\ new}$, for the plaintext P given by 25 and the new key given by 30, i.e.,

$$C_{k\ new} = \begin{bmatrix} 10110010000 \\ 1010010011 \\ 01110001110 \\ 01010111011 \\ 01111001101 \\ 00110000000 \\ 001010111101 \\ 00110110111 \\ 10110011010 \\ 01110001110 \\ 1100 \end{bmatrix} \quad (31)$$

On comparing the two ciphers given by 28 and 31, it can be seen that these two ciphers differ in 62 bits out of 112 bits.

This once again clearly proves that the algorithm has a pronounced avalanche effect.

EXPERIMENTAL RESULTS AND CONCLUSIONS

In this study, we have developed a block cipher for a block of size 112 bits. Here, the plaintext size is taken as 112 bits and key size is also taken as 112 bits.

In the generation of the cipher, the plaintexts and the subkey generated in each iteration are permuted. The subkey generation involves a very complex procedure, which provides enough confusion.

The algorithm, which involves the permutation of intermediate ciphertexts and the permutation of subkeys generated in each iteration, provides a very strong diffusion. Hence, the algorithm provides a very strong diffusion and confusion, which are the fundamental requirements for a block cipher.

In this study, we have briefly discussed the cryptanalysis and have logically deduced that no cryptanalytic attack can break the cipher

Towards the end, we have evaluated the algorithm against the avalanche effect, which clearly indicated that this algorithm is indeed a very strong one with inherent cryptographic strength and it cannot be broken by any cryptanalytic attack.

REFERENCES

1. Schneier, B., 1994. The blowfish encryption algorithm. Dr. Dobbs' J., 19: 38-40.
2. Rivest, R.L., 1995. The RC5 encryption algorithm. Dr. Dobbs J., 20: 146-148.
3. Adams, C.M., 1997. The CAST-128 encryption algorithm. RFC 2144, May 1997.
4. Daemen J. and V. Rijmen, 2001. Rijndael, the advanced encryption standard (AES). Dr. Dobb's J., 26: 137-139.
5. Daemen, J., S. Borg and V. Rijmen, 2002. The Design of Rijndael: AES-the Advanced Encryption Standard. Springer-Verlag, ISBN 3-540-42580-2.
6. Hussein Ahmad Al Hassan, Magdy Saeb and Hassan Desoky Hamed, 2005. The Pyramids Block Cipher. Int. J. Network Secur., 1: 52-60.
7. Feistel, H. 1973. Cryptography and computer privacy. Sci. Am., 228: 15-23.
8. Feistel, H., W. Notz and J. Smith, 1975. Some Cryptographic techniques for machine-to-machine data communications. Proceedings of the IEEE, 63: 1545-1554.
9. William Stallings, 2006. Cryptography and Network Security: Principles and Practices. 3rd Edn., Chapter 2, pp: 37.
10. Sastry, V.U.K., S. Udaya Kumar and A. Vinaya babu, 2006. A large block cipher using modular arithmetic inverse of a key matrix and mixing of the key matrix and the plain text. J. Comput. Sci., 2: 698-703
11. Udaya Kumar, S., V.U.K. Sastry and A. Vinaya babu, 2006. An iterative process involving interlacing and decomposition in the development of a block cipher. Int. J. Comput. Sci. Network Secur., 6: 236-245.
12. Udaya Kumar, S., V.U.K. Sastry and A. Vinaya babu, 2006. A large block cipher using an iterative method and the modular arithmetic inverse of a key matirx. IAENG Int. J. Comput. Sci., 32: 395-401.
13. Udaya Kumar, S., V.U.K. Sastry and A. Vinaya babu, 2006. A block cipher basing upon a revisit to the feistel approach and the modular arithmetic inverse of a key matrix. IAENG Int. J. Comput. Sci., 32: 386-394.
14. Udaya Kumar, S., V.U.K. Sastry and A. Vinaya babu, 2007. A block cipher involving interlacing and decomposition. Inform. Technol. J., 6: 396-404
15. Udaya Kumar, S., V.U.K. Sastry and A. Vinaya babu, 2007. A block cipher using an iterative method and the modular arithmetic inverse of a key matrix. Int. J. Sci. Comput., 1: 69-78.
16. Udaya Kumar, S., V.U.K. Sastry and A. Vinaya babu, 2007. A block cipher basing upon permutation, substitution and iteration. J. Inform. Privacy Secur., 3: 47-62.