International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 5 (2018) pp. 2847-2856

Inspired Feistel DNA Based Crypto System Using D-Box

P.Bharathi Devi[1] and Dr.R.Kiran Kumar[2]

[1]Research Scholar, Department of Computer Science, Krishna University, Machilipatnam, Andhra Pradesh, India.
[2]Department of Computer Science, Krishna University, Machilipatnam, Andhra Pradesh, India.

Abstract

DNA Cryptography is one of the encouraging aspects in the field of DNA Computing. DNA Computing or Bio Molecular Computing is used to describe the properties of DNA in parallel computation. DNA is used not only as a computation tool but also as an information carrier. The inherent property of storing large information nearly Terabytes of data into a single gram of DNA, the random nature of DNA making the code unbreakable made the researchers give higher priority to DNA Computation. As we all know traditional crypto algorithms are broken, there is a need of harnessing the biology, chemistry and computer science into single discipline so that it gives better security to the information.

In this paper, we proposed a model which uses Feistel Cipher implementation and adding the Shannon properties of Confusion and Diffusion using DNA Codons. One more characteristic of the proposed scheme is it used randomly generated key in each round so that it is highly impossible to detect the plaintext in a cryptanalytic attack. It is also observed that when the performance of the proposed algorithm is measured through avalanche effect it is found that it is better than some of the existing algorithms.

Keywords: DNA Cryptography, Symmetric Cryptography, Feistel Cipher, DNA Codons.

INTRODUCTION

The advancements in the technology look towards the improvement in the security issues. Many of the fields like banking sector, e-mailing system, etc., needs to increase the security policies to gain trust from the customers. To provide security to the information, they use cryptographic techniques. Cryptography is an art of converting the text information into unreadable format. The process of converting the text data into unscrambled message is called as Encryption process and the reverse process is called as decryption process. The unscrambled message is nothing but cipher text. Many cryptographic approaches exist to provide security to the information, namely Modern Cryptography, Quantum Cryptography and DNA Cryptography. In modern cryptography, it faced difficulty in prime factorization and in Quantum cryptography deals with photons and the signals are limited to 90 miles [1]. After that, Gehani et.al.[2] discovered the DNA structure used to provide security to the information. He designed DNA based cryptography with substitution and One Time Pads. DNA cryptography which is a branch of DNA computing is a new technique which secures the data using biological structure. DNA Computing uses DNA sequences for computation instead of traditional silicon based chips. The year 1994 is the new era in the field of computer science when Adleman used the bio molecular technology to solve the mathematical complex problem such as Hamiltonian path [3] which is a NP-complete problem. He demonstrated the problem using seven cities and each city named as a DNA sequence. By using the process of DNA synthesizing, he generated all possible routes within lesser time when compared to traditional approach and this process is called massive parallelism. The approach of massive parallelism using DNA helped many researchers to solve several NP Hard problems. With the inspiration of Adleman, Lipton showed a Boolean formula to solve the satisfiability problem known as SAT[4] using DNA. Solving complex mathematical problems in all possible solutions at once is the central idea of a DNA computer, since it stores 700 terabytes data in 1g of DNA which is proved by George Church's team of Harvard University[5]. The idea behind using cryptography in the arena of biology is to come up with a remarkable technology which provides unbreakable algorithms. But the limitation of these algorithms are that it lacks the theoretical approach which become a hurdle to model the good DNA cryptographic schemes[6]. The present paper provides the DNA based security based upon the traditional Feistel cipher that uses the D-box, a key element in this algorithm and the encryption process is done based on the number of rounds required. The key is also generated randomly in each round. The detailed process is described in the next sections. The Section2 describes the related works done based on DNA cryptography. In Section 3 and 4 the scope of proposed scheme and the implementation results are shown. The Section 5 and 6 describes experimental results, the performance analysis of proposed algorithm and comparison of avalanche effect of existing algorithms with proposed scheme. Finally, Section 7 is the conclusion of the paper.

BIOLOGICAL BACKGROUND

DNA is an acronym for Deoxyribonucleic Acid which is a hereditary element in human beings and all other living organisms. It is made up of nucleotides. Each nucleotide contains a Sugar group, a Phosphate group and a Nitrogen base. The four Nitrogen bases which are called as Adenine (A), Thymine (T), Cytosine(C) and Guanine (G)(Figure 1). The entire information is represented in the form of these DNA four bases as against to the traditional methodology (0's and 1's)[7,8]. Generally, DNA exists as double helix in its nature by pairing the complimentary bases from a single strand. These DNA bases pair up i.e., A with T and C with G according to Watson-Crick Complementary rule. These pairs can be called as base pairs. The central dogma of biological

structure for processing the genes into proteins is described in two step process: one is transcription and other is translation. In the transcription phase the DNA bases are converted into RNA having four bases. The difference between DNA and RNA is, instead of Thymine, Uracil (U) is used in RNA. In the translation phase each three nucleotide bases can form an amino acid which is useful in protein synthesization. The amino acid also called as Codon. There are total 64 codons can be formed from the 4 bases. All 64 Codons can form only 22 amino acids[9,10].

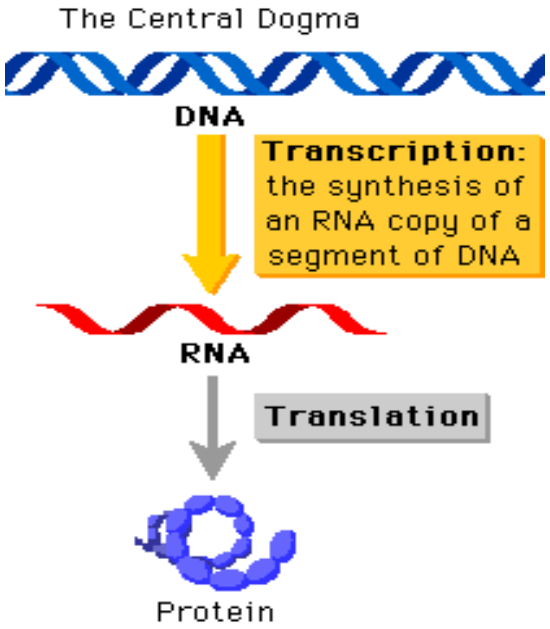


Figure 1: Central Dogma of DNA

Table 1: DNA Table

Base	Gray Code
A	00
C	01
G	10
T	11

RELATED WORK

The field of network security always looks for unbreakable cryptographic algorithms to protect the information while storing the data on a cloud or while passing the data on a network or some other security issues. In this regard, many researchers performed more work on cryptographic approaches and found that DNA carries the information more confidential and the size of DNA strand is large than the traditional methods. There are numerous works taking place for the last few years. In the beginning, Gehani et.al., introduced DNA based cryptography in which he used two methods. One is based on OTP that is One Time Pad is one of the encryption technique which cannot be cracked and the other one is steganography method [2]. Sherif et.al., implemented an encryption algorithm, each plaintext character replaces with four DNA nucleotides or the single pixel of an image replaced by four DNA nucleotides[11]. Fatma et.al., proposed a new DNA Cryptographic algorithm which used amino acid codes for implementing OTP[12]. Carbon nanotube based information passage and DNA based cryptosystem was proposed by Jie Chen[13]. Pakaj Rajkheja added a DNA cipher layer to the traditional cryptography algorithm that is IDEA. In this algorithm the Key size is incremented to immune the information from cryptanalytic attacks[14]. LAI Xuejia, proposed DNA-Public Key Cryptography which provides digital signature along with encryption[15].

Now, in this paper an effort has been done to adding DNA cipher layer to traditional Feistel Cipher. Since several years there had been numerous modifications taking place in the classical Feistel cipher. It is a symmetric cryptography algorithm[16] which contains 64-bit message as a plaintext. In traditional Feistel cipher the plaintext is divided into two blocks, each block containing 32-bits. The left part L0 contains 32-bits and the right part R0 contains 32-bits. Apply round function with Key value and the right part of plaintext. A new key value is generated for each round. The derived result will be XOR with L0. Now, interchange the R0 and L0 values. Continue the same process for n number of rounds which is described as follows.

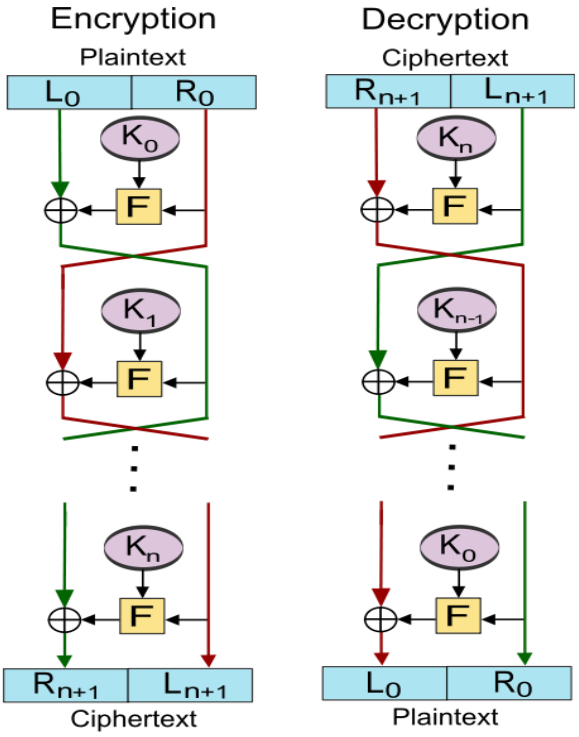


Figure 2: Traditional Feistel Cipher[16]

1. Divide the plaintext block into two equal parts(L_0,R_0).
2. For each round i=1,2,3,…..n
   $L_{i+1}=R_i$
   $R_{i+1}=L_i \oplus F(R_i, K_i)$
   Now the cipher text is (R_{n+1}, L_{n+1}).

In the decryption process perform the reverse process that is,

1. $R_i=L_{i+1}$
2. $L_i=R_{i+1} \oplus F(L_{i+1},K_i)$

Then L_0, R_0 is the plaintext. Later it is modified by many researchers.

FEISTEL INSPIRED DNA BASED CRYPTOSYSTEM

DNA cryptography when combined with Feistel cipher gives more security and can be utilized in DNA computers. The algorithm can be implemented by appending the DNA codon table which is called as D-box (Figure 4), to the traditional Feistel Cipher. The D-Box is used as Substitution box where the codons are arranged in 4^3 ways. A codon is a unit that consists of three adjacent bases on a DNA molecule and that determines the position of a specific amino acid in a protein molecule during protein synthesis. Out of 64 codons 61 codons are used to form 20 amino acids which can be represented as 20 alphabets (A, B, C…) and 3 codons are used as stop signals. For example, Phenylalanine amino acid (F) can be replaced with either TTT or TTC giving rise to ambiguity when these amino acids are used as an alphabet letter. So, in the present work, the names of amino acids are not mentioned and the 64-codons (Figure 3) are arranged in the D-BOX (8x 8 matrixes) in a random manner. The number of codons formed is 64 permutations of 3 DNA bases from the set of 4 DNA bases. The algorithm is implemented for 64-bit block. If the bits are less, zeroes are padded to make it 64.

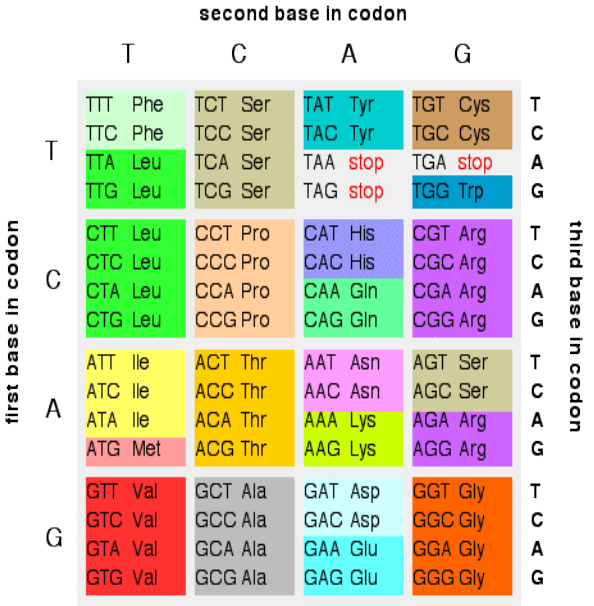


Figure 3: Structured Codon Table

The proposed cryptosystem follows two functions one is E_K (M) =C which describes the encryption process and D_K(C) =M describes the decryption process. The plaintext (M) can be any character but the cipher text C ∈ {A,C,G,T}*. First, the sender and receiver must agree for number of rounds to be performed and the random generated key for each round. The random generated Key K ∈{0, 1}*.

Suppose that Alice wants to transmit a message to the Bob, both agree for number of rounds to be performed and then Alice transmits the plaintext message into binary sequence(64 bit data) and split them into 32-bit each. Generate random key for each round and store them securely. Perform XOR operation for the generated key and the right part (R_i) of the text. Then, pad 4 zeroes to the result to convert them into octal values. Based upon these values find the codon which is placed on the D-Box. Now the DNA sequence is converted into binary sequence (A-00, C-01, G-10, T-11)(Table1). Remove the last four padded bits and interchange Left and Right parts. After n rounds convert the binary sequence into DNA sequence which is cipher text sent to the Bob. The cipher text is transmitted to Bob through public channel and the stored keys through secure medium. Bob perform the reverse process and get the original plaintext message.

4.1 Algorithm for Encryption

1. Read M, n
2. Convert the Plaintext M of 64-bit block into its equivalent Binary values.
3. Split into L_1 and R_1 each of 32-bits respectively.
4. For i=1 to n
   Begin
   a. Generate Key (K_i) Randomly.
   b. $R_i=R_i \oplus K_i$
   c. Pad "0000" to R_i
   d. Split them into 3-bits string each and convert them into its equivalent octal value (We get total 12 strings out of 36 bits).
   e. By taking two octal numbers we get 6 codons from the D-box(Refer point(c) in each round of encryption process).
   f. Convert the above DNA strand into its equivalent binary according to DNA table.
   g. Remove the last "1111" and the value is stored in R_i.
   h. Interchange L_i, R_i.
   end
5. $C=L_i||R_i$.
6. Convert C into DNA bases(00-A,01-C,10-G,11-T), the cipher text.

D-BOX	0	1	2	3	4	5	6	7
0	TTT	TTC	TCT	TTG	TTA	TCC	TCA	TCG
1	TAT	TAC	TGT	TAG	TAA	TGC	TGA	TGG
2	ATT	ATC	ACT	ATG	ATA	ACC	ACA	ACG
3	CAT	CAC	CGT	CAG	CAA	CGC	CGA	CGG
4	CTT	CTC	CCT	CTG	CTA	CCC	CCA	CCG
5	AAT	AAC	AGT	AAG	AAA	AGC	AGA	AGG
6	GTT	GTC	GCT	GTG	GTA	GCC	GCA	GCG
7	GAT	GAC	GGT	GAG	GAA	GGC	GGA	GGG

Figure 4: D-BOX

For e.g., the Codon which is placed in 4th row and 6th column is CCA.

4.2 Algorithm for Decryption

1. Read C, n
2. Convert C into binary from(A-00,C-01,G-10,T-11) and store the same in C.
3. Split the cipher text C into L_n and R_n each of 32-bits respectively.
4. For i=n to 1

Begin

i) Pad "1111" to L_i.

ii) Convert the binary data into its equivalent DNA bases according to DNA table.

iii) Split the DNA strand into codons.

iv) Find the position of each codon from the D-Box.

v) Convert the positions into its equivalent binary values.

vi) Remove last "0000" from L_i.

vii) $L_i=L_i \oplus K_i$

viii) Interchange L_i & R_i.

End

5. $M=L_i||R_i$
6. Convert the above binary data into its equivalent ASCII Characters which is the plaintext M.
7. Write M.

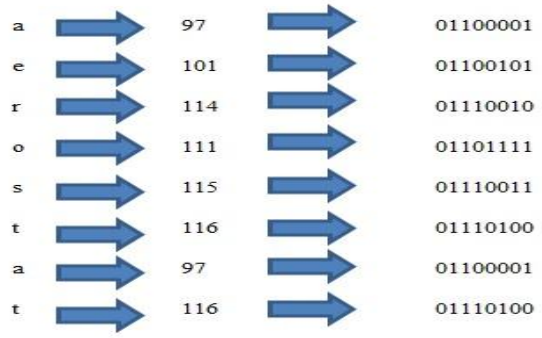
ALGORITHM IMPLEMENTATION

This section demonstrates the detailed implementation of proposed algorithm for the plaintext "aerostat". The section 5.1 represents the encryption process which converts the given plaintext into cipher text and the section 5.2 represents the decryption process which converts the cipher text into plaintext.

Encryption Process

Let the Plaintext M=aerostat

1. Convert the text into its equivalent binary values
2. Split them into 32-bits each and named as L1 and R1.
3. Perform the following step for n number of rounds(let n=4)



ROUND1

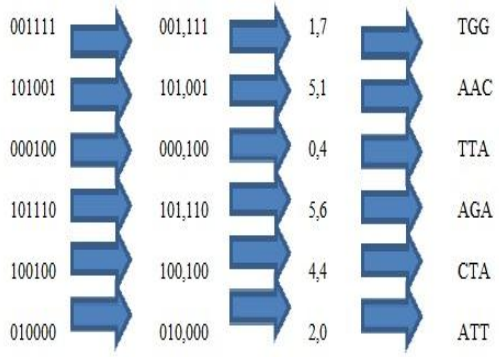
a) Generate the key randomly for each round.

K1=01001101111001010100110111100101

b) Perform XOR operation between R1⊕K1 and the result stored in R1. i.e.,
R1=01110011011101000110000101110100
K1=01001101111001010100110100000101
R1=00111110100100010010110001110001

c) Divide the result into six bits each. Here total numbers of bits are 32, when we divide 32 with 6 it leaves the remainder 2. To make this remainder 2 as six we pad 0000. Convert three bits of them into equivalent octal values. Put the equivalent Codon form D-Box.

d) Convert the above DNA strand into equivalent binary from DNA table.



TGGAACTTAAGACTAATT
11101000000111110000100001110000 1111

e) Remove the last 1111 from the DNA strand and store the 32-bit result into R1.
R1=11101000000111110000100001110000

f) Interchange L1 and R1 and store the result in plaintext M.
Now
M=11101000000111110000100001110000011000010110011010111001001101111.

ROUND2

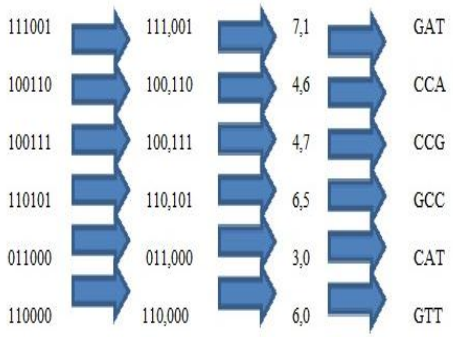
a) Generate the key randomly for this round.

K2=10000111000011001000011100001100

b) Perform XOR operation between R2⊕K2 and the result stored in R2. i.e.,

R2=01100001011001010111001001101111
K2=10000111000011001000011100001100
R2=11100110011010011111010101100011

c) Divide the result into six bits each. Here total numbers of bits are 32, when we divide 32 with 6 it leaves the remainder 2. To make this remainder 2 as six we pad 0000. Convert three bits of them into equivalent octal values. Put the equivalent Codon form D-Box



d) Convert the above DNA strand into equivalent binary from DNA table.

GATCCACCGGCCCATGTT
10001101010001011010010101010011101111

e) Remove the last 1111 from the DNA strand and store the 32-bit result into R2.

R2=10001101010001011010010101001110

f) Interchange L2 and R2 and store the result in plaintext M.

Now
M=10001101010001011010010101001110111
10000001111100001000011100000.

ROUND3

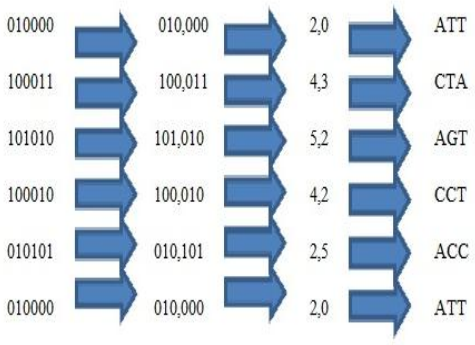
a) Generate the key randomly for this round.

K3=10101010001001011010101000100101

b) Perform XOR operation between R3⊕K3 and the result stored in R3. i.e.,

R3=11101000000111110000100001110000
K3=10101010001001011010101000100101
R3=01000010001110101010001001010101

c) Divide the result into six bits each. Here total numbers of bits are 32, when we divide 32 with 6 it leaves the remainder 2. To make this remainder 2 as six we pad 0000. Convert three bits of them into equivalent octal values. Put the equivalent Codon form D-Box



d) Convert the above DNA strand into equivalent binary from DNA table.

ATTCTGAGTCCTACCATT
00111101111000101101011100010100111 1

e)

f) Remove the last 1111 from the DNA strand and store the 32-bit result into R3.

R3=00111101111000101101011100010100

g) Interchange L3 and R3 and store the result in plaintext M.

Now
M=00111101111000101101011100010100100011101010001011010010101001110.

ROUND4

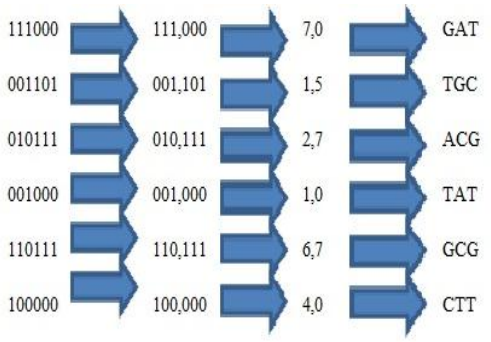
a) Generate the key randomly for each round.

K4=01101101100100000110110110010000

b) Perform XOR operation between R4⊕K4 and the result stored in R4. i.e.,

R4=10001101010001011010010101001110
K4=01101101100100000110110110010000
R4=11100000110101011100100011011110

c) Divide the result into six bits each. Here total numbers of bits are 32, when we divide 32 with 6 it leaves the remainder 2. To make this remainder 2 as six we pad 0000. Convert three bits of them into equivalent octal values. Put the equivalent Codon form D-Box.



d) Convert the above DNA strand into equivalent binary from DNA table.
GATTGCACGTATGCGCTT
1000111110010001101100111001100111111.

e) Remove the last 1111 from the DNA strand and store the 32-bit result into R4.
R4=10001111100100011011001110011001

f) Interchange L4 and R4 and store the result in plaintext M.
Now
M=10001111100100011011001110011001001111011110001011010111000101000.

4. After n rounds convert the Plaintext M into DNA form and store in to Cipher text C
Now Cipher text C=GATTGCACGTATGCGCATTCTGAGTCCTACCA.

Decryption Process

The cipher text sent by the sender C=GATTGCACGTATGCGCATTCTGAGTCCTACCA and convert this into binary form C=1000111110010001101100111001100100111101111000101101011100010100.

1. Perform the step for 4 rounds.

ROUND4

a) Divide the binary string into two parts, each part contain 32-bit each and name it as L4 and R4.
L4=10001111100100011011001110011001
R4=00111101111000101101011100010100

b) Take the left part L4.
L4=10001111100100011011001110011001.

c) Convert the above binary strand into its equivalent DNA base as per the DNA table. Pad 1111 to this for converting them into equal length codons.

10001111100100011011001110011001111
1
GATTGCACGTATGCGCTT
GAT,TGC,ACG,TAT,GCG,CTT

d) Find the location of each codon in the D-Box and convert the values into equivalent binary values and remove last four zeroes to perform XOR operation.
7 0, 1 5, 2 7, 1 0, 6 7, 4 0
11100000110101011100100011011110000

e) Perform XOR operation with K4 and store it into L4. i.e., L4=L4⊕K4.
L4=11100000110101011100100011011110
K4=01101101100100000110110110010000
L4=10001101010001011010010101001110

f) Interchange L4 and R4. Now the cipher text C=00111101111000101101011100010100100011101010001011010010101001110

Round3

a) Divide the binary string into two parts, each part contains 32-bit each and name it as L3 and R3.
L3=00111101111000101101011100010100
R3=10001101010001011010010101001110

b) Take the left part L3.
L3=00111101111000101101011100010100

c) Convert the above binary strand into its equivalent DNA base as per the DNA table. Pad 1111 to this for converting them into equal length codons.

00111101111000101101011100010100111
1
ATT,CTG,AGT,CCT,ACC,ATT

d) Find the location of each codon in the D-Box and convert the values into equivalent binary values and remove last four zeroes to perform XOR operation.
2 0, 4 3, 5 2, 4 2, 2 5, 2 0
01000010001110101010001001010101000

e) Perform XOR operation with K3 and store it into L3.i.e., L3=L3⊕K3.
L3=01000010001110101010001001010101
K3=10101010001001011010101000100101
L3=11101000000111110000100001110000.

f) Interchange L3 and R3. Now the cipher text C=10001101010001011010010101001110111010000001111100001000011100000.

Round2

a) Divide the binary string into two parts, each part contains 32-bit each and name it as L2 and R2.
L2=10001101010001011010010101001110
R2=11101000000111110000100001110000

b) Take the left part L2.
L2=10001101010001011010010101001110

c) Convert the above binary strand into its equivalent DNA base as per the DNA table. Pad 1111 to this for converting them into equal length codons.

10001101010001011010010101011101111

GAT, CCA, CCG, GCC, CAT, GTT

d) Find the location of each codon in the D-Box and convert the values into equivalent binary values and remove last four zeroes to perform XOR operation.
7 1, 4 6, 4 7, 6 5, 3 0, 6 0
11100110011010011111010101100011000 0

e) Perform XOR operation with K2 and store it into L3. i.e., L2=L2⊕K2.
L2=11100110011010011111010101100011

K2=10000111000011001000011100001100

L2=01100001011001010111001001101111

f) Interchange L2 and R2. Now the cipher text C=11101000000111110000100001110000011 0000101100101011100100110 1111.

Round1

a) Divide the binary string into two parts, each part contains 32-bit each and name it as L1 and R1.
L1= 11101000000111110000100001110000
R1=01100001011001010111001001101111

b) Take the left part L1.

L1= 11101000000111110000100001110000

c) Convert the above binary strand into its equivalent DNA base as per the DNA table. Pad 1111 to this for converting them into equal length codons.

11101000000111110000100001110000 1111

TGG, AAC, TTA, AGA, CTA, ATT

d) Find the location of each codon in the D-Box and convert the values into equivalent binary values and remove last four zeroes to perform XOR operation.
1 7, 5 1, 0 4, 5 6, 4 4, 2 0
00111110100100010010111010010001 0000

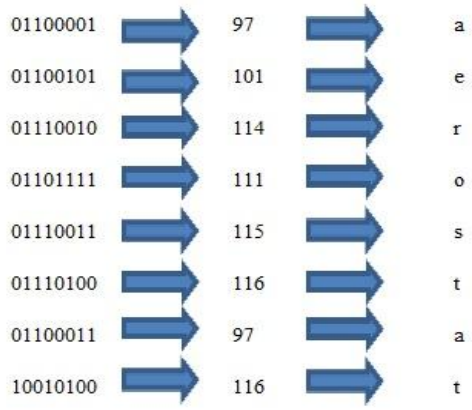
e) Perform XOR operation with K1 and store it into L1. i.e., L1=L1⊕K1.
L1=00111110100100010010111010010001

K1=01001101111001010100110100000101

L1=01110011011101000110001110010100

Interchange L1 and R1. Now the ciphertext C=01100001011001010111001001101111011 1001101110100011000111001 0100.

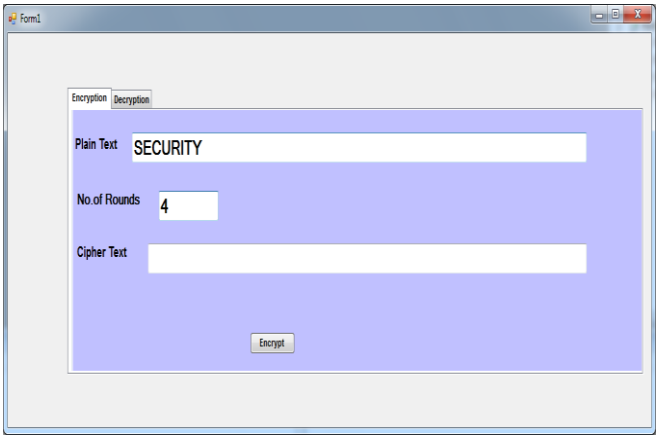
2. Converting the above cipher text into its equivalent ASCII Values, we get the original plaintext "aerostat".

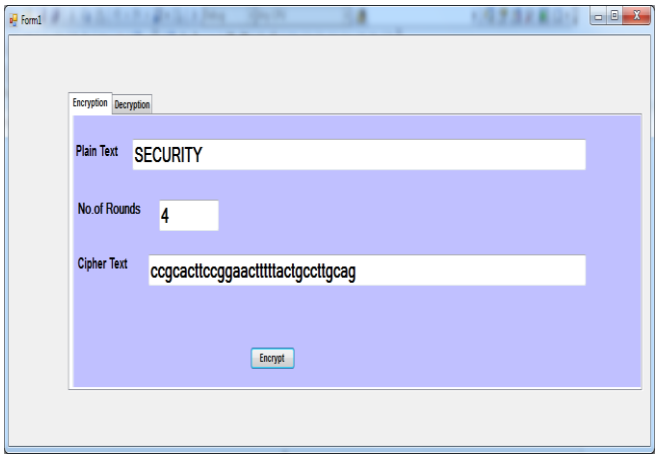


PERFORMANCE ANALYSIS OF FEISTEL INSPIRED DNA BASED CRYPTOSYSTEM

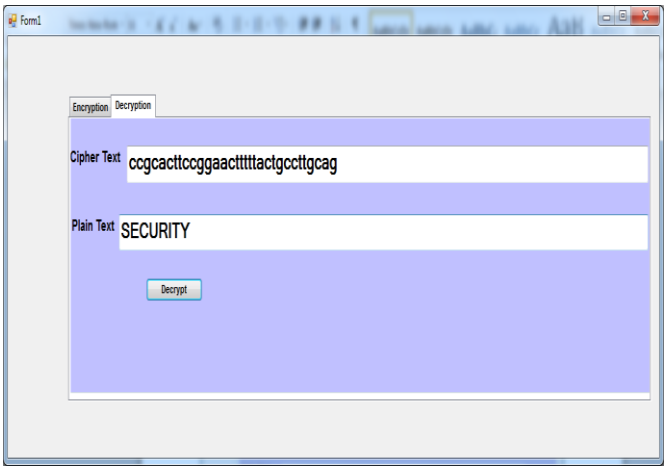
The above project is implemented using .NET environment and the simulations are performed by using .net programming on Windows 7 system. The system configuration used is core i3 processor/4 GB RAM and the results are displayed as below.

Encryption Process





Decryption Process



The following is the table (Table 2 & Table 3) which shows the time taken to encrypt and decrypt the text in Milliseconds. The performance analysis is compared with the existing algorithms and an effort has been done to propose a better algorithm. The analysis is observed for each round and it is witnessed that if the number of rounds increased then the time taken for encryption and decryption of text also increased.

Table 2: Performance analysis in relation to the number of rounds for the encryption of the "security" string.

Number of Rounds	Encryption Time for Proposed Algorithm(in ms)	Encryption Time for Traditional Approach(in ms)
4	0.0124734	0.0042995
5	0.0132133	0.0044041
6	0.0138891	0.0046298
7	0.0148977	0.0049995
8	0.0163741	0.0054139
9	0.0170691	0.0057232
10	0.0190141	0.0063381

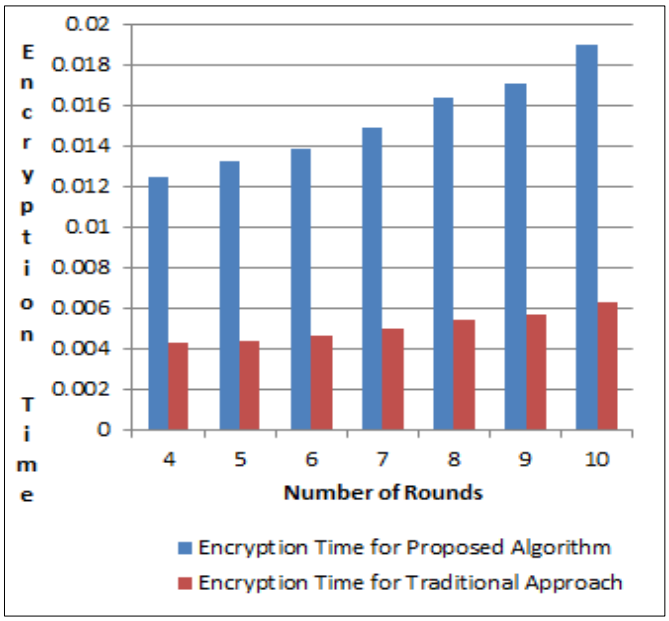


Figure 5: Performance analysis in relation to the number of rounds for the encryption of the "security" string.

No.of Rounds	Decryption Time for Proposed Algorithm	Decryption Time for Traditional Approach
4	0.0038991	0.0011874
5	0.0052451	0.0019017
6	0.0076595	0.0025531
7	0.0090701	0.0030923
8	0.009899	0.0033039
9	0.0108781	0.0036529
10	0.0119875	0.0039621

Table 3: Performance analysis in relation to the number of rounds for the decryption of the cipher text "GATTGCACGTATGCGCATTCTGAGTCCTACCA".

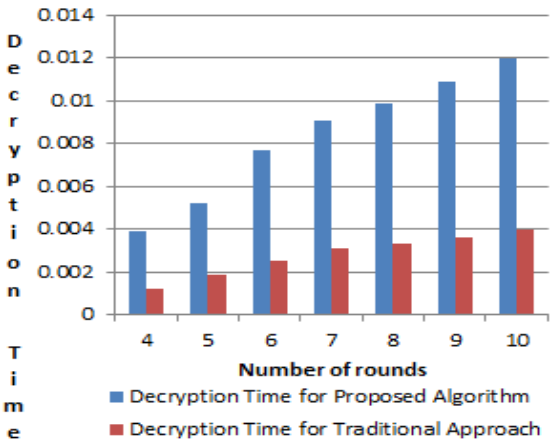


Figure 6: Performance analysis in relation to the number of rounds for the decryption of the cipher text "GATTGCACGTATGCGCATTCTGAGTCCTACCA".

AVALANCHE EFFECT

In cryptography, the avalanche effect refers to a desirable property of cryptographic algorithms. The avalanche effect is the outcome result , when an input is modified slightly(for example, flipping a single bit) the output changes consequently(Example half of the output bits flipped). In the case of high-quality block ciphers, such a small change in either the key or the plaintext is too difficult to afford.

The avalanche effect calculated by using the formula

$$Avalanche\ Effect = \frac{Number\ of\ Flipped\ bits\ in\ cipher\ text}{Number\ of\ bits\ in\ the\ cipher\ text} X100\%$$

The following is the calculation of avalanche effect to various cryptographic techniques and the proposed work. Let us take

Plaintext1:aerostat

Plaintext2:aerrstat

Key: eggplant

1. **Caesar Cipher**
   Ciphertext1:01100110011010100111011101101000111100001111001011001100111001
   Ciphertext2:01100110011010100111011101101101111000011110010110011001111001
2. **Playfair Technique**
   Ciphertext1:01100111011000010111001101110001011100010110001101110100010011010000
   Ciphertext2:01100111011000010111100101111001011100010110001101110100010011010000
3. **Vigenere Cipher**
   Ciphertext1:01100101011010110111100001100100011001000111010001101110011011101
   Ciphertext2:01100101011010110111100001100111001000111010001101110011011101
4. **Blowfish**
   The number of bits flipped in the cipher text in this algorithm is approximately 19 bits and the average percentage of avalanche effect is 29.68[17].
5. **Feistel Inspired DNA based crypto system**

   Ciphertext1:
   1001000011111011001110111100111100001100111000011111001101111100

   Ciphertext2:000010011010001010001001110010010100101100000100011001001001100001000001

The following table(Table 4) shows the comparison results of various crypto algorithms and the proposed algorithm.

Table 4: Comparison results of Avalanche Effect for strings "aerostat" and "arrstat"

Name of the Algorithm	Number of bits flipped	%
Ceaser Cipher	2	3.125
Playfair	3	4.69
Vigener Cipher	2	3.125
Blowfish	19	29.68
Feistel Inspired DNA based Cryptosystem	**34**	**53.12**

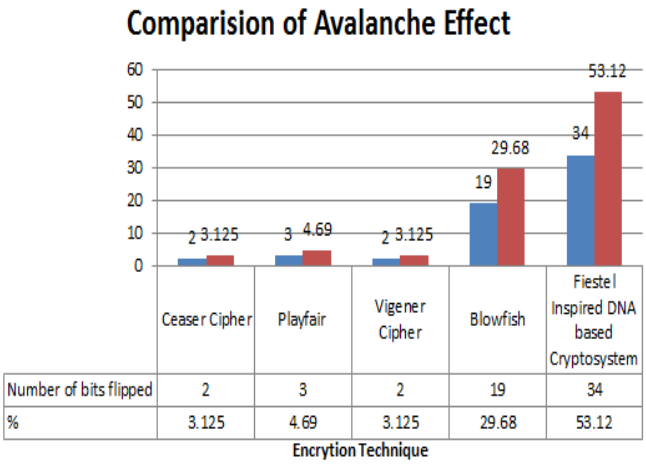


Figure 7: Comparison of the results of the avalanche effect for strings "aerostat" and "arrstat"

CONCLUSIONS

In this paper, we developed a symmetric block cipher algorithm as same as Feistel cipher adding a layer of DNA codons to it. Take a plaintext of 64-bit length which is further divided into two parts and perform XOR operation with randomly generated key with the right part and retrieve the codon values from the D-Box and then convert the DNA strand into its equivalent binary values, which is the cipher text. At the receiver end, the cipher text is divided into two blocks. Consider the left part and convert it into equivalent DNA bases and split them into Codons. Find the position of codons in the D-Box and convert the positions into equivalent binary values, perform the left bit XOR with the key value. It has been observed that, in the block of 64-bit plaintext more than half of the bits have changed in the cipher text in each round. Hence, it makes highly impossible for the intruders to identify the plain text. Further, comparison of performance analysis is done for the existing and proposed algorithms. Avalanche effect is also studied and recorded. Despite the fact that DNA is termed as a medium for ultra- compact information storage and the future appears promising to fascinate the researchers in the area of research , there are still some of the issues like-requirements of quantum attacks

and bio molecular labs that need to be taken into consideration.

REFERENCES

[1] Beenish Anam, Kazi Sakib, Md.Alamgir Hossain and Keshav Dahal, 2010, "Review on the Advancements of DNA Cryptography," in 4th International Conference on Software, Knowledge, Infomration Management and Applications(SKIMA'10).

[2] A.Gehani, T.H.LaBean, and J.H.Reif, 1999, "DNA based Cryptography," in proceedings 5th DIMACS Workshop on DNA Based Computers, pp.233-249.

[3] L.M.Adleman, 1994, "Molecular computation of solutions to combinatorial problems," Science, Vol.266, no.5187, pp.1021-1024.

[4] R.J.Lipton,1995, "Using DNA to solve NP-complete problems," Science, vol.268, pp.542-545.

[5] https://www.extremetech.com/extreme/134672-harvard-cracks-dna-storage-crams-700-terabytes-of-data-into-a-single-gram.

[6] E.Suresh Babu, C.Naga Raju and Munaga HM Krishna Prasad, 2016, "Inspired Pseudo Biotic DNA Based Cryptographic Mechanism Against Adaptive Cryptographic Attacks," International Journal of Network Security, Vol.18, No.2, PP.291-303.

[7] Noorul Hussain UbaidurRahman, Chithrealekha Balamurugan and Rajapandian Mariappan, 2015, "A Novel DNA Computing based Encryption and Decryption Algorithm," International Conference on Information and Communication Technologies, pp. 463-475.

[8] Ben-Aroya and E. Biham,1994, "Differential Cryptanalysis of Lucifer," Advances in Cryptology — CRYPTO '93 Proceedings, Spinger-Verlag.

[9] Atito A, Khalifa A, Rida SZ,2012, "DNA-based data encryption and hiding using Play fair and insertion techniques," Journal of Communications and computer Engineering, pp.44-49.

[10] Mona Sabry, Mohamed Hashem, Taymoor Nazmy, Mohamed Essam Khalifa, 2010, "A DNA and Amino Acids-Based Implementation of Playfair Cipher," International Journal of Computer Science and Information Security, Vol.8, No.3.

[11] Sherif T. Amin, Magdy Saeb and El-Gindi Salah, 2006, "A DNA-based implementation of YAEA encryption algorithm," International Conference on Computational Intelligence, pp. 120-125.

[12] Fatma E.Ibrahim, M.I.Moussa and H.M.Abdalkader, 2014, "A Symmetric Encryption Algorithm based on DNA Computing," International Journal of Computer Applications, Vol.97, No.16, pp.41-45.

[13] J.Chen, 2003, "A DNA –based Biomolecular Cryptography design," Proceedings of IEEE internal symposium on Circuits and Systems(ISCAS'03), Vol.3, pp.822-825.

[14] Pankaj Rakheja, 2011, "Integrating DNA Computing in International Data Encryption Algorithm(IDEA),", International Journal of Computer Applications, Vol.26, No.3, pp.1-6.

[15] Lai, Xuejia, and Massey, James L.,1991, "A Proposal for a New Block Encryption Standard, Advancesin Cryptology ," EUROCRYPT '90, Lecture Notes in Computer Science, Springer-Verlag, pp.389-404.
[16] https://en.wikipedia.org/wiki/Feistel_cipher.

[17] Janan Ateya Mahdi, 2009, "Design and Implementation of proposed B-R Encryption Algorithm," IJCCCSE, Vol.209, No.1.