Dr. K. Anup Kumar Int. Journal of Engineering Research and Application www.ijera.com
ISSN : 2248-9622, Vol. 8, Issue 4, (Part -I) April 2018, pp.90-100

RESEARCH ARTICLE OPEN ACCESS

A Novel Method of Encryption Using Variable Block Sizes in Different Rounds.

Dr. K. Anup Kumar
Professor, Department of Computer Science and Engineering,
Sreenidhi Institute of Science and Technology,
Ghatkesar, Hyderabad, 501301, Telangana, India.

ABSTRACT
we know that the strength of any cipher depends on the degree of confusion and diffusion induced in it. Since most of the transformations used for this purpose are well known to every one, it gives scope for cryptanalysis. This is mainly because of the block sizes remaining constant in all the rounds; which will introduce linearity in the cipher. This helps the crypt analyzer in breaking the cipher. Therefore, we have investigated on a new technique and found that, during encryption the block cipher sizes can be varied in different rounds depending on round key. Such that, a crypt analyzer cannot analyze the transformations used due to variable block sizes being unknown in different rounds. The cryptanalysis carried out in this regard shows that the cipher obtained through this process is a strong one and cannot be broken by any crypt analytic attack.
Index Terms — Cipher Text, Decryption, Encryption, Key, Permutation, Plaintext, Substitution, Variable block size.

Date of Submission: 31-03-2018 Date of acceptance: 16-04-2018

I. INTRODUCTION

In the survey of literature, majority of block ciphers are based on the feistel cipher (Tavares and Heys, 1995; Stallings, 2003). In this process, bits of plaintext undergo a series of permutations, substitutions and exclusive OR operations. This creates confusion and diffusion in cipher which is achieved by the classical round function F of feistel structure.

In our recent papers published, see references [4, 5, 6], we have discussed how key based random permutations, key based random substitutions, interlacing, and decomposition helps us in generating the feistel cipher of good strength. We have used these features in the current paper also. In the present research work, our interest is to develop a stronger version of encryption technique by which one can counter attack the crypt analyzer. This is accomplished by using a new technique called key based variable block sizes in different rounds. As feistel cipher uses same number of bits in a block in all the 16 rounds, there is a scope for cryptanalysis. Because, one can analyze on "how many bits are permuted? XORed? and which set of bits are going into which substitution box etc". Due to key based variable block sizes in different rounds. A crypt analyzer has no information on "what is the block size used in each round?". Hence he cannot decode the transformations applied during encryption.

II. USING KEY BASED VARIABLE BLOCK SIZES IN DIFFERENT ROUNDS

Let 'K' be the key containing 16 integers. Let $d_i = K_i$ mod 4. Such that $d_i \in \{0, 1, 2, 3\}$. These values of d_i help us in permuting the block sizes in respective rounds.

Let us consider a block of plaintext 'P' of 256 bits. Let $C^0 = P$ be the initial plaintext. Let $b^i = \{ 32, 48, 80, 96 \}$ be the different block sizes used in i^{th} round. Such that, in every i^{th} round before encryption, the 256 bit block C^{i-1} is decomposed into 4 blocks B^i_0, B^i_1, B^i_2 and B^i_3 which are encrypted separately. In every round, the block size of B^i_n can be varied based on the corresponding round key value.

Illustration of variable block sizes

Let C^{i-1} be the 256 bit intermediate cipher obtained as the input to the i^{th} round encryption process.

Such that $C^{i-1} = \{ c_1\ c_2\ c_3 \ldots c_{255}\ c_{256} \}$

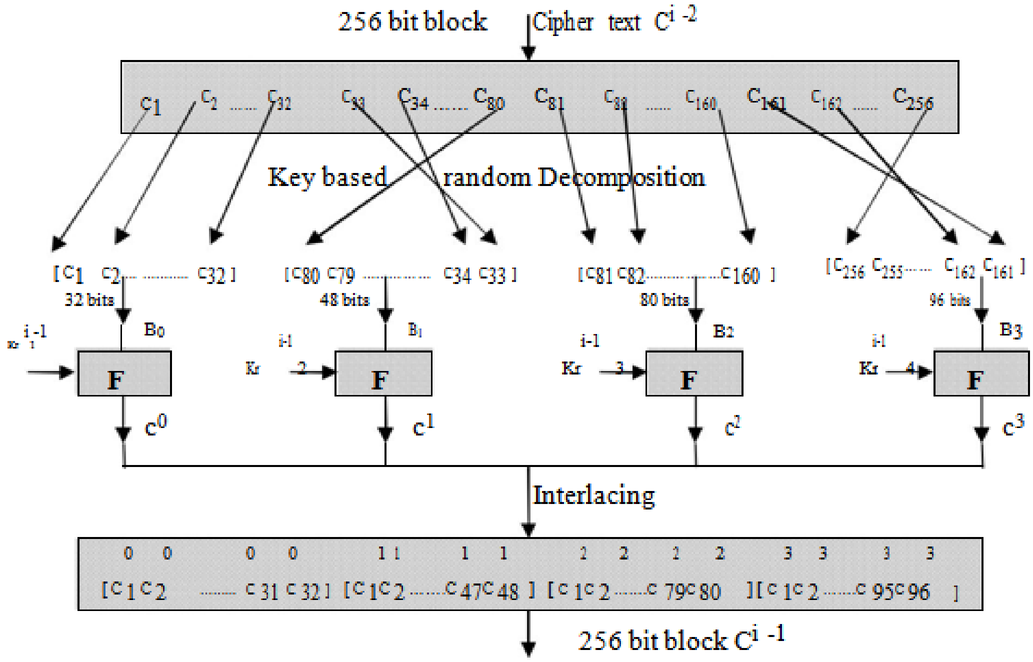
Let $b^{i-1} = \{ 32, 48, 80, 96 \}$ be the order of the block sizes used in $(i-1)^{th}$ round. Such that, B^{i-1}_0 used 32 bits, B^{i-1}_1 used 48 bits,

B^{i-1}_2 used 80 bits and B^{i-1}_3 used 96 bits for encryption respectively. Let d_i be the value obtained from the key for the i^{th} round. Now using the value d_i , permute the block sizes in b_{i-1} to get the new block sizes order b^i to be used in i^{th} round. See algorithm (*IV. h*).

Let the b^i obtained through this process be b^i = { 96, 80, 48, 32 }. Therefore, we notice that in $(i-1)^{th}$ round B^{i-1}_0 block had 32 bits for encryption whereas; in i^{th} round B^i_0 block has 92 bits for encryption. Similarly, block sizes of B^i_1, B^i_2 and B^i_3 will also vary in i^{th} round when compared with block sizes of B^{i-1}_1, B^{i-1}_2 and B^{i-1}_3 of the previous $(i-1)^{th}$ round. Therefore, through this process, we are actually introducing greater confusion and nonlinearity in the process of encryption which enables us in counter attacking the crypt analyzer.

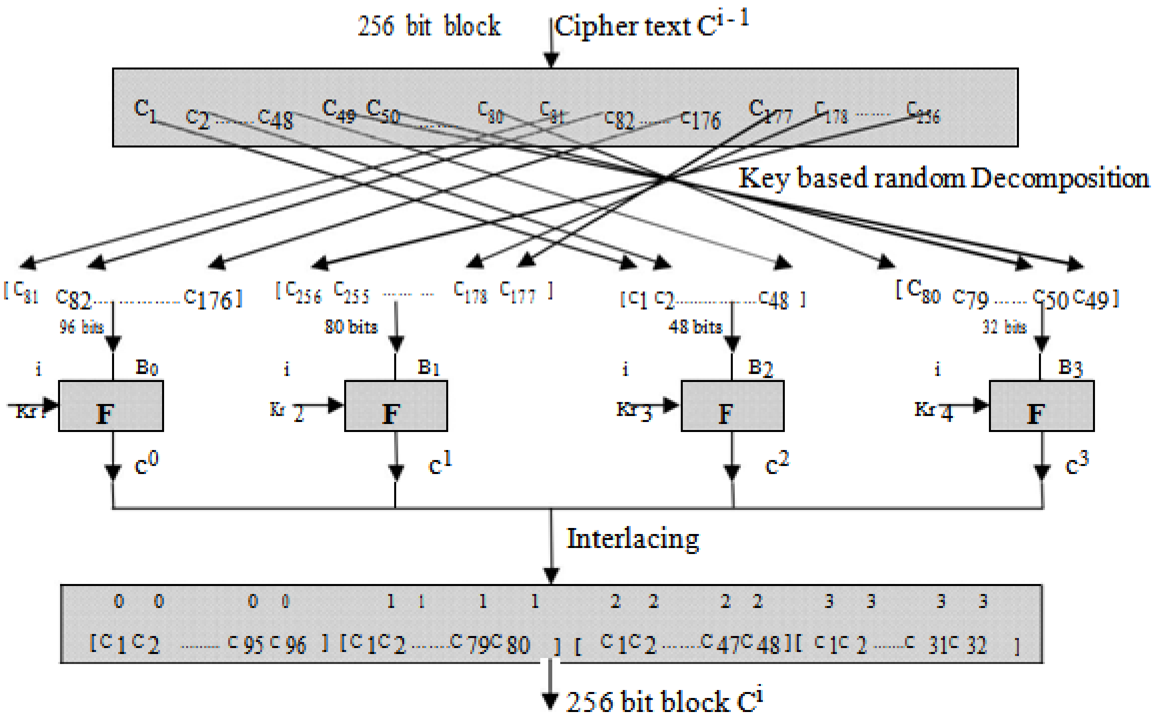
Let K_{i-1}= 56; $d_i = K_{i-1}$ % 4 = 0; b^{i-1} = { 32, 48, 80, 96 }; $m=K_{i-1}\%2$;
order for d_i^{th} block= left to right if m=0 otherwise from right to left;
order for next block if from right to left and vise versa for remaining blocks in cyclic fashion.

Fig.01



Let K_i = 58; $d_i = K_i$ % 4 = 2; b^i = { 96, 80, 48, 32 }; $m=K_i\%2$; order for d_i^{th} block= left to right if m=0 otherwise from right to left;
order for next block is from right to left and vise versa for remaining blocks in cyclic fashion.

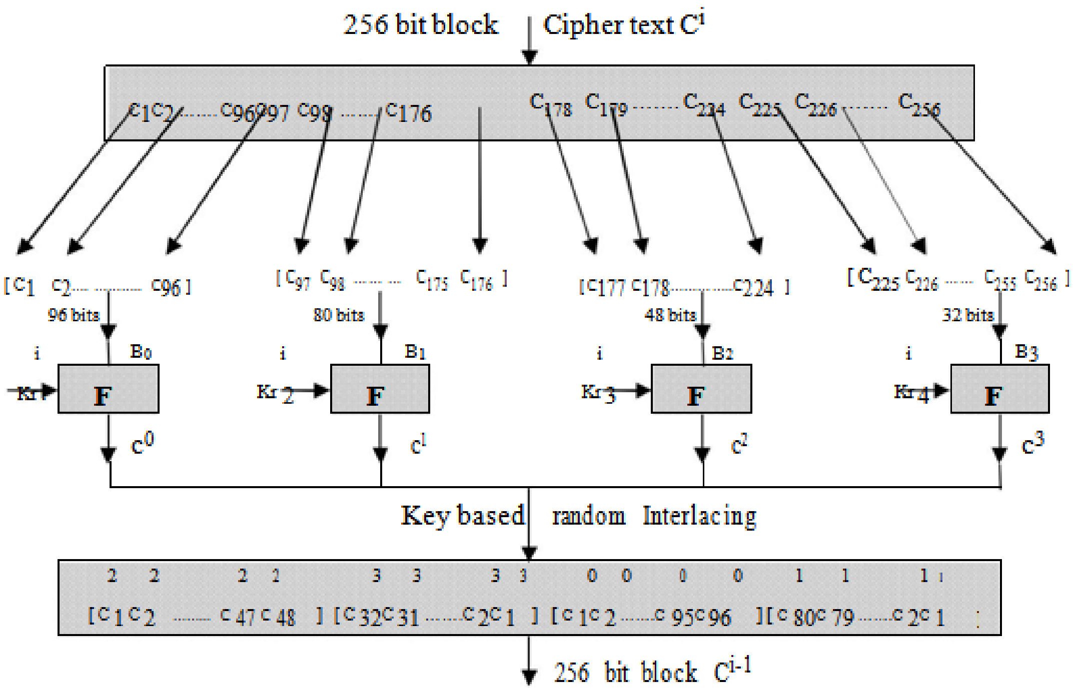
Fig. 02



Variable block sizes in two consecutive rounds during encryption.

The corresponding variable block sizes and its related operations during decryption in i^{th} round and $(i-1)^{th}$ round is demonstrated in the following diagrams.

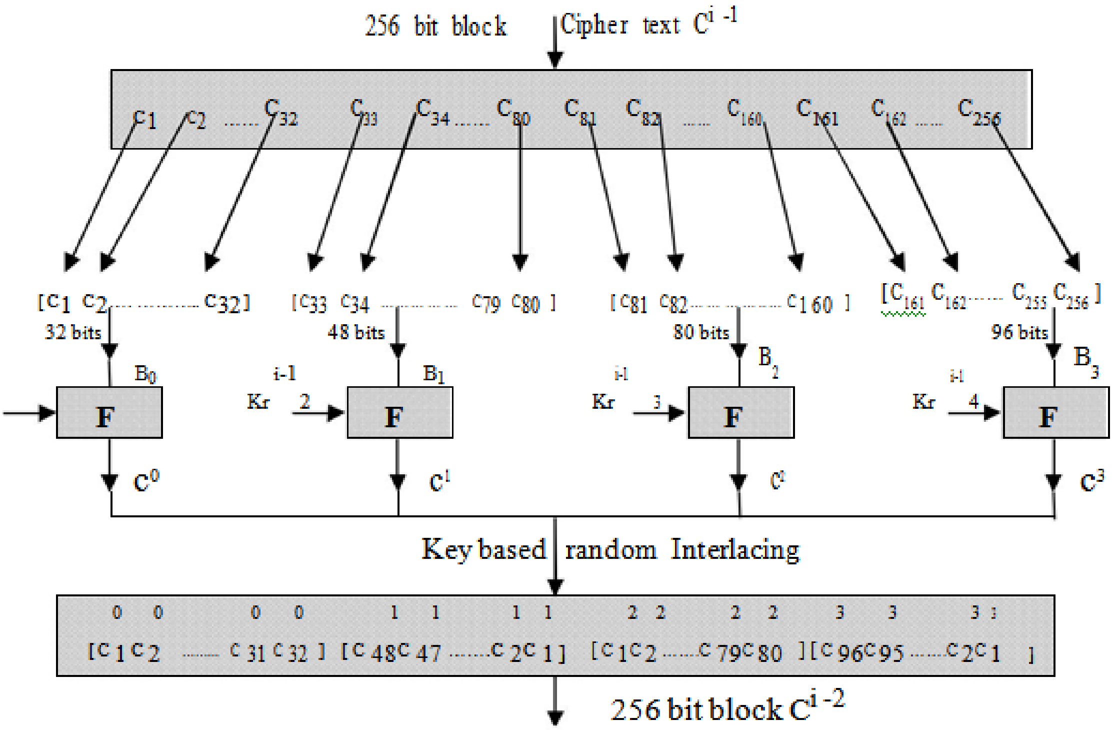
Fig. 03



Decryption in i^{th} round

ISSN : 2248-9622, Vol. 8, Issue 4, (Part -I) April 2018, pp.90-100

Fig. 04



Decryption in (i -1)th round

III. DEVELOPMENT OF CIPHER

Let us consider a block of plaintext 'P' consisting of 32 characters. By representing each character with 8 bits, we get a block of plaintext of 256 bits and denote them as C^0.

Let 'K' be the key containing 16 integers. Then the 8 bit binary equivalent of these integers will give us a block of 128 bit key represented as 'k'.

Let b^0 = {32, 48, 80, 96} be the initial order of block sizes.
Let $d_i = K_i$ mod 4.
Using the algorithm (*IV.h*). Permute b^0 to get the new order of block sizes to be used in respective rounds and denote them as b^1 b^2 b^3 ….. b^{16}.
Next, generate the respective round keys.
Consider i^{th} round and let i = 1.

Then the first block key of round 1 contains $b^1{}_1$ divided by 2 number of bits from 'k' and treat it as k_1.

The second block key of round 1 contains $b^1{}_2$ divided by 2, number of bits from 'k' and treat it as k_2.

Similarly, we get two more block keys of round one as 'k_3' and 'k_4'.
By performing the transformation on k_1, k_2, k_3 and k_4 published in our previous paper, we get the final block keys of respective rounds. Treat them as $kr^1{}_1$, $kr^1{}_2$, $kr^1{}_3$, kr14. See reference [4] for these transformations.

As we use four different blocks B_0, B_1, B_2, B_3 during encryption, $kr^1{}_1$, $kr^1{}_2$, $kr^1{}_3$, $kr^1{}_4$ are used as the respective keys for these blocks.

Now decompose the plaintext C^0 into four blocks B_0, B_1, B_2, and B_3. Start the process of decomposition beginning with (B_{di})th block. Collect the bits from C^0 in sequential manner and place them in B_{di} in respective order.The number of bits collected into the block B_{di} is equal to the block size denoted by $b^1{}_{di}$. Similarly, we get the other three variable size blocks of this round. See algorithm (*IV.e*) for key based random decomposition; fig 01and fig 02.

Let the blocks obtained after key based random decomposition be represented as $B^1{}_0$, $B^1{}_1$, $B^1{}_2$, and $B^1{}_3$. Therefore, Let

B^{m+1}_i = ← C^m → Here, 'm' indicates the round after which decomposition is performed, 'i' indicates the block number;
i = 0 to 3 and ← C^m → indicates key based random decomposition.
Encryption in the n^{th} round is done in the following way.
$c^n_i = F_{kr^n_{i+1}} (B^n_i)$;
i = 0 to 3 indicates i^{th} block.
'F' indicates encryption and kr^n_{i+1} indicates the round key for 'n^{th}' round on i^{th} block and n = m+1.
Next, we perform the process of interlacing after encryption.
After encryption in n^{th} round, we get cipher text as four blocks c^n_0, c^n_1, c^n_2, c^n_3.
Combine the four blocks c^n_0, c^n_1, c^n_2, c^n_3 to get the 256 bit
intermediate block cipher.
C^n = > c^n_i < ;
Here i = 0 to 3 , indicates the cipher block. n = 1 to 16.

Indicates the round after which interlacing is performed.

> c^n_i <, represents interlacing. See Fig 01, Fig 02 and algorithm (*IV.c*) for interlacing during encryption. Similarly, by following the steps of Fig 01, Fig 02 and algorithm (*IV.a*). We get the final cipher C^{16} after encryption of 16 rounds. Similarly, during decryption, the receiver follows the steps of Fig 03, fig 04 and algorithm (*IV.b*) for sixteen rounds to get back the original plaintext.

IV. ALGORITHMS

a) Algorithm for Encryption.

Let K be an array containing 16 integers.

Let d_i be an array containing 16 numbers. Such that, $d_i = K_i$ mod 4 such that, d_i = { 0, 1, 2, 3}.

BEGIN
C^0 = P // initialize 256 bits plaintext
for i = 1 to 16
{
for j = 1 to 4
{ B^{i-1}_{j-1} = ← C^{i-1} → // Key based random Decomposition
}
for j = 0 to 3
{ $c^j = F_{kr^i_{j+1}} (B^{i-1}_j)$ // Encryption
}
for j = 0 to 3
{ C^i => c^j < // Interlace
}
}

END

b) Algorithm for Decryption

C^{16} = cipher text // initialize 256 bits cipher text
BEGIN
for i = 16 to 1
{
for j = 0 to 3
{ B^i_j = < C^i > } // Decompose

for j = 0 to 3
{ $c^i_j = F_{kr^i_{j+1}} (B^i_j)$ // Decryption
}
for j = 0 to 3
{ C^{i-1} = → c^j ← // Key based random Interlacing.
}
}

END

c) Algorithm for Interlacing during Encryption

> c^i_j <

BEGIN

p = 0
s = d_i
While (s is not equal to j)
{ p = p + b^i_s
s = (s + 1) mod 4

}

for n = 1 to b^i_j
{ C^i [p + n] = c^i_j [n]

}

END

d) Algorithm for Decomposition during Decryption

< C^i > // during i^{th} round

BEGIN

```
p = 0

for j = 0 to 3

{
   for   n = 1 to b^i_j
   {       B^i_j [n] = C^i[p + n]
   }
    p = p+n

END
```

e) Algorithm for Key based random Decomposition during Encryption.

```
← C^(i-1) → // during i^th round

BEGIN
t = 0 p = 0 s = d_i

While ( t is not equal to j) {p=p + b^i_s
  s = ( s + 1 ) mod 4
}

j = d_i
if ( ( K_i mod 2 ) = = 0 ) { order = 0

}

else

{ order = 1
}            // 1: R → L       and  0: L → R   order
for m =      1 to    4
{
        if ( order = = 0                              )
      {     for    n    = 1                        to b^i_j
              {     B^i_j                  [n] = C^(i-1)[p + n]
              }
              p = p + n

         j = ( j + 1 ) mod 4

        }

        else
        {   for   n =   b^i_j  to  1
              {
                    B^i_j  [n] = C^(i-1)[p + n]

              }
         p = p + b^i_j

         j = ( j + 1 ) mod 4
            }

}

END
```

f) Algorithm for Key based random Interlacing during Decryption

```
→ c^j ←// during i^th round

BEGIN
p = 0
s = d_i

While ( s is not equal to j)
{  p = p + b^i_s
     s = ( s + 1 ) mod 4
}
if ( ( K_i mod 2 ) = = 0 )

{ order = 0

}

else

{ order = 1

}             // 1: R → L and  0: L → R          order

if ( order = = 0 )

{

    for   n = 1  to        b^i_j
    {  C^(i-1) [ p + n ] = c^i_j[n]
    }

}

else
{   for   n = 1  to      b^i_j
              {  C^(i-1)[p + b^i_j – (n-1)]= c^i_j[ n ]
              }

 }

END
```

g) Algorithm for Key generation of Variable sizes for respective blocks in different rounds.

kr^i_j

BEGIN

for i = 1 to 16

{ p = 1

Left shift (k^{i-1})

k^i<=Permute (k^{i-1}, d_i) // permutations used are published in our previous papers. see reference[4]

for j = 1 to 4

{ for n = 1 to (b^i_j) / 2

{

$kr^i_j[n]$ = k[p]

p = p + 1

}

}

}

END

h) Algorithm for generating Variable block sizes for corresponding rounds.

Let b^0 = {32, 48, 80, 96} be the initial order of block sizes. By permuting this block size order in respective rounds, we get the random block size orders for corresponding sixteen rounds.

Let the order of variable block sizes obtained through this process be,

b^1 = {32,48,80,96}, b^2 = {96,80,48,32}, b^3 = {80,32,48,96}, b^4 = {96,48,32,80}, b^5 = {32,96,80,48}, b^6 = {48,80,96,32}, b^7 = {80,32,96,48}, b^8 = {48,96,32,80}, b^9 = {32,96,48,80}, b^{10}= {96,32,80,48},b^{11}= {32,48,96,80}, b^{12}={48,32,80,96}, b^{13}= {80,96,48,32},b^{14}= {48,32,96,80}, b^{15}={32,48,80,96}, b^{16}= {80, 32, 48, 96}.

BEGIN

for i = 1 to 16

{

b^i <= Permute (b^{i-1}, d_i)

}

// permutations used are published in our previous papers. see reference[4]

Fig 05. Process of Encryption

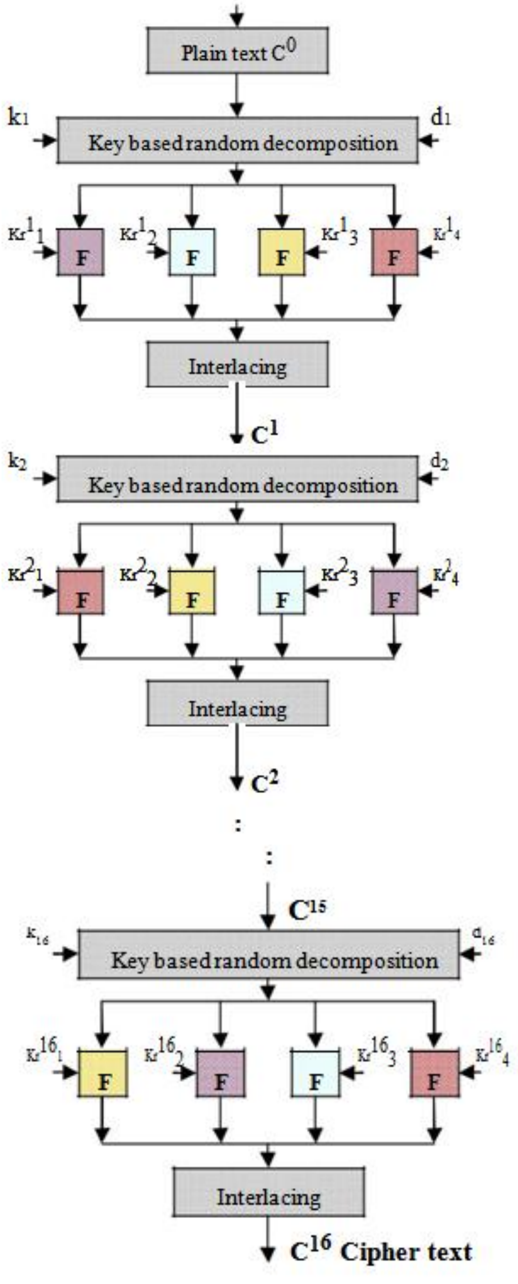
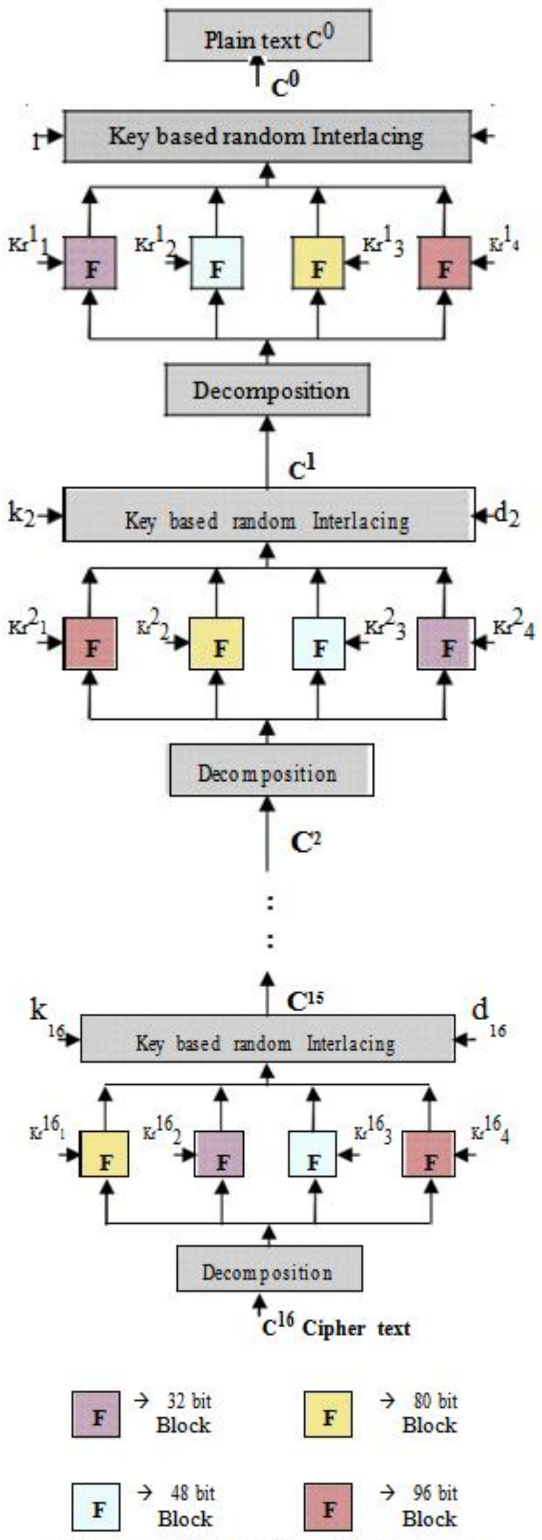


Fig 06. Process of Decryption



Colors used for different block sizes in fig 05 and fig 06

V. ILLUSTRATION OF CIPHER

Consider the plaintext

P = {transfer energy from one to many}.

Let the key be K = {Do u like it sir}.

Let the 8 bit binary representation of plaintext P be
011101000111001001100001011011100111001101
100110
011001010111001000100000011001010110111001
100101
011100100110011101111001001000000110011001
110010
011011110110110100100000011011110110111001
100101
001000000111010001101111001000000110110101
100001 0110111001111001.

Let the 8 bit binary representation of key ‘k’ be
010001000110111100100000011101010010000001
101100
011010010110101101100101001000000110100101
110100 0010000001110011011010010111001O.

Initialize the plaintext C^0 = P.
Let $d_i = K_i \bmod 4$

We get $d_1 = 0$, this indicates that key based random decomposition begins with B^1_0 in first round. As K_1 is an even number, the order for B^1_0 is from left to right, order for B^1_1 is from right to left, and order for B^1_2 is from left to right and right to left for B^1_3.

Let the initial order of variable block sizes be denoted as b^0 = { 32, 48, 80, 96 }.

Now permute the order of variable block sizes in first round. Let the new order of variable block sizes in first round be denoted as b^1 = {96, 32, 48, 80}.

As we use four different blocks B_0, B_1, B_2, B_3 of variable block sizes for encryption, B_0 contains 96 bits, B_1 contains 32 bits, B_2 contains 48 bits, and B_3 contains 80 bits. Use algorithm (*IV.e*) to get these four blocks. Also see Fig.01.

B^1_0={011101000111001001100001011011100111O
011011
001100110010101110010001000000110010101101
110011 00101}
B^1_1 = {00000100100111101110011001001110}
B^1_2={011001100111001001101111011011O100100
000011

01111}

B^1_3={10011110011101101000011010110110000001 00111

10110001011100000001001010011001110110}

Now, Permute the bits in key 'k' by using the random key based permutations published in our previous paper. See reference [4].

Let the key ' k' be divided into four blocks of variable sizes used as round keys kr^1_1, kr^1_2, kr^1_3, kr^1_4 for blocks B_0, B_1, B_2, B_3 respectively. See algorithm (*IV.g*).

Now, we encrypt these four blocks with their respective round keys with the help of round function 'F'. Key based random permutations and key based random substitutions used in a round are similar to the one we derived in our previous paper published. See reference [4].

Let the corresponding cipher blocks obtained after first round
be c^1_0, c^1_1, c^1_2, c^1_3.
c^1_0={01100110011001010111001000100000011001 011000

01010001010110110110110010000100000100000 1001001 1110}
c^1_1 = {11100110010011101010011001110110}
c^1_2={10001010001010111101000010010010001001000 000 0110}\

c^1_3={00100000011101000001010100011111101101100 000 0100111101101010100000001010001110000}

We get the 256 bit cipher block C^1 after first round by applying interlacing described in Fig.01and Fig.02. See algorithm (*IV.c*).

C^1={01100110011001010111001000100000011001 011000
01010001010110110110110010000100000100000 1001001
11101110011001001110101001100111011010001 0100010
10111101000010010010001001000000001100010000 00111
01000001010100011111101101100000001001111011 01010 10000001010001110000.

Similarly, we continue the encryption process up to 16 rounds and we get the final cipher as

C^{16}={10001111001000111011110000100100111001 11101
0010001110111111000111101100111101011001011 010111
10001111011001101011101011011010111010111 0011000
10000101101111001101001100110001101110000 1010011
11110100001110100011011001000110001100010 1101100 000110000011001100111}

In order to decrypt the cipher text, we follow the transformations described in Fig.03 and Fig.04 for sixteen rounds and use algorithm (*IV.b*). Thus, we get back the required original plaintext.

VI. CRYPTANALYSIS

To asses the strength of our encryption algorithms, we analyze the following aspects.

- ✓ Why brute force attack is not possible on our cipher?
- ✓ Why known plaintext attack is not possible on our cipher? And how can variable block sizes counter attack known plaintext attack?
- ✓ How variable block can sizes counter attack brute force attack?
- ✓ How is the avalanche effect when variable block sizes are introduced in feistel cipher?

a) Brute force attack

According to brute force attack, if key space is small, then one can test all possible combinations of keys on encryption-decryption algorithms in a definite time which is acceptable to break the cipher. Therefore, key space should be large enough so that testing of all possible key combinations will take lot of time which is not acceptable in breaking a cipher.

As we have used 128 bit key in each round, the key space is

$$\mathbf{2^{128} \approx (2^{10})^{13} \approx (10^3)^{13} \approx 10^{39}}$$

Let us assume testing of one key on a computer takes 1 nano second. Then testing of 10^{39} keys will take [(10^{39}) / (10^9 x 60 x 60 x 24 x 365)] years. Since one cannot spend so much time in breaking the cipher, brute force attack is not possible on our algorithm.

b) Known Plaintext attack

According to known plaintext attack, if enough number of plaintext – cipher text pairs are available then, one can understand the transformation used in developing the cipher. Our classical feistel cipher with fixed block sizes is prone to known plaintext attack due to the linearity that exists in transformations during encryption. Since we have used variable block sizes in every round, we have restricted the bits to get into different random blocks of different sizes basing upon the key and the round. Through this process, we have introduced a high degree of nonlinearity in our encryption algorithm. Due to this, more confusion and diffusion is added in the cipher. Thus, known plaintext attack is not possible on our algorithm as an attacker is clueless about the number of bits used in different blocks in different rounds. Therefore, bits permuted, XOR'ed and entering into substitution boxes are not clear to the crypt analyzer.

c) How variable block sizes counter attack the brute force attack?

During encryption, in every round, we have used the variable block sizes b^i = {32, 48, 80, 96 } that means, in every round 4!= 24 different block size orders are possible. Similarly, in two rounds 4! x 4! = 24 x 24 = 576 different block size orders are possible. Therefore, in sixteen rounds, the number of block size orders that are possible is

4! x 4! x 4!............. x 4!(sixteen times)

This is equal to

$1211657479094510655 8976 \approx (10)^{22}$

Therefore, if one follows the brute force attack and tries to guess the block sizes in various rounds. They have to test $(10)^{22}$ possibilities. If testing takes 1 nano second for a single possibility on a computer, one would spend time equal to $(10)^{22}$ / [10^9 * 60 * 60 * 24 * 365] years, to understand the exact block size order. Therefore brute force attack is not possible in this case also.

d) Avalanche effect

According to avalanche effect, for a plaintext P, if C^1 is an equivalent cipher then by keeping the key constant, if there is one bit change in plaintext P and we get an equivalent cipher as C^2. Then the strength of the cipher is good if C^1 and C^2 differ by around 50% of the bits. Similarly, the algorithm can be tested for a one bit change in key. Let the plaintext be

P = {transfer energy from one to many}.

Let the key K = {Do u like it sir}

Then by following the process of encryption described in algorithm () and in Fig 01 and Fig 02. We get the following cipher after 16 rounds as

${}_1C^{16}$={1000111100100011101111000010010011100 11110

1001000111011111000111101100111101011001 01101011
1100011110110011010111010110110101110101110 01100

0100001011011110011010011001100011011100001 01001

1111101000011101000110110010001100011000101 10110

000011000001100110 0111}

Now, Let us change the plaintext by one bit. This can be done by changing the first letter in plaintext from 't' to 's' as the ASCII values of 't' and 's' differ by one. Keep the key as constant.
We get the new cipher text for this new plaintext after 16 rounds of encryption as
${}_2C^{16}$={0010110010010011111101111010111111001 01110
10000111011111101010101011101001111100110 0100100

1100011111011110111100111100001010101010000 10110

0100100110110010100110011010011101110111101 11111

0111111010001001000110011100111010011000100 10000

010100111000101110 0110}

From ${}_1C^{16}$ and ${}_2C^{16}$ we find that 120 bits differ out of 256 bits. Since around 50% of the bits differ in corresponding ciphers for a one bit change in plaintext; we say that the strength of the cipher is good when variable block sizes are used.

Now let us keep the plaintext as constant and change the key by one bit. This can be accomplished by changing the key character from 'D' to 'E' as their ASCII values differ by one bit. Let the corresponding cipher obtained after 16 rounds of encryption be
${}_3C^{16}$={1100000000000101010000111101100001000 10001

1101111110110010010011101111000001100100111111111
1110011001001111111010001010001100011010011111010

1010110000111010010100110011010110100000001111100

1011010111011001110011010101110001001011001001111

0110001100001011101101}

From $_{1}C^{16}$ and $_{3}C^{16}$, we readily notice that 146 bits differ out of 256 bits. Therefore for a change in one bit in a key, there is a difference of around 50% of bits in the corresponding ciphers. Thus, the avalanche effect is good for our ciphers when variable block sizes are used in different rounds of encryption-decryption algorithms.

VII.CONCLUSION

In conventional feistel cipher, we observed that known plaintext attack is possible because a set of bits will undergo into similar transformations and enter into same substitution box in each round. Due to this linearity, cryptanalysis becomes easy. In our recent research work, see reference [4, 5], we proved how "random key based permutations and substitutions" and "key based random interlacing and decomposition" bring variable transformations in each round. In the present paper, we have used the strategy of "key based variable block sizes in different rounds" to strengthen the cipher further. This new strategy helps us in making the cryptanalysis more difficult and impossible. The results of avalanche effect discussed in this paper indicates that the "key based variable block sizes" introduced to counter attack the known plaintext attack provides good strength to the cipher.

ACKNOWLEDGMENT

The author is very thankful to the management of Sreenidhi Institute of Science and Technology, for their support and encouragement given during this research work.

REFERENCES

Books:

[1]. William Stallings, "Cryptography and Network Security: Principles & Practices", Third edition, 2003, Chapter 2 & 3.

Journals:

[2]. "Avalanche Characteristics of Substitutions – permutation Encryption Networks" Tavares S. Heys H. IEEE Transactions on Computers 44 (9): 1131 – 1139, 1995. "http://ieeexplore.ieee.org/Xplore/login.jsp?url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel1%2F12%2F9728%2F00464391.pdf%3Farnumber%3D464391&authDecision=-203"

[3]. Shakir M. Hussain and Naim M. Ajilouni, "Key based random permutation", "Journal of Computer Science 2(5): 419 – 421, 2006. ISSN 1549 -3636. "http://www.scipub.org/fulltext/jcs/jcs25419-421.pdf"

[4]. **K. Anup Kumar** and S. Udaya Kumar, "Block cipher using key based random permutations and key based random substitutions", "International Journal Of Computer Science and Network Security", Seoul, South Korea. ISSN: 738-7906. Vol. 08, No. 3, March 2008. pp. 267-277. "http://paper.ijcsns.org/07_book/200803/20080339.pdf"

[5]. **K. Anup Kumar** and V.U.K. Sastry, "Modified Feistel cipher involving Interlacing and Decomposition", "International Journal of Computer and Network Security", Austria, Vienna. ISSN Print: 2076 – 2739; ISSN online: 2076 – 9199. Vol. 1, No. 2, November 2009. pp. 77-82."http://ijcns.org/papers/Vol.1_No.2/091112.pdf"

[6]. **K. Anup Kumar** and V.U.K. Sastry, "Block cipher involving key based random Interlacing and key based random Decomposition", "Journal of Computer Science", NewYork. , ISSN Print: 1549 – 3636; Vol.6, No.2, 2010. pp.133-140 http://www.scipub.org/fulltext/jcs/jcs62133-140.pdf

Dr. K. Anup Kumar"A Novel Method of Encryption Using Variable Block Sizes in Different Rounds."InternationalJournal of Engineering Research and Applications (IJERA) , vol. 8, no. 4, 2018, pp. 90-100