International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

Implementation of DNA Cryptography using IBROS Cypher

[1]Abhishek Pandey, [2]Sourav Chanda

[1,2]Department of Computer Science & Engineering, Swami Vivekananda Institute of Science & Technology, Kolkata, India

***Abstract* - This paper presents a new DNA cryptographic system based on IBROS Cypher. IBROS mainly deals with manipulation in the binary form of the plain text message to encrypt and decrypt it. IBROS uses a completely randomized two key system, which makes it hard to break. Even after such complex methods, the applicability of each method is simple which makes it a fast DNA cryptography system. IBROS is an abbreviation for the five steps used in this cypher. They are – ICFM (Index Comparison Flip Mutation), BFM (Bit Flip Mutation), RFS (Rail Fence Straightening), OMB (One-to-Many Breeding), and Swap Mutation. All of these five steps are inspired from different fields of computing and cryptography, yet they are completely unique as cryptographic method. ICFM is inspired by the working principles of a Turing Machine, whereas BFM, OMB, and Swap Mutation are inspired by Genetic Algorithms, the method named RFS is a modified form of the traditional Rail Fence Cypher. IBROS works by taking the binary form of the plain text message, then passing it through each step serially, and finally encoded in to DNA bases using one of the eight rules of DNA encoding. Some of the presented methods in IBROS can be used not only for binary data but also for normal text data. This new technique will prove to be promising in the field of DNA computing.**

Keywords: ICFM (Index Comparison Flip Mutation), BFM (Bit Flip Mutation), RFS (Rail Fence Straightening), OMB (One-to-Many Breeding), Swap Mutation.

I. INTRODUCTION

The ever-growing number of internet users follows the growth of data as well. That data might be sensitive, which can cause serious damage if somehow breached. To avoid such threats, that data must be protected by such means, that no one other than the designated user can access it. Even after so many security paradigms and protocols, preventing a data breach is still near impossible. So, the most established method is to use different data hiding techniques, so that even after an infringement that data will remain in confidence. Such an establishment can be achieved using cryptography. In absence of any such enterprise an attacker can easily violate the data in many ways. The attacks can be classified into two types – Active attack and Passive attack. In an active attack the victim is acknowledged about the attack. Whereas, in a passive attack, the victim has no knowledge of being attacked. In either of the cases the receiver is the only victim. Active attacks obstruct the accessibility of the data, whereas, passive attacks harm the confidentiality and integrity of the data. Out of four types of attacks, i.e. interruption, interception, modification and fabrication; interception is considered as an active attack and rest are considered as passive attacks. In interruption data is obstructed from reaching the receiver. In interception the attacker snoops the data while it is being transferred to the receiver. In case of modification, the intruder modifies the data before it reaches the designated receiver. Whereas, in fabrication the attacker sends a forged message to the receiver acting as the intended sender, compromising the integrity of the data. Through cryptography interruption can be completely obliterated and to some extent modification can also be dealt with. These attacks cause harm to the confidentiality, integrity and accessibility of the data.

Cryptography is a systematic approach to encoding and decoding a message which ensures the confidentiality of the message between the sender and the receiver by preventing any interpretations by a third party. Cryptography mainly consists of two parts, namely encryption and decryption. Encryption is the process of encoding the plain text into cipher text using a key and decryption follows the exact reverse process and converts the cipher text into plain text using same or a different key. In earlier days' cryptography was achieved by the means of some manual techniques. With the advent of computer systems, such results can be achieved in less time with more security[1]. Cryptography not only deals with text messages but also treats images, videos, audios etc. The method of hiding image, video and audio data is called Steganography [2]. Modern cryptography systems are divided into two categories namely, symmetric key cryptography and asymmetric key cryptography [1, 3]. Some of the well-known symmetric key algorithms are DES, Blowfish, and AES etc. The concept of having only one key for both encryption and decryption makes them quite vulnerable. So out of essential need asymmetric key algorithms like, RSA, DSA, ECC etc. were developed. In asymmetric key cryptography, the basic idea is to encrypt the data with a public key which can only be decrypted with its corresponding private key.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

In 1994, Leonard M. Adleman's pioneering work suggested a scheme of the utilization of the biological structure of the DNA to solve a Directed Hamiltonian Path problem [4]. In extension to Adleman's work Richard J. Lipton investigated solutions of Satisfiability of Propositional Formula setting up new possibilities for DNA computing [5]. If we just consider the human race it's unthinkable for us to even begin to comprehend how much data, our DNA has in store. It has in store everything from genesis through conception, and up hitherto. DNA is capable of hoarding large amounts of data. One gram of DNA can withhold about 108 TB of data, which certainly exceeds the capacity of any of the traditional storage devices [6].

In essence, DNA cryptography is the process of hiding data by the means of DNA strands, using alphabets of oligonucleotides [6, 7]. A very basic form of DNA cryptography can be described as –

DNA:

DNA stands for deoxyribonucleic acid. Every living organism on this planet consists of DNA. It carries all the hereditary information from parent to children, which result decides all the traits that a living organism will poses. DNA stores this information using four nitrogenous bases, namely Adenine (A), Thymine (T), Cytosine (C), and Guanine (G)[8]. DNA consists of two polynucleotide chains [9], twisted around each other in such a way that they form a double helical configuration [9, 10]. This model is also noted as the "Watson-Crick model".

As can be seen two strands are connected by chemical bonds between nitrogen bases; adenine with thymine and cytosine with guanine. These bonding units are called base pairs. These base pairs are further attached to a sugar molecule and a phosphate molecule. Altogether, they are called nucleotides. The bases are of two classes, purines (adenine and guanine) and pyrimidines (thymine and cytosine) [9]. A DNA sequence can be of any length containing "ATCG". Those sequences can inherit permanent changes through different kinds of mutations. There are primarily two kinds of mutations – transitions and transversions. Transitions deal with pyrimidine-to-pyrimidine (e.g., T to C) and purine-to-purine (e.g., A to G) substitutions. Whereas Transversions deal with pyrimidine-to-purine (e.g., T to G or A) and purine-to-pyrimidine (e.g., A to C or T) substitutions [11].

II. LITERATURE STUDY

Several research papers have reconnoitered DNA cryptographic techniques. This section summarizes some of them.

R. Terec et al.[3]presented a method to increase security by the generation of an asymmetric key inside a DNA security algorithm, where an asymmetric key generation algorithm starting from a password is designed. To implement that the BioJava API has been utilized.

J. Chen [6] proposed a method to use a carbon nanotube-based probe to recast data between DNA and conventional binary storage media. Then a one-time-pad based DNA cryptographic algorithm is presented. The proposed algorithm assembles a large one-time-pad in the form of DNA, which is assembled randomly from oligonucleotides. The DNA strands are to be isolated and cloned and the specific one-time-pad is supposed to be shared in advance between the sender and the receiver of the enciphered message.

A. Hazra et al.[7] used a fusion of symmetric key techniques to achieve the finalized cipher text. The proposed method utilizes XOR operation and substitution methods. The algorithm works by converting the plain text message to its binary form. Then the binary form of a randomly selected DNA sequence, which is acting as a key, is XOR-ed with the binary form of the plain text. The key is then complemented and passed through a complex substitution scheme.

In [12] an asymmetric key cryptography scheme is presented using DNA synthesis, PCR amplification, and DNA digital coding. In the proposed scheme, the sender and receiver both designs and exchanges a 20-mer oligonucleotides long DNA sequence, which acts as a forward primer for PCR amplification. The receiver is presumed to have a pair of public and private keys. The encryption key is generated using the previously generated pairs of PCR primers and the receiver's public key, whereas the decryption key is generated using the pair of PCR primers and the receiver's private key. The plain text message is first converted to its equivalent hexadecimal form, and that is converted to its corresponding binary form. Then the previously generated encryption key is used to encrypt the message.

M. R. Abbasy et al.[13] proposed a method to hide the text message within a DNA reference sequence. Both then sender and receiver decide on a common DNA reference sequence. Then at first, the text message is converted to binary sequence and then to DNA sequence as per base pairing rules. To increase the complexity complementary rules are then applied. Then by extracting the index of each couple of nucleotides in the DNA reference sequence, the final secret message is obtained. Because of the identical DNA reference sequences, the receiver then extracts the exact same positions of those indexes and then after applying the complementary rule he obtains the DNA encoded message. The DNA encoded

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

message can then be converted back to their original binary pairs and then back to the plain text.

In[14] another method is proposed whose security is mainly based on the DNA hybridization and one-time-pad scheme in which the encrypted message is sent in small packets. Here, sender, after generating a one-time-pad, which is a randomly generated single-stranded DNA (ssDNA) string, which is acting as the key, sends it to the receiver. Here the length of that ssDNA is decided by product of the length (no. of bits) of the binary form of the plain text message, and the length (in mer) of the oligonucleotides s to encrypt one plaintext message bit. The plain text is first converted to its equivalent ASCII form and then to the corresponding binary form. The encryption algorithm works by, reading the binary from of the message from left to right, where for every "1" it selects a 10-mer oligonucleotide sequence from the end of the ssDNA sequence, which is then passed through Watson-Crick complementary rule. Then a sequential integer number (packet number) is generated and attached to the obtained DNA sequence and then sent to the receiver as a small packet. The receiver, with the help of that packet number obtains the starting position of the packet content. By applying the complementary process, the 10-mer oligonucleotide sequence is obtained. Then the position of this oligonucleotide is hybridized into the one-time-pad key. The matching position of that sequence is replaced by a "1" and all other unmatched positions are replaced by a "0" which furnishes the binary form of the message from which the plain text message is then obtained.

A. Atito et al.[15] proposed an encryption and hiding scheme using Play fair Cipher and Insertion Techniques. To begin the plain text message is converted to its equivalent binary form and then mapped to DNA sequences. That DNA form is then transferred into Amino acids form according to a standard universal table of Amino acids and their codons representation in the form of DNA. Then the English alphabets of the Amino acid sequence are passed through the traditional Playfair sequence, using a secret key. Then those encrypted form of the Amino acids are transferred back to a DNA sequence. That DNA sequence is then passed through the insertion process to hide the data. Substrings of random lengths from of the encrypted DNA sequence and a DNA reference sequence are interleafed respectively. Those random lengths are generated beforehand for both the sequences using a random number seed for each. Receiver, already in possession of those seeds and the secret key can generate the random numbers and the Playfair matrix again, using which they can reverse the insertion process and get back the encrypted DNA sequence, then decrypt the Amino acid form of that DNA sequence, to get the original Amino acid sequence. That Amino acid sequence can then be converted to its equivalent DNA sequence using the same universal table and then back to the plain text message.

Dr. R. Surrenderinget al. [16] discusses the significance of the Internet of Things (IoT) in various applications due to its ability to establish effective communication between interconnected devices. The potential threat of unauthorized access to sensitive information during storage raises concerns about data confidentiality, integrity, and privacy. To address these concerns the paper introduces a novel approach named DNA-based Elliptic Curve Cryptography technique with RedFox Optimization algorithm for clustering (RF-DECC). This approach is aimed at enhancing data security in fog computing. The process involves identifying cluster heads using the RedFox optimization technique. Once clustering is completed, the cluster head initiates the data encryption process by applying DNA-Elliptic Curve Cryptography (ECC) to encrypt the data of cluster members. ECC is highlighted as a lightweight public key cryptography algorithm, and the article underscores that using DNA in conjunction with ECC adds complexity to the encryption process.

In [17] a system is proposed that demonstrate its effectiveness in withstanding attacks. The technique offers the ability to retrieve the original data (audio, video, image, or text files) without any alterations during the decryption process. The paper addresses the limitations of conventional cryptographic methods in securing online activities and introduces DNA cryptography as an innovative approach. By combining the Trellis algorithm with DNA sequences and random key generation, the technique aims to enhance security and resist attacks, ensuring that original data can be securely retrieved after decryption.

III. PROPOSED METHOD

The proposed method consists of five distinct steps of encryption. Which must also be followed in a reverse fashion for decryption. Including key generation and the formation of dynamic DNA encoding, makes it a seven-phase process incompleteness. This method mainly deals with the manipulation of the binary form of the plain text message. Consisting of two keys and five steps used for the encryption process, the process is quite hard to break with common cryptanalysis practices. We call the sender as Alice and the intended receiver as Bob. All the phases of encryption and decryption are described below.

Phase1 – Key generation:

The key generation process begins with the sender setting a limit within which the first key will be generated. As mentioned earlier there will be two keys, let, K_P and K_R. K_P will be a set of Pythagorean-Triplets randomly chosen

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

between 1000 and the limit set by the sender, and K_R is a number randomly chosen between 0 to 7. The lower limit of the key K_P is set to be 1000 to make it long enough to be secured. The range of 0 to 7 is chosen for the key K_R is chosen because all the characters of the plain text message will be converted to 8-bit binary sequence, meaning each character will have an index ranging from 0 to 7. This index numbers will be used for the manipulation of that binary data.

Generation of K_P and K_R:

There are many possible ways to generate a set of Pythagorean-Triplets, but the easiest one found so far is by using the Euclid's Formula for the proof of Infinite Pythagorean-Triples. In his proof, Euclid states that if there are "n" numbers, then there can be "n" Pythagorean-Triples.

Euclid's Proof:

Consider the identity, $n^2 + 2n + 1 = (n + 1)^2$ …… (1)

Whenever 2n + 1 is a perfect square, the identity (1) forms a Pythagorean Triplet. But 2n + 1 comprise all the odd numbers; every other square number is odd; there are an infinite number of odd squares; hence there are an infinite number of Pythagorean-Triples.

Same can be said for the following identities –

$n^2 + 4n + 4 = (n + 2)^2$ where, 4n + 4 is a square.

$n^2 + 6n + 9 = (n + 3)^2$ where, 6n + 9 is a square.

$n^2 + 8n + 16 = (n + 4)^2$ where, 8n + 16 is a square.

And so on…

Considering the above theorem, we generate Pythagorean-Triples between a particular range, then select one random triplet from that list. To generate K_P Alice selects a range, say "N". The algorithm will generate all the Pythagorean-Triples between 1000 to N and then select a random triple amongst them. To decide that random selection point R_P, we use the following method:

A random floating point number F_P is generated from 0.1 to 0.9 and then rounded off to 1 decimal place. Then F_P is multiplied to N and the ceiling value of F_P is taken. Then a random number between 0 and F_P is generated which results into R_P. Consider a function ρ(x, y), where x and y are the lower and upper limits for generating a random integer, the mathematical deduction for generating R_P can be described as –

$R_P = \rho (0, \lceil N \times \rho (0.1, 0.9) \rceil)$

Out of all the generated triples, R_P^{th} triple will be select as K_P.

To generate K_R, the following expression is considered –

$K_R = \rho (0, 7)$

Phase2 – Implementing Proposed Algorithms:

The encryption process consists of five different small algorithms; they are –

1. Index Comparison Flip Mutation (ICFM).
2. Bit Flip Mutation (BFM).
3. Rail Fence Straightening (RFS).
4. One to Many Breeding (OMB).
5. Swap Mutation.

The workings of each of the above-mentioned algorithms are described below.

Index Comparison Flip Mutation (ICFM):

The idea of ICFM is mainly inspired by the workings of a Turing Machine. The basic working principle of the Turing Machine in [18] is explained as, there is a "tape", which can be as long as it needs to be. The tape is divided into small sections called "squares". Each of those squares can bear a "symbol". At any moment if there is just one square bearing a particular symbol, we may call it "scanned square", and that symbol on that particular square can be termed as "scanned symbol". The scanned symbol is the only one, of which, the machine "directly aware", and it can also remember the symbols which it had "seen" (scanned) previously. "Configuration" determines the behavior of the machine. In some configurations if the scanned square bears no symbol, the machine writes down a new symbol, in any other configuration it may erase the scanned symbol. The machine may also change the scanned square, by shifting it one place to right or left.

ICFM takes a tape X as an input and returns the tape $\overline{X}$ as an output after encryption and does the exact opposite during decryption. In ICFM also there is a tape, which consists of 8squares, containing 0s and 1s. To be precise each tape holds the 8 – bit representation of a character. There is a *"head"*, which scans the current square's value and the index of the squares in both X and $\overline{X}$, the current position of the head on the tape X can be considered as *"current square"* denoted by X_i (where, $i \epsilon \{0\ to\ 7\}$), and the rear part of that head is connected to the current square in tape denoted by $\overline{X}_j$ (where, $j \epsilon \{0\ to\ 7\}$). The head is also capable of performing the *"flip"* operation on the current square of either tapes,

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

which primarily denotes a NOT operation of the value of the current square. The head also keeps track of all the scanned squares, so that they are not scanned again. The basic purpose of the *"head"* is to take input from tape X and put the output in tape $\overline{X}$. The initial position of the head is at the "R^{th}" index where, $R = K_R$ and the outputs generated by the head are placed in $\overline{X}$ sequentially, starting from 0 to 7, which is empty at the initial stage. The head when pointing at the current square, checks for 4 different conditions.

Condition 1: If the *i and j is even* or *i and j is odd* and the value held in X_i is 0 then, put $\overline{X}_j = X_i$, and move the head to the left on X and do j = j + 1.

Condition 2: If the both *i and j is not even* or both *i and j is not odd* and the value held in X_i is 0 then, put $\overline{X}_j$ = flip (X_i), and move the head to the left on X and do j = j + 1.

Condition 3: If the *i and j is even* or *i and j is odd* and the value held in X_i is 1 then, put $\overline{X}_j = X_i$, and move the head to the right on X and do j = j + 1.

Condition 4: If the both *i and j is not even* or both *i and j is not odd* and the value held in X_i is 1 then, put $\overline{X}_j$ = flip (X_i), and move the head to the right on X and do j = j + 1.

Basically, the indexes of the tapes are compared and if both of them are of same kind (i.e. even or odd), then the value scanned from X is put as it is on $\overline{X}$, and the movement of the head (i.e. left or right), depends on the value held in X.

Let us consider an example, where X holds a binary sequence 10100101, of which each bit is equally distributed between 8 squares on the tape. Let us assume R = 4.

(a)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0							
i	0	1	2	3	4	5	6	7

The head is initially placed at X_4, *Condition 1* matches, the head moves to the left.

(b)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0						
i	0	1	2	3	4	5	6	7

The head is now placed at X_3, *Condition 1* is satisfied, the head moves to the left.

(c)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0	1					
i	0	1	2	3	4	5	6	7

The head is now at X_2, *Condition 3* is satisfied, the head moves to the right.

(d)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0	1	1				
i	0	1	2	3	4	5	6	7

The squares X_3 and S_4 is already scanned, so the head goes to X_5. *Condition 1* is satisfied; head goes to the right.

(e)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0	1	1	0			
i	0	1	2	3	4	5	6	7

The head is now at X_6, *Condition 1* is satisfied, the head moves to the left.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

(f)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0	1	1	0	0		
i	0	1	2	3	4	5	6	7

X_2, X_3, X_4, X_5 are already visited, so the head is now at X_1, *Condition 1* is satisfied, head moves to the left.

(g)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0	1	1	0	0	1	
i	0	1	2	3	4	5	6	7

The head is now placed at X_i, *Condition 3* is satisfied, the head moves to the right.

(h)

X	1	0	1	0	0	1	0	1
i	0	1	2	3	4	5	6	7

$\overline{X}$	0	0	1	1	0	0	1	1
i	0	1	2	3	4	5	6	7

The head takes the input from $\overline{X}_7$ and places the input in X_7, as all the squares in X are filled up we can assume that we have successfully reverted back to X.

Bit Flip Mutation (BFM):

BFM is inspired by the Non-Uniform Mutation technique in Genetic Algorithm [19]. In Non-Uniform mutation the bit value at a particular position is flipped. IN BFM the bit value at position K_R of an 8 – bit binary sequence is flipped.

Let, $K_R = 4$

Before BFM:	1	0	0	1	0	1	0	1
After BFM:	1	0	0	1	1	1	0	1

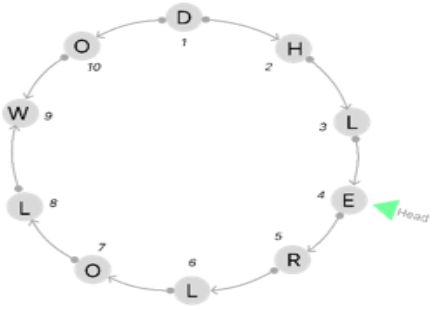
To go back to the previous state, the bit value at the position K_R needs to be flipped again.

Rail Fence Straightening (RFS):

RFS is a modified version of the traditional Rail Fence cypher. In Rail fence Cypher the characters of the plain text message is arranged in a zig-zag pattern so that the arranged character would look like Railway fence.

In RFS, the first half of the message is placed on the top vertices of the fence starting from left to right and the second half of the message is placed in a reverse order, starting from right-to-left. To understand the working of RFS (Rail Fence Straightening) algorithm, let us take an example of a string that is placed like a triangular waveform, andthe message is divided into two halves. The characters of the first half of the message are placed on the peaks of the triangular wave staring from Left-to-Right and the second half of the message is placed on the valleys of the triangular wave in a reverse order from Right-to-Left as illustrated.

Now, if we pull the two ends of the string and straighten it, the characters of the message would be transposed.



Now, let us assume that every character of the transposed message is a node in a circular linked list where the head of

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

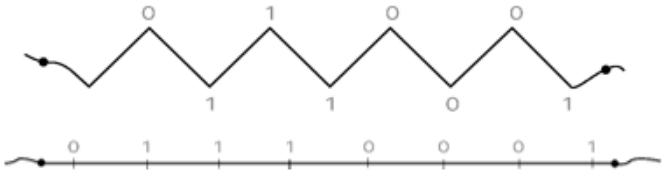
Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

the list is at K_R^{th} position. The list is traversed starting from the head until all the nodes are visited.

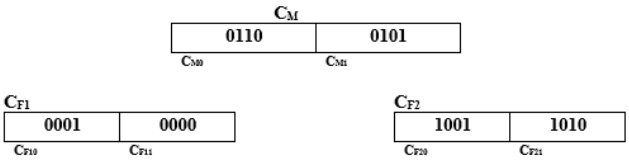
To go back to the previous state, we first have to find the position where the head of our circular linked list should be placed. For that we can just deduct K_R from the length of the message. After traversing through the whole linked list starting from the head. We just have to put the string back in the same triangular wave form and first read the characters on the peaks of the waveform, and then read the characters at the valleys in a reverse order.

We are implementing RFS to accomplish transposition of an 8-bit binary sequence as illustrated.



One to Many Breeding (OMB):

One-to-Many Breeding is inspired by the single point heuristic crossover technique in Genetic Algorithms. In a single point crossover, there are two chromosomes and a crossover point for both of them is selected randomly. Then, the genetic manipulation process called mutation is carried out, where the bits at the randomly selected position of the chromosomes are altered. After the alteration, new off springs are produced. The illustration of the same can be seen below.

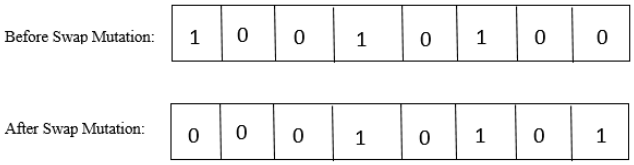


In OMB, we consider one male chromosome C_M, and many female chromosomes C_{Fi} where, $i \epsilon \{1\ to\ n\}$. The male chromosome is used as a key which is a subset of previously generated key K_P. K_P consists of three values that is, a^2, b^2, and $c^2$of the Pythagorean-Triplet. C_M holds the 8-bit binary representation of c. In case, any particular value of c cannot be represented using 8-bits, i.e., more than 8-bits are required, last 8-bits of the binary representation of c will be used. C_{Fi} holds the 8-bit binary form of the characters of the message. C_M is divided into two halves namely C_{M0} and $C_{M1.}$ All of the C_{Fi} are also divided into two halves each named as C_{Fi0} and C_{Fi1}. For each C_{Fi} MSB and LSB of C_{Fi0} are XOR-ed. If the result is 0, the C_{Fi1} will be XOR-ed with C_{M0}. And, if the result is 1, C_{F1} will be XOR-ed with $C_{M1.}$ After the XOR operation is performed, the results are the off springs C_{Oi} of breeding simulated between C_M and C_{Fi}. Same is illustrated below.

For reverting the binary sequences back to their original self, the same should be done using C_{Oi} where, $i \epsilon \{1\ to\ n\}$, and C_M.

Swap Mutation:

Swap Mutation works by swapping the MSB and LSB of an 8-bit binary sequence. To revert back, same swap operation should be done again.



Phase 3 – Mapping to DNA bases:

In this phase the final binary sequence will be mapped to its corresponding DNA sequence. Here, *00, 01, 10, 11* are encoded by four bases, *A, C, G, T*. In any DNA encoding scheme,[3, 6, 7, 11, 12, 13, 14]A is taken as the complement of T and G as the complement of C, and vice versa. So can be seen in Table 2. This mapping can produce up to 4! i.e. 24 rules to map binary pairs to DNA bases. But only 8 of them satisfy the Watson-Crick complement rule [10].

DNA	Complement
C = 00	G = 11
T = 01	A = 10
A = 10	T = 01
G = 11	C = 00

Rules	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8
00	A	A	G	G	T	T	C	C
01	C	G	A	T	C	G	A	T
10	G	C	T	A	G	C	T	A
11	T	T	C	C	A	A	G	G

Here we will use the K_R^{th} rule to map binary pairs to DNA bases. Assuming K_R = 4, the mapping of the binary sequence 1001110001010011, according to table 3 will be as follows –

10011100 01010011→ ATCGTTGC

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 7, Issue 8, pp 72-82, August-2023
https://doi.org/10.47001/IRJIET/2023.708010

IV. RESULTS AND DISCUSSIONS

Plain text message: Hello
$\mathbf{K_P}$ = [*a = 1740, b = 59, c = 1741*]
$\mathbf{K_R}$ = 6

Encryption:

Encryption of the plain text to cypher text:

Step 1: The plain text message is converted to its equivalent ASCII form and then to its corresponding 8-bit binary form.

72, 101, 108, 108, 111 → 01001000, 01100101, 01101100, 01101100, 01101111

Step 2: ICFM will be applied to each of the 8-bit binary sequences using the key Kr. After applying ICFM, the sequences will look like – 00101101, 01011001, 01101001, 01101001, 11001001

Step 3: BFM will be applied to each of the above binary sequences using the key Kr. After applying BFM, the sequences will look like – 00101111, 01011011, 01101011, 01101011, 11001011

Step 4: RFS will be applied to each of the above binary sequences using the key Kr. After applying RFS, the sequences will look like – 01010111, 11011100, 01011110, 01011110, 01111100

Step 5: OMB will be applied to each of the above binary sequences using the key Kp. After applying OMB, the sequences will look like – 01011010, 11011000, 01010011, 01010011, 01110001

Step 6: Swap Mutation will be applied to each of the above binary sequences using the key Kr. After applying Swap Mutation, the sequences will look like – 01011010, 01011001, 11010010, 11010010, 11110000

Step 7: Mapping to DNA bases is done, using 6th rule, which results into the following – AATT, AATA, GACT, GACT, GGCC

Step 8: All of the above DNA sequences are concatenated together to form the final cypher text.

Cypher Text: AATTAATAGACTGACTGGCC

Encryption of the plain key to cypher key:

The values of *a*, *andb* from the generated triplet K_P will be sent to the user along with K_R.

Step 1: Zeros will be put between every pair of digits in *a*, and after every digit in *b*, then they will be added to form S_1.
a = 1070400 and *b* = 5090
S_1 = 1075490

Step 2: Now S_1 will be divided into two halves to form X and Y.
X = 107
Y = 5490

Step 3: Generate the closest value, to the maximum of X and Y, in the power of 2 table (let us call it bitRep). Below formula can be used to generate a bitRep.

$$\text{bitRep} = 2^{\lceil \log_2 \max(X, Y) \rceil}$$

Therefore, bitRep = $2^{\lceil \log_2 5490 \rceil} = 2^{13} = 8192$

Step 4: Now, X and Y will be used to calculate Z using the following expression –

Z = (8192 * X) + Y, where 8192 is the minimum bit value required to represent both X and Y.

Z = (8192 * 107) + 5490 = 882034

Step 5: K_R will be concatenated to Z.

8820346

Step 6: The digits of the above number are then converted to their equivalent 4-bit binary form, and then mapped to DNA bases using Rule 1 from Table 3.

8, 8, 2, 0, 3, 4, 6 →
1000, 1000, 0010, 0000, 0011, 0100, 0110

1000, 1000, 0010, 0000, 0011, 0100, 0110 →
GA, GA, AG, AA, AT, CA, CG

Step 7: Then these DNA encoded digits will be concatenated to form the encrypted key.
Cypher Key: GAGAAGAAATCACG

Step 8: So that the bitRep can be retrieved during decryption, it will be stored in the file-name of the encrypted key file. E.g. ENCK-013, so 13 will be put as an exponent to 2 to retrieve bitRep.

Decryption:

Decryption of the cypher key to plain key:

Key = GAGAAGAAATCACG

Step 1: They encrypted key will be dissociated in pairs of two. The converted to its corresponding binary form using Rule 1

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 7, Issue 8, pp 72-82, August-2023
https://doi.org/10.47001/IRJIET/2023.708010

from Table 3.Then each of those binary sequences will be converted to its equivalent decimal number, those numbers will be concatenated to form a larger number.

GA, GA, AG, AA, AT, CA, CG →
1000, 1000, 0010, 0000, 0011, 0100, 0110

1000, 1000, 0010, 0000, 0011, 0100, 0110 →
8, 8, 2, 0, 3, 4, 6

8820346

Step 2: Now, the K_R will be extracted from the above numbers i.e., the last digit. And the remaining is termed as Z.

$K_R = 6$

Z = 882034

Step 3: The value of bitRep will be retrieved using the numbers in the key file name. Key file name is – ENCK-013.

So, bitRep = 2^{13} = 8192

Step 4: Now if we perform normal division between Z and bitRep, we will have X and modular division will result in Y.

X = □882034÷ 8192□ = □107.33□ = 107

Y = 882034% 8192 = 5490

Step 5: Now after concatenating X and Y, if we read the alternate digits, they will result into *b*, and the digits we skip were the ones that of *a*.

1 **0** 7 **5** 4 **9** 0

a = 1740
b = 59

Step 6: Now that we have *a* and *b* we can calculate *c*.

$$c = \sqrt{a^2 + b^2}$$

□ $c = \sqrt{(1740)^2 + (59)^2} = 1741$

Decryption of the cypher text to plain text:

Cypher Text: AATTAATAGACTGACTGGCC

Step 1: Dissociate the DNA sequence in set of four. Then convert them to their respective binary forms using the K_R^{th} rule in Table 3.

AATT, AATA, GACT, GACT, GGCC →
01011010, 01011001, 11010010, 11010010, 11110000

Step 2: Apply Swap Mutation on the above binary sequences.

01011010, 11011000, 01010011, 01010011, 01110001

Step 3: Apply OMB on the above binary sequences.

01010111, 11011100, 01011110, 01011110, 01111100

Step 4: Perform RFS decryption method on the above binary sequences.

00101111, 01011011, 01101011, 01101011, 11001011

Step 5: Perform BFM on the above binary sequences.

00101101, 01011001, 01101001, 01101001, 11001001

Step 6: Perform ICFM decryption method on the above binary sequences.

01001000, 01100101, 01101100, 01101100, 01101111

Step 7: Convert the above binary sequences to their equivalent decimal numbers, which gives us the ASCII encoded message. Then by converting them back to the characters using universal ASCII table will result in the plain text message.

01001000, 01100101, 01101100, 01101100, 01101111 →
72, 101, 108, 108, 111

72, 101, 108, 108, 111 →
H, e, l, l, o

Plain Text Message: Hello

V. CONCLUSION

The main aim of the proposed DNA cryptographic system is to allow, secured information exchange, and to develop a faster system for encryption and decryption process. With the growth in the number of internet users every day, more and more data are going online, if that data is sensitive then implementation of a fast yet secured system is a necessity. The algorithms such as ICFM, RFS, and OMB are not known to anyone so far because, they are completely new. That makes this algorithm complex enough. To add further, the steps of BFM and Swap Mutation are added to provide extra complexity. In *IBROS* cypher, keys are also encrypted which makes it easier for the sender. They can send the message and the keys together through any unsecured channel. As *IBROS* is a two-key system, which makes it harder for attackers to guess two keys in less time. Other than that, even if the odds of them guessing the random number is 1 out of 8, but guessing the randomly chose Pythagorean-Triplet will be a great deal to accomplish. Even though the process is susceptive to brute force, it will take a lot of time. And a message not decrypted in time is of no use to any attacker.

ISSN (online): 2581-3048
Volume 7, Issue 8, pp 72-82, August-2023
https://doi.org/10.47001/IRJIET/2023.708010

Also, unlike other algorithms out there, we are not using only one rule to encode binary pairs to DNA bases, so at the very first stage attackers have 8 set of cypher text messages to brute force through. This method is not a conventional Cryptographic method, so it's far stronger and not easily crackable. The use of DNA is a huge success due to low power dissipation, large storage and faster transmission of data. We have dry run our work, and have found it successful so far.

REFERENCES

[1] A. Kahate, "Cryptography: Concepts and Techniques," in CRYPTOGRAPHY AND NETWORK SECURITY, Second ed., NEW DELHI, Tata McGraw-Hill Publishing Company Limited, 2008, pp. 38-83..

[2] D. Tulpan, C. Regoui, G. Durand, L. Belliveau, S. Leger, "HyDEn: A Hybrid Steganocryptographic Approach for Data Encryption Using Randomized Error-Correcting DNA Codes," BioMed Research International, vol. 2013, pp. 1-11, 29 June 2013..

[3] R. Terec, M. Vaida, L. Alboaie, L. Chiorean, "DNA Security using Symmetric and Asymmetric Cryptography," International Journal on New Computer Architectures and Their Applications (IJNCAA), pp. 34-51..

[4] L. M. Adleman, "Molecular Computation of Solutions to Combinatorial Problems," Science, vol. 266, no. 5187, pp. 1021-1024, 1994..

[5] R. J. Lipton, "Using DNA to Solve NP-Complete Problems," Science, vol. 268(4), pp. 542-545, 1995..

[6] J. Chen, "A DNA-based, Biomolecular Cryptography Design," in International Symposium on Circuits and Systems, 2003..

[7] A. Hazra, S. Ghosh, S. Jash, "A New DNA Cryptography Based Algorithm Involving the Fusion of Symmetric-Key Techniques," in Advanced Computational and Communication Paradigms, Advances in Intelligent Systems and Computing, 2018..

[8] P. Roy, D. Dey, D. De, Swati Sinha, "DNA Cryptography," in Handbook of Research on Natural Computing for Optimization Problems, vol. II, IGI Global, pp. 813-840..

[9] "The Structure of DNA," in Molecular Biology of the Gene, 7 ed., Pearson, pp. 78-81..

[10] James D. Watson, Francis H. C. Crick, "MOLECULAR STRUCTURE OF NUCLEIC ACIDS," NATURE, 1953..

[11] "The Mutability and Repair of DNA," in Molecular Biology of the Gene, 7 ed., Pearson, pp. 313-314..

[12] G. Cui, L. Qin, Y. Wang, X. Zhang, "An Encryption Scheme Using DNA Technology," IEEE, pp. 37-42, 2008..

[13] M. R. Abbasy, P. Nikfard, A. Ordi, M. R. N. Torkaman, "DNA Base Data Hiding Algorithm," International Journal on New Computer Architectures and Their Applications, vol. 2, no. 1, pp. 183-192, 2012..

[14] S. Pramanik, S. K. Setua, "DNA Cryptography," in International Conference on Electrical and Computer Engineering, Dhaka, Bangladesh, 2012..

[15] A. Atito, A. Khalifa, S. Z. Rida, "DNA-Based Data Encryption and Hiding Using Playfair and Insertion Techniques," Journal of Communication and Computer Engineering, vol. 2, no. 3, pp. 44-49, 2012..

[16] P. K. R. Dr. R.Surendiran, "A Fog Computing Approach for Securing IoT Devices Datausing DNA-ECC Cryptography," *DS Journal of Digital Science and Technology*, vol. 1, no. 1, pp. 1, 2, 2022.

[17] E. B. K. Rama Devi, "An Enhancement in Data Security Using Trellis Algorithmwith DNA Sequences in Symmetric DNA Cryptography," *Wireless Personal Communications*, p. 1, 2022.

[18] A. M. Turing, "ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM", Journal of Communication and Computer Engineering, vol. 2, no. 3, pp. 230-265, 1936..

[19] Dr. M. Panda, Dr. M. R. Patra, "Genetic Algorithms," in Soft Computing: Concepts and Techniques, Second ed., NEW DELHI, University Science Press, 2013, pp. 82-92..

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 8, pp 72-82, August-2023

https://doi.org/10.47001/IRJIET/2023.708010

Citation of this Article:

Abhishek Pandey, Sourav Chanda, "Implementation of DNA Cryptography using IBROS Cypher" Published in *International Research Journal of Innovations in Engineering and Technology - IRJIET*, Volume 7, Issue 8, pp 72-82, August 2023. Article DOI https://doi.org/10.47001/IRJIET/2023.708010

****

