SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

INSPIRED FEISTEL DNA BASED CRYPTOSYSTEM USING D-BOX AND IMAGE BASED KEY GENERATION

G. Ramaswamy*1 & Dr. R. Satya Prasad*2
*1Research Scholar, Acharya Nagarjuna University
*2 Professor in the Department of Computer Science and Engineering, Acharya Nagarjuna University

Abstract:
DNA cryptography is an exciting development in the growing area of DNA computing. Computing at the molecular level, such as that found in DNA, is known as bio-molecular computing. DNA is not just a data storage medium, but also a computational instrument. Researchers have placed a premium on DNA Computation because to its unique ability to store vast quantities of information. Traditional crypto algorithms have been shown to be flawed. Better information security may be achieved by integrating the fields of biology, chemistry, and computer science into a single, cohesive field of study. In this paper, we present a model that take image as a key and encrypts data by making use of DNA codons. Using a different key created by a key generation method in each cycle is a defining feature of the proposed technique, making it almost difficult to decrypt the picture using a cryptanalytic assault. It is also shown that the suggested method outperforms several preexisting algorithms when assessed by an avalanche effect.
Keywords: DNA Cryptography, Encryption, Decryption, DNA Codons

1. INTRODUCTION:

Innovations in technology improve levels of safety. In order to earn back customers' faith, several industries, such as banking and emailing, need to improve their security measures. The use of cryptography ensures the data's safety. Using cryptography, one may encrypt text. Encryption is the process of transforming data consisting of text into a message that has not been jumbled, whereas decryption performs the reverse function. The term "ciphertext" refers to a communication that has been decrypted. Quantum cryptography employs photons and can communicate at distances of up to 90 miles [1], whereas modern encryption has trouble with prime factorization]. After that, Raj and his colleagues [2] found the structure that was used to safeguard the data. Both DNA-based substitution and One-Time Pads were developed by him. Using biological components to encrypt data is what DNA cryptography does. It belongs to the realm of computers based on DNA. DNA computing involves the use of DNA sequences rather than traditional silicon processors. To solve NP-complete problems such as the Hamiltonian approach. He illustrated the problem with seven towns named after DNA. By synthesizing DNA, he was able to synthesis all possible pathways more quickly than by using the traditional approach. The name for this phenomenon is "huge parallelism." DNA parallelism was essential in researchers solving NP Hard problems. Using DNA as an example, Lipton instructed Kolte on how to solve SAT questions using the Boolean technique. The goal of cryptography in the

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160
field of biology is to develop algorithms that cannot be broken[4]. Modeling DNA cryptography systems is made harder as a result of the absence of a theoretical approach in these methods [6]. The present research provides DNA security that is based on the traditional encryption, which makes use of the D-box. The method of encryption changes depending on the required quantity of iterations. The key generator mechanism is used during each cycle to manufacture keys. The process is going to be covered in the next section. In Section 2, we will discuss the many aspects of DNA cryptography. Both Section 3 and Section 4 will discuss the essential generating methods. The experimental data, the performance analysis of the suggested approach, and avalanche effect comparisons with existing methodologies are discussed in Sections 5, respectively. The sixth section serves as the article's conclusion.

2. RELATED WORK:

Network security constantly seeks unbreakable cryptographic techniques to safeguard data in the cloud, on a network, or in other situations. Recent years have seen several initiatives. Raj et al. developed two approaches for DNA-based cryptography. One uses OTP, which cannot be broken, while the other uses steganography [2]. Sherif et al. encrypt each plaintext character with four molecules of DNA [11]. Hazra [13] presented a DNA-and carbon nanotube-based cryptosystem. Ponnavaiklu introduced a DNA encryption layer to IDEA. This approach increases the key size to prevent cryptanalysis [14]. Kaushik introduced DNA-Public KeyCryptography, which combines encryption with digital signature [15]. He developed an RSA key-generation algorithm. It works on text and photos. It turns the input message into ASCII code and then into a string. Then, the number is encrypted using the receiver's public key to get another series. Convert these digits to binary, then DNA to get the encrypted text. It employs RSA key generation cryptography .

Bony[6] created an amino acid playfair cipher. This model converts plaintext to binary. Binary data becomes DNA sequences. DNA sequences may be mapped into codons and, subsequently, amino acids. Codon to amino acid conversion is ambiguous as the 64 codons have only 20 alphabetical values. The author added all 64 English codons. The author avoided ambiguity by supplying the ambiguity number .

Tornea presented a DNA-based encryption method. DNA comprises just A, C, G, and T as nucleotides. The author also presented a key distribution mechanism that uses a secure channel and a secret code book. The sender sends a codebook-matching sequence to the recipient via the public channel. Convert plaintext to binary, then DNA. map DNA to RNA and protein. DNA to protein is a molecular biology dogma, but not the opposite. Here, the author detected redundancy and advocated additional protein-to-protein sequence rounds. Protein reverse translation reproduces two-thirds of RNA. The receiver will recover the mismatched letters after receiving the sequence and finding matches. The author constructed steganography utilizing DNA chemical bonding . One researcher encrypts using transposition. They employed fixed-size blocks, and the key matrix size should be the same. They transform plain text into ascii values. Simultaneously, a random key and DNA sequence are created. The transposed

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

block value and DNA sequence value form a new matrix. The matrix is then rotated row-by-column. Ascii to characters (ciphertext) [10]. Another researcher suggested encrypting the data using a pixel-based algorithm. First, picture pixels were jumbled. Then they utilize camouflaged graphics and watermarking to obscure the data [8].Nath[11] also developed an image encryption scheme based on RGB pixel displacement. This method extracts the picture's RGB values. Transform the vector to a matrix with the same RGB dimensions as the original picture.

3. KEY GENERATION:

G generates a private key from an image seed. Network D must differentiate between the generator's private key and transformation data. Source and transformation domains create keys. The source domain includes "seed" photographs. Transformation domain contains image's destination. The transformation domain reflects the private key's "style," such as a chaotic high-security key. Here's how DeepKeyGen loses keys.

$$L = L_G + L_D \qquad (1)$$

In the equation, L_G and L_D denote the generator and discriminator loss functions.

1) G-Network alters the picture. Generates a secure private key. G has three down sampled, six residual, two transposed, and one convolutional layer. Three times down sampling picture characteristics. Six identical blocks [37] make things. Then, two convolutional layers are transposed. Convolution makes pictures from low-level features. Outputs DeepKeyGen's private key. Instance normalization enhances picture quality, speeds model convergence, and avoids gradient explosion.

$$L_G = \min(\mathrm{E}_{x\sim pdata(x)} \log(1 - D(G(x))) \quad (2)$$

x represents the beginning image, The created key is near the transformation domain, so the discriminator believes it's from there.

2) Discriminator Network D: Checks whether picture is in transformation domain. Five difficult stages in Dis. Four convolution layers extract features. Last layer processes 1-D characteristics. This identifies fakes. Network D's accuracy is 50% when the networks balance. D can't distinguish between the produced key and transformation domain key.

3.1 Private Key Image

DeepKeyGen's private key is a stream cypher picture. Pixels make up each picture. These pixels include value and geographical information. The private key is thus a composite of picture pixels.

$$KEY_{definition} = [V_1, V_2, \ldots V_i, \ldots V_n] (3)$$

In the equation, Vi is one pixel.It's a key sequence value.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

$V_i = [p_i, x, y, c]$ (4)

pi is the pixel value, x is vertical, and y is horizontal. From 0 to 255, Pi, x, y, and c range. 4D private key, not stream cyphers. The key values (pixel values and 3-D space location information) make the private key challenging and increase its security. Networks learn source-to-transformation mapping from keys. Before training, convolutional parameters are random. I indicates the ith DeepKeyGen convolution layer. DeepKeyGen W is convolutional.

W=[W1,W2.. .Wn] (5)
W=x=G (6)

W indicates network settings and x is the initial image. Convolutional networks turn input images into feature vectors during training. Forward propagation produces the original private key, which is used to compare the current and target private keys in the transformation domain. Backpropagation transfers loss to convolutional layers. Back propagation using gradient descent improves performance

(Wjn, I = Wj1n, I J (Wjn)) (7)

J (Wjn) represents the gradient of the nth convolutional layer's Jth training loss. Gradient descent improves network mapping. G and D are distinct. After the training phase, the loss stabilizes and the transformation domain key is established. Alg. 1 depict the keygeneration procedure.

3.2 Algorithm: Key Generation Algorithm.

Initialization: Set the DeepKeyGen's W parameters to a random value using the formula Wn = random [wn,1, wn,2,..., wn,i]. KEY is 256× 256× 3.

Step 1: Convert training images to a 256× 256× 3 matrix. x = Convert(IMAGE source domain); y = Convert (IMAGE transformation domain).
Step2: Forward propagation of generator network G, key G (x).
Step 3: G's deepest layer produces KEY. D(y) = Result / Forward propagation of discriminator network D.
Step 4: D(y) = Result / Forward propagation of D. The transformation domain judgment should be printed.
Step 5. LG + LD = L / Determine the overall loss. It's backwards propagation, or the number It's backwards propagation, or the number and it's happening in reverse.
Step 6: Wjn,i = Wj1n,i J(Wjn) / It is necessary to compute the gradient that will be sent back to each layer before updating the network settings.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

Step 7: The output, KEY, is quite near to the transformation domain, so we'll just stop here and call it a day.

4. PROPOSED ALGORITHM

DNA cryptography and Feistel cipher improve computer security. Add the DNA D-box table to the Feistel Cipher to compute. The D-box swaps codons 43 times. Three DNA bases dictate the location of an amino acid in a protein molecule during protein synthesis. 20 amino acids (A, B, C) are made from 61 codons, and 3 are stop signals. Phenylalanine (F) may be substituted with TTT or TTC, producing confusion in letter order. In this work, amino acid names aren't supplied and 64-codons (Figure 1) are constructed randomly (8 x 8 grids). 64 combinations of 3 DNA bases from 4 create codons. Block 64-bit calculation If bits are less than 64, zeroes are added.

$E_K(M) = C$ indicates encryption, whereas $D_K(C) = M$ shows decoding. Plaintext (M) is any character; encrypted text $C \in \{A,C,G,T\}^*$. Sender and recipient must agree on the number of rounds and arbitrary keys. Random key $K \in \{0, 1\}^*$

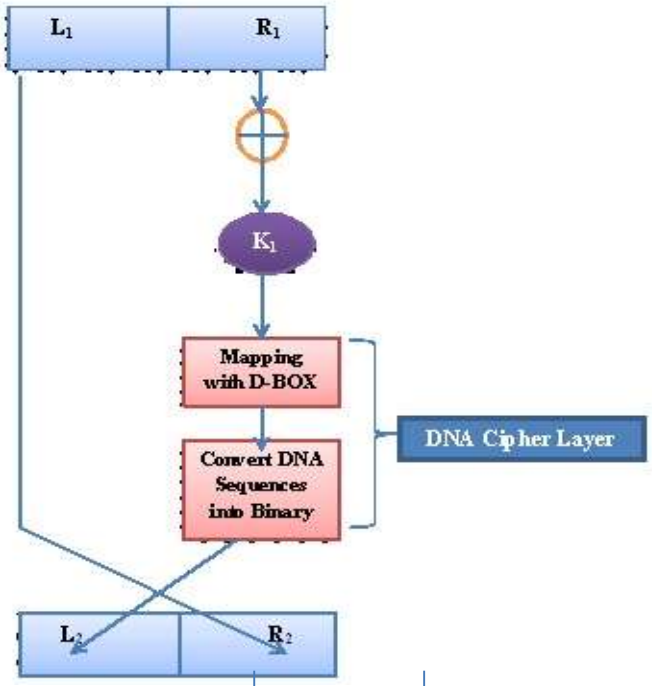


Figure 1: Inspired Feistel DNA Cryptosystem using D-Box

Bob needs to hear from Alice. They agree on the number of rounds, and Alice splits the plaintext into 32-bit pairs. Every round needs a unique key. XOR key and content (Ri). Add 4 zeros to get octal. This sets the D-Box codon. DNA is binary (A-00, C-01, G-10, T-11). Remove the remaining four cushioned pieces and exchange them. After n cycles, they offer Bob the binary sequence as DNA. Bob gets the cipher and keys via an open channel. Bob reverses and gets plaintext.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

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 2: D-Box of Inspired Feistel DNA Cryptosystem

4.1 Algorithm for Encryption

Algorithm Encryption(M,num)

M is the plaintext and num is number of rounds to be performed

BEGIN

1. Transform the Message M of into ASCII and in turn into binary values.
2. Divide the message into two equal halves names as L_1 and R_1
3. For j=1 to num

Begin

1. Randomly generate the key of length 32-bit and store it in K_j.
2. $R_j=R_j \oplus K_j$
3. $R_j=R_j+$"0000"
4. Rj has 36 3-bit bits. Convert to octal.
5. Consider pairwise octal values and map them into D-box then the results is 6 codon values.
6. Transform the DNA sequence into its equivalent binary.
7. Delete the last four padded characters and store it in R_i.
8. Swap L_j, R_j.

End

4. $C=L_j||R_j$.
5. C is ciphertext (00-A, 11-T, 10-G, 01-C)..

END

4.2 Algorithm for Decryption

Algorithm Decryption(C, num)

C is cipher text and num is number of rounds. K is an array of set of keys for number of rounds received from the sender through secure medium.

BEGIN

1. Transform the cipher text C into its equivalent binary form according DNA digital coding.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

2. Divide C into 32-bit Lnum and Rnum..
3. For j=num to 1

Begin

i) Lj=Lj+"1111"
ii) Convert binary to DNA bases (00-A, 11-T, 10-G, 01-C).
iii) Codonize DNA sequences.
iv) Retrieve D-Box codon locations.
v) Convert octal locations to binary
vi) Remove "0000" from LJ.
vii) Lj=Lj□Kj
viii) Swap Lj&Rj.

End

4. $M=L_j||R_j$
5. Convert the binary data to ASCII M.

END

4.3 Empirical Analysis

4.3.1Process of Encryption

Let us take the plaintext as M=Grooming and number rounds n=4. Convert the plaintext into ASCII and in turn into binary then,

Character	ASCII	Binary
G	71	01000111
r	114	01110010
o	111	01101111
o	111	01101111
m	109	01101101
i	105	01101001
n	110	01101110
g	103	01100111

First Round

a) Perform XOR operation between right part of previously obtained result with randomly generated Key value.

R_1 =01101101011010010110111001100111
K_1 =10010000001101011100011011011000
$R_1 \oplus K_1$ =11111101010111001010100010111111

b) Divide the binary into octal values. The result of XOR contains only 32 bits when we split them 3 bits each and make a pair we need to pad four more bits to the result.

111,111,010,101,110,010,101,000,101,111,110,000
7,7,2,5,6,2,5,0,5,7,6,0

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

c) Retrieve the codon value which placed in the D-box.
GGG,ACC,GCT,AAT,AGG,GTT

d) Convert them into equivalent binary values according to DNA Digital Coding.
10101000010110011100001100101010111 1

e) Remove final four bits and swap L_1 and R_1
10101000010110011100001100101010010001110111001001101111011011111

Second Round

a) Perform XOR operation between right part of previously obtained result with randomly generated Key value.

R_2 =01000111011100100110111101101111
K_2 =01011010110010110010010100101011
$R_2 \oplus K_2$ =00011101101110010100101001000100

b) Divide the binary into octal values. The result of XOR contains only 32 bits when we split them 3 bits each and make a pair we need to pad four more bits to the result.

000,111,011,011,100,101,001,010,010,001,000,000
0,7,3,3,4,5,1,2,2,1,0,0

c) Retrieve the codon value which placed in the D-box.
TCG,CAG,CCC,TGT,ATC,TTT

d) Convert them into equivalent binary values according to DNA Digital Coding.
11011001001001010111101100110111111

f) Remove final four bits and swap L_2 and R_2
11011001001001010111101100110111101010000101100111000011001010101

Third Round

a) Perform XOR operation between right part of previously obtained result with randomly generated Key value.

R_3 =10101000010110011100001100101010
K_3 =01001011100110100011001001011101
$R_3 \oplus K_3$ =11100011110000111111000101110111

a) Divide the binary into octal values. The result of XOR contains only 32 bits when we split them 3 bits each and make a pair we need to pad four more bits to the result.

111,000,111,100,001,111,110,001,011,101,110,000
7,0,7,4,1,7,6,1,3,5,6,0

b) Retrieve the codon value which placed in the D-box.
GAT,GAA,TGG,GTC,CGC,GTT

c) Convert them into equivalent binary values according to DNA Digital Coding.
10001110000011101010110101100110111

g) Remove final four bits and swap L_3 and R_3
10001110000011101010110101100110110100100100101011110110011011 1

Fourth Round

a) Perform XOR operation between right part of previously obtained result with randomly generated Key value.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

R_4 =11011001001001010111101100110111

K_4 =01101001010100010010101001010101

$R_4 \oplus K_4$ =10110000011101000101000101100010

a) Divide the binary into octal values. The result of XOR contains only 32 bits when we split them 3 bits each and make a pair we need to pad four more bits to the result.

101,100,000,111,010,001,010,001,011,000,100,000

5,4,0,7,2,1,2,1,3,0,4,0

b) Retrieve the codon value which placed in the D-box.

AAA,TCG,ATC,ATC,CAT,CTT

c) Convert them into equivalent binary values according to DNA Digital Coding.

00000011011000110100110101001101111

d) Remove the last four bits and interchange L_4 and R_4 then the string is

000000110110001101001101010011011000111000001110101011010110011 0

After 4 rounds, the binary data is converted into DNA form which is a cipher texti.e.,

00,00,00,11,01,10,00,11,01,00,11,01,01,00,11,01,10,00,11,10,00,00,11,10,10,10,11,01,01,10, 01,10

AAATCGATCATCCATCGATGAATGGGTCCGCG

4.3.2 Process of Decryption

The received Cipher text from the Sender is

C= AAATCGATCATCCATCGATGAATGGGTCCGCG

1. Transform the ciphertext C into its equivalent binary form as per the DNA Digital Coding.

000000110110001101001101010011011000111000001110101011010110011 0

First Round

a) Divide the ciphertext into two parts L_4 and R_4.

L_4=00000011011000110100110101001101

R_4=10001110000011101010110101100110

b) L_4=00000011011000110100110101001101 L_4=AAATCGATCATCCATC

Split them into codons and find their positions in the D-Box and pad four bits to make them into codons.

L_4=AAA,TCG,ATC,ATC,CAT,CTT,

i.e. In D-box, the position of AAA is 5th row and 4th column and that of TCG is 0th row and 7th column. This process is repeated for remaining codons.

L4=5,4,0,7,2,1,2,1,3,0,4,0

Convert octal to binary and eliminate padding.

L_4=10110000011101000101000101100010 0000

After removing the padded bits,

L_4=10110000011101000101000101100010

c) Perform XOR operation between L_4 with K_4

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

L_4 = 10110000011101000101000101100010

K_4 = 01101001010100010010101001010101

$L_4 \oplus K_4$ = 11011001001001010111101100110111

d) Interchange L_4& R_4 and convert into DNA form

L_4=11011001001001010111101100110111

R_4=10001110000011101010110101100

C=1000111000001110101011010110011011011001001001010111101100110111

Second Round

a) Divide the Ciphertext received from the previous round into two equal parts names as L_3 and R_3.

L_3=10001110000011101010110101100110

R_3=11011001001001010111101100110111

b) Take the left part of the data

L_3= 10001110000011101010110101100110

Convert them into DNA and then split them into codons and pad four bits to make even number of codons.

GAT, GAA, TGG, GTC, CGC, GTT

Find the position of codons from the D-Box.

L_3=7,0,7,4,1,7,,6,1,3,5,6,0

c) Convert the octal values into binary and remove the padded four bits.

L_3=111000111100001111110001011101110000

After removing padded bits

L_3=11100011110000111111000101110111

d) Perform XOR operation between L_3 with K_3

L_3 =11100011110000111111000101110111

K_3 =01001011100110100011001001011101

$L_3 \oplus K_3$ =10101000010110011100001100101010

e) Interchange L_3& R_3 and convert into DNA form

L_3=10101000010110011100001100101010

R_3=11011001001001010111101100110111

C=1101100100100101011110110011011110101000010110011100001100101010

Third Round

a) Divide the ciphertext of the previous round into two equal parts i.e., L_2 and R_2

L_2=11011001001001010111101100110111

R_2=10101000010110011100001100101010

b) Take the left part

L_2=11011001001001010111101100110111

Convert them into DNA, split them into codons, and pad four bits to make them even codons

L_2=TCG,CAG,CCC,TGT,ATC,TTT

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

Find the positions of each codon from the D-Box

L_2=0,7,3,3,4,5,1,2,2,1,0,0

c) Convert the octal values into binary and remove the padded bits.

L_2=000111011011100101001010010001000000

After removing padded bits

L_2=00011101101110010100101001000100

d) Perform XOR operation between L2 with K2.

L_2 =00011101101110010100101001000100

K_2 =01011010110010110010010100101011

$L_2+\oplus K_2$ =01000111011100100110111101101111

e) Interchange L_2& R_2 and convert them into DNA

L_2=01000111011100100110111101101111

R_2=10101000010110011100001100101010

C=1010100001011001110000110010101001000111011100100110111101101111

Fourth Round

a) Divide ciphertext into two equal parts i.e., L_1 and R_1.

L_1=10101000010110011100001100101010

R_1=01000111011100100110111101101111

b) Take the left part and convert into DNA form

L_1=10101000010110011100001100101010

L_1=GGGACCGCTAATAGGG

Split them into Codons and pad four bits into binary form to make them even codons

L_1=GGG,ACC,GCT,AAT,AGG,GTT

Find the positions codons from the D-Box

L_1=7,7,2,5,6,2,5,0,5,7,6,0

c) Convert the octal values into binary form and remove the padded bits.

L_1=111111010101110010101000101111110000

After removing the padded bits,

L_1=11111101010111001010100010111111

d) Perform XOR operation with L_1& K_1.

L_1 =11111101010111001010100010111111

K_1 =10010000001101011100011011011000

$L_1\oplus K_1$ =01101101011010010110111001100111

e) Interchange L_1 and $R_1$and convert them into DNA.

L_1=01101101011010010110111001100111

R_1=01000111011100100110111101101111

C=0100011101110010011011110110111101101010110100101101110011001111

C=CACTCTAGCGTTCGTTCGTCCGGCCGTGCGCT

At final round, the cipher text convert into binary and then into equivalent ASCII character

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

C=CACTCTAGCGTTCGTTCGTCCGGCCGTGCGCT

C=01000111011100100110111101101111011011010110100101101110011001111

Binary Data	ASCII Value	ASCII Character
01000111	73	G
01110010	114	r
01101111	111	o
01101111	111	o
01101101	109	m
01101001	105	i
01101001	110	n
01100111	103	g

5. RESULTS

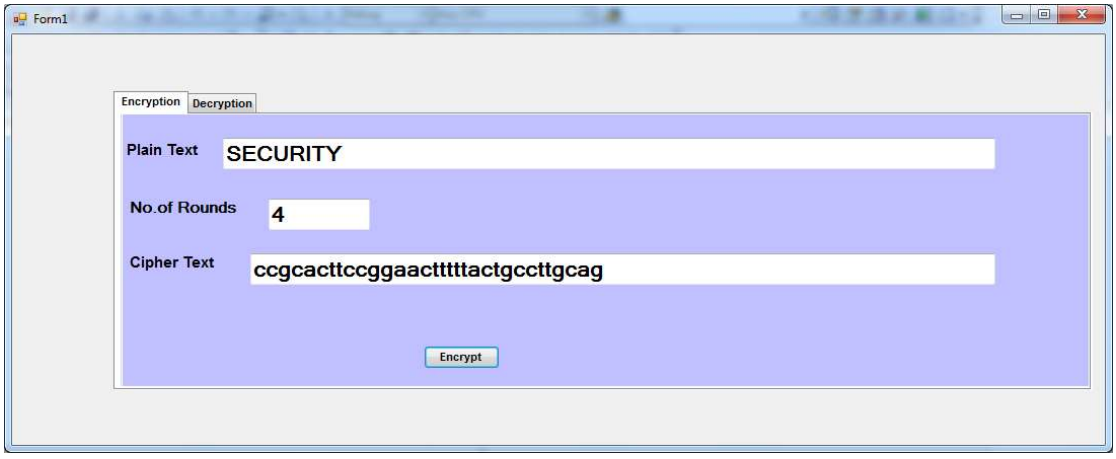
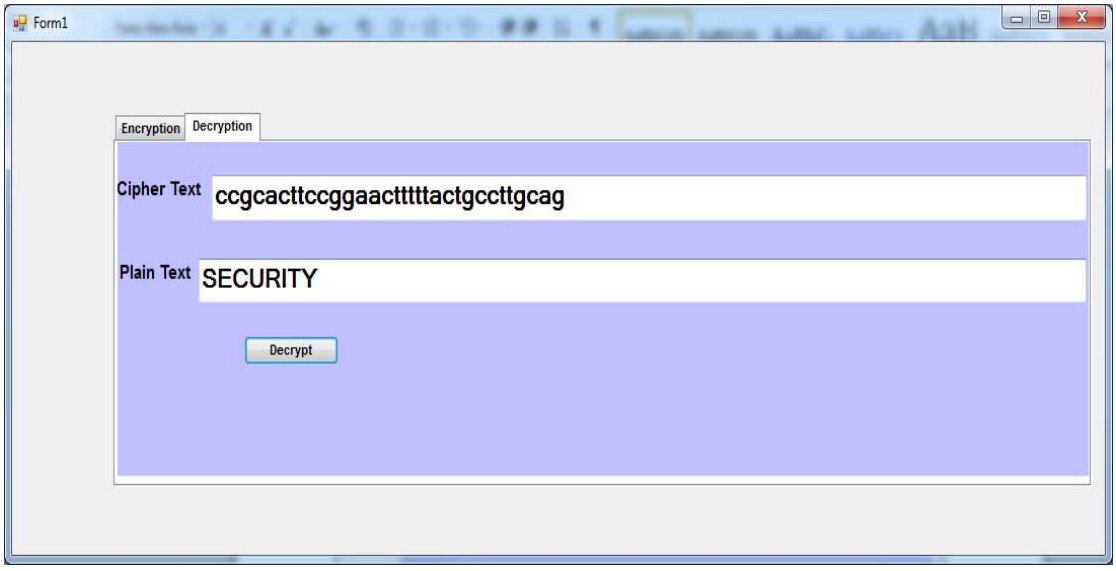


Figure 3: Encryption Process for the Plaintext "SECURITY" for 4 rounds.



SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

Figure 4: Decryption process for the cipher text "ccgcacttccggaacttttttactgccttgcag".

5.1 PERFORMANCE MEASUREMENT

The analysis of an algorithm measured in Intel i3 processor in .NET environment for the string "SECURITY" with 4 number of rounds. The following tables showed the difference in the time taken for encryption and decryption process between the proposed model and the existing traditional feistel approach model .It is observed that, though the proposed model takes more time because of DNA cipher layer, still it may be preferred because of more security.

Table 1 : Comparison table of Encryption time between Proposed and Traditional approach

Number of Rounds	Encryption Time for Proposed Algorithm	Encryption Time for Traditional Approach
4	0.011	0.00423
5	0.012	0.00456
6	0.013	0.00487
7	0.013	0.00512
8	0.0145	0.00543
9	0.0154	0.00654
10	0.018	0.00543

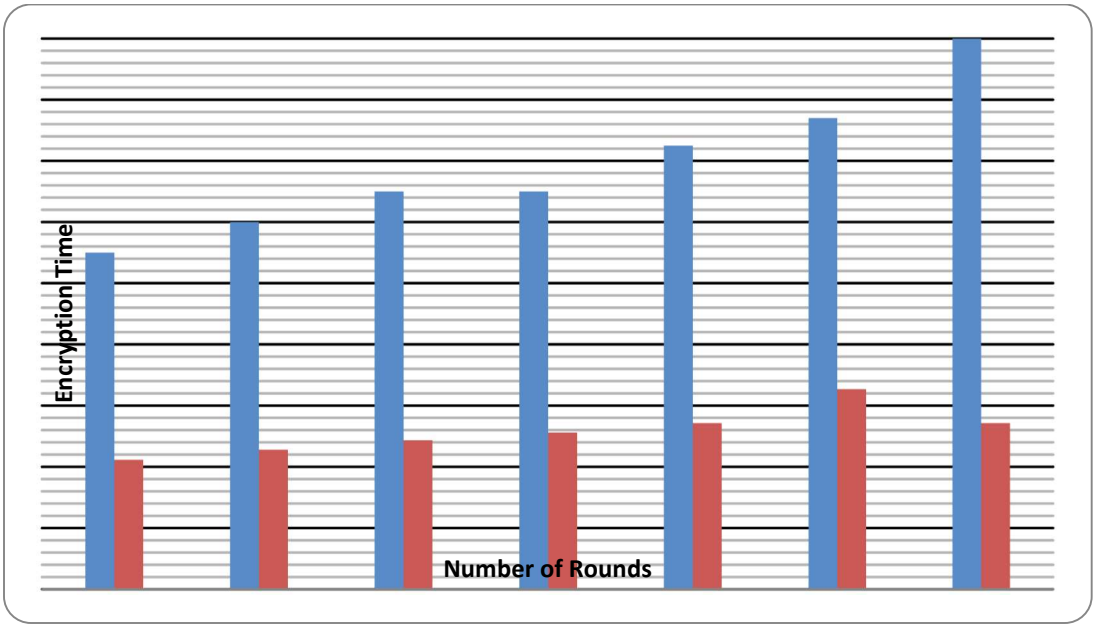


Figure 5: Performance analysis for the message SECURITY in relation between number of rounds and encryption time.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

Table 2: Comparison of Decryption time between Proposed and Traditional Approach

No.of Rounds	Decryption Time for Proposed Algorithm	Decryption Time for Traditional Approach
4	0.0032	0.0018
5	0.0048	0.0017
6	0.0064	0.0023
7	0.0089	0.0045
8	0.0065	0.0035
9	0.011	0.0034
10	0.0123	0.0032

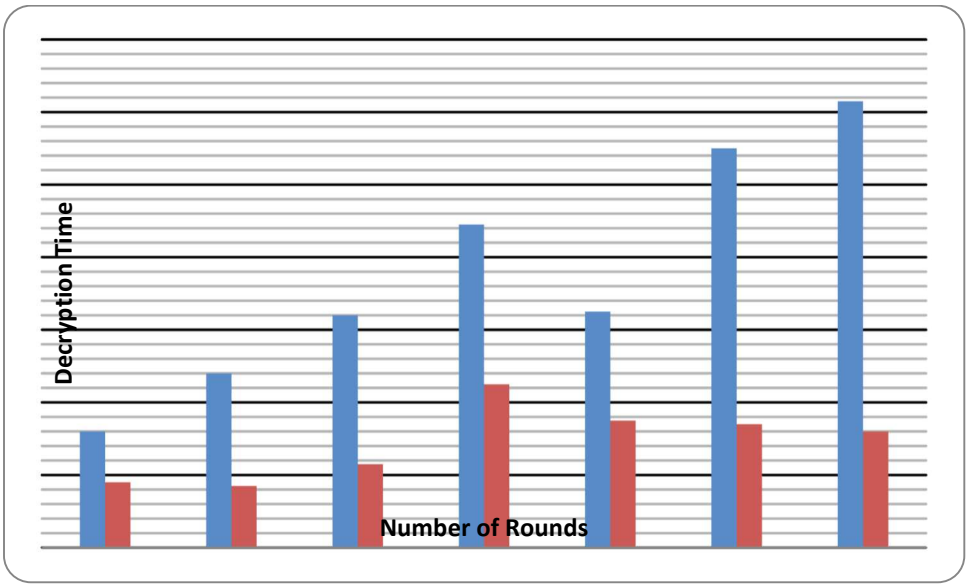


Figure 6: Performance analysis for the message SECURITY in relation with number of rounds and time.

5.2 AVALANCHE EFFECT

The following table 5.3 represents the percentage of bits flipped in the cipher text when a single character in the plaintext is modified.

Plaintext1= security

Plaintext2=secxrity

Total 34 bits are flipped in the cipher text when change of single character in the plaintext are shown in red colour.

Ciphertext1=

01110011011001010110001101111000011100100110100101110100011111001

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

Ciphertext2=
00111100110100110001000110010000101101100011000111011010000000010

Table 3:Comparison of Avalanche Effect for strings security and secxrity to the proposed and traditional models

Name of the Algorithm	Number of bits flipped	%
Ceaser Cipher	2	4.132
Playfair	3	4.76
Vigener Cipher	2	4.124
Blowfish	19	25.78
DES	32	49.65
Feistel Inspired DNA based Cryptosystem	**34**	**54.13**

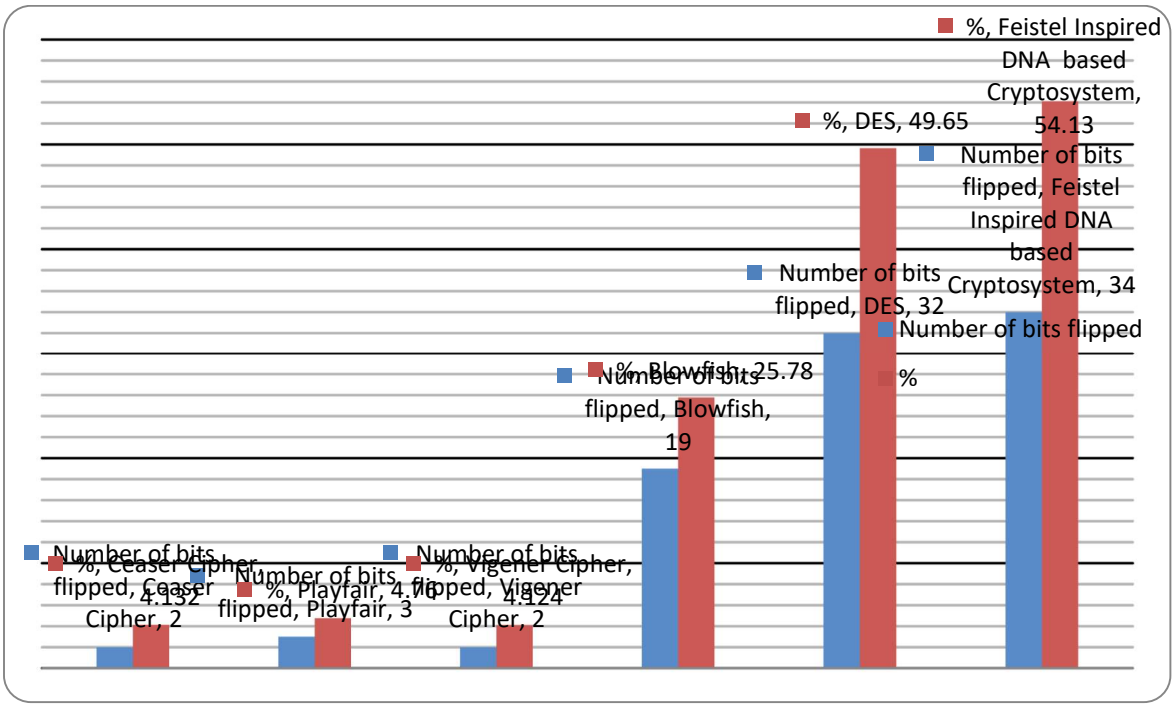


Figure 7 :Comparison of the results of the avalanche effect for strings security and secxrity.

In the considered plaintext “security”, the change of one character leads to change in 34 bits in the ciphertext which was very difficult to the intruder in identifying the plaintext. It was observed that the total number of bits flipped in the present model is more when compared to existing models.

SEYBOLD Report
ISSN: 1533 - 9211
DOI: 10.5281/zenodo.7135160

6. CONCLUSION & FUTURE SCOPE

In this, DNA codons are organized in a D-box for a symmetric blockcipher. 64-bit plaintext is split in two, the randomly generated key is XORed with the right segment, and the D-codon box values are obtained. DNA's binary values encode the text. Encryption text is separated at the receiver. The left section becomes DNA bases and codons. The D-Box codons are converted to octal and XORed with the key value. In each cycle of 64-bit plaintext ciphering, most bits change. Simple text can't be hacked. Compares two models' performance. Avalanche effect is explored. Even though DNA is cited as a medium for ultra-minimal data storage and the future seems bright, there are still worries like quantum attacks and bio molecular labs. Future work may include building a public key cryptosystem using PCR Amplification. Increasing LOOKUP table size for additional character sets improves performance. More compression methods will be used on DNA sequences to improve compression ratio, and combining encryption and compression leads to a stronger cryptosystem. Future assaults will be practiced. More models will be added by increasing Avalanche effect.

References:

[1] A. S. Abad, H. Hamidi, “An Architecture for Security and Protection of Big Data”, in International Journal of Engineering (IJE), TRANSACTIONS A: Basics Vol. 30, No. 10, (October 2017), pp. 1479-1486

[2] B. B. Raj, J. Frank, T.Mahalakshmi, “Secure Data Transfer through DNA Cryptography using Symmetric Algorithm”, in International Journal of Computer Applications, Vol 133-No 2, pp. 0975-8887, January 2016

[3] A. Roy, A. Nath, “DNA Encryption Algorithms: Scope and Challenges in Symmetric Key Cryptography”, in International Journal of Innovative Research in Advanced Engineering, ISSN: 2349-2763, Issue 11, Volume 3, Nov, 2016.

[4] N. S. Kolte, K. V. Kulhalli and S. C Shinde, “DNA Cryptography using Index-based Symmetric DNA Encryption Algorithm”, International Journal Of Engineering Research and Technology, ISSN 0974-3154 Vol 10, No1 ,2017.

[5] S. Karthiga, E. Murugavalli, “DNA Cryptography”, in International Research Journal of Engineering and Technology, p-ISSN 2395-0072, Vol 5, March 2018.

[6]. Bonny B.Raj, V. Ceronmani sharmimila,” An Survey on DNA Based Cryptography” IEEE 2018 International Conference on Emerging Trends and Innovations In Engineering And Technological Research (ICETIETR) - Ernakulam (2018.7.11-2018.7.13)] 2018.

[7]. Saijisha K S,S.Mathew,” An encryption based on DNA cryptography and steganography”IEEE 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)COIMBATORE, India (2017.4.20-2017.4.22)] 2017 .

[8]. S.Roy, Sudipta Singha, Shahriyar, Shaikh Akib, Asaf-Uddowla, Md, Alam, Kazi Md. Rokibul; Morimoto, Yasuhiko”A novel encryption model for text messages using delayed chaotic neural network and DNA cryptography[IEEE 2017 20th International Conference of Computer and Information Technology (ICCIT) - Dhaka, Bangladesh(2017.12.22-2017.12.24)].

DOI: 10.5281/zenodo.7135160
[9]. K.KALAISELVI “Enhanced AES Cryptosystem by using Genetic Algorithm and Neural Network in S-box 978-1-5090-1936-6/16/$31.00 ©2016 IEEE. 5. Panagiotis Papadimitratos”. Secure Data Communication in Mobile Ad Hoc Networks”, IEEE journal on selected areas in communications 0733-8716.
[10]. Md .Rafiul Biswas,Kazi Md.Rokibul Alam, Ali Akber,Ya Suhiko Mori-moto”A DNA cryptographic technique based on dynamic DNA encoding and asymmetric cryptosystem” Published in 2017 4th International Conference on networking system and security(NSysS)IEEE.
[11]A. Nath, A. Dodia, “Symmetric Key Encryption Algorithm using DNA Sequence,” Int. J. Adv. Res.Comput. Sci. Manag. Stud., vol. 6, no. 4, pp. 108–115, 2018.
[12] K. C. Jithin, S. Sankar, “Colour image encryption algorithm combining Arnold map, DNA sequence operation, and a Mandelbrot set,” J. Inf. Secur. Appl., vol. 50, no. 102428, pp. 1–22, Feb. 2020.
[13] A. Hazra, C. Lenka, A. Jha, M. Younus, “A Novel. Two Layer Encryption Algorithm Using One-Time Pad and DNA Cryptography,” in Lecture Notes in Networks and Systems, vol. 103, 2020, pp. 297–309.
[14] P. B. Narasingapuram , M. Ponnavaikko, “DNA Cryptography Based User Level Security forCloud Computing and Applications,” Int. J. Recent Technol. Eng., vol. 8, no. 5, pp. 3738–3745, 2020.
[15] A. Kaushik, V. Thada, “VG1 Cipher – A DNA Indexing Cipher,” Int. J. Innov. Technol. Explor.Eng., vol. 9, no. 3, pp. 221–226, 2020.

[16] Rajni, “DNA Computing,” Int. J. Eng. Comput. Sci., vol. 6, no. 1, pp. 19972–19976, 2017.

[17] O. Tornea, M. E. Borda, V. Pileczki,“Cryptographic Algorithm Based on Dna and RNA Properties,” Int. J. Adv. Res. Comput. Eng. Technol., vol. 7, no. 11, pp. 237–241, 2018.